



VERİTABANI YÖNETİM SİSTEMLERİ

ENDÜSTRİ MÜHENDİSLİĞİ LİSANS TAMAMLAMA PROGRAMI

PROF. DR. İBRAHİM ÇİL

İSTANBUL ÜNİVERSİTESİ AÇIK VE UZAKTAN EĞİTİM FAKÜLTESİ

İSTANBUL ÜNİVERSİTESİ AÇIK VE UZAKTAN EĞİTİM FAKÜLTESİ
ENDÜSTRİ MÜHENDİSLİĞİ LİSANS TAMAMLAMA PROGRAMI



**VERİTABANI YÖNETİMİ
SİSTEMLERİ**

PROF. DR. İBRAHİM ÇİL

İSTANBUL ÜNİVERSİTESİ AÇIK VE UZAKTAN EĞİTİM FAKÜLTESİ

Yazar Notu

Elinizdeki bu eser, İstanbul Üniversitesi Açık ve Uzaktan Eğitim Fakültesi’nde okutulmak için hazırllanmış **bir ders notu niteliğindedir.**

İÇİNDEKİLER

İÇİNDEKİLER.....	I
ÖNSÖZ	VIII
KISALTMALAR.....	IX
1. VERİTABANI YÖNETİM SİSTEMLERİNE GİRİŞ	1
1.1 Veritabanı Temel Kavramları	8
1.2 Veri Yönetimi Yaklaşımları.....	9
1.2.1 Geleneksel Dosya Sistemleri Yaklaşım	9
1.2.2 Veritabanı Yaklaşımı	10
1.3 Veritabanı Yönetim Sistemleri.....	11
1.3.1 Veritabanı Yönetim Sistemlerinin Sınıflandırılması.....	14
1.3.2 Veri Tabanı Sistemlerinin Kısa Tarihçesi	15
1.4 Veri Saklama İle İlgili Terimler ve Veri Hiyerarşisi	17
1.5 Veri Saklama Modelleri	18
1.5.1 OLTP (Online Transaction Processing) Sistemler (Çevrimiçi Hareket İşleme (ÇİHİ) Sistemleri	18
1.5.2 OLAP (Online Analytical Processing) OLAP- Eş Zamanlı Analistik Veri İşleme	19
2. VERİTABANI TASARIMI	31
2.1 Veritabanı Tasarımında Yapılması Gereken Hususlar.....	37
2.2 Veritabanı Geliştirme Aşamaları.....	37
2.2.1 İhtiyaç Analizi.....	38
2.2.2 Veritabanı Tasarımı.....	39
2.3 Veritabanının Tamamlanması.....	43
2.4 Sistemin Test Edilmesi ve Bakımı	43
2.5 Veri Tabanının İşletilmesi ve Yönetilmesi	43
2.5.1 Veritabanı Yöneticisi	44

2.5.2 Veritabanı Güvenliği.....	44
2.5.3 Veri Güncelleme	45
2.5.4 Veritabanı Yedekleme.....	45
2.5.5 Veritabanı Kurtarma.....	45
3. VERİ MODELİ	4
3.1. Veri Modellerinin Sınıflandırılması.....	11
3.2. Veri Modelleri: Kısa Tarihçe	12
3.3. Veri Modeline Göre Veri Tabanı Yapıları	12
3.3.1. Hiyerarşik Veritabanları.....	12
3.3.2. Ağ Veritabanları.....	13
3.3.3. İlişkisel Veritabanı Sistemi	13
3.3.4. Nesneye Yönelik Veritabanları	14
3.4 Varlık İlişki Diyagramları	16
3.4.1. Varlık-İlişki Diyagramlarının Bileşenleri	17
3.5 Varlık-İlişki Şemasının Tablolara Dönüşürtlmesi	23
3.5.1 Bire-Bir İlişkilerin Dönüşürtlmesi	23
3.5.2 Bire-Çok (1-n) İlişkilerin Tablolara Dönüşürtlmesi	25
3.5.3 Coğa-Çok (N-M) İlişkilerin Tablolara Dönüşürtlmesi	27
3.6 Çok Değerli Özelliklerin Dönüşürtlmesi	28
4. İLİŞKİSEL VERİ TABANI	37
4.1 İlişkisel Veri Tabanın Genel Yapısı.....	43
4.2 Kısıtlamalar.....	44
4.3 İlişkisel Veri Tabanının Temel Öğeleri.....	47
4.3.1 Tablolar	47
4.3.2 Sütunlar	47
4.3.3 Değerler.....	47

4.3.4 Anahtarlar.....	47
4.3.5 Şemalar.....	47
4.3.6 İlişkiler	48
5. NORMALİZASYON.....	56
5.1 Veri Tabanı Tasarımında Yapılması Gereken Hususlar	63
5.2 İlişkisel Veritabanları ve Normalleştirme	64
5.3 Normalleştirmenin Amaçları.....	66
5.3.1 Veri Bütünlüğünü Sağlamak	66
5.3.2 Uygulamadan Bağımsızlık.....	66
5.3.3 Performansı Arttırmak	67
5.4 Veritabanı Normalizasyonu Kuralları	67
5.5 Normalizasyon Kurallarının Uygulama Örneği	69
5.6 İşlevsel Bağımlılıklar	75
5.6.1 Tam İşlevsel Bağımlılık ve Kısmi Bağımlılık (Partial Dependency)	76
5.6.2 Dolaylı Bağımlılık (Transitive Dependency).....	76
6. SQL- YAPISAL SORGU DİLİ	87
6.1 SQL ile Gerçekleştirilecek İşlemler	94
6.1.1 Veri Tabanı Oluşturma.....	94
6.1.2 Tablo Oluşturma.....	94
6.1.3 Tabloya Veri Girişi	95
6.2 Tabloda Sorgulamalar Yapma.....	95
6.2.1 Temel SQL Sorgulamaları Select Komutu	95
6.2.2 Tablo Bilgilerinin Sıralanmış Olarak Listelenmesi.....	97
6.2.3 Birden Çok Alana Göre Sıralama	98
6.2.4 Tekrarlı Satırların Ortadan Kaldırılması-SELECT DISTINCT	99
6.2.5 Koşula Bağlı Olarak Listeleme	100

6.2.6 Çeşitli Veri Tipleri İçin Basit Sorgulamalar	101
6.2.7 Birden Çok Koşula Dayalı Sorgulamalar (Not, And, Or).....	102
6.2.8 Bir Veri Kümesi İçinde Arama (In Operatörü).....	103
6.2.9 Aralık Sorgulaması (Between Sözcüğü)	103
6.2.10 Karakter Türü Bilgi İçinde Arama Yapma (Like Sözcüğü).....	104
7. SQL'DE ARİTMETİKSEL İFADELER ve FONKSİYONLAR.....	110
7.1 Aritmetiksel İfadeler	116
7.2 Gruplama Fonksiyonları.....	116
7.2.1 SUM Fonksiyonu	116
7.2.2 AVG Fonksiyonu	117
7.2.3 MAX Fonksiyonu	118
7.2.4 MIN Fonksiyonu	118
7.2.5 COUNT Fonksiyonu	119
7.3 Gruplandıracak İşlem Yapma	120
7.4 Tablolarda Değişiklik Yapmak	123
7.4.1 Tabloya Veri Ekleme	123
7.4.2 Tablo Satırlarını Silme	123
7.4.3 Tablo Satırlarındaki Verilerde Değişiklik Yapma-Güncelleme İşlemi.....	124
7.4.4 Tablonun Yapısında Değişiklik Yapma	125
7.4.5 Bir Tablonun Adını Değiştirme – Rename Table Komutu	127
7.4.6 Mevcut Bir Tablonun Bir Kolonunun Adını Değiştirme – Rename Komutu	127
7.4.7 Mevcut Bir Tablonun Tümüyle Silinmesi – Drop Table Komutu	127
7.5 QUERY Penceresinde SQL Komutları İle Veri Tabanı Oluşturma ve Sorgular Hazırlama	128
7.5.1 QUERY Penceresinde Yazılan SQL Komutlarının Listesi.....	130
7.5.2 Çalıştırılan Sorguların Sonuçları.....	132

8. VERİTABANI MİMARİSİ	149
8.1 Veritabanı Mimarisi	155
8.1.1 Yerleşik Model Veritabanları.....	155
8.1.2 İstemci/Sunucu Modeli Veritabanları	155
8.1.3 Veri Tabanlarının Web İle Birleştirilmesi.....	157
8.1.4 Dağıtık Veritabanları.....	157
8.2 Standart Veri Tabanı Mimarisi	158
8.2.1 Üç Düzeyli Veri Tabanı Mimarisi.....	159
8.3 Katmanlı Yazılım Mimarileri.....	160
8.3.1 Tek-Katmanlı Veritabanı Mimarisi.....	160
8.3.2 İki katmanlı (2-Tier) Veritabanı Mimarisi	161
8.3.3 Üç-Katmanlı Veritabanı Mimarisi	162
9. VERİ TABANI YAZILIMI.....	169
9.1 WAMP SERVER Kurulumu.....	176
9.1.1 WAMP SERVER Kurulum Adımları.....	177
9.1.2 Sunucuların (Servislerin) Çalıştırılması	182
9.1.3 WAMP SERVER Kullanımı.....	182
9.2 Mysql Workbench Kurulumu	186
9.2.1 Kurulan MySQL Workbench programının çalıştırılması	190
9.2.2 MySQL Workbench Kullanımı.....	190
9.2.3 MySQL Workbench Kullanarak Tablo ve İlişki Oluşturma	194
9.2.4 MySQL Workbench Kullanarak Veri Aktarımı.....	200
9.3 Wampserver Sunucusundaki Bir Veritabanını Modeli Olarak Model Ekranına Aktarma.....	201
9.4 MySQL Workbench Kullanarak Veri Aktarımı.....	205
9.4.1 Hazırlanan Modeli Veritabanı Olarak Wampserver Sunucusuna Aktarma	205
10. WEB TABANLI VERİ TABANI SİSTEMLERİ	212

10.1 Web İle İlgili Temel Kavramlar	219
10.1.1 Internet'in Yapısı ve İstemci ve Sunucu Yaklaşımı.....	220
10.1.2 Sunucu (Server) Bilgisayarlar	221
10.1.3 Web Tarayıcılar (Web Browsers)	221
10.1.4 Web Üzerinde Veritabanı.....	222
10.1.5 Veri Tabanlarına Web'de Bağlanma.....	222
10.2 Web Teknolojileri ve Geliştirme Araçları.....	223
10.2.1 HTML	223
10.2.2 XML, Extensible Markup Language.....	224
10.2.3 Web Editörleri (Düzenleyiciler).....	224
10.2.4 Web Tabanlı Veri tabanı Yönetim Sistemleri.....	224
10.3 PHP-MYSQL-APACHE Sunucu Üçlüsü	225
10.3.1 Apache Web Sunucusu	225
10.3.2 PHP	226
10.3.3 MYSQL.....	227
10.4 Web Uygulamaları	228
10.4.1 Web Uygulamalarının Çalışma Biçimi	229
10.5 Web Uygulaması Geliştirme	232
10.5.1 APACHE-PHP-MYSQL Kurulumu	232
10.6 Dreamweaver Üzerinde Yeni Bir Site Açma	236
10.6.1 Local Info	236
10.6.2 Remote Info.....	237
10.6.3 Testing Server	237
10.7 Dreamweaver İle MYSQL Bağlantısı.....	238
10.7.1 Connection Name.....	239
10.7.2 Mysql Server	239

10.7.3 User Name.....	239
10.7.4 Password	239
10.7.5 Database	240
10.8 Dreamweaverde PHP ile Veritabanına Kayıt Ekleme	240
10.8.1 Form Oluşturma	241
10.8.2 Dreamweaver'de PHP ile Kullanıcı Adı-Şifre Korumalı Sayfalar Oluşturmak	244
10.8.3 İkinci Yol (Dreamweaver'de PHP İle Kullanıcı Adı-Şifre Korumalı Sayfalar Oluşturmak)	246
10.8.4 Dreamweaver'de PHP İle Veritabanından Kayıt Okuma ve Rapor Oluşturma	247

ÖNSÖZ

Bilgisayarların en önemli özelliği istediğimiz an istediğimiz verilere ulaşabilmemizi sağlamasıdır. Bunun arkasında ise büyük miktarlarda verilerin bilgisayarlarda tutulması yatar. Bu verileri bilgisayarda tutmak için birçok yöntem geliştirilmiştir. En temel yöntem ise hiç şüphesiz veritabanı yaklaşımıdır. Veri tabanı, tüm bilgi sistemlerinin en temel elemanlarından biridir. Kişisel yazılımlarından, gelişmiş kurumsal sistemlere, ERP yazılımlarından, doküman arşiv sistemlerine, CAD/CAM uygulamalarından ve veri ambarı uygulamalarına kadar her sistemin içerisinde mutlaka bir veri tabanı bulunur.

Veri tabanları, mühendislik çalışma alanlarında, işletme yönetiminde, sağlık sektöründe ve eğitim gibi hemen hemen her alanda kullanılmaktadır. ATM'ler ile banka işlemleri yaparken, kütüphane bilgisayarında tarama yaparken, internet yoluyla alışveriş yaparken, üniversite kayıt işlemlerinde, otel, uçak bilet rezervasyonlarında hep veri tabanlarından yararlanılır.

Veri tabanı verinin depolanması yanında, verilerin bilgiye dönüştürülerek diğer verilerle işlendiği bir dönüşüm süreci olarak görmek daha doğru olacaktır. Veritabanı bilginin hammaddesini depolamanın yanında bilgiyi oluşturan bileşenleri sınıflara ayırarak aralarındaki ilişkileri de dikkate alarak depolar, istendiği zaman birleştirerek sunar.

Bu dersin amacı, öğrencilere kavramsal seviyede nasıl veri modelleneceğinin ve SQL'de nasıl tasarılanacağının öğretilmesidir. Bu derste veri modellemenin uygulamaları pratik olarak Varlık-İlişki Modeli, Normalleştirme gibi konular üzerinde durulmaktadır. SQL sorgu dili, Microsoft SQL Serverde, Microsoft Access ve MySQL kullanılarak öğretilmektedir. İşlenen ana konular: Veri Tabanı Yönetim Sistemi (VTYS) bileşenleri, Veri-İlişki Modeli kullanılarak veri tasarıımı, Veri-İlişki Modelinin tabloya dönüştürülmesi, normalleştirme, sorgulama, SQL Kodları, Veritabanı mimarileri, Veritabanı yazılımları, MySQL, MsSQL, MySQL Workbenc gibi ağ sunucu programlarını kapsamaktadır.

PROF. DR. İbrahim ÇİL

KISALTMALAR

VTYS: Veritabanı Yönetim Sistemi

1. VERİTABANI YÖNETİM SİSTEMLERİNE GİRİŞ

Bu Bölümde Neler Öğreneceğiz?

- 1.1 Veritabanı Temel Kavramları
- 1.2 Veri Yönetimi Yaklaşımları
 - 1.2.1 Geleneksel Veri Yönetimi Yaklaşımı
 - 1.2.2 Veritabanı Yaklaşımı
 - 1.2.2.1 Veritabanı Yaklaşımının Temel Özellikleri
 - 1.2.2.2 Veritabanı Kullanım Nedenleri
- 1.3 Veritabanı Yönetim Sistemleri
 - 1.3.1 Veritabanı Yönetim Sistemlerinin Sınıflandırılması
 - 1.3.2 Veri Tabanı Sistemlerinin Kısa Tarihçesi
- 1.4 Veri Saklama İle İlgili Terimler ve Veri Hiyerarşisi
- 1.5 Veri Saklama Modelleri
 - 1.5.1 OLTP
 - 1.5.2 OLAP
 - 1.5.2.1 Çok Boyutlu Veri Analizi
 - 1.5.4 Big Data Kavramı ve 4 V

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Veri tabanı nedir?
- 2)** Geleneksel veri saklama biçimi neden yetersiz kalmaktadır?
- 3)** Veri tabanı hangi alanlarda gereklidir? Veri tabanlarının kullanım alanlarına örnekler veriniz.

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Veri tabanı kavramlarını bilir	Anlatım, Soru-Cevap, Tartışma
	Veri tabanının gereğini değerlendirebilir	Alıştırma ve Uygulama, Örnek Olay
	Veri tabanı yönetim sistemlerini bilir	Bireysel Çalışma, Problem Çözme,
	Veri saklama modellerini analizedebilir	Proje Temelli Öğrenme

Anahtar Kavramlar

- Veri
- Veritabanı
- Veritabanı Yönetim Sistemi
- Veri Saklama Modelleri

Giriş

Veri tabanı (Database) düzenli bilgiler topluluğudur. Bilgisayar terminolojisinde, sistematik erişim imkânı olan, yönetilebilir, güncellenebilir, taşınabilir, birbirleri arasında tanımlı ilişkiler bulunan bilgiler kümesidir. Bir başka tanımı da, bir bilgisayarda sistematik şekilde saklanmış, programlarca istenebilecek veri yiğinidir. Veritabanı, Belirli bir örgütteki uygulama sistemleri tarafından kullanılan depolanmış verilerdir. Veritabanı, kolayca erişilebilecek, yönetilebilecek ve güncellenebilecek şekilde düzenlenmiş olan bir veri topluluğu. Bir veritabanı, satış işlemleri, ürün bilgileri, stoklar ve müşteri bilgileri ile ilgili kayıtları içerir. Bir kurumda personele ait bilgiler, bir doktorun hastalarıyla ilgili bilgiler, okullarda öğrenci kayıtları hemen akla gelen veri tabanı örnekleridir. Tüm bunlardan hareketle aşağıdaki yapılan tanımlamalar onu anlamakta bir kanat vermektedir.

- Veritabanı, belli bir alanda ve birbiriyle ilişkili olarak düzenlenmiş veriler topluluğudur.
- Veritabanı, birçok kullanıcı tarafından kullanılan birbirleriyle ilişkili geniş bir veri kümесinin düzenlenmesi, depolanması ve sorgulanması için kurulan sistemdir.
- Veritabanı, birçok uygulamaya hizmet vermek için verilerin saklandığı bir veri topluluğudur.

Bütün veri tabanlarının, kayıt giriş, düzeltme, silme, bazı koşullara uyan kayıtları arama, listeleme, seçilecek olan alan veya alanlara göre sıralama yapma, toplam veya alt toplamlar alabilen raporlar oluşturma ve veri dosyaları arasında bağlantı kurabilme özellikleri vardır. Veri tabanlarının amacı büyük miktarındaki kurumsal verileri işlemektir. Veriler düzenli bir biçimde elektronik ortamda kaydedilirler. Düzenli olarak yedeklenen ve kontrol edilen bu bilgiler çok sayıda uygulamanın ve kullanıcının hizmetine sunulur. Büyük miktarlardaki verilenin hızlı ve güvenli bir biçimde gereksinim duyulan bilgiye dönüştürülmesi veri tabanlarının en önemli özelliklerinden birisidir.

Veri tabanındaki yeni tredler aşağıdaki konularda olabilecektir;

NoSQL: SQL kullanılmayan, Sabit tablo yapıları bulunmayan, İlişki tanımlanmayan, ACID kurallarını sağlamayabilen VTYS ler., Akademisyenler bu sistemlere ‘structured storage’ adını veriyor.

Big Data: 4 ExaByte veri, 70 mobil erişim, Sosyal veri. Uygulanan çözümler: MPP veritabanları, Apache Hadoop framework, Dağıtık dosya sistemleri, Dağıtık veritabanları, MapReduce algoritmaları, Bulut bilişim altyapısı

In-memory database: Tüm veri hafızada – hem avantaj hem problem, Daha hızlı, Güç gidince tüm veri kaybolur. Önleme yöntemleri; Snapshot dosyaları, Replication, TRX loglaması, NVRAM kullanımı

Sosyal İş Zekası (Social BI): Sosyal medya iş yapmanın kurallarını değiştiriyor: Satış yapma, doğru kitle ile buluş, Büyük kampanyalar yerine ufak/kişisel aksiyonlar al, Kontrolü bırak, açık ol, İletişim kanalları kapama, her yerde varol.

Database Machine/Appliance: Özelleştirilmiş donanım/yazılımlar, Donanım/yazılım beraber ve entegre geliştiriliyor, Özel çözümler olduğu için entegrasyon problemleri minimum, Daha çok veri ambarı çözümlerinde kullanılıyor

Bulut VTYS (Cloud DBMS)

Mobile BI

1.1 Veritabanı Temel Kavramları

Veri tabanı, birbirleriyle ilişkisi olan verilerin tutulduğu, kullanım amacına uygun olarak düzenlenmiş veriler topluluğunun mantıksal ve fiziksel olarak tanımlarının olduğu ve bunların sayısal ortamlarda saklandığı ve gerektiğinde tekrar bir erişime olanak sağlayan, büyük boyutlarda veriler barındıran bilgi depolarıdır. Veri Tabanı bir sistem olarak düşünüldüğünde dört temel elemandan oluştuğu görülür. Bunlar veri, donanım, yazılım ve kullanıcılar.

Veri: Olguların, kavramların veya talimatların, insan tarafından veya otomatik yolla iletişim, yorumlama ve işleme amacına uygun bir biçimde ifadesidir. Veri şu şekillerde de ifade edilir:

- Bir anlamı olan ve kaydedilebilen gerçekler,
- Bilgisayarda işlenebilen her türlü bilgi,
- Veri kaydedilebilir bilinen gerçeklerdir.

Bu tür veriler bir indekslenmiş deftere kaydedilebilir veya PC yardımıyla diskete veya Excel gibi programlar yardımıyla bilgisayara kaydedilebilir. Genellikle, biz veri veya veri birimleri üzerindeki işlemlerimizi varlık hakkında herhangi bilgi almak için gerçekleştiririz.

Donanım: Kullanılan yazılımin donanımla uygun olması, depolama birimleri ve donanımın güvenliği önemlidir.

Yazılım: Fiziksel veri tabanı ile sistem kullanıcıları arasındaki iletişimini sağlayan yazılıma Veritabanı Yönetim Sistemi (VTYS) adı verilir. Veritabanı Yönetim Sistemi: Bilgisayarda saklanacak bir veritabanının oluşturulmasını ve bakımını sağlayacak yazılım paketi. Veri tabanına ulaşabilmek için kullanıcından gelen bütün istekler VTYS tarafından yerine getirilir. Bir veri tabanını oluşturmak, saklamak, çoğaltmak, güncellemek ve yönetmek için kullanılan programlara Veri Tabanı Yönetme Sistemi adı verilir.

Kullanıcılar: Kullanıcılar üç sınıfta toplanabilir. Bunlar uygulama programcısı, veri tabanı yöneticisi ve son kullanıcılar. Uygulama programcısı: son kullanıcıların kullandıkları programları yazanlar. Bir programlama diliyle programı yazan, veri girişi, ekleme, silme, değiştirme işlemini yapan kişi uygulama programcısıdır. Son kullanıcı, uygulama programcısının hazırladığı programı kullanır. Son kullanıcılar, Veri üzerinde işlem yapanlar ve teknik konularda bilgileri olmadığı varsayılmır. Veri Tabanı Yöneticisi, veri tabanı sistemi kullanan bir kurumda bütün verilere ulaşılabilen ve onlar üzerinde merkezi sorumluluğu olan yöneticilik ve teknik bilgiye sahip kişidir. Bu kişinin en önemli görevi arasında, işletme/kurum/okula ait bilgilerin dağıtımasına ve kontrolün yapılmasını sağlamaktır.

1.2 Veri Yönetimi Yaklaşımları

Veri tabanı oluşturulmasında ya da verinin yönetilmesinde 2 yaklaşımından söz edilebilir.

1. Geleneksel Dosya Sistemleri Yaklaşım

2. Veri Tabanı Yaklaşımı

1.2.1 Geleneksel Dosya Sistemleri Yaklaşım

Geleneksel Yaklaşım (Dosya Sistemleri), kağıt ortamındaki veri tabanlarıdır. Daha sonra bu dosyalama mantığı bilgisayar ortamına da taşınmıştır. Dünyadaki bilginin önemli bir kısmını bu tür sistemler içeriyor ve Dosya tabanlı veri işleme sistemleri erken dönem (genelde) iş verilerinin toptan işlenmesi (batch processing) şeklinde yürütülmekteydi. Geleneksel Veri yönetimi yaklaşımı, dosya kökenlidir. Bu yaklaşımın her bir uygulama kendi dosyalarıyla yürütülmektedir. Yani her bir uygulama problemi için ayrı veri dosyaları oluşturulmakta ve saklanmaktadır. Bilgisayarlar kullanılmaya başlamadan önce dolaplar ve çekmeceler dosya ve klasörlerle doldurulurdu. Bilgiler bu dolap veya çekmecelerde saklanır. Bilgisayarların kullanılmaya başlanmasıyla bilgiler yine yukarıda anlatılan teknikle fakat dolap veya çekmeceler yerine elektronik ortamlarda saklanmaya başlanmıştır. Veri saklama birimlerinde depolanan veri topluluklarına "dosya" adı verilmektedir. Bu sistemlerde saklanacak bilgilerle, saklayacak ve işleyecek programlar birbirine bağlı olarak çalışır. Bilgileri işlemek için kullanılacak olan programın, kullanacağı dosyaların yapıları ve erişim biçimleri hakkında bilgi sahibi olması gerekmektedir. Klasik dosya sistemleri kullanıcıların ihtiyaçlarını karşılayan bir bilgisayar programı yardımı ile verilerin saklanması, aranması ve güncelleştirilmesi işlemidir. Bu tür dosya sistemlerinde her program kendi verisini belirler ve sadece o veriye erişebilir.

Dosya Sistemlerinin Sakıncaları

Geleneksel yaklaşım bir takım dezavantajlara sahiptir. Bunlar şöyle sıralanabilir:

- **Veri Tekrarı:** Aynı veri çeşitli dosyalarda birden fazla yer alabilmektedir buda sistemin hantallaşmasına neden olur. Mesela bir stok dosyasında stok numarası verisinin malzeme dosyasında, fatura dosyasında ve ambar girişi dosyasında yer alması gibi.

- **Verinin Birkaç Dosyada Güncellemesi:** Veri birden fazla dosyada tekrar edilebildiği için, verinin bir dosyada güncellenip diğerlerinde güncellenmemesi Veri Bütünlüğünün (Data Integrity) bozulmasına neden olabilir. Buna bağlı olarak birbiri ile çelişen raporlar üretilebilir.

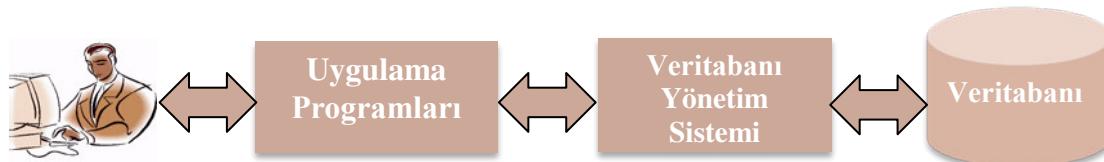
- **Belleğin Tekrarlı Bilgi Nedeniyle İsrafı:** Aynı verinin birden fazla dosya içinde bulunması nedeniyle kullanılan veri hard diskte fazla yer işgal edecek. Yani hard disk tekrarlı veriler için kullanılmış olacaktır.

- **Sadece Belirli Bir Dilin Kullanılması:** Verilerin dosya sisteminde saklandığı ortamlar için değişik programlama dillerinden bir tanesi kullanılır. Kullanılan bu programlama dili ise SQL dili gibi esnek değildir.

Tüm bu olumsuzluklara karşılık, veri tabanı, verileri bir merkezde toplayarak ve veri tekrarını minimize ederek birçok uygulamaya verimli bir şekilde hizmet etmekte üzere organize edilmiş bir veri koleksiyonudur. Her uygulama için verileri farklı dosyalara saklamak yerine, burada veriler fiziksel olarak tek bir yerde depolanır. Tek bir veri tabanı birden çok uygulamaya hizmet verir. Örneğin, işçi bilgilerini personel ve bordro dosyalarında ayrı ayrı saklamak yerine, bu veriler tek bir insan kaynakları veri tabanında oluşturulabilir. Veri tabanı teknolojisi geleneksel dosyalama sistemlerinin neden olduğu çoğu problemi ortadan kaldırır.

1.2.2 Veritabanı Yaklaşımı

Bu yaklaşımın her bir uygulama, birleştirilmiş veri dosyalarını kullanmaktadır. Yani veri dosyaları bütünlüğünü korumaktadır. Bu yaklaşım, verinin birden fazla program tarafından kullanılmasına izin veren bir yaklaşımındır. Veri yönetimine VT yaklaşımını kullanmak için ilave yazılım yani VTYS gerekmektedir. VTYS, bir organizasyonun, veriyi merkezileştirmesine, onları etkin bir şekilde idare etmesine ve saklanmış veriye uygulama programlarında erişilmesine imkân sağlayan bir yazılımı tanımlamaktadır. Bir VTYS, uygulama programları ve fiziksel veritabanı arasında bir arayüz olarak görev yapmaktadır. VTYS; bir veri tabanını oluşturmak, üzerinde istenilen bilgiyi aramak, gerektiğinde bilgi eklemek-silmek-değiştirmek ve veri tabanı ile ilgili her türlü işletim gereksinimleri karşılamak için kullanılan geniş kapsamlı yazılım sistemidir. Veritabanı sistemleri, veri kümelerinin düzenli biçimde tutulduğu ve bu verilerin yazılımlar aracılığı ile yönetildiği ortamlardır. Veri tabanı yaklaşımı, geleneksel yaklaşımın sahip olduğu dezavantajları ortadan kaldırmaktadır.



Şekil 1: Veritabanı Yaklaşımı

1.2.2.1 Veritabanı Yaklaşımının Temel Özellikleri

Veritabanı kendi kendini tanımlar: Veritabanı yönetim sistemi (VTYS), veritabanının tanımını saklar. Bu tanıma ara veri (meta-data) denir. Bu sayede, VTYS yazılımı değişik veri tabanlarını aynı anda ele alabilir.

- **Programlar ve Verinin Bağımsızlığı:** VTYS erişim programlarını değiştirmeden, veri yapılarının ve bunlar üzerinde yapılan işlemlerin değiştirilebilmesi.
- **Verinin Soyutlanması:** Veri modeli ile disk üzerindeki depolama detayları gizlenerek, kullanıcılar veritabanının kavramsal bir görünümünü sunar.

- **Verinin Değişik Görünümleri:** Kullanıcılara veritabanının sadece kendilerini ilgilendiren belli bir görünümünün sunulması.
- **Verinin Paylaşılması ve Çok Kullanıcılı İşlemler:** Kullanıcıların veritabanında eş zamanlı olarak sorgulama ve güncelleme yapabilmesi. VTYS yazılımlarındaki eş zamanlı erişim kontrolü, her işlemin doğru olarak tamamlanmasını veya tamamen iptal edilmesini garanti altına alır.

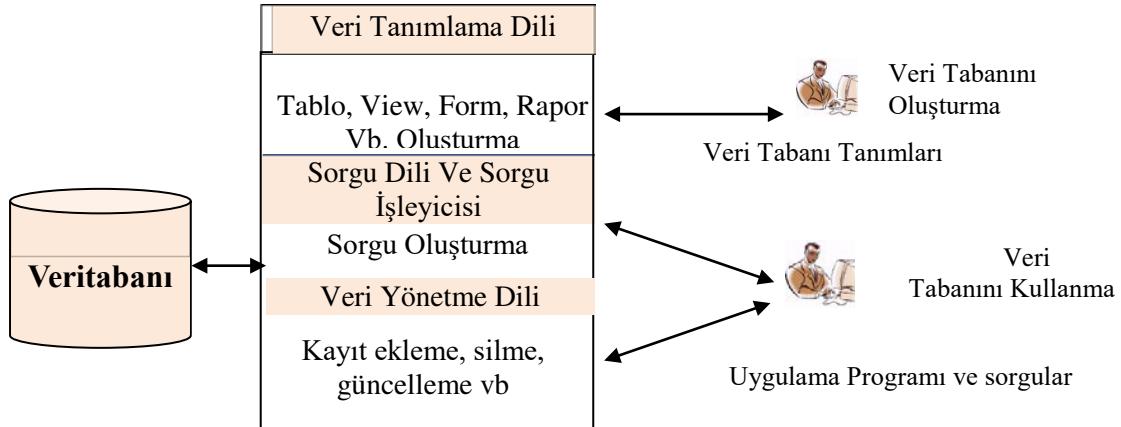
1.2.2.2 Veritabanı Kullanım Nedenleri

Öncelikle veritabanları bir verinin sadece bir kez girilmesini sağlar. Çünkü veri tabanında bir veri sadece bir kez girildiği taktirde verilerde tutarsızlık meydana gelmez ve bellekten tasarruf sağlanır. Veritabanının bir başka kullanım nedeni verilerin paylaşımıdır. VTYS veri tabanındaki verilere aynı anda farklı kullanıcıların erişmesine imkânı sağlar. Verilerin güvenliği de veritabanı kullanımı için sebeplerden biridir. Kullanıcı hakları ve kimlik denetimleri güvenliği sağlayıcı mekanizmalardır. Soru dili imkânları ile veri tabanlarında bulunan verilere hemen her ortamdan erişmek mümkündür. Dosyalarla sağlanamayacak derecede karmaşık ilişkiler VTYS ile sağlanabilmektedir. Veri tabanına doğruluk kıstasları konularak hatalı veri girişi engellenebilmektedir. Bu kıstaslar iki şekilde kullanılabilir. Birinci tip kıstas ile sayısal veri girilecek yere sayısal veriden başkası kabul edilmeyebilir. İkinci tip kıstas için bir örnek ise olmayan üyeye ödünç kitap verilmemesi olabilir. VTYS aynı zamanda yedekleme konusunda da çözümler sunmaktadır. Bazı VTYS’lerde veriler iki yerde kayıt edilirler. Veriler ile uygulamalar birbirinden bağımsızdır.

1.3 Veritabanı Yönetim Sistemleri

Veri tabanı yönetim sistemi (VTYS), yeni bir veri tabanı oluşturmak, veri tabanını düzenlemek, geliştirmek ve bakımını yapmak gibi çeşitli karmaşık işlemlerin gerçekleştirildiği birden fazla programdan oluşmuş yazılım sistemidir. Veri tabanı yönetim sistemi, kullanıcı ile veri tabanı arasında bir arabirim oluşturmaktadır. Veri tabanı yönetim sistemi, veri tabanına her türlü erişimi sağlar. Veri tabanı yönetim sistemi, veri tabanı tanımlama, veri tabanı oluşturma ve üzerinde işlem yapma yeteneği olan bir yazılım sistemidir. Veri tabanının tanımlanması, veri tabanını oluşturan verilerin tip ve uzunluklarının belirlenmesidir. Veri tabanının oluşturulması ise veri için yer belirlenmesi ve saklama ortamına verilerin yüklenilmesini ifade eder. Veri tabanı üzerinde işlem yapmak ise, belirli bir veri tabanının güncellenmesi ve rapor üretilmesi gibi işlerin yapılabilmesini temsil eder. Ayrıca veri tabanı yönetim sistemi, verinin geri çağrılmasını sağlayabilmelidir. Veri tabanına yeni kayıt eklemek, eskileri çağırma ve gerekli düzeltmeleri yapmak yoluyla, verinin bakımını ve sürekliliğini gerçekleştirir. Ayrıca, kayıtlara yeni veri eklemek ve yeni kayıtlar oluşturmakla veri tabanını genişletir. Veri tabanı yönetim sistemi, ilgili kayıtları veya kayıtlar içerisindeki veriyi geri çağrıbilmektedir. Ayrıca veriyi yetkisi olmayan kişilerden korumak, donanım veya yazılım arızaları halinde yeniden çalıştırılmasına yardımcı olmakla, veri tabanındaki verinin güvenliğini de sağlamış olur. Şekil 3 Genel bir Veri tabanı yönetim sisteminin yapısını göstermektedir.

Veri Tabanı Yönetim Sistemi



Şekil 2: Veri Tabanı Yönetim Sisteminin Yapısı

Veri Tanımlama Dili: programcıların veri tabanının içeriğini ve yapısını tanımlamada kullandıkları dildir. Tablo, trigger, view gibi veri tabanı nesneleri tanımlanır. Veri tabanı üzerinde işlem yapan, uygulamaları gerçekleştiren kullanıcıların ise veri tanımlama ya da mevcut tanımları değiştirmeye yetkisi yoktur.

Sorgu Dili: Veritabanları ile iletişim kurmak ve onlar üzerinde işlem yapmak için kullanılır. Veri tabanı uygulamaları için kullanılan en yaygın araç sorgu dilidir.

Sorgu İşleyicisi: VTYS'nin, sorguların işlenmesi ile ilgili görevleri gerçekleştiren bileşenine Sorgu İşleyici (Query Processor) adı verilir. Sorgu İşleyicisinin görevlerinden bazıları şunlar; Sorgunun sözdizimsel ve anlamsal özümlemesini yapmak, Kullanıcının verilen işlemi yapmaya yetkili olup olmadığını denetlemek, Sorguyu işletmek için kullanılabilen algoritmaları (işletim senaryolarını) belirlemek ve “Query Optimizer” alt bileşeni yardımıyla en iyisini seçmek, Sorgunun işletimini gerçekleştirdikten sonra yanıtını oluşturup kullanıcıya iletmek.

Veri Yönetme Dili: Veri tabanı üzerinde, veriyi değiştirme, silme ve güncelleme gibi sorgularla ifade edilemeyecek işlemler gerçekleştirilir. VTYS'de tanımlı Roller ve kullanıcılar için ifade ve nesne kullanma izni tanımlar.

Bir veri tabanı yönetim sistemi yazılımı, karışık ve kapsamlı bir yazılımdır. Kısaca şu işleri gerçekleştirir:

- Veri tabanını oluşturmak ve yönetmek,
- Veri tabanına erişim yetkisi olan kullanıcırlara erişim izni vermek,
- Kullanıcı isteklerine uygun olarak veriye erişimi sağlamak,
- Veri tabanında güncellemeler yapmak.

Veritabanı Yönetim sistemleri en basit Web Sitesinden en karışık bilgilere sahip şirketlere kadar veri yönetmek isteyen herkese hizmet vermek üzere geliştirilmiş programlardır. Başlıcaları DB2, Oracle, MS SQL Server, Sybase, Informix, MySQL, PostgreSQL, Access, Tamino, BerkeleyDB vb. gibi sıralanabilir.

Veri Tabanı Yönetim Sistemlerinin Sağladığı Yararlar

- **Veri Tekrarı Azaltılır:** Aynı veri değişik kişilerin PC'lerinde veya değişik bilgisayarlarda tekrar tekrar tutulmaz; veri tekrarı azaltılır.
- **Veri Tutarlılığı:** Aynı verinin değişik yerlerde birkaç kopyasının bulunması “bakım” zorluğu getirir: bir yerde güncellenen bir adres bilgisi başka yerde güncellenmeden kalabilir ve bu durum veri tutarsızlığına yol açar.
- **Verinin Paylaşımı Sağlanır / Eşzamanlılık:** VTYS kullanılmadığı durumlarda veriye sıralı erişim yapılır. Yani birden çok kullanıcı aynı anda aynı veriye erişemez. Bir VTYS'de ise verinin tutarlığını ve bütünlüğünü bozmadan aynı veritabanlarına saniyede yüzlerce, binlerce erişim yapılabilir.
- **Veri Bütünlüğü:** Bir tablodan bir öğrenci kaydı silinirse, öğrenci var olduğu diğer tüm tablolardan silinmelidir.
- **Veri Bağımsızlığı:** Programcı, kullandığı verilerin yapısı ve organizasyonu ile ilgilenmek durumunda değildir. Veri bağımsızlığı, VTYS'lerinin en temel amaçlarındandır.
- **Verilerin Güvenliğini Sağlar:** Verinin isteyerek ya da yanlış kullanım sonucu bozulmasını önlemek için çok sıkı mekanizmalar mevcuttur. Veri tabanına girmek için kullanıcı adı ve şifreyle korumanın yanı sıra kişiler sadece kendilerini ilgilendiren tabloları ya da tablo içinde belirli kolonları görebilirler. Kullanıcıların her alana erişememesi iyi bir özelliktir. Bunun için çeşitli yetkiler atanır ve verilerle birlikte bu yetkiler de saklanır. Her kullanıcının erişeceği veriler ayrı ayrı tanımlanabilir. Yetkiler ve kısıtlamalar ile istenilen kullanıcı erişim ayarları gerçekleştirilebilir.
- **“Veri Tekrarı” Engellenir:** Aynı veri farklı dosyalarda tekrar tekrar yer almaz. Bu verinin daha az yer kaplamasını sağlar.
- **“Çoklu Güncelleme” Yapılabilir:** Birden fazla dosyada tekrarlanan verinin herhangi birini değiştirdiğimizde diğer dosyalardaki veri de aynı anda değişir.
- **“Gereksiz Bellek Kullanımı” Engellenir:** Aynı veriler defalarca tekrarlanmadığı için bilgisayar belleğinde gereksiz yer işgal etmez.
- **“Erişim Dili” Standarttır:** Veritabanına erişim dili uygulanmadan uygulamaya değişmez. Standart bir dil kullanımı vardır.
- Ayrıca herhangi bir evrak saklamaya gerek kalmaz.

- Bilgiler istenildiği zaman görülebilir.
- Verilerin merkezi kontrolü sağlanır. Bilgilerin kontrolleri tek bir noktadan yapılabilir.
- İstendi anda genel veya özel raporlar alınabilir.

1.3.1 Veritabanı Yönetim Sistemlerinin Sınıflandırılması

Veritabanları genel olarak üç grupta toplanabilir. Birinci grup kişisel diyebileceğimiz veritabanları. Bu veritabanı sistemlerinin kapasiteleri diğerlerine göre daha sınırlıdır. Bunlara örnek olarak MS Access, dBase, FoxPro, Paradox ve Excel gösterilebilir. İkinci grup İlişkisel veritabanı uygulamalarıdır. İlişkisel veritabanları bilgiyi saklama, işleme, yedekleme, raporlama ve geri getirme konularında çözümler getirmektedir. Kurumsal firmaların tercihi bu sistemlerdir. Bugün bilinen ilişkisel veritabanları arasında Oracle, DB2, Sysbase, Informix, Progress, Ms SQL Server bulunmaktadır. Üçüncü grup kurumların çok büyük ve çok boyutlu veri tabanı analizlerine dayalı gereksinimlerini karşılamak amacıyla kurduğu Veri ambarı (Datawarehouse) türü teknolojilerdir. Geleneksel anlamdaki veri ambarı projeleri yalnızca veritabanı sistemlerinin değil, donanım ve işletim sistemini de içine almaktadır. Bu ana sınıflandırma yanında başka sınıflandırmalarda vardır.

Kullanıcı Sayısına Göre: Tek Kullanıcılı ve Çok Kullanıcılı diye ikiye ayrılır. Ancak günümüz teknolojisinin hızla gelişmesiyle PC tabanlı bilgisayarlarla dahi çok kullanıcılı veri tabanı yönetim sistemi kullanılmaya başlanmıştır.

Saklama Biçimine Göre: Veri tabanı yönetim sistemleri veri tabanı verilerini bilgisayarda dosyalarda saklar. Bu saklamayı gerçekleştirmek için bazı yöntemler kullanılır. Günümüzde veri tabanı verilerini saklamak için iki yöntem kullanılır: Tek Dosyada ve Birden Fazla Dosyada. Bu tür veri tabanı yönetim sistemlerinde bütün oluşturulan veri tabanı dosyaları tek bir dosya gibi görünen dosyaların içinde farklı yerlerde tutulur.

Veri Tabanının Fiziksel Konumuna Göre: Veri tabanı yönetim sistemi fiziksel olarak bulunduğu yere göre üç tipe ayrılmaktadır.

- 1)** Merkezi Veritabanları
- 2)** Dağıtılmış Veritabanı
- 3)** Birleştirilmiş Veritabanıdır

Merkezi Veri Tabanları: Merkezi veri tabanında tek bir veri tabanı vardır ve fiziksel olarak tek bir merkezde toplanır.

Dağıtılmış Veri Tabanı: Tek bir veri tabanı veya veri tabanının bazı parçaları çeşitli fiziksel merkezlere dağıtılmış şekilde bulunur. Eğer tüm merkezlerde aynı veri tabanı yönetim sistemi kullanılıyorsa buna homojen, eğer farklı veri tabanı (birinde Oracle diğerinde Sybase

gibi) yönetim sistemi kullanılıyorsa buna heterojen veri tabanı yönetim sistemi denir. Birden fazla veri tabanının birleştirilmesiyle oluşturulmuştur.

1.3.2 Veri Tabanı Sistemlerinin Kısa Tarihçesi

Tarihsel açıdan veri tabanı sistemlerinin gelişimini 3 kuşağa ayırmak mümkündür. 1960 ve 70'li yıllarda kullanılan birinci kuşak veri tabanı sistemlerinde çizge kuramına dayalı hiyerarşik ve ağ modellerinin kullanıldığını görüyoruz. *Birinci kuşak* veri tabanı sistemlerinin getirdiği başlıca yenilikler ve kütük sistemlerinden farklılıklar arasında fiziksel-mantıksal veri modeli ayrimını, veri tanımlarının işlem tanımlarından (uygulama programlarından) ayrılması ve veri tanımlarının kalıcılığının sağlanması ve bir ölçüde fiziksel veri bağımsızlığının sağlanması sayabiliriz.

- 1960lar: Charles Bachmann ilk VTYS'i (IDS-Integrated Data Store) geliştirdi. Ağ modelinde veri bağlantıları çizge ile ifade edildi.
- 1950'lerde ve '60'larda tüm uygulamalar belli gereksinimler için özel olarak geliştirildi ve Dosya temelliydi
- 1960'ların sonları: İlk başarılı ticari VTYS, IBM'de geliştirildi. Hiyerarşî modelde veri bağlantıları ağaç biçiminde ifade edildi. Bu gün de kullanılmaktadır (SABRE reservations; Travelocity)
- 1960'ların sonları: Conference On DAtaSystems Languages (CODASYL) modeli tanımlandı. Bu ağ modeli idi, fakat daha çok standartlaşmıştır.
- 1970'ler: 1970: Ted Codd IBM San Jose Laboratory (şimdi IBM Almaden) laboratuvarında ilişkisel veri modelini tanımladı.
- 1970 - E.F. Codd ve İlişkisel Modeli tanımladı
- 1979 - Ashton-Tate ve ilk mikrobilgisayara dayalı VTYS'ni geliştirdi
- System R (IBM San Jose Laboratory)
- 1976: Peter Chen Varlık İlişkisel(ER) modeli tanımladı

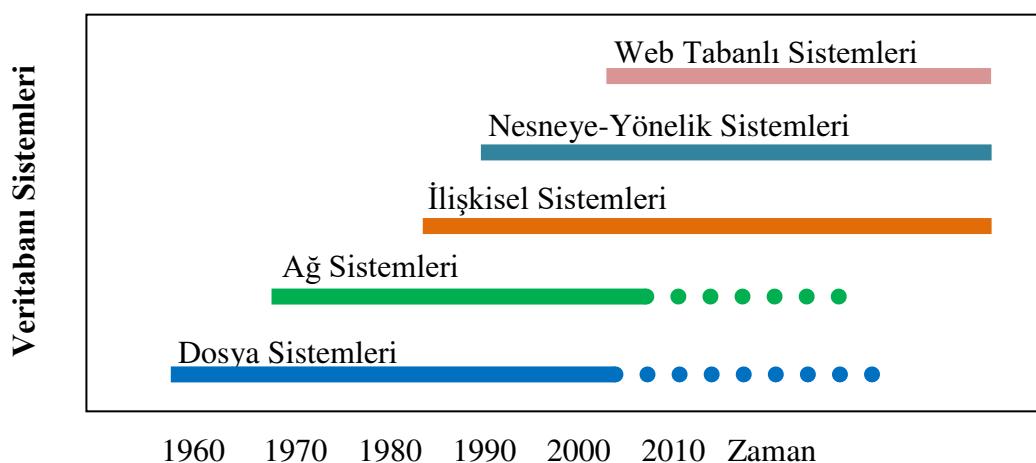
1980'li yıllarda kullanılmaya başlanan **ikinci kuşak** veri tabanı sistemlerin en belirgin özelliği ilişkisel sistemler olmasıdır- ilişkisel sistemlerde mantıksal ve fiziksel veri modeli ayrimının daha da belirginleştiği, fiziksel veri bağımsızlığının daha üst düzeyde sağlandığı ve kullanıcılara küme-yönelik güçlü dil olanaklarının sunulduğu görülmektedir. İlişkisel sistemler 1980'li yılların başından beri çok yaygın biçimde kullanılmaktadır. 1980'leri ilişkisel Veri tabanları teknolojisinin gelişmesi ve SQL'in standartlaştırılması (1980'lerin ortaları-sonu) ve yükseliş devri olarak görülebilir.

İşletim ve yönetimsel amaçlı kurumsal veri tabanları için oldukça uygun ve yeterli olan ilişkisel sistemlerin, başta CAD, CAM, CASE ve CIM gibi bilimsel ve teknik uygulamalar, ses ve görüntü kullanan çoklu ortam uygulamaları, ofis bilişim sistemi uygulamaları, coğrafi bilgi sistemi uygulamaları vb. olmak üzere karakter tabanlı işletimsel-yönetimsel uygulamalar dışındaki birçok uygulama için yetersiz kaldığı görülmüştür.

İlişkisel sistemlerin bu yetersizliği nedeniyle 3. Kuşak veri tabanı sistemleri dediğimiz genişletilmiş-ilişkisel, nesneye-yönelik ve nesne-ilişkisel veri tabanı sistemleri geliştirilmiştir. Günümüze kadar geçen sürede, veri modeli alanındaki gelişmelerin üç noktasını nesneye yönelik veri modelleri oluşturmaktadır. Bunun üzerine web teknolojilerindeki gelişmelerle yeni boyutlar kazanmıştır.

- 1990'lar İlişkisel teknolojinin yaygınlaşması yükselişi, paylaşılan sistemlerin oluşumu ve yeni veri modelleri: nesneye yönelik, tümdengelimli.
- 1990'ların sonu: nesneye yönelik teknolojinin ilişkisel VTYS'le birleştirilmesi→Nesne-ilişkisel VTYS.
- Büyük ölçekli karar destek sistemleri ve veri madenciliği uygulamaları. Büyük veri ambarlarının oluşturulması.
- 2000 lerden sonra XML and XQuery standardlarının geliştirilmesi. Çok büyük veri depolama sistemleri- Google BigTable, Yahoo PNuts, Amazon,..

Organizasyonlar veri tabanlarında sakladıkları bilgileri daha iyi kullanabilmek için güçlü veri analizi araçlarından ve WWW'e bağlı veritabanı teknolojilerinden faydalıyor. Yeni uygulama alanları: Veri ambarları ve OLAP, Web Internet, çoklu ortam ve metinlerin işlenmesi. Otomatik veri tabanı sistemlerinin geliştirilmesi (Şekil 3).



Şekil 3: Zaman İçerisinde Veritabanı Sistemlerinin Gelişim Seyri

Bulut Bilgişleme ve Bulut Üzerinde Veri Tabanı

Bulut bilgişleme adını veren bulut yapısı aslında interneti simgeliyor. Uzmanlar bu bulutun internetin bir sonraki evrimi olduğunu söylüyor. Bu bilgi bulutu ihtiyacınız olan tüm verileri içine **depolayacak**. Bulut (Cloud) olarak ifade edilen teknoloji, sadece webmaster ve site sahiplerini ilgilendiren “hosting”, “sunucu”, “uzak masaüstü”, “veritabanı”, “bandwidth”, “trafik” gibi hizmetlerin herkese arz edilmesine sebep olmuştur. Bulut: birden fazla sunucunun birbiri ile bağlanmasıdan oluşan bir servistir. Sunucuların aynı veri merkezi içinde olma zorunluluğu da yoktur, bir ağ ile birbirlerine bağlı olmaları yeterlidir. Bulut, güçlü senkronizasyon özelliği olan sunucuların bir araya gelmesidir. Bulut sunucusu, kullanmakta olduğu özel donanımlar ve yazılımlar nedeniyle yüksek performanslı çalışan, dinamik olarak ölçeklendirilebilir ve ihtiyaçlarınıza uygun hale getirilebilir bir sanal sunucu hizmetidir. Bulut bilişim sayesinde artık kişisel verilerimizi yanımızda taşımamız gerekmiyor istediğimiz zaman istediğimiz yerden hatta telefonlarımızdan bile depoladığımız veriyi değiştirip düzenleyip yada indirip veya yükleyebileceğiz. Temeli verilerimizi internet ortamına depolamak ve de internet ortamında bunları çalıştırın programlara erişmektir. Binlerce belki yüzbinlerce sayıdaki sunucuya depolanan verileri ister ücretsiz hizmet kullanarak tıpkı Google gibi ya da belli bir miktar ücret karşılığı bu hizmetten faydalana bilinecek.

1.4 Veri Saklama İle İlgili Terimler ve Veri Hiyerarşisi

Bir bilgisayar sistemi verileri şu sıradır organize eder (Şekil 4): Bit ve byte, field (alan), record (kayıt), file (dosya), veri tabanı. Bir Bit bilgisayarın saklayabileceği en küçük birimdir. Bir grup bit, yani byte, tek bir karakteri temsil eder. Bu karakter bir harf, bir sembol yada sayı olabilir. Bir grup karakterin bir kelime yada kelimeler grubu haline gelmesiyle bir “alan” oluşturulur(Örn: İsim). Birbirile ilişkili bir grup “alan” bir araya gelerek bir kayıt (record) teşkil eder. Kayıtlar bir araya gelerek bir tabloyu teşkil eder. Tablolar ise bir araya gelince veri tabanı oluşturur. Veri Ambarı da veri tabanlarının entegrasyonundan oluşur.

Bit (1/0)	Bit (1/0) 0,1
Byte (8 bit)	Byte (8 bit) 01000001
Karakter (ASCII- UNICODE vb.)	Karakter (ASCII- UNICODE vb.) A
Kelime (1 veya birden çok karakter)	Kelime (1 veya birden çok karakter) Ali
Kayıt (Kelime topluluğu)	Kayıt (Kelime topluluğu) Ali Kaya 1989 Adana
Veritabanı(Kayıtlar Topluluğu)	Veritabanı(Kayıtlar Topluluğu) Ali Kaya 1989 Adana Veli Keser 1986 Çorum Ayşe Bal 1988 Kayseri
Veri Ambarı (whereshere) (Veritabanı topluluğu)	Veri Ambarı (Veritabanı topluluğu)

Şekil 4: Veri Hiyerarşisi

1.5 Veri Saklama Modelleri

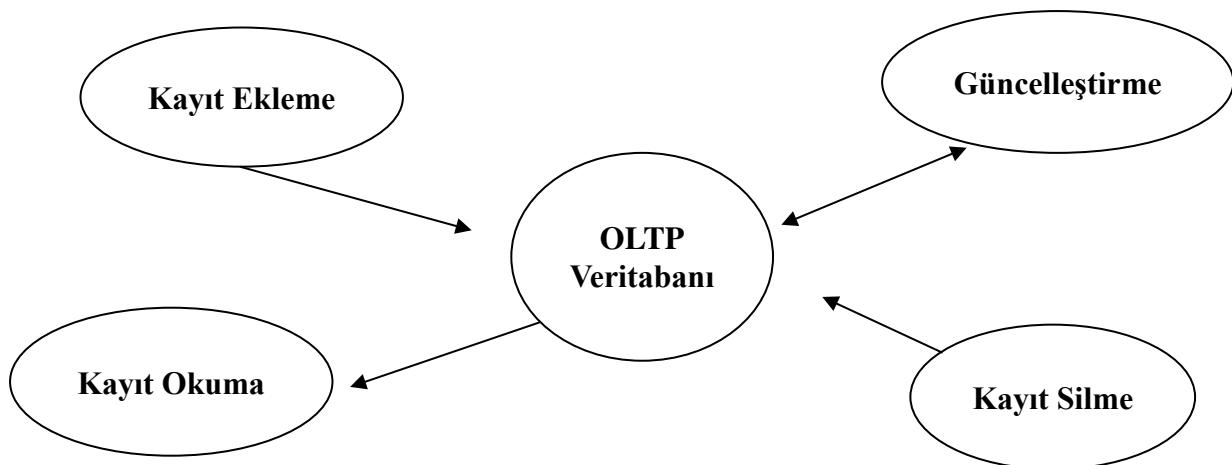
Veritabanları günümüzde yoğun olarak veri üretilen yerlerde iki genel amaca yönelik olarak kullanılır. Bunlardan biri, üretilen verilerin anlık olarak saklanması işlemidir ki bu türden işlemler için optimize edilmiş sistemlere OLTP (On-line Transaction Processing) adı verilir. Bir OLTP veritabanı içinde veriler genellikle ilişkisel tablolar şeklinde organize edilir. Gereksiz veri yığınları azaltır ve veri güncelleme hızını arttırmır. Örneğin SQL Server çok sayıda kullanıcının gerçek zamanlı olarak veri analiz edebilmesini ve güncellemesini sağlar. Örnek olarak OLTP veritabanları havayolu bilet satış bilgileri ve bankacılık işlemlerini içerir. OLTP, veri girişinin sürekli olduğu bankacılık ve taşımacılık gibi sektörlerde kayıt takip işlemlerini yapmak için kullanılan uygulamaları yöneten bir program türüdür. OLTP sistemlerde sürekli olarak veride değişiklikler olur, her an eklenmeler yada silinmeler olabilir.

SQL Server, Oracle ve Sybase gibi bir çok VTYS, hem OLAP hem de OLTP sistem olarak ayarlanabilir özellikleri bünyesinde barındırır. Ancak bir veritabanı yöneticisinin ilgili ayarlamaları yapması gereklidir. Veritabanının kurulumu, yedeklenmesi, replication gibi düzenli bakım gerektiren işlemlerinin gerçekleştirilmesi de ayarlamaları ile birlikte, kullanıcı yönetimi ve yetkilendirme gibi işler 'Veritabanı Yönetimi' olarak adlandırılmaktadır. OLAP teknolojisi büyük verilerin organize edilmesi ve incelenmesini sağlar. Bir analist büyük miktarlardaki verileri hızlı ve gerçek zamanlı olarak değerlendirebilir. Örneğin SQL Server Analiz Servisi toplu raporlama ve analizde, veri modelleme ve karar desteği kadar geniş alanda çözümler sunar.

1.5.1 OLTP (Online Transaction Processing) Sistemler (Çevrimiçi Hareket İşleme (ÇİHİ) Sistemleri

Bir kurumun verilerinin işlendiği ortamlara OLTP (Online Transaction Processing) sistemler adı verilmektedir. OLTP (Online Transaction Processing, Çevrimiçi Hareket İşleme) sistemler günümüzde yaygın olarak kullanılan veri tabanı sistemlerine verilen isimdir. Veri tabanları, aynı anda birden fazla kullanıcının, verileri hızlı ve etkin bir şekilde girişi, çıkıştı ve güncellenmesi için tasarlanmış yapılardır. OLTP veri tabanları; sürekli yeni veriler eklenen, verilerin modifiyesinin yapıldığı anlık durumlar için oluşturulurlar.

Sisteme yeni bilgilerin eklenmesi, varolan bilgilerin değiştirilmesi ya da silinmesi gibi işlemler ile sürekli olarak güncellenen ve bir değişim içindeki verilerden oluşan sistemlere özgü işlemleri kapsar. Örn: Bankacılık, borsa, okul sistemleri, ÇİHİ işlemleri, veri tabanı üzerinde sürekli olarak yeni operasyonların aynı anda birçok kullanıcı tarafından gerçekleştirilmesini sağlar. Örneğin bir işletmenin sahip olduğu stok sistemi ile depoya giren ve çıkan ürünleri ve ödemeleri izlenebilir. OLTP sistemlerine ilişkin veritabanlarına veri kaydedilebilir, veriye erişilerek raporlanabilir ve istendiğinde veri silinebilir.

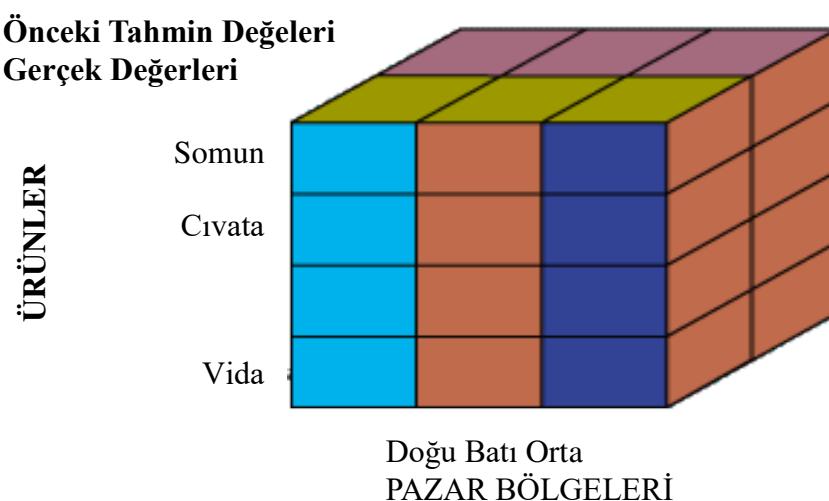


1.5.2 OLAP (Online Analytical Processing) OLAP- Eş Zamanlı Analistik Veri İşleme

OLAP (Online Analytical Processing), daha çok raporlama ve karar destek amacı ile kurulan sistemlerdir. OLAP- Eş Zamanlı Analistik Veri İşleme şeklinde Türkçeye çevrilebilir. Bu türden sistemler aracılığıyla veri ambarı ve veri pazarı gibi yapılar kullanılarak yoğun bir şekilde üretilmiş verilerin analizleri ve raporları oluşturulur. Böylece tüketici ve satış eğilimleri, üretim maliyetleri gibi konularda kullanılacak sonuçlar elde edilir. Bu araç çok boyutlu veri analizinde kullanılır. Bilginin her bir yönü (ürün, fiyat, bölge, maliyet, zaman) farklı bir boyutu temsil eder. Çok boyutlu analizler birden çok boyut kullanarak, kullanıcıların aynı verileri farklı şekillerde görmesini sağlar. Çevrimiçi Analistik İşleme (OLAP), Veri tabanları (özellikle veri ambarları) üzerinde çok boyutlu tablolamaya, hızlı biçimde özet veriler sunma (miktar, ortalama, standart sapma vb.) sürecine Çevrimiçi Analistik İşleme denilmektedir. OLAP işlemi çok boyutlu veritabanlarında veya çok boyutlandırılmış veritabanlarında gerçekleştirilmektedir.

1.5.2.1 Çok Boyutlu Veri Analizi

Organizasyonlar veritabanlarında sakladıkları bilgileri daha iyi kullanabilmek için güçlü veri analizi araçlarından ve web tabanlı veritabanı teknolojilerinden faydalaniyor. Bazen yöneticiler eldeki verilerini geleneksel veritabanı modellerinin sunduğu yöntemlerden farklı şekillerde analiz etmek ister. Örneğin, somun, cıvata, vida ve delikli pul (rondela) olmak üzere dört farklı ürünü üç farklı bölgede satan bir şirket, gerçek satışlarının碌nlere göre ve bölgelere göre miktarını bilmek isteyebilir. Ve bu satış rakamlarını önceki tahmin değerleri ile karşılaştırabilir. Bu tür bir analiz veriye çok boyutlu bir bakış açısını gerektirir. Bu nedenle Çok boyutlu veri analizi olarak ta ifade edilir.



Bu tür bir bilgi sunabilmek için, organizasyonlar ya özel bir çok boyutlu veri tabanı kullanmalıdır, yada ilişkisel veritabanındaki verilerin çok boyutlu görünümlerini veren araçlar kullanmalıdır. Çok boyutlu analizler birden çok boyut kullanarak, kullanıcıların aynı verileri farklı şekillerde görmesini sağlar. Bir diğer ismi: on-line analytical processing (OLAP). Bilginin her bir yönü (ürün, fiyat, bölge, maliyet, zaman) farklı bir boyutu temsil eder.

Çok boyutlu VTYS;

- İlişkisel VTYS nin bir türevi
- Çok boyutlu – boyut / değer ilişkisine dayalı,
- İlişkisel VTYS lerden beslenirler,
- MDX standart dili ile sorgulanabilir.

Veri Ambarı (Data Warehouse): birçok kaynaktan bilgi toplayan ve bu bilgileri son kullanıcının erişebilmesi için birleştiren ve tutarlı hale getiren bir karar destek sistemidir. Diğer sistemlerin veri tabanlarından veri alabilemeyeceğini sağlarlar. Veri ambarı, veri tabanlarının entegrasyonudur. Birçok veritabanından gelen bilgiyle beslenir ve son kullanıcının erişimi için hazır bulundurulur. Karar destek sistemlerinde kullanılır. Veriyi yüksek maliyetli bireysel sistemlerde değil, düşük maliyetli büyük sistemlerde kullandıklarından maliyetlerde düşme sağlarlar. Son kullanıcılara kolay kullanılan sorgulama ve analiz araçlarıyla ve grafik arayüzlerle bilgiye ulaşım imkanı sunarlar. Bilgiler, ayrı sistemlerden değil, sürekli güncellenen tek bir sistemden alınır.

Veri Pazarları (Datamart): Veri ambarlarının daha küçük versiyonlardır ve daha ziyade departmanlara yönelik küçük veri kümelerini depolarlar.

İş Zekâsı: Ham veriyi iş kararları alabilecek şekilde raporlanabilir, anlamlı hale getiren, analize yardımcı olan yazılım süreçlerinin tamamıdır. İş zekası Veri toplama/konsolidasyon, Veri dönüşümü, Veri temizliği, Veri madenciliği ve Raporlama faaliyetlerini kapsar.

Big Data Kavramı: "Zamanla toplanan büyük miktarda verilerin ekonomik ve ölçeklenebilir bir biçimde ve ilişkisel veritabanı tekniklerinin yetmediği noktalarda kullanılabilir ve anlaşılmıştır kılınması kavramıdır". Verinin önemi ve değeri her geçen gün artmaktadır. Verinin öneminin ve değerinin artmasının nedeni ise veri miktarlarının ve çeşitliliğinin artıyor olması.

1.5.2.2 Big Data Kavramı ve 4 V

Volume (Hacim): Big Data kavramının oluşmasına sebep olan belkide en önemli neden elimizdeki verinin hacminin her geçen gün logaritmik olarak artıyor olması. Verilerin hacmi bu kadar artarken doğal olarak şirketlerin BT maliyetleri de artmaktadır. Artan BT maliyetlerini kısmak ve bütün bu verilerin saklanması ve yönetileceği ortamı ayarlamak gerekmekte.

Velocity (Sürat): Hacmi artan verilere ek olarak bu verilerin çok hızlı bir şekilde sisteme aktığını ve karşılamak durumunda olduğunu düşünün. İlişkisel veritabanına bu kadar hızlı veriyi yüklemek zahmetli ve maliyetli olacaktır. Dolayısıyla verinin hacmi dışında sisteme çok süratli bir biçimde akıyor olması da Big Data kavramını ve kullanımını açıklayan bir diğer V'dir.

Variety (Çeşitlilik): Sosyal medya, sensör verileri, CRM dosyaları, dokümanlar, imajlar, video'lar vb. aklınıza gelebilecek bütün verileri, kaynakları ve tiplerini hayal edin. Bunların tamamını ilişkisel bir veritabanında sakladığınızı hatta veritabanını geçtim, bildiğimiz bir file system üzerinde bile saklamak pek mümkün değil ve maliyetlidir. Verilerin çeşitliliği artmışsa ve bütün bu verileri işlemek, analiz etmek ve saklamak istiyorsak Big Data kavramı bunun için biçilmiş kaftan.

Value (Değer): Diğer 3 V'nin bir araya gelmesi ile bir başka V oluşturmaktadır. Bu da value, yani değerdir. Yüksek hacimlerde, çeşitli ve çok hızlı akan ve sisteme giren verinin bir de değerinin olması lazım. Aksi halde katlanılan maliyet $>$ elde edilen değer haline gelir. Bunun olmaması için sahip olduğumuz verileri anlamlandırmamız, değer katmamız ve analiz çalışmasını yapmamız gerekmekte. Böylece Big Data kavramının dördüncü V'si value olmuş oluyor.

Tablo: Bilgi Sistemlerinin Karakteristik Özellikleri

Bilgi Sistemi	Amacı
Kayıt İşleme Sistemleri TPS (Transaction Processing System)	Günlük işlemler sonucu oluşan verinin toplanması ve saklanması
Yönetim Bilişim Sistemleri MIS (Management Information System)	Bir TPS'deki veriyi bir organizasyonu planlamak, kontrol etmek ve yönetmek için bilgiye (information) dönüştürür.
Karar Destek Sistemleri DSS(Decision Support System)	Veri analiz edilerek ve işlenerek yönetimin karar vermesine destek sağlar.
Üst Düzey Bilgi / Destek Sistemleri EIS (Executive Information System)	Üst yönetime organizasyonun performansını artırmak, stratejiler geliştirmek ve uygulamak için bilgi sağlar.
OLAP- Eş Zamanlı Analitik Veri İşleme OLAP (Online Analytical Processing)	Verilere hızlı erişim ve çok boyutlu analiz sağlar.
Veri Madenciliği Data Mining	Verideki gizli (saklı) ilişkileri ortaya çıkarmak için istatistik analiz ve yapay zeka tekniklerini kullanır.

Uygulamalar

Veri Tabanı Örneği-1: Mary Richards Housekeeping

– Serbest girişimci

– Tek kullanıcılı veri tabanı

– 3 Tablo(Customers, Jobs, Source)

– Veri gereksinimleri:

- Müşteri, iş ve referansların (tavsiyelerin) birbirile ilişkilerini izle

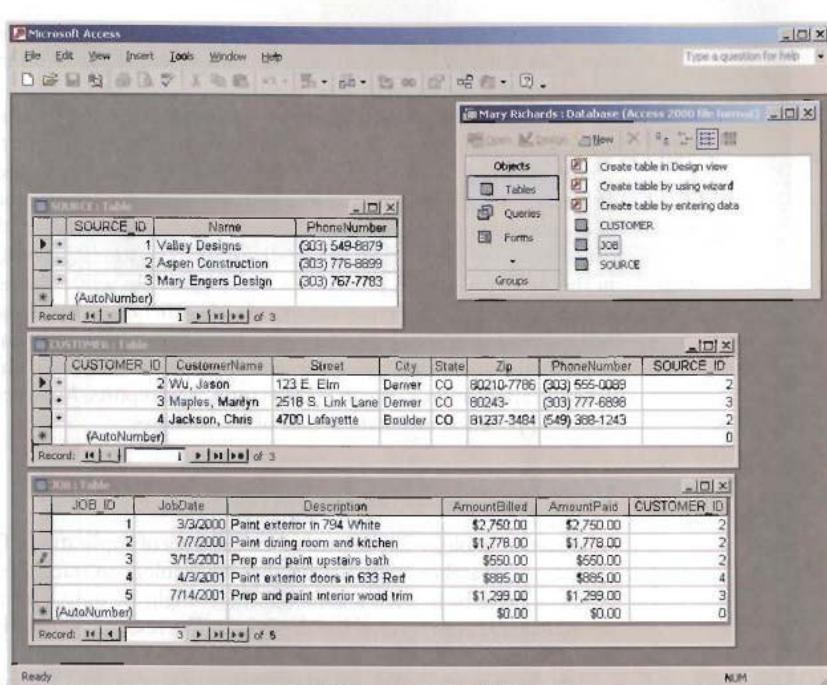
- İhale tahminlerini kaydet

- Referans kaynaklarını izle

- Adres etiketleri üret

FIGURE 1-10

Tables of Data for
Mary Richards
Housepainting



Veri Tabanı Örneği-2: Treble Clef Music

– Yerel ağ üzerinde çok kullanıcılı veri tabanı

– 3 Tablo(Customers, Instruments, Rentals)

– Veri gereksinimleri:

- Kiralanan enstrümanları izle

- Birden çok kullanıcının sorunlarını hallet

CUSTOMER

Treble Clef Music – Customer Form

CustomerName	Mary & Fred Jackson	Children
HomePhone	(703) 443-7788	Katherine
WorkPhone	(703) 443-4482	Jaymalina
Street	1200 Seventeenth Ave	*
City	Alexandria	Record: [◀] [◀] [1] [▶]
State	VA Zip 02234-5567	

INVOICES

InvoiceNumber	InvoiceDate	Total
100087	10/16/2001	\$45
98884	10/16/2000	\$37
*	0	\$0

Record: [◀] [◀] [1] [▶] [▶] [*] of 2

Record: [◀] [◀] [1] [▶] [▶] [*] of 2

Rental Agreement

Treble Clef Music – Rental Agreement Form

InvoiceNumber	100087	Customer	Mary & Fred Jackson
InvoiceDate	10/16/2001	WorkPhone	(703) 443-4482
		HomePhone	(703) 443-7788

Rental Items

SerialNumber	Category	DateOut	DateReturned	MonthlyFee
478990	B flat clarinet	10/16/2001		\$17.50
556788	Standard violin	10/16/2001		\$27.25
478990	B flat clarinet			
556788	Standard violin			
556790	Premium violin			

Record: [◀] [◀] [1] [▶] [▶] [*] of 2

INSTRUMENT

Treble Clef Music — Instrument Data Form

SerialNumber	478990	MonthlyFee	\$18
Category	B flat clarinet	Rented?	No

INVOICES

InvoiceNumber	InvoiceDate	Total
100087	10/16/2001	\$44.75
*		

Record: [◀◀] [◀] 1 [▶] [▶▶] [▶*] of 1

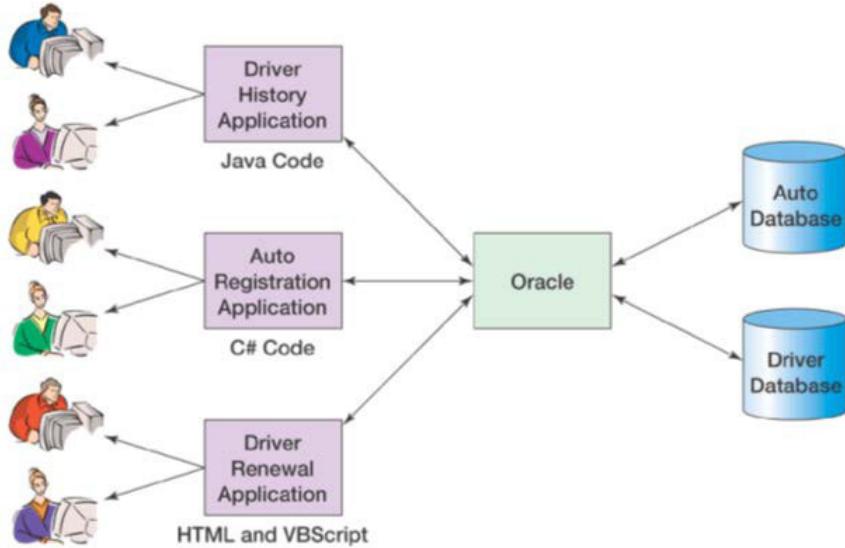
Record: [◀◀] [◀] [▶] 1 [▶] [▶▶] [▶*] of 3

Veri Tabanı Örneği-3: State Licensing & Vehicle Registration Bureau

- 52 Merkez, 37 Ofis, Yüzlerce kullanıcı
- 40 Tablo
- Veri gereksinimleri:
 - Ehliyetle ilgili sorunları izle
 - Trafik cezaları, kazalar, tutuklamalar, sınırlamalar
 - Otomobil ruhsatlarıyla ilgili sorunları izle
 - Gelirler, yasa uygulama
 - Birçok bölümün gereksinimlerini bütünlleştir

Örgütsel Veri Tabanı

Figure 1.15 Organizational Database System



Veri Tabanı Örneği-4: Calvert Island Reservations Centre

- Ticaret Odası
- Verilere erişim sağlayan reklam (promosyon) veri tabanı
- Müşteri ve yer ayırtma veri tabanı süreçleri
- Veri gereksinimleri:
 - Çoklu ortam verileri depola (fotoğraflar, video ve ses klipleri)
 - Web'den erişilebilir olmalı
 - HTTP, DHTML, ve XML gibi web teknolojilerini kullanmalı

Kaynak: David M. Kroenke, 2004, Database Processing: Fundamentals, Design and Implementation.

Uygulama Soruları

- 1)** Bir bankada hangi veriler veritabanında saklanmalı, bu veri tabanının kullanıcıları kimlerdir?

Bu Bölümde Ne Öğrendik Özeti

Bu haftada veritabanı temel kavramları, veri yönetimi yaklaşımı, veri modeline göre veri tabanı yapıları, veri tabanı sistemlerinin kısa tarihçesi, veritabanı yönetim sistemleri, veritabanı yönetim sistemlerinin sınıflandırılması ile ilgili konular ele alınmıştır.

Bölüm Soruları

1. Aşağıdakilerden hangisi veritabanı hakkında söylenemez?

- a) Birbirleriyle ilişkisi olan veriler tutulur.
- b) Kullanım amacına uygun olarak düzenlenmiş veriler topluluğunun mantıksal ve fiziksel olarak tanımları yapılır.
- c) Veriler sayısal ortamlarda saklanır ve gerektiğinde tekrar bir erişime olanak sağlar.
- d) Büyük boyutlarda verileri barındırır.
- e) Hiçbiri.

2. Bir veri tabanı yönetim sistemi yazılımı, karışık ve kapsamlı bir yazılımdır ve aşağıdaki işlerden hangisini gerçekleştirmez?

- a) Veri tabanını oluşturmak ve yönetmek.
- b) Verinin birkaç dosyada güncellemesini sağlamak.
- c) Veri tabanına erişim yetkisi olan kullanıcılarla erişim izni vermek.
- d) Kullanıcı isteklerine uygun olarak veriye erişimi sağlamak.
- e) Veri tabanında güncellemeler yapmak.

3. Veri Ambarı aşağıdaki ifadelerin hangisiyle ifade edilebilir?

- a) Veri tabalarının entegrasyonu.
- b) Bir veya birden çok tablonun birbiriyle ilişkilendirilmesi.
- c) İlişkisel veri tabanı.
- d) OLAP
- e) OLTP

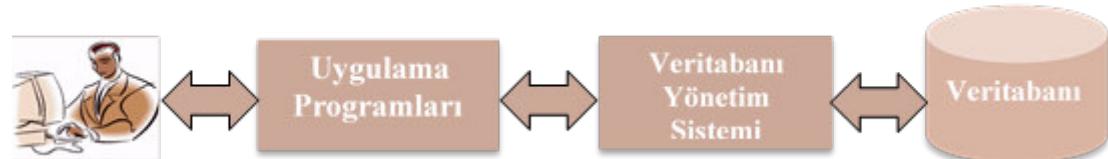
4. Hangisi veri tabanı kullanmanın avantajı değildir?

- a) Bilgi Saklamak
- b) Bilgiyi Şifrelemek
- c) Bilgiyi Analiz Etmek
- d) Bilgiyi Paylaşmak

CEVAPLAR

1-E, 2-B, 3-A, 4-C

5. Aşağıdaki şema neyi anlatmaktadır?



2. VERİTABANI TASARIMI

Bu Bölümde Neler Öğreneceğiz?

2.1 Veritabanı Tasarımında Yapılması Gereken Hususlar

2.2 Veritabanı Geliştirme Aşamaları

2.2.1 İhtiyaç Analizi

2.2.2 Veritabanı Tasarımı

2.2.2.1 Kavramsal Tasarım

2.2.2.2 Mantıksal Tasarım

2.2.2.3 Fiziksel Tasarım

2.2.2.3.1 Veri Gösteriminin Belirlenmesi

2.2.2.3.2 Erişim Yöntemlerinin Seçimi

2.2.2.3.3 Verinin Dış Belleklere Atanması

2.2.2.3.4 Veritabanının Yüklenmesi

2.2.2.3.5 Veritabanının Tekrar Düzenlenmesi

2.3 Veritabanının Tamamlanması

2.4 Sistemin Test Edilmesi ve Bakımı

2.5 Veri Tabanının İşletilmesi ve Yönetilmesi

2.5.1 Veritabanı Yöneticisi

2.5.2 Veritabanı Güvenliği

2.5.3 Veri Güncelleme

2.5.4 Veritabanı Yedekleme

2.5.5 Veritabanı Kurtarma

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Bir veri tabanı nasıl geliştirilir?
- 2)** Bir kurumun ihtiyaç duyduğu bir veri tabanı için gereksinim analizi nasıl yapılmalı?
- 3)** Kavramsal tasarım nedir?
- 4)** Kavramsal tasarım nasıl gerçekleştirilir?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Veri tabanı için gereksinim analizi yapabilir	Anlatım, Soru-Cevap, Tartışma
	Bir veri tabanını tasarılayabilir	Alıştırma ve Uygulama, Örnek Olay
	Veri tabanını tasarım ekiplerinde çalışabilir.	Bireysel çalışma, problem çözme,
	Veri tabanını uygulamaya koyabilir	Proje temelli öğrenme

Anahtar Kavramlar

- Gereksinim Analizi
- Veri Tabanı Tasarımı
- Kavramsal Tasarım
- Mantıksal Tasarım
- Fiziksel Tasarım

Giriş

Bir veritabanı üzerinde çalışmaya başlanmadan önce, yapılacak işe uygun bir veri tabanı tasarımları yapılmalıdır. Tasarım veri tabanı oluşturma işinin en önemli aşamasıdır. Başlangıçta iyi tasarlanamayan bir veritabanı, ileride geriye dönüşü olmayan verimsiz bir bilgi yığınına dönüşebilir. En basit hali ile veritabanı tasarımda; hangi tabloların olacağı, bu tablolarda hangi alanların olacağı, tablolar arasındaki alan ilişkilerinin neler olacağı ve alanlara ait özelliklerin tanımlanması yapılır. Alan özelliklerinde alan adı, alan tipi, alanın uzunluğu, alanın varsayılan değeri, bu alana yazılacak verilerin geçerlilik koşullarının başta tasarlanması gereklidir.

Veritabanı tasarımlı işlemine belli ilkeler yön verir. Bilgilerin yinelenmesi, gereksiz veri olarak da bilinir, istenmezler; çünkü bunlar gereksiz alan kaplar, hata ve tutarsızlık olasılığını artırır. Bilgilerin doğru ve tam olması önemlidir. Veritabanında yanlış bilgiler varsa, veritabanından bilgi alan raporlar da yanlış bilgi içerecektir ve kararlar yanlış bilgilere dayanacaktır. Bu nedenle iyi bir veritabanı tasarımlı; Gereksiz verileri azaltmak için bilgileri konulara göre oluşturulmuş tablolara böler. Bilgilerin doğruluğunu ve tutarlığını sağlar ve destekler. Veri işleme ve raporlama gereksinimleri birbirleriyle uyumlu hale getirir.

İyi tasarlanmış bir veritabanı, güncel ve doğru bilgilere erişiminizi sağlar. Doğru tasarım, veritabanı çalışmalarınıza yönelik hedeflerinizi başarmanız açısından son derece önemli olduğu için tasarım ilkelerini öğrenmek adına gerekli zaman yatırımı yapmak akıllıca olacaktır. Veri tabanı tasarımlı yapılmırken ileride çıkabilecek sorunlar en baştan iyice düşünülmelidir. Yapılacak çalışmanın diğer kurumlarda kullanabilecek şekilde düzenlenmeli.

Tasarım gerçekleştirildiğinde gerekmeyen ama teknolojinin gelişmesiyle veya şirketin gelişmesiyle ortaya çıkabilecek bazı veriler, bilgiler ve ek işlemler için fazladan alanlar veya tablolar oluşturulmalıdır. Sistemde tasarlanan alanlar mümkün olduğunda standartlaştırılmalı. Sistem tasarımında mümkün olduğunda var olan verilerden elde edilen sonuçlara yer verilememelidir. Fiyat ve ıskonto miktarı sistemde mevcutken ıskontolu fiyat sistemde tutulmamalıdır.

2.1 Veritabanı Tasarımında Yapılması Gereken Hususlar

Veritabanı tasarımına başlamadan önce tasarım yapılacak konu hakkında ayrıntılı bir bilgi toplamamız yani tasarım yapılacak sistemin analizinin yapılması gerekmektedir. İkinci olarak veritabanının tasarım işlemi gerçekleştirilir. Tasarım yapılan yeni sistem üzerinde çeşitli uyarlama işlemleri yapılarak en iyi çalışacak sistem bulunur. Tasarım işlemi bittikten sonra sistemi test etme işlemi gerçekleştirilir. Sistem üzerine veriler girilerek test işlemi gerçekleştirilir. Bu test işleminden elde edilen sonuçlar kullanılarak sisteme son şekli verilir.

2.2 Veritabanı Geliştirme Aşamaları

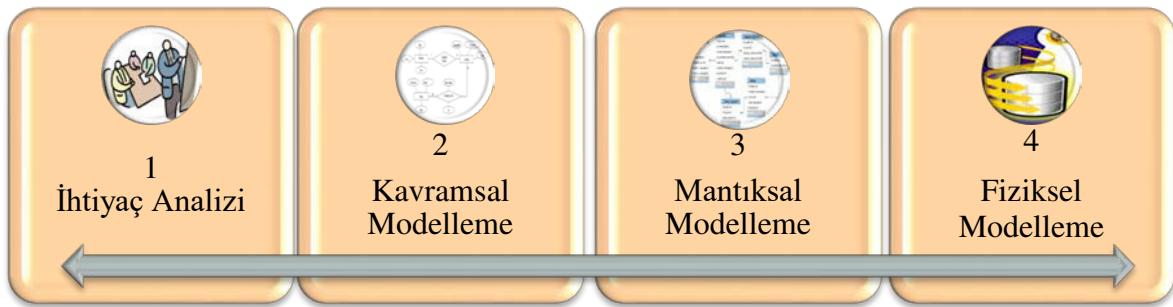
Genel olarak Veri Tabanı Sistemi Geliştirme 3 Aşamadan oluşmaktadır.

1. Gereksinim analizi Aşaması: Bu aşamada bir veri modeli geliştirilir. Veri modeli veri tabanı yapısının mantıksal gösterimidir
2. Tasarım Aşaması: Veri modeli tablolara ve ilişkilere dönüştürülür.
3. Kurulum Aşaması: Tablolar, ilişkiler ve sınırlılıklar oluşturulur. Saklı yordamlar ve tetikleyiciler yazılır. Veri tabanına veri girilir ve sistem denenir. Veri tabanı ve uygulamaları(aynı üç aşamayı kullanarak) yeni gereksinimleri karşılamak için değiştirilir

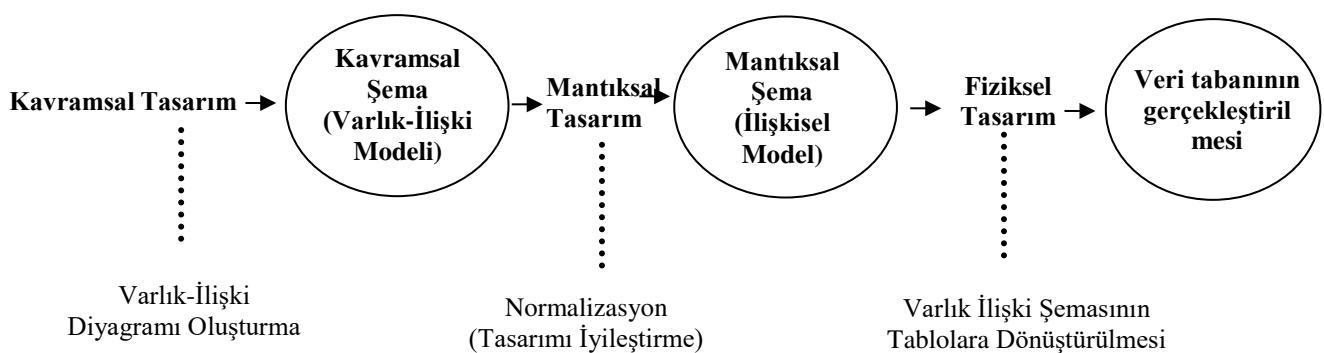
Bu kısımda, yeni bir veritabanı tasarımları ve bir veritabanının yönetimi ve organizasyonu ile ilgili gereksinimleri ve gerekli faaliyetleri ele alacağız.

Bir veri tabanı tasarlanırken genellikle şu süreçler ele alınır:

1. Gereksinim Analizi
2. Veri Tabanı tasarımları
 - Kavramsal Tasarım
 - Mantıksal Tasarım
 - Fiziksel Tasarım
3. Veri Tabanının Tamamlanması (uygulama programları, testler, dokümantasyon)
4. Veri Tabanının İşletilmesi, Test Edilmesi ve Bakımı



Şekil 1 Veri Tabanı Geliştirme Aşamaları



Şekil 2

2.2.1 İhtiyaç Analizi

Veri tabanı tasarlama başlamadan önce analizinin doğru yapılması gerekmektedir. Veri tabanı ihtiyaç analizi yapılrken hazırlanacak olan sistemin neye hizmet edeceğini, veri tabanını ne iş yapacağı ve hangi ihtiyaçları karşılayacağına, veri tabanının hangi verileri depolayacağı, veri tabanını oluşturan tabloların neler olacağı ve ne tür verileri saklayacağı vb. gibi sorulara cevap vermek gerekmektedir. Tüm bunları kâğıt üzerinde tasarladıkta sonra fiziksel tasarıma geçmek çalışmanızın daha sistemli yürütmesi açısından avantajınıza olacaktır. Veri tabanı kullanılması istenilen uygulama için gereksinimler tanımlanır. Ne tür bilgiler saklanacak? Örneğin öğrenci bilgileri, ders bilgileri, hoca bilgileri vb. Bu bilgiler arasındaki ilişkiler ne şekilde olacak? Örneğin bir derse en çok kaç hoca girebilir, bir öğrenci bir dönemde en fazla kaç kredilik ders seçebilir. Bir sistemin analizi yapılmış ve bir veri tabanı kurulması önerilmiş ise, önce veri tabanından beklenen amaçların ve işlevlerin, donanım-yazılım ve personel gereksinimlerinin, kullanılacak veri miktarı ve ilişkilerin saptanması gerekmektedir. Bu bilgiler gereksinim analizi yolu ile elde edilir. Sistemin inceleme ve analiz aşamasında belirlenen gereksinimler, bir veri tabanı geliştirme ekibi tarafından ayrıntılı incelenir. Veri tabanı uygulamalarından yararlananlar kullanıcılardır. Bu nedenle veri tabanı gereksinim analizinde kullanıcıların görüşlerine de başvurulmalıdır. Özellikle girdiler-çıktılar ve işlem sınırlamaları bakımından gereksinimleri sorulmalıdır. Veri tabanı geliştirme ekibi, formlar, raporlar ve menüler (seçenekler) için örnekler hazırlayarak, bunlar üzerinde kullanıcıların görüşlerini almalıdır. Gereksinim analizi sonunda belirlenen gereksinimler; metin, veri akış

diyagramları, varlık-ilişki diyagramları, nesne diyagramları şeklinde olabilmektedir. Gereksinim belgeleri yeniden gözden geçirilip gerekli düzeltmeler yapıldıktan sonra kullanıcıların ve proje sahibinin onayına sunulur.

Verilerin Değerlendirilmesi

Kullanıcıların ve proje sahibinin anlayabileceği şekilde hazırlanan gereksinim modeli bu hali ile veri tabanında yer almaz. Çok kullanıcılı sistemlerde bu gereksinimler çelişkili ve tutarsız da olabilmektedir. Bu nedenle gereksinim modeli, veri tabanı uzmanları tarafından yeniden yorumlanmakta, veri tabanı tasarımda kullanılabilecek bir modele yani Kavramsal Modele dönüştürülür. Hata ve eksiklerin değerlendirme aşamasında bulunup düzeltilmesi kolay ve ucuzdur. Oysa daha sonraki aşamalarda fark edilmesi halinde geriye dönülmesi büyük iş kaybına ve gidere yol açmaktadır. Bu aşamanın sonunda; sistem mimarisi, bir dizi yapılabılır ve maliyeti uygun gereksinimler ve kavramsal veri modeli elde edilir. Bu sonuçlar onaylaması için proje sahibine sunulur ve tasarım aşamasına girdi olarak aktarılır. Değişik uygulama sistemi yapılarının tanımlanması ve içlerinden birinin seçimi yapılır. Uygulamanın yapılabılırlığını (fizibilitesi) yeniden belirlenir. Seçilen çözümde bütün gereksinimlerin yer aldığıının denetlenmesi ve eksikler var ise, ilerde eklenmesi yada elenmesi gereği karşılaştırılır.

2.2.2 Veritabanı Tasarımı

Bir veritabanı oluşturmak için üç tasarım aşamasından geçilmelidir:

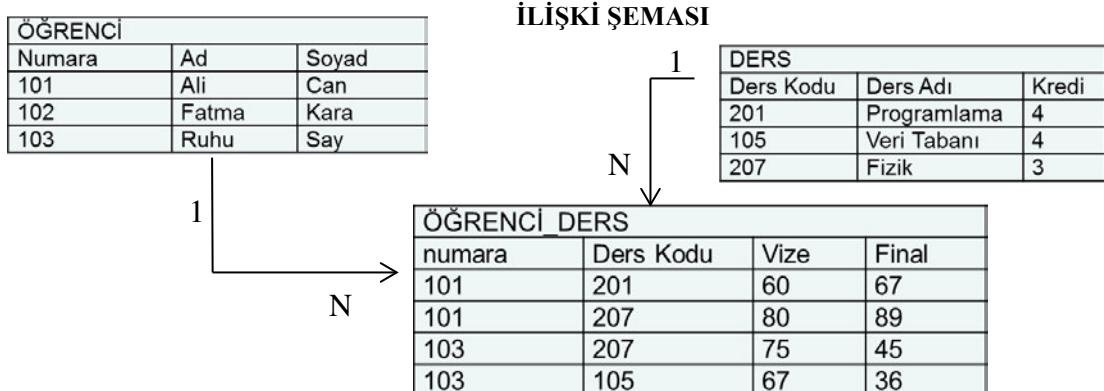
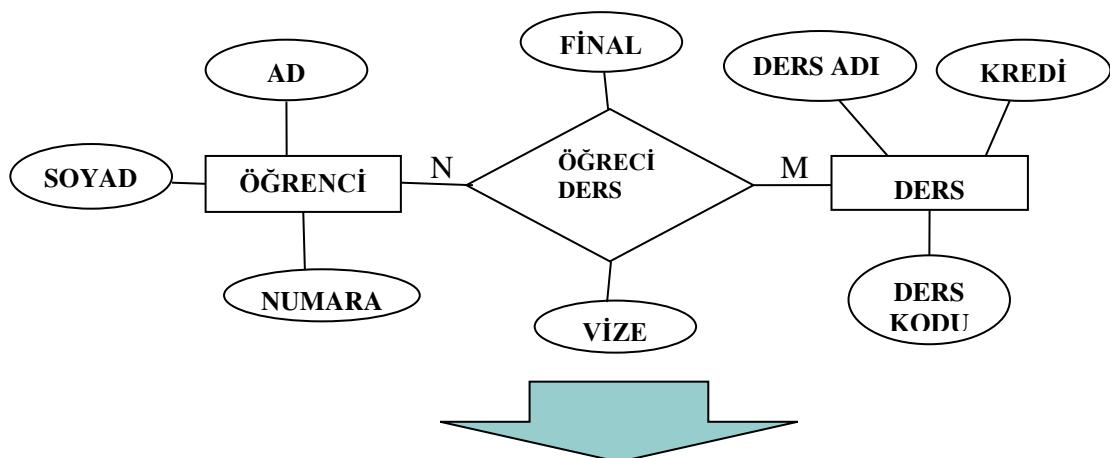
- 1. Kavramsal tasarım**
- 2. Mantıksal tasarım ve**
- 3. Fiziksel tasarım.**

Kavramsal tasarım veritabanının işletme perspektifinden görülen soyut halidir. Mantıksal tasarım, veritabanının işletmedeki son kullanıcıların bilgi gereksinimlerinin detaylı bir tanımlanmasını gerektirir. İdeal durumda, veritabanı tasarımını tüm organizasyon çapındaki veri planlama çabalarının bir parçası olmalıdır. Fiziksel tasarım ise veritabanının doğrudan erişimli saklama cihazlarında nasıl düzenlendiğini gösterir.

2.2.2.1 Kavramsal Tasarım

Gereksinim analizinin sonuçları göz önünde bulundurularak, saklanması istenilen verileri saklayacak veri tabanının kavramsal modeli geliştirilir. Gereksinim Analizi sırasında keşfedilen bilgileri bütünlüğe getirerek oluşturulan Kavramsal Modelin öğeleri Varlık-İlişki Diyagramları ile gösterilir. Varlık İlişki diyagramları öğelerin anlamlarını ve birbiriyle ilişkilerini belirli bir veri tabanı sisteminden ya da kurulum ayrıntılarından bağımsız olarak gösterir.

Varlık-İlişki Şeması



Şekil 3: Kavramsal Tasarımdan Mantıksal Tasarıma Geçiş

Kavramsal veritabanı tasarımlı, veri elemanlarının veritabanı içinde nasıl gruplanacağını tanımlar. Tasarım sürecinde, veri elemanları arasındaki ilişkiler ve bilgi gereksinimlerini karşılayabilmek için bu veri elemanlarını bir arada grupperleme en etkin yolu belirlenir. Bu süreç aynı zamanda gereksiz (tekrarlanan) veri elemanlarını ve spesifik uygulama programları için istenen veri elemanlarını da tanımlar. Veri tabanındaki veri elemanları arasında genel bir mantıksal görüntü elde edilinceye dek, bu veri grupları organize edilir, rafine edilir ve verimlilik düzeyleri artırılır. Kavramsal tasarım aşamasında tüm uygulamaların ortak gereksinimleri birleştirilir. Hangi Varlıkların kullanılacağı kararlaştırılır. Varlık, hakkında bilgi tutulan nesnedir. Varlıkların Özellikleri belirlenir. Varlığı tekil olarak belirleyen anahtar özellikler de belirlenir. Varlıklar arasındaki ilişkileri belirlenir. Varlıklar birbirileyle nasıl etkileşiyor? İlişkilerin Özellikleri belirlenir.

2.2.2.2 Mantıksal Tasarım

Gereksinim analizi ve değerlendirilmesi aşamasında, varlıkların ve aralarındaki ilişkilerin belirlenmesi ve tanımlanması işlemi varlık-ilişki modeli ile kavramsal olarak hazırlanmaktadır. Mantıksal Tasarım aşamasında bu yapı bilgisayara yönelik özel bir veri yapısına yani mantıksal modele dönüştürülmektedir. Mantıksal model temelde normalizasyon

işlemi üzerine kuruludur. Gereksiz bilgi tekrarını, bilginin kaybını veya yetersizliğini önlemek için ayrıca bir de normalleştirme işlemi uygulanarak ilişkiler normal forma getirilmelidir.

Mantıksal modelin kurulmasından önce düzenlenmiş olan kavramsal model için en uygun veri modelinin belirlenmesi gereklidir. Bu aşamada şu modellerden (hiyerarşik veri modeli, ağ veri modeli, ilişkisel veri modeli ve nesne tabanlı veri modeli) hangisi kullanılacaksa, seçilen modele uygun olarak mantıksal model geliştirilir. Bu derste ilişkisel veri modeli temel alındığı için bunun üzerine mantıksal tasarım geliştirilir.

2.2.2.3 Fiziksel Tasarım

Fiziksel tasarımın amacı en az giderle en uygun performansın sağlanmasıdır.

Fiziksel tasarımın dört temel aşaması vardır:

1. Veri Gösteriminin Belirlenmesi
2. Erişim Yöntemlerinin Seçimi
3. Verinin Dış Belleklere Atanması
4. Veri Tabanının Yüklenmesi ve Tekrar Düzenlenmesi

2.2.2.3.1 Veri Gösteriminin Belirlenmesi

Mantıksal tasarım sırasında şema oluşturulurken, aynı zamanda veri öğesinin tipi ve uzunluğu gibi fiziksel özellikler de kararlaştırılmalıdır. Ayrıca, fiziksel kütüklerin boyutları ya da veri setleri saptanmalıdır. Veri tabanı tasarımcısı, veri tanımlama dili (DDL) kullanarak veri gösterimini ayrıntılı olarak belgelendirir.

Veritabanı Büyüklüğü Hesaplaması

Örneğin MS SQL Serverda veriler disk üzerinde sayfa (page) şeklinde tutulur. Her sayfa 8 KB lik bloklar şeklinde depolanır. Örneğin 1MB lik bir database de bilgiler; $1024/8 = 128$ sayfa olarak tutulacaktır $128*8=1024$ KB (1MB). Eğer bir tabloda birden fazla sütun varsa toplam satır boyutunu hesaplarken 9 Byte 'lik ilave yapmak gereklidir (row overhead). Satırlar (rows) sürekli, bölünmeyen yapıdadır ve tek satırda maximum miktar 8096 Byte (8KB) dır.

Örnek: Aşağıdaki gibi tanımlanmış bir tabloda 250.000 adet kayıt vardır. Buna göre tablonun databese içerisinde kapladığı alanı bulalım:

Adı	Soy	Tel	CepTEL	Dtarihi	Vize	Final
30 byte	30 byte	30 byte	30 byte	8 byte	4 byte	4 byte

create table ogrenci (adi char(30), soy char(30), tel cahr(30), ceptel char(30), dtarihi DateTime, vize int, final int)

1) Toplam Satır Boyutu = $4*30 + 1*8 + 2*4 + 9 = 147$ byte

2) Her Sayfadaki Satır Sayısı = $8096/147 = 55$ satır

3) Tablo İçindeki Sayfa Sayısı = $250.000 / 55 = 4545$ sayfa (page)

4) Tablo Boyutu = $4545 * 8 \text{ KB} = 36.363 \text{ KB}$ veya 36 MB

Hesaplanan değer, Indexleme yapılan sütuna göre %25 ile %50 oranında artacaktır.

2.2.2.3.2 Erişim Yöntemlerinin Seçimi

Erişim yöntemleri, VTYŞ'ne bağlı olduğu için belirlidir. Yine de veri tabanındaki her bir kayda erişilecek yol saptanmalıdır. Veri tabanına kendi anahtarı ile doğrudan erişebilen kayıt tipleri, başka kayıtların göstergeleri yada indeksler aracılığı ile girilebilenlerden ayrı edilmelidir.

2.2.2.3.3 Verinin Dış Belleklere Atanması

Erişim yöntemleri ile tanımlanmış olan her kaydın ve kütüğün fiziksel araçlarda, yani dış belleklerde saklanacağı yerin belirlenmesi gereklidir. Atamada çok kullanılan veriye öncelik verilmesi yada en büyük olasılığı sağlayan veri düzeninin bellekte bir araya getirilmesi (kümeleme-clustering) performansı arttırmır.

2.2.2.3.4 Veritabanının Yüklenmesi

Mantıksal Tasarım sonucunda elde edilen tablolar veri tabanı üzerinde oluşturulur. Yükleme, bir veri tabanı yükleme programı (SQL script) oluşturularak, yada yardımcı bir program kullanılarak gerçekleştirilir. Tablolar görsel bir arayüz üzerinden oluşturulabileceği gibi, Veri Tanımlama Dili komutlarından olan CREATE TABLE komutu ile de oluşturulabilir.

Bu aşamada; mantıksal düzeyde yapılan düzenlemeler oluşturulan yapılar ve her yapıda hangi verilerin yer aldığı, her verinin türü, uzunluğu, varsa varsayılan değeri ve diğer özellikleri, veriler arası ilişkiler ve her türlü kısıtlamalar, fiziksel veri yapıları ile ilgili tercihler ve parametreler ve kullanıcı tanımları ve kullanıcıların hangi veriler üzerinde hangi işlemleri yapmaya yetkili olduklarına ilişkin tanımlar.

2.2.2.3.5 Veritabanının Tekrar Düzenlenmesi

Tekrar düzenlenmemeyi gerektirebilecek nedenler: Yeni veri öğelerinin veya kayıt tiplerinin ortaya çıkması (mantıksal tasarım da tekrar düzenlenmeli), Yeni işlemlerin gereklmesi (sadece fiziksel tasarım tekrar düzenlenir) ve Veri tabanını değiştirmek yoluyla işlem etkinliğinin arttırılması (sadece fiziksel tasarım tekrar düzenlenir).

2.3 Veritabanının Tamamlanması

Veri tabanının tamamlanması aslında uygulama programlarının tamamlanmasını ifade eder. Uygulama programları aslında mantıksal model tasarlanırken tasarılanırlar. Bu aşamada kullanıcı arabirimleri de (form yapıları, raporlar) tasarılanarak program tamamlanır. Veri tabanının ve uygulama programlarının doğruluğunun, bütünlüğünün ve performansının test edilmesi gerçekleştirilir. Dokümantasyonun hazırlanması tamamlanır. Tüm uygulamalar tam-ölçekli çalıştırılır. Gizlilik, güvenlik ve erişim kontrolü sağlanır. Kurtarma ve yedekleme prosedürleri kurularak ve kullanılmaya başlanır.

2.4 Sistemin Test Edilmesi ve Bakımı

Veri tabanı kurulduktan sonra bazı testlerden geçirilerek doğru ve geçerli işlemler yapıp yapmadığına bakılır. Hatalı noktalar düzelttilir. Veri tabanı kullanımına açıldıktan sonra sürekli olarak başarımı kontrol edilir. Beklenenden yavaş çalışması yada doğru çalışmaması durumlarında düzeltme işlemleri gerçekleştirilir. Veri tabanının sık sık yedeği alınır (backup), disk arızası gibi durumlarda yedekten geri yüklenir (recovery).

2.5 Veri Tabanının İşletilmesi ve Yönetilmesi

Her sistem gibi veri tabanı sistemi de ne kadar mükemmel tasarılmış olursa olsun, kötü işletilmesi halinde sorunlar ortaya çıkabilemektedir. Aşağıdaki durumlarda uygulanacak yöntemler mutlaka belirlenmelidir: Normal ve anormal durumlarda sistemin kapatılması, başarısızlık halinde kurtarma ve Veri tabanının yedeklenmesi ve onarılması. Bilgisayar sisteminde dosyalar dış ortamlarda saklanır. Bunlar sabit disk ya da teyp olabilir. Sabit disklere direkt erişim mevcuttur. Yani verinin sabit diskteki yeri bilinirse, buradan dosyalar direkt olarak alınabilir. Teyp gibi varlıklar ise bilgiye erişim sıralı olarak ifade edilebilir. Yani veri bulununcaya kadar teyp üzerinde arama yapılır.

Veri tabanı sistemlerde direkt erişimli disk sistemi kullanılır. Dış ortamlara veri giriş/çıkışı bilgisayar işletim sisteminin görevidir. İşletim sisteminin programlar ile donanım arasında ilişkiyi sağladığı için hafıza yönetim ve girdi/çıktı ilişkilerini de işletim sistemi sağlamaktadır. Veri tabanı yönetim sistemi de bir program olduğu düşünülürse, yukarıda anlatılan işlerin veri tabanı yönetim sistemine işletim sistemi tarafından sağlanacağı anlaşıılır. O halde işletim sisteminin bir veritabanı yönetim sisteme sağladığı en önemli özellik dosya yönetim sistemidir. Bilgisayar diske veri kaydettiği zaman, bu yazma karakterlerinin yazılması şeklinde gerçekleşmez. Bilgisayar bu veriyi diske bloklar şeklinde yazar. Genellikle bu blokların büyüklükleri 2-4 kilobyte'dir. Blok büyülüğu 4 kilobyte olarak varsayılsa 4 kilobyte'lik bir veri bilgisayar hafızasından (genellikle RAM) diske (genellikle Harddisk) kaydedilecektir. 96 kilobyte'lik bir dosya 24 blokluk bir yere ihtiyaç duyar. Dosya yöneticisi, disk yöneticisine blok ihtiyacını bildirir. Bu sayfalara erişim disk yöneticisine aittir.

2.5.1 Veritabanı Yöneticisi

Cök kullanıcılı veri tabanları veri tabanının performansını arttıran veri tabanının bütünlüğünü sağlayan bir veri tabanı yöneticisine ihtiyaç duyarlar. Yönetici kullanıcıların özelliklerini ve yetkilerini belirler. Bunun amacı sınırlı veri tabanı kaynaklarının verimli olarak kullanılmasıdır. Veri tabanı sisteminin devamının sağlanması için veri tabanı yönetim sisteminin ve veri tabanı yöneticisinin yapması gereken bazı özel işlemler vardır. Bunlar; Güncelleme, Yedekleme ve Kurtarma olarak sıralanabilir.

2.5.2 Veritabanı Güvenliği

Verinin güvenli bir ortamda saklanması gerekmektedir. Bu yüzden bir işletmenin veritabanı stratejik bir öneme sahiptir ve veriler güvenli bir şekilde muhafaza edilmeli ve gizli tutulmalıdır. Veri Tabanı İşletim Sistemine erişim güvenlik açısından son derece önemlidir. Veritabanına sadece izin verilen kullanıcılar erişebilmelidir. Güvenlik kavramı, yetkisi olmayan kişilerin veritabanına bilerek ya da bilmeyerek yasal olmayan erişimini önlemektir.

Veri Tabanı Güvenliği iki şekilde incelenebilir: Tabi afetlere karşı olan güvenlik ve Yetkisi olmayan kişilere karşı güvenlik. Hırsızlık, deprem, yangın ve sel gibi durumlarda, veri tabanı ikinci bir bilgisayar kullanılarak yedeklenir. Bu bilgisayara gölge bilgisayar denir ve farklı bir yere yerleştirilir. İlk bilgisayar zarar görürse bilgisayar aktif hale gelir. Yetkisi olmayan şahıslardan veritabanını korumada ise, şahısların hangi verilere ulaşabileceği, hangi verileri silebileceği, değiştirebileceği ve veri kaydı yapabileceği belirlenmesi verilerin gereklidir. Bu işlem ise veri tabanı sisteminin kullanıcıları kontrol etmesiyle mümkündür. Veri tabanlığı güvenliğinde veri tabanı yöneticisi sorumludur. Veri tabanı güvenliği bilgisayarlı ve bilgisayarsız kontrol metodlarını kullanarak bilerek ya da bilmeyerek yapılan tehditlere karşı veri tabanının güvenliğinin sağlanmasıdır. Güvenlik önlemleri sadece veritabanı için düşünülmemelidir. Veri tabanının çevresi de önemlidir, veritabanı güvenliği yazılım, donanım ve çevredeki kişileri de kapsamalıdır.

Bir veritabanının güvenliğinde şu unsurlar dikkate alınmalıdır. Bu alanlarda risk azaltılmalıdır.

Hırsızlık ve Sahtekârlık: Çalışan maaşları ve buna benzer tipteki veriler gizli tutulmalıdır. Çalışan tarafından bu bilgilere erişim imkanı olabilir ve bazı bilgiler değiştirilebilir.

Gizliliğin İhlali: Kişisel ve kurumlara ait veriler gizli tutulmalıdır. Kurumlara ait bilgiler yetkisi olmayan kişiler tarafından ortaya çıkarılırsa, bu durum işletmenin rekabet gücünün azalmasına yol açabilir. Kişilere ait verilerin yetkisi olmayan kişiler tarafından ortaya çıkarılması durumunda ise hukuki ihtilaflar söz konusu olabilir.

Veri Bütünlüğünün Kaybolması: Veri bütünlüğün kaybı verinin geçersiz hale gelmesine veya bozulmasına yol açabilir. Veri tabanı yönetim sistemine herhangi bir sebepten

dolayı ulaşılamaması durumunda yine sistemdeki verilerin bir kısmının kaybolması söz konusu olabilir.

Veri tabanı güvenliğini etkileyen olaylar şunlardan oluşabilir:

- Veriye erişim hakkı olan kişilerin yetkilerini kullanarak yasal olmayan şekilde verilere ulaşmaları.
- Yetkisi olmayan kişilerin veriyi kopyalamaları ya da değişiklik yapmaları.
- Veri tabanına yasal olmayan girişler.

2.5.3 Veri Güncelleme

Veri tabanı yönetim sistemi çok kullanıcılı olarak birçok kullanıcıya hizmet etmelidir. Birçok kullanıcı istediği verilere erişebilmeli ve istediği bilgileri güncelleştirebilmelidir. Bu süreç içinde veri tabanı bilgileri paylaştırma ve kilitleme mekanizmalarını çalıştırabilmelidir.

- Bir veri tabanı yönetim sisteminin yapması gerekenler:
- Veri saklama, erişme ve güncelleştirme: VTYS kullanıcılarının verileri saklamasını, onlara erişebilmesini ve güncelleştirebilmesini sağlamalıdır.
- Bir grup işlemi yapabilme: veri tabanında bir grup işlem yapılabilmeli
- Aynı anda işlem yapabilme: veri tabanı aynı anda birçok kullanıcı tarafından aynı anda kullanılabilмелidir.

2.5.4 Veritabanı Yedekleme

Veri tabanı yönetim sistemi verilerin yedeğinin alınması gibi gerekli işlemleri yapmalıdır. Kullanıcılar kontrol edilmeli ve sisteme zarar vermemelidirler. Bununla birlikte bozulan bilgilerin veya herhangi bir nedenden dolayı ulaşılamayan veri tabanlarının düzeltilebilmesi gibi kurtarma hizmetleri verebilmelidir. Bu kurtarma işlemlerini yapabilmek için düzenli olarak veritabanın yedeğinin alınması gerekmektedir.

2.5.5 Veritabanı Kurtarma

Sistemin çökmesi durumunda veri tabanını tekrar çalışır hale getirilmesi işlemine veri tabanı kurtarma işlemi denir. Bu işlem için, log dosyaları kullanılır. Bu dosyalarda veri tabanında yapılan bütün işlemler kaydedilmiştir. Veri tabanı sistemi çöktüğü zaman bu log dosyalardaki işlem bilgileriyle veri tabanı kurtarılmaya çalışılır. Yönetici belirli aralıklarla veri tabanını arşiv dosyası olarak saklamak zorundadır. Sistem çöktüğü zaman log dosyalardaki işlemler listesi, arşiv dosyası üzerinde uygulanır. Böylece veri tabanı kurtarılmış ve güncelleştirilmiş olur. Veri tabanını kurtarmak için diğer bir metot ise kontrol noktasıdır. Burada veri tabanını son hali ele alınır. Log dosyasındaki işlemler listesi kullanılarak kontrol

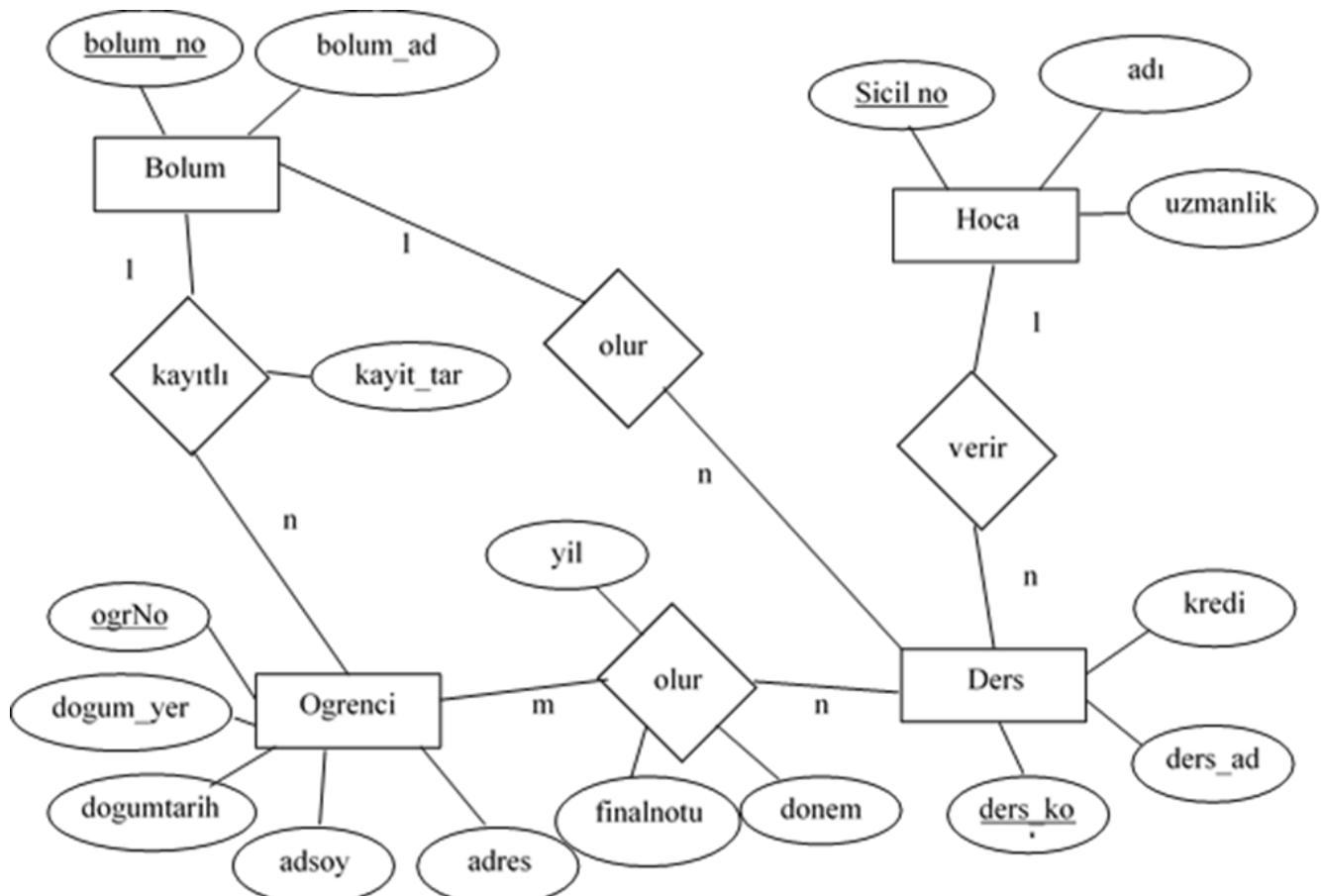
noktasına kadar geri getirilir. Veri tabanı yöneticisi, en verimli en ekonomik kurtarma metodunu belirlemek zorundadır.

Uygulamalar

Kavramsal Tasarım Aşamasında Da Kullanılan Varlık-İlişki Şeması

Kullanıcıdan elde edilen gereksinimler ile ilgili bir analiz çalışmasının yapılması ve birbiriyle bağıntılı verilerin gruplanarak bir düzenleme içinde modellenmesi gereklidir. Bu modeli grafiksel olarak Varlık-İlişki diyagramları ile gösterilir. Varlık İlişki diyagramları öğelerin anımlarını ve birbiriyle ilişkilerini belirli bir veri tabanı sisteminden ya da kurulum ayrıntılarından bağımsız olarak gösterir. Varlıkların ve aralarındaki ilişkilerin belirlenmesi ve tanımlanması işlemi varlık-ilşki modeli ile kavramsal olarak hazırlanmaktadır.

Aşağıda örnek bir Varlık-İlişki Şeması gözükmemektedir.



Uygulama Soruları

Soru-1:Bir avukatlık bürosu için veri tabanı tasarımlı nasıl yapılır? Gerekli çalışmaları yaparak veri tabanını geliştiniz.

Soru-2:Kütüphane veri tabanının kavramsal tasarımlı.

Veri Tabanının İhtiyaç ve İstekleri

- 1.** Veri tabanı birden fazla kütüphane, kütüphanelerin içерdiği kitaplar ve üyeleri temeline dayanacaktır.
- 2.** Kütüphanelerin adres ve isim bilgileri tutulmaktadır ve her kütüphane benzersiz bir numaraya sahip olmalıdır.
 - Kütüphanelerin adres bilgileri ayrı olarak tutulacaktır.
 - Kütüphanede bir yazarın birden fazla kitabı olabilir.
 - Kitapların ISBN numarası, başlık ve yayın bilgileri tutulur.
- 3.** ISBN numarası ile takip edilecektir.
 - Aynı ISBN numarasına sahip birden fazla kitap olamaz miktar olarak olabilir.
 - Kitaplar kategorilere ayrılmıştır. ISBNolar ile kategorilere atanmıştır. Bir kitap birden fazla kategoride yer alabilir.
 - Bir kitabın birden fazla yazarı olabilir ve yazarla ayrı varlık olarak tanımlanmalıdır.
- 4.** Yazarların ad ve soyad bilgisi tutulmaktadır ve her yazar benzersiz bir numara ile takip edilmektedir.
- 5.** Üyelerin ad, soyad, telefon, e-posta ve adres bilgileri tutulmaktadır. Her üye benzersiz bir numaraya sahiptir.
 - Üye kütüphaneden bağımsızdır.
 - Üye istediği kütüphaneden kitabı alabilir.
 - Üye adres bilgisi de ADRESLER içinde adres numarası ile kullanılmaktadır.
 - Üyelerin hangi kütüphaneden hangi kitabı ne zaman aldığı ve ne zaman teslim ettiği bilgileri tutulmaktadır. İşlem takibi için benzersiz olan emanet no kullanılır.

Veri Tabanında Olması Gereken Varlıklar

- | | |
|-------------|---------------|
| – KÜTÜPHANE | – YAZARLAR |
| – KİTAPLAR | – ADRESLER |
| – ÜYELER | – KATEGORİLER |

Bu Bölümde Ne Öğrendik Özeti

Bu derste Veri Tabanı Tasarımı konusu üzerinde duruldu. Veritabanı tasarımlı süreci ve önemi vurgulanarak, tasarım araçlarından yararlanılarak baştan sona bir veri tabanının nasıl tasarlanacağı sergilendi. Daha sonra Veritabanı uygulamalarının somut kavramlarını kavrayıp bunları gerçek veritabanları geliştirme sürecinde nasıl kullanılacağı üzerinde duruldu.

Bölüm Soruları

1. Mantıksal model temelde aşağıdaki çalışmalardan hangisi üzerinde durur?

- a) Varlık ilişki diyagramlarının hazırlanması
- b) İhtiyaç analizi üzerinde odaklanır.
- c) Veri akış diyagramlarının hazırlanması üzerinde durur.
- d) Veri tabanının SQL komutlarının hazırlanması üzerinde durur.
- e) Normalizasyon işlemi üzerine kuruludur.

2. Aşağıdakilerden hangisi Fiziksel Tasarım aşamasının temel işlerinden biri değildir?

- a) Verilerin Değerlendirilmesi
- b) Veri Gösteriminin Belirlenmesi
- c) Erişim Yöntemlerinin Seçimi
- d) Verinin Dış Belleklere Atanması
- e) Veri Tabanının Yüklenmesi ve Tekrar Düzenlenmesi

3. Varlıkların belirlenmesi veri tabanı geliştirme sürecinin hangi aşamasında gerçekleştirilir?

- a) Gereksinim Analizi
- b) Kavramsal Tasarım
- c) Mantıksal Tasarım
- d) Fiziksel Tasarım
- e) Testler ve Dokümantasyon

YANITLAR:

1-E, 2-A, 3-B

3. VERİ MODELİ

Bu Bölümde Neler Öğreneceğiz?

- 3.1. Veri Modellerinin Sınıflandırılması
- 3.2. Veri Modelleri: Kısa Tarihçe
- 3.3. Veri Modeline Göre Veri Tabanı Yapıları
 - 3.3.1. Hiyerarşik Veritabanları
 - 3.3.2. Ağ Veritabanları
 - 3.3.3. İlişkisel Veritabanı Sistemi
 - 3.3.4. Nesneye Yönelik Veritabanları
- 3.4 Varlık İlişki Diyagramları
 - 3.4.1. Varlık-İlişki Diyagramlarının Bileşenleri
 - 3.4.1.1 Varlık ve Özellikleri
 - 3.4.1.2 Özellikler (Attributes)
 - 3.4.1.3 İlişkiler
 - 3.5 Varlık-İlişki Şemasının Tablolara Dönüşürtülmesi
 - 3.5.1 Bire-Bir İlişkilerin Dönüşürtülmesi
 - 3.5.2 Bire-Çok (1-n) İlişkilerin Tablolara Dönüşürtülmesi
 - 3.5.3 Çoka-Çok (N-M) İlişkilerin Tablolara Dönüşürtülmesi
 - 3.6 Çok Değerli Özelliklerin Dönüşürtülmesi

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Veri modeli nedir?
- 2)** Veri Modeline Göre Veri Tabanı Yapıları nelerdir?
- 3)** Varlık-İlişki şeması nasıl oluşturulur?
- 4)** Aşağıda verilen tablolardan oluşan KUTUPHANE veritabanının Varlık-İlişki (E-R) diyagramını oluşturunuz.

KISI (TC Kimlik No, Adı, Soyadı, Adresi, Telefonu)

KATEGORI (Kategori No, Kategori Adı)

YAZAR (Yazar No, Yazar Adı, Yazar Soyadı)

YAYINEVİ (Yayinevi No, Yayinevi Adı)

KİTAP (Kitap No, Kitap Adı, Sayfa Sayısı, Yayinevi No, Kategori No)

YAZAR_KITAP(Yazar No, Kitap No)

ÖDÜNC (TC Kimlik No, Kitap No, Alış Tarihi, İade Tarihi, Ceza)

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Veri modelini bilir.	Anlatım, Soru-Cevap, Tartışma
	Varlık- ilişki şeması hazırlayabilir.	Alıştırma ve Uygulama, Örnek Olay
	Varlık ilişki şemasından tablo tasarımını yapabilir.	Bireysel Çalışma, Problem Çözme
		Proje Temelli Öğrenme

Anahtar Kavramlar

- Veri Modeli
- Varlık- İlişki Şeması

Giriş

Model bir nesnenin, bir sistemin veya bir fikrin temsilidir. İnsana ait düşünceleri, açık ve analiz edilebilir sunumlara dönüştürme sürecine ise modelleme denilir (Olson ve Courtney, 1992). Webster sözlüğünde model, doğrudan gözlemleyip deneyemediğimiz bir şeyi anlamak için yapılan analogi olarak tanımlanır. Model kelimesi; isim, sıfat ve fiil olarak ve her birinde oldukça farklı çağrımlar yapacak şekilde kullanılmaktadır. İsim olarak “model”, bir temsili ifade eder. Bu temsil; bir mimarın, bir binanın küçük ölçekli modeli veya bir fizikçinin bir atomun büyük ölçekli modelini oluşturmaması anlamındadır. Sıfat olarak “model”, mükemmeliyetin veya idealin ölçüsünü ifade eder. “Model ev”, “model öğrenci” ve “model eş” ifadelerinde olduğu gibi. Fiil olarak “model” ise, bir şeyin nasıl olduğunu ispat etmek, açıklamak, göstermek anlamındadır.

Bilimsel modeller bütün bu çağrımları bünyelerinde bulundururlar. Onlar; durumların, nesnelerin ve olayların temsilleridir. Gerçeklerden daha az karmaşık ve böylece araştırma amacıyla kullanılmaları daha kolay olduğundan, bu anlamda ideal hale getirilmişlerdir. Gerçek durumlarla karşılaşıldıklarında, modellerin basitliğinin sebebi, gerçeklerin sadece uygun özelliklerini temsil etmelerinden kaynaklanmaktadır. Örneğin, yeryüzünün bir kısmının modeli olan bir yol haritasında, bitki örtüsü gösterilmmez. Çünkü bu durum, o haritanın bir yol haritası olarak kullanımı açısından uygun değildir. Güneş sisteminin bir modelinde, gezegenleri temsil eden topların, gezegenlerle aynı maddeden yapılmış olmaları veya aynı sıcaklığa sahip olmalarına ihtiyaç yoktur.

Bilimsel modellerden, gerçeklerin farklı boyutlarılarındaki bilgiyi artırmak ve birbirleri ile ilişkilendirmek için faydalıdır. Modeller, gerçeği ortaya çıkarmak ve bundan daha fazla olarak, geçmiş ve şimdiki durumu açıklamak ve geleceği tahmin ve kontrol etmek için kullanılır. Modeller uygulanarak, gerçekler üzerinde bilimin kontrolü sağlanır. Modeller gerçeğin tarifi ve açıklamasıdır. Bir bilimsel model, aslında, gerçek hakkında bir veya bir dizi ifadelerdir. Bu ifadeler oylara dayanan, kanun benzeri ya da teorik olabilir.

Bir bilişim sistemi modeli, gerçek bilgi kümесinin alt kümесini oluşturur ve onun daha basit bir şeklidir. Bu şekil, işlenebilmeye imkân verir ve bunu kullanarak elde edilen çözüm veya cevap, gerçek hayatı uygulanmaya çalışılır. Model, var olan bilgi yiğinına bir düzen getirmeyi, hatta bir yapı oluşturmayı amaçlar. Tek bir model yoktur. Var olan bilgi yiğinına, uygulanan farklı modeller doğal olarak farklı yorumlar getirir.

Mühendisler bir sistemi tasarlarken;

Tasarladıkları sisteme ait bir model geliştirirler

Model üzerinde denemeler yaparlar

Model'in farklı durumlarda davranışlarını incelerler

Böylece problemi kolayca çözülebileceği bir yapıya oturtmaya çalışırlar.

Temel modelleme araçları iki gereksinimi karşılamalıdır:

1) İfade Edebilme Gücü: Gerçek dünyadaki durumları tanımlayabilecek şekilde genel bir yapıda olmalı.

2) Kullanışlılık: Kolay kullanılabilen ve yönetilebilen bir yapıda olmalıdır. Mühendislik sistemlerinin tasarımında olduğu gibi veri tabanı tasarımında da modele başvurulur. Bu amaçla geliştirilen model bir veri modelidir.

3. VERİ MODELİ

Bir veri tabanı yapısının temelini veri modeli kavramı oluşturmaktadır. Veriyi mantıksal düzeyde düzenlemek için kullanılan kavramlar, yapılar ve işlemler topluluğuna veri modeli adını veriyoruz. Veri modeli, bir veritabanının yapısını açıklayan kavramlar bütünüdür. Veritabanı yapısı ise; veri türleri, bağıntılar ve veri kısıtlamalarından oluşur. Verileri mantıksal düzeyde düzenlemek için kullanılan yapılar, kavramlar ve işlemler topluluğuna veri modeli (data model) denir. Bir veri tabanı yönetim sistemi, bir veri modeli üzerine inşa edilir. Veri modelleri, veri tabanındaki ilişkileri soyut bir şekilde canlandırdığından kavramsal düzeyde oluşturulur. Veri tabanının oluşturulması amacıyla geçmişten günümüze kadar bir takım farklılıklarla çeşitli veri modelleri geliştirilmiştir. Her biri, bir öncekinin eksikliğini tamamlamak amacıyla yöneliktir.

Veri modeli; veritabanının yapısını tanımlayan kavramların bir kümesidir. Veritabanının modeli; veri tipleri, işlemler ve kısıtlamalardan oluşur. Veri modeli veritabanının nasıl oluşturulacağı, nasıl kullanılacağı ve hangi sınırlar içerisinde tanımlı olduğu ile ilgilidir. Veri modeli içerisinde yer alan işlemler kümesi veritabanından seçim yapma ve veritabanını güncelleme ile ilgilidir.

3.1. Veri Modellerinin Sınıflandırılması

Basit Veri Modelleri: Geliştirilen ilk iki model (Hiyerarşik veri modeli ve Ağ veri modeli), diğerlerine göre basit bir yapı oluşturduğu için bu modelleri basit veri modelleri olarak adlandırmamız yerinde olacaktır.

Modern Veri Modelleri: Zamanla bilgisayarda bir işlem yapabilmek için sadece verinin işlenmesi yeterli olmamaya başladı. Bu nedenle veri ilişkilerinin daha kolay, anlaşılır ve esnek olabilmesi amacıyla yapılan çalışmalar modern veri modellerinin ortaya olmasını sağladı.

Modern veri modelleri aşağıdaki gibi sıralanır:

Varlık-ilişki Veri Modeli

İlişkisel Veri Modeli

Nesne İlişkisel Veri Modeli

Nesne Yönelimli Veri Modeli

Yüksek Seviyeli Veri Modelleri: Bu modeller varlıklar (entities), özellikler (attributes) ve ilişkiler(relationships) gibi kavramlardan oluşur. Bu modeller için en bilinen örnek Varlık-İlişki veri modelidir.

Mantıksal Veri Modelleri: Bu modeldeki kavramlar verilerin bilgisayarda fiziksel olarak organize olma biçimine çok benzerler. Bunlara bir örnek ilişkisel veri modelidir. İlişkisel

veri modelinde iki boyutlu tablolar kullanılır. Tabloların her bir satırında bir kişiye ait değerler, her bir sütununda ise bir özelliğe ait değerler bulunur.

Fiziksel Veri Modelleri: Bu modeller verilerin bilgisayarda nasıl tutulduklarını detaylı olarak gösteren konseptlere sahiptirler. Bunlar kayıtların biçiminde, kayıtların sırasıyla, ulaşma yollarıyla ilgili bilgiler içerirler.

3.2. Veri Modelleri: Kısa Tarihçe

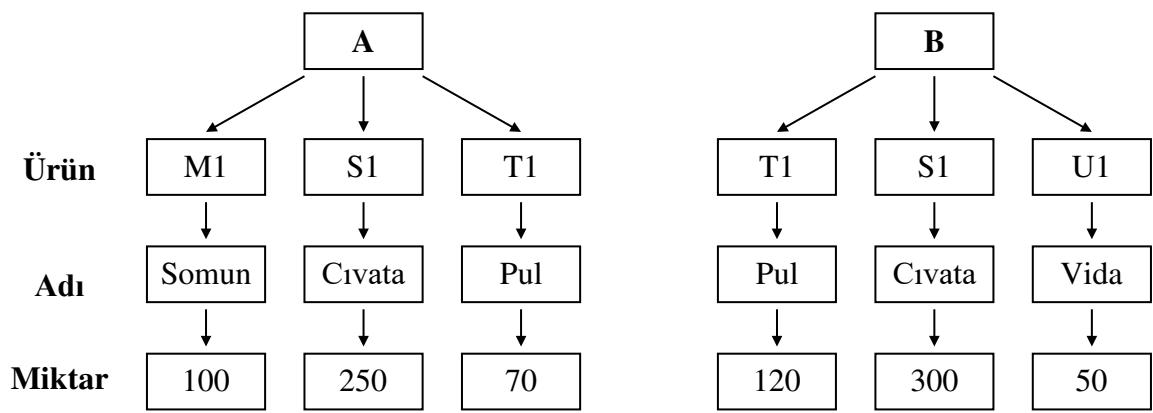
Şu ana dek birçok farklı veri modeli geliştirilmiştir. Bunlara örnek olarak Hiyerarşik veri modeli, Ağ (network) veri modeli ve İlişkisel veri modeli verilebilir. Sayılan bu veri modellerinin içinde en yaygın kullanılanı, ilişkisel veri modelidir. Günümüzde kullanılan VTYS'lerin hemen hemen tümü ilişkisel veri modeline dayalıdır. Son zamanlarda ortaya çıkan nesneye yönelik veri modeli, ilişkisel veri modeli ile birlikte bazı VTYS'lerde kullanılmaktadır. Hiyerarşik Veri Modeli en eski model olup 60-70'li yıllarda kullanılmıştır. 1969'da ortaya çıkan Ağ Veri Modeli 1970'li yıllarda ve 1980'li yılların ilk yarısında kullanılmıştır. İlişkisel veri modeli de ilk kez 1969 yılında ortaya atılmış, 1970'li yılların sonunda kullanılmaya başlanmıştır ve 1985 yılından sonra yaygınlaşmış bir yaklaşımdır. 1990'lı yıllarda yaygın kullanılan VTYS'lerin hemen hemen tümünün ilişkisel tabanlı olduğu söylenebilir. Nesneye-yönelik veri modeli yaklaşımı ise on yılı aşkın süredir gündemde olan, günümüzde çok yaygın kullanılmasa bile, kullanımını giderek yaygınlaşan bir yaklaşımdır. Geçmişe baktığımızda, ilişkisel yaklaşımın kullanılmaya başlanması ile sıradüzensel ve ağ yaklaşımının terk edildiği görülmektedir. Buna karşılık nesneye-yönelik yaklaşımın kullanılmaya başlanması ile ilişkisel yaklaşım terk edilmemiştir. Günümüzde hem ilişkisel hem de nesneye-yönelik yaklaşımı birlikte kullanan VTYS'lerinin yaygınlığı görülmektedir.

3.3. Veri Modeline Göre Veri Tabanı Yapıları

Bilinen temel veritabanı yapıları yukarıda özet olarak açıklandığı üzere; hiyerarşik, ağ ve ilişkisel ve nesne yönelimli veri tabanlarıdır. Kullanıcıların, kullanıcı ihtiyaçlarına en iyi hizmet verebilecek uygulama programları geliştirebilmeleri için veri tabanındaki veri elemanları arasındaki ilişkinin mantıksal olarak yapılması gereklidir. Amaç, verilerin nasıl ilişkilendirileceğinin gösterilmesidir. Veri tabanlarında depolanan kayıtlar ve onların birbirleriyle ilişkileri değişik mantıksal yapılarla gösterilebilir. Veri Tabanı Yönetim Sistemleri bu yapıların oluşumları ve onların fonksiyonları kullanılarak tasarlanır. Veri tabanları aşağıdaki 4 yapıldan herhangi birisiyle yapılandırılabilirmektedir. Burada yapılan sınıflandırma veri modeline göre yapılmaktadır.

3.3.1. Hiyerarşik Veritabanları

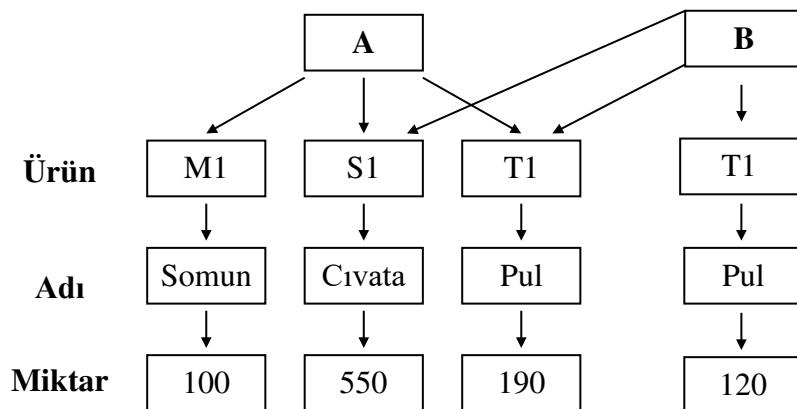
Bu yapı, bilinen en eski veri modelleme sistemidir. Kullandığı veri modeli ağaç yapısıdır. Ebeveyn ve çocuk ilişkisine dayalı bir sistemdir. Hiyerarşim modelde veri türleri üstten aşağıya doğru bir usulde düzenlenir. Oluşturulan mantıksal bağlantılarla veri türleri ilişkilendirilir. Buda bir ağaç gibi veya organizasyon haritası şeklinde görülür. Şekil 2 de iki ayrı müşteri (A ve B) için hiyerarşik veri yapısı aşağıdaki şekilde modellenmektedir.



Şekil 2 Hiyerarşik Veritabanları Sistemi

3.3.2. Ağ Veritabanları

1960'lı yılların sonunda yapılan bir konferansın ardından toplanan veritabanı çalışma grubu hiperarşik modeldeki bazı eksiklikleri gidermek adına bu ağ modelini geliştirmiştir. Bilinen en karmaşık metodudur ve çizge veri yapısını temel alır. Bu yapı daha çok karmaşık bağlantılarla izin verir. Veri türleri arasında yan bağlantılarla ilişki kurulur. Burada aynı veri türlerinin paylaşımı söz konusudur (Mesela, A ve B müşterileri Şekil 3 de görüldüğü gibi S1 ve T1 'i paylaşmaktadır). Bu yapı aynı zamanda CODASYL model olarak adlandırılır.

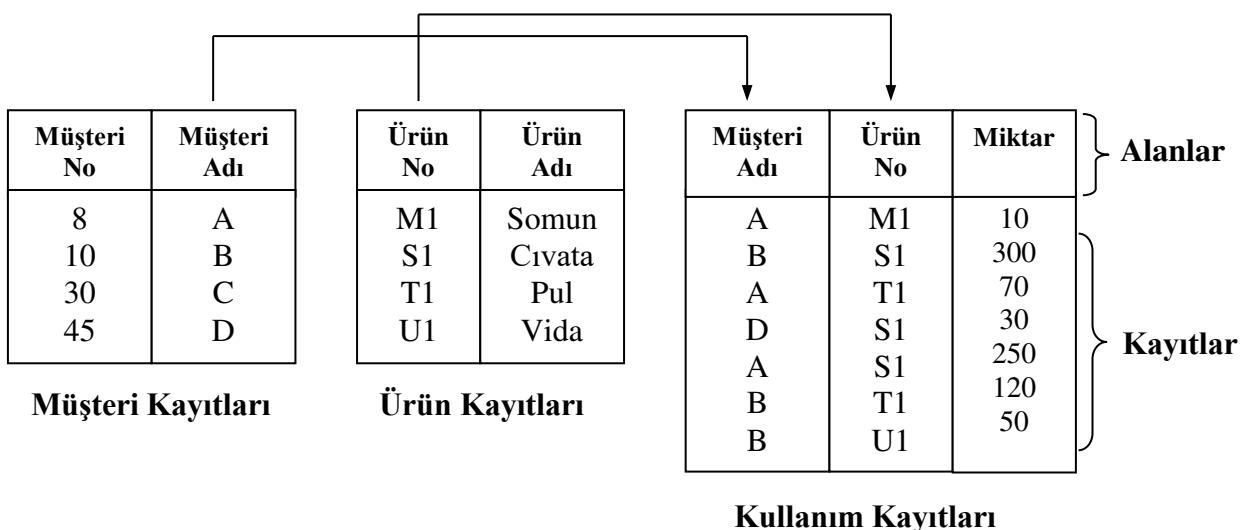


Şekil 3 Ağ Veritabanları Sistemi

3.3.3. İlişkisel Veritabanı Sistemi

Bu modelde temel alınan şeyler verilerin birbirine ilişkileridir. 1970'lü yılların başında geliştirilmiştir. Günümüzde en çok kullanılan modeldir. PC'lerden sunuculara kadar yayılmış bir yelpazeye sahiptir ve bu mimari kullanılarak birçok yazılım üretilmiştir.

Bu formda veri tabanı organizasyonu iki boyutlu tablolar şeklinde düşünülecek kullanıcuya sunulur. Veri tabanı kullanıcıları veri raporlarını bu yolla görür. İlişkisel VTYS birçok giriş sorgusu sunar. Böylece bir veri dosyası bir sayfada aşağıya doğru ilerleyen birkaç sütun içerir. Bu sütunlar kişisel alanları teşkil eder. Bir sayfadaki satırlar çeşitli alanların oluşturduğu kişisel kayıtları gösterir. İki yada daha fazla veri dosyasında bulunan ortak veri alanları vasıtıyla bu tür veri dosyalarında ilişki kurulabilir. Bu ortak alanlar tümüyle benzer olmalı ve aynı boyutta ve tipte olmalıdır. Örneğin Şekil 4'da Müşteri Adı diye adlandırılan veri alanı hem müşteri kayıtlarda hem de kullanım kayıtlarında bulunmaktadır. Bu şekilde ilişki kurulur. Ürün diye adlandırılan veri alanı hem ürün kayıtlarında ve hem de kullanım kayıtlarında bulunmaktadır. Bu şekilde üç veri alanı birbirleriyle ilişkilendirilir. Böylece ilişkisel veri tabanı dediğimiz bütünsel form meydana gelir. Veri tabanında bu formun avantajı kullanıcının öğrenmesini basitleştirmede kolaylık sağlar.



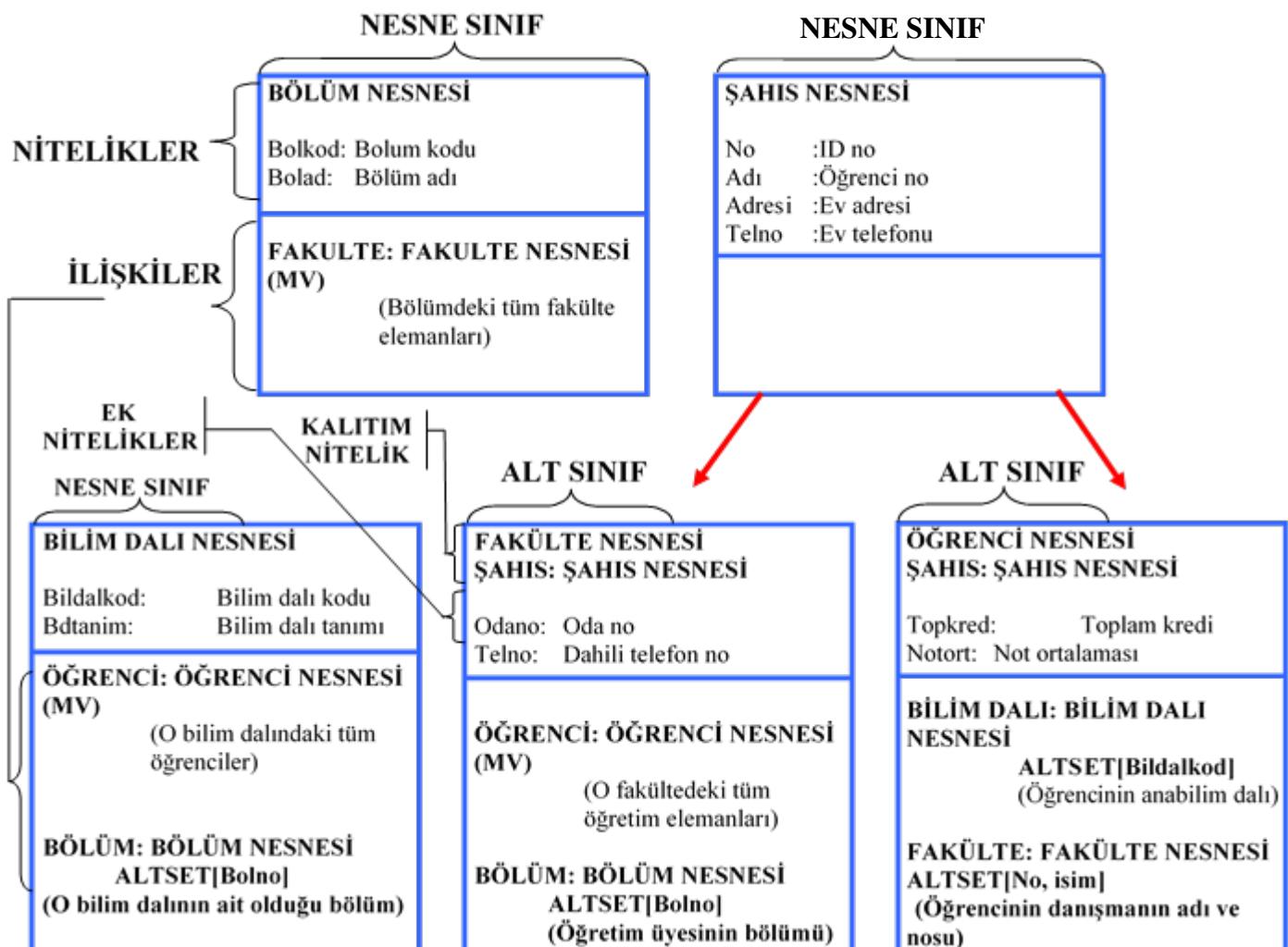
Şekil 4 İlişkisel Veritabanı Yapısı

3.3.4. Nesneye Yönelik Veritabanları

İlk geliştirilen VTYS'ler kolayca önceden tanımlanmış veri alanlarına göre yapılandırılabilem homojen veriler için tasarlanmıştır. Fakat bugünkü ve gelecekteki uygulamalar, sadece önceden tanımlanmış sayıları ve karakterleri değil, çizimleri, resimleri fotoğrafları, sesleri ve videoları da saklayabilen veritabanlarını gerektirecek.

İlk VTYS'ler bu tür grafik ve multimedya uygulamaları için uygun değildir. Bu tür verileri ilişkisel sistemlerde yönetmek için büyük çaplı programlama gereklidir çünkü bu karmaşık veriler tablolara ve satırlara dönüştürülmelidir. Fakat nesneye yönelik (object-oriented) VTYS'ler verileri ve prosedürleri nesneler halinde saklar ve bu nesneler istediginde otomatik olarak alınabilir ve paylaşılabilir. Nesneye yönelik veritabanı yönetim sistemleri (Object-oriented VTYS, OOVTYS) popüler olmaya başladı çünkü bu sistemler web uygulamalarında kullanılan çeşitli multimedya bileşenlerini yada Java "appletlerini" yönetmek için kullanılabilir.

OOVTYS, tekrarlı veriler gibi veri türlerini saklamak için de oldukça kullanışlıdır. Örneğin üretim uygulamalarında parçalar içinde parçaların bulunduğu (malzeme listesi) tekrarlı veriler bu sistemlerde kolayca baş edilebilir. Finans ve ticaret uygulamaları da genellikle OOVTS kullanır çünkü bu uygulamalar yeni ekonomik şartlar altında kolayca değiştirilebilen veri modelleri ister. Şekil 5'de Nesne yönelimli veri tabanı yapısı görülmektedir.



Şekil 5 Nesne Yönelimli Veri Tabanı Yapısı

Her ne kadar nesneye yönelik veritabanları daha karmaşık veri türlerini saklayabilse de, büyük miktarda verilerin işlenmesi gerektiği durumlarda bu sistemler ilk çıkan ilişkisel VTYS'ler ile karşılaşıldığında yavaş kalır. Bugün hibrid object-relational VTYS'ler gündeme gelmiştir. Bu melez sistemler her iki sistemin yeteneklerini birleştirir.

Bu tür bir melez sistem yaklaşımı üç farklı şekilde elde edilebilir:

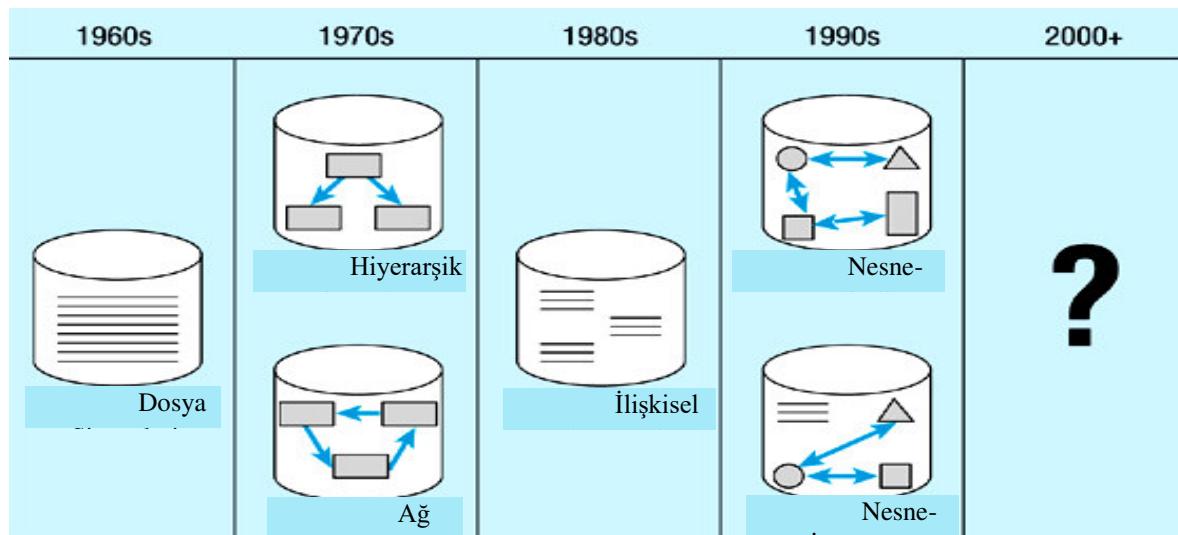
1- İlişkisel VTYS'lerine nesneye yönelik erişim sağlayan araçlar kullanarak,

2- Mevcut ilişkisel VTYS'ne nesneye yönelik uzantılar ekleyerek, yada

3- Melez bir “object-relational” veri tabanı yönetim sistemi kullanarak.

Nesneye Yönelik Veritabanı, Nesneye dayalı veri tabanında, yapısı gereği arama işlemleri çok hızlı yapılabilir. Özellikle büyük tablolarla uğraşırken ilişkisel veri tabanlarından çok daha hızlı sonuca ulaşırlar. Ancak çalışma mantığı tümüyle değişir.

Tüm bu özellikler tamamen nesneye yönelik olan veri tabanları için geçerlidir. Bazı ilişkisel veri tabanları ile çalışan yazılımlarda da nesnenin bazı özellikleri kullanılır, ama nesneye yönelik veri tabanı bunu kendini ilişkisel veri tabanı kurallarına uydurarak gerçekleştirebilir.



Şekil Veri Modeline Göre Veri Tabanı Yapıları

3.4 Varlık İlişki Diyagramları

İlişkisel veritabanı tasarımda en fazla kullanılan yapısal tasarım yada modelleme aracı Varlık İlişki Diyagramlarıdır (Entity-Relationship E-R). Varlık İlişki Diyagramları, veritabanı tasarımda kullanılan ve varlıklar (entity) arası ilişkileri gösteren diyagamlardır. Bu diyagamlar, bir veritabanı için referans dokümanlar olarak da kullanılabilirler. İlişkisel veritabanı tasarımda amaç, veri tekrarını azaltan ve veri tutarlığını yükselten bir yapının oluşturulmasıdır. Veri tabanı tasarım ilk olarak, veritabanı yapılacak kurum ya da kuruluşun ihtiyacı olan bilgilerin analizi yapılır. Bu süreçte, veritabanı tasarımcısı, veritabanını kullanıcıları ile görüşerek kullanıcı ihtiyaçlarını tespit eder. Veriler toplanıp sistem analizi yapıldıktan sonra, ilişkisel veri modeline göre veritabanı şeması oluşturulur. Bu şema, veritabanı kullanıcılarının ihtiyaç duyukları verilerin kısa açıklamasıdır. Bu şema içerisinde varlık tipleri ve ilişkilerin açıklaması bulunur. Daha sonra bu şemaya göre veri tabanını oluşturacak tablolar çıkarılır. Varlık-ilşki modeli, 1976 yılında P.P. Chen tarafından geliştirilen bir modeldir. Varlık-ilşki modeli, VTYS'den bağımsız veri çözümlemeye en çok kullanılan modeldir. Bu model kullanılarak önce; VTYS'den bağımsız olarak veriler çözümlenir, veri modellemesi yapılır, veriler ve veriler arası ilişkilerin anlamları ve özellikleri incelenerek

Varlık-İlişki Diyagramları oluşturulur. Ardından kullanılacak VTYS belirlenir. VTYS'de de fiziksel olarak veri tabanı şemaları oluşturulur.

3.4.1. Varlık-İlişki Diyagramlarının Bileşenleri

Varlık-İlişki diyagramları oluşturulmada varlıkların ve onlarla ilgili özelliklerin belirlenmesi gereklidir. Varlıklar ve özellikler belirlendikten sonra yapılacak olan, varlıklar arasındaki ilişkilerin oluşturularak Varlık-İlişki Diyagramının yapılandırılmasıdır. Varlık-İlişki Diyagramlarının aşağıdaki temel semboller kullanılır (Şekil 1)

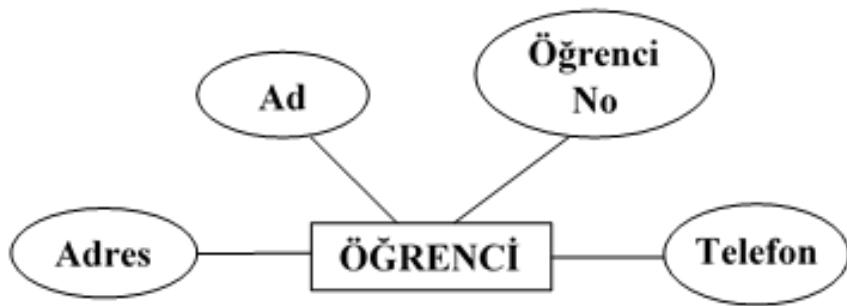


Şekil 1.Varlık-İlişki Diyagramlarında Kullanılan Semboller

3.4.1.1 Varlık ve Özellikleri

Gerçek hayatı diğerlerinden ayırt edilebilen nesnelere varlık denir. Bir varlık, bir çevresel veri ya da bir kaynak olabilir. Bir varlık, kişi, araba, ev veya çalışan gibi fiziksel nesneler olabileceği gibi, şirket, iş veya ders gibi fiziksel olmayan nesneler de olabilir. Örneğin, kişi, yer, müşteri, depo, parça, ürün, ekipman, sipariş vb hepsi birer varlıktır. Her varlık kendisini tanımlayan kendisine has özelliklere sahiptir.

Varlık kümesi, Veritabanında benzer varlıklar ve özellik değerlerinden oluşan kümedir. Varlık-İlişki şemasında varlık kümeleri dikdörtgen içinde belirtilir. Özellikler ise oval bir daire içinde belirtilerek ilgili varlık kümeye çizgi ile bağlanır. Şekil 2'de öğrenci varlık kümesi ve ad, numara, adres ve telefon özelliklerinin şeması görülmektedir.



Şekil 2. Öğrenci varlık kümesi ve özellikleri

Varlıklar güçlü ve zayıf varlık olmak üzere ikiye ayrılır:

Güçlü varlıklar (Strong Entity): Diğer varlık türlerinden bağımsız olabilen varlık türüdür.

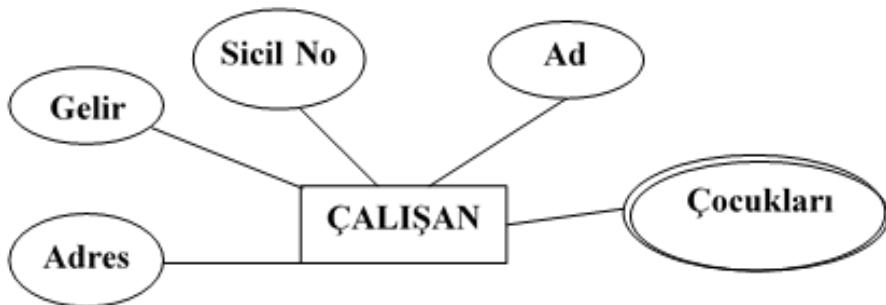
Zayıf varlıklar (Weak Entity): Varolması başka bir varlığa bağlı olan varlıklardır. Bir iş yerindeki çalışanlar güçlü varlıklar iken bu çalışanların çocukları zayıf varlıklardır. Çalışanlardan birisi ayrıldığında direkt olarak çocukları da veritabanından silinir. Zayıf varlıkların birincil anahtarları elips içinde ve altı kesikli çizgi ile gösterilir.

3.4.1.2 Özellikler (Attributes)

Özellik, herhangi bir varlığa ait durumlardır. Örneğin müşteri no, müşteri adı, müşteri adresi vb. durumlar, müşteri varlığının özellikleri olarak ifade edilebilirler. Yada bir çalışan varlığı, çalışan adı, yaşı, adresi, geliri ve görevi özellikleri ile tanımlanır. Varlıklar Varlık-İlişki Diyagramı da dikdörtgen sembolüyle ifade edilirler. Özellikler o varlığı diğer varlıklardan ayırt etmeye yarar. Özellikler Varlık-İlişki Diyagramlarında elips şeklinde gösterilirler. Özellikler de kendi aralarında farklılık arzederler.

Tek Değerli Özellikler - Çok Değerli Özellikler

Bir varlığın sadece 1 değer alabilen niteliğine denir. Örneğin bir iş yerindeki çalışanların tek sosyal sigorta numaraları vardır. Genellikle bir varlığın tek bir değeri vardır. Bir varlığın bir özelliğinin aldığı değer tek ise bu niteliğe tek değerli özellik denir. Ancak bazı özelliklerin birden çok değeri olabilir. Örneğin Çalışan varlığının yabancı dil özelliği, birden çok değer alabilir. Bu durumda Çalışan varlığının yabancı dil özelliği çok değerli özellik denir. Diğer ifadeyle çok değerli özellik, Bir varlığın birden fazla değer alabilen özellikleridir. Örneğin bir çalışanın birden fazla telefon numarası olabilir. Başka bir örnek olarak, bir çalışanın çocukları özelliği bir kişi de olabileceği gibi birden çok kişi de olabilir. Bu durumda Çalışan varlığının Çocukları özelliği çok değerli özelliktir. Varlık-İlişki şemasında çok değerli özellikler çift çizgili oval olarak gösterilir (Şekil 3).



Şekil 3. Çalışan Varlık Kümesi ve Çok Değerli Özelliği

Veri Kümesi (Domain): Bir özelliğin alabileceği değerler. Varlık kümesinde özelliklerin alabileceği değerler aralığını belirler. Örneğin öğrenci varlık kümesinde, doğumTarih özelliği 01.01.1980 ile 01.01.1990 arasında olabileceğini belirlemek için domain kullanılır. Örnekleri şu şekilde çoğaltabiliriz Cinsiyet → Erkek, Kız. Göz Rengi → Mavi, Ela, Kara, vs. Vize → [0-100]. Domain aralığı Varlık-İlişki şemasında gösterilmez.

Veri Türü: Bir özelliğin içerebileceği verinin hangi türden olacağını belirler. Tamsayı - INTEGER, Değişken sayıda karakter - VARCHAR, Tarih – DateTime ve Para - Money gibi.

Türetilmiş Özellik: Bir varlığa ait özellikler kullanılarak yeni bir özellik türetilabilir. Örnek; Doğum Tarihi özelliğinden türetilen Yaş özelliği.

Anahtarlar Özellik: Varlıklarını ya da ilişkilerini birbirlerinden ayırt etmek için kullanılan Özellik ya da özellik grubuna denir. Her varlığa ait bir özellik anahtar olmalıdır. Varlık kümesindeki bir veya daha fazla özelliğin değeri, her bir varlık için farklı ise bu özellik anahtar özellikle. Örneğin öğrenci varlık kümesinde öğrencino anahtar özellikle. Çünkü bir üniversitede, öğrenci varlık kümesinde hiçbir varlığın öğrencino özelliği aynı olamaz. Benzer şekilde Çalışan varlık kümesinde, sivilno özelliği anahtar özellikle. Hiçbir Çalışanın sivilno'su aynı olamaz. Anahtar özellik, Varlık-İlişki şemasında özelliğin altı çizilerek gösterilir.

Aday Anahtar (Candidate Key): Bir varlığı kesin olarak tanımlayan ve tek olan Özelliklere aday anahtar denir. Örneğin bir çalışanın sigorta numarası ve TC kimlik numarası aday anahtar olabilir fakat, çalışanın adı aday anahtar olamaz çünkü aynı ada sahip birden fazla çalışan olabilir.

Birincil Anahtar (Primary Key): Aday anahtarlar arasından seçilen anahtardır. Varlık-İlişki Diyagramlarında bir niteliğin birincil anahtar olduğunu belirtmek için o Özellik elips içinde altı çizili olarak yazılır.

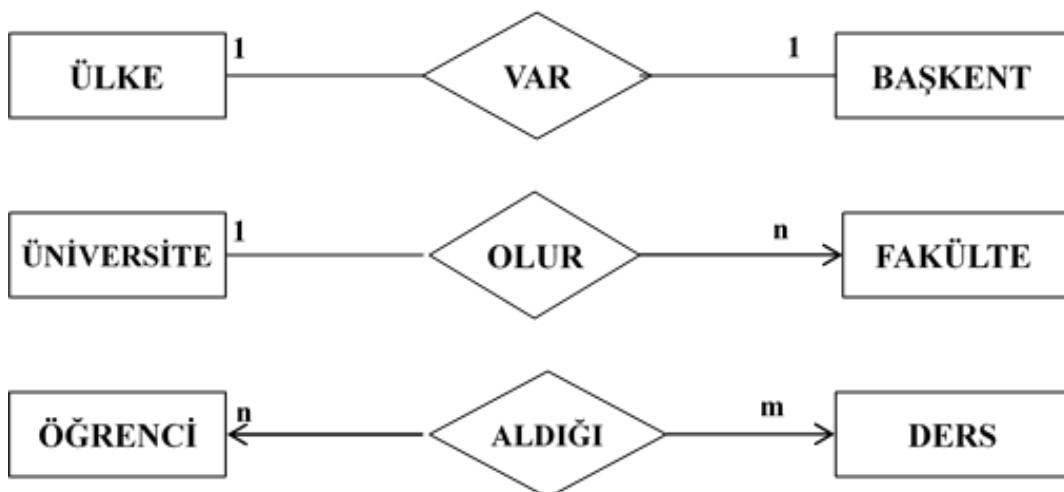
Birleşik Anahtar (Composite Key): Eğer bir aday anahtar iki ya da daha fazla özellikten oluşuyor ise bu anahtar birleşik anahtardır.

3.4.1.3 İlişkiler

İlişki (Relationship): Varlıklar arasındaki bağlantılardır. İki veya daha fazla varlık kümesi arasında kurulan anlamlı bağıntılara ilişki denir. Aynı varlıklar arasında farklı türden ilişkiler olabilir veya bir ilişki bir ya da daha fazla varlık arasında olabilir. Bir ilişki, iki ya da daha fazla veri varlığı arasındaki mantıksal etkileşimdir. Varlık-ilişki, hem varlıklarını hem de ilişkileri göstermektedir. İlişkiler, eşkenar dörtgen sembolüyle ifade edilirler. Dörtgen içine ilişkinin adı yazılır. Müşteri varlık kümesi ile ürün varlık kümesi arasında satın alır ilişkisi örnekleri verilebilir (Şekil 4). Diğer örnekler ise Şekil 5'de görülmektedir.



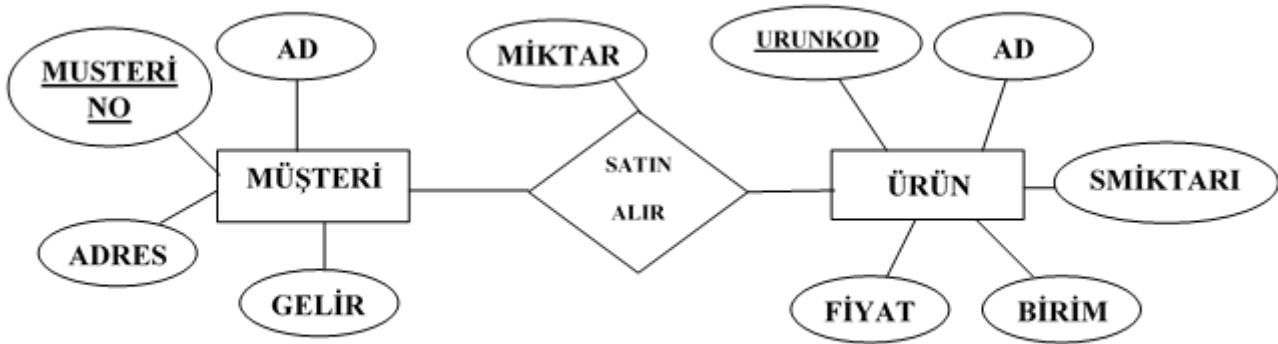
Şekil 4. Müşteri-Ürün Varlık Kümesi Satın Alır İlişkisi



Şekil 5 İlişki Kümesi Örnekleri

İlişki Derecesi: İlişkide bulunan varlık kümelerin sayısı ilişkinin derecesini oluşturur. Genelde ilişkiler iki varlık kümesi arasında yapılır. Ancak bazı durumlarda, ilişkide ikiden fazla kümesi yer alabilir. Örneğin bir Çalışanın hangi projede hangi görevi yaptığı bilgisi gerekli ise; Çalışan, proje ve görev varlık kümeleri arasında çalışır ilişkisi yapılır.

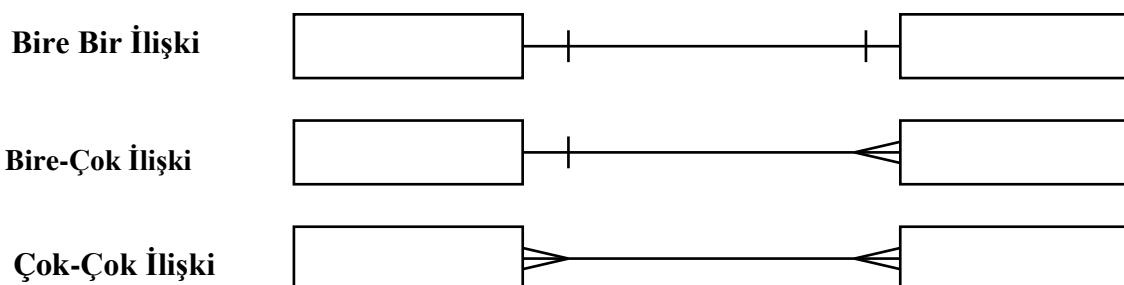
Tanımlayıcı Özellik: Varlık kümeleri arasındaki ilişkiden oluşan özelliklere tanımlayıcı özellik denir. Tanımlayıcı özellikler ilişkinin olması ile var olabilir. İlişki oluşmaz ise bu özellikler de var olamaz. Müşteri varlık kümesi ile ürün varlık kümesi arasında satın alır ilişkisinde, müşterinin satın aldığı ürünün miktarı özelliği tanımlayıcı özellikleştir. Tanımlayıcı özellikler ilişki ile bağlanır (Şekil 6).



**Şekil 6. Müşteri – Ürün Satın Alır İlişkisinde Miktar Tanımlayıcı Özellikler
İlişki Türleri**

İki varlık arasında genelde 3 ilişkiden bahsedilebilir. Bunlar (Şekil 7);

- i) Bire-Bir (one-to-one) ilişki (1:1)
- ii) Bire-Çok (one-to-many) ilişki (1:M)
- iii) Çok-Çok (many-to-many) ilişki (M:M) dir.



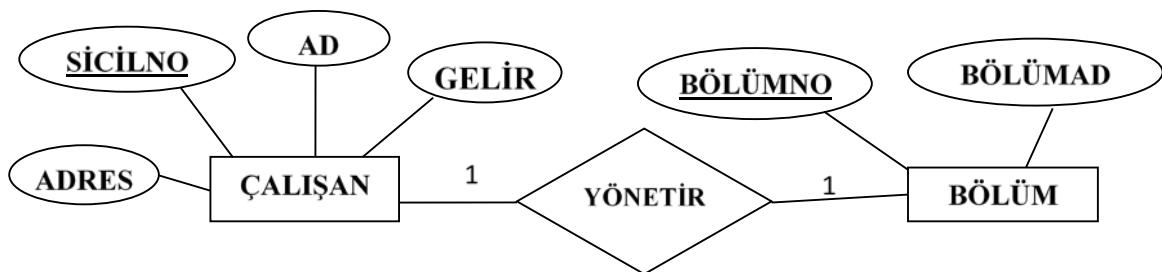
Şekil 7 İlişki Türleri

Bire-bir (1-1) ilişki

A varlık kümelerinin bir elamanı, B varlık kümelerinin bir elemanı ile ilişki kurabiliyorsa bu ilişki türüne bire-bir ilişki denir. Örneğin Her müşterinin bir hesabı olabilir ilişkisi bire-bir ilişkidir(Şekil 8). Kişi ve kimlik, zaman kartı ve bordro fişi, araba ve plaka vb arasındaki ilişkiler bire-bir ilişkiye örnek olarak verilebilir. Şekil 9. Çalışan - Bölüm yönetir ilişkisini gösterir.

Şekil 8: Her Müşterinin Bir Hesabı Olabilir 1-1 İlişkisi

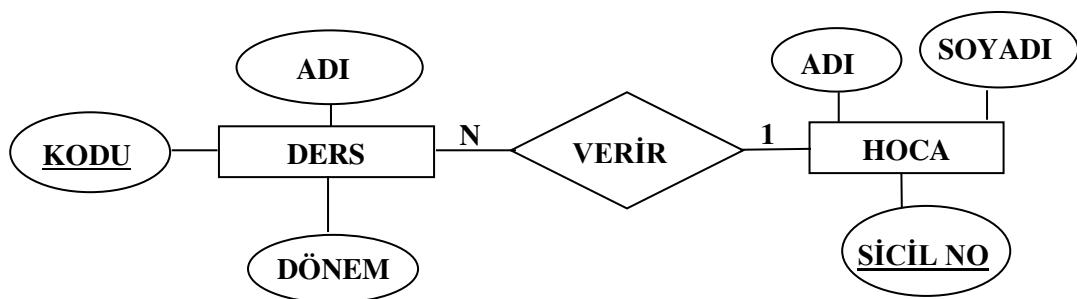
Müşteri No	Müşteri Adı	Hesaplar	Bakiye
101	Ali	34013	5.000
203	Ayşe	33790	45.000
405	Mehmet	33567	2.500,
507	Derya	33344	1.000



Şekil 9. Çalışan - Bölüm Yönetir 1-1 İlişkisi

Bire-Çok (1-N) İlişki

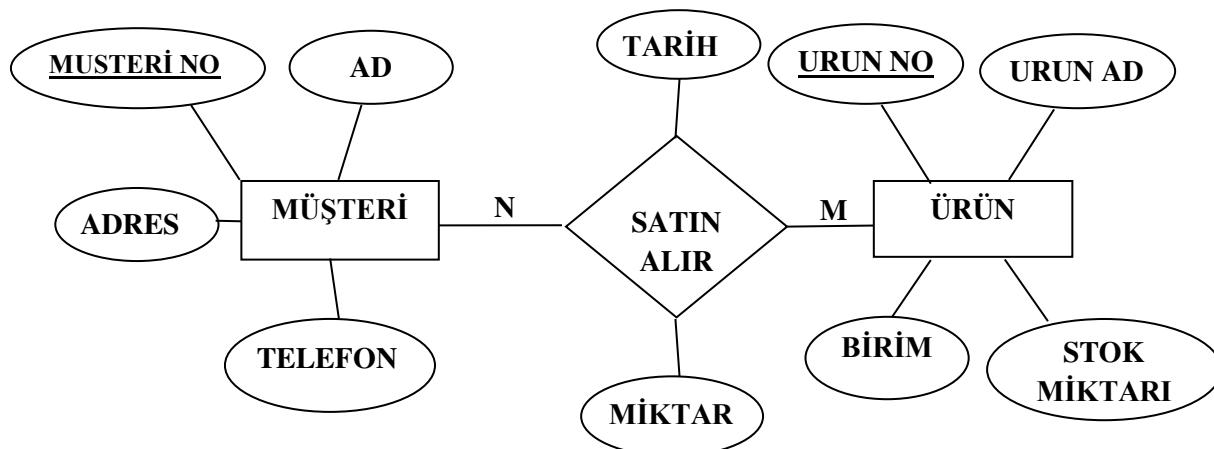
A varlık kümelerinin bir elemanı B varlık kümelerinin birden çok elemanı ile ilişki kurabiliyorsa bu ilişki bire-bir ilişkidir. Hoca ile ders varlıklarları arasında bire-çok bir ilişkinin olduğu görülmektedir. Yani bir hoca, birden fazla derse girebildiği, buna karşılık bir dersi sadece bir hoca tarafından verildiği ifade edilmektedir (Şekil 10).



Şekil 10. Hoca Ders Verir Bire-Çok İlişkisi

Çoğa-Çok (n-n ya da n-m) İlişki

A varlık kümesinin bir elemanı, B varlık kümesinin sıfır ya da birçok eleman ile ilişki kurabiliyor ve B varlık kümesinin bir elemanı, A varlık kümesinin sıfır ya da birçok elemanı ile ilişki kurabiliyorsa bu ilişki türü çoğa çok ilişki türüdür. Çoğa çok ilişki kısaca n-m ilişki olarak da adlandırılabilir. Kitap ile Yazar varlıklarları arasında çoğa-çok bir ilişkinin olduğu görülmektedir. Yani bir kitabın birden fazla yazar tarafından yazılabildiği, aynı zamanda bir yazarın da birden fazla kitap yazabilmesidir. Benzer şekilde, Proje ile Çalışan, Müşteri ile Ürün arasındaki ilişkiler de çoşa-çok ilişkiye örnek olarak verilebilirler. Örneğin müşteri ile ürün varlık kümeleri arasında satın alır ilişkisi çoşa çok ilişkidir. Çünkü bir müşteri birden çok ürün satın alabilirken, bir ürünü birden çok müşteri satınalabilir(Şekil 11).

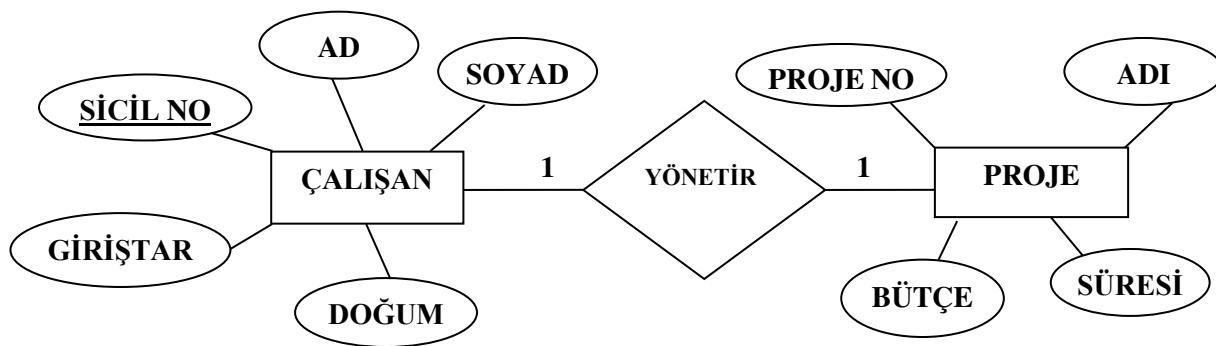


Şekil 11: Müşteri-Ürün Satın Alır N-M İlişkisi

3.5 Varlık-İlişki Şemasının Tablolara Dönüşürtülmesi

3.5.1 Bire-Bir İlişkilerin Dönüşürtülmesi

Birebir ilişkiyi oluşturan varlık kümeleri tablolara dönüştürülür. Özellikleri tabloların alanlarına dönüşür. Uygun olan varlık kümesinin anahtar alanı diğer varlık kümesine yabancı anahtar olarak eklenir. Bire-bir ilişkide belirtilen tanımlayıcı özellikler, yabancı anahtar eklenen tabloya alan olarak eklenir. Örneğin proje ve Çalışan varlık kümeleri arasındaki yönetir birebir ilişkisinde iki şekilde dönüştürme yapılır (Şekil 12). Birincisi, her iki varlık kümesi proje ve Çalışan tablolara dönüştürülür. Proje tablosuna Çalışan tablosunun anahtar alanı olan sicilno alanı yabancı anahtar olarak eklenir. İkincisi, benzer şekilde her iki varlık kümesi proje ve öğrenci tablolara dönüştürülür. Öğrenci tablosuna proje tablosunun anahtar alanı olan projeno alanı yabancı anahtar olarak eklenir.



Şekil 12 Çalısan-Proje Yönetir Bire-Bir İlişkisi Varlık-İlişki Şeması

Varlık-İlişki şemasının tablolara dönüştürme ya birinci anahtar alanı diğer tablonun içerisinde koyarak ya da ikinci tablonun anahtar alanını birinci tablonun içerisinde koyarak yapılır.

Proje					
1	Proje No	Proje Adı	Bütçe	Süresi	
	1	P1	200	10 yıl	
	2	P3	100	3 yıl	
	3	P16	100	4 yıl	

Çalışan					
Sicil No	Adı	Soyadı	Giriş Tarihi	Doğum Tarihi	Proje No
1	Veli	Tane	06/03/2004	15/10/1970	1
2	Ayşe	Şahin	15/10/2003	01/08/1966	2
3	Murat	Al	29/05/2002	10/02/1971	3
4	Ali	Taş	12/05/2005	18/07/1974	2
5	Gül	Mor	07/08/2007	23/02/1972	1

Şekil 8.1. Çalışan-Proje Yönetir İlişkisi Tabloları

Bire-Bir (1-1) İlişki; Tablolar arası ilişki kurulan alanların her iki tabloda da tek olması anlamına gelir (Şekil 13).

Kimlik Bilgileri			Şifre	
TcNo	Ad	Soyad	Tc No	Şifre
121	Ayşe	Berk	121	sifre1
243	Kemal	Kurt	243	asd123
982	Musa	Tufan	982	111111

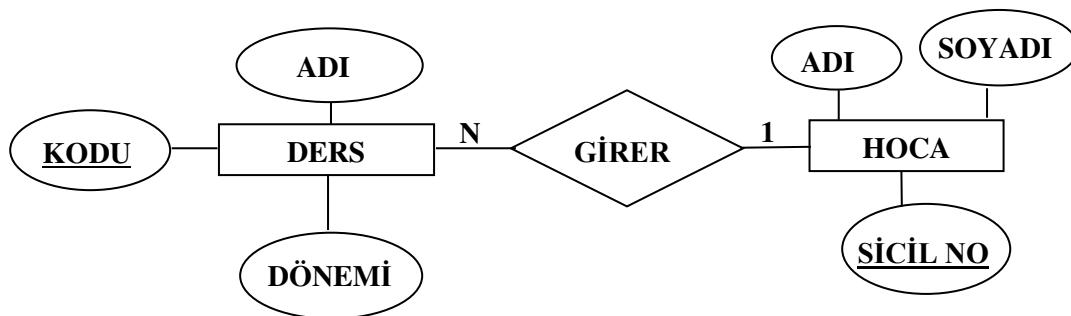
1

1

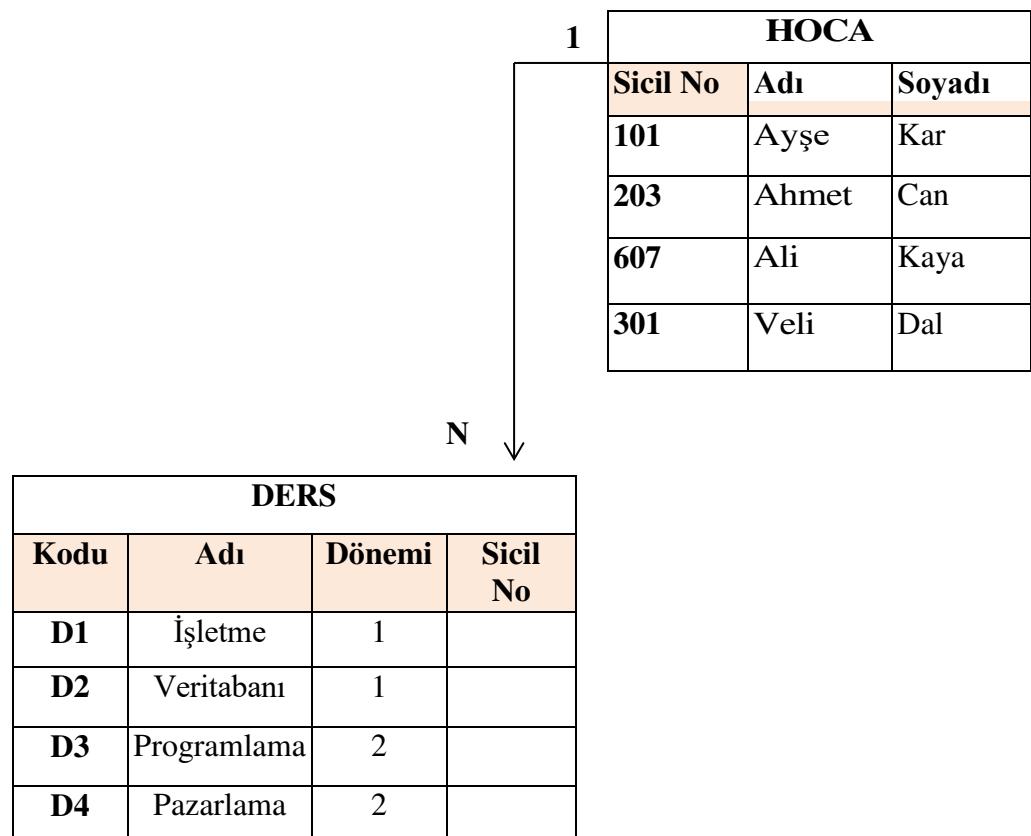
Şekil 13 Bire-Bir (1-1) İlişki: 1 Kişiye Ait Sadece 1 Şifre Olabilir, 1 Şifre 1 Kişiye Ait Olabilir.

3.5.2 Bire-Çok (1-n) İlişkilerin Tablolara Dönüşürtlmesi

İlişkiyi oluşturan varlık kümeleri tablolara dönüştürülür. İlişkinin n tarafındaki tabloya 1 tarafındaki tablonun anahtar alanı yabancı anahtar olarak eklenir. İlişkide belirtilen tanımlayıcı özellikler n tarafına alan olarak eklenirler. Hoca ders varlıklarını arasındaki bire bir ilişki ve tablo yapısı şekilde gözükmektedir(Şekil 14)

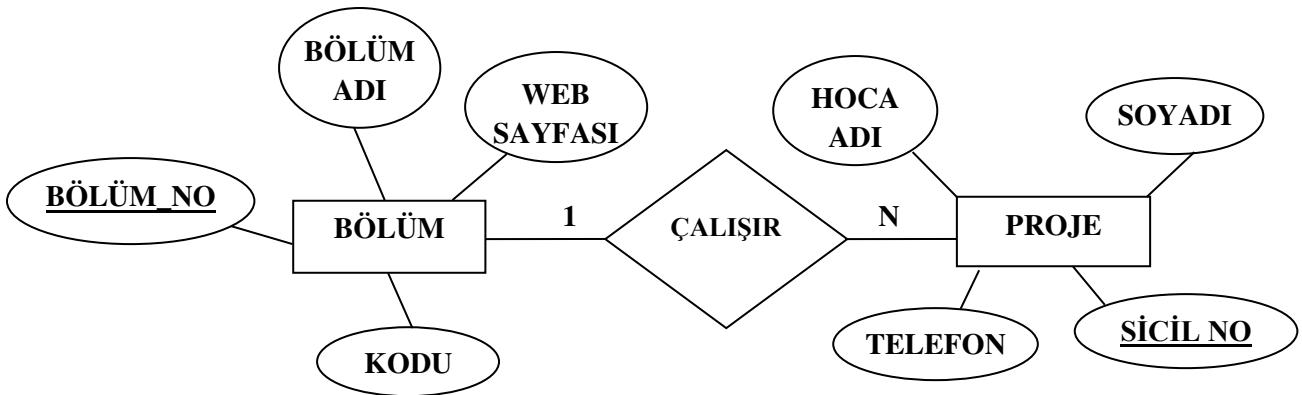


Şekil 10. Hoca Ders Verir Bire-Çok İlişkisi



Şekil 14 Hocalarbirden fazla derse girmektedir.

Örneğin Hoca bölüm varlık kümeleri arasındaki 1-n çalışır ilişkisinde Hoca ve bölüm varlık kümeleri tablolara dönüştürülür (Şekil 15). İlişkinin n tarafındaki tablo olan Hoca tablosuna bölüm tablosunun anahtar alanı olan bölümno alanı yabancı anahtar olarak eklenir.



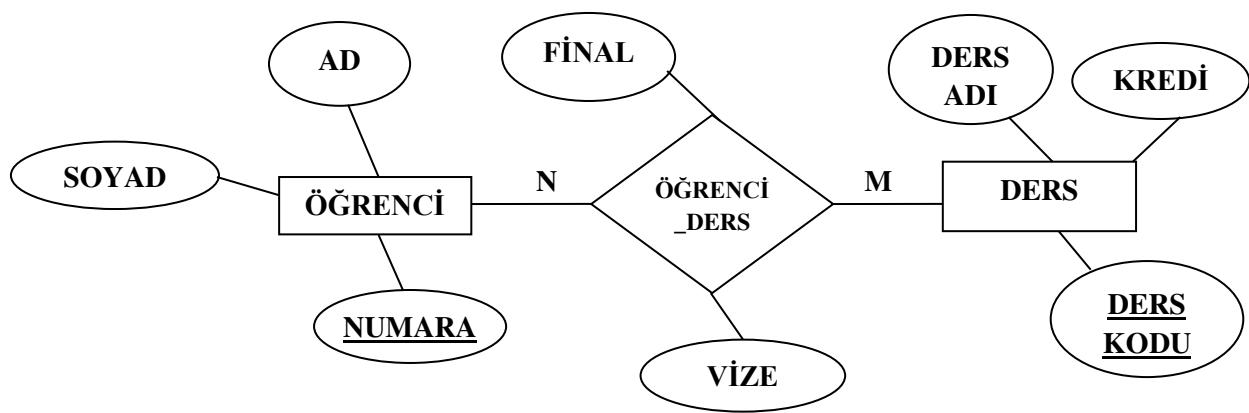
BÖLÜM			
BölümNo	Adı	Kodu	Web Sayfası
1	Matematik	38	www.sak.edu.tr
2	Fizik	47	www.sak.edu.tr
3	Kimya	18	www.ist.edu.tr
4	Biyoloji	293	www.sak.edu.tr

HOCA				
SicilNo	Adı	Soyadı	Telefon	Bölüm
546	Ali	Can	29298	2
342	Fatma	Kara	29245	1
124	Ruhu	Say	29284	3
432	Ali	Demir	29283	4
263	Hüseyin	Cemal	29283	4

Şekil 15 Hoca Bölüm Varlık Kümeleri Arasındaki 1-N Çalışır İlişkisi ve Oluşturulan Tablolar

3.5.3 Coğa-Çok (N-M) İlişkilerin Tablolara Dönüşürtülmesi

İlişkiyi oluşturan varlık kümeleri tablolara dönüştürülür. Ancak ilişki isminde yeni bir tablo oluşturulur. İlişkiyi oluşturan tabloların anahtar alanları yeni tabloya yabancı anahtar olarak eklenir. İlişkide belirtilen tanımlayıcı özellikler varsa yeni tabloya alan olarak eklenir. Yeni tablonun anahtar alanı ilişkiyi oluşturan tabloların yabancı anahtarlarından oluşan ikili veya daha fazla alandan oluşur. Diğer ifadeyle N-M ilişkiler için ayrı bir ara-tablo oluşturulur. Bu ara-tabloda anahtar alan iki tablonun anahtar alanlarının birleşimidir(Şekil 16).



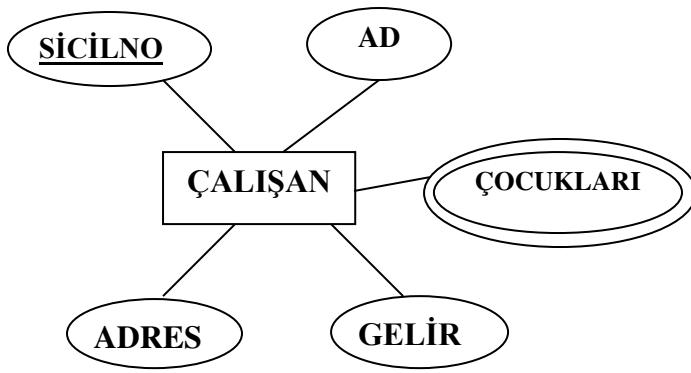
ÖĞRENCİ			1	DERS		
NUMARA	AD	SOYAD		DERS KODU	DERS ADI	KREDİ
101	Ali	Can		201	Programlama	4
102	Fatma	Kara		105	Veri Tabanı	4
103	Ruhu	Say		207	Fizik	3

ÖĞRENCİ_DERS			
NUMARA	DERS KODU	VİZE	FİNAL
101	201	60	67
101	207	80	89
103	207	75	45
103	105	67	36

Şekil 16 Öğrenci Ve Dersin-M İlişkisi

3.6 Çok Değerli Özelliklerin Dönüşürtlmesi

Çok değerli her bir özellik için yeni bir tablo oluşturulur. Yeni tabloya çok değerli alan adı ve çok değerli alanın bulunduğu ilk tablonun anahtar alanları eklenir. Bu yeni tablonun anahtar alanı ise, çok değerli alan ile birlikte eklenen yabancı anahtarın birleşimidir. Örneğin şekil 17'deki Çocukları özelliği çok değerli bir özelliktir. Çünkü her Çalışanın birden çok Çocukları olabilir. Bu durumda Çocukları adında yeni bir tablo oluşturulur. Bu tablonun alanları olarak Çocukları ve Çalışan tablosunun anahtar alanları olan sicilno yabancı anahtar olarak eklenir. Bu tablonun anahtar alanı Çocukları ve sicilno alanlarının birleşimidir.



Şekil 17. Çalışan Varlık Kümesinin Çoklu Değer Özelliği Ve Diğer Özellikleri

Çalışan varlık kümesi tabloya dönüştürülürse;

Çalışan (sicilNo, adres, ad, Gelir) Çocukları (Cocukları, sicilNo)

Varlık-İlişki Diagramı hazırlayabilmek için hazır toollar vardır. Bunlarından bazıları Microsoft Visio, Sybase Power Designer ve ERwindir. Bu tool sayesinde primary key, foreign key, index, view gibi unsurlar da gösterilebilmektedir. Varlık-İlişki diyagramları veritabanı tasarıminın olmazsa olmazlarındandır. Özellikle büyük projelerde veriler arasındaki ilişkiler ve bu verilerin çözümlenmesi işlemleri çok karışık olmaktadır. Bu tür durumlarda Varlık-İlişki diyagramları tüm bu ilişkilerin açıkça belirtilmesi açısından önemlidir. Bu yazımada sizlere Varlık-İlişki diyagramlarının ne işe yaradığını ve temel kavramlarının nasıl kullanıldığını bir örnek üzerinde anlatmaya çalıştım.

Uygulamalar

Veritabanı Gereksinimleri (ŞİRKET)

- Şirket, BÖLÜM'lerden oluşmaktadır. Her bölümün numarası ve adı vardır.
- Her BÖLÜM, belli sayıda PROJE kontrol etmektedir. Her projenin numarası, adı, bütçesi ve süresi vardır.
- Veritabanında şirket ÇALIŞAN'larının sicil numarası, adı, soyadı, maaşı, cinsiyeti ve doğum tarihi saklanacaktır.
- Her çalışan yalnızca bir BÖLÜM'de yer almaktadır, fakat birden fazla PROJE'de çalışabilir. Yine bir projede birden çok çalışan vardır.
- Her çalışanın, projelerde haftalık çalışma saatleri kaydedilecektir.

Tablolarda Saklanacak Bilgiler

ÇALIŞAN→SICILNO, ADI, SOYADI, DTARIHI, MAAS, CINS, BOLUMNO

BÖLÜM→BOLUMADI, BOLUMNO

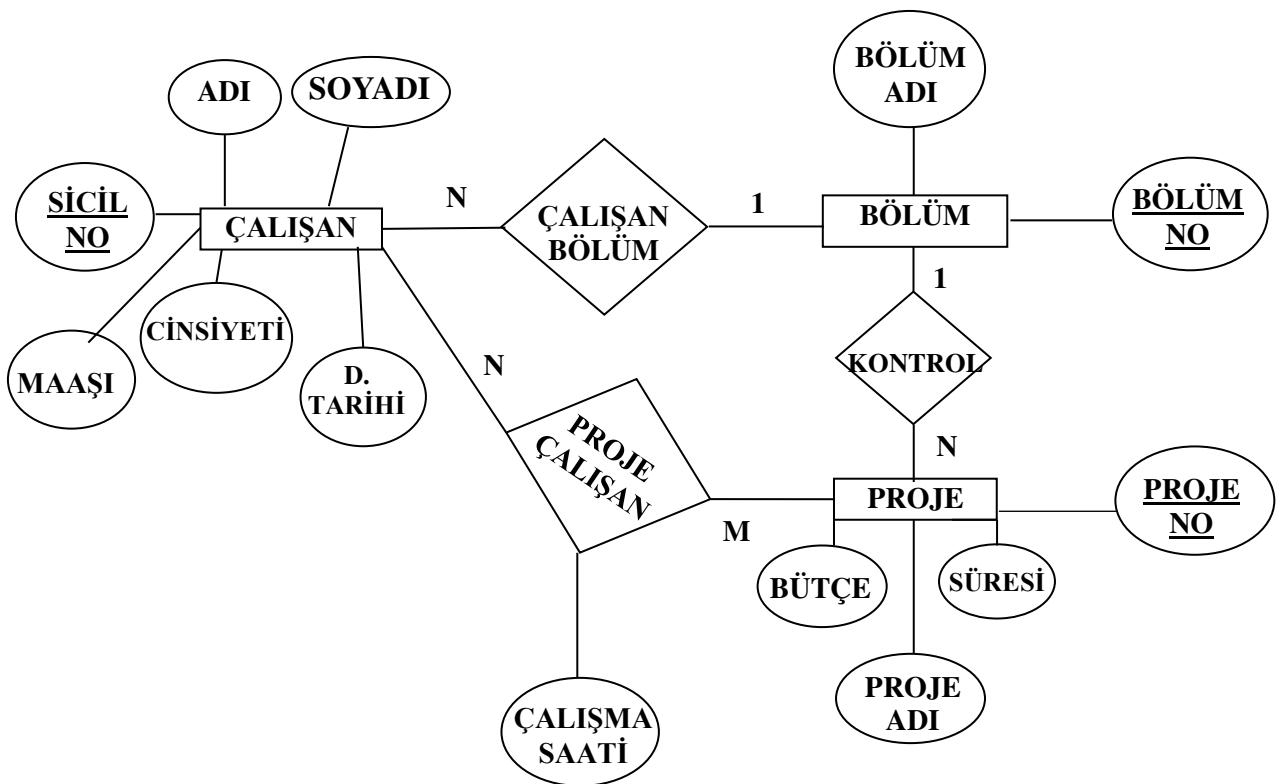
PROJE→PROJENO, PROJEADI, BUTCE, SURE, BOLUMNO

PROJE_CALISAN→PROJENO, SICILNO, HAFTALIKSAATI

NOT-1: Veritabanı gereksinimlerine göre 1:N ilişkiler için yukarıdaki tablolara yabancı ahahtarlar eklenebilecektir. Bunlar yukarıda belirtilmemiştir. Gereksinimler doğrultusunda kendiniz karar vermelisiniz.

NOT-2: Veritabanındaki N:N ilişkilere karşılık gelecek tablolarda yer olması gereken bilgilere veritabanı gereksinimleri doğrultusunda kendiniz karar vermelisiniz.

Varlık-İlişki Diyagramı



Uygulama Soruları

Soru-1: Bir Kütüphane veritabanı için Gereksinimleri ve tablolara ait özelliklerini belirleyiniz. Varlık-ilişki diyagramını çiziniz.

Soru-2: Büro Mobilyaları üreten bir firmanın tüm bilgilerini içerecek bir Varlık-ilişki modelini çiziniz ?

Soru-3: Öğrencilerin almış olduğu dersler ve bu derslere ait vize ve final notlarının bulunacağı Varlık-ilişki diyagramını çiziniz ?

Soru-4: İlişkisel veritabanı oluşturma:

Diyelim ki bir firmamız var ve e-ticaret yapmayı düşünüyoruz. Sadece üyelerle satış yapacağız. Bu nedenle üç tane liste tutmamız gerekiyor. Biri müşteri listesi, diğeri ürün listesi. Biz bu listeleri veritabanında iki ayrı tabloda tutacağız. Üçüncü listede ise siparişleri tutacağız. Gerekli bilgiler şu şekildedir:

İlk Liste (Müşteri)

Listede Yer Alan Bilgiler;

Üye Kodu: Kaçinci Müşterimiz Olduğunu Kaydedeceğiz.

Adı: Üyenin Adı

Soyadı: Üyenin Soyadı

E-Posta,

Adres,

İli,

Telefon.

İkinci Liste (Ürün)

Listede Yer Alan Bilgiler

Barkod Numarası

Marka

Ürün Adı

Özellikleri

KDV Oranı

Alış Fiyatı

Satış Fiyatı

Üçüncü Liste (Sipariş)

Listede yer alan bilgiler.

Sipariş Kodu: Kaçinci Sipariş Olduğu Girilecek.

Sipariş Veren: Üyenin adı soyadını yazmak yerine, numarasını yazmak yeterli ve sağlam çözümüdür. Bunun anlamlı olması için bu alan ile Üye kodu arasında ilişki olmalıdır.

Sipariş Verdiği Ürün: Ürün detayını doldurmak yerine barkod numarasını yazmak yeterlidir. Yine iki alan arasında ilişki olmalıdır

Bu Bölümde Ne Öğrendik Özeti

Bu derste Veri Tabanı Tasarımında önemli yeri olan veri modeli anlatıldı. Veri tabanında kaydı tutulacak varlıklar, özellikler ve varlıklar arasındaki ilişkilerin kavramsal olarak veri modelinin nasıl geliştirildiği açıklandı. Konuya ilgili temel kavramlar açıklandı. Daha sonra bu modellerin tablolara nasıl dönüştürüldüğü çeşitli örneklerle gösterildi.

Bölüm Soruları

1) Ülkeler ile şehirlerarasındaki başkentlik ilişkisi ne tür bir ilişkidir?

- a) Bire-Bir ilişki (1: 1)
- b) Bire-Çok ilişki (1: M)
- c) Çoka-Bir ilişki (M: 1)
- d) Çoka -çok ilişki(N: M)
- e) Hiçbiri

2) Çok Değerli Özellikler tabloya nasıl dönüştürülür?

- a) N-M ilişkiler için uygulana kural bu durumda da aynen uygulanır.
- b) 1-N ilişkiler için uygulana kural bu durumda da aynen uygulanır.
- c) 1-1 ilişkiler için uygulana kural bu durumda da aynen uygulanır.
- d) İlişkide belirtilen çok değerli özellikler mevcut tabloya yeni alanlar olarak eklenir.
- e) Çok değerli her bir özellik için yeni bir tablo oluşturulur.

3) Bir firmada çalışanlar varlık olarak düşünülebilir. Çalışanlardan birisi ayrıldığında direk olarak çocukları da veri tabanından silinir.

- a) Güçlü Varlık
- b) Zayıf Varlık
- c) Bağımsız Varlık
- d) Türetilmiş Varlık
- e) Sanal Varlık

4) Verileri mantıksal düzeyde düzenlemek için kullanılan yapılar ve kavramlar bütünü.

Doğru () Yanlış ()

- 5)** Mobilya üreten bir firmanın tüm bilgilerini içerecek bir E-R modelini çiziniz.
- 6)** Ekmek üreten bir fırının tüm bilgilerini içerecek bir E-R modelini çiziniz.
- 7)** Uzaktan eğitim yapan bir Üniversitenin tüm bilgilerini içerecek bir E-R modelini çiziniz.
- 8)** Öğrencilerin almış olduğu dersler ve bu derslere ait vize ve final notlarının bulunacağı E-R diyagramını çiziniz.
- 9)** Bir sanal alışveriş sitesi için geliştirilen veri tabanında ürünler, kategoriler, tedarikçi firmalar, müşteriler ve siparişler varlık kümelerini, bu kümeler arasındaki ilişki kümelerini (ilişki türlerini de belirterek) ve size göre bu kümelerin sahip olması gereklilikleri de gösterecek şekilde varlık ilişki modeli çizelgesini oluşturunuz.
- 10)** Varlık-İlişki modelindeki Güçlü/Zayıf ve Baskın/Bağımlı varlık kümelerini açıklayınız. Aralarındaki fark nedir? Zayıf varlık kümeleri nasıl güçlendirilir? Örnek veriniz.

CEVAPLAR

1-A, 2-D, 3-B, 4-Doğru

Cevap 10:

A ve B varlık kümeleri arasında birden-bire, ya da A'dan B'ye birden-çoğa bir R ilişkisi varsa ve bir b'nin varolması bu b ile bir a arasında r ilişkisinin kurulmuş olmasına bağlı ise (r ilişkisi yüzünden bir a'ya bağlı olmayan b'ler var olamıyor); b a'ya varolma bağımlıdır denir. Bu durumda; a baskın (dominant) varlık, b ise bağımlı (subordinate) varlık olarak nitelenir. Eğer "bölmü belli olmayan öğrenci bulunamaz" kuralı geçerli ise, bölüm baskın varlık, öğrenci ise bağımlı varlıktır.

Eğer bir varlık kümесinin niteliklerinden en az bir anahtar oluşturulabiliyorsa bu varlık kümese güçlü varlık kümeleridir. Eğer bir varlık kümесinin niteliklerinin tümü alınsa bile bir anahtar oluşturulmuyorsa bu varlık kümese zayıf varlık kümeli denir. Türkiye'deki tüm lise öğrencilerinin bilgilerini içeren ÖĞRENCİ varlık kümeli zayıf bir varlık kümeleridir. Çünkü farklı liselerde öğrenci numarası, adı ve soyadı aynı olan öğrenciler bulunabilir. ÖĞRENCİ varlık kümeli ile LİSE varlık kümeli arasında bir OKUYAN ilişkisi kurulursa, öğrencileri birbirinden ayırt etmek için kullanılan ÖĞR_NO niteliğine, LİSE varlık kümescinin anahtarı olan LİSE_KODU eklenerek zayıf varlık kümeli güçlendirilmiş olur.

4. İLİŞKİSEL VERİ TABANI

Bu Bölümde Neler Öğreneceğiz?

- 4.1 İlişkisel Veri Tabanın Genel Yapısı
- 4.2 Kısıtlamalar
- 4.3 İlişkisel Veri Tabanının Temel Öğeleri
 - 4.3.1 Tablolar
 - 4.3.2 Sütunlar
 - 4.3.3 Değerler
 - 4.3.4 Anahtarlar
 - 4.3.5 Şemalar
 - 4.3.6 İlişkiler

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** İlişkisel veri tabanı nedir
- 2)** İlişkisel veri tabanın genel yapısı nasıldır?
- 3)** İlişkisel veri tabanının temel öğeleri nelerdir?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın Nasıl Elde Edileceği veya Geliştirileceği
	İlişkisel veri tabanını bilir.	Anlatım, Soru-Cevap, Tartışma
	İlişkisel veri tabanını geliştirebilir.	Alıştırma ve Uygulama, Örnek Olay
	Tablolar arası ilişkileri çıkarabilir.	Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

Anahtar Kavramlar

- İlişkisel Veri Tabanını
- İlişkisel Veri Tabanı Elemanları

Giriş

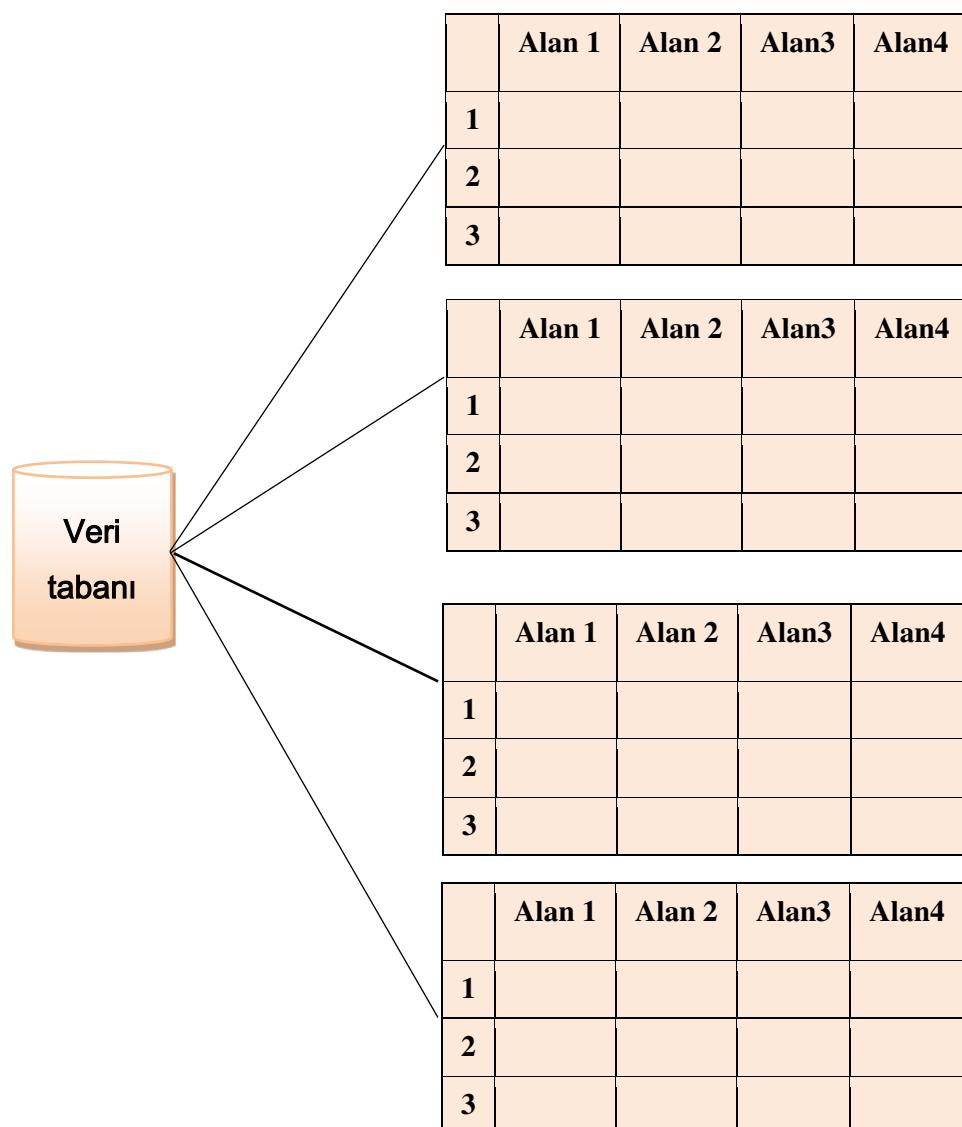
İlişkisel veri tabanı, birbirinden farklı tablolara yerleştirilmiş olan verilerin birbirleri ile belirli alanlara göre ilişkilendirilerek düzenlenen veri tabanlarıdır. İlişkisel Veritabanı Sistemleri büyük miktarlardaki verilerin güvenli bir şekilde tutulabildiği, bilgilere hızlı erişim imkânlarının sağlandığı, bilgilerin bütünlük içerisinde tutulabildiği ve birden fazla kullanıcıya aynı anda bilgiye erişim imkânının sağlandığı programlardır. İlişkisel veritabanı günümüzde en yaygın kullanılan ilişkisel veritabanı sistemlerinden biridir. En çok kullanılan ilişkisel Veritabanı Yönetim Sistemlerine (VTYS) Oracle, MS SQL Server, Sybase, Informix, MySQL gibi veritabanı yönetim sistemlerini örnek olarak verebiliriz.

4. İLİŞKİSEL VERİ TABANI

İlişkisel veri tabanı modeli ilk defa 1970 yılında Dr. E.F. Codd tarafından ortaya atılmıştır. Model, veri tabanına kayıt edilmiş bilgilerin belli kurallara uymasını kapsar. İlişkisel veri modeli İlişkisel veri tabanı dillerini ortaya çıkarmıştır.

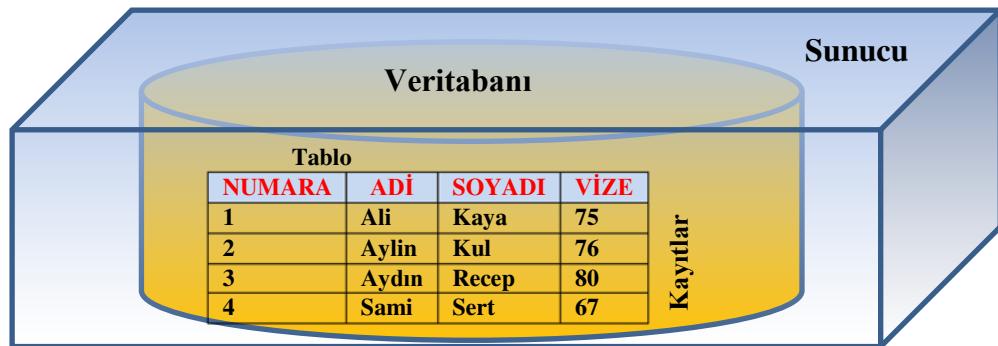
4.1 İlişkisel Veri Tabanının Genel Yapısı

Excel gibi basit bir Elektronik Tablolama programı, verileri genelde düz bir dosya olarak başvurulan yalnızca tek bir tabloda depolayabilir. Bu basit Veri Tabanları “Düz Veri Tabanı” olarak adlandırılır. Günümüzde hemen tüm VTYS’ler ilişkisel veri modelini kullanırlar. Bu model verileri birden çok ilişkili tablolarda tutabilir ve böylece “İlişkisel Veri Tabanı” olarak adlandırılan veri tabanlarını oluştururlar.



Şekil 1 Veri Tabanının Yapısı

Burada bir veri tabanının hangi parçalardan oluştuğunu öğreneceğiz. Tablolar veri tabanlarının temelidir, çünkü tablolar bilgileri organize olmuş bir şekilde tutar ve bizim bilgilere ulaşmamızı sağlar. Bir veri tabanı en az bir tablodan oluşur (Şekil 1). Tablo, verileri düzgün kayıtlar şeklinde belirli bir düzene göre kalıcı ortamlarda bir veri tabanında bir isim altında saklayan, veriler üzerinde SQL yardımıyla ekleme, silme, güncelleme ve listelemeabilen bir veri tabanı nesnesidir. Veri tabanları günümüzde sunucularda oluşturulmaktadır. Bu nedenle veritabanının genel bir çerçevesi Şekildeki gibi olmaktadır (Şekil 3).



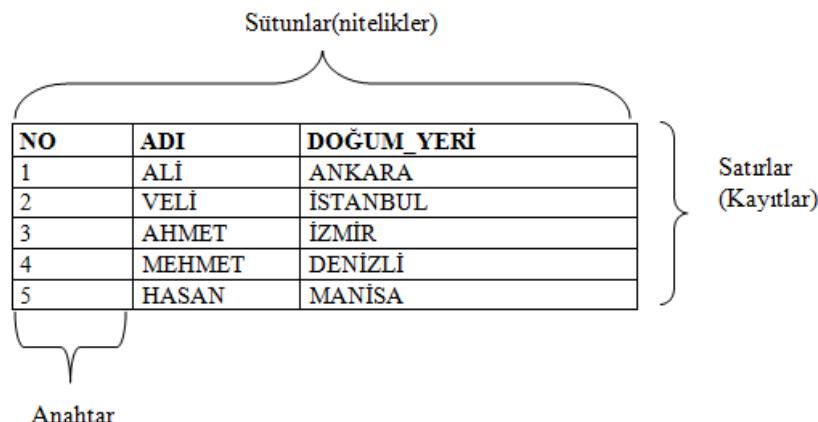
Şekil 3 Sunucu-Veri Tabanı-Tablo-Kayıt İlişkisi

4.2 Kısıtlamalar

Veri tabanında depolanan bilgiler arasında bir bütünlük olması yani verilerin birbirleri ile uyumlu olması gerekmektedir. Büyüklük kısıtlamaları, veri tabanında depolanacak verileri kısıtlayan koşullar olarak adlandırılır. Bir veritabanına kısıtlamalar uygulanarak sadece geçerli verilerin depolanması sağlanır.

Anahtar Kısıtlamaları: Büyüklük kısıtlamalarının sağlanmasında anahtar kısıtlamaları önemli bir role sahiptir. Herhangi bir tablodaki her bir satır için kullanılan anahtarın tek olması gerekmektedir. Aksi takdirde kayıtlar arasında tutarsızlıklar meydana gelebilmektedir.

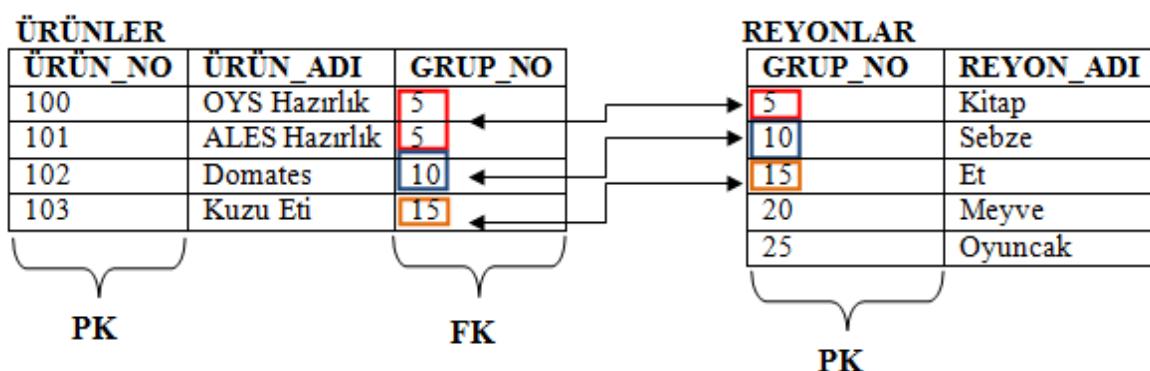
Birincil anahtar(primary key) veya **yabancı anahtar(foreign key)** türlerinden birisi seçilerek kısıtlamaların gerçekleştirilmesi sağlanmaktadır.



Şekil 4: Birincil Anahtar

Birincil Anahtar (Primary Key) Kısıtlamaları: Bir tablonun birincil anahtarı, tabloda depoladığınız her satırı benzersiz şekilde tanımlayan bir veya daha çok alandan oluşur. Genellikle, birincil anahtar olarak işlev gören bir TC kimlik numarası gibi benzersiz bir tanımlayıcı vardır. Günlük hayatımızda da, TC Kimlik Numaramız, illerin plaka ve posta kod numaraları bağlı bulundukları ülkelerde benzersiz numaralardır birincil anahtarları başlarındaki sayısal ifadelerdir. Birincil anahtarlar hiçbir zaman NULL(bos) veya birbiri ile aynı olan değerleri içeremez.

Yabancı Anahtar (Foreign Key) Kısıtlamaları: Tablo içerisindeki verilerin birbirleri ile iletişim kurabilmeleri amacıyla kullanılan anahtarlardır. Birincil anahtarlar hiçbir zaman NULL(bos) veya birbiri ile aynı olan değerleri içeremezken, yabancı anahtarlar birbirleri ile aynı olan değerler içerebilirler. Bir tabloda birden fazla yabancı anahtar kullanılabilir. Kısacası yabancı anahtar, bir tabloya girilebilecek verileri başka bir tablonun herhangi bir alanında yer alabilecek veriler ile sınırlandırmak ve ilişkilendirmek için kullanılır. Yabancı anahtara, başka bir tablonun birincil anahtarıdır da denilebilir.



Şekil 4: Yabancı Anahtar

Veri Kısıtlamaları Tablo tasarlarken kullanılan verilerin tutarlığını sağlamak ve ne tür değerlere sahip olabileceğini belirlemek için de kısıtlamalar getirilebilir. Veri kısıtlamaları sırasında kullanılan bazı kısıtlamalar Not Null, Default, Unique ve Check'tir.

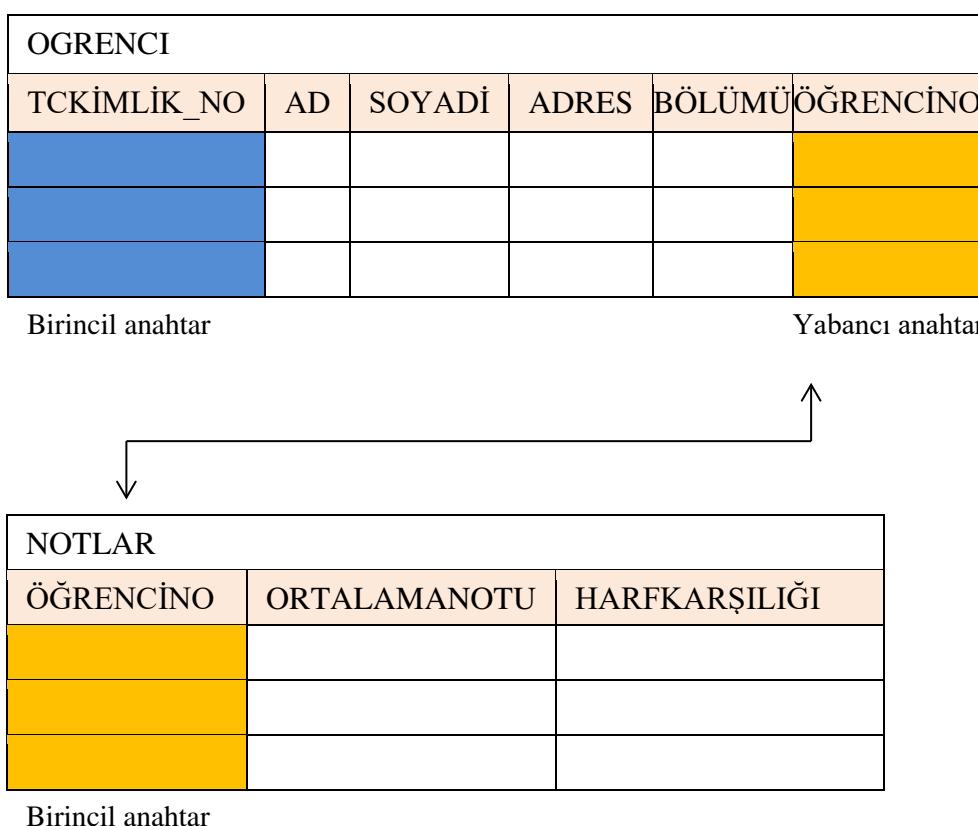
Not Null Kısıtlaması: Veri girişi yapılacak bir tablodaki sütunun değer alıp (NULL) almaması (NOT NULL) gerektiğini belirlemek için kullanılan kısıtlamadır.

Default Kısıtlaması: Veri girişi sırasında bir alanın alabileceği varsayılan bir değer atamak için kullanılır.

Unique Kısıtlaması: Tablodaki bir alana girilen verinin tekrarsız olmasını sağlamak için kullanılır.

Check Kısıtlaması: Kontrol kısıtlayıcı olarak da adlandırılır. Veri girişlerinin belirtilen kriterlere göre yapılmasını sağlar. Örneğin kişinin T.C. Kimlik numarası girilirken 11 haneden fazla değer girilmesi engellenebilir.

Örnek: Öğrenci Bilgilerinin olduğu OGRENCI tablosu, TCKİMLİK_NO, ADI, SOYADI, NOSU, ADRES, BOLUMU alanları ile, öğrencinin NOSU, ORTALAMA NOTU Ve HARF KARŞILIĞI 'nın tutulduğu NOTLAR tablosu hazırlanmaktadır. Bu tablolarda birincil ve yabancı anahtar ve tablolar arası ilişki şekildeki gibi oluşturulur(Şekil 5).



4.3 İlişkisel Veri Tabanının Temel Öğeleri

İlişkisel Veri Tabanı altında inceleyeceğimiz alt kavramlar: Tablolar, Sütunlar, Satırlar, Değerler, Anahtarlar, Şemalar ve İlişkilerdir.

4.3.1 Tablolar

Bir veritabanı tablolarda saklanan verilerden oluşur. Bir veritabanı tablolarda saklanan verilerden oluşur. Tablolar verilerin satırlar ve sütunlar halinde düzenlenmesiyle oluşan veri grubudur. Örneğin öğrenci bilgilerini veritabanında saklamak için tablo oluşturulur: Tablo içindeki her bir bilgi kayıt, Sütunlar ise alan olarak isimlendirilir. Örneğin öğrenci bilgileri tablosunda Öğrenci No, ad, soyad ve E-Posta bilgileri yer alabilir.

Öğrenci No	Ad	Soyad	E - Posta
101	Ahmet	Bilgi	ahmet@bilgi.com
102	Deniz	Yumak	deniz@yumak.net
103	Elif	Keskin	elif@keskin.org

4.3.2 Sütunlar

Tablodaki her sütun benzersiz bir ada sahiptir ve farklı veriler içerir. Her sütunun ilişkilendirilmiş bir veri tipi vardır. Örnek tabloda “Öğrenci No” sütunu sayısal veri tipi ile ilişkilendirilmiştirken diğer alanlar metin veri tipi ile ilişkilendirilmiştir.

4.3.3 Değerler

Tablodaki her satır sütunlara karşılık gelen bir dizi değerden oluşur. Her değer, sütunu tarafından belirtilen veri tipinde olmalıdır. Örnek tabloda yer alan “Ahmet”, “yumak”, “101” ve benzeri sayı ve metin katarlarının her biri değer olarak ifade edilir.

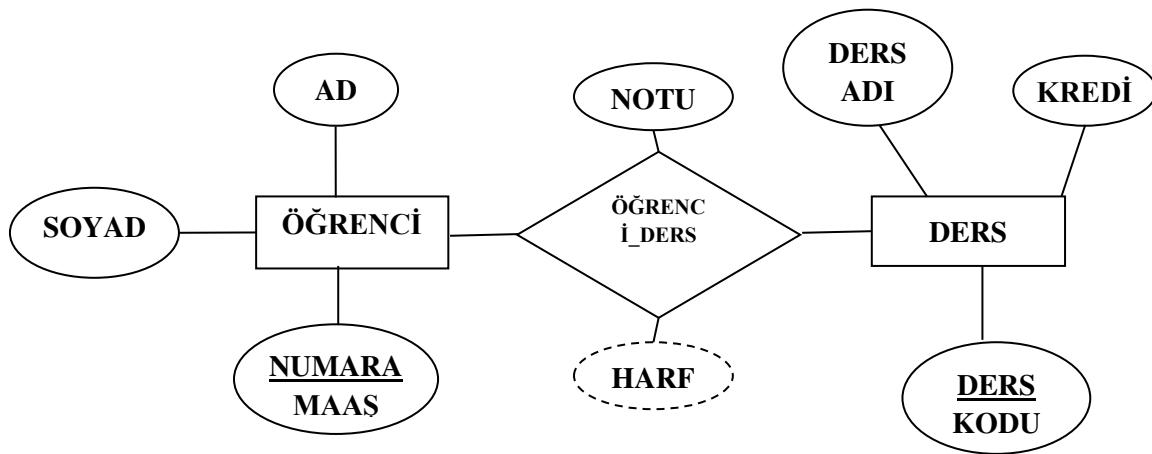
4.3.4 Anahtarlar

Anahtarlar satırları (kayıtları) tanımlayan özel sütunlardır (alanlardır). Farklı veri tipleri anahtar olarak belirlenebilir. Anahtar olarak belirlenen sütunların içinde yer alan her bir satır (değer) benzersizdir. Örnek tabloda yer alan “Öğrenci No” alanı anahtar olarak belirlenmek için uygun sütundur.

4.3.5 Şemalar

Bir veritabanının tüm tablo tasarımlarına, veritabanı şeması denir. Şema veri içermez; veritabanının taslağı olarak kullanılabilir. Şema; tabloları, sütunları ve her tablonun birincil

anahtarları ile birlikte varsa yabancı anahtarları da gösterir. Şemadaki altı çizili terimler, ilgili tablonun birincil anahtarlarıdır. Örnek veri tabanımızın şeması aşağıdaki gibi olacaktır.



Veri Tabanı Şeması

ÖĞRENCİ (Numara, Ad, Soyad)
DERS (Ders Kodu, Ders Adı, Kredi)
ÖĞRENCİ_DERS (Numara, Ders Kodu, Notu, Harf notu)

Şekil 2 Örnek Bir Veri Tabanı Şeması

4.3.6 İlişkiler

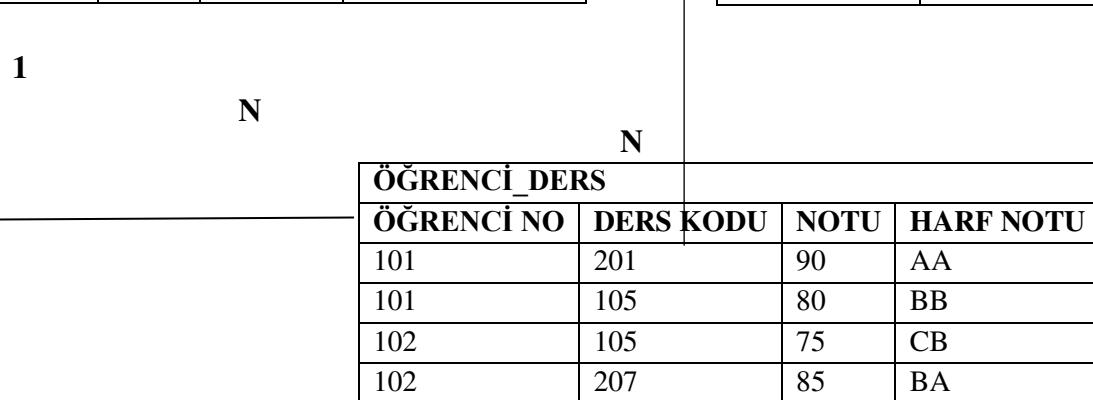
İki tablonun verileri arasındaki ilişkiyi temsil ederler. Bilgileri bir tabloda tutmak hem pratik hem de mümkün olmayabilir. Bu nedenle yapılan işlemler ile ilgili bilgiler çoğu zaman bir tabloda birden fazla tabloda tutulur. Ancak ayrı tablolarda tutulan bu bilgilerden yararlanabilmek için tablolar arası bağlantı veya ilişki kurulması gerekmektedir. Tablolar veya veritabanı dosyaları arasında ilişkiye imkân tanıyan Access gibi veritabanı programlarına İlişkisel Veri Tabanı programı adı verilmektedir. Tablolar arası bağlantı ilişkiler aracılığıyla sağlanır. İlişkiler iki tabloda bulunan ortak alanlarla yapılır. İki tabloyu ilişkilendirmek ilişkisel veri tabanının en temel çalışma şeklidir. Böylece farklı tablolardaki bilgilerin birbirleriyle etkili biçimde kullanılması sağlanmış olur. Veri tutarlılığının sağlanması için ilişkiler çok önemlidir. İlişkisel veri tabanının en önemli özelliği ilişkilerdir. Tablolar arasında ilişkiler tanımlayacağız, böylece herhangi bir tablodaki yapılan değişiklik diğer tabloları da anında etkiler.

Veri tabanı içindeki ilişkilerin tanımlanması tabloların birçoğundan aralanarak veri bütünlüklerinin oluşturulmasını sağlar. Birçok rapor ve veri giriş ekranında birden çok tablodan yararlanılır. İlişkiler kurularak tablolar arasındaki verilere kolaylıkla erişilir.

Tablolar arasındaki verileri bağlamak için çeşitli ilişki türleri vardır. Örneğin Şekil 3 deki gibi öğrenci tablosundaki birden fazla satır ders tablosundaki birden fazla satırla bağlantılıdır.

ÖĞRENCİ			
ÖĞRENCİ NO	AD	SOYADI	E-POSTA
101	Ali	Can	ahmet@bilgi.com
102	Fatma	Kara	deniz@yumak.net
103	Ruhu	Say	elif@keskin.org

DERS		
DERS KODU	DERS ADI	KREDİ
201	Programlama	4
105	Veri Tabanı	4
207	Fizik	3



Şekil 3 Öğrenci ve Ders Tablosu Arasındaki İlişki

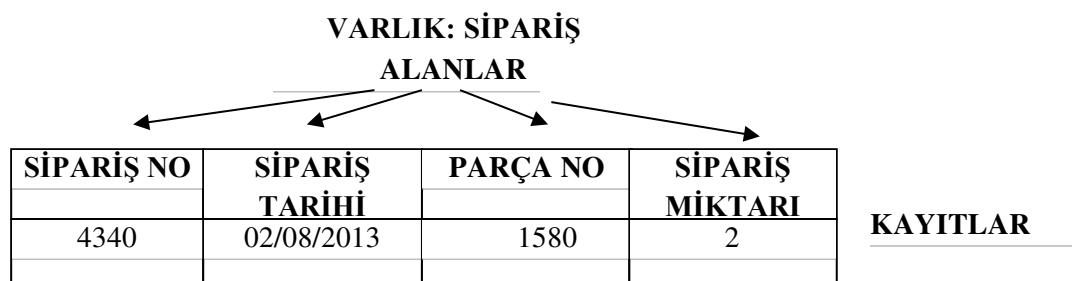
Uygulamalar

ÖRNEK 1

Tablolar verilerin satırlar ve sütunlar halinde düzenlenenmesiyle oluşan veri grubudur. Tablo veri tabanı mantığı içerisindeki en önemli kavramdır. Veritabanı binlerce, milyonlarca parçalanmış bilgiye etkili ve doğru şekilde ulaşmamızı sağlar. Veri tabanı aslında tablolarda saklanan verilerden oluşur. Tablo belli bir konu hakkındaki veriler topluluğudur. Tablodaki veri sütun ve satırlarla ifade edilir. Sütunlar alanları, satırlar kayıtları gösterir. Her sütün, bir bilgi kategorisi olan bir alanı temsil eder. Her satır ise bir öğe için bilgi saklayan kayıttan oluşur.

KAYIT	ALAN					
	OGR NO	AD	SOYADİ	D.TARİH	D.YERİ	TEL
	1	Veli	Özel	01.11.1984	İzmir	32433244
	2	Sami	Gülen	24.05.1986	Ankara	23423423
	3	Sami	Özdemir	06.06.1989	Adana	23423423
	4	Efe	Dinar	11.02.1988	Niğde	34324244

Bir kayıt, bir kişi, bir yer ya da bir olay gibi hakkında bilgi saklanacak olan tek ve bağımsız bir varlığı (entity) tanımlar. Mesela bir satış siparişi dosyasındaki bir “sipariş” tipik bir varlıktır (entity) Şekil 7. Bir varlığı tanımlayan her bir özelliğe alan denir. Örneğin, sipariş no, sipariş tarihi, sipariş miktarı, vb.



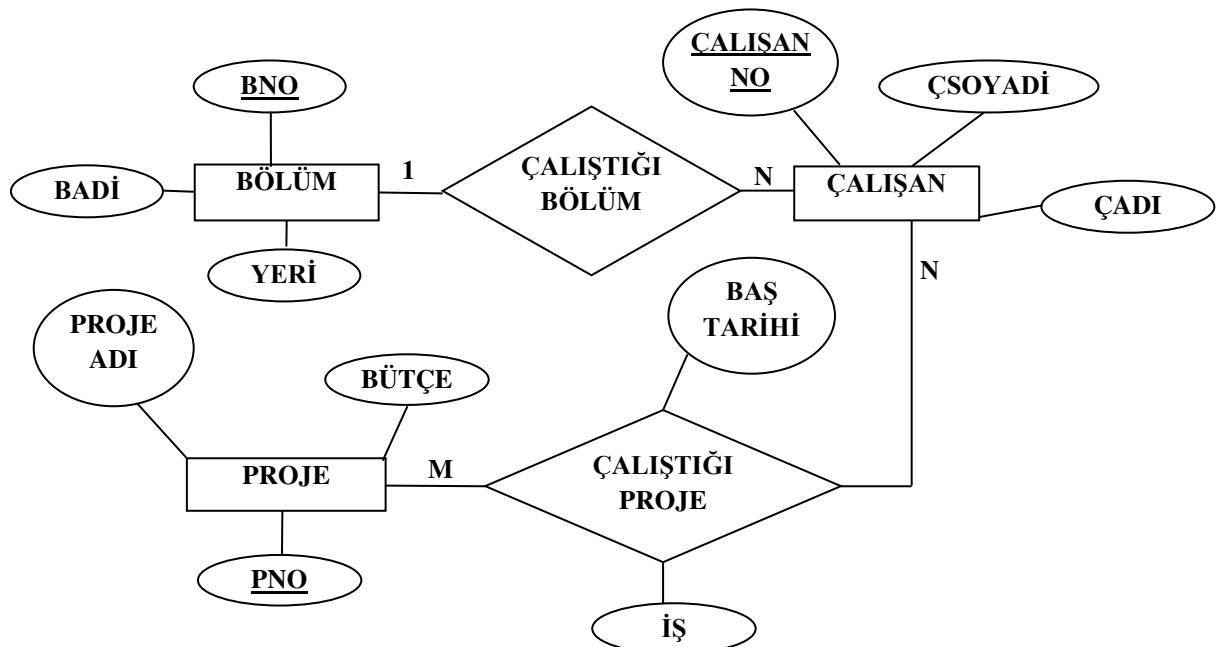
ANAHTAR ALAN

Şekil 7 Veri Hiyerarşisi

Bu alanların alabileceği spesifik değerler sipariş kaydını teşkil eden alanlarda bulunabilir. Bir dosya içindeki her bir kayıt, o kaydın çağrılmamasında kullanılabilcek en az bir tane tanımlayıcı alan içermelidir. Bu tanımlayıcı alana anahtar alan adı verilir. Bu örnekte sipariş numarası anahtar alandır çünkü her bir siparişe kendine özgü bir tanımlayıcı numara verilir.

ÖRNEK 2

Bu örnek bölümleri ve çalışanları ile bir şirketi temsil etmektedir. Her çalışan yalnızca tek bir bölüme aittir. Bir bölümde bir ya da daha çok sayıda çalışan bulunabilir. Çalışanların görevleri projeler üzerinedir. Her biri aynı zamanda birkaç projede çalışır ve projede çalışan birden çok çalışan bulunmaktadır. Buradan hareketle önce veri tabanında oluşturulacak varlıklar belirlemeye çalıştığımızda bunların; Bölüm, Çalışan ve proje olduğu görülür. Bu varlıkların özellikleride göz önüne alındığında aşağıdaki varlık ilişki şeması oluşturulur.



Şekil Varlık İlişki Diyagramı

İlişki Şemaları

Bölüm (Bno, Badi, Yeri)

Çalışan (ÇalışanNo, Çadi, Çsoyadı, Bno)

Proje (ProjeAdı, Pno, Bütçe)

Çalıştığıproje(ÇalışanNo, Pno, Baş Tarihi, iş)

Uygulama Soruları

- 1)** İlişkisel Veri Tabanının Temel Öğelerinin her biri için bir uygulama örneği geliştiriniz?

Bu Bölümde Ne Öğrendik Özeti

Bu derste ilişkisel veri tabanı ve temel elemanları üzerinde duruldu.

Bölüm Soruları

1) Bir tablodaki bir sütuna ait verilerin başka bir tablonun sütunundan getirilmesini sağlayan anahtar aşağıdakilerden hangisidir?

- a) Birincil Anahtar
- b) Yabancı Anahtar
- c) Sütunlar
- d) Unique Kısıtlaması

2) Aşağıdakilerden hangisi “birincil anahtar” alanı için uygun değildir?

- a) Doğum Tarihi
- b) T.C. Kimlik Numarası
- c) Müşteri Numarası
- d) Bir Otomatik Sayı Alanı

3) Veri girişlerinin belirtilen kriterlere göre yapılmasını sağlayan kısıtlayıcı aşağıdakilerden hangisidir?

- a) Not Null Kısıtlaması
- b) Default Kısıtlaması
- c) Unique Kısıtlaması
- d) Check Kısıtlaması

4) Bir veri tabanında mutlaka bulunması gereken nesne hangisidir?

- a) Tablo
- b) Form
- c) Sorgu
- d) Rapor

5) Tablolar ile ilgili hangisi yanlıstır?

- a) Tabloda aynı bilgi tekrar eder.
- b) Birden fazla tabloda aynı bilgi bulunabilir.
- c) Her tablonun birincil anahtarı vardır.
- d) Alan türleri istediğimiz gibi seçilebilir.

6) Aşağıdaki ilişkide(tabloda) kaç tane alan kaç tane kayıt vardır?

Nitelik İsimleri →

Değer Alanları (Kayıtlar)

A	B	C
a	2	x
a	2	y
a	3	y
b	1	x
c	1	y
c	3	x

7) Veri tabanında depolanacak verileri kısıtlayan koşullar ne denir?

CEVAPLAR

1-B, 2-A, 3-D, 4-A, 5-C, 6- 3 alan, 6 kayıtlı bir tablo olarak görülebilir.

5. NORMALİZASYON

Bu Bölümde Neler Öğreneceğiz?

- 5.1 Veri Tabanı Tasarımında Yapılması Gereken Hususlar
- 5.2 İlişkisel Veritabanları ve Normalleştirme
- 5.3 Normalleştirme menin Amaçları
 - 5.3.1 Veri Bütünlüğünü Sağlamak
 - 5.3.2 Uygulamadan Bağımsızlık
 - 5.3.3 Performansı Artırmak
- 5.4 Veritabanı Normalizasyonu Kuralları
- 5.5 Normalizasyon Kurallarının Uygulama Örneği
- 5.6 İşlevsel Bağımlılıklar
 - 5.6.1 Tam İşlevsel Bağımlılık ve Kısmi Bağımlılık (Partial Dependency)
 - 5.6.2 Dolaylı Bağımlılık (Transitive Dependency)

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Normalizasyon kavramının ne olduğunu ve neden bir formun normalize edilmesi gerektiğini araştırınız?
- 2)** Kaç tür normalizasyon biçimini vardır?
- 3)** Veri tabanı tasarımda hangi normalizasyon biçimini yeterlidir?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Normalizasyonu bilir.	Anlatım, Soru-Cevap, Tartışma
	Normal formlara dönüşümü gerçekleştirebilir.	Alıştırma ve Uygulama, Örnek Olay
		Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

Anahtar Kavramlar

- Normalizasyon
- Normal Olmayan Form
- 1. Normal Form
- 2. Normal Form
- 3. Normal Form

Giriş

Normalizasyon, bir veritabanındaki verileri düzene koyma işlemidir. Normalizasyon, veri tabanlarında çok fazla sütun ve satırdan oluşan bir tabloyu tekrarlardan arındırmak için daha az satır ve sütun içeren altkümelere ayrıştırma işlemidir. Normalizasyon, aynı zamanda “ilk taslak” veri tabanı tasarımının üzerinde revizyonlar yaparak, taslağı son haline yaklaştırmanın yöntemlerden birisidir. Bu çerçevede en iyi tasarımını gerçekleştirmek için dikkat edeceğimiz kurallara Normalizasyon işlemleri diyoruz.

Normalizasyon, birincil anahtarları ve işlevsel bağımlılıkları kullanarak ilişkileri analiz etme tekniğidir. Normalizasyon kuralları bir tablo içerisinde yer alacak kaydın nelerden oluşmasına karar vermeye yarar. İlişkisel veritabanı tasarımını aşamasında verinin tekrarlanması, kaybını veya yetersizliğini önlemek için Normalizasyon işlemi önem arzeder. İlişkisel veritabanı tasarımında amaç, veri tekrarını azaltan ve veri tutarlığını yükselten bir yapının oluşturulmasıdır. İşte bunun için yapılması gereken işlemler Normalizasyonla sağlanır.

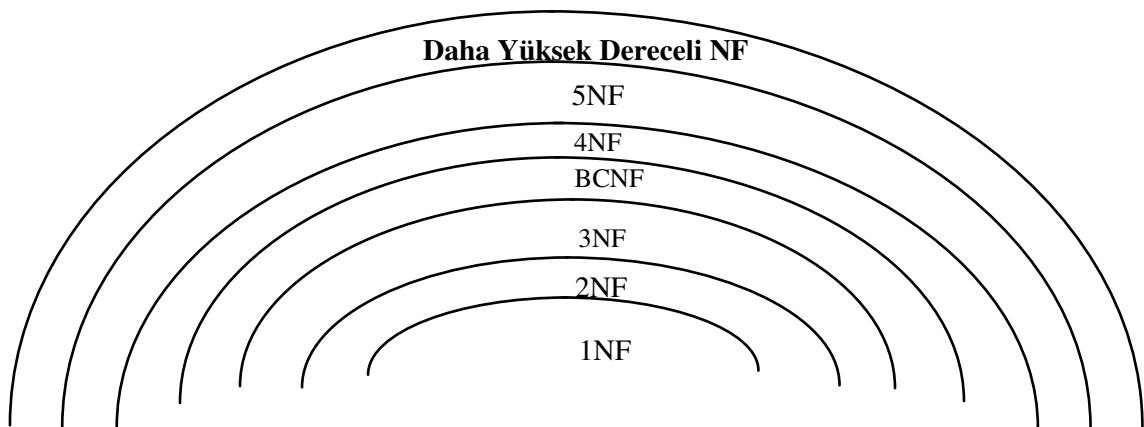
5. NORMALİZASYON

İlişkisel veri tabanları tasarlarken verilerin gereksiz tekrarını, bilgilerin kaybını önlemek amacıyla normalizasyon işlemi uygulanması gereklidir. Normalizasyon işlemi uygulanarak ilişkilerin normal forma getirilmesi sağlanır. Normalizasyon, taslak veri tabanı üzerinde birtakım işlemler yapılarak taslağı son haline yaklaştırma yöntemidir. İyi tasarlannamış olan bir veri tabanında güncelleme, ekleme veya silmeden kaynaklanan sorunlar nedeniyle birtakım kullanım zorlukları meydana getirmektedir. Normalizasyon, veri tabanı tasarımında bu tür sorunların da ortadan kaldırılmasını sağlayan bir süreçtir.

Normalizasyon, veri tabanı tasarım aşamasında veri tekrarını, veri kaybını veya veri yetersizliğini önlemek için gerçekleştirilir. İlişkisel veritabanı tasarımında amaç, veri tekrarını azaltan ve veri tutarlığını yükselten bir yapının oluşturulmasıdır. Bunun için yapılması gereken işlemler bütününe Normalizasyon adı verilmektedir. En iyi tasarımını gerçekleştirmek için dikkat edeceğimiz kurallara Normalizasyon işlemleri olarak isimlendiriyoruz.

Normal Formlar

Normal formlar normalleştirmenin derecelerini veren formlar olup 1NF, 2NF, 3NF, BCNF, 4NF ve 5NF şeklindedir. En dışta Daha Yüksek Dereceli NF ve en içerde 1NF olmak üzere her üst form aynı zamanda alt formun özelliklerini de taşır. Bir başka tanımlamada ise 1.NF, 2.NF, 3.NF, 4.NF, 5.NF hatta 6.NF, BOYCE-CODD, DOMAIN/KEY olmak üzere 8 adet normalizasyon kuralı vardır.



Şekil1 Normal Formlar

Çeşitli kaynaklarda Normalizasyon kuralları birçok farklı kurallar şekilde tanımlansa da temel olarak 3 Normalizasyon kuralı yaygın olarak kullanılmaktadır. İyi bir veritabanı tasarımını yapabilmek için yetenek, bilgi ve tecrübe çok önemlidir. Ancak öncelikle, ilişkisel veritabanının tanımını ve bununla ilgili 3 Normalizasyon kuralını çok iyi bilmek gereklidir. Aslında bu normal form olayına veritabanı öğrenme aşamasındayken dikkat ediliyor, daha sonra veritabanı analizi yaparken tecrübelerimizle otomatik olarak normalize yapmaya başlıyoruz. Normalizasyon, tasarım aşamasında yol göstermek yerine hangi şartlara uygun tasarım yapılması gerektiğini anlatır. Bazen, bu kurallardan vazgeçmek durumunda olunabilir

ancak, veri tabanında saklanacak verilerin hacmi arttıkça yani veri tabanı büyündükçe bu kuralların daha sıkı uygulanması gereklidir.

Normalleştirme, aynı zamanda “ilk taslak” veri tabanı tasarımının üzerinde revizyonlar yapmanın yolu, taslağı son haline yaklaştırmanın yöntemlerden birisidir.

5.1 Veri Tabanı Tasarımında Yapılması Gereken Hususlar

Bir veri tabanını nasıl tasarlanması gerekiği oldukça önemlidir. Başlangıçta yanlış tasarlanan bir veri tabanı ile yapılan projede sonradan yapılacak düzenlemelerle geri dönüş yapılamaz. Öncelikle ihtiyaç analizi yapılarak tüm paydaşların gereksinimleri birleştirilmelidir. Bir veritabanı oluşturabilmek için şu konular iyi anlaşılmalıdır. Bunlar; veri tabanında saklanacak olan verilerin türü, veriler arasındaki ilişkiler, veriler nasıl ve nerede kullanılacak ve firma çapında tüm verileri yönetebilmek için organizasyon yapısı nasıl değişecektir. O nedenle veri tabanı tasarımları yapılırken aşağıdaki maddelere uyularak yapılması gereklidir.

1. Varlıklar Tanımlanır: Varlık, çeşitli özellikleri bulunan bir varlıktır: Herhangi bir proje de öncelikle Varlıklar tanımlanır. Birkaç proje için Varlıklara örnek verilecek olunursa; Üniversite Sisteminde; Öğrenciler, hocalar, dersler, derslikler, projeler vb. olabilir. Üretim sisteminde; Ürünler, müşteriler, siparişler, teslimat, fatura bilgileri, üreticiler, tedarikçiler, dağıtıcılar olabilir. Futbol Liginde; Takımlar, sahalar, oyuncular, hakemler, antrenörler vb. olabilir. Kütüphane sisteminde; Kitaplar, üyeleri, türler, ödünç hareketleri olabilir.

2. Her Varlık İçin Bir Tablo Oluşturulur: Her Varlık için bir tablo oluşturulur ve her bir tabloya içereceği veriyi en iyi anlatan bir isim verilir. Tüm proje bitirilinceye kadar bu tablolar üzerinde muhtemel değişiklikler yapılabilir.

3. Her Bir Tablo İçin Bir Anahtar Alan Seçilir: Veri tabanındaki herhangi bir veriye erişilmeden önce tabloya erişilir. Bir veri tabanında üzerinde en çok işlem yapılan Varlık grubu genellikle tablolardır. Bu aşamada, tabloda yer alacak her bir kaydı bir diğerinden ayırbilecek bir sütuna ihtiyaç duyulur. Örneğin araçlar ile ilgili bir tablo yapılrken, plakalar anahtar alan olarak belirlenebilir. Öğrenci tablosu için, öğrenci numarası doğal bir anahtar alandır.

4. Varlıkların Gerekli Her Bir Özelliği İçin Tabloya Bir Sütun Eklenir: Tablo adları tanımlandıktan ve anahtar adları belirlendikten sonra, tablolara sırasıyla adını veren Varlıkların her bir özelliği için bir alan (sütun) eklenir. Örneğin, kitap için; Kitap no, ISBN no, kitap adı, yazarı, türü, fiyatı, baskı yılı.

5. Tekrarlayan Varlık Özellikleri İçin Ek Tablolar Oluşturulur: Tabloda veri tekrarı olacağsa, bu durumda eldeki tablonun en az bir tabloya daha ayrılması gereklidir. Örneğin, her bir kitap için tür belirledik ama bir kitap hem kişisel gelişim kategorisine hem de hikâye kategorisine girebilir. Bu türden bir sorunu çözmek için ilk akla gelen şey, Kitap tablosunda tür alanı için 2.sütun daha eklemek olabilir. Bu tabloya 2.Tür ve 3.Tür diye iki sütun alanı daha eklemek. Ama çoğu kitap bir tek türdendir ve bu kitap için eklenen 2 alan hep boş kalacaktır. Öte yandan, 4.türe birden giren bir kitap olduğunda 4.tür bilgisi nereye yazılacaktır? Aynı alana mı? Ya da dört adet bölüm mü açılacak? Bunlar, veritabanı tasarımının doğasına terstir. Bu

durumda, türler diye bir yeni tablo oluşturup, bir de kitap türler diye 2.tablo'yu oluşturuktan sonra bu türden bilgileri burada tutmak gerekecektir.

6. Anahtar Alana Bağlı Olmayan Alanlar Belirlenir: İlişkisel veritabanında, tablodan herhangi bir tek kayda erişmek için mutlaka bir farklı özellik sağlanmalıdır ve bu özellik de anahtar alan tarafından sağlanır. Ancak bazen, anahtar alan ile aynı satırda yer aldığı halde, anahtar alan ile birebir ilişkisi olmayan bir alan yer alabilir. Bu türden alanların elimine edilip ayrı tablolara ayrılması gereklidir. Örneğin, ödünç tablosu ele alınacak olursa, ödünç verilen her kitap için ödünç alanın adresi de bilinmek istenirse, bu ödünç tablosuna yazılmalıdır. Çünkü ödünç tablosunun birincil anahtar alanı oduncNo'dur ve bu alan, ödünç verme işlemi ile ilgilidir. Oysa ödünç alanın adresi, ödünç alan kişinin kendisine bağlı bir özelliktir. Bu kişinin her aldığı kitap için adresini tekrar yazmaya gerek yoktur. Aynı şekilde otomasyon içerisinde başka yerlerde de bu kişinin adres bilgilerine muhtemelen ihtiyaç duyulabilir çünkü adres, üyenin bir özelliğidir. Ödünç verilen kitabın adresi öğrenilmek istenildiğinde, üyeler adında bir tablo daha açılıp, burada herkesin adres bilgisi tutulmak zorunda kalınır. Ödünç tablosunun ise, oduncAlan bilgisi olarak, Üyeler tablosunun birincil anahtar alanına bir bağlantı (yabancı anahtar) içermesi daha doğru olur.

7. Tablolar Arasındaki İlişkiler Tanımlanır: Her biri bir Varlığa dair özellikleri barındıran tabloların tümü göz önüne alınır ve birbirleri ile olan ilişkileri tanımlanmaya çalışılır. Örneğin kitabı ödünç verebiliriz. Bu durumda, ödünç tablosu ile Kitap tablosu ilişkili olacaktır. Kitap üyelerde ödünç verilir. Bu durumda, ödünç ile üyeleri arasında da bir ilişki vardır. Türler ile Kitap arasında bir ilişki vardır, bir kitabın en az bir türde dahil olması gereklidir. İlişkili her iki tablo bir birincil alan ve bir yabancı anahtar alan üstünden birbirine bağlanır.

5.2 İlişkisel Veritabanları ve Normalleştirme

En basit yol olarak, veritabanları veriyi tablolar içine yerleştirerek saklarlar. Bu tablolar satır ve sütunlardan oluşmaktadır. Burada bahsedilen her satır bir kayıt, her sütun ise alan olarak isimlendirilmektedir. Bu isimlendirme pratikte tabloların bu şekilde kullanılmasından kaynaklanmaktadır. Şimdi gerçek dünyadan bir uygulama yapalım ve bir sorunun çözümüne nasıl yaklaştığımızı gösterelim:

Problem: Web sitemizin bütün kullanıcılarını kayıt etmek istiyoruz. Bunu yapmanın en kolay ve belki de en temel yolu bir tablo oluşturup her kullanıcıyı bu tablonun içeresine kayıt etmektir. Kullanıcı bilgileri ve sitemizdeki girdiği sayfaya ait bütün bilgileri bir tek tabloda açacağımız alanların içine yerleştirmek zor değildir. Bu durumda oluşturacağımız tablo aşağıdaki gibi 6 alandan oluşan bir tablo olsun.

KULLANICI ADI	GERÇEK ADI	SOYADI	KULLANICI ULKESI	SAYFA	SON ERIŞİM
Hikmet	Hikmet	Gumus	Turkiye	hakkında/index.php	2013-01-26
Oktay	Oktay	Ceylan	Almanya	duyurular/index.php	2013-06-20
Serkan	Serkan	Kaya	USA	dersler/index.php	2013-04-02

Bu şekilde oluşturduğum 6 sütunlu (alanlı) tabloya, örnek olması açısından 3 tanede kayıt (satır) girildi. Hemen hemen istediğimiz bütün ihtiyaçlara cevap veriyor. Böylece her kullanıcının web sitesinde gezerken hangi sayfalara, hangi tarihte uğradıklarını görmüş olacağız.

Yukarıda oluşturduğum veritabanı tablosuna yakından bakmanızı istiyorum. Tablonun yapısından dolayı her kullanıcı için bir kayıt açılması ve her seferinde bir satırı kullanmak zorunluluğu vardır. Bu şekilde aynı kişiye ait kayıtlar bile tekrarlanarak yazılacak ve veritabanı genişledikçe daha verimsiz bir şekilde zor duruma girecektir. Herkes için böyle bir işlem yapmak gereksizdir.

Bu şekilde yapılandırılan bir tablo aynı zamanda ileride çok daha fazla sorun oluşturabilir. "Serkan" isimli kullanıcının USA yerine Türkiye'de yaşamaya karar vermesi durumunda veya kullanıcı adını değiştirmesi halinde ne yapılacak?

Bu durumda tabloda yer alan ve bu kullanıcıya ait her satır teker değiştirilmek zorunda kalacaktır. Tablonun güncellenmesi bu durumda günler veya haftalar sürebilir.

İste konu başlığımız olan "Normalleştirme" işlemi burada devreye girmektedir. Burada anlatılmak istenen, bu işlemlerle veritabanı içindeki tekrarlanan kayıt sayısını azaltmak ve performansı bu şekilde artttırmaktır.

"Normalleştirme" işlemi, verinin birkaç tabloya bölünmesi işlemidir. Bu şekilde aynı verinin tekrarlanması en aza indirilir. Örnek vermek gerekirse yukarıdaki tabloyu, kullanıcı ve erişim bilgilerinin saklandığı iki farklı tabloya bölebiliriz.

Kullanıcılar Tablosu

KULLANICI ADI	GERÇEK ADI	SOYADI	KULLANICI ÜLKESİ
Hikmet	Hikmet	Gumus	Turkiye
Oktay	Oktay	Ceylan	Almanya
Sercan	Sercan	Kaya	Türkiye

Erişim Tablosu

KULLANICI ADI	SAYFA	SON ERİŞİM
Hikmet	hakkında/index.php	2013-01-26
Oktay	duyurular/index.php	2013-06-20
Sercan	dersler/index.php	2013-04-02

Bu şekilde ayrı, bağımsız tablolara bölmek "normalizasyon" işleminin en önemli işidir. Simdi bu işlemin ikinci ve en önemli adımını inceleyelim. Böyle parçalara ayırdığımız tablolar için en az bir tane tekil (unique) alanı seçilmesi zorunludur. Bu alanın seçilmesiyle tablo içerisinde aynı verinin tekrarlanması ve gereksiz alan kaybına son verilmiş olacaktır. Tablo içerisindeki her kaydın farklı olduğunu garanti etmek için anahtar alan oluşturulur. Bu alanda genelde her kaydı birbirinden ayırmak ve tanımlayabilmek için tabloda ID'ler tutulur.

Her tablo için yalnız ve yalnız bir tane anahtar alan alanı tanımlayabiliriz. Örnek vermek ve konuyu kafanızda netleştirmek için tablomuz üzerinde yukarıda anlatılanları tablomuz üzerinde uygulayalım.

Kullanıcılar tablosu için "Kullanıcı Adı" alanını anahtar alan olarak belirleyelim. Bunun sebebi bu alanda yer alacak verilerin tekil olması. Burada akılınızda su soru gelebilir: "Gerçek Adı" alanı da tekil veriler içeriyor onu neden anahtar alan olarak seçmedik? Hemen cevap verelim. Burada veritabanı tasarıımı yapılrken bazı kriterler göz önüne alınmalıdır. Bunlar ne olabilir?

Mesela, "Gerçek Adı" bolumndeki veri her zaman değişebilir. Kişi evlenirse büyük ihtimal soyadı değişecektir. Veya kişi bazı gereksinimlerden dolayı "Gerçek Adı" alanını gizli tutmak isteyebilir vb. Bu durumda bu alana veri girilmeyeceği için ilerde büyük bir sorunla karşılaşılabilir.

Bu örnek için gördüğünüz gibi "kullanıcı adı" çok fazla değişecek gibi görünmüyör. Bizim için bu alanı "primary key" olarak seçip bununla tablodaki verileri istediğimizde bulacak gibi indekslemek ve biricik olmalarını sağlamak mümkün.

Normalizasyon ne kazandırdı?

Şimdi bu yetiklerimizin bize kazandırdıklarına kısaca degeñelim. Çünkü bir tabloyu iki farklı tablo haline getirdik ve bazı belirlemeleri yaptık.

Bu iki tabloyu ayırip onların bir alanını ("kullanıcı adı" alanı gördüğünüz gibi her iki tabloda da mevcut) aynı yapmakla veri tabanını "ilişkili" hale getirmiş olduk. Bu su demek: İlişkili tablolardan alacağımız verileri, karmaşık sorguları gerçekleştirebilmek için kullanabiliriz. Yalnızca kendi başına hazırlanmış bir tablodan, (ilişkisiz tablodan) başka tabloların verilerini kombine bir şekilde kullanmamız imkânsızdır.

5.3 Normalleştirme Meninin Amaçları

5.3.1 Veri Bütünlüğünü Sağlamak

Eğer veri gereksiz yere tekrarlanıyorsa, bu değişik kopyalar, bunlardan habersiz olan uygulama kodları yüzünden bir süre sonra birbirinden farklı değerleri taşımaya başlayabilirler. Bu, doğruluk ve tutarlılık açısından kötü bir sonuçtur. Bu gibi durumlarda ilişkisel veri tabanı yönetim sisteminin otomatik bütünlük mekanizmaları bile işe yaramaz. Düzeltmenin, uygulama seviyesinde yapılması gereklidir. Fakat bu da uygulama programlarını daha karmaşık hale getirecek, dolayısıyla bakımını zorlaşdıracaktır.

5.3.2 Uygulamadan Bağımsızlık

İlişkisel model, uygulamaya göre değil, verinin içeriğine göre kurulmalıdır. Bu sayede veri modeli, üzerinde onu kullanan uygulama değişse bile, daha tutarlı, sabit ve değişmez olarak

kalacaktır. Uygulama programının gereksinimlerinin veri tabanının mantıksal modeli üzerinde minimum etkisi olmalıdır.

5.3.3 Performansı Arttırmak

Yabancı anahtarların haricinde, tamamıyla normalleştirilmiş bir veri tabanı gereksiz yere kopyalanmış veri miktarını en aza indirecektir. Verilerin daha az kopyasının olması saklama kapasitesinin azalmasına ve veri tabanı motorunun arama süresinin azalmasına yol açar. Bu da performansın artması demektir.

5.4 Veritabanı Normalizasyonu Kuralları

Normalizasyon kuralları bir tablo içerisinde yer alacak kaydın nelerden oluşmasına karar vermeye yarar. Genel kabul görmüş 5 normalizasyon kuralı vardır. Burada her bir kuralı tam olarak anlatmak mümkün değildir. Ancak bu kurallar, ilişkisel veritabanının tanımı ile birlikte ortaya konulmuştur. Özet olarak fikir vermesi açısından normalizasyon kurallarına aşağıda yer verilmiştir:

- Bir satırındaki bir alan yalnızca bir tek bilgi içerebilir
- Bir tablo için, anahtar olmayan her alan, birincil anahtar olarak tanımlı alanlara bağlı olmak zorundadır.
- Bir tablo için, anahtarı olmayan bir alan, anahtarı olmayan başka hiç bir alana bağlı olamaz.
- Birincil anahtar alanlar ile anahtarı olmayan alanlar arasında, birden fazla bağımsız bire-çok ilişkisine izin verilmez.
- Tekrarlamaları ortadan kaldırmak için her bir tabloyu mümkün olduğunda küçük parçalara bölmek gerekir.

1. Normalizasyon Kuralı:

Bir satırındaki bir alan yalnızca bir tek bilgi içerebilir. Örneğin Ad ve soyad farklı alanlarda girilmeli. Verileri virgül veya bir başka karakter ile ayrılop aynı alana girilmemelidir. Bu durum ilişkisel veritabanının doğasına terstir. Birden fazla yazarı olan kitap için yazar1, yazar2 ve yazar3 diye alanların açılması ile bu kurala uyulmamış olunur. Böyle bir durumda, ayrıca yazarlar tablosu da oluşturularak kural çiğnenmemiş olur.

2. Normalizasyon Kuralı:

Bir tablo için, anahtar olmayan her alan, birincil anahtar olarak tanımlı tüm alanlara bağlı olmak zorundadır. Ya da anahtar alanın birden fazla alandan olduğu tablolarda, anahtar alanlardan sadece birine bağlı veriler tabloda yer almamalı, ayrı bir tabloya taşınmalıdır. Bunun

tersi de geçerlidir. Yani iki ya da daha fazla tablonun birincil anahtarı aynı olamaz. Böyle bir durum söz konusu ise, bu iki tablo tek tabloya indirilmelidir.

3. Normalizasyon Kuralı:

Bir tablo için, anahtarı olmayan bir alan, anahtarı olmayan başka hiç bir alana bağlı olamaz. Örneğin, kitaplar için cilt tipi adında bir alan eklenip burada da karton kapak için K, deri cilt için D, spiral cilt için S yazılsaydı, bu kodlama, kitap tablosunun birincil anahtarı olan kitapNo alanına bağlı bir kodlama olamazdı. Çünkü bu kodlama bir başka anahtarı olmayan alana bağlıdır. Bunun sonucunda da veri tabanında, karşılığı olmayan bir kodlama yer almış olurdu. Cilt tipi bilgisini kodlu olarak tutan alan aslında cilt tipi açıklaması olan başka bir alana bağlıdır. Bu ilişki başka bir tabloda tutulmalıdır. Bu durumda, cilt şekillerini tutan bir tablo açılması gereklidir. Bu tablonun alanları da ciltTipKodu ve ciltSekli olabilir. Ancak bundan sonra, kitaplar tablosunda ciltTipi adında bir sütun açıp buraya da D, S, K gibi kodlar yazılabilir.

4. Normalizasyon Kuralı:

Birincil anahtar alanlar ile anahtarı olmayan alanlar arasında, birden fazla bağımsız bire-çok ilişkisine izin verilmez. Örneğin, tabloda yer alan bir kitap, hem hikaye kitabı hem de kişisel gelişim kitabı olabilir. (Bu durumda kitabı adı, kişisel gelişim hikâyeleri olurdu her halde) Bu durum Kitap tablosunda nasıl ifade edilebilir? 4. Normal formu sağlamak için, her bağımsız bire çok ilişki için ayrı bir tablo oluşturulması gereklidir. Bu örnekte, türler için yeni bir tablo açılması gereklidir. Tablonun adına türler denilebilir. Daha sonra kitapTurleri diye bir başka tablo daha açılması gereklidir. ‘Kişisel Gelişim Hikâyeleri’ adlı kitap için, öncelikle kitap numarası, Hikaye bölümünün kodunun yer aldığı bir satır; ardından da yine kitap numarası, ardından da kişisel gelişim türünün kodunun aldığı yeni bir satırın daha eklenmesi gereklidir.

5. Normalizasyon Kuralı:

Tekrarlamaları ortadan kaldırmak için her bir tablonun mümkün olduğunca küçük parçalara bölünmesi gereklidir. Aslında ilk 4 kural sonuçta bu işe yarar ancak, bu kurallar kapsamında olmayan tekrarlamalar da 5 normalizasyon kuralı ile giderilebilir. Örneğin, kitaplar için bir edinme şekli bilgisi girilecek sütun eklenmek istenebilir: Bu bölüme girilebilecek bilgiler bellidir: Bağış veya satın alma. Bu bilgiler başka bir tabloda tutulabilir. Böylelikle, kullanıcıların bu alana geliştiğinde bilgiler girmesi engellenmiş olur. Bu da sorgulama esnasında veriler arasında bir tutarlılık sağlar. Bu işlem sonucunda, tutarsızlıklara neden olabilecek ve sık tekrarlayan veriler başka bir tabloya taşınmış olur. Bu tablo için, veritabanı programlamada ‘look-up table’ terimi kullanılır. Ancak, veritabanı normalizasyon kuralları, bir ilişkisel veritabanının tasarılanma aşamalarını değil de ilişkisel veri tabanında yer alacak kayıtların ilişkisel veritabanı ile uyumlu olup olmadığını denetlemeye yönelikir. Özette ilişkisel bir veritabanı tasarıımı şu öğeleri bünyesinde barındırmalıdır; Veri tekrarı yapılmamalıdır, boş yer mümkün olduğunda az olmalıdır, veri bütünlüğü sağlanmalıdır ve veriler, aralarında bir ilişki tanımlanmaya müsaith olmalıdır.

Veri Fazlalığı

Aynı alanda birden fazla veri bulunması ve tablolarda aynı tipte bilgiyi içerecek alanların bulunması mutlak veri fazlalığı sorunudur.

Öğrenci No	Adı Soyadı	Derskodu	Derskodu2
13	Ali Veli	Mat101	Edb101
23	Ahmet Hasan	Mat101	Edb101
25	Mehmet Oğuz	Kmy101	Fiz101
44	Hakan Ak	Kmy101	Fiz101

Aynı tipte bilgiyi içeren alanlar olamaz

Öğrenci No	Adı Soyadı	Derskodu
13	Ali Veli	Mat101, Edb101
23	Ahmet Hasan	Mat101, Edb101
25	Mehmet Oğuz	Kmy101, Fiz101
44	Hakan Ak	Kmy101, Fiz101

Bir alana birden fazla veri girişi olamaz

Burada Derskodu tekrarlayan grup olmakla birlikte, tasarım sırasında tekrarlayan gruplar yer almamalı, tablonun her hücrende tek bir değer bulunmalıdır.

5.5 Normalizasyon Kurallarının Uygulama Örneği

Öğrencilerin aldığı derslerin ilan tablolardında asılan listelerini ele alarak normalizasyon işlemlerini açıklanacaktır. Böyle bir örneği veritabanı yönetim sistemi ortamında gerçeklemeye başlamadan önce ilişkisel veritabanı mantığına uygun bir şekilde tasarlayarak veri tabanımızda hangi tablolarda, tablolar arasında nasıl ilişki olacak, tablolar arasında tutarlılık nasıl sağlanacak v. b. durumları veri tabanımızı gerçeklemeden önce tasarlamanız gerekmektedir.

Normalizasyon Öncesi Öğrencilerin Aldığı Derslerin Listesi

Ogr No	Ogr Adı	Ogr Adres	Ogr Tel	Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri	Belge
47	Murat Keskin	İstanbul	235	A01	Türkçe	3	Sosyal	D1	Katılım
21	Cem Serin	Ankara	412	A01	Türkçe	3	Sosyal	D1	
47	Murat Keskin	İstanbul	235	S01	Fen	2	Teknik	D2	EKC
54	Mert Sert	Sakarya	122	A01	Türkçe	3	Sosyal	D1	Katılım
21	Cem Serin	Ankara	412	S01	Fen	2	Teknik	D2	

1. Normalizasyon kuralı Tablomuzda yer alacak her alan tek bir veriyi göstermelidir.

Örneğimizi incelersek Öğr. Ad sütununda yer alan değerler aslında öğrencinin adını ve soyadını içermektedir. Tasarım sırasında bu alan ad ve soyad olarak iki farklı alan tarafından temsil edilmelidir. Böylece soyad isimli yeni alan oluşturmalıyız.

Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel	Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri	Belge
47	Murat	Keskin	İstanbul	235	A01	Türkçe	3	Sosyal	D1	Katılım
21	Cem	Serin	Ankara	412	A01	Türkçe	3	Sosyal	D1	
47	Murat	Keskin	İstanbul	235	S01	Fen	2	Teknik	D2	EKC
54	Mert	Sert	Sakarya	122	A01	Türkçe	3	Sosyal	D1	Katılım
21	Cem	Serin	Ankara	412	S01	Fen	2	Teknik	D2	

2. Her tabloda o tabloda yer alan kayıtları tek başına temsil edecek bir alan bulunması gerekmektedir.

Okuldaki öğrencilerin okul numaraları, Bir ülkede yaşayan insanların vatandaşlık numarası gibi bilgiler her tabloda kayıtları tek başına temsil edecek eşsiz bir alanın olması gerekmektedir. Bu alanları Birincil Anahtar (Primary key) olarak isimlendiriyoruz. Birincil anahtar bir tabloda sadece bir tane olabilir yani aynı değere sahip ikinci bir birincil anahtar değeri bulunamaz. Bir ülkede yaşayan insanlar içerisinde aynı vatandaşlık numarasına sahip iki kişi olamaz.

Nosu	Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel	Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri	Belge
1	47	Murat	Keskin	İstanbul	235	A01	Türkçe	3	Sosyal	D1	Katılım
2	21	Cem	Serin	Ankara	412	A01	Türkçe	3	Sosyal	D1	
3	47	Murat	Keskin	İstanbul	235	S01	Fen	2	Teknik	D2	EKC
4	54	Mert	Sert	Sakarya	122	A01	Türkçe	3	Sosyal	D1	Katılım
5	21	Cem	Serin	Ankara	412	S01	Fen	2	Teknik	D2	

Örneğimizi incelersek tablomuzda Öğrenci numarasını temsil eden bir alan olmasına rağmen bu alan tekrarlı bir yapıdadır. Bu tabloda yer alan Öğr. No alanı bu tabloda yer alan kayıtları tek başına temsil edemez. Bu tabloya her kaydı tek başına temsil edecek bir alan ekleriz. Böylece tablodaki kayıtları tek başına temsil edecek bir alan eklemiş olduk.

3. Her tabloda tek bir varlığı ait veriler saklanmalıdır.

Varlık veritabanında özellikleri saklanacak nesneleri temsil eder. Her tabloda sadece tek bir varlığın özellikleri yer almmalıdır. Örnek tablomuzda yer alan varlıklarını incelersek ilk dikkatimizi çeken öğrenci ve Ders gibi iki ayrı varlığın özelliklerinin tablo içerisinde yer aldığı olacaktır. Bu durumu ortadan kaldırmak için yeni bir tablo oluşturarak varlıklardan birine ait

tüm özellikleri diğer tabloya aktarmaktır. Ders varlığı için yeni bir tablo oluşturarak normalizasyon kurallarına dikkat ederek tablomuzu oluşturuyoruz.

Öğrenci Tablosu

Nosu	Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel	Ders Kod	Belge
1	47	Murat	Keskin	İstanbul	235	A01	Katılım
2	21	Cem	Serin	Ankara	412	A01	
3	47	Murat	Keskin	İstanbul	235	S01	EKC
4	54	Mert	Sert	Sakarya	122	A01	Katılım
5	21	Cem	Serin	Ankara	412	S01	

Ders Tablosu

Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri
A01	Türkçe	3	Sosyal	D1
A01	Türkçe	3	Sosyal	D1
S01	Fen	2	Teknik	D2
A01	Türkçe	3	Sosyal	D1
S01	Fen	2	Teknik	D2

Ders Tablosu

Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri
A01	Türkçe	3	Sosyal	D1
S01	Fen	2	Teknik	D2

Dikkat edilirse hangi öğrencinin hangi dersi aldığı bilgisini göstermek için öğrenci varlığının özelliklerinin gösterildiği tabloda ders tablosunda yer alan ve dersleri tek başına temsil eden alan bilgileri de saklanmıştır. Ayrıca belge alanında yer alan bilgiler ders tablosunun bir özelliği olmadığı için öğrenci tablosunda bırakılmıştır. Ders tablosunda normalizasyonun 1. ve 2. Kuralına uygun olarak her alan bir veriye ait özelliklerini göstermeyecektir ve tablodaki kayıtları tek başına temsil edecek alan tekrar etmeyecek şekilde gerçeklemiştir.

4. Tabloda tekrar eden kayıtlar veya tekrar eden alanlar bulunmamalıdır.

İlişkisel veri tabanlarında tablolar arasında bağlantılar oluşturarak sorgulamalar gerçekleştirerek farklı tablolardaki veriler tek bir liste halinde gösterilebilir. Eğer tablomuzda aynı kayıt fazla sayıda tekrar ediyorsa veya birden fazla sütunda yer alan aynı veriler birden fazla tekrar ediyorsa bu verileri ayrı birer tablo oluşturarak yeni oluşturduğumuz tabloya aktarabiliriz.

Öğrenci Tablosu

Nosu	Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel	Ders Kod	Belge
1	47	Murat	Keskin	İstanbul	235	A01	Katılım
2	21	Cem	Serin	Ankara	412	A01	
3	47	Murat	Keskin	İstanbul	235	S01	EKC
4	54	Mert	Sert	Sakarya	122	A01	Katılım
5	21	Cem	Serin	Ankara	412	S01	

Örnek tablomuzu incelersek öğrenci bilgilerinin yer aldığı alanların bir bölümünün tekrar ettiği görülmektedir. Bu tekrarı ortadan kaldırmak için sadece öğrenciye ait özellikler ayrı bir tablo oluşturularak oluşturulan tabloya aktarılır.

Liste Tablosu

Nosu	Ogr No	Ders Kod	Belge
1	47	A01	Katılım
2	21	A01	
3	47	S01	EKC
4	54	A01	Katılım
5	21	S01	

Böylece tablomuzda Tekrar eden kayıt kalmamış olacaktır. Öğrenciye ait özellikleri ayrı bir tabloya aktarırken Belge alanı yeni oluşturulan tabloya aktarılmayacaktır. Dikkat edilirse belge alanında yer alan verilen öğrencinin aldığı derse göre değişiklik oluşturmaktadır.

Öğrenci Tablosu

Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel
47	Murat	Keskin	İstanbul	235
21	Cem	Serin	Ankara	412
47	Murat	Keskin	İstanbul	235
54	Mert	Sert	Sakarya	122
21	Cem	Serin	Ankara	412

Başlangıçta birincil anahtar olarak kullanmadığımız Öğrenci No alanını yeni oluşturduğumuz tabloda birincil anahtar yani tabloda yer alan kayıtları tek başına temsil edecek alan olarak belirleyebiliriz.

Öğrenci Tablosu

Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel
47	Murat	Keskin	İstanbul	235
21	Cem	Serin	Ankara	412
54	Mert	Sert	Sakarya	122

Böylece tablomuzda tekrar eden kayıt kalmayacaktır.

Tablolar arasında tutarlılığı sağlamak için tablolar arasında ilişkiler oluşturabiliriz.

5. Tablolarda boş geçilen (NULL) alanların sayısının mümkün olduğunda azaltmalıyız.

Tablolar içerisinde boş olarak geçilen alanlar olabilir. Eğer Alanları boş geçilen kayıtların sayısı fazla ise bunlar bellekte fazladan yer kaplayacaktır. Bunu önlemek için bu alanlar için yeni tablolar tanımlanabilir.

Liste Tablosu

Nosu	Ogr No	Ders Kod	Belge
1	47	A01	Katılım
2	21	A01	
3	47	S01	EKC
4	54	A01	Katılım
5	21	S01	

Örneğimizdeki liste tablomuzu incelersek Belge alanında boş olarak geçilen kayıtları görebiliriz. Boş geçilen alanlardan kaynaklanan yer israfından kurtulmak için belge bilgilerinin yer aldığı yeni bir tablo tasarlayabiliriz.

Belge Tablosu

İdsi	Liste No	Belge
1	1	Katılım
2	3	EKC
3	4	Katılım

Belge isimli tabloda belgelere ait bilgiler yer almaktadır.

Liste Tablosu

Nosu	Ogr No	Ders Kod
1	47	A01
2	21	A01
3	47	S01
4	54	A01
5	21	S01

Belge tablosundaki "Liste No" alanı ile liste tablosundaki "Nosu" alanı ilişkilidir. Belge tablosundaki kayıtları tek başına temsil etmesi için "idsi" isimli alan eklenmiştir.

Böylece başlangıçta var olan örneğimiz şu tablolarından oluşacaktır;

Liste Tablosu

Nosu	Ogr No	Ders Kod
1	47	A01
2	21	A01
3	47	S01
4	54	A01
5	21	S01

Ders Tablosu

Ders Kod	Ders Ad	Ders Kredi	Ders Bolum	Ders Yeri
A01	Türkçe	3	Sosyal	D1
S01	Fen	2	Teknik	D2

Öğrenci Tablosu

Ogr No	Ogr Ad	Ogr Soyad	Ogr Adres	Ogr Tel
47	Murat	Keskin	İstanbul	235
21	Cem	Serin	Ankara	412
54	Mert	Sert	Sakarya	122

Belge Tablosu

İdsi	Liste No	Belge
1	1	Katılım
2	3	EKC
3	4	Katılım

5.6 İşlevsel Bağımlılıklar

Fonksiyonel Bağımlılık (Functional Dependencies) Tanımı

R bir ilişki şeması, X ve Y nitelikleri de R'nin iki alt kümesi olsun. $\text{XR} \text{ Y R}$ Eğer X nitelikler kümesinin değerleri Y nitelikler kümesinin değerlerini belirliyorsa Y X'e fonksiyonel bağımlıdır denir. R ilişkisinin Y niteliği, R'nin X niteliğine ikişisel olarak bağımlı ve X'in her bir değeri Y'nin bir değerine karşılık geliyorsa fonksiyonel bağımlılıktan söz edilir.

İşlevsel bağımlılık kavramı, genel anlamda, ilişkisel tasarımların olması gerektiği gibi (iyi biçimde) yapıldığının formal ölçütlerini belirlemede kullanılır. İşlevsel bağımlılıklar + anahtarlar → ilişkiler için normal biçimleri tanımlamada kullanılır. İşlevsel bağımlılıklar, bir ilişkideki özelliklerin anlamı ve birbirleri arasındaki ilişkilerden türetilen kısıtlamalardır. X özellikler kümesinin aldığı değerler, Y özellikler kümesindeki tek bir değere karşılık geliyorsa, X kümesi Y kümesini fonksiyonel olarak belirliyor demektir.

($X \rightarrow Y$: Y işlevsel olarak X'e bağımlıdır)

Bir R ilişkisinde $X \rightarrow Y$ ise, t_1 ve t_2 satırları için $t_1[X]=t_2[X]$ ise $t_1[Y]=t_2[Y]$ olmalıdır.

R ilişkisinde $X \rightarrow Y$ ise, tüm $r(R)$ ilişki örnekleri üzerinde bu anlamda bir kısıtlama söz konusudur. İşlevsel bağımlılıklar, özellikler üzerinde gerçek-dünyadaki kısıtlamalardan türetilir. Eğer K, R ilişkisinde anahtar özellik ise, R ilişkisindeki tüm özellikleri işlevsel olarak belirler. (Çünkü iki farklı satır için $t_1[K] \neq t_2[K]$)

İşlevsel Bağımlılık Kısıtlamaları (Örnekler)

Öğrenci numarası, öğrencinin adını belirler.

{ Öğrenci No } \rightarrow { Öğrenci Adı }

Öğrenci numarası, öğrencinin adını ve soyadını belirler.

{ Öğrenci No } \rightarrow { Öğrenci Adı, Öğrenci Soyadı }

Açılan dersin numarası, açılan dersin adını, dönemini ve açan bölümü belirler.

{ Açılan Ders No } \rightarrow { Açılan Ders Adı, Açılan Dönem, Açılan Bölüm }

Sipariş No ve Ürün No, üründen kaç adet sipariş verildiğini belirler.

{ Sipariş No, Ürün No } \rightarrow { Adet }

ŞİRKET (Şirket No, Şirket Adı, Şirket Adresi, Şirket Telefonu, Ortak Adı, Hisse)

{ Şirket No } \rightarrow { Şirket Adı, Şirket Adresi, Ortak Adı Soyadı, Hisse}

{ Şirket No, Şirket Adı } \rightarrow { Şirket Adresi }

{ Şirket Adı } → { Şirket Adresi, Şirket Telefonu }

{ Şirket Telefonu } → {Şirket Adı, Şirket Adresi }

{ Şirket Telefonu } → {Ortak Adı, Hisse}

{ Şirket Adı, Ortak Adı } → {Hisse}

{ Ortak Adı } → { Hisse }

{ Ortak Adı } → { Şirket No, Şirket Adı }

5.6.1 Tam İşlevsel Bağımlılık ve Kısmi Bağımlılık (Partial Dependency)

Tam İşlevsel Bağımlılık → A ve B bir R ilişkisinin özellik kümeleri ise, eğer B işlevsel olarak A'ya bağımlı ise fakat A'nın herhangi alt kümesine bağımlı değilse, bu durumda B özellik kümesi A özellik kümesine tam işlevsel bağımlıdır. Kısmi Bağımlılık → A ve B özellik kümeleri işlevsel bağımlı ise ($A \rightarrow B$) ve A özellikler kümesinden herhangi bir özelliğin çıkarılması bu bağımlılığı bozmazsa, $A \rightarrow B$ bağımlılığına kısmi bağımlılık denir.

{ Şirket No, Şirket Adı } → { Şirket Adresi } bağımlılığı tam işlevsel değil (kısmi işlevsel), çünkü “Şirket Adresi” aynı zamanda “Şirket No” alanına da bağımlıdır.

5.6.2 Dolaylı Bağımlılık (Transitive Dependency)

A, B ve C özellik kümelerini içeren bir R ilişkisinde $A \rightarrow B$ ve $B \rightarrow C$ işlevsel bağımlılıkları bulunmakta ise, C, A'ya B aracılığı ile dolaylı bağımlıdır.

<u>Sicil No</u>	Personel Adı	Personel Soyadı	Birim	Yönetici
1	Sevil	KALA	Personel	Ali kaya
2	Ayşe	KAYA	Kalite	Serap er
3	Aslan	SELVİ	Bilgi işlem	Aylin ker
4	Can	CİVELEK	Muhasebe	Ahmet Celp



Sicil No, Birim'i belirlemektedir. Anahtar olmayan Birim özelliği de, Yönetici özelliğini belirlemektedir. Yönetici özelliği, Sicil No'ya dolaylı bağlıdır.

Tam İşlevsel (Fonksiyonel) Bağımlılık

A ve B bir ilişki, eğer B işlevsel olarak A'ya bağlı ise, bu durumda B özellik kümesi A özellik kümesine tam işlevsel bağlıdır.

PERSONEL

ID ADI ŞEHİR

1 Ali Burdur

2 Fatma İstanbul

3 Arda Antalya

Personel tablosu ile ilgili neler söyleyebiliriz?

Eğer ID numarasını biliyorsam, ismini de biliyorum"

ID numarası ismi belirmektedir.

ADI niteliği, ID'ye fonksiyonel bağlıdır.

$A \rightarrow B$ ise A fonksiyonel olarak B 'yi tanımlar.

numara	adsoyad	bolum	sınıf	tck
1	Ali	Bilgisayar	1	11
2	Fatma	Elektronik	2	22
2	Arda	Makine	1	33

Yukarıdaki OGRENCI tablosunu ele aldığımız zaman aşağıdaki bağımlılıklardan söz edebiliriz.

numara -> adsoyad

numara -> adsoyad, bolum, sınıf, tck

tck -> numara, adsoyad, bolum, sınıf

Kısmi Bağımlılık (Partial Dependency)

A ve B özellik kümeleri işlevsel bağımlı ise ($A \rightarrow B$) ve A özellikler kümesinden herhangi bir özelliğin çıkarılması bu bağımlılığı bozmazsa, $A \rightarrow B$ bağımlılığına kısmi bağımlılık denir.

Dolaylı Bağımlılık (Transitive Dependency)

A, B ve C özellik kümelerini içeren bir ilişkide

$A \rightarrow B$ ve $B \rightarrow C$ işlevsel bağımlılıkları bulunmakta ise,

C, A 'ya B aracılığı ile dolaylı bağımlıdır.

Uygulamalar

SORU 1: Aşağıda normal olmayan formda verilen tablonun 1. Normal form sonucunda ulaşılacak biçimini oluşturunuz?

Sicil No	Personel Adı Soyadı	Telefonları
1	Yunus Asil	5322902451, 5322941821, 5322986883
2	Mekki Taş	5322903912, 5322982039
3	Fatma Sarı	5322905793, 5322819283

ÇÖZÜM:

Çok değerli özellik ve Birleşik özellikte alanlar içerdiginden dolayı Normal Olmayan Biçimdedir. Önce birleşik özellikteki alan Personel Adı ve Personel Soyadı şeklinde iki ayrı alana ayrılarak aşağıdaki tablo oluşturulur. Sicil No anahtar olarak belirlenir.

Sicil No	Personel Adı	Personel Soyadı	Telefon
1	Yunus	Asil	5322902451
1	Yunus	Asil	5322941821
1	Yunus	Asil	5322986883
2	Mekki	Taş	5322903912
2	Mekki	Taş	5322982039
3	Fatma	Sarı	5322905793
3	Fatma	Sarı	5322819283

Çok değerli özellik olan için ayrı bir tablo oluşturularak ilişki kurularak aşağıdaki personel ve telefon şeklinde iki tablo oluşturularak başlangıçtaki tablo normalize edilir. Telefon tablosu için satır no olarak anahtar alan oluşturulur. Sicil no telefon tablosuna yabancı anahtar alan olarak eklenir.

Sicil No	Personel Adı	Personel Soyadı
1	Yunus	Asil
2	Mekki	Taş
3	Fatma	Sarı

Satır No	Telefon	Sicil No
1	5322902451	1
2	5322941821	1
3	5322986883	1
4	5322903912	2
5	5322982039	2
6	5322905793	3
7	5322819283	3

Uygulama Soruları

Normal olmayan bir tablo Normalizasyon kurallarına göre bölünerek daha işlevsel hale getirilmelidir.

Ö.NO	Ö.AD	Ö.SOYAD	D_NO	DERS_ADI	VIZE	FINAL	H.NO	H.AD	H.SOYAD
12	ALİ	Ada	22	Matematik	45	95	101	Ece	Ay
12	ALİ	Ada	23	Fizik	58	65	101	Ece	Ay
12	ALİ	Ada	24	Enformatik	65	48	13	Efe	Kel
12	ALİ	Ada	25	Bilişim	59	58	19	Zeki	Tan
12	ALİ	Ada	26	Türk Dili	87	97	102	Nur	Kara
15	CAN	Gür	22	Matematik	45	25	101	Ece	Ay
15	CAN	Gür	23	Fizik	15	57	101	Ece	Ay
15	CAN	Gür	24	Enformatik	69	48	13	Efe	Kel
22	CEM	Ada	72	Veri Tabanı	35	68	16	Nuri	Dağ
13	ALP	Şan	72	Veri Tabanı	68	35	16	Nuri	Dağ
14	OYA	Şen	72	Veri Tabanı	85	100	16	Nuri	Dağ

Tablo 2.1:Birinci Normal Form Biçiminde Tablo

Kısmi Bağımlılıkları ortadan kaldırarak Birinci Normal Formda(1NF) olan tablomuzu İkinci Normal Forma(2NF) göre normalize edersek tablomuz şekil 2.14'teki gibi olacaktır.

NOTLAR

O.NO	D_NO	VIZE	FINAL
12	22	45	95
12	23	58	65
12	24	65	48
12	25	59	58
12	26	87	97
15	22	45	25
15	23	15	57
15	24	69	48
22	72	35	68
13	72	68	35
14	72	85	100

ÖĞRENCİ

Ö.NO	Ö.AD	Ö.SOYAD
12	ALİ	Ada
15	CAN	Gür
22	CEM	Ada
13	ALP	Şan
14	OYA	Şen

DERSLER

D_NO	DERS_ADI	VIZE	FINAL	H.NO	H.AD	H.SOYAD
22	Matematik	45	95	101	Ece	Ay
23	Fizik	58	65	101	Ece	Ay
24	Enformatik	65	48	13	Efe	Kel
25	Bilişim	59	58	19	Zeki	Tan
26	Türk Dili	87	97	102	Nur	Kara
72	Veri Tabanı	85	100	16	Nuri	Dağ

Şekil 2.14: 2NF Biçiminde Tablolar

Şekil 2.14'te dersler tablosunda geçişli bağımlılıkları kaldırarak tablomuzu şekil 2.15'te olduğu gibi Üçüncü Normal Forma(3NF) dönüştürüyoruz.

NOTLAR

O.NO	D_NO	VIZE	FINAL
12	22	45	95
12	23	58	65
12	24	65	48
12	25	59	58
12	26	87	97
15	22	45	25
15	23	15	57
15	24	69	48
22	72	35	68
13	72	68	35
14	72	85	100

ÖĞRENCİ

O.NO	O.AD	O.SOYAD
12	ALİ	Ada
15	CAN	Gür
22	CEM	Ada
13	ALP	Şan
14	OYA	Şen

DERSLER

D_NO	DERS_ADI	H.NO
22	Matematik	101
23	Fizik	101
24	Enformatik	13
25	Bilişim	19
26	Türk Dili	102
72	Veri Tabanı	16

HOCALAR

H.NO	H.AD	H.SOYAD
101	Ece	Ay
13	Efe	Kel
19	Zeki	Tan
102	Nur	Kara
16	Nuri	Dağ

Şekil 2.15: 3NF Biçiminde Tablolar

SORU 2: Aşağıdaki tabloyu normalleştiriniz. Gerekliyse normalleştirirken yeni ID alanları ekleyebilirsiniz.

Film_No	Film_Adı	Yılı	Yönetmeni	Senaristi	Türü	Film Şirketi
84934	Yol	1982	Serif Gören	Yılmaz Güney	Drama	Güney Film
84934	Yol	1982	Yılmaz Güney	Yılmaz Güney	Drama	Güney Film
110912	Pulp Fiction	1994	Quentin Tarantino	Quentin Tarantino	Aksiyon	Miramax Films
110912	Pulp Fiction	1994	Quentin Tarantino	Roger Avary	Aksiyon	Miramax Films
114369	Se7en	1995	David Fincher	A. Kevin Walker	Polisiye	New Line Cinema
1375666	Inception	2010	Christopher Nolan	Christopher Nolan	Aksiyon	Warner Bros Pic.
1375666	Inception	2010	Christopher Nolan	Christopher Nolan	Bilim-Kurgu	Warner Bros Pic.

CEVAP

1) Verilen şema 1. Normal Formda (atomik değerler). Anahtar alanlar Film_No, Yönetmeni, Senaristi ve Türü.

- Filmler (Film_No, Film_Adı, Yılı, Yönetmeni, Senaristi, Türü, Film Şirketi)

1NF → 2NF (Film_Adı, Yılı ve Film Şirketi nitelikleri Film_No niteliğine kısmi olarak bağımlı)

- Filmler (Film_No, Film_Adı, Yılı, Film Şirketi)
- FYST (Film_No, Yönetmeni, Senaristi, Türü)

2NF → 3NF (Geçişli bağımlılık yok, fakat ŞirketID gibi bir alan eklendiye Şirketler tablosu yaratılmalı)

- Filmler (Film_No, Film_Adı, Yılı, Film Şirketi)
- FYST (Film_No, Yönetmeni, Senaristi, Türü)
- Şirketler (ŞirketID, Şirket_Adı)

3NF → 4NF (Yönetmen, Senarist ve Türün aynı tabloda olması gereksiz tekrarlara neden oluyor)

- Filmler (Film_No, Film_Adı, Yılı, Film Şirketi)
- FilmTür (Film_No, Türü)
- FilmYönetmen (Film_No, Yönetmeni)
- FilmSenarist (Film_No, Senaristi)

Bu Bölümde Ne Öğrendik Özeti

Bu derste veri tabanı tasarımda mantıksal tasarım aracı olan normalizasyon konusu açıklandı. Önce normalizasyonun önemi vurgulanmaktadır. Ardından normalizasyon kuralları ayrıntılı olarak açıklandı. Daha sonra bu kuralların uygulaması ile ilgili örnek sunuldu. Normalizasyon yapılırken uyulması gereken kurulların her birine normal form adı verilir. Genel olarak üç normal form uygulanır; Birinci Normal Form (1NF), İkinci Normal Form (2NF) ve Üçüncü Normal Form (3NF). Daha yüksek düzey formlar var ama çok fazla kullanılmıyor. İlk üç düzey ihlal edilirse Kayıt güncelleme, kayıt silme ve Kayıt bulmada zorluk çekilir.

Bölüm Soruları

1) Aşağıdaki tabloya göre şıkların hangisinde bir işlevsel bağımlılık vardır?

Sicil No	Proje No	Proje Adı	Personel Adı	Personel Soyadı	Unvan	Çalışma Saati
1	23	P10	Yunus	Asil	Yönetici	20
2	17	P20	Mekki	Taş	Mühendis	30
3	21	P22	Fatma	Sarı	Mühendis	25

- a) Sicil No → Proje No
- b) Personel Adı → Proje Adı
- c) Sicil No → Proje Adı
- d) Proje Adı → Unvan
- e) Proje No → Proje Adı

2) Normalizasyon ile ilgili olarak aşağıdakilerden hangisi yanlıştır?

- a) Bir veritabanında verilerin saklanmış olduğu nesneler tablo olarak adlandırılır.
- b) İlişkisel veri tabanında tablolarda birbirleri ile tamamen aynı olan iki kayıt kullanılabilir
- c) Tablolarda aynı tipte bilgiyi içerecek alanların bulunması mutlak veri fazlalığı sorunudur.
- d) Birinci normal formda hazırlanmış bir tabloda ekleme, silme ve güncelleme sırasında sorunlar olabilir.
- e) Birinci normal formdan ikinci normal forma geçişte kısmi bağımlılıkların ortadan kaldırılması gerekmektedir.

3) Normalizasyon ile ilgili olarak aşağıdakilerden hangisi doğrudur?

- a) 1NF'den 2NF'e geçebilmesi için tüm geçmişen bağımlılıklar kaldırılmalıdır.
- b) Tablolarda bir bilginin birden fazla tekrarlanması bellekte fazla yer kaplamaz.
- c) Normalleştirme, taslak veri tabanı üzerinde birtakım işlemler yapılarak taslağı son haline yaklaşma yöntemidir.
- d) Veri tabanı tablolarında bir alana birden fazla veri girişi olabilir.
- e) Herhangi bir tablonun tekrarlı bilgiler içerdiği duruma 2NF denir.

4) Bir kuruluşta evrak kayıt defterinin;

- E-R modelini yapınız.
- Tablo yapısını çiziniz.
- Normal formunu oluşturunuz.

CEVAPLAR

1-E, 2-B, 3-C

6. SQL- YAPISAL SORGU DİLİ

Bu Bölümde Neler Öğreneceğiz?

6.1 SQL ile Gerçekleştirilecek İşlemler

 6.1.1 Veri Tabanı Oluşturma

 6.1.2 Tablo Oluşturma

 6.1.3 Tabloya Veri Girişi

6.2 Tabloda Sorgulamalar Yapma

 6.2.1 Temel SQL Sorgulamaları Select Komutu

 6.2.2 Tablo Bilgilerinin Sıralanmış Olarak Listelenmesi

 6.2.3 Birden Çok Alana Göre Sıralama

 6.2.4 Tekrarlı Satırların Ortadan Kaldırılması-SELECT DISTINCT

 6.2.5 Koşula Bağlı Olarak Listeleme

 6.2.6 Çeşitli Veri Tipleri İçin Basit Sorgulamalar

 6.2.6.1 Nümerik Veri Tipi

 6.2.6.2 Tarih Veri Tipi

 6.2.6.3 Mantıksal Veri Tipi

 6.2.7 Birden Çok Koşula Dayalı Sorgulamalar (Not, And, Or)

 6.2.8 Bir Veri Kümesi İçinde Arama (In Operatörü)

 6.2.9 Aralık Sorgulaması (Between Sözcüğü)

 6.2.10 Karakter Türü Bilgi İçinde Arama Yapma (Like Sözcüğü)

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** SQL Neyin kısaltmasıdır?
- 2)** SQL Tüm veritabanı yönetim sistemlerinde ortak bir dilmidir? Farklılıklar vardır?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	SQL komutlarını bilir.	Anlatım, Soru-Cevap, Tartışma
	SQL komutları ile sorgu oluşturabilir.	Alıştırma ve Uygulama, Örnek Olay
		Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

Anahtar Kavramlar

- SQL (Structred Query Language)
- Yapısal Sorgu Dili

Giriş

SQL (Structred Query Language),-Yapısal Sorğu Dili; tüm veritabanları tarafından desteklenen bir dildir. Veritabanları ile iletişim kurmak ve onlar üzerinde işlem yapmak için kullanılır. Program geliştiricileri ve Veritabanı kullanıcıları, bir veritabanına veri eklerken, silerken, güncellerken veya sorgularken bu dili kullanırlar. Bu sorgulama dili ile veritabanından istenilen kriterlere sahip veri seçilebilir, istenirse de çeşitli güncellemeler, silmeler ve veritabanı yapısında değişiklikler yapabilir.

SQL veri tabanında yeni tablo oluşturma, veri ekleme, silme, düzeltme, güncelleme, sorgulama ve koruma ve daha çok sayıda işlemin bir anda yapılmasını sağlar. SQL dili IBM tarafından geliştirilen System R projesinde geliştirilen matematiksel bir söz dizilimine sahip bir ilk örnek Veri Tabanı yönetim sistemi SQUARE kadar uzanır. Daha sonra oluşturulan bu ilk veri tabanı yönetim sistemi daha kullanışlı olması için İngilizce dilinin söz dizilimine sahip bir hale getirilerek IBM şirketi 1979 yılında SEQUEL (Structured English Query Language) olarak adlandırılan yapısal sorgulama dilini oluşturdu. Daha sonra sonra bu dil geliştirilerek SQL adını almıştır.

6. SQL- YAPISAL SORGU DİLİ

SQL sorgulama dilinin kullanıldığı bazı bilindik veritabanı işletim sistemleri: Oracle, Sybase, Microsoft SQL Server, Access vb. dir. SQL bir programlama dili değildir. Bir kullanıcı arayüzü tanımlayamaz yada bir dosya yönetimi yapamaz. SQL deyimleri veritabanları üzerinde çeşitli işlemleri yerine getirirler. Bu deyimler işlevlerine göre şu şekilde sınıflandırılır: 1) Veri tanımlama dili, 2) Veri düzenleme dili, ve 3) Veri kontrol dili.

Veri Tanımlama Dili: Bu gruptaki komutlar kullanılarak, tablo, trigger, view gibi veri tabanı nesneleri tanımlanır. Üç temel ifadesi, CREATE ile bir nesne tanımlanır, ALTER ile nesne üzerinde değişiklik yapılır ve DROP ifadesi ile bir nesne silinebilir.

Veri Düzenleme Dili: Bu alt dil bir tabloya veri ekleme (INSERT), silme (DELETE) ve güncelleme(UPDATE) yapmanın yanı sıra verileri seçmek ve raporlamak için SELECT ifadesi ve SELECT ifadesi ile birlikte kullanılan, INTO, FROM, WHERE, LIKE, GROUP BY, ORDER BY, HAVING... gibi çok yan ifade vardır. Bu ifadelere ait örnekleri bir çok yerde bulmak mümkündür.

Veri Kontrol Dili: Temel 2 ifadeden oluşur. VTYS'de tanımlı Roller ve kullanıcılar için ifade ve nesne kullanma izni tanımlar. Erişim (GRANT) ve erişim kaldırma (REVOKE) ifadeleri ile bu yetkiler ayarlanır. SQL Server gibi VTYS'lerinde bunlara ek olarak erişim engelleme(DENY) ifadesi de yer almaktadır.

Yeni gelişmeler karşısında SQL diline bir çok yeni özellikler (özellikle nesneye yönelik olmak üzere) eklenmiş ve yeni uygulamaların ihtiyaçlarını karşılamak için yeni eklentiler yapılmıştır. SQL, isminin belirttiği gibi sadece bir veri tabanı sorgulamak ve onun verisini idare etmek için değil onu tanımlamak için de kullanılır. SQL aslında iki alandan meydana gelmiştir.

- Veri tabanı ve tabloların oluşturulması için komutlar içeren kısmı
- Sorgu komutları içeren kısmı

SQL'in kullandığımız bir kaç sorgulama temel komutu vardır. Bunlarla ilgili bazı örnekler aşağıda verilmektedir.

SQL komutları kullanılarak aşağıdaki işlemler yapılabilir:

- Veritabanı nesnelerinin oluşturulması ve bu nesnelerle ilgili işlemlerin yapılması
- Bilgilerin istenilen koşullara göre görüntülenmesi ve sorgulama işlemleri
- Tablolara veri girişi yapılması
- Bilgilerin güncelleştirilmesi
- Tabloların veya tablolardaki verilerin silinmesi

6.1 SQL ile Gerçekleştirilecek İşlemler

6.1.1 Veri Tabanı Oluşturma

SQL ile gerçekleştirilecek işlemlerde, üzerinde işlem yapılacak tablolar, bir veri tabanı içinde oluşturulur. Bu veri tabanını oluşturmak için,

CREATE DATABASE isim; şeklindeki SQL komutunu kullanmak gereklidir. Bu komut belirtilen isim'deki veri tabanını oluşturur.

Veri tabanında, bir işletmenin çalışanlarının bilgilerini yüklemek istediğimizi varsayıyalım: Bunun için veri tabanı ve gerekli tablonun nasıl oluşturulacağını görelim. Tablonun adı personel ve alanlarıda aşağıdaki gibi belirlenmiş olsun.

Personel Tablosu:

Sicil No	Ad	Soyad	Doğum Tarihi	Adres	Cinsiyet	Gelir	Böl-No
----------	----	-------	--------------	-------	----------	-------	--------

6.1.2 Tablo Oluşturma

SQL ile giriş bölümünde verilen tabloların oluşturulması için, CREATE TABLE komutunu kullanmak gereklidir. Aşağıda bu tabloyu oluşturacak SQL komutları verilmiştir.

CREATE TABLE tabloadı; komutu ile tablolar oluşturulur. Bu komutu kullanırken SQL'in uygun veri tipleri kullanılır.

```
CREATE TABLE Personel  
    (sicil INTEGER NOT NULL,  
     ad CHAR(10) NOT NULL,  
     soyad CHAR(10) NOT NULL,  
     dog_tar DATE,  
     adres CHAR(50),  
     cinsiyet LOGICAL,  
     gelir NUMERIC(13,2),  
     bol_no SMALLINT);
```

Yukarıdaki örnekte Giriş kısmında verilen Personel Tablosunun nasıl oluşturulduğu verilmiştir. NOT NULL ifadesi söz konusu alan ile ilişkili olarak mutlaka veri yüklenmesi

gerektigini, ilgili alanin bos bırakılmayacağını anlatmaktadır. NOT NULL ifadesi yoksa, o alan NULL anlamındadir. Yani o alan ilişkili olarak veri yüklenmemesi durumuna da müsaade edilmektedir.

6.1.3 Tabloya Veri Girişi

Bir tabloya veri girişi işlemi için;

INSERT INTO Tabloadı VALUES

komutu kullanılır.

Örnek: INSERT INTO Kimlik

VALUES ('Erkam','Sert','20.12.1985','Adana');

Tablodaki Sütun İsimleri ve Tablo İsimleri İle İlişkili Kurallar şu şekildedir; İsim uzunlukları 18 karaktere kadar olabilir. (Bazı SQL uygulamalarında 8 karaktere kadar). İlk karakter bir harf olmalıdır. Onu izleyen karakterler, harf, rakam ya da alt çizgi simbolü (_) olabilir.

6.2 Tabloda Sorgulamalar Yapma

SQL içinde, tek bir tablo içinde çeşitli kriterlere göre bilgi sorgulama, bilgiyi sıralı olarak elde etme, bilgi özetleme, ortalama vb. gibi matematiksel işlemleri gerçekleştirmeyi sağlayan komut ve fonksiyonlar vardır. Ayrıca, doğal olarak, aynı tipte işlemleri birden çok tabloyu birlikte ele alarak gerçekleştirmeye mümkün. Bu bölümde öncelikle tek tablo ile ilişkili sorgulamalar ve gerekli SQL komutları inceleneciktir.

6.2.1 Temel SQL Sorgulamaları Select Komutu

SELECT YAPISI

SELECT < * | [kolonadi [AS takmakolonadi] [, ...]] >

FROM tablo [takmatabloadı] [, ...]

[WHERE <koşul>]

[GROUP BY <kolon_listesi>]

[HAVING <koşul>]

[ORDER BY <kolon_listesi> [ASC | DESC]]

GROUP BY <kolon_listesi> içindeki kolonların değerleri aynı olan satırlar grupperler.

HAVING GROUP BY ile oluşan gruppardan <koşul> ile uygun olanlar seçilir.

ORDER BY <kolon_listesi> içindeki kolonlara göre satırlar artan veya azalan biçimde sıralanır.

```
create table stok
(
    sno int primary key,
    stokadi varchar(20),
    adet int,
    fiyat real,
    sonkultar DateTime
```

```
SELECT sno, stokadi, adet, fiyat, (adet*fiyat) AS [tutar] FROM stok
SELECT sno, stokadi, adet, fiyat, tutar= (adet*fiyat) FROM stok
```

SNO	STOKADI	ADET	FİYAT	TUTAR
101	Kalem	100	2	200
102	Defter	120	5	600
103	Silgi	200	1.5	300

SELECT TOP N ve SELECT TOP N PERCENT

SELECT TOP N <sütun listesi> FROM <tablolar>

TOP N → N adet kaydı listele

SELECT TOP 2 * FROM stok

stok tablosundan 2 kaydı listele

SELECT TOP N PERCENT <sütun listesi> FROM <tablolar>

TOP N PERCENT → % N adet kaydı listele

SELECT TOP 40 PERCENT * FROM stok

stok tablosundaki kayıtların %40'ını listele

Tek tablodan gerekli bilgileri elde etmek için sorgulama yapabilecek SQL komutu olan SELECT'in en basit şekli aşağıdaki gibidir.

```
SELECT *  
FROM Tabloadı;
```

Bu komut Tabloadı kısmında adı yazılı tablo içindeki bütün bilgileri koşulsuz olarak listeleyecektir. SELECT sözcüğünü izleyen kısımda * simbolünün bulunması, ilgili tablodaki bütün sütun (kolon) isimlerinin ve ilgili bilgilerin listelenmesini sağlayacaktır. Burda istersek * simbolü yerine bütün alanların adını veya işlem yapacağımız alan ya da alanların adını yazarız.

Örnek: SELECT *

```
FROM Kimlik;
```

Örnek: SELECT ad,soyad,dogumtar,dogumyer

```
FROM Kimlik;
```

Yukarıdaki iki örnekte Kimlik isimli tablodaki bütün alanları listeler. Eğer sadece ad alanının listelenmesini istersek;

```
SELECT ad
```

```
FROM Kimlik;
```

yazılır.

6.2.2 Tablo Bilgilerinin Sıralanmış Olarak Listelenmesi

Tablodan listelenecek bilgilerin, belirli bir sütun adına göre (ad'a göre v.b.) sıralanmış olarak görüntülenmesi için, SELECT komutuna ORDER BY sözcüğü ilave edilir.

Örnek: Personel isimli bir tabloda sicil, ad, soyad, gelir sütunları olsun. gelire göre artan sırada (küçükten büyüğe doğru) sıralı olarak listeleyiniz.

SİCİL	AD	SOYAD	GELİR
2746	Benan	Kar	7800
1728	Neşe	Şener	5600
1116	Murat	Pek	8950
1022	Salih	Berkan	7500

```

SELECT sicil,ad,soyad,gelir
FROM Personel

ORDER BY gelir ASC;

```

SİCİL	AD	SOYAD	GELİR
1728	Neşe	Şener	5600
1022	Salih	Berkan	7500
2746	Benan	Kar	7800
1116	Murat	Pek	8950

SONUÇ:

ASC sözcüğü ascending (artan) anlamındadır. Veriler azalan sırada (büyükten küçüğe ya da alfabetik olarak Z'den A'ya doğru) sıralamak için ASC yerine DESC (descending) sözcüğü kullanılmalıdır.

6.2.3 Birden Çok Alana Göre Sıralama

Bir tablo içinde verilerin aynı anda birden çok sütun (alana) göre sıralamakta mümkündür. Örneğin Personel tablosunu ad ve gelir alanlarına göre sıralamak isteyelim.

```

SELECT sicil,ad,soyad,gelir
FROM Personel

ORDER BY ad,brüt;

```

Burada tablo öncelikle ad'a göre artan sırada (A'dan Z'ye doğru) sıralanacak, sadece aynı ad'a sahip olanlar kendi aralarında gelir'e göre küçükten büyüğe (artan) sıralanacaktır.

Ad ve Gelir'e Göre Sıralama.

SİCİL	AD	SOYAD	GELİR
1215	Benan	Kar	2000
3712	Benan	Pek	6000
1152	Benan	Murat	8000
3712	Erol	Akın	4000
8145	Erol	Çelen	8500
1248	Erol	Okur	11000

Burada, çok sayıda alana göre sıralama, farklı sıralama kriterlerine göre gerçekleştirilebilir. Örneğin aşağıdaki SELECT komutu ile ad alanına göre artan, soyad alanına göre azalan, gelir alanına göre artan sıralanmış tablo elde edilmektedir.

```

SELECT sicil,ad,soyad,gelir
FROM Personel
ORDER BY ad ASC,soyad DESC,gelir ASC;

```

veya aynı komut için alternatif yazılış:

```

SELECT sicil,ad,soyad,gelir
FROM Personel
ORDER BY ad,soyad DESC,gelir;

```

şeklinde olacaktır. Örnek çıktı aşağıdaki gibidir.

Ad'a Göre Artan, Soyad'a Göre Azalan, Gelire Göre Artan Sıralama.

SİCİL	AD	SOYAD	GELİR
2742	Erkam	Kaner	8000
1712	Erkam	Kaner	16000
3112	Erkam	Berk	17000
2712	Salih	Caner	12000
1317	Salih	Berat	18000
2718	Zerhan	Şen	7000

6.2.4 Tekrarlı Satırların Ortadan Kaldırılması-SELECT DISTINCT

SQL dilindeki tablo yapısı, formal olarak tanımlanan ilişkisel veri tabanı tablo yapısından farklıdır. SQL de tablo içinde, birbirinin aynı data içeren satırlara müsaade edilir. İlişkisel veri modelinde ise müsaade edilmez. Birbirinin aynı olan satırların, listeleme esnasında, bir kez yazılması için, SELECT komutuna DISTINCT sözcüğü eklenir.

SELECT DISTINCT <sütun listesi> FROM <tablolar>

DISTINCT → listelenen kayıtlardan tekrarlayan kayıtlardan birini göster

Örnek

SELECT ad FROM müsteri

Ad
Ali
Cemal
Aynur
Ali
Arda
Ali
Aynur

Örnek

SELECT DISTINCT ad FROM musteri

Ad
Ali
Cemal
Aynur
Arda

SELECT DISTINCT Sat_no

FROM Par_sat;

Bu komut ile Par_sat adlı tablodan Sat_no'lar tekrarsız olarak listelenecektir. Örneğin;

Par_sat

Sat_no	Par_no	Miktar
S1	P1	200
S1	P3	300
S2	P1	50
S2	P4	150

Sat_no

S1
S2

şeklinde bilgi içeriyorsa, komutun icrası sonucu listesi elde edilecektir

6.2.5 Koşula Bağlı Olarak Listeleme

SELECT komutu ile bir tablonun satırları içinde sadece verilen bir koşulu sağlayanlar da listelenebilir. Örneğin, geliri 5000 'dan fazla olan personel listelenmek istenirse, SELECT komutu aşağıdaki gibi yazılmalıdır:

SELECT *

FROM Personel

WHERE gelir>5000;

Burada WHERE sözcüğünü izleyen kısımda koşul belirtilmektedir. Koşul belirtilirken iki veri birbiri ile karşılaştırılmaktadır. SQL içinde, verileri çeşitli açılardan karşılaştırılmak için kullanılabilen, karşılaştırma operatörleri Tablo 2.4'te verilmiştir. Karşılaştırma ifadesinde karşılaştırılan verilerin türü aynı olmalıdır. Yani, bir karakter türü veri ile ancak karakter türünde başka bir veri, bir nümerik veri ile ancak nümerik olan başka bir veri karşılaştırılabilir.

Tablo 2.4 SQL'de Karşılaştırma Operatörleri.

OPERATÖR	ANLAMI
<	Küçük
>	Büyük
=	Eşit
<=	Küçük veya eşit
>=	Büyük veya eşit
<>	Eşit değil

6.2.6 Çeşitli Veri Tipleri İçin Basit Sorulamalar

6.2.6.1 Nümerik Veri Tipi

Nümerik (sayıda) veri tipi, SMALLINT, INTEGER, DECIMAL, NUMERIC ya da FLOAT tipi bildiri sözcüklerinden biri ile tanımlanan, matematiksel işlemlere sokulabilen, özel semboller içine alınmayan (" " semboller gibi) verileri kapsar. gelir, ürün miktarı vb. tipteki bilgiler bu türde olmalıdır.

Örnek: geliri 8000'dan fazla olmayan personeli listelemek.

```
SELECT *  
FROM Personel  
WHERE gelir<=8000;
```

6.2.6.2 Tarih Veri Tipi

Tarih tipi veriler, {} simgeleri içinde yazılmalıdır.

Örnek: Hangi personelin, doğum tarihi 1960 yılından daha öncedir?

```
SELECT *  
FROM Personel  
WHERE dog_tar <={12/31/59};
```

Burada kullanılan SQL versiyonunda, tarih tipi verinin aa/gg/yy (ay/gün/yıl) formatında temsil edildiği varsayılmıştır.

SONUÇ

SİCİL	AD	SOYAD	DOG_TAR
712	Benan	Pek	05/02/58
718	Hasan	Akın	04/03/59

6.2.6.3 Mantıksal Veri Tipi

Mantıksal veriler için mümkün olabilen sadece iki değer söz konusudur. Doğru (true, T) ile simgelenir, yanlış (false, F) ile simgelenir. Personel tablosunda, personelin cinsiyetini belirleyen bir alanımız olsun. Cinsiyet adlı bu alanda cinsiyeti erkek olanlar true (T), kadın olanlar false(F) ile kodladığımızı kabul edersek, işletmede çalışan personel içinden erkek olanları listelemek için aşağıdaki gibi bir SQL kodu yazmak gerekecektir.

```
SELECT *  
FROM Personel  
WHERE cinsiyet=T;
```

Bu komut aşağıdaki şekilde de kullanılabilir;

```
SELECT *  
FROM Personel  
WHERE cinsiyet;
```

Bu durumda cinsiyet alanı T olanlar (erkek olanlar) listelenir.

6.2.7 Birden Çok Koşula Dayalı Sorulamalar (Not, And, Or)

NOT, OR ve AND mantıksal operatörleri yardımcı ile birden çok koşulun gerçekleşmesine bağlı olarak ifade edilebilecek karmaşık ya da birleşik koşullu listelemeleri gerçekleştirmek mümkün olmaktadır.

Örnek: geliri 5000 TL'den fazla olan ve cinsiyeti erkek olan personelin listelenmesi gibi bir işlemde, söz konusu personel için iki koşul verilmekte ve ikisinin de gerçekleşmesi istenmektedir:

- 1.Koşul → gelirin 5000'dan fazla oluşu
- 2.Koşul → Cinsiyetin Erkek olması

Her iki koşulunda aynı anda gerçekleşmesi istediği için, VE (AND) sözcüğü ile birbirlerine bağlanmıştır. Bu işlemi gerçekleştiren SQL komutu aşağıdaki gibidir.

```
SELECT *  
FROM Personel  
WHERE gelir>5000 AND cinsiyet=T;
```

6.2.8 Bir Veri Kümesi İçinde Arama (In Operatörü)

Aşağıdaki örnek sorunun cevabını, şu ana kadar öğrendiğimiz SQL komutları ile gerçekleştirebiliriz: bol_no'su 1,2 ya da 3 olan personeli listeleyelim.

```
SELECT *  
FROM Personel  
WHERE bol_no=1 OR bol_no=2 OR bol_no=3;
```

Fakat SQL'de bu işlemi gerçekleştirmenin daha kısa ve daha sık bir yolu vardır; IN sözcüğünü kullanarak bu işlemi yaparız.

```
SELECT *  
FROM Personel  
WHERE bol_no IN(1,2,3);
```

Bu komut, OR ile düzenlenen 1. SELECT komut grubuna denktir. Fakat belirtildiği gibi daha kısa ve anlaşılır bir ifade oluşturmaktadır.

IN operatörü NOT ile birlikte de kullanılabilir. Örneğin aşağıdaki soru ile ilişkili üç ayrı eşdeğer SELECT komutu verilmiştir.

6.2.9 Aralık Sorgulaması (Between Sözcüğü)

Örnek: geliri 5000 – 10000 arasında olan personel kimlerdir.

```
SELECT *  
FROM Personel  
WHERE gelir>=5000 AND  
gelir<=10000;
```

şeklindeki bir SELECT komutu ile bu işlem gerçekleştirilebilir. Aynı soruya daha kısa ve daha etkin cevap verebilecek bir SQL komutu ise BETWEEN sözcüğü ile aşağıdaki gibi düzenlenebilir.

```
SELECT *
  FORM Personel
 WHERE gelir BETWEEN 5000
   AND 10000;
```

6.2.10 Karakter Türü Bilgi İçinde Arama Yapma (Like Sözcüğü)

Satırlarımızdaki verileri LIKE ifadesi ile bazı Joker işaretleri-wildcard'ları kullanarak filtreleyebiliriz. LIKE ifadesini sadece char, nchar, varchar, nvarchar, binary, varbinary, smalldatetime ve datetime veri tiplerinde kullanabiliriz. Bu özel simbol yada Joker işaretleri (wildcardlar) aşağıdaki gibidir:

% 0 veya daha fazla karakterli string tek bir karakter

[] belirtilenler arasından tek bir karakter

[^] belirtilenler arasında olmayan tek bir karakter

Bu wildcarların LIKE ile kullanım örnekleri aşağıdaki gibidir;

LIKE 'BR%' BR ile başlayan tüm isimler

LIKE 'Br%' Br ile başlayan tüm isimler. Büyük küçük harf farkı

LIKE '%een' een ile biten tüm isimler

LIKE '%en%' en karakterlerini içeren tüm isimler

LIKE '_en' en ile biten 3 karakterli isimler

LIKE '[CK]%' C veya K ile başlayan tüm isimler

LIKE '[S-V]ing' Tüm 4 karakterli ve ilk karakteri S ile V arasında olan ve ing ile biten tüm isimler

LIKE 'M[^c]%' M ile başlayan ve ikinci karakteri c olmayan tüm isimler

ÖRN: *SELECT companynname*

FROM customers

WHERE companynname LIKE '%Restaurant%'

Uygulamalar

Tablo oluşturma komut dizilimi.

create table stok

```
(  
sno int primary key,  
stokadi varchar(20),  
adet int,  
fiyat real,  
sonkultar DateTime  
)
```

SELECT sno, stokadi, adet, fiyat, (adet*fiyat) AS [tutar] FROM stok

SELECT sno, stokadi, adet, fiyat, **tutar=** (adet*fiyat) FROM stok

Sno	Stokadi	Adet	Fiyat	Tutar
101	Kalem	100	2	200
102	Defter	120	5	600
103	Silgi	200	1.5	300

SELECT TOP 2 * FROM stok -- stok tablosundan 2 kaydı liste

SELECT TOP 40 PERCENT * FROM stok -- stok tablosundaki kayıtların %40'ını liste

SELECT sno, stokadi, adet, fiyat FROM stok

SELECT * FROM stok -- stok tablosundaki bütün kayıtların bütün alanlarını (*) liste. Yukardaki iki komut aynı sonucu verir.

SELECT * FROM musteri WHERE adres LIKE '%bursa%'

--Adres alanının herhangi bir yerinde 'bursa' kelimesi geçen kayıtları listeler

SELECT * FROM musteri WHERE ad LIKE '*AKR+__'

--ad alanının ilk karakteri A veya K olan ve toplam 5 karakter olan kayıtları listeler

SELECT * FROM musteri WHERE ad LIKE '*^MCS+ %'

--ad alanının ilk karakteri M veya C veya S ile başlamayan kayıtları listeler

SELECT * FROM stok WHERE fiyat >=10 AND fiyat <=50

SELECT * FROM stok WHERE fiyat BETWEEN 10 AND 50

--fiyatı 10 ile 50 arasında olanları listeler

SELECT * FROM stok WHERE sno=3 OR sno=7 OR sno= 9

SELECT * FROM stok WHERE sno IN (3,7,9)

--sno 3 veya 7 veya 9 olanları listeler

SELECT * FROM stok WHERE stokadi IS NULL

--stok adı girilmeyenleri listeler

SELECT * FROM stok WHERE stokadi IS NOT NULL

--stok adı girilenleri listeler

Uygulama Soruları

Oluşturduğunuz veri tabanında sizde yukarıdakilere benzer sorular hazırlayınız.

Bu Bölümde Ne Öğrendik Özeti

Bu derste SQL- yapısal sorgu dili konusu anlatıldı. SQL komutları kullanılarak veri tabanları işlemlerinin nasıl yapıldığı örneklerle açıklandı. Özellikle Veritabanı nesnelerinin oluşturulması ve bu nesnelerle ilgili işlemlerin yapılması, bilgilerin istenilen koşullara göre görüntülenmesi ve sorgulama işlemleri konuları üzerinde duruldu.

Bölüm Soruları

1.Aşağıdakilerden hangisi veri tanımlama dili komutlarından değildir?

- A) CREATE
- B) DELETE
- C) DROP
- D) ALTER

2.Aşağıdaki komut dizilişinde eksik olan nedir?

create table

```
(  
sno int primary key,  
stokadi varchar(20),  
adet int,  
fiyat real,  
sonkultar DateTime
```

-)
- a) Oluşturulacak veri tabanı adı yok.
 - b) Oluşturulacak tablo adı yok.
 - c) Girilecek değerler yok.
 - d) Ne oluşturulacağı açıklanmamış.
 - e) Veri türleri eksik.

3.Karakter türü bilgi içinde arama yapmak için yazılmış olan LIKE 'AH%' ifadesinin anlamı aşağıdakilerden hangisidir?

- a) Ah nin yüzde kaç olduğunu hesaplar.
- b) AH ile başlayan tüm isimler.
- c) AH ile biten tüm isimler.
- d) Ah ile başlayan tüm isimler.
- e) Ah ile biten tüm isimler.

CEVAPLAR

1-B, 2-A, 3-B

7. SQL'DE ARİTMETİKSEL İFADELER ve FONKSİYONLAR

Bu Bölümde Neler Öğreneceğiz?

- 7.1 Aritmetiksel İfadeler
- 7.2 Gruplama Fonksiyonları
 - 7.2.1 SUM Fonksiyonu
 - 7.2.2 AVG Fonksiyonu
 - 7.2.3 MAX Fonksiyonu
 - 7.2.4 MIN Fonksiyonu
 - 7.2.5 COUNT Fonksiyonu
- 7.3 Gruplandırarak İşlem Yapma
- 7.4 Tablolarda Değişiklik Yapmak
 - 7.4.1 Tabloya Veri Ekleme
 - 7.4.2 Tablo Satırlarını Silme
 - 7.4.3 Tablo Satırlarındaki Verilerde Değişiklik Yapma-Güncelleme İşlemi
 - 7.4.4 Tablonun Yapısında Değişiklik Yapma
 - 7.4.4.1 Mevcut Bir Tabloya Bir Kolon Ekleme
 - 7.4.4.2 Mevcut Bir Tablonun Kolonlarında Değişiklik Yapma (Modify Komutu)
 - 7.4.4.3 Mevcut Bir Tablodan Bir Kolon Silme (Drop Komutu)
 - 7.4.5 Bir Tablonun Adını Değiştirme – Rename Table Komutu
 - 7.4.6 Mevcut Bir Tablonun Bir Kolonunun Adını Değiştirme – Rename Komutu
 - 7.4.7 Mevcut Bir Tablonun Tümüyle Silinmesi – Drop Table Komutu
- 7.5 QUERY Penceresinde SQL Komutları İle Veri Tabanı Oluşturma ve Sorgular Hazırlama
 - 7.5.1 QUERY Penceresinde Yazılan SQL Komutlarının Listesi
 - 7.5.2 Çalıştırılan Sorguların Sonuçları

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Query penceresinde sql komutları ile veri tabanı oluşturma ve sorguların hazırlanması nasıl yapılır?
- 2)** Başlıca Gruplama Fonksiyonları nelerdir
- 3)** AVG fonksiyonu nasıl uygulanır?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Query Penceresinde SQL Komutlarını kullanabilir.	Anlatım, Soru-Cevap, Tartışma
	Sorguların Sonuçlarını değerlendirebilir. Gruplandırarak İşlem Yapabilir	Alıştırma ve Uygulama, Örnek Olay
	Query penceresinde sql komutları ile veri tabanı oluşturup, sorgular hazırlayabilir.	Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

Anahtar Kavramlar

- Query Penceresi
- Aritmetiksel İfadeler
- Gruplama Fonksiyonları
- SUM fonksiyonu
- AVG fonksiyonu
- MAX fonksiyonu
- MIN fonksiyonu

Giriş

Programlama dillerinde olduğu gibi, SQL'de de bazı aritmetik işlemler için yada tip dönüşümü yapmak için hazır olarak sunulan fonksiyonlar mevcuttur. Bu fonksiyonların bazıları (SUM, AVG, MIN, MAX, ...) birçok kayıt üzerinde işlem yapıp tek bir sonuç üretirken, bazıları ise (Örneğin; tip dönüşümü ile ilgili olanlar) üzerinde işlem yaptığı her kayıt için ayrı sonuç üretir.

7. SQL'DE ARİTMETİKSEL İFADELER ve FONKSİYONLAR

7.1 Aritmetiksel İfadeler

SELECT komutu ile, veri tabanında mevcut tablolardan listeleme yaparken, tabloda ayrı bir sütun (alan) olarak yer almamış ve ancak bir hesaplama sonucunda üretilebilecek bilgileri de liste içine katmak mümkündür. Aşağıdaki SELECT komutu ile, personelin şu anda geçerli olan Geliri ile, bu Gelirin %32 zamlı şekli listelenmektedir.

```
SELECT ad, soyad, Gelir, Gelir*1.32
```

```
FORM Personel;
```

Hesaplanmış alanları elde etmek için oluşturulacak aritmetiksel ifadelerde, Tablo 3.1'de belirtilen semboller kullanılabilir.

Tablo 3.1 SQL'de Aritmetiksel Semboller.

OPERATÖR	İŞLEVİ
** veya ^	Üs alma
*	Çarpma
/	Bölme
+	Toplama
-	Çıkarma

Öncelikli sırası, matematikte ve diğer bilgisayar dillerinde olduğu gibidir. Üs alma, hepsinden öncedir. Sonra çarpma (*) ve bölme (/) gelir. Toplama (+) ve çıkarma (-) en son önceliklidir.

Parantez kullanılarak öncelikler değiştirilebilir.

7.2 Gruplama Fonksiyonları

SQL tablo içinden, çeşitli matematiksel işlemlerin sonucunu otomatik olarak üretmeyi sağlayan, fonksiyonlara sahiptir. Bu fonksiyonlar, örneklerle birlikte aşağıda verilmiştir:

7.2.1 SUM Fonksiyonu

Fonksiyonun argümanı olarak belirtilen sütun ile ilişkili olarak toplama işlemini gerçekleştirir.

Örnek 1 / İşletmedeki personelin Gelirlar toplamı ne kadardır?

```
SELECT SUM(Gelir)
```

```
FROM Personel;
```

Örnek: Bilgi işlem bölümündekilerin Gelirleri toplamı ne kadardır?

```
SELECT SUM(Gelir)  
      FROM Personel  
     WHERE bol_no=5;
```

İfadesi ile sonuç elde edilebilir. Sonuç, sadece bilgi işlem bölümündekilerin Gelirleri toplamı şeklinde olacaktır.

Örnek 2 / Satış, muhasebe ve bilgi işlem bölümündeki personelin Gelirleri toplamı nedir?

Satış bölümü için bol_no 1, muhasebe için 2 ve bilgi işlem için bol_no 5 olarak alınırsa

```
SELECT SUM(Gelir)  
      FROM Personel  
     WHERE bol_no IN(1,2,5);
```

Örnek 3 / Gelirleri 5000 TL'nin altında olan bayan personelin Gelirleri toplamı nedir?

Bayan personeli, daha önceden cinsiyet alanına F. yerleştirerek kodlamış ise

```
SELECT SUM(Gelir)  
      FROM Personel  
     WHERE cinsiyet=.F. AND  
           Gelir<5000;
```

İfadesi istenilen çözümü verecektir.

7.2.2 AVG Fonksiyonu

Aritmetiksel ortalama (avarage) hesaplamak için kullanılır.

```
SELECT AVG(Gelir)  
      FROM Personel;
```

Komutu, işletmedeki ortalama Geliri hesaplayarak görüntüleyecektir. Bu fonksiyon ile de, koşula bağlı olarak hesaplatma yaptırılabilir.

Örnek 1 / Bilgi işlem bölümündekilerin Gelir ortalaması ne kadardır?

Bilgi işlem bölümünün bol_no'su 5 ise

```
SELECT AVG(Gelir)
```

```
FROM Personel
```

```
WHERE bol_no=5;
```

ifadesi istenilen çözümü verecektir.

7.2.3 MAX Fonksiyonu

Tablo içinde, belirlenen sütun (alan) içindeki en büyük değeri bulur.

Örnek 1 / İşletme içindeki en yüksek Gelir ne kadardır?

```
SELECT MAX(Gelir)
```

```
FROM Personel;
```

Örnek 2 / Bilgi işlem bölümündeki en yüksek Gelir ne kadardır?

Bilgi işlem bölümünün 5 ile kodlandığı varsayımlı ile

```
SELECT MAX(Gelir)
```

```
FROM Personel
```

```
WHERE bol_no=5;
```

Örnek 3 / Bayan personel içinde en yüksek Gelir ne kadardır?

```
SELECT MAX(Gelir)
```

```
FROM Personel
```

```
WHERE cinsiyet=.F.;
```

7.2.4 MIN Fonksiyonu

Tablo içinde, belirlenen sütun (alan) içindeki en küçük değeri bulur.

Örnek 1 / İşletme içinde 4 Mayıs 1970'den önce doğanlar için, asgari ücret nedir?

```
SELECT MIN(Gelir)
```

```
FROM Personel
```

```
WHERE dog_tar>{05/04/70};
```

7.2.5 COUNT Fonksiyonu

Tablo içerisinde herhangi bir sayma işlemi gerçekleştirmek için kullanılır.

Örnek 1 / Personel tablosunda kaç satır vardır? (Bu, her satırda farklı bir personel bulunduğu düşünülürse, personel sayısı anlamına gelmektedir.)

```
SELECT COUNT(*)
```

```
FROM Personel;
```

Örnek 2 / Geliri 6000'dan fazla olan personel sayısı nedir?

```
SELECT COUNT(*)
```

```
FROM Personel
```

```
WHERE Gelir>6000;
```

COUNT fonksiyonu, DISTINCT sözcüğü ile de kullanılabilir. Örneğin, personel tablosunda mevcut personelin, işletme içinde kaç tane farklı bölümde çalıştığı bulunmak istenirse aşağıdaki SELECT komutu kullanılabilir.

```
SELECT COUNT(DISTINCT bol_no)
```

```
FROM Personel;
```

COUNT komutunda, * argümanının kullanılması, bütün sütunların (alanların) işleme sokulması, alan adının belirtilmesi ise (COUNT(bol_no) gibi), sadece, belirtilen sütunun işleme sokulmasını sağlar.

round: Sayısal ifadeyi yuvarlar.

round(9,8) => 10, round(9,3) => 9, round(9,5) => 9 yada 10

Örnek 3 / Her personelin adı ve Günlük net ücretini yuvarlayarak gösteriniz.

```
Select ad, round(maas/30) from personel;
```

int: Sayısal ifadenin tam kısmını verir.

Örnek Her personelin adı ve Günlük net ücretini küsüratını atarak gösteniz.

```
Select ad, int(maas/30) from personel;
```

7.3 Gruplandırarak İşlem Yapma

SUM, AVG, MAX, MIN, COUNT fonksiyonları, tablodaki bilgileri, bazı özelliklere göre gruplandırarak bu gruplandırılmış veri üzerine de uygulamak mümkündür. Bu işlem, GROUP BY sözcükleri yardımı ile gerçekleştirilebilir. Örneğin, aşağıdaki soru bu konuda bir fikir verecektir:

Örnek 1 / Her bölümdeki ortalama Gelir nedir?

Burada istene, bölümler bazında ortalama Gelir olduğuna göre, personel tablosundaki satırlar, bölüm numaralarına göre (bol_no) gruplandırarak, her bir grubun Gelir ortalaması ayrı ayrı hesaplanarak listelenebilir. Aşağıdaki SELECT komutu bu işlemi gerçekleştirmektedir.

```
SELECT bol_no, AVG(Gelir)
```

```
FROM Personel
```

```
GROUP BY bol_no;
```

SONUÇ

bol_no	AVG (Gelir)
1	2500
2	6800
3	7400
4	12500

Her bölümdeki en yüksek Geliri alan kişiler listelenmek istenirse, aşağıdaki komut kullanılabilir:

```
SELECT bol_no, MAX(Gelir), ad, soyad
```

```
FROM Personel
```

```
GROUP BY bol_no;
```

Personel tablosundaki bilgiler

GELİR	AD	SOYAD	BOL_NO
5000	Benan	Kar	1
3000	Neşe	Pek	1
8000	Akın	Oran	2
10000	Yeşim	Şensoy	2

şeklinde ise, yukarıdaki SELECT komutunun çıktısı;

Bol_No	Max_Gelir	Ad	Soyad
1	5000	Benan	Kar
2	10000	Yeşim	Şensoy

şeklinde olacaktır.

Gruplandırarak, Graplama fonksiyonlarını uygularken, koşul da verilebilir. Bu durumda, grup üzerindeki hesaplamalarla ilişkili koşul belirtirken, HAVING sözcüğü kullanmak gereklidir. Aşağıdaki örnek soru, bu konuda fikir vermektedir.

Örnek 1 / En yüksek Gelirin, 9000'dan fazla olduğu bölümlerdeki personele ait ortalama Gelirleri listeleyiniz.

```
SELECT bol_no, AVG(Gelir)
      FROM Personel
      GROUP BY bol_no
      HAVING AVG(Gelir) > 9000;
```

Personel tablosunda aşağıdaki veri mevcut olsun:

.....	Bol_No	Gelir
	1		6000
	1		17000
	2		7500
	2		8000
	3		12000
	3		11000
	1		14000
	1		18000

Yukarıdaki SELECT komutu sonucunda

Bol_No	AVG_Gelir
1	13750
3	11500

tablosu elde edilecektir.

HAVING sözcüğü, SELECT komutunda GROUP BY sözcükleri bulunmadığı zaman, geçersizdir. HAVING sözcüğünü izleyen ifade içinde, SUM, COUNT (*), AVG, MAX ya da MIN gibi Graplama fonksiyonlarından en az biri bulunmalıdır.

WHERE sözcüğü bir tablonun tek tek satırları üzerinde işlem yapan koşullar için geçerli iken, HAVING sözcüğü sadece, gruplanmış veriler üzerindeki işlemlerde geçerlidir.

Bazı durumlarda, HAVING ve WHERE sözcükleri birlikte, SELECT komutu içinde kullanılabilir.

Örnek 2 / Personel tablosu içinde, her bölümde, erkek personele ait Gelirlar için, ortalamanın 9000'den fazla olduğu bölümleri listeleyiniz.

```
SELECT bol_no, AVG(Gelir)
      FROM Personel
     WHERE cinsiyet=.T.
      GROUP BY bol_no
     HAVING AVG(Gelir)>9000;
```

Personel tablosunda aşağıdaki bilgiler olsun:

.....	Bol_No	Gelir	Cinsiyet	
	1	6000	.T.	
	1	17000	.F.	
	2	7500	.F.	
	2	8000	.F.	
	3	12000	.T.	
	3	11000	.F.	
	1	14000	.T.	
	1	18000	.T.	

Yukarıdaki uygulanan SELECT komutu, her bölümdeki erkek personele ait ortalama Geliri hesaplayacak (erkek personel.T. ile belirtilmiştir) ve erkek personel Gelir ortalaması, 9000'den olan bölümler listelenecaktır. Komutun çıktısı, aşağıdaki gibidir.

Bol_No	AVG_Gelir
1	12666.67
3	12000

7.4 Tablolarda Değişiklik Yapmak

7.4.1 Tabloya Veri Ekleme

SQL'de, mevcut bir tabloya veri eklemek için kullanılacak olan komut INSERT komutudur. Standart SQL'de, oluşturulan bir tabloya veri yüklemek için tek imkân INSERT komutudur. INSERT komutu ile, tabloya, belli bir anda, tek bir satır eklemek imkanı vardır. INSERT komutunun yazılış biçimi aşağıdaki gibidir.

```
INSERT INTO Tabloadı  
(Sütunadı1,Sütunadı2,.....,Sütunadı n)  
VALUES (değer1,değer2,....,değer n);
```

Örneğin, Personel tablosuna, sicil no'su 275 olan personel ile ilişkili bilgiler aşağıdaki gibi bir INSERT komutu ile yüklenebilir:

```
INSERT INTO Personel(sicil,ad,soyad,dog_tar,adres,cins,Gelir,bol_no)  
VALUES ('275','Benan','Serter',{01/05/62},'Fatih-İstanbul',T.,27,2);
```

Karakter türü verilerin ‘ ‘ sembollerı arasında yüklenidine diğer veriler içinse buna gerek olmadığına dikkat ediniz. Burada, tabloya tüm kolonlarla ilgili veri yükleniği için, istenirse kolon isimleri ihmal edilebilir.

7.4.2 Tablo Satırlarını Silme

Bir tablonun satırlarını silmek için gerekli komut DELETE komutudur. Satır silme koşullu ya da koşulsuz olarak gerçekleştirilebilir.

```
DELETE FROM Tabloadı;  
Örnek:      DELETE FROM Personel;  
  
25 Rows Deleted
```

Bu komut ile Personel tablosundaki tüm satırlar silinecektir. 25 Rows Deleted mesajı ile, o anda tabloda bulunan 25 satırın silindiği bildirilmektedir.

Koşula bağlı olarak satır silmeyi gerçekleştirmek için, DELETE komutuna WHERE sözcüğü eklenmeli ve bunu izleyen ifade koşulu göstermelidir.

```
Örnek:      DELETED FROM Personel  
  
WHERE bol_no=2;
```

5 Rows Deleted

Bu komut ile, 2 numaralı bölümdeki personelin tümü tablodan silinecektir. 5 Rows Deleted mesajı ile de, o anda 2 numaralı bölümde çalışan 5 personele ait satırların silindiğini belirtmektedir.

Aşağıdaki örnekte ise Gelir alanı boş olmayan tüm personel silinecektir.

DELETE FROM Personel

WHERE Gelir IS NOT NULL;

25 Rows Deleted

7.4.3 Tablo Satırlarındaki Verilerde Değişiklik Yapma-Güncelleme İşlemi

Tablo satırlarında güncelleme yapmak için SQL'de UPDATE komutu kullanılır. DELETE komutunda olduğu gibi, UPDATE komutunu da koşullu ya da koşulsuz olarak kullanmak mümkündür. Koşul belirtilmemişse, belirtilen değişiklik tüm tablo satırları üzerinde gerçekleştirilir. Koşul belirtildiği takdirde, sadece koşulu sağlayan satırlar üzerinde değişiklik gerçekleştirilir. UPDATE komutunun yazılış biçimi aşağıdaki gibidir.

Koşulsuz ise,

UPDATE Tabloadı

SET Kolonadı1=değer1,Kolonadı2=değer2,.....,Kolonadı n=değer n;

Koşullu olduğu takdirde,

UPDATE Tabloadı

SET Kolonadı1=değer1,Kolonadı2=değer2,.....,Kolonadı n=değer n

WHERE Koşul;

Aşağıdaki UPDATE komutunun kullanılışı ile ilgili örnekler verilmiştir.

Örnek 1 / Tüm personelin Gelirlerine %12 zam yapma işlemini gerçekleştiriniz.

UPDATE Personel

SET Gelir=Gelir*1.12;

Örnek 2 / 5'inci bölümde çalışan kişilerin Gelirlerine %35 zam yapan UPDATE komutunu yazınız.

```
UPDATE Personel  
SET Gelir?Gelir*1.35  
WHERE bol_no=5;
```

Örnek 3 / 2. bölümün yürüttüğü projelerde kullanılan tüm parçaların fiyatlarına %7 zam yapan UPDATE komutunu yazınız.

```
UPDATE Parça  
SET fiyat=fiyat*1.07  
WHERE pr_no IN (SELECT proj_no  
FROM Proje  
WHERE bl_no=2);
```

Örnek 4 /Sicil numarası 27265421 olan personelin bölüm numarasını 5 olarak değiştiren ve Gelirine %14 zam yapan UPDATE komutunu yazınız.

```
UPDATE Personel  
SET bol_no=5,Gelir=Gelir*1.14  
WHERE sicil='27265421';
```

7.4.4 Tablonun Yapısında Değişiklik Yapma

ALTER TABLE komutu ile bir tablonun yapısında değişiklik yapmak mümkündür. Standart SQL'de bu değişiklikler, tabloya yeni bir kolon ekleme (ADD sözcüğü yardımı ile) ve mevcut bir kolonun özelliklerini değiştirme (MODIFY komutu ile kolon genişliğini değiştirme ya da kolondaki verinin NULL ya da NOT NULL özelliğini değiştirme) şeklindedir.

Standart dışına çıkan bir çok SQL gerçekleştiriminde ise ayrıca tablodan bir kolon silme (DROP), mevcut bir kolonun adını değiştirme (RENAME) ya da tablonun adını değiştirme (RENAME TABLE) özellikleri de, ALTER TABLE komutu içinde mevcuttur.

7.4.4.1 Mevcut Bir Tabloya Bir Kolon Ekleme

ALTER TABLE komutu içinde ADD sözcüğü kullanılarak, mevcut tabloya bir satır eklenebilir.

Mevcut bir tabloya, yeni bir kolon eklenirken, o kolon içindeki verinin türü, uzunluğu ve bu kolondaki verinin boş bırakılıp bırakılmayacağı (NULL veya NOT NULL) özellikleri de belirtilir.

Örnek 1 / Personel tablosuna, işe başlama tarihini belirten yeni bir kolon ekleyiniz.

```
ALTER TABLE Personel
```

```
    ADD is_bas_tar DATE;
```

Yeni eklediğimiz is_bas_tar alanı içinde veri yüklü olmayacağı için boş olacak yani NULL değerler taşıyacaktır. Eğer ADD is_bas_tar DATE NOT NULL; şeklini kullandık, bu kolon satırları gene boş olacaktı; fakat bu kolon ile ilişkili yeni boş değerler eklenmek istendiğinde, buna müsaade etmeyecekti (INSERT komutu ile). ADD sözcüğü ile aynı anda birden çok kolon eklenebilir.

7.4.4.2 Mevcut Bir Tablonun Kolonlarında Değişiklik Yapma (Modify Komutu)

Mevcut bir kolon üzerinde değişiklik yapma, değişken uzunluklu bir veri tipine sahip olan kolonun genişliğini artırma ile sınırlıdır. Bu anlamda, kolon genişliğini azaltma ya da veri tipini değiştirmeye mümkün değildir.

Bu işlem için MODIFY sözcüğü ALTER TABLE komutu içinde kullanılır.

Örnek 1 / Daha önce Proje adlı tabloda VARCHAR(15) olarak tanımlanmış olan yer adlı alanı, 25 olarak genişleten SQL komutunu yazınız.

```
ALTER TABLE Proje
```

```
    MODIFY yer VARCHAR(25);
```

Aynı anda birden çok kolon üzerinde değişiklik yapılabilir. Yukarıda belirtildiği gibi, tabloda daha önce tanımlanmış bir tür (type) başka bir tipe çevrilmez. Örneğin DATE'i MODIFY komutu ile CHAR, ya da INT olan bir alanı VARCHAR şekline dönüştürmek mümkün değildir.

7.4.4.3 Mevcut Bir Tablodan Bir Kolon Silme (Drop Komutu)

Mevcut bir tablodan, bir kolon silmek için, ALTER TABLE komutu içine DROP sözcüğü eklemek gerekecektir. Örneğin, Personel tablosundan, is_bas_tar kolonunu silmek için

```
ALTER TABLE Personel
```

```
    DROP is_bas_tar;
```

komutunu kullanmak gerekir.

Aynı anda birden çok kolon silinebilir. Bu durumda, DROP komutu içinde bunları virgülerle ayırmak gereklidir.

```
ALTER TABLE Personel
```

```
DROP is_bas_tar;
```

Personel tablosundan işe başlama tarihi alanı silinmiştir. Bir tablodan bir kolon silindiği takdirde, bu tablo kullanılarak üretilmiş VIEW'lerdeki ilgili kolonlar da otomatik olarak silinir. İndeks alanı olarak tanımlanmış alanların tablodan silinmesi, sistem tarafından kabul edilmez; önce indeks özelliğinin iptal edilmesi gerekecektir.

7.4.5 Bir Tablonun Adını Değiştirme – Rename Table Komutu

Mevcut bir tablonun adını değiştirmek için, ALTER TABLE komutu içinde RENAME TABLE ifadesi kullanılmalıdır. Örneğin Personel tablosunun adını elemanlar olarak değiştirmek istersek aşağıdaki komutu kullanmak gerekecektir.

```
ALTER TABLE Personel
```

```
RENAME TABLE elemanlar;
```

7.4.6 Mevcut Bir Tablonun Bir Kolonunun Adını Değiştirme – Rename Komutu

Mevcut bir tablonun, bir kolonunun adını değiştirmek için, ALTER TABLE komutu içinde RENAME sözcüğü kullanılmalıdır. Örneğin, Personel tablosunda Gelir alanını, br_Gelir olarak değiştirmek için aşağıdaki komutu kullanmak gereklidir.

```
ALTER TABLE Personel
```

```
RENAME Gelir br_Gelir;
```

7.4.7 Mevcut Bir Tablonun Tümüyle Silinmesi – Drop Table Komutu

Bir tablonun tümünü silmek için DROP TABLE komutu kullanılmalıdır. Örneğin, Proje adlı tablonun silinmesi için aşağıdaki komut gereklidir:

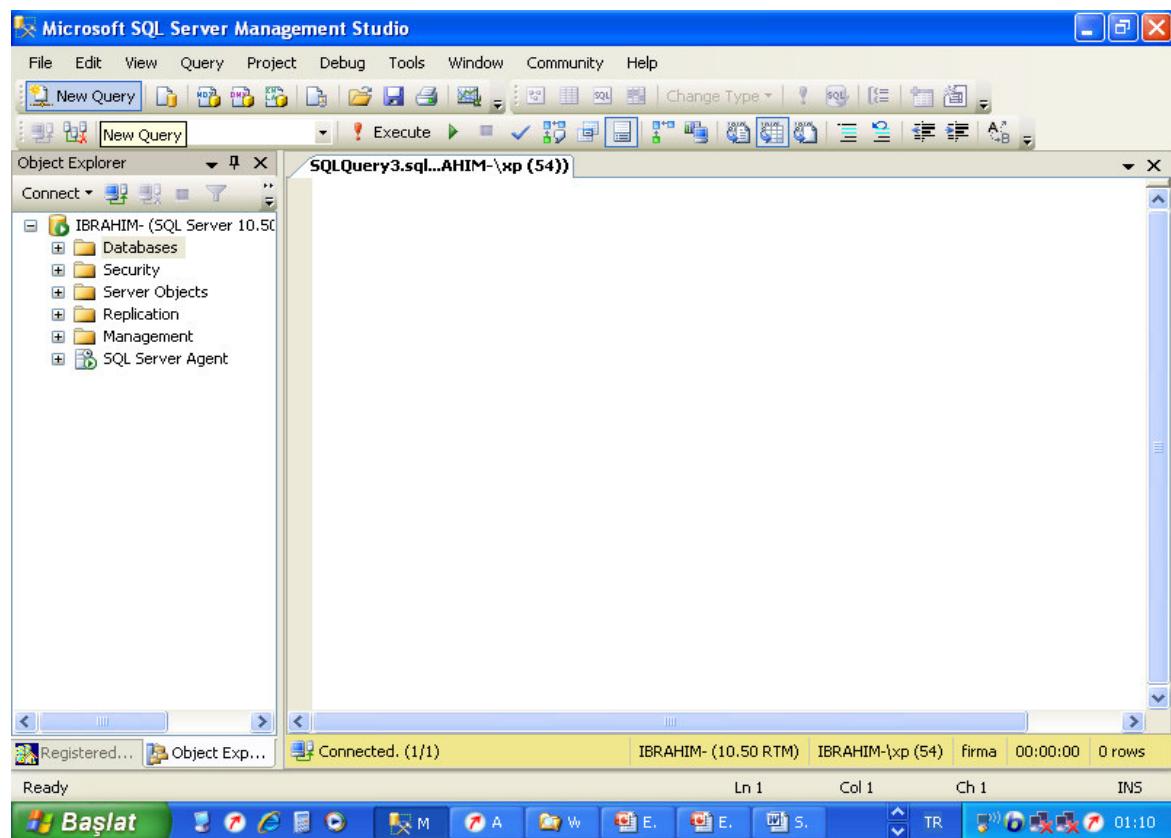
```
DROP TABLE Proje;
```

Veri tabanından bir tablo, DROP TABLE komutu ile silindiği takdirde, bu tablodan üretilmiş bütün VIEW'ler, bu tablodan üretilmiş eş tablolar, tablo üzerindeki indeksler ve tablo için konulmuş bütün öncelikler de sistemden silinir.

7.5 QUERY PENCERESİNDE SQL KOMUTLARI İLE VERİ TABANI OLUŞTURMA VE SORGULAR HAZırlAMA

Bir şirkette personel bilgilerini oluşturan tablo ve bu tablo üzerinde yapılan işlemler aşağıda sırayla ele alınmaktadır. Öncelikle bir çalışma tablosu hazırlanarak hazırlanan bu tabloya kayıtlar girilmekte, kayıtlar üzerinde sorgular yapılmaktadır. Sonuçlarında nasıl gözükeceği ilgili komutun altında verilmektedir.

Önce New query sekmesine tıklayarak yeni bir sorgu penceresi açılır. Daha sonra komutlar burada yazılarak Execute komutuyla çalıştırılır.



CREATE DATABASE bir veritabanı oluşturmak için verilmesi gereken bir komuttur. CREATE DATABASE yazdıktan sonra bir veri tabanı ismi yazılır

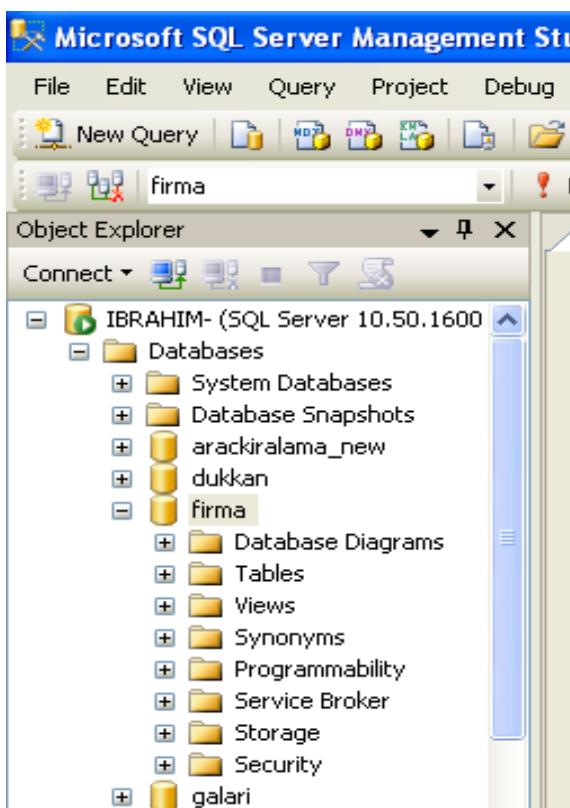
[CREATE DATABASE firma](#)

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, several databases are listed: arackiralama_new, dukkan, galari, karma, market, mirket, and ogrenci. In the center pane, a query window titled 'b2_sgl_ornek...IM-\xp (57)*' contains the following SQL code:

```
CREATE DATABASE firma
use firma
```

In the bottom pane, the 'Messages' tab displays the message: 'Command(s) completed successfully.'

Veritabanı dizininde bu yeni açılan veri tabanı kontrol edilebilir



CREATE TABLE bir tablo oluşturmak için verilmesi gereken bir komuttur. Devamında tablonun adı vardır. Gerekli parametreler parantez içinde int ve varchar ifadelerinden biriyle birlikte verilir. Gireceğimiz veri bir sayı ise int, metin ise varchar ifadesiyle birlikte kullanılır. Unutulmaması gereken tüm komutların ardından noktalı virgül koymaktır. Ayrıca tablo yâda sütun isimleri tek bir kelime olmalıdır.

Bir şirkette personel bilgilerini oluşturan calisanlar tablosu aşağıdaki gibi oluşturulabilir.

```
CREATE TABLE calisanlar( pno int, isim varchar(20), dep int,gorev varchar(20), yil int, Gelir int);
```

Veriler INSERT INTO komutu ardından tablo adı ve VALUES komutuyla ve devamında parantez içinde değerlerle girilir. Metin ifadeleri mutlaka tırnak içerisinde olmalıdır. En sonunda da noktalı virgülü koymalıyız. Verilerin girişinde dikkat edilmesi gereken bir husus aralarına virgül koymaktır. Bir başka husus da metinlerin girişi tırnak içinde olmalıdır.

SELECT * FROM komutu ve ardından tablo adıyla oluşturduğumuz tabloyu görebiliriz. Burada * işaretini yardımıyla sütun isimlerini yazmaya gerek kalmadan tüm sütunları görüşürüz.

```
CREATE DATABASE firma
```

```
use firma;
```

```
CREATE TABLE calisanlar( pno int, isim varchar(20), dep int, gorev varchar(20), yil int, Gelir int);
```

```
INSERT INTO calisanlar VALUES (10, 'Benan', 20, 'yonetici', 7, 900);
```

```
INSERT INTO calisanlar VALUES (11, 'Melike', 20, 'satis', 8, 780);
```

```
INSERT INTO calisanlar VALUES (12, 'Elif', 38, 'yonetici', 5, 880);
```

```
INSERT INTO calisanlar VALUES (13, 'Bantu', 38, 'satis', 6, 920);
```

```
INSERT INTO calisanlar VALUES (14, 'Mehmet', 15, 'yonetici', 10, 1000);
```

```
INSERT INTO calisanlar VALUES (15, 'Esra', 38, 'satis', 1, 650);
```

```
INSERT INTO calisanlar VALUES (16, 'Ece', 15, 'satis', 7, 650);
```

```
INSERT INTO calisanlar VALUES (17, 'Mert', 20, 'memur', 1, 400);
```

```
INSERT INTO calisanlar VALUES (18, 'Erkan', 42, 'memur', 2, 520);
```

```
INSERT INTO calisanlar VALUES (19, 'Seda', 42, 'yonetici', 7, 730);
```

7.5.1 QUERY Penceresinde Yazılan SQL Komutlarının Listesi

```
SELECT * FROM calisanlar;
```

```
SELECT pno, isim, dep, gorev FROM calisanlar;
```

```
SELECT * FROM calisanlar WHERE dep=20
```

`SELECT pno, isim, dep, gorev FROM calisanlar WHERE gorev = 'yonetici' AND dep = 15;`

`Select isim from calisanlar Where isim Like '%an%'`

`SELECT isim, dep, yil FROM calisanlar WHERE yil =7 OR yil =2;`

`SELECT isim, yil, Gelir FROM calisanlar WHERE Gelir < 900 AND Gelir > 650;`

`SELECT pno,Gelir FROM calisanlar WHERE Gelir BETWEEN 400 AND 700;`

`SELECT DISTINCT isim,Gelir,gorev FROM calisanlar WHERE gorev IN ('yonetici', 'satis');`

`SELECT pno,Gelir FROM calisanlar WHERE Gelir BETWEEN 400 AND 700;`

`SELECT DISTINCT isim,Gelir,gorev FROM calisanlar WHERE gorev IN ('yonetici', 'satis');`

`SELECT isim, gorev, yil, Gelir FROM calisanlar WHERE gorev LIKE '%one%';`

`SELECT * FROM calisanlar ORDER BY Gelir;`

`DELETE FROM calisanlar WHERE pno=10;`

`SELECT AVG (Gelir) FROM calisanlar;`

`SELECT max(Gelir) FROM calisanlar;`

`SELECT SUM(Gelir) FROM calisanlar;`

`SELECT MAX(Gelir),gorev FROM calisanlar GROUP BY gorev;`

`SELECT MAX(Gelir),yil FROM calisanlar GROUP BY yil;`

`SELECT gorev, max(Gelir) FROM calisanlar GROUP BY gorev HAVING MAX(Gelir)>7000;`

`UPDATE calisanlar SET Gelir = Gelir * 2 WHERE yil >=6;`

`UPDATE calisanlar SET dep=38 WHERE pno=17;`

`SELECT * FROM calisanlar;`

`SELECT isim,dep, Gelir, Gelir*0.20 as KDV FROM calisanlar;`

7.5.2 Çalıştırılan Sorguların Sonuçları

SELECT *FROM calisanlar;

#calisanlar tablosu aşağıdaki gibi oluşur.

PNO	ISIM	DEP	GOREV	YIL	GELİR
17	Mert	20	Memur	1	400
18	Erkan	42	Memur	2	520
15	Esra	38	Satis	1	650
16	Ece	15	Satis	7	650
19	Seda	42	Yonetici	7	730
11	Melike	20	Satis	8	780
12	Elif	38	Yonetici	5	880
10	Benan	20	Yonetici	7	900
13	Banu	38	Satis	6	920
14	Mehmet	15	Yonetici	10	1000

Tablodan istediğimiz değerleri artık kullanabiliriz. Sadece bazı sütunları görmek isteyebiliriz. SELECT komutu ardından istediğimiz sütunların isimlerini yazar devamında FROM komutu ve tablo adıyla noktalı virgül kullanarak komutu oluştururuz.

#calisanlar tablosundan sadece numarası, adı, departmanı ve görevi hakkındaki verilere su şekilde ulaşılır.

SELECT pno, isim, dep, gorev FROM calisanlar;

#oluşan tablo su şekilde dir.

PNO	ISIM	DEP	GOREV
10	Benan	20	Yonetici
11	Melike	20	Satis
12	Elif	38	Yonetici
13	Banu	38	Satis
14	Mehmet	15	Yonetici
15	Esra	38	Satis
16	Ece	15	Satis
17	Mert	20	Memur
18	Erkan	42	Memur
19	Seda	42	Yonetici

Bazı ilişkisel operatörler var. = eşitliklerde != eşit değil küçük > büyük <= küçük veya eşit >= büyük veya eşit)

SELECT * FROM komutu ardından tablo adını verir ve devamında WHERE komutuyla aradığımız veriyi belirtiriz. En sonunda noktalı virgül kullanmayı unutmamalıyız.

#calisanlar tablosundan 20 nolu departmanda çalışanları görmek istersek vereceğimiz komut söyle olur.

```
SELECT * FROM calisanlar WHERE dep=20
```

#oluşan tablo şöyledir.

PNO	ISIM	DEP	GOREV	YIL	GELİR
10	Benan	20	Yonetici	7	900
11	Melike	20	Satis	8	780
17	Mert	20	Memur	1	400

#yada bazı özelliklere sahip personelin istediğimiz verileri için

```
SELECT pno, isim, dep, gorev FROM calisanlar WHERE gorev = 'yonetici' AND dep = 15;
```

Komutunu kullanırız. Burada WHERE komutuyla birlikte AND komutu iki ayrı özellik istediğimizi ifade ederiz. Yani görevi yönetici olan ve 15 departmanında çalışan insanların numara ve isimlerini ifade eder.

#oluşan tablo şöyledir.

PNO	ISIM	DEP	GOREV
14	Mehmet	15	Yonetici

Aynı şekilde AND kullanımı gibi OR da kullanılabilir.
#Calisanlar tablosundan 7 veya 2 yıl çalışanların listesi isim ve departmanıyla birlikte su şekilde ulaşılır.

SELECT isim, dep, yil FROM calisanlar WHERE yil =7 OR yil =2;

#tablo da şöyledir

ISIM	DEP	YIL
Benan	20	7
Ece	15	7
Erkan	42	2
Seda	42	7
Benan	20	7
Ece	15	7
Erkan	42	2
Seda	42	7

#calisanlar tablosundan Geliri 900 den kucuk 600 den büyük olanların isim ve yıllarıyla birlikte su şekilde ulaşılır.

SELECT isim, yil, Gelir FROM calisanlar WHERE Gelir < 900 AND Gelir > 650;

ISIM	YIL	GELİR
Elif	5	880
Melike	8	780
Seda	7	730

istedigimiz belli değerlerede ulaşabiliriz.

#calisanlar tablosundan Gelirleri 900 den küçük fakat 650ye eşit olmayanların isim yıl ve Gelirları söyle elde edilir.

SELECT isim, yil, Gelir FROM calisanlar WHERE Gelir < 900 AND Gelir <> 650;

ISIM	YIL	GELİR
Elif	5	880
Erkan	2	520
Melike	8	780
Mert	1	400
Seda	7	730

Bunların yanında BETWEEN ve IN kullanımlarında mevcuttur.
#calisanlar tablosundan Gelirları 400 'le 700 arasında olanların listesi için
SELECT pno, Gelir FROM calisanlar WHERE Gelir BETWEEN 400 AND 700;

PNO	GELİR
18	520
18	520
17	400
17	400

#calisanlar listesindeki satis ve yonetici görevindekilerin listelerinin IN kullanılmasıyla söyle görülür.

SELECT DISTINCT isim,Gelir,gorev FROM calisanlar WHERE gorev IN ('yonetici', 'satis');

ISIM	GELİR	GOREV
Ayhan	7200	Yonetici
Banu	7360	Satis
Ece	5200	Satis
Elif	3520	Yonetici
Mehmet	8000	Yonetici
Melike	6240	Satis
Seda	5840	Yonetici

#calisanlar tablosundan görevinin içinde 'one'gecenlerin listesi isim,yil,ve Gelirleriyle birlikte

SELECT isim, gorev, yil, Gelir FROM calisanlar WHERE gorev LIKE '%one%';

ISIM	GOREV	YIL	GELİR
Ayhan	Yonetici	7	900
Elif	Yonetici	5	880
Mehmet	Yonetici	10	1000
Seda	Yonetici	7	730

Hazırladığımız tabloyu istediğimiz özelliğe göre dizebiliriz.

#calisanlar tablosunu Gelira göre düzenlemek için.

`SELECT * FROM calisanlar ORDER BY Gelir;`

PNO	ISIM	DEP	GOREV	YIL	GELİR
17	Mert	20	Memur	1	400
18	Erkan	42	Memur	2	520
15	Esra	38	Satis	1	650
12	Elif	38	Yonetici	5	3520
16	Ece	15	Satis	7	5200
19	Seda	42	Yonetici	7	5840
11	Melike	20	Satis	8	6240
10	Ayhan	20	Yonetici	7	7200
13	Banu	38	Satis	6	7360
14	Mehmet	15	Yonetici	10	8000

Tablodan istediğimiz değerleri silmememizde mümkün.

#calisanlar tablosundan 15 nolu personeli sileceğiz.

`DELETE FROM calisanlar WHERE pno=15;`

PNO	ISIM	DEP	GOREV	YIL	GELİR
10	Ayhan	20	YONETİCİ	7	7200
11	Melike	20	SATİŞ	8	6240
12	Elif	38	YONETİCİ	5	3520
13	Banu	38	SATİŞ	6	7360
14	Mehmet	15	YONETİCİ	10	8000
16	Ece	15	SATİŞ	7	5200
17	Mert	38	MEMUR	1	400
18	Erkan	42	MEMUR	2	520
19	Seda	42	YONETİCİ	7	5840

Kullanılan bazı fonksiyonlar vardır. Bunlar MIN en küçük değeri, MAX en büyük değeri, SUM toplamı, AVG ortalama değeri, COUNT şutuna girilen değerlerin toplamını gösterir, COUNT(*) satırı girilen verilerin sayısını gösterir. Hepsinin kullanımını birbirine benzer.

#calisanlar tablosundaki Gelirlain ortalamasını görüntüleyelim.

`SELECT AVG (Gelir) FROM calisanlar;`

AVG
4920

Tablodaki herhangi bir verinin maximimunu veya minimumunu görebiliriz.

#calisanlar tablosundaki en yüksek Geliri görmek için,

SELECT max(Gelir) FROM calisanlar;

MAX
8000

#calisanlar de bulunan Gelir sutununun toplamı

SELECT SUM(Gelir)as [maap toplamý] FROM calisanlar;

SUM
88560

#calisanlardeki maximum Geliri goreve gore gruplandırılalım.

SELECT MAX(Gelir),gorev FROM calisanlar GROUP BY gorev;

MAX	GOREV
520	Memur
7360	Satis
8000	Yonetici

Yukarıdaki tabloda her görevin maximimum Gelirini gördük. Fakat eğer istediğimiz belli değerler ise HAVING komutunu kullanabiliriz.

#calisanlar tablosundaki Gelirlarin gorevlere Gore grublandırduğımızda maximumu 7000 den yüksek görevleri görelim.

SELECT gorev, max(Gelir) FROM calisanlar GROUP BY gorev HAVING MAX(Gelir)>7000;

GOREV	MAX
Satis	7360
Yonetici	8000

Hazırladığımız tabloyu güncelleştirme olanağımızda var. UPDATE komutu ardından tablo adı verir, SET komutundan sonra değiştirmek istediğimiz parçayı nasıl değiştmesini anlatmalıyız.

#Calisanlar tablosunda altı ve 6 yıldan daha fazla çalışanların Gelirlerini iki katına çıkarmak için:

UPDATE calisanlar SET Gelir = Gelir * 2 WHERE yil >=6;

PNO	ISIM	DEP	GOREV	YIL	GELİR
10	Ayhan	20	Yonetici	7	7200
11	Melike	20	Satis	8	6240
12	Elif	38	Yonetici	5	3520
13	Banu	38	Satis	6	7360
14	Mehmet	15	Yonetici	10	8000
15	Esra	38	Satis	1	650
16	Ece	15	Satis	7	5200
17	Mert	20	Memur	1	400
18	Erkan	42	Memur	2	520
19	Seda	42	Yonetici	7	5840

17 numaralı personelin departmanini 38 olarak değiştiriyorum.

UPDATE calisanlar SET dep=38 WHERE pno=17;

PNO	ISIM	DEP	GOREV	YIL	GELİR
10	Ayhan	20	Yonetici	7	7200
11	Melike	20	Satis	8	6240
12	Elif	38	Yonetici	5	3520
13	Banu	38	Satis	6	7360
14	Mehmet	15	Yonetici	10	8000
15	Esra	38	Satis	1	650
16	Ece	15	Satis	7	5200
17	Mert	38	Memur	1	400
18	Erkan	42	Memur	2	520
19	Seda	42	Yonetici	7	5840

tabloya geçici olarak bir sutun eklenebilir

#calisanlar tablomuzda Gelirin %20 sini KDV olarak bir sutunda görmek istersek

SELECT isim,dep, Gelir, Gelir*0.20 as KDV FROM calisanlar;

ISIM	DEP	GELİR	KDV
Ayhan	20	7200	1440
Banu	38	7360	1472
Ece	15	5200	1040
Elif	38	3520	704
Erkan	42	520	104
Esra	38	650	130
Mehmet	15	8000	1600
Melike	20	6240	1248
Mert	38	400	80
Seda	42	5840	1168

Uygulama Soruları

Soru 1

Firma averitabanını oluşturalım.

```
use Firma
```

```
go
```

```
CREATE DATABASE Firma
```

Soru 2

Özellikleri aşağıda verilen Müşteri tablosunu oluşturunuz.

Kolon Adı	Data Tipi	Kısıtlamalar
MusteriNo	İnt	
MusteriAdi	varchar(30)	
Adres	varchar(100)	

```
Use Firma
```

```
go
```

```
CREATE TABLE tbl_Musteri (MusteriNo int, MusteriAdi varchar(30), Adres  
varchar(100) )
```

Soru 3

Özellikler aşağıda verilen Siparis tablosunu oluşturunuz.

Kolon Adı	Data Tipi	Kısıtlamalar
SiparisNo	int	
MusteriNo	int	
SiparisTarihi	datetime	

Use Firma

go

`CREATE TABLE tbl_Siparis`

(

SiparisNo int, MusteriNo int, SiparisTarihi datetime

)

go

Soru 4

Özellikleri aşağıda verilen SiparisDetay tablosunu oluşturunuz.

Kolon Adı	Data Tipi	Kısıtlamalar
SiparisNo	int	
UrunNo	char(10)	
Miktar	int	
Fiyat	money	
Birim	char(5)	

Use Firma

go

`CREATE TABLE tbl_SiparisDetay`

(

SiparisNo int,

UrunNo char(10),

Miktar **int**,
Fiyat **money**,
Birim **char(5)**
)

Soru 5

Özellikleri aşağıda verilen Ürün tablosunu oluşturunuz.

Kolon Adı	Data Tipi	Kısıtlamalar
UrunKodu	Char(10)	Birincil Anahtar not null
UrunAdi	Varchar(30)	Not null
Grubu	Char(2)	Varsayılan '01'
Fiyati	Money	
Birim	Char(5)	Varsayılan ADET

CREATE TABLE tbl_Urun(

UrunKodu **char(10) primary key** not null, UrunAdi **Varchar(30) NOT NULL**, Grubu **char(2) default '01'**, Fiyati **money**, Birimi **char(5) default 'ADET'**)

Soru 6

Ürün tablosuna veri giriniz.

INSERT tbl_Urun (UrunKodu, UrunAdi) **VALUES** ('15010','Defter')

INSERT tbl_Urun (UrunKodu, UrunAdi) **VALUES** ('15015','Ataç')

INSERT tbl_Urun (UrunKodu, UrunAdi,Grubu,Birim) **VALUES**
('15045','Kalem',NULL, 'KUTU')

INSERT tbl_Urun (UrunKodu, UrunAdi,Grubu) **VALUES** ('19010','Silgi',NULL)

INSERT tbl_Urun (UrunKodu, UrunAdi,Grubu,Fiyati,Birim) **VALUES** ('12014','Kuru Boya', **DEFAULT**, 15, NULL)

INSERT tbl_Urun (Fiyati, Grubu,UrunAdi, UrunKodu) **VALUES**
(19,'21','KAĐIT','21151')

Tüm listeyi Görmek için aşağıdaki kodu yazınız.

SELECT * FROM tbl_Urun

Soru 7

Müşteri tablosuna veri giriniz.

`INSERT tbl_Musteri (MusteriNo, MusteriAdi, Adres) VALUES (1,'A LTD','Beykent')`

`INSERT tbl_Musteri (MusteriNo, MusteriAdi, Adres) VALUES (2,'B LTD','Beykent')`

`INSERT tbl_Musteri (MusteriNo, MusteriAdi, Adres) VALUES (3,'C LTD','İstanbul')`

`INSERT tbl_Musteri (MusteriNo, MusteriAdi, Adres) VALUES (13,'A A.Ş.','B.Çekmece')`

Tüm listeyi Görmek için aşağıdaki kodu yazınız.

`SELECT * FROM tbl_Musteri`

Soru 8

Sipariş tablosuna veri giriniz.

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (1,1,'2008-09-10')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (2,3,'2008-10-10')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (3,3,'2008-09-10')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (4,2,'2008-11-30')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (5,1,'2008-11-18')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (6,3,'2008-11-18')`

`INSERT tbl_Siparis (SiparisNo, MusteriNo,SiparisTarihi) VALUES (7,13,'2008-05-27')`

Tüm listeyi Görmek için aşağıdaki kodu yazınız.

`SELECT * FROM tbl_Siparis`

Soru 9

SiparisDetay tablosuna veri giriniz.

INSERT tbl_SiparisDetay (SiparisNo, UrunNo,Miktar,Fiyat,Birim) **VALUES**
(1,'15010', 10, 1.5, 'ADET')

INSERT tbl_SiparisDetay (SiparisNo, UrunNo,Miktar,Fiyat,Birim) **VALUES**
(1,'15015', 20, 2.7, 'PAKET')

INSERT tbl_SiparisDetay (SiparisNo, UrunNo,Miktar,Fiyat,Birim) **VALUES**
(2,'15015', 12, 7, 'ADET')

INSERT tbl_SiparisDetay (SiparisNo, UrunNo,Miktar,Fiyat,Birim) **VALUES**
(3,'15045', 1, 1.99, 'KUTU')

INSERT tbl_SiparisDetay (SiparisNo, UrunNo,Miktar,Fiyat,Birim) **VALUES**
(3,'15010', 7, 1.5, 'ADET')

Go

Tüm listeyi Görmek için aşağıdaki kodu yazınız.

SELECT * FROM tbl_SiparisDetay

Soru 10

Birimi Null olan kayıtlarda birimi değerini KUTU yapınız.

UPDATE tbl_Urun SET Birimi='KUTU' WHERE Birimi is null

Soru 11

15045 nolu ürünün Grubunu 44, Fiyatını =4.7 yapınız.

UPDATE tbl_Urun SET Grubu='44',

Fiyati = 4.7 WHERE UrunKodu = '15045'

Soru 12

Ürün tablosunda grubu null olan kayıtları 44 yapınız.

`UPDATE tbl_Urun SET Grubu='44' WHERE Grubu is null`

Soru 13

Sipariş tablosunda 3 nolu siparişin Sipariş Tarihini bugün yapınız.

`UPDATE tbl_Siparis SET SiparisTarihi = getdate() WHERE SiparisNo=3`

Soru 14

Sipariş tablosunda 6 nolu siparişin Sipariş Tarihini 25-12-2007 yapınız.

`UPDATE tbl_Siparis SET SiparisTarihi = '2007-12-25' WHERE SiparisNo=6`

Soru 15

Sipariş tablosunda 13 nolu siparişin Sipariş Tarihini NULL yapınız.

`UPDATE tbl_Siparis`

`SET SiparisTarihi = NULL`

`WHERE SiparisNo=13`

Soru 16

SiparişDetay tablosundan 1 nolu Siparişte bulunan 15015 nolu ürünün miktarını 5 arttırınız ekleyiniz.

`UPDATE tbl_SiparisDetay`

`SET Miktar = Miktar + 5`

`WHERE SiparisNo=1 and UrunNo='15015'`

Soru 17

SiparişDetay tablosundan 3 nolu Siparişte bulunan 15010 nolu siparişte ürünü 3 eksitiniz ve fiyatını %5 artturınız.

UPDATE tbl_SiparisDetay

SET Miktar = Miktar - 3,

Fiyat = Fiyat * 1.05

WHERE SiparisNo=3 and UrunNo = '15010'

Soru 18

SiparişDetay tablosundan Miktar değeri 10 dan fazla olanın fiyatı değerini %7 azaltınız.

UPDATE tbl_SiparisDetay

SET Fiyat = Fiyat * 0.93

WHERE Miktar > 10

Bu Bölümde Ne Öğrendik Özeti

Bu hafta aritmetiksel ifadeler ve gruptama fonksiyonları ile ilgili sorgu komutlarının yapısını anlamaya çalıştık. Ayrıca Query penceresinde SQL komutları ile ilgili uygulamalar ile bu sorguların nasıl çalıştığını sergilemiş olduk.

Bölüm Soruları

1. SELECT isim, dep, Gelir, Gelir*0.20 as KDV FROM calisanlar;

- a) Hatalı bir sorgu yazılmış
- b) calisanlar tablosunda tüm alanları gösterir
- c) calisanlar tablosunda Gelirin %20 sini hesaplar
- d) calisanlar tablosunda Gelirin %20 sini KDV olarak bir sutunda gösterir
- e) calisanlar tablosunda Gelirin KDV sini hesaplar

2. COUNT fonksiyonu ile ne yapılır?

- a) Sayısal ifadeyi yuvarlama hesabını yapar.
- b) Tablo içinde, belirlenen sütun (alan) içindeki en küçük değeri bulur.
- c) Sadece, belirtilen sütunun işleme sokulmasını sağlar.
- d) Tablo içinde, belirlenen sütun (alan) içindeki en büyük değeri bulur.
- e) Tablo içerisinde herhangi bir sayma işlemi gerçekleştirmek için kullanılır.

3. HAVING sözcüğünü izleyen ifade içinde, aşağıdakilerden hangisi kullanılmaz

- a) SUM,
- b) COUNT (*),
- c) ROUND
- d) AVG,
- e) MAX

CEVAPLAR

1-D, 2-E, 3-C

8. VERİTABANI MİMARİSİ

Bu Bölümde Neler Öğreneceğiz?

8.1 Veritabanı Mimarisi

8.1.1 Yerleşik Model Veritabanları

8.1.2 İstemci/Sunucu Modeli Veritabanları

8.1.3 Veri Tabanlarının Web İle Birleştirilmesi

8.1.4 Dağıtık Veritabanları

8.2 Standart Veri Tabanı Mimarisi

8.2.1 Üç Düzeyli Veri Tabanı Mimarisi

8.2.1.1 Dış (Görünüm) Seviye

8.2.1.2 Mantıksal Seviye

8.2.1.3 İç (Fiziksel) Seviye

8.3 Katmanlı Yazılım Mimarileri

8.3.1 Tek-Katmanlı Veritabanı Mimarisi

8.3.2 İki katmanlı (2-Tier) Veritabanı Mimarisi

8.3.3 Üç-Katmanlı Veritabanı Mimarisi

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Veritabanı Mimarisi nedir?
- 2)** İstemci/Sunucu Modeli Veritabanları nasıl çalışır?
- 3)** Katmanlı Yazılım Mimarileri hangileridir?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Veritabanı Mimarısını bilir	Anlatım, Soru-Cevap, Tartışma
	İstemci/Sunucu Modelinin işleyişini bilir	Alıştırma ve Uygulama, Örnek Olay
	Katmanlı Yazılım Mimarilerini bilir	Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

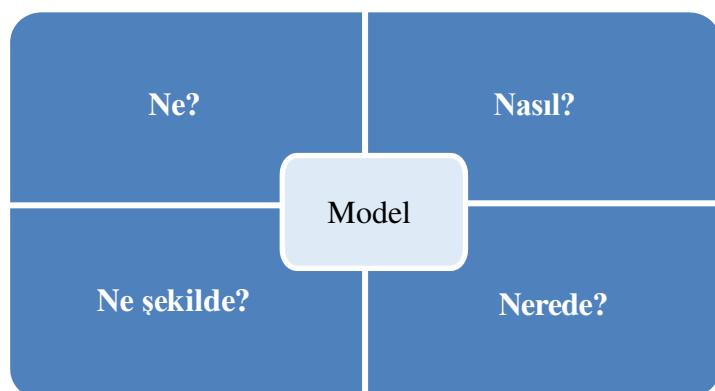
Anahtar Kavramlar

- Veritabanı Mimarisi
- İstemci/Sunucu Modeli
- Katmanlı Yazılım Mimarileri

1. Giriş

Sistem mimarisi, veri tabanları, arayüz araçları ve iletişim gibi bileşenlerin şekillendirilme biçimidir. Geleneksel Sistemler, ana bilgisayar ya da mini bilgisayar kullanan, terminaller üzerinden erişilen böülümlere yönelik çözümler olarak gelişmiştir. Başlangıçta kişisel bilgisayarlar ise, ana sistemlerden ayrı olarak kelime işlemcileri ya da hücre tabanlı programlar gibi özel uygulamalar için kullanılırdı. Daha sonraki aşamalarda bir ofisteki kişisel bilgisayarlar birbirlerine bir bölgesel alan ağı (Local Area Network-LAN) şeklinde bağlandılar. Böylece kullanıcılar, dosya paylaşımı, elektronik posta ve bunlar gibi diğer uygulamaları kullanabildiler. Bu ağların tüm şirkete genişletilmesiyle geniş alan ağları (Wide Area Network – WAN) oluştu. Sonunda bu ağların kişisel bilgisayarların gücünden ve kolay kullanımlı grafik arayüzlerinden faydallanması amacıyla yeni sistemler geliştirildi. Bu sistemlerde PC'lere istemci (client) ve ana işlemcilere ise sunucu (server) denmektedir. İstemci/Sunucu sistemleri, bazı süreçlerin birçok kullanıcı için merkezi olarak yerine getirildiği bazlarının da kullanıcının kişisel bilgisayarında yerine getirildiği bir dağıtılmış işleme şeklidir.

Veri mimarisi verilerin ne olduğunu, nasıl, ne şekilde ve nerede saklandığının biçimlendirilmesidir (Şekil 1).



Şekil 1.Veri Mimarisi

8.1 Veritabanı Mimarisi

Temel olarak iki adet "veritabanı mimarisi" vardır:

- Yerleşik model (Standalone Model)
- İstemci/Sunucu model (Client/server Model)

8.1.1 Yerleşik Model Veritabanları

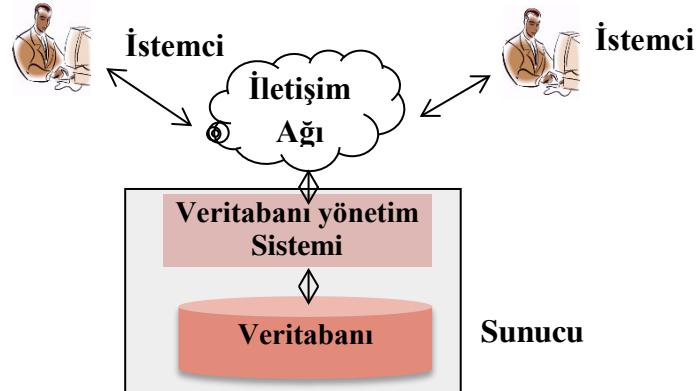
Yerleşik Model veritabanı modelinde, veritabanı ve veritabanı motoru aynı makine üzerinde yerleşik dosya sistemi içerisinde yer almaktadır. Sadece bir tek kullanıcı veri tabanına erişebilir. (Aynı anda iki kişi veri tabanına erişemez.). Genelde makine ağ sistemlerine bağlanmaz. Zaten veritabanının kendisi ağ kullanıcılarına destek vermez. Veritabanı içindeki bilgiler diğer makinelere paylaştırılamaz. Her kullanıcı "lokal" olarak kendi verisini düzenleyebilir. Bu tip veri tabanlarına örnek MS Access verilebilir.

8.1.2 İstemci/Sunucu Modeli Veritabanları

Bu tip veritabanları biraz önce açıklanan, "Yerleşik Model veritabanı Sisteminden" devir alınan sınırlamaları kaldırması amacıyla geliştirilmişlerdir. Yapısı bir "yemek restoranın" isleyiş sistemi ile benzerlikler gösterir: "Gelen tüm müşteriler garsondan servis almak için istekte bulunurlar. " Sistem isleyişinin anahtar kelimesi budur. "İsteği alan garson, isteğin yapısına göre şef veya barmenden yardım almak isteyecektir. Müşteriler servisin nasıl ele alındığını bilmek durumunda veya kaygısı içinde değildirler. Bu isteğin karşılanması için kaç kişi çalışmıştır bu durumda isteği yapan müşteriyi pek ilgilendirmez. Aynı anda gelen bütün müşterilerden, garson sorumludur. Eğer garson herkesi memnun edecek kadar verimli ise, belirli bir zaman dilimi içinde gelen istekleri isleyecek ve sonuçlarını müşteriye ulaştıracaktır. " Burada sözü edilen garson ile istemci/sunucu ilişkisindeki sunucu temsil edilmektedir. Sunucu kaynakları ne kadar güçlü ise, müşteriler programların çalışması sırasında, işlemcinin çalışması için gecen sureyi anlamayacaklardır.

Başka bir örnek vermek gerekirse, amazon.com'dan bir kitap almaya çalıştığımızı düşünelim. Bu noktada istemci programınız -web tarayıcınız amazon.com web sunucusuna bir "istek" te bulunur. Bu istek "web sunucu" tarafından kendi iç sisteminde artık bir "istemci" olarak tabir edebileceğimiz Başka bir programa geçirilir. Bu program yardımıyla "veri tabanında" sorgulama yapılarak "dönen cevap" tekrar isteğin yapıldığı "web sunucusuna" bildirilir. Gelen bilgiyi isleyen web sunucusu bunu direkt olarak kullanıcının tarayıcısına gönderecektir. Bu tür veritabanı sistemlerinde, veritabanı bir dosya sunucu üzerinde bulunur ve veriler bu bilgisayar üzerinde saklanır. Bu bilgisayar istemci/Sunucu veritabanının Sunucu kısmını belirtir. Bu veritabanına başka bilgisayarlar üzerinden erişen kullanıcılar ise istemci kısmını belirtir. istemci/Sunucu veritabanı kullanıcıları bir ağ ortamına yayılmış durumdadırlar. Bu şekilde her kullanıcı farklı konumlardan veritabanına aynı anda erişebilmektedir. Bu kullanıcılar sunucu üzerindeki veritabanı ile direkt olarak hemen hemen hiç uğraşmazlar. Bunun yerine kullanıcının çalışmakta olduğu bilgisayar üzerindeki uygulama ile veritabanına

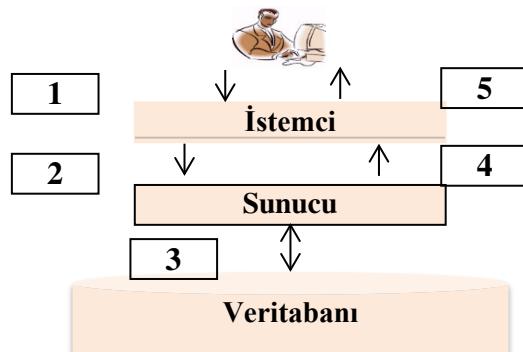
erişim sağlanır. Yerel bilgisayar üzerindeki bu uygulamalara istemci uygulamaları denir. istemci/Sunucu mimarilerinde işletim sistemi seçimindeki yaklaşım, istemci uygulamaların yer aldığı istemci bilgisayar üzerinde Windows gibi işletim sistemlerinin bulunması, sunucu bilgisayar üzerinde ise sunucu işletim sistemlerinin bulunması şeklidir. Çünkü sunucu işletim sistemlerinin güvenli olması, daha çok sunucu hizmetleri odaklı olması ve pahalı olması gibi nedenlerden dolayı bu yaklaşım tarzı seçilmektedir. (Şekil 1).



Şekil 1 Basit İstemci-Sunucu Mimarisi

İstemci İle Sunucu Aynı Makinada Olması

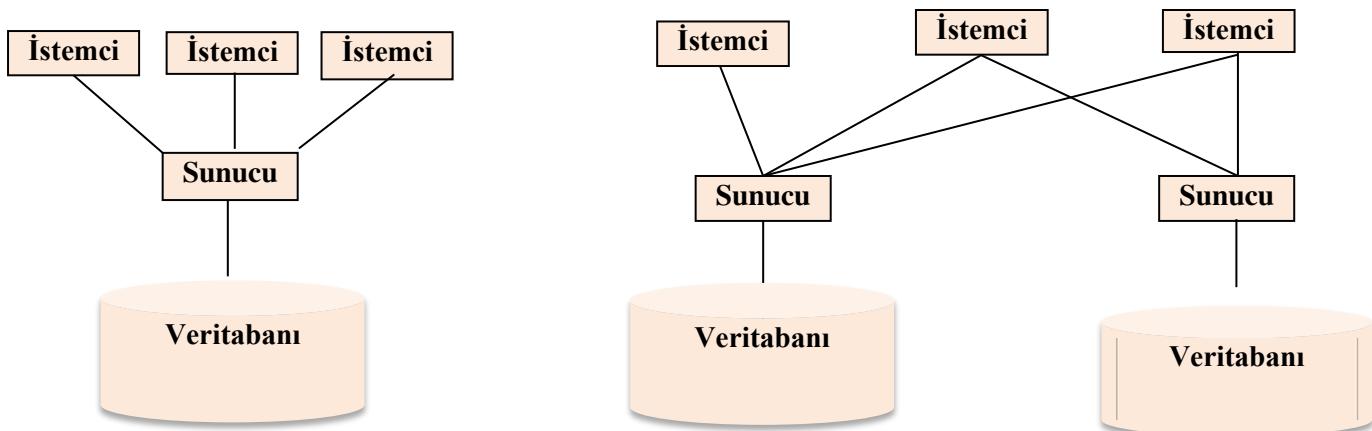
İstemci ile sunucu aynı makinada ya da farklı makinalarda olabilirler (Şekil 2). Sunucu: VTYS işlevlerini yerine getirir. İstemci: kullanıcı ile sunucu arasında etkileşimi sağlar. Hazır paketler (sorgu dili işleyiciler, rapor üreteçleri vs.). Yada uygulama programlarının yazdıkları programlarla sağlanır.



Şekil 2 İstemci ile sunucu aynı makinada

İstemci İle Sunucu farklı Makinalarda Olması

Birden çok istemci bir sunucuya bağlanarak çalışabilir. Örnek (Banka), Merkezde Bir Sunucu ve Şubelerde istemiciler. Sunucular da dağıtık olabilir. Örnek (banka), Her şube kendi hesaplarının sunucusu (ve istemcisi) ve Her şube diğer şube hesaplarının istemcisi



Şekil 3 a) Çok İstemci/Tek Sunucu b) Çok İstemci/Çok Sunucu

8.1.3 Veri Tabanlarının Web İle Birleştirilmesi

Internet de yapısı itibariyle bir istemci/sunucu şéklidir. Yerel PC tarayıcıları, sunuculardan alınan HTML sayfalarını ve Java uygulamalarını işlemektedir. İstemci/sunucu modeli, istemicilerin web sunucusuna bağlı bir web tarayıcısı olduğu web merkezli bir yapıya doğru kaymaktadır. Bilgisayar sektörü, bir ağ PC modeline ulaşmıştır. Bu modelde ağ PC, gerekli uygulamaları indirebilen bir web tarayıcısıdır ve normal PC'den ucuzdur. Ancak sunucularda çok daha fazla işlemin yapılması gereği ve ağda daha fazla trafik olacağı sorun olarak görülmektedir.

Bugün, bilgisayarlar çoğunlukla artık tek başına değil Ağ'lar halinde kullanılmaktadır. Bir iletişim Ağ'ı ile birbirine bağlanmış birden çok bilgisayar birlikte kullanılmaktadır. Bütün işleri tek bir büyük bilgisayarın yaptığı “merkezi işlem”in aksine, “dağıtık işlem” şeklinde dağıtilır. Yaygın olarak kullanılan bir “dağıtık işlem” şekli istemci/sunucu yaklaşımıdır.

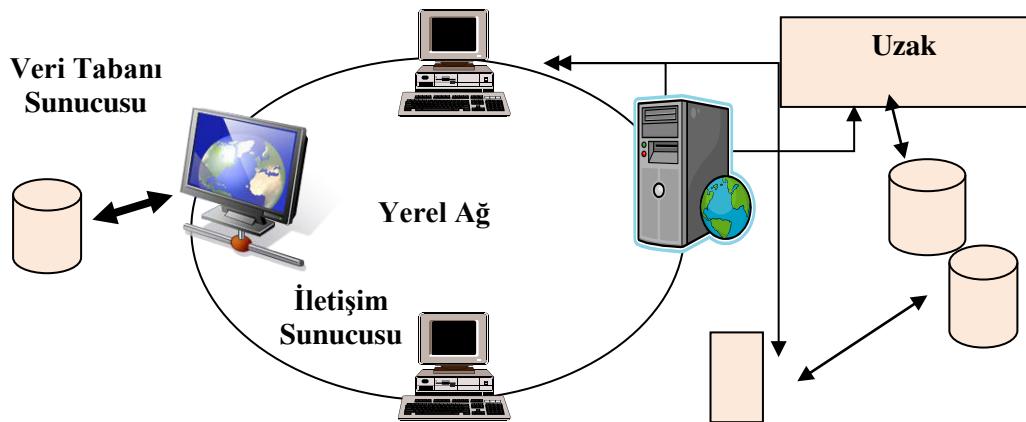
İstemci/Sunucu yaklaşımında işlemler “istemiciler” ve “Sunucu’lar” arasında paylaştırılır. Hem istemci hem de sunucu aynı Ağ'dadır fakat her biri en iyi yaptığı işi yapmakla görevlidir. İstemci/Sunucu yaklaşımında, kullanıcılar istemci makinesini kullanarak Sunucu'daki hizmetlere ve verilere erişebilir.

İstemci/sunucu modeli, istemicilerin web sunucusuna bağlı bir web tarayıcısının olduğu web merkezli bir yapıya doğru kaymaktadır. İstemci, normal olarak bir masaüstü ya da dizüstü PC olup, kullanıcı tarafını temsil eder. Kullanıcı genel olarak uygulamanın istemci kısmı ile muhatap olur ki, işi ya veri girmek ya da veri almaktır. Sunucu ise istemci 'ye bu çerçevede çeşitli hizmetler sunar.

8.1.4 Dağıtık Veritabanları

İletişimlerin ve ağların kullanımının artması, dağıtık işlem (distributed processing) yönüne doğru bir eğilim başlatmıştır. Dağıtık veri tabanlarına geçmeden önce, dağıtık işlemenin kısaca açıklanmasında fayda vardır. Bir iletişim ağıyla bağlanmış birden fazla bilgisayarın işleme amaçlı kullanımı, dağıtık işlem olara bilinmektedir. Büyük bir merkezi

bilgisayarla tüm işlemlerin yapıldığı merkezi işleme ye karşılık, dağıtık işleme, işleme işini birbirine bağlı PC ve mainframe ler arasında dağıtmaktadır.



Şekil 4 Dağıtık Veri Tabanları

Her bir bilgisayar, diğerine dayanmaksızın görevlerin bir alt kümesinde çalışmaktadır. Dağıtık işlemenin en fazla kullanım şekli Sunucu-İstemci (Client-Server) hesaplama şeklärindedir. Sunucu-istemci hesaplama, işlemleri "istemci" ve "sunucu" arasında bölmektedir. Kullanıcı, uygulamanın istemci parçasıyla genellikle veri girilmesi ya da çağrılmazı şeklinde doğrudan etkileşmektedir. Sunucular, ağ üzerindeki diğer bilgisayarlara yazılım ve diğer kaynakları sağlamak için özel olarak optimize edilmiş bilgisayarlardır. Sunucu, istemciye veri ve hizmet sağlar, paylaşılmış veriyi saklar, işler ve ağ faaliyetlerini yönetme gibi kullanıcıya görünmeyen fonksiyonları da yerine getirebilir. Veri tabanı tasarımlı, verilerin nasıl dağıtilacağını da dikkate almaktadır. Bilgi sistemleri, tek bir merkezi işlemci ya da sunucu-istemci ağındaki birden fazla işlemci tarafından kullanılan merkezi bir veritabanı ya da dağıtık bir veritabanı olarak tasarlanabilir. Dağıtık veritabanı, gerçek verinin birden fazla farklı fizikal lokasyona dağıtılabildiği (saklandığı) bir veritabanıdır. Veritabanının parçaları bir lokasyonda, diğer parçalan da diğer lokasyonda fiziksel olarak saklanabilmektedir. Bir veritabanının dağıtılmrasında iki ana yol vardır. Birincisi, her bir uzak işlemcinin (remote processor) kendi kısmi bölgesine hizmet etmek için gerekli veriye sahip olabilmeleri açısından merkezi veritabanının bölünmesidir. Kısıtlı dosyalardaki değişiklikler, parti olarak (genellikle akşam) merkezi veri tabanıyla birleştirilebilir. Diğer strateji, merkezi veri tabanını tüm uzak noktalarda kopya (tekrar) edilmesidir. Lufthansa hava yolları tekrarlı veri tabanıyla çalışmaktadır. Lufthansa'nın Frankfurt VTYS'ye yapılan herhangi. Bir değişiklik, otomatik olarak New York ve Hong Kong'da tekrarlanacaktır. Dağıtılmış sistemler, genellikle küçük pahalı olmayan bilgisayarlar üzerinde çalıştırılabilirler ancak genellikle yüksek kalite iletişim ağı gerektirmektedirler.

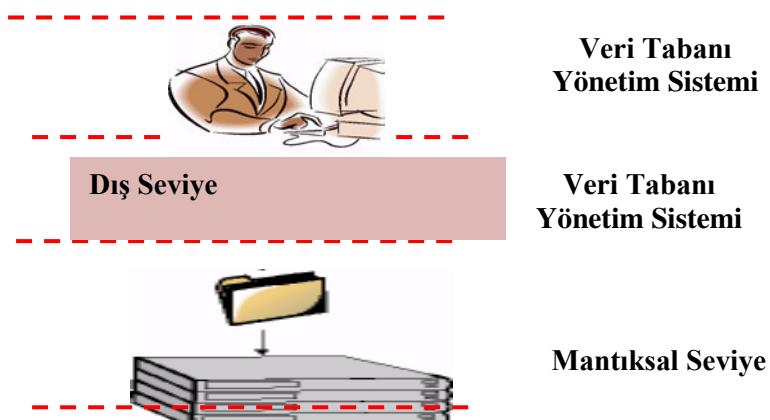
8.2 Standart Veri Tabanı Mimarisi

Bir veri tabanı kullanıcısı için, veri tabanı dış şemadır. Kullanıcı dış şemayı görür ve sağlanan yazılım olanakları ile dış şemada yetkili olduğu işlemleri gerçekleştirir. Kullanıcıların çoğu kavramsal ve iç şemadan habersizdir, verilerin dış şemaya uygun biçimde saklandığını

düşünür. Oysa dış ve kavramsal şemalar tümüyle mantıksaldır. Kullanıcı tarafından dış şemaya göre oluşturulacak isteklerin iç şemada karşılanması gereklidir. Bunun için de kullanıcı tarafından dış şemaya göre tanımlanan verilerin önce kavramsal şemadaki, sonra da iç şemadaki karşılıklarının belirlenmesi ve kullanıcı isteğinin fiziksel veri tabanı üzerinde gerçekleştirilmesi gereklidir. Eğer istenen bir soru ise, iç şemaya göre seçilen veriler, bu defa önce kavramsal sonra da dış şemaya göre dönüştürülecek kullanıcıya sunulmalıdır. Bir VTYS'nin mantıksal ve fiziksel veri bağımsızlığının sağlanması için, şema tanımlarına ek olarak şemalar arası eşleme tanımlarının da saklanması gereklidir. İç şemada bir değişiklik yapıldığında, iç şema - kavramsal şema eşlemesinde gerekli uyarılar yapılarak değişikliğin kavramsal ve dış şemaları, dolayısıyla da kullanıcıları etkilemesi önlenerek fiziksel veri bağımsızlığı sağlanmış olur. Kavramsal şemada bir değişiklik yapıldığında ise, bir yandan kavramsal şema - iç şema eşlemeleri, diğer taraftan da bazı dış şemalar ile kavramsal şema arasındaki eşlemeler uyarlanır. Böylece kavramsal şemadaki değişiklikten iç şema (fiziksel veri bağımsızlığı) ve dış şemaların en azından bir kesimi (mantıksal veri bağımsızlığı) etkilenmemiş olur.

8.2.1 Üç Düzeyli Veri Tabanı Mimarisi

Veri tabanı mimarisi üç seviyeden oluşur. Üç düzeyli mimari ilk defa 1975 yılında “Bilgisayarlar ve Bilgi İşlem konusundaki Amerikan Ulusal Standartlar Komitesinin (ANSI/SPARC) VTYS Çalışma Takımı” tarafından önerilmiş ve zamanla benimsenmiştir. Veri bağımsızlığının sağlanması için VTYS mimarisinin mutlaka üç düzeyli mimaride olması gereklidir.



Şekil 6 Üç Düzeyli Veri Tabanı Mimarisi

8.2.1.1 Dış (Görünüm) Seviye

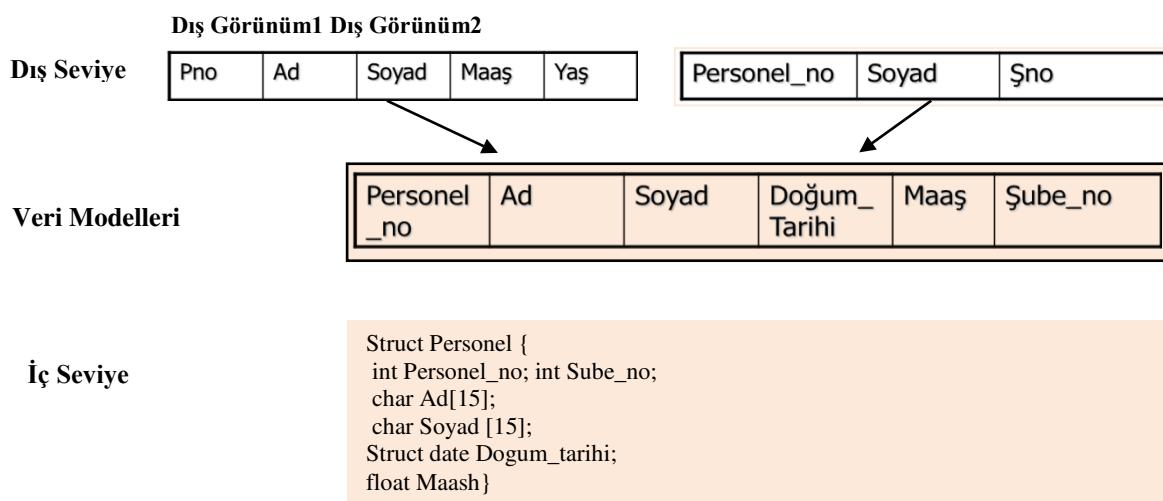
Veri tabanının sadece bir kısmı olup, kullanıcı seviyesidir. Kullanıcının bakışıdır. Her kullanıcı veri tabanıyla kendi dilinde çalışmak ister. Kullanıcı nasıl görmek istiyorsa bu seviye o şekilde yansıtılır. Her görünüm seviyesi kavramsal seviyenin bir alt kümesidir. Bu yüzden bir tek kavramsal seviye olmasına rağmen birçok dış seviye vardır.

8.2.1.2 Mantıksal Seviye

Bu seviyede verinin, veri tabanı içerisinde nasıl saklandığı ve veri ilişkileri tanıtılmaktadır. Bu bilgiler kullanıcıya gerekli olmayıp sadece veri tabanı yöneticisini ilgilendirmektedir. Bu seviye, veri tabanının tüm bilgilerini kapsar ve onları soyut bir formda yansıtır. Bu seviye diğer iki seviyeden farklıdır. Genelde kavramsal seviye verileri gerçek anlamda yansıtır.

8.2.1.3 İç (Fiziksel) Seviye

Burada karmaşık veri yapıları ayrıntılı olarak açıklanmaktadır. Verilerin fiziki ortamda saklanmasıdır. Verileri makinenin imkanları dahilinde yansıtırlar. Fiziksel bellek cihazlarına en yakın seviyedir. Kavramsal düzeydeki verilerin disk dosyalarında saklandığı yerdir.



Sekil 7 Seviyelere Örnekler

8.3 Katmanlı Yazılım Mimarileri

Yazılım dünyasında ihtiyaçlar ortaya çıktıktan sonra cevap verecek çeşitli yazılım mimarileri ortaya çıkmıştır. Bu çerçevede Tek-Katmanlı, İki-Katmanlı ve Çok-Katmanlı Veritabanlarından söz edilebilir. Katman (tier) kavramı ilk kez 80'lerin başlarında mini bilgisayar üreticileri tarafından kullanılmıştır. Bu üreticiler akılsız terminalleri Katman-1, mini bilgisayarları Katman-2 ve ana bilgisayarları da Katman-3 olarak sınıflamışlardır. Günümüzde katman tanımı daha farklı bir anlam taşımakta olup, uygulama yazılımının istemci-sunucu (client-server) arasındaki mantıksal paylaşımını ve yük dağılımını belirtmek üzere kullanılmaktadır. Buna göre:

8.3.1 Tek-Katmanlı Veritabanı Mimarisi

Tek Katmanlı (1-Tier) mimaride veritabanı ve uygulama aynı bilgisayar üzerinde bulunur. Dezavantajları -Bu mimari tek kullanıcılı yazılımlar içindir. Merkezi mimari yaklaşımında uygulama yazılımının dağıtılması ya da iş yükünün paylaştırılması gibi kavramlar

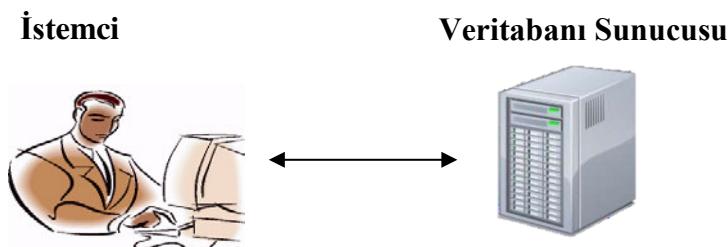
yoktur. Uygulama yazılımının üç bileşeni olan kullanıcı arayüzü, uygulama yordamları ve veriler aynı bilgisayar üzerinde yer alır. Tek-Katmanlı veritabanı, kayıtlar üzerinde herhangi bir değişikliğin anında meydana geldiği veritabanıdır. Bu veritabanını kullanan uygulamalar veritabanına direkt bağlantı sağlayabilmektedir. Çünkü program ve veritabanı dosyaları aynı bilgisayar üzerinde bulunmaktadır. Yerel (merkezi) veritabanları Tek-Katmanlı veritabanı olarak da geçmektedir. iki-Katmanlı veritabanında ise istemci uygulama veritabanı sürücülerini kullanarak veritabanı sunucusuna erişim sağlar. Çok-Katmanlı veritabanı modelinde client uygulama, veritabanı için bir veya daha fazla uygulama sunucusu ile bağlantı kurar.

8.3.2 İki katmanlı (2-Tier) Veritabanı Mimarisi

Düzen adı *Client/Server* (İstemci /Sunucu) olan bu mimaride veritabanı ayrı bir bilgisayar üzerinde yer alır ve uygulamalar ağ üzerinden bu veritabanı ile iletişim kurarlar.

2-Katmanlı Mimaride: Uygulama yazılımları her istemciye ayrı ayrı yüklenmekte olup, güncelleme gerektiğinde her istemciye yazılımın yeniden yüklenmesi gerekmektedir. İstemciler veri tabanına doğrudan erişmektedir. Her erişimde ayrı bir iletişim kanalı açılmakta, bu nedenle aynı anda desteklenebilecek kullanıcı sayısı, sunucuda kullanılan işletim sisteminin ve sunucu donanımının nitelikleri ile sınırlıdır, işletim sistemi kaynaklarının ve iletişim kanallarının etkin paylaşımı söz konusu değildir.

Avantajları; Aynı veriler üzerinde birçok kişi çalışabilir.



Şekil 8. 2-Katmanlı İstemci-Sunucu Mimarisi

2-Katmanlı mimari, iş yükünü ve uygulama yazılımını ikiye böler. Uygulama yazılımının kullanıcı arayüzü ve uygulama yordamları istemci adı verilen bilgisayarda yer alırken, veriler sunucu olarak adlandırılan ve görece daha güçlü bir bilgisayarda tutulur. Günümüzde dünyada kullanılan istemci-sunucu mimarisinde geliştirilmiş uygulama yazılımlarının çoğu 2 Katmanlı mimari kapsamında yer alır.

3-Katmanlı mimari ise iş yükünü ve uygulama yazılımını üçe böler. Uygulama yazılımının kullanıcı arayüzü istemcide, uygulama yordamları uygulama sunucusunda (application server) ve veriler veritabanı sunucusunda (database server) yer alır. Bir başka deyişle, 2-Katmanlı mimariden farklı olarak istemcide yer alan uygulama yordamları ayrı bir sunucuya taşınmıştır.

İki-Katmanlı Modelin Zayıf Yönleri: İş mantığı uygulamada yer aldığı için en ufak bir değişiklikte bütün istemcilerin güncellenmesi gereklidir. Her sunucu veri tabanına fiziksel bir

bağlantı kurduğu için kullanıcı sayısı arttıkça performans düşer. Veritabanı dışarıya açık olduğu için daha fazla risk söz konusudur. Uygulamanın çalışabilmesi için istemcilerin yapılandırılması gereklidir (veritabanı sürücülerinin kurulması vb.). Kullanıcı arayüzünün uygulama yordamlarından ayrı tutulmasının önemini anlaşılabilmek için 2-Katmanlı modelin zayıf noktalarının özetlenmesi gereklidir:

2-Katmanlı mimaride uygulama yazılımının her istemciye ayrı ayrı yüklenmesi gereklidir. Bunun çeşitli sakıncaları vardır. Öncelikle uygulama yazılımı değişikçe istemcilere yeniden yükleme yapılması gereklidir. Kullanıcı sayısının yüksek olduğu durumlarda ya tüm istemcilere yeniden yükleme yapılana kadar uygulamanın durdurulması ya da uygulama yazılımının a da uygulama yazılımının eski ve yeni sürümlerinin aynı anda kullanımın sonuçlarına katlanması gereklidir. Günümüzde bunların her ikisi de geçerli olamaz. 2-Katmanlı mimaride istemciye yüklenen uygulama yazılımı genellikle sunucu üzerindeki Veri tabanına doğrudan erişen görsel yazılım araçlarıyla gerçekleştirilmiş yazılımlardır. Sonuç olarak, her istemci, sunucu ile arasında kendisine özel bir iletişim kanalı açar. İşletim sistemi kaynaklarını hızla tüketen bu yaklaşım, kullanıcı sayısı arttığında sunucunun performansının hızla düşmesine neden olur. Kimi yazılım geliştirme araçları ise bağlandıkları Veri tabanına özel olarak geliştirilmiş ürünler olup, değişik veritabanlarına farklı yöntemler ve özel geçitler (gateway) aracılığıyla bağlanabilirler. Bunlar yazılımın karmaşıklığının çok fazla artmasına neden olur. 2-Katmanlı mimaride istemci üzerine yüklenen uygulama yazılımında sunucudaki Veri tabanına ilişkin tablo adları, veri alanı isimleri, veritabanı yordamlarının adları ve parametreleri gibi pek çok bilgi yer almaktadır. Veri tabanına erişim ise yalnızca kullanıcı kodları ve şifreler aracılığı ile denetlenir. Geçmişte, özellikle genel kullanıma açılan uygulamalarda, uygulama yazılımının, hizmeti sağlayan kurumun sınırları dışına çıkmasından ve kullanıcı kodu ve şifre denetiminden sonra Veri tabanına doğrudan erişime izin verilmesinden kaynaklanan pek çok güvenlik sorunu yaşanmıştır.

8.3.3 Üç-Katmanlı Veritabanı Mimarisi

Bu yapı en az 3 bileşenden oluşur: istemci(Client), Uygulama sunucusu(Application Server) ve Veri tabanıdır. Sırasıyla sunum katmanı, iş katmanı ve veri katmanı da denir. Çok-Katmanlı veritabanları 3 Katmanlı veritabanları olarak tanımlanmaktadır. Çünkü bu modelde Çok-Katmanlı uygulama aşağıdaki gibi 3 bölüme ayrılmıştır:

Client Application (istemci Uygulama): Kullanıcının kendi bilgisayarında bir kullanıcı ara birimi sunar.

Application Server (Uygulama Sunucusu): Bir ağ ortamında bütün istemcilerin erişebileceği merkezi bir konumda bulunur ve istemcilerin ihtiyaçlarını karşılamak üzere ortak veri servisleri sunar.

Remote Database Server (Uzaktan Erişimli Veritabanı Sunucusu): ilişkisel Veritabanı Yönetim Sistemi sağlar. Bu modelde istemci uygulama veritabanına doğrudan değil de aradaki katmanda bulunan uygulama sunucusu üzerinden bağlanır.

3-Katmanlı mimaride, uygulama yazılımları yalnızca uygulama sunucusuna yüklenmiştir. Uygulama yazılımı güncellendiğinde, her istemci aynı anda yeni sürümü kullanıyor olacaktır. Sonuç olarak yazılımın yeni sürümünün yayılması sürecinin güçlüğünde kullanıcı sayısı ve konumu belirleyici unsur olmaktan çıkacaktır. Uygulama sunucusundaki uygulamalar Veri tabanına doğrudan erişirler. Her iletişimde ayrı bir iletişim kanalı açılır. Bu bakımdan veritabanı sunucusu açısından 2-Katmanlı mimariye göre toplam iletişim kanalı sayısında bir farklılık olmayacağıdır. 3-Katmanlı mimari, bu haliyle 2-Katmanlı mimarinin uygulama yazılımının yaygınlaştırılması ve bakımıyla ilgili sorunlara çözüm getirmektedir. Ancak, kaynak paylaşımına yönelik herhangi bir destek vermemektedir. Uygulamaların birim işlem (transaction) gereksinimi de tipki 2-Katmanlı mimaride olduğu gibi büyük ölçüde veritabanı yönetim sisteminin sağladığı olanaklar kullanılarak karşılanır.

En basit açıklama ile 3-Katmanlı mimaride istemci ve veritabanı sunucusu arasında bir ara katmanın yerleştirildiği söyleyebilir. Bu ara katmanın amacı, istemci adına veritabanı bağlantılarını kurmak ve izlemek ve veritabanı sunucusundan gelen sonuçları istemciye yansıtırken istemci adına veritabanı üzerinde işlem yapmaktadır. Buna göre 3-Katmanlı mimarinin üstünlükleri aşağıda sıralanmıştır: Internet ortamında yaygın olarak kullanılan web tarayıcıları (Netscape ve IE) evrensel istemciler olarak değerlendirilebilir, istemcilerde yalnızca web tarayıcıları kullanıldığında, 3-Katmanlı mimari aynı zamanda bir web tabanlı nitelik kazanır. Bu durumda istemci bilgisayarların bakımı, uygulama yazılımını kullanıma açan grubun sorumluluğu olmaktan çıkar. İstemci, daima uygulama sunucusunda yüklü olan yazılımı kullandığından, yazılımın yeni sürümünün yayılması, yalnızca uygulama sunucusuna yazılımın yüklenmesinden ibarettir. Kullanıcı sayısının çokluğu ve yayıldığı coğrafya bu işlemde belireyici faktör olmaktan çıkar. İstemcinin Veri tabanına doğrudan erişimi sözkonusu olmadığından ekgüvenlik önlemleri kolaylıkla alınabilir.

3-Katmanlı mimaride, uygulama katmanı diğer iki katmandan tamamen bağımsızdır. Başlangıçta veritabanı sunucusuyla aynı bilgisayara yüklenebilir ve uygulamanın boyutları arttıkça ayrı bir sunucuya taşınabilir. Uygulama katmanı, başlangıçta tek bir bilgisayarda tutulan veri tabanının birden fazla bilgisayara yayılacak biçimde genişlemesine izin verecek ve farklı bilgisayarlarda geniş bir coğrafyaya yayılmış değişik veritabanları arasında birim işlem eşgüdümü yapacak biçimde tasarlanabilir. Bu amaçla genellikle uygulama sunucusunda bir TPM (Transaction Processing Monitör) ya da OTM (Object Transaction Monitör) kullanılır. TPM her İstemci için veritabanı sunucusuna ayrı bir iletişim kanalı açmaz. Bunun yerine hazır tuttuğu sınırlı sayıda iletişim kanalını, istemcilerden gelen istekleri karşılamakta paylaşımı olarak kullanarak veritabanı sunucusunun performansının düşmesini önler. TPM ayrıca, veritabanı sunucuları arasında yük dağılımı, yüksek öncelikli işlerin öncelikle ele alınması, kilitlenme denetimi, hata denetimi, ek güvenlik önlemleri ve sistem yönetimi gibi konularda gelişmiş yetenekler sunar.

Sonuç olarak 2-Katmanlı mimaride ölçeklenebilirlik (scalability) sorunu olduğu ve istemcilerin bakımının zahmetli bir süreç olduğu açıklıdır. Pazar baskısı nedeniyle eldeki 1-Katmanlı mimariye sahip yazılımların olduğu gibi web uygulamasına dönüştürülerek internet ortamına açılmasının ise bu sorunları çözmediği artık bilinen bir gerçekdir.

3-Katmanlı mimariye geçişten beklenen yararların sağlanabilmesi için, bu mimarinin gereği olan kurallar ve araçlar kullanılmalıdır. Bu nedenle özellikle büyük uygulamalarda, uygulama yordamı katmanı ayrı bileşenler biçiminde geliştirilmekte ve bir TPM ile birlikte kullanılmaktadır. Bileşen tabanlı modüler gerçekleştirim yaklaşımı, değişik programlama dillerinde ve değişik yazılım geliştirme takımları tarafından geliştirilen modüllerin birarada kullanımını olanaklı kılarken, TPM farklı veritabanlarının birarada kullanılmasını gerektiren veritabanı işlemlerinin eşgündümü, birden fazla sunucu arasında yük dağıtım ve hata denetimi gibi temel işlevleri yüklenir.

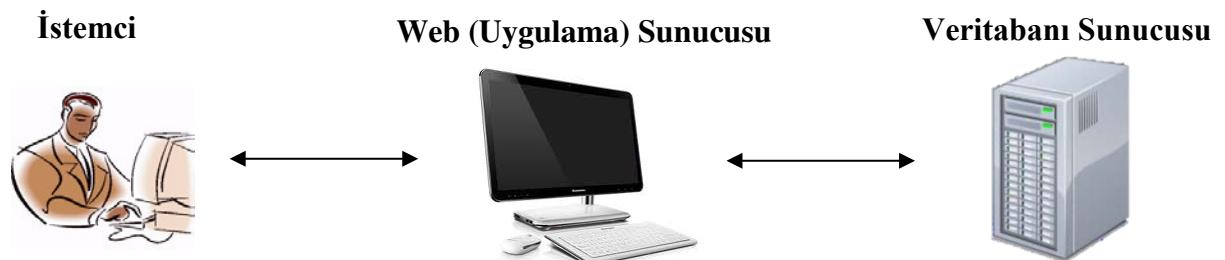
3-Katmanlı yazılım mimarisinin yazılımın yayılması, bakımı (maintenance) geliştirilebilirliği ve ölçeklenebilirliği açısından sunduğu üstünlükler vardır. Ancak bu üstünlüklerin bedeli vardır. Yazılım geliştirme ve proje yönetim sürecinde gereken uzmanlık düzeyi daha yüksektir ve nesneye yönelik programlama konusunda uzman kişilerle çalışılmasını gerektirir. Yazılım geliştirme süresi ve bedeli de 2-Katmanlı mimariye göre %50-80 daha fazladır. Buna karşılık aynı kurum içinde uzun yıllar değişmeden kullanılacak ve içerik açısından çok sayıda işlev içeren donuk sistemler hariç, sürekli gelişmesi beklenen, geniş bir coğrafyaya yayılmış çok sayıda kullanıcısı olan günümüzün dinamik uygulamalarının geliştirilmesinde 3-Katmanlı ve web tabanlı mimariler vazgeçilmez hale gelmiştir. Yazılım üreticisi büyük firmalar yukarıda belirtilen sorunlara çözüm getirmek amacıyla orta-katman yazılımı (middleware) olarak isimlendirilen yeni ürünler geliştirmiştir. Özellikle geçmiş ana bilgisayarlarda kullanılan birim işlem (transaction) yönetim sistemlerine dayanan birim işlem eşgündüm yazılımları (TMP -Transaction Processing Monitör) birbiri ardından piyasaya sürülmüştür. TMP yazılımları birim işlem eşgündümünün dışında sistem performansını optimize etme, yük dağılımı, görev yönetimi, iletişim kaynaklarının eşgündümü, birim işlem geri alma ve onarma, sistem izleme ve yönetimi, görevlerin önceliklere göre işletimi gibi pek çok yeteneği sunmaktadır.

Avantajları

- İstemci tarafında herhangi bir yapılandırmaya ihtiyaç olmaz (genelde).
- İş mantığı orta katmanda (iş katmanı) yer aldığı için bu katmanla ilgili değişikliklerde istemcileri güncellemek gerekmez.
- İstemciler veritabanı yerine orta katmana bağlılığı için hem veritabanı performansında fazla bir azalma olmaz hem de veritabanının güvenliği üst seviyeye çıkmış olur.
- İstemciler veri tabanından bağımsız hale gelirler. Veritabanı değiştiği zaman orta katmanı ilgili veritabanı için yeniden uyarlamak yeterlidir.

Dezavantajları

- Bu tur mimariye sahip bir yazılımı yazmak diğerlerine göre daha zor ve karışiktır.



Şekil 9. Üç Katmanlı İstemci-Sunucu Mimarisi

Uygulama Soruları

AUZEF'in kullandığı veritabanı mimarisinin yapısı nasıldır? Tartışınız.

Bu Bölümde Ne Öğrendik Özeti

Bu hafta, veri tabanları, arayüz araçları ve iletişim gibi bileşenlerin şekillendirilme biçimini olan Sistem mimarisi konusu anlatıldı. Veritabanı mimarisi ile ilgili gelişmeler değerlendirildi. İstemci/sunucu mimari yapı şekillerle açıklandı. Veri tabanlarının web ile birleştirilmesi, dağıtık veritabanları, standart veri tabanı mimarisi ve katmanlı veritabanı mimarileri yapılar karşılaştırmalı olarak anlatıldı.

Bölüm Soruları

1.Program ve veritabanı dosyaları aynı bilgisayar üzerinde bulunan Veritabanı mimarisi aşağıdakilerden hangisidir?

- a) Tek-Katmanlı Veritabanı,
- b) Çok-Katmanlı Veritabanları
- c) İki-Katmanlı Veritabanı
- d) Üç-Katmanlı Veritabanı
- e) İstemci/Sunucu Veritabanı

2.İki katmanlı (2-Tier) yazılımlarla ilgili hangisi yanlış bir ifadedir?

- a) Diğer adı Client/Server (İstemci /Sunucu olan mimaridir.
- b) Veritabanı ayrı bir bilgisayar üzerinde yer alır ve uygulamalar ağ üzerinden bu veritabanı ile iletişim kurarlar.
- c) Uygulama yazılımları her istemciye ayrı ayrı yüklenmekte olup, güncelleme gereğinde her istemciye yazılımın yeniden yüklenmesi gerekmektedir.
- d) İstemciler veri tabanına doğrudan erişmektedir.
- e) Uygulama yazılımının kullanıcı arayüzü istemcide, uygulama yordamları uygulama sunucusunda ve veriler veritabanı sunucusunda yer alır.

3.Kavramsal düzeydeki verilerin disk dosyalarında saklandığı seviye hangisidir?

- a) İç (Fiziksel) Seviye
- b) Mantıksal Seviye
- c) Dış (Görünüm) Seviye
- d) Hiçbiri
- e) Hepsi

CEVAPLAR 1-A, 2-E, 3-A

9. VERİ TABANI YAZILIMI

Bu Bölümde Neler Öğreneceğiz?

9.1 WAMPERVER Kurulumu

9.1.1 WAMPERVER Kurulum Adımları

9.1.2 Sunucuların (Servislerin) Çalıştırılması

9.1.3 WAMPERVER Kullanımı

9.2 Mysql Workbench Kurulumu

9.2.1 Kurulan MySQL Workbench programının çalıştırılması

9.2.2 MySQL Workbench Kullanımı

9.2.3 MySQL Workbench Kullanarak Tablo ve İlişki Oluşturma

9.2.4 MySQL Workbench Kullanarak Veri Aktarımı

9.3 Wampserver Sunucusundaki Bir Veritabanını Modeli Olarak Model Ekranına Aktarma

9.4 MySQL Workbench Kullanarak Veri Aktarımı

9.4.1 Hazırlanan Modeli Veritabanı Olarak Wampserver Sunucusuna Aktarma

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Veri tabanı yazılımı nedir?
- 2)** WAMPSERVER ne işe yarar?
- 3)** MySQL Workbench nasıl kurulur?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Veri tabanı yazılımlarını bilir.	Anlatım, Soru-Cevap, Tartışma
	MySQL Workbench kurabilir.	Alıştırma ve Uygulama, Örnek Olay
	MySQL Workbench Kullanabilir.	Bireysel Çalışma, Problem Çözme,
		Proje Temelli Öğrenme

Anahtar Kavramlar

- Veri tabanı yazılımları
- MySQL Workbench
- Wampserver

Giriş

Veri tabanı yazılımı: veri tabanının bilgiyi verimli bir Şekilde düzenleyebilmesini, gerektiği zaman bilgiye ulaşılabilmesini sağlayan, birden çok kullanıcıya bilgiye aynı anda erişme olanağı tanıyan, verilerin düzenli bir Şekilde saklanması imkan sağlayan yazılımlardır.

Bilgilerin bilgisayarlarda daha rahat saklanması ile —veri tabanı yazılımlarına olan ilgi daha çok artırmıştır. Veri tabanı yazılımları, veri tabanının bilgiyi verimli bir Şekilde düzenleyebilmesini, gerektiği zaman bilgiye ulaşılabilmesini sağlayan birden çok kullanıcıya bilgiye aynı anda erişme olanağı tanıyan verilerin düzenli bir Şekilde saklanması imkan sağlayan yazılımlardır.

Basit bir uygulamadan tutun da çok büyük kuruluşların verilerine kadar, ister bir web uygulaması olsun ister bir masaüstü uygulaması olsun, günümüzde bu gibi birçok alanda veri tabanı uygulamalarına ihtiyaç duyulmaktadır. Son yıllarda yapılan birçok proje çok sayıda bilgisayar tarafından kullanılabilecek şekilde tasarılmaktadır. Bu yüzden, ağ ortamında birden fazla kullanıcı aynı proje üzerinde çalışabilmektedir. Bu bölümde bir uygulama için gerekli olan ve birden çok bilgisayarın erişebileceği veri tabanı yönetim sistemlerinin temel bilgileri ve kurulumları hakkında bilgi verilecektir.

Özellikle MySQL ve Ms SQL ile çeşitli uygulamalar geliştirilecektir.

9. VERİ TABANI YAZILIMI

Veri tabanı kavramı bilgisayarın kullanılmaya başlanmasından yıllar sonra ortaya çıksa da günümüzde neredeyse tüm uygulamalarda veri tabanına ihtiyaç duyulmaktadır. Basit bir web uygulamasından çok büyük ölçekli kuruluşların ağır verilerine kadar birçok alanda veri tabanına ve bu verilerin yönetimine ihtiyaç duyulmaktadır.

Veri tabanı, verilerin düzenli bir şekilde saklanmış halidir. Veri tabanı yönetim sistemleri ise bu verilerin fiziksel hafızadaki durumları, birbirleri ile olan ilişkileri, kullanıcıların bu verilere erişim yetkileri gibi birçok detayın yönetildiği yazılımlardır.

Veri tabanı yönetim sistemi (Database Management System, kısaca DBMS), veri tabanlarını oluşturmak, kullanmak ve değiştirmek, veri tabanı üzerinde kullanıcılar tanımlamak, bu kullanıcılara yetkiler atamak ve veri tabanı sistemleri ile ilgili her türlü işletimsel gereksinimleri karşılamak için tasarlanmış sistem ve yazılımdır.

Veri tabanı yönetim sistemlerinin avantajlarını maddeler halinde sıralayacak olursak:

Veri Takrarını Önlemek: Aynı veri farklı kullanıcıların bilgisayarlarında tekrar tekrar tutulmaz. Böylelikle veri tekrarı engellenmiş olur.

Veri Tutarlılığı: Aynı verinin değişik kullanıcılarda birkaç kopyasının bulunması (bir yerde değiştirilen verinin diğer yerde aynı kalması durumu) veri tutarsızlığı oluşumuna neden olur. Bunu engellemek için kullanılır.

Veri Paylaşımı / Eş zamanlılık: Veri tabanı yönetim sistemi (VTYS) kullanılmadığı durumlarda veriye sıralı erişim yapılır. Yani birden çok kullanıcı aynı anda aynı veriye erişemez. Bir VTYS'de ise aynı veri tabanlarına saniyede yüzlerce, binlerce erişim yapılabilir.

Veri Bütünlüğü: Bir tabloda değişiklik yapılan verinin ilişkili olduğu diğer tablo veya tablolarda da aynı işlemin yapılması gerekebilir.

Veri Güvenliği: Verinin isteyerek ya da yanlış kullanım sonucu bozulmasını önlemek için çok sıkı mekanizmalar mevcuttur. Veri tabanına girmek için kullanıcı adı ve şifreyle korumanın yanı sıra kişiler sadece kendilerini ilgilendiren tabloları ya da tablo içinde belirli kolonları görebilirler.

Veri Bağımsızlığı: Programcı, kullandığı verilerin yapısı ve organizasyonu ile ilgilenmek durumunda değildir. Veri bağımsızlığı, VTYS'lerin en temel amaçlarındandır.

MySQL Server

MySQL veri tabanı yönetim sistemi, yüksek performans, yüksek güvenilirlik ve kullanım kolaylığı nedeniyle dünyanın en popüler açık kaynak kodlu veri tabanı yönetim sistemi haline gelmiştir. MySQL 20'den fazla platform üzerinde çalışabilmektedir.

MSSQL Server

Doğrudan veri tabanı içinde yapılmış ve yarı yapılmış belgelerin yanında resim ve zengin medya gibi yapılanmamış belgelerden gelen verileri depolayan, verilerinizle sorgu, arama, senkronizasyon, raporlama ve analiz gibi daha fazla işlem gerçekleştirmenizi sağlayan, zengin bir entegre hizmeti sunan veri tabanı yönetim sistemidir.

Oracle Server

Gelişmiş bir ilişkisel veri tabanı yönetim sistemidir. Tüm ilişkisel veri tabanı sistemleri gibi büyük miktarda verinin çok kullanıcılı ortamda depolanmasını ve güvenli bir şekilde erişimini yönetir. Oracle veri tabanı yönetim sistemi özellikle kurumsal alanda kullanılan yaygın bir veri tabanı sistemidir. Oracle çok sayıda araçtan oluşur ve uygulama geliştiricilerinin kolay ve esnek uygulamalar geliştirmesini sağlar.

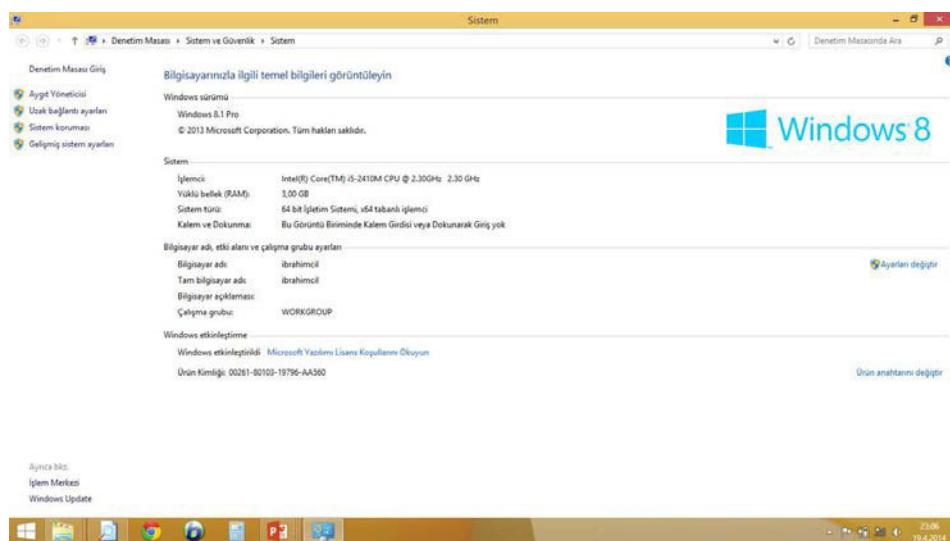
9.1 WAMPSERVER Kurulumu

Wampserver yazılımı ücretsiz olarak <http://www.wampserver.com/en/> adresinden indirebilirsiniz.



İşletim sisteminizin desteklediği veri yoluna(32 Bit-64 Bit) uygun olan linki tıklayınız. Bilgisayarınızın işletim türünü bilmiyorsanız, öğrenmek için şu yolu izleyiniz:

Bilgisayaram => Sağ Tık Özellikler => Sistem başlığı altında "SİSTEM TÜRÜ" (32 Bit yada 64 Bit İşletim Sistemi)



9.1.1 WAMP SERVER Kurulum Adımları

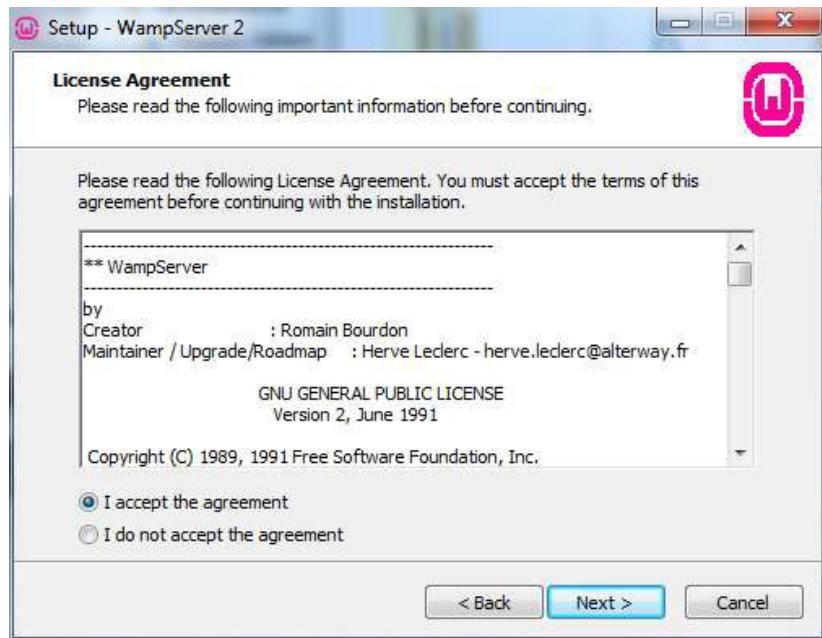


1.adım => Yukarıda görülen İndirdiğiniz Wamp Server iconuna "çift tıklıyoruz" ve "çalıştır" diyoruz.

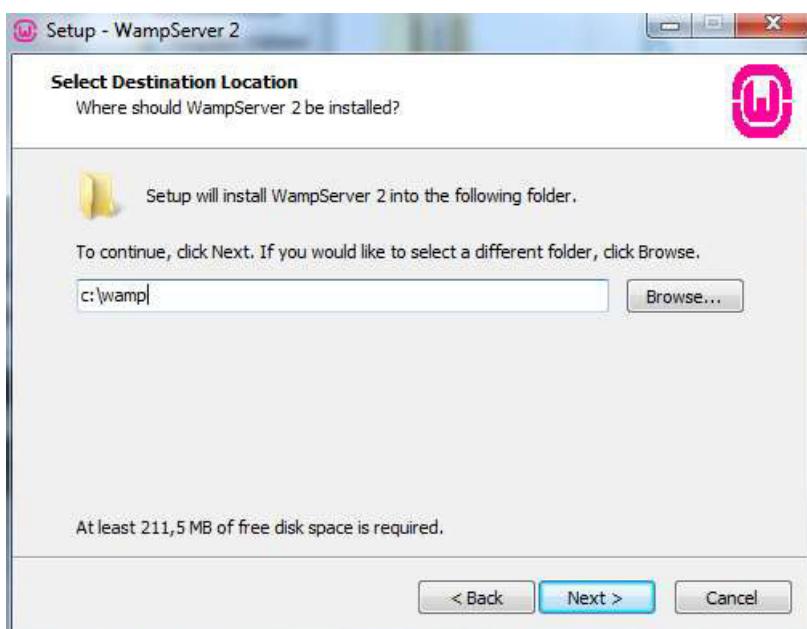


2.adım => Karşımıza çıkan iletişim ekranında "Next" butonuna tıklayarak geçiyoruz.

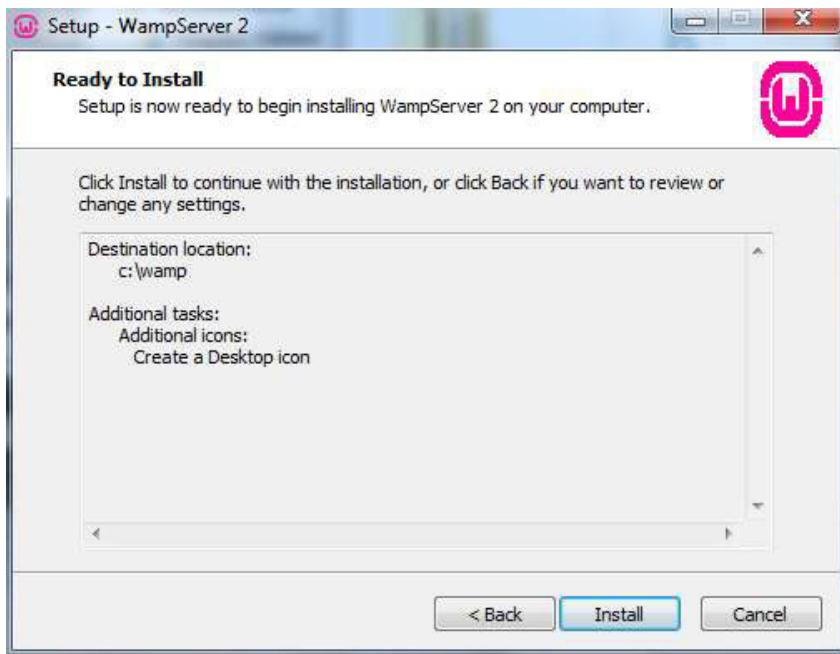
3.adım => Bu ekranda lisans koşullarını kabul etmemiz isteniyor. Üstteki kutucuğu işaretleyip "Next" butonu ile devam ediyoruz.



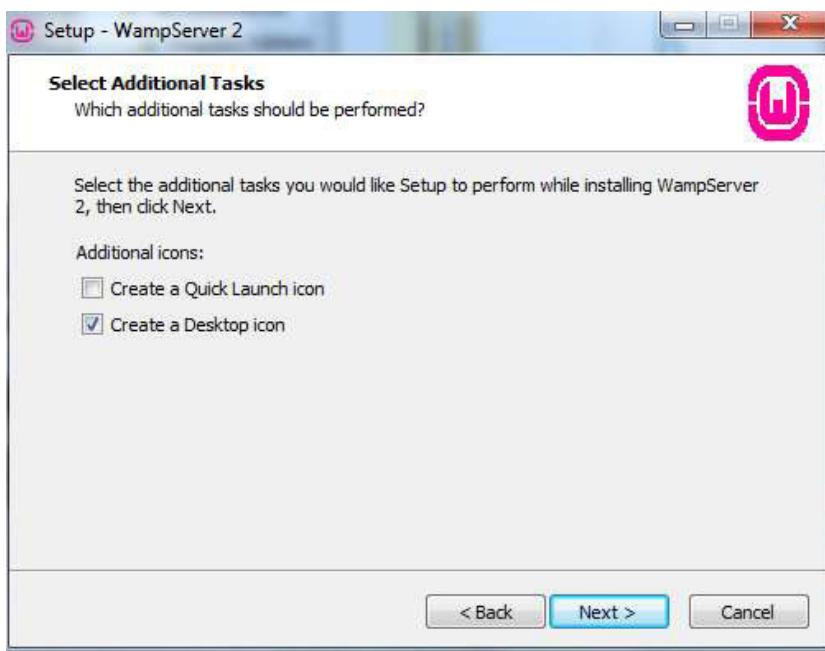
4.adım => Karşımıza çıkan bu pencere WampServer'ımızı hangi sürücüye kuracağımızı belirtmemizi istiyor. "C" sürücüsüne kuralım ve "Next" butonuna tıklayıp devam edelim.



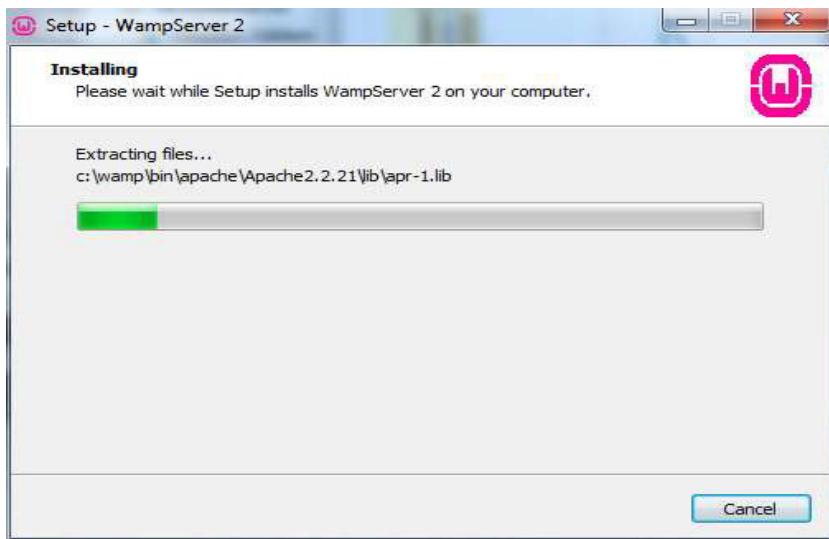
5.adım => WampServer iconunu, kurulum tamamlandıktan sonra kolay erişebilmemiz için masaüstüne alalım."Create a Desktop Icon" seçeneğinin yanındaki kutucuğu işaretleyerek "Next" butonuna tıklayalım.



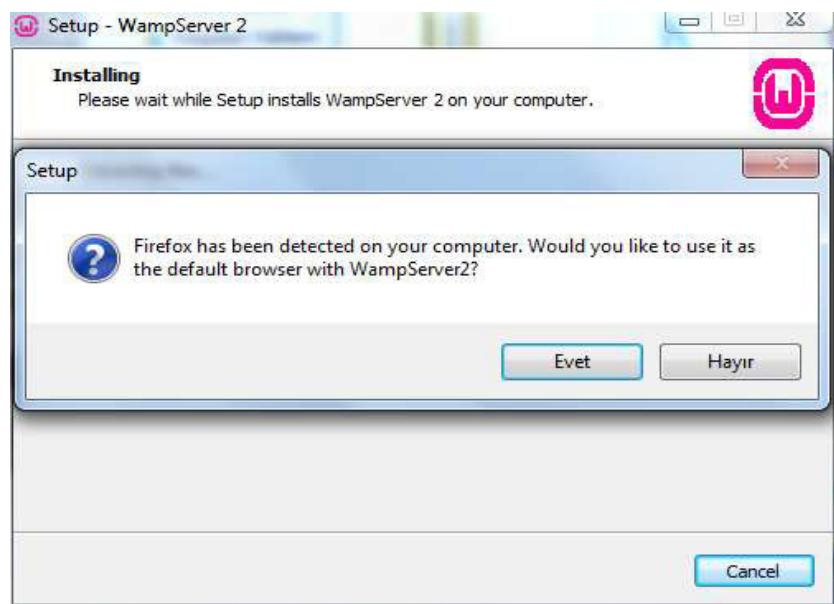
6.adım => Kurulumu başlamanız için "Install" butonuna tıklayabiliriz.



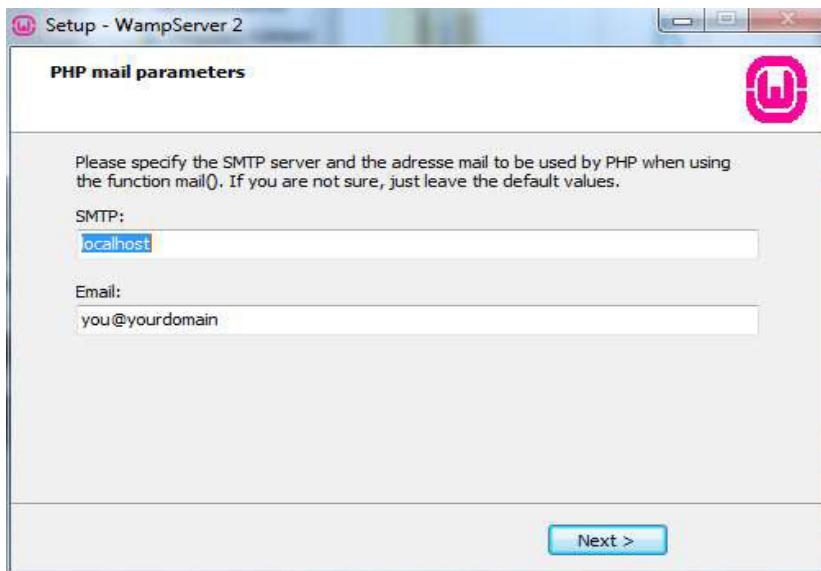
Yükleme ekranı görüntülenecektir. Yükleme işlemi tamamlanana kadar bekleyiniz.



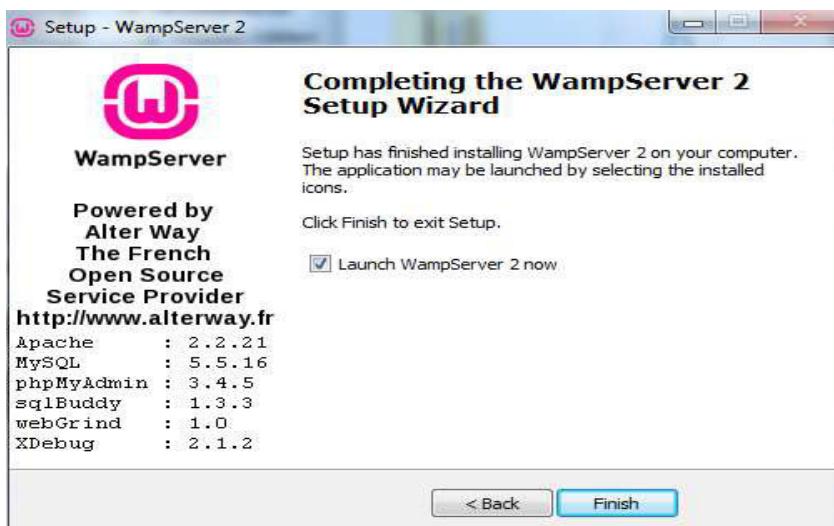
7.Adım=>Bu ekran WampServer'in, bilgisayarınızda kurulu olan tarayıcılarınızı(internet Explorer-Google Chrome-Mozilla Firefax v.s.) varsayılan olarak kullanmasını isteyip istemediğinizi soruyor. "Hayır" seçeneğini işaretleyip geçelim.



8. Adım => Bu ekran, PHP mail atmak ister ise hangi adres'e mail atacağını belirtmemizi istiyor. Kendinize ait bir mail adresi yazabilirsiniz. Üst kısım ise; kurulumu localde yapacağımız için "localhost" olarak kalmalıdır."Next" deyip geçiyoruz.



9. Adım => "Finish" butonuna tıklayarak kurulumu sonlandırıyoruz.



10.adım => Kurulum tamamlandıktan sonra masaüstümüzün sağ alt köşesine WampServer'ın iconu gelecektir.Kontrol edelim.

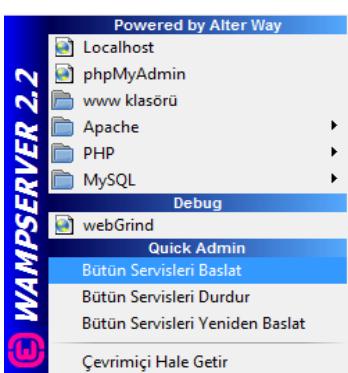


11.adım => Sağ alt köşedeki WampServer iconuna sağ tıklayalım. "Language" - "Turkish" seçimlerini yaparak WampServer'ımızı Türkçeştirelim.



9.1.2 Sunucuların (Servislerin) Çalıştırılması

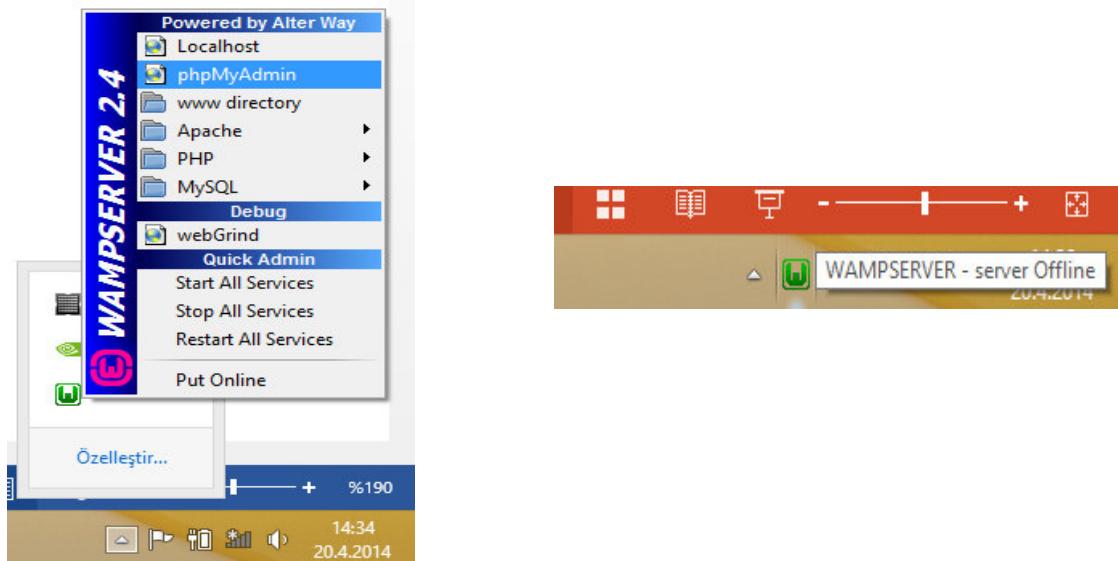
12.adım => Ve son olarak WampServer iconunun üzerinde bir kez sol tıklayarak açılan menüden "Bütün Servisleri Başlat" seçeneğine tıklayalım.



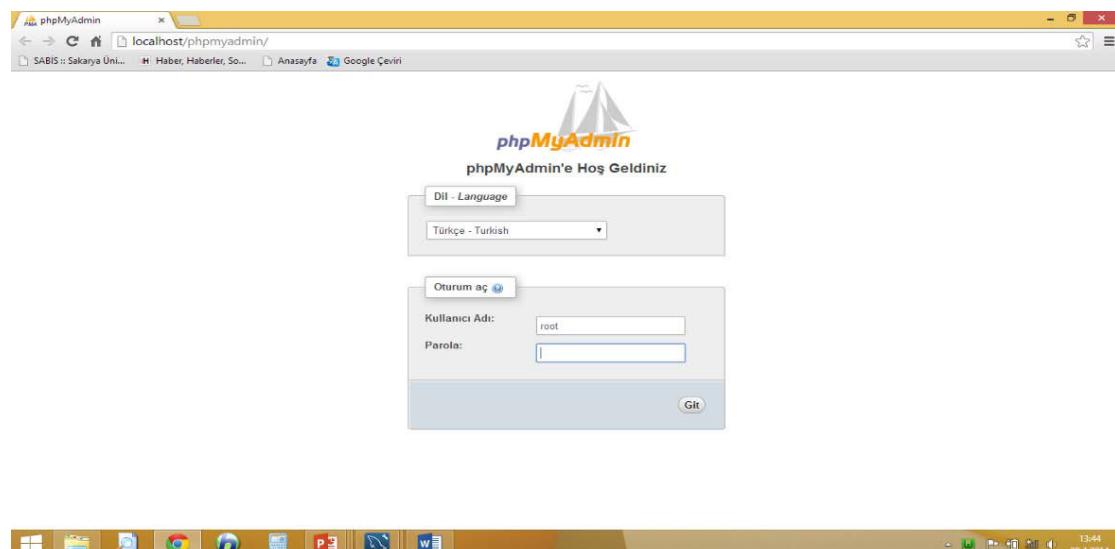
Not: Sunucular düzgün çalıştığında W simgesi yeşil gözükecektir.

9.1.3 WAMP SERVER Kullanımı

Wampserver tıklanarak açılan ekrandan phpMyAdmin sekmesi tıklanarak Localhost'ta çalışan veritabanı sunucuna erişilir.



phpMyAdmin sekmesi tıklandıktan sonra gelen ekranda kullanıcı adı yerine «root» girilir eğer şifre tanımlandıysa şifre girilir. Şifre yoksa boş geçilerek git düğmesi tıklanır.



Ana ekran bu şekildedir.

The screenshot shows the phpMyAdmin configuration interface. On the left, there's a sidebar with database names: abc, information_schema, mysql, performance_schema, t2db, and test. The main area has two tabs: 'Genel Ayarlar' (General Settings) and 'Veritabanı sunucusu' (Database Server). In 'Genel Ayarlar', the host is set to 'localhost' via TCP/IP, port is 3306, and the character set is utf8_general_ci. In 'Veritabanı sunucusu', it shows the MySQL Community Server (GPL) version 5.6.12-log. The bottom right corner displays the date and time as 13:49 20.4.2014.

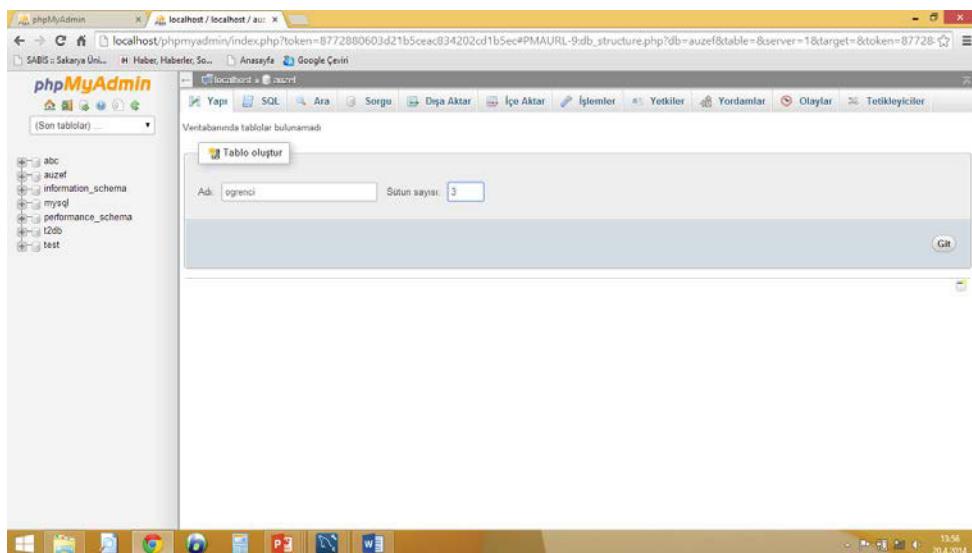
Veritabanı sekmesi tıklanarak yeni bir veri tabanı oluşturulur.

The screenshot shows the 'Veritabanları' (Databases) page in phpMyAdmin. A new database named 'auzel' is being created. The 'Veritabanı oluştur' (Create Database) form is filled with 'auzel' and 'Kullanıcı adı' (User Name) set to 'root'. Below the form, a table lists existing databases: abc, information_schema, mysql, performance_schema, t2db, and test. A note at the bottom states: 'Not: Buradaki veritabanı istatistiklerini etkinleştirmek web sunucusu ile MySQL sunucusu arasında yüksek trafik yol açabilir.' (Note: Enabling statistics for this database may cause high traffic between the web server and MySQL server).

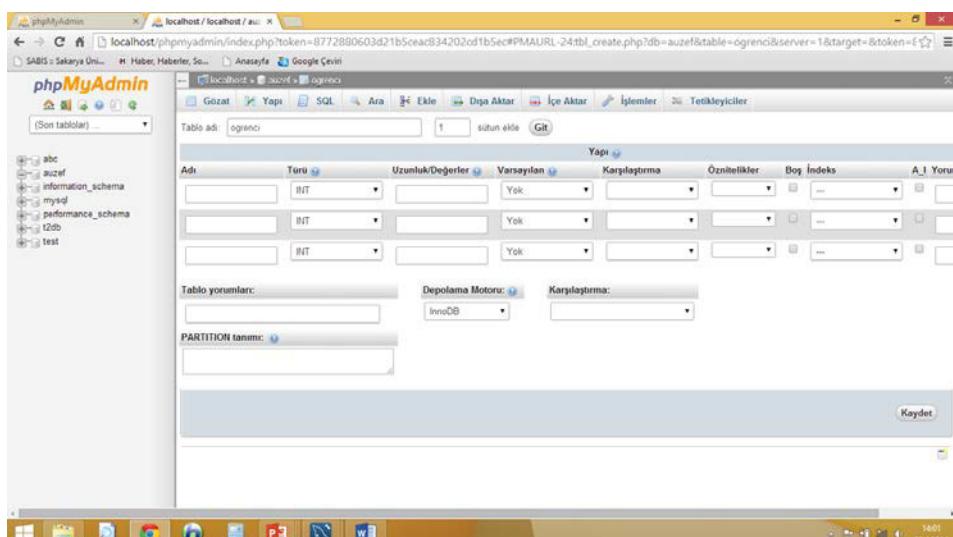
Bir veri tabanı oluşturduk.

The screenshot shows the same 'Veritabanları' (Databases) page as before, but now the 'auzel' database is listed among the others. A green success message box says 'Veritabanı auzel oluşturuldu.' (Database auzel created). The bottom note remains the same: 'Not: Buradaki veritabanı istatistiklerini etkinleştirmek web sunucusu ile MySQL sunucusu arasında yüksek trafik yol açabilir.' (Note: Enabling statistics for this database may cause high traffic between the web server and MySQL server).

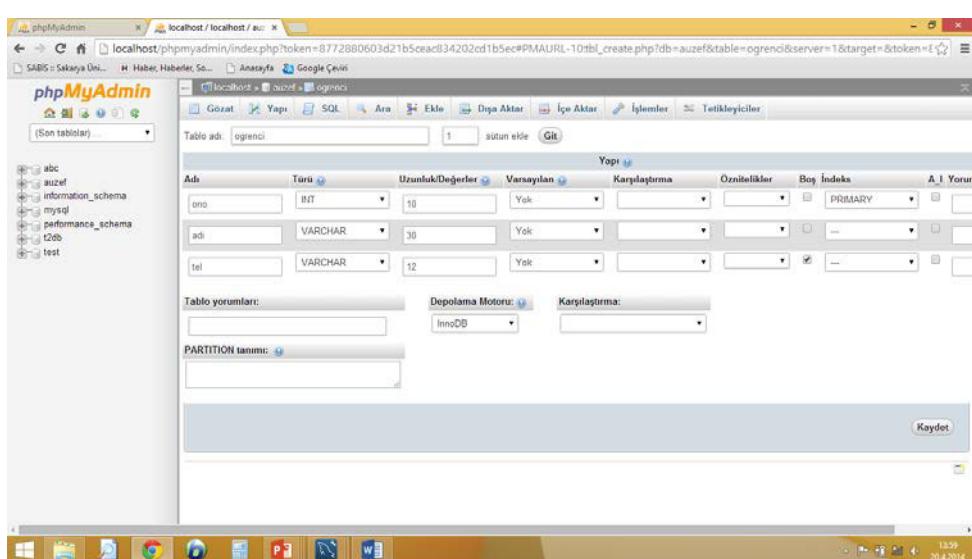
Veritabanında tablo oluşturulmaktadır.



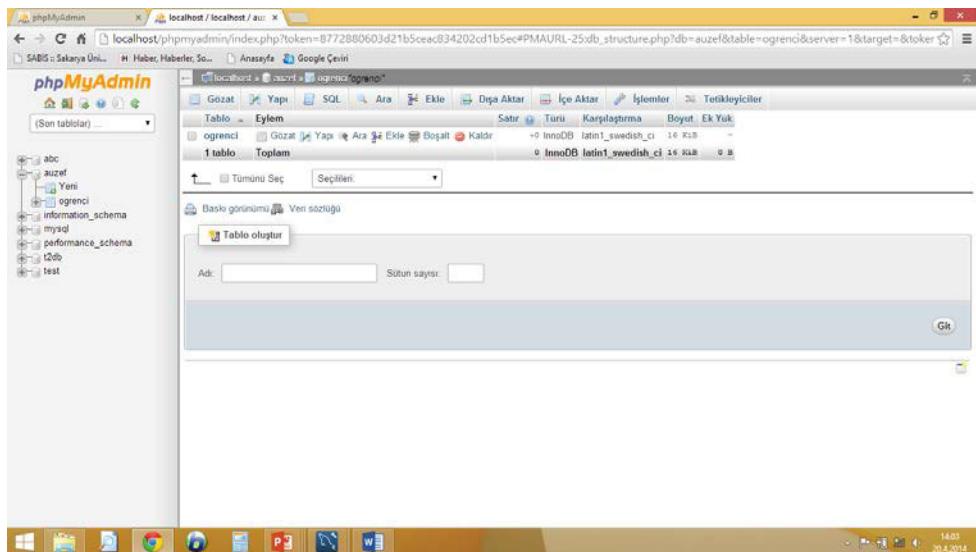
Tablonun alanları tasarılanır.



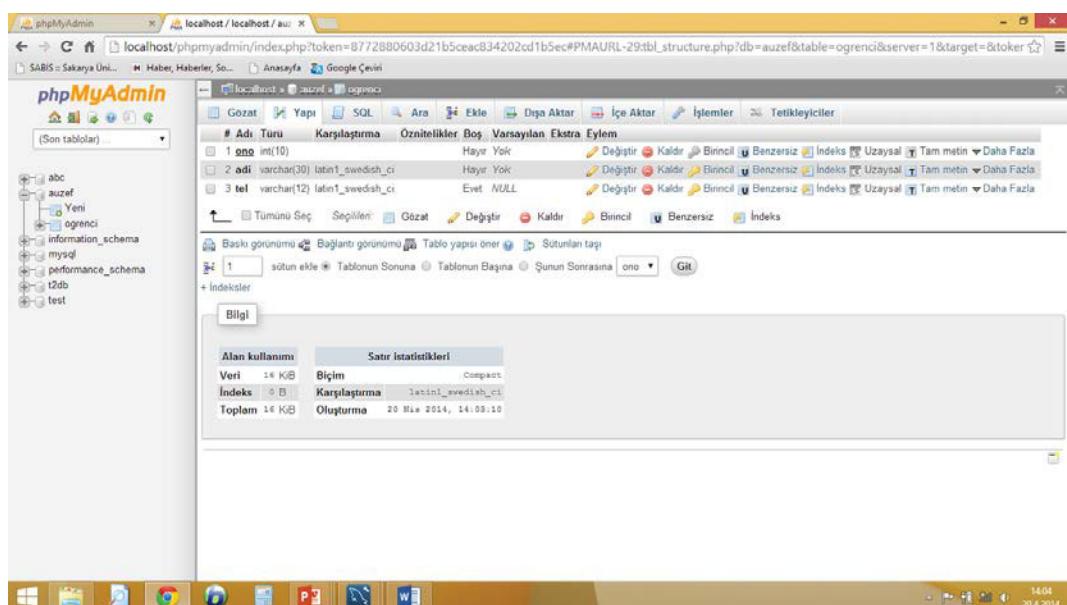
Tablonun alanları tasarılanır.



Oluşturulan tablo veritabanı içerisinde kontrol edilir.



Burada oluşturulan tablo üzerinde istenirse değişiklikler yapılabilir.



9.2 Mysql Workbench Kurulumu

MySQL Workbench programı ücretsiz olup aşağıdaki adresden temin edebilirsiniz.

<http://dev.mysql.com/downloads/tools/workbench/>

Yukarıdaki linke tıkladığınızda karşınıza aşağıdaki ekran gelecek ve buradan istediğiniz dosya sistemindeki yazılımı indirebilirsiniz.

MySQL Workbench 6.1.4

Select Platform: Microsoft Windows ▾

Looking for previous GA versions?

Windows (x86, 32-bit), MSI Installer	6.1.4	30.0M	Download
(mysql-workbench-community-6.1.4-win32.msi)			MDS: 2e585bd7e881d6e38ac4d9f8c51de7 Signature
Windows (x86, 32-bit), ZIP Archive	6.1.4	37.9M	Download
(mysql-workbench-community-6.1.4-win32-noinstall.zip)			MDS: 4eda950b6b904ba1e0025e6c4c0898fa Signature

! We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

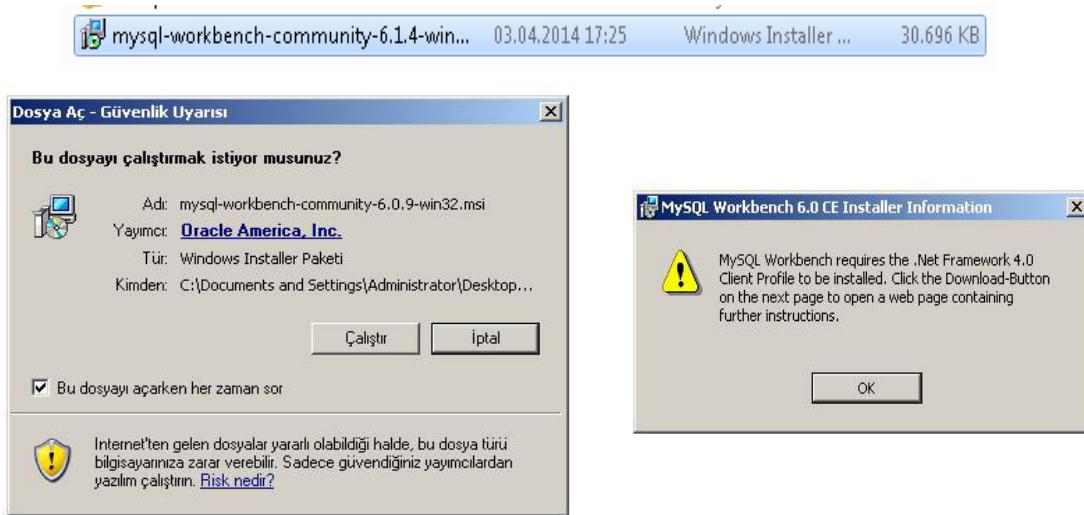
Buradan istediğiniz dosya sisteminin Download butonuna tıkladığınızda karşınıza Oracle sistemine giriş yapıp yapmayacağınız sorulmaktadır. Bu size kalmış bir tercihtir. İster sisteme kayıt yaptırabilir ister sen kayıt yapmadan bu sayfayı atlayabilirisiniz.



No thanks, just start my download.

Biz kayıt yapmadan bu sayfayı atlamak istedigimizden “No thanks, just start my download.” linkine tıklıyoruz ve indirme işlemimiz otomatik olarak başlıyor.

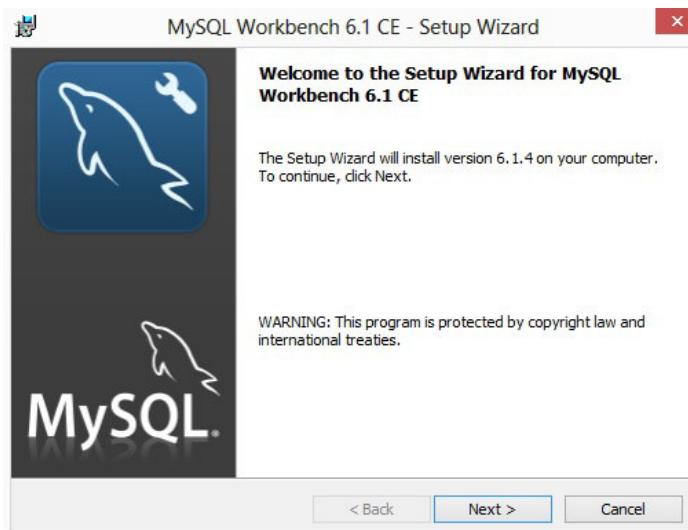
İndirmiş olduğumuz dosyaya çift tıklıyoruz ve gelen ekranda ÇALIŞTIR butonuna tıklıyoruz.



İşletim sisteminiz windows xp ve daha eski bir sürüm ise ekranınıza bu şekilde bir hata mesajı alırsınız işletim sisteminizde .Net framework sürümünün eski bir versiyonu olduğunu söylemektedir. Yeni sürümünü yüklemek için url adresimiz:

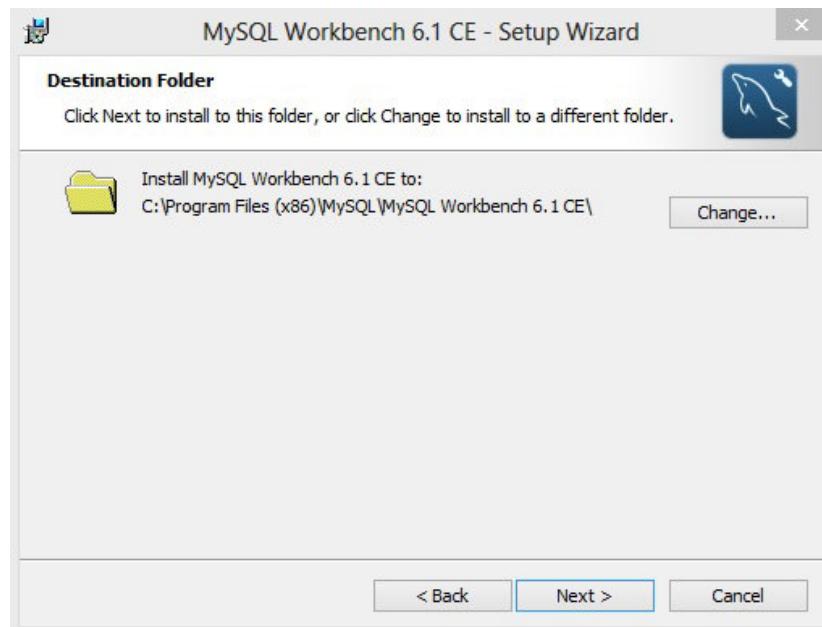
<HTTP://WWW.MICROSOFT.COM/TR-TR/DOWNLOAD/DETAILS.ASPX?ID=30653>

Net Framework’ ün versiyonunu yükselttikten sonra tekrardan indirmiş olduğumuz. MySQL Workbench’ in setup dosyasına çift tıklayarak çalıştır diyoruz. Buradan karşımıza çıkacak yükleme ekranı şu şekildedir.



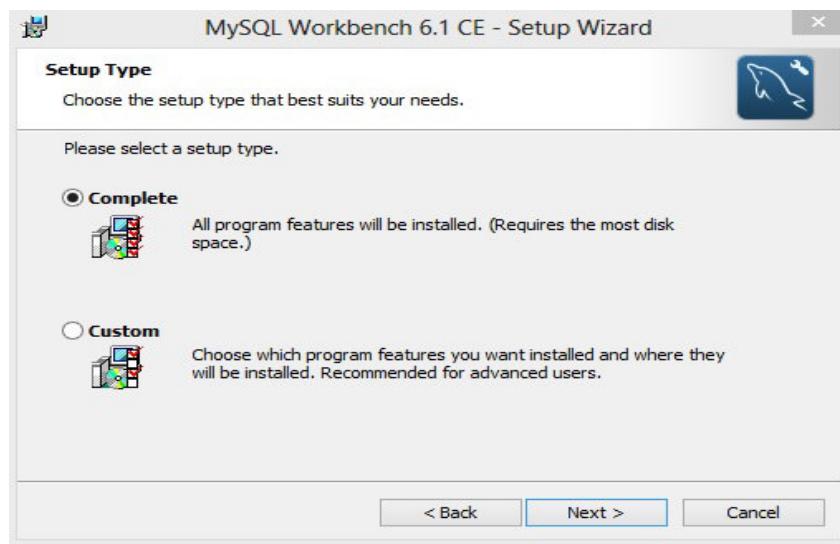
Bu pencereden NEXT’e tıklıyoruz.

Burada bize programı kuracağımız yolu sormaktadır. Bırakalım program önceden belirlenmiş yere kursun. NEXT’e tıklayarak devam edelim.



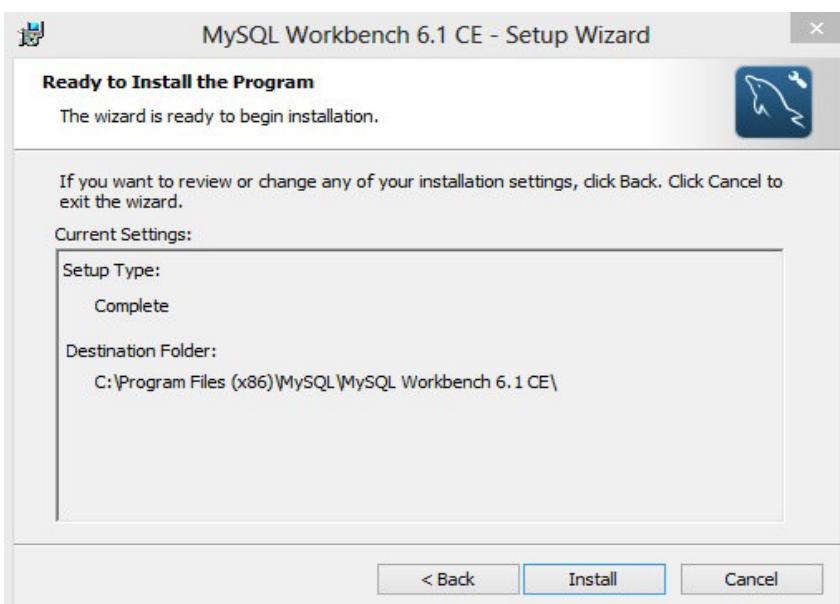
Kurulum seçeneklerini görmekteyiz. İsteğe bağlı olarak CUSTOM dan kendi belirleyeceğimiz dosyaları yükleyebiliriz.

Biz COMPLETE diyerek NEXT’e tıklayalım.

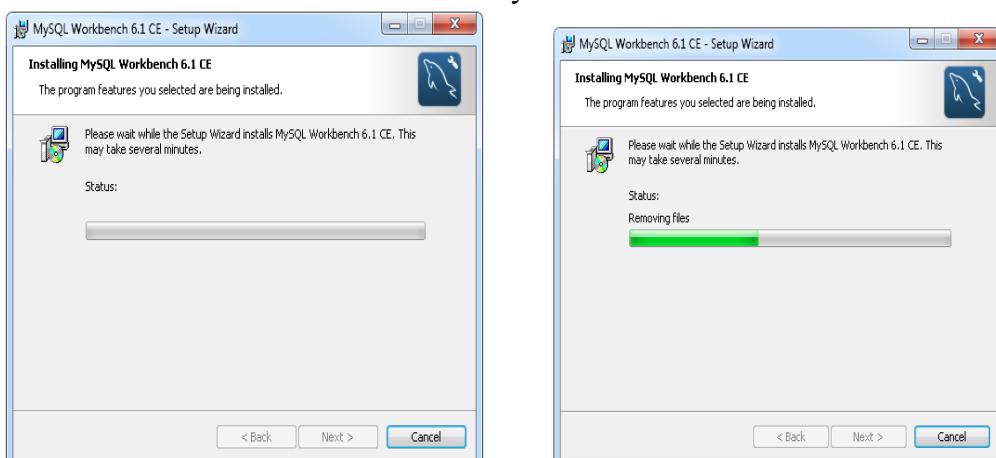


Seçmiş olduğumuz kurulum sistemini ve kurulacak yerin neresi olduğuna dair ve bize bilgi vermekte.

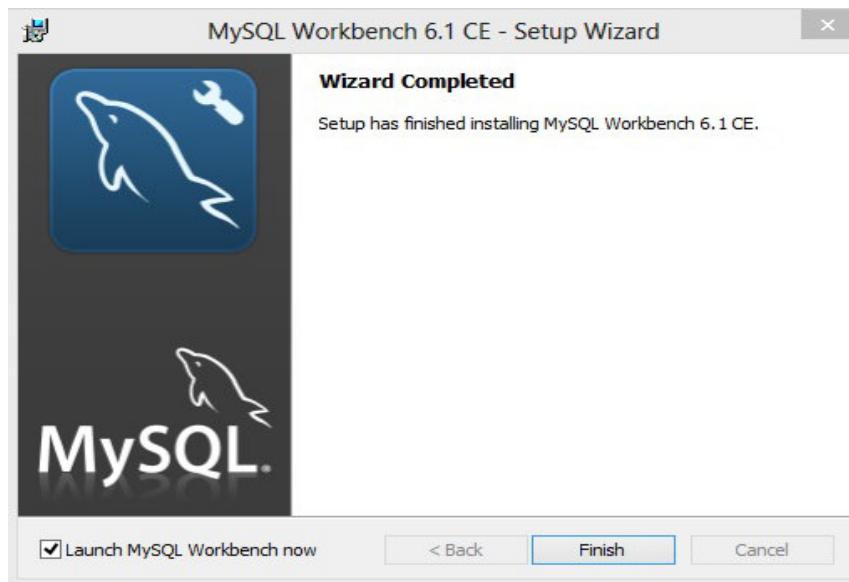
INSTALL'e tıklayarak kurulumuna başlıyoruz.



Kurulumun tamamlanmasını bekliyoruz.

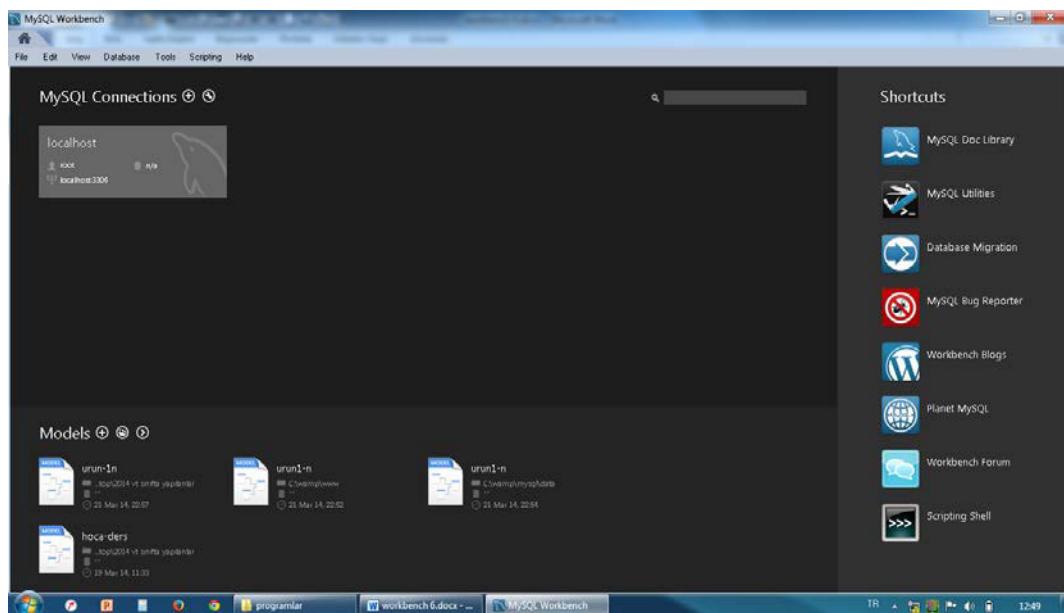


Kurulum işlemimiz FINISH'e tıklayarak bitiriyoruz.



9.2.1 Kurulan MySQL Workbench programının çalıştırılması

Bilgisayarımıza kurmuş olduğumuz MySQL Workbench programını “Başlat” menüsünden bulup tıklıyoruz. Karşımıza ilk gelen ekran ana ekranımız bu şekilde olmalıdır.



9.2.2 MySQL Workbench Kullanımı

MySQL Workbench görsel tasarım ara birimi ile veri yapılarını yönetmenize tablo ve veritabanlarını tasarlamanıza ve kullanıcılar tanımlamanıza yardımcı olur.

MySQL Workbench karmaşık ER(Varlık-İlişki) model oluşturma ve tersine mühendislik için ihtiyaç duyulan her türlü aracı içerir.

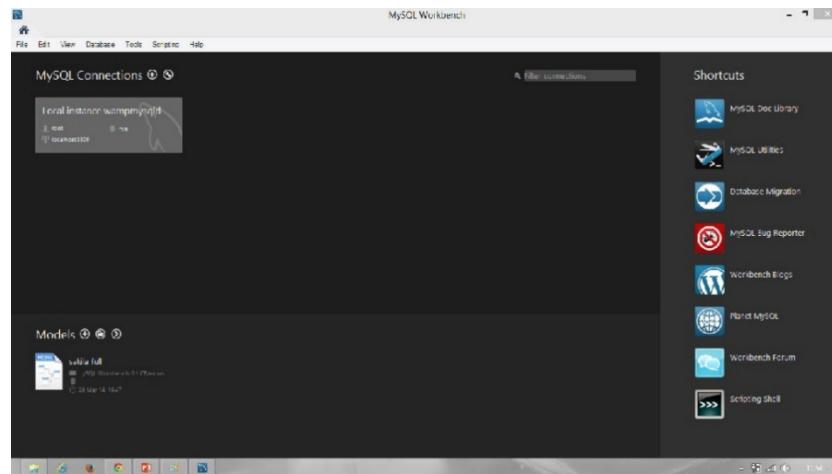
Workbench, SQL sorgusu yazmadan ER(Varlık-İlişki) diyagramına göre veritabanını şekillendiriyor.

Tablo ve ilişkileri oluşturuyor. Değişiklikleri de otomatik şekilde güncelleyebiliyor.

Tablo verilerini düzenlemek SQL sorgularını çalıştırırmak komut dosyalarını düzenlemek var olan veri tabanını yönetmek için MySQL Workbench kullanabiliriz.

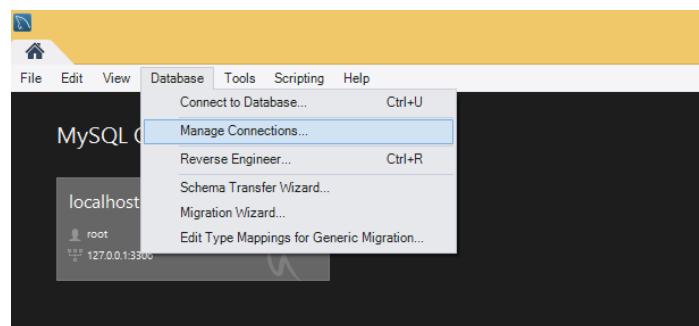
MySQL Workbenchi kurduktan sonra, MySQL Workbench açıyoruz

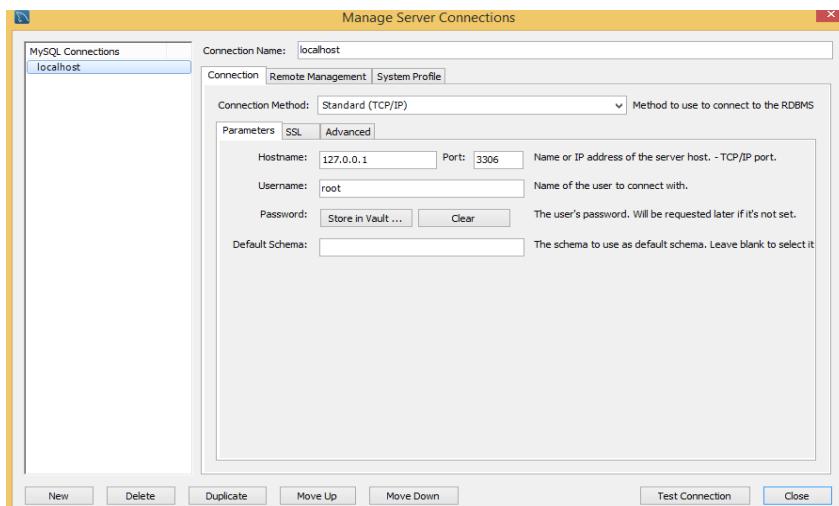
Bilgisayarımıza kurmuş olduğumuz MySQL Workbench programını “Başlat” menüsünden bulup tıklıyoruz. Karşımıza ilk gelen ekran ana ekranımız bu şekilde olmalıdır.



Kendi bilgisayardaki local hosttaki MySQL'e bağlanacağımız için Localhost bağlantısının sağlanması gereklidir.

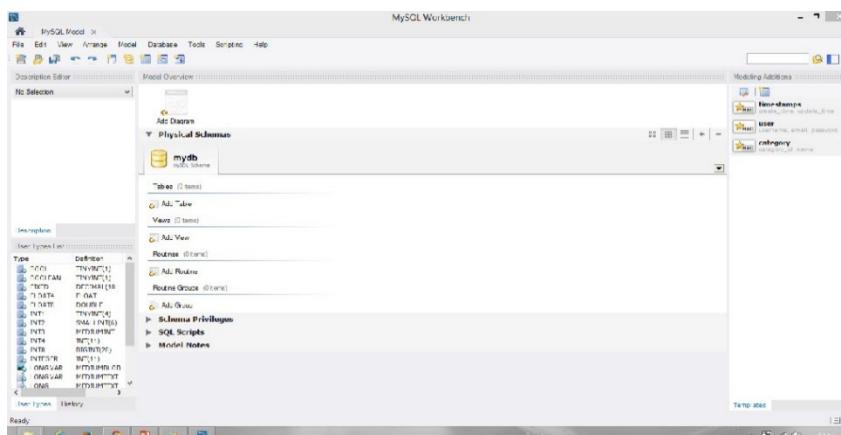
Eğer Localhost bağlantısı yoksa Database → Manage Connection sekmesi tıklanarak gelen ekranda alttaki New sekmesi tıklanarak gelen ekranda localhost olarak yeni bir bağlantı oluşturulur.





Yeni bir veri modeli oluşturmak için öncelikle FILE bölümünün altında bulunan NEW MODEL butonuna basıyoruz.

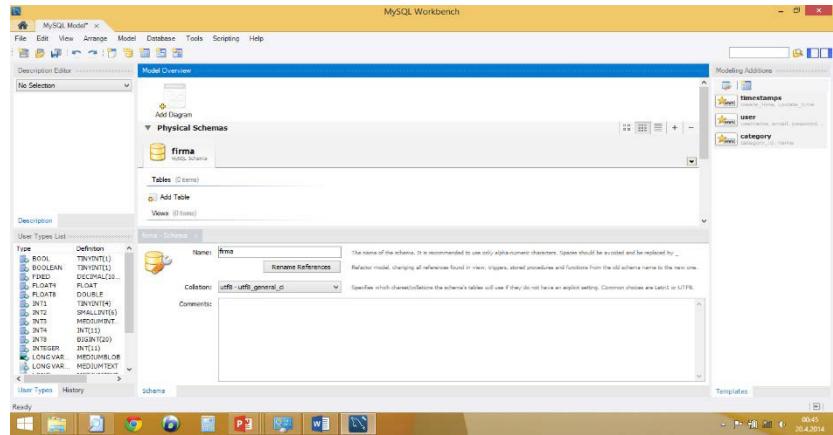
MYDB adında bir veri tabanı oluşturuyor ismini değiştirelim. Örneğin firma olsun bunun için MYDB ye çift tıklıyoruz.



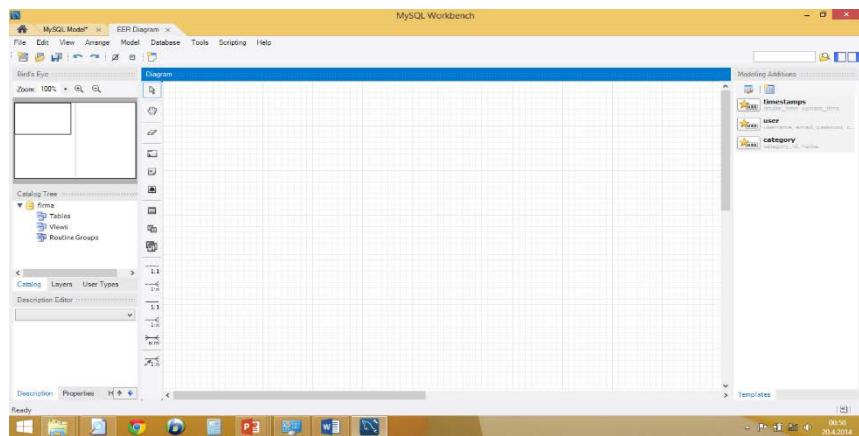
Daha sonra karşımıza aşağıdaki gibi bir ekran gelmektedir. NAME yazan yere firma yazıyoruz ve artık veri tabanımızın ismi firma olmuştur.

NOT: Veri tabanlarının isimleri değiştirmeden önce mutlaka yedeğini alınız.

Şimdi EER DİYAGRAMI oluşturalım. Bunun için ADD DIAGRAM'a çift tıklayalım.



EER DIAGRAM'a çift tıkladığımızda karşımıza gelecek olan ekran bizim tablolarımızı ve bunlar arasındaki ilişkileri kuracağımız yerdir.

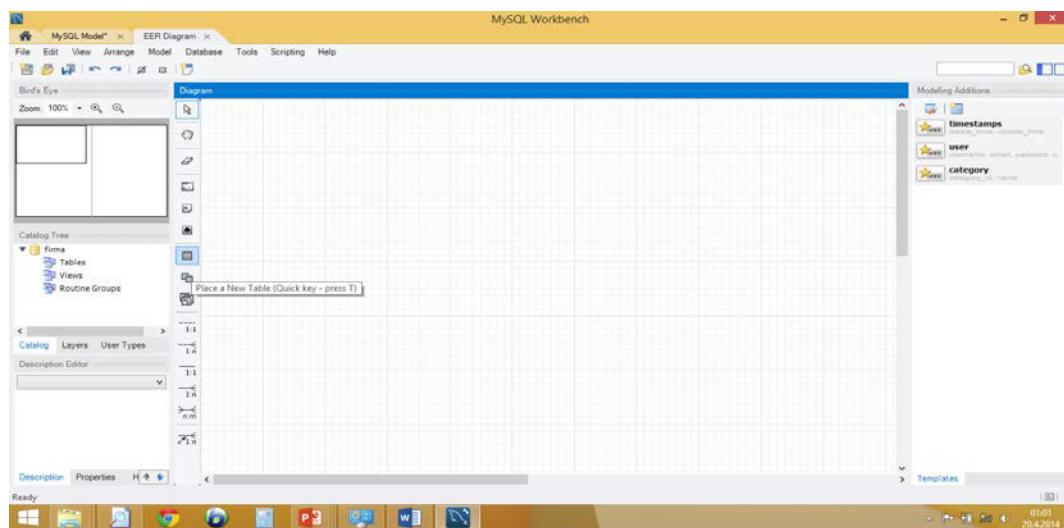


Çalışma alanındaki araç çubuğuuna bakacak olursak:

- | | |
|--|---|
| | Nesneleri seçmeye yarar. |
| | Taşıma işlemleri için kullanılır. |
| | Silme işlemleri için kullanılır. |
| | Birbiriyle ilişkili tabloların aynı renk altındaki bir katmanda toplanmasını sağlamak için yapılmış bir araç. |
| | Çalışma alanına yazı eklemek için kullanılır. |
| | Çalışma alanına resim yerleştirmek için kullanılır. |
| | Tablo oluşturmak için kullanılır. |
| | View oluşturmak için kullanılır. |
| | Routin yazmak için kullanılır. |
| | Tablolar arasında tanımsız bire-bir ilişki kurmak için kullanılır. |
| | Tablolar arasında tanımsız bire-çok ilişki kurmak için kullanılır. |
| | Tablolar arasında tanımlı bire-bir ilişki kurmak için kullanılır. |
| | Tablolar arasında tanımlı bire-çok ilişki kurmak için kullanılır. |
| | Tablolar arasında çoka-çok ilişki kurmak için kullanılır. |
| | Varolan kolonları kullanarak ilişki oluşturmak için kullanılır. |

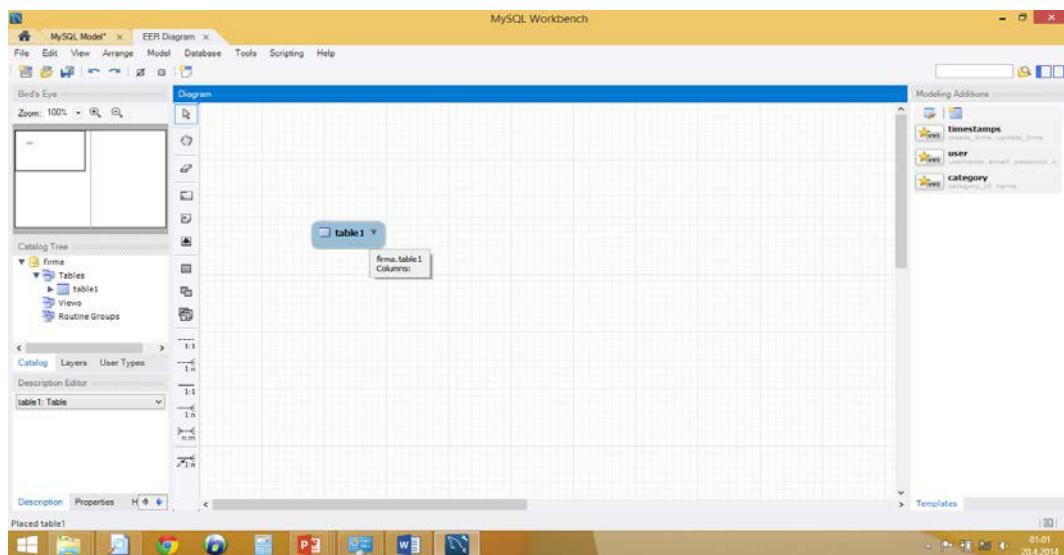
9.2.3 MySQL Workbench Kullanarak Tablo ve İlişki Oluşturma

Şimdi tablo oluşturmakla işe başlayacağız. Aşağıdaki ekran görüntüsünde okun gösterdiği yere bir kere tıkliyoruz.



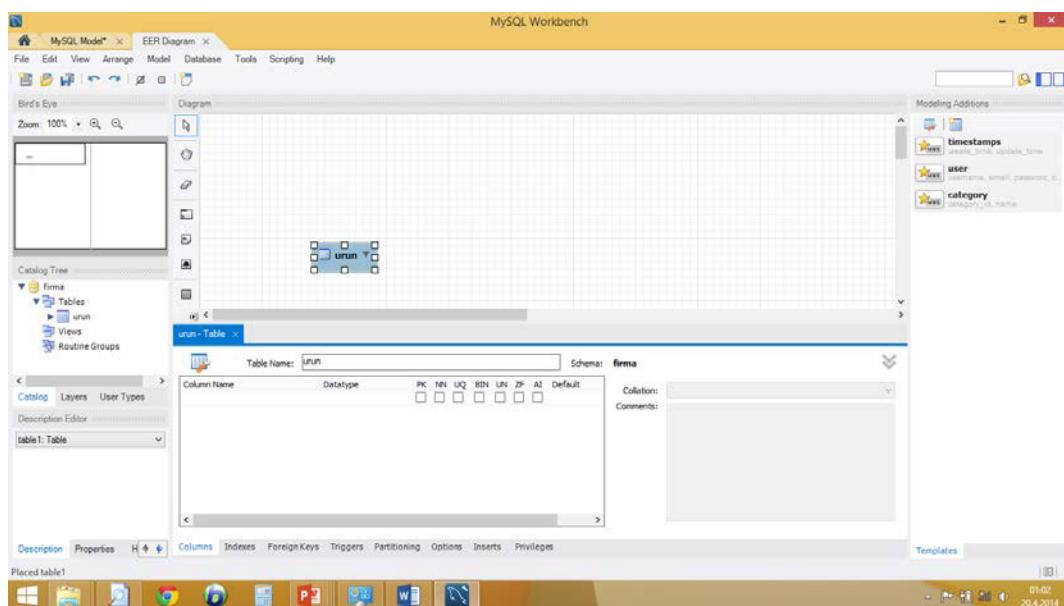
Daha sonra çalışma alanında istediğimiz yere tek tıklamayla tablo oluşturuyoruz.

Tablo ilk oluşturulduğunda aşağıdaki gibi görünmektedir. Burada tablo ismi otomatik olarak program tarafından belirlenmiştir. Bunu değiştirmek için tablonun üstüne çift tıkliyoruz.



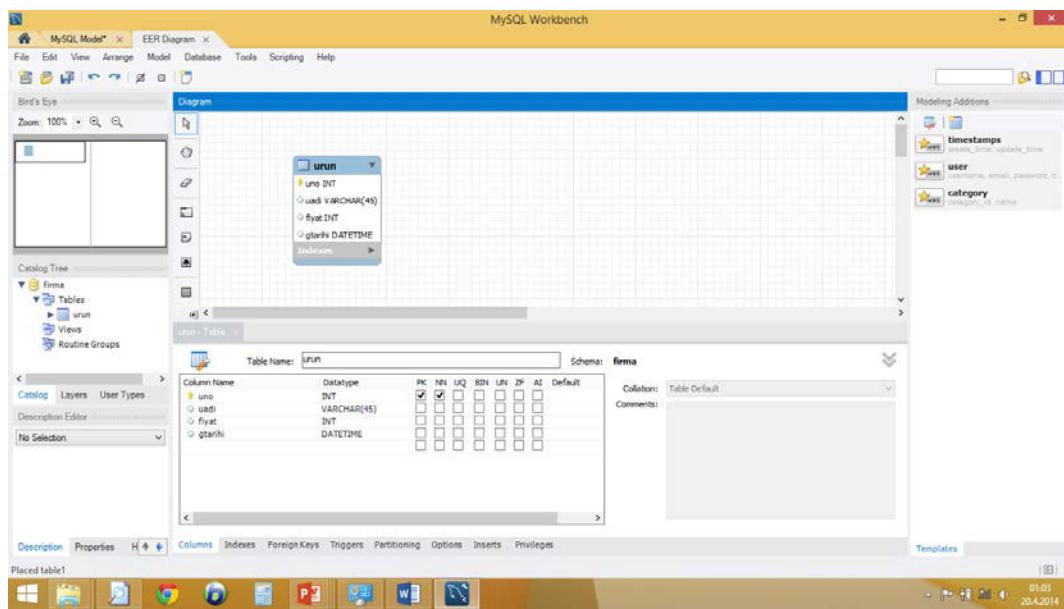
Aşağıdaki gibi bir görüntü ortaya çıkması gerekiyor.

Burada tablo'nun özelliklerinin değiştirilebileceği bir sekme açılıyor. "Name" yazan yerin karşısındaki tabloyu değiştirdiğimizde isim değişecektir.

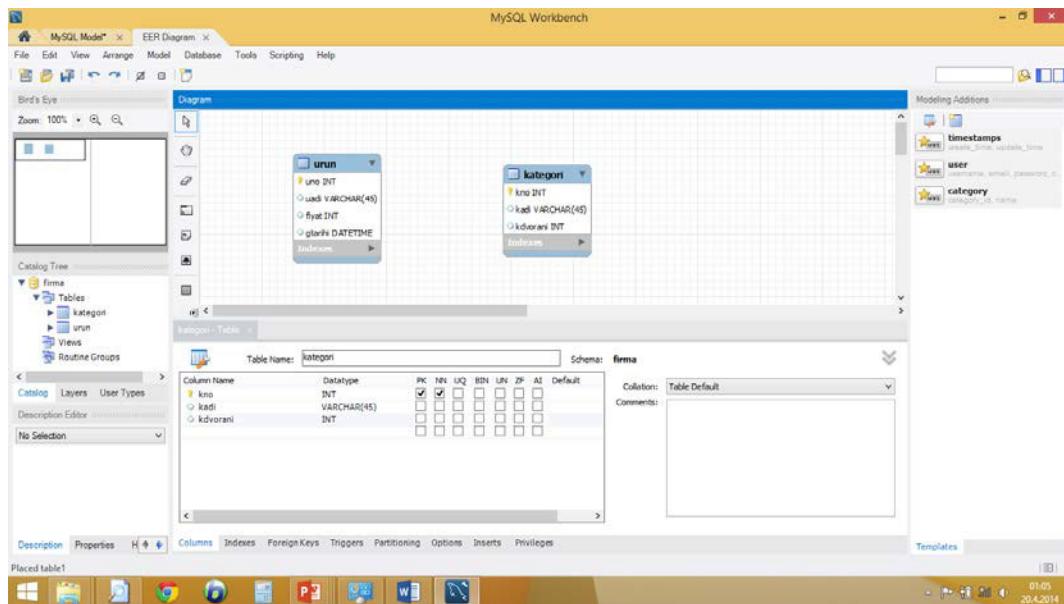


- PK->Primary Key,
- NN->Not Null,
- UQ->Unique (Benzersiz),
- BIN->Binary,
- UN->Unsigned(işaretsiz-sayılarda pozitif olma durumunu belirler),
- ZF->Zero Fill,
- AI->Auto Increment (Otomatik artırma) manasına gelir.

Tablodaki alanların adları, kısıtlamalar, veri türleri, uzunlukları vb. tanımlanır.



İlişki kurmak için ikinci bir tablo aynı şekilde oluşturulur.



İlişki oluşturmak için aşağıdaki araç çubuğundan faydalanzıız, bu araç çubuğundaki seçenekler daha önce anlatılmıştı. Yukarıdaki ilişkiyi oluşturmak için tıklaırız. Bu bire-çok ilişki oluşturacağımıza manasına gelmektedir. “Çok” ilişki kuracağımız tabloya önce tıklaırız. Sonra da “bir” ilişki kurulacak tabloya tıklaırız ve arada kesik çizgili bire-çok ilişkinin kurulduğunu artık görebiliriz. Bu tanimsız bire-çok ilişkidir. İlişkiler için tanımlı olanları düz çizgiyle, tanimsız olanları kesik çizgiyle belirtılır. Diğer sayfadaki resimde iki çizgiyle gösterilen taraf bir, ok işaretiley gösterilen taraf ise çok ilişkiyi tanımlamak için kullanılır.

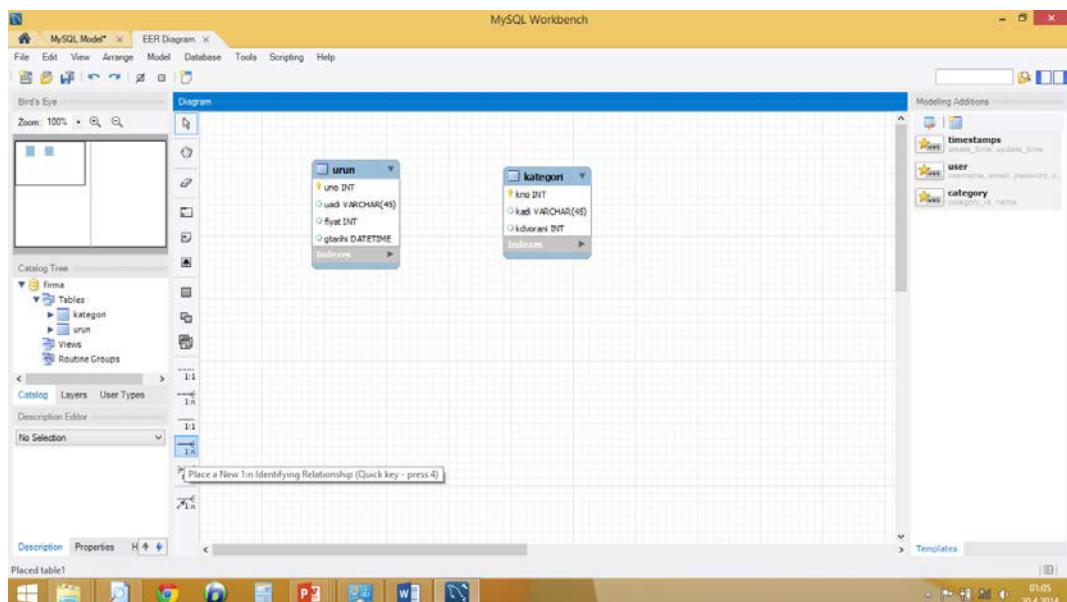
Bunların dışında Mysql workbench'te ilişkiler tanımlı ve tanimsız olarak kurulabilmektedir. Tanımlı, tanimsız ayrimını örneklerle anlatırsak:

Bir kitap bir sahibe aittir ve bir sahip çok kitaba sahip olabilir. Ama kitap sahipsiz var olabilir ve sahibini değiştirebilir. Kitap ve sahibi arasındaki bu bağıntı tanımsız bağıntıdır. (non-identifying)

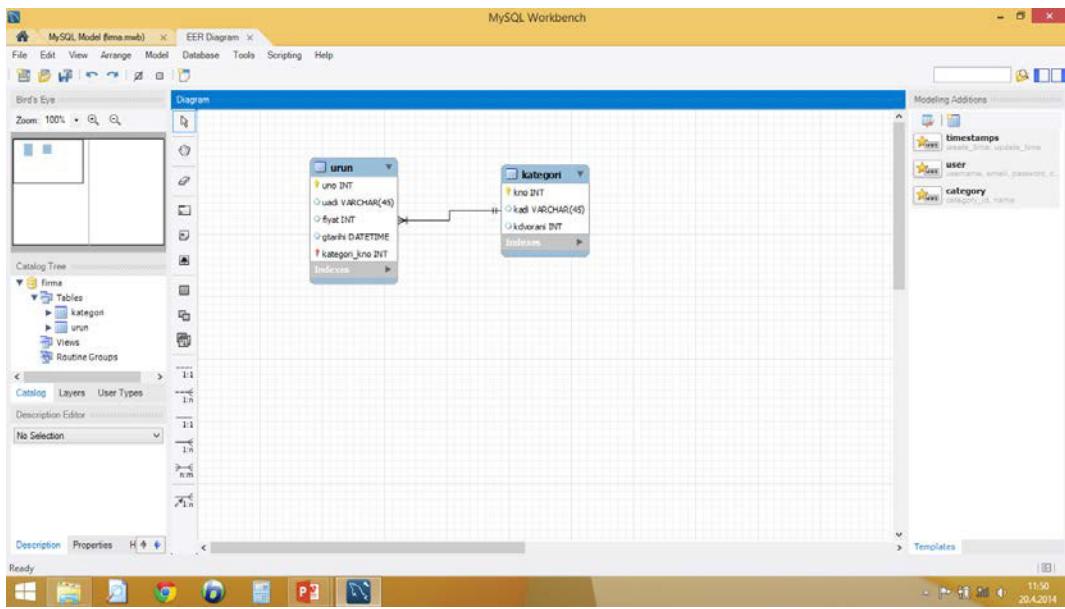
Bir kitap bir yazar tarafından yazılmış olmasına rağmen, bir yazar bir çok kitap yazmış olabilmektedir. Ama kitap yazarsız yazılamaz. Böylece kitap ve yazar arasındaki ilişki tanımlı ilişki olmuş oluyor. (identifying relation)

Bir bağıntının tanımlı veya tanımsız olmasının veritabanında bir karşılığı vardır. Eğer bir ilişkiyi tanımlı olarak belirlerseniz, foreign key hem kendi görevini yapar hem de primary key gibi davranır.

İlişki türü bire çok olduğu için ve ayrıca biz anahtar alanları belirlediğimizden okun olduğu ilişki seçilerek önce n tarafından tablo üzerine tıklanır, sonra da 1 tarafından tablo üzerine tıklanarak ilişki oluşturulur.

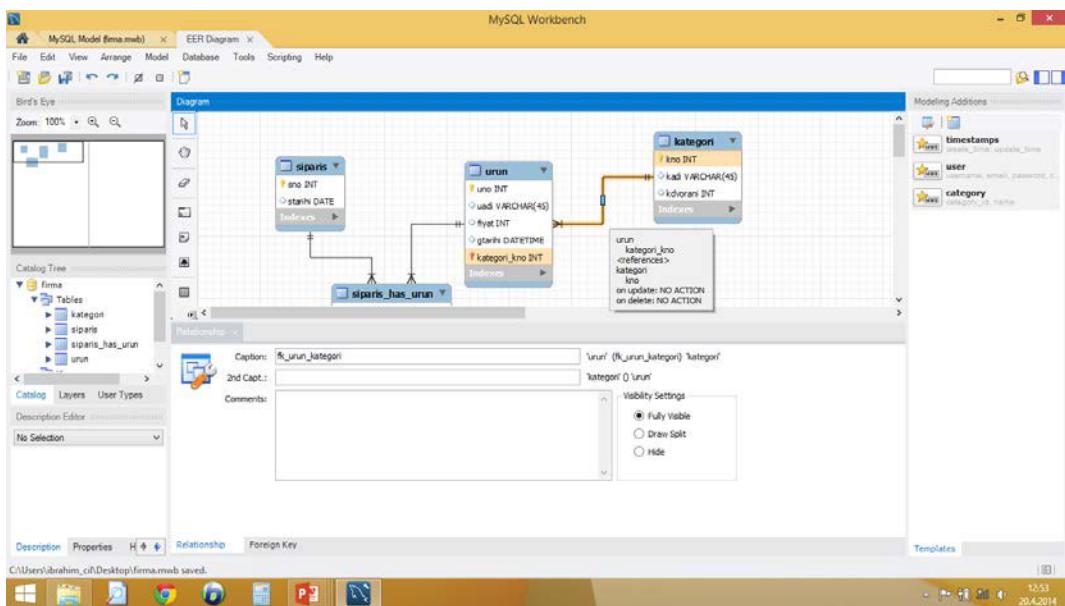


Yazılım kendisi otomatikman ilişkisi oluşturur.

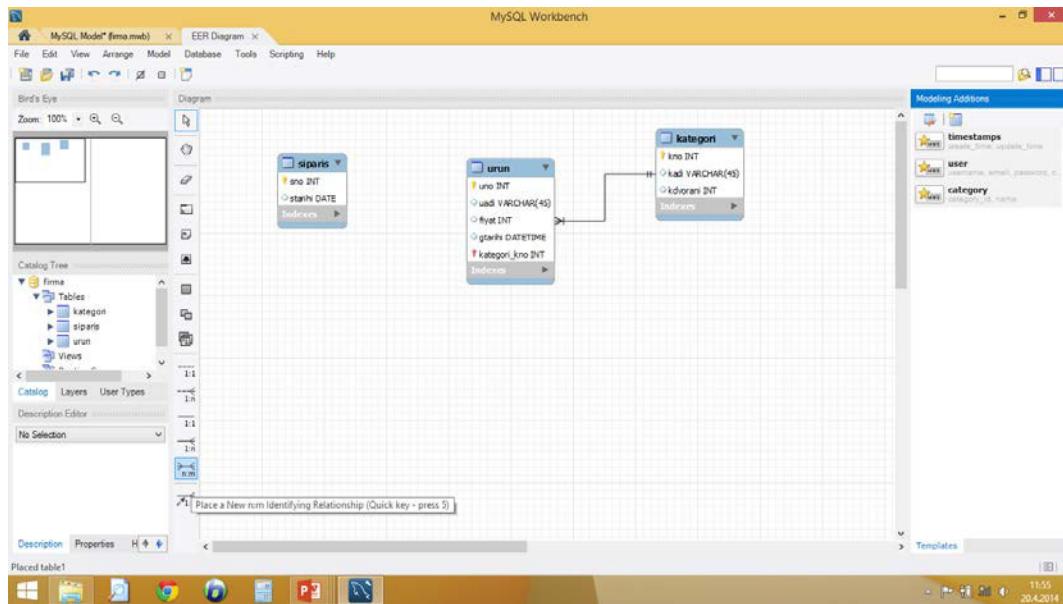


Şimdi ilişkiyi inceleyelim. Daha önce kurduğumuz ilişkinin üzerine çift tıkladığımızda aşağıdaki resimdeki gibi bir ayar penceresi alta ortaya çıkacaktır. “Relationship” ve “Foreign Key” olmak üzere iki sekme vardır. “Relationship” sekmesinde bu ilişkinin görünülürlüğüyle ilgili ayarlar yapılabilir. İlişkilerin hepsi gözüksün diyorsanız “Fully Visible”, çizgilerin gözükmemesini istemiyorsanız, ama ilişkiler gözüksün diyorsanız “Draw Split”, hiçbir şey gözükmesin diyorsanız “Hide” seçeneğini seçebilirsiniz.

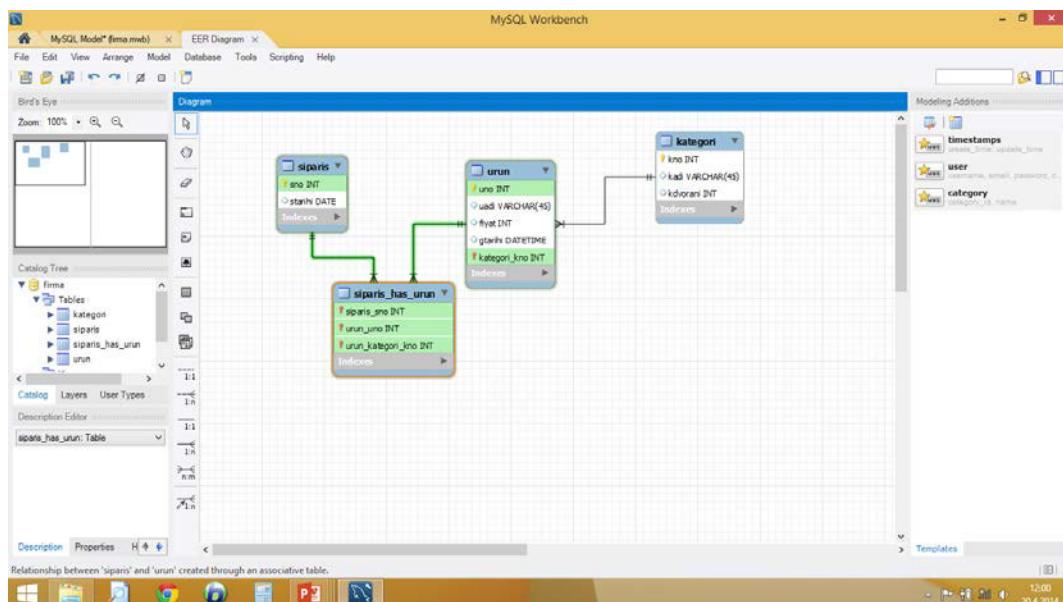
“Foreign Key” sekmesini açtığımızda ise aşağıdaki gibi bir görüntüyle karşılaşırız. Burada kurduğumuz ilişkinin tanımlı veya tanımsız olma durumunu “Identifying Relation” check box’ını kullanarak değiştirebiliriz. Mandatory foreign keylerin zorunluluk durumlarını belirler. Bunun veritabanındaki karşılığı null olma/olmama durumudur. Eğer zorunluluk check box’ları işaretli olursa bu foreign key’in null olamayacağı manasına gelmektedir. Kendi tasarıınıza göre istediğiniz şekilde değiştirebilirsiniz.



Çoka-çok ilişki oluşturmak içinde siparis adında üçüncü tabloyu oluşturduk. Okun olduğu ilişki simgesine tıklayıp ilişki kurulacak iki tabloya sırasıyla tıklanır.

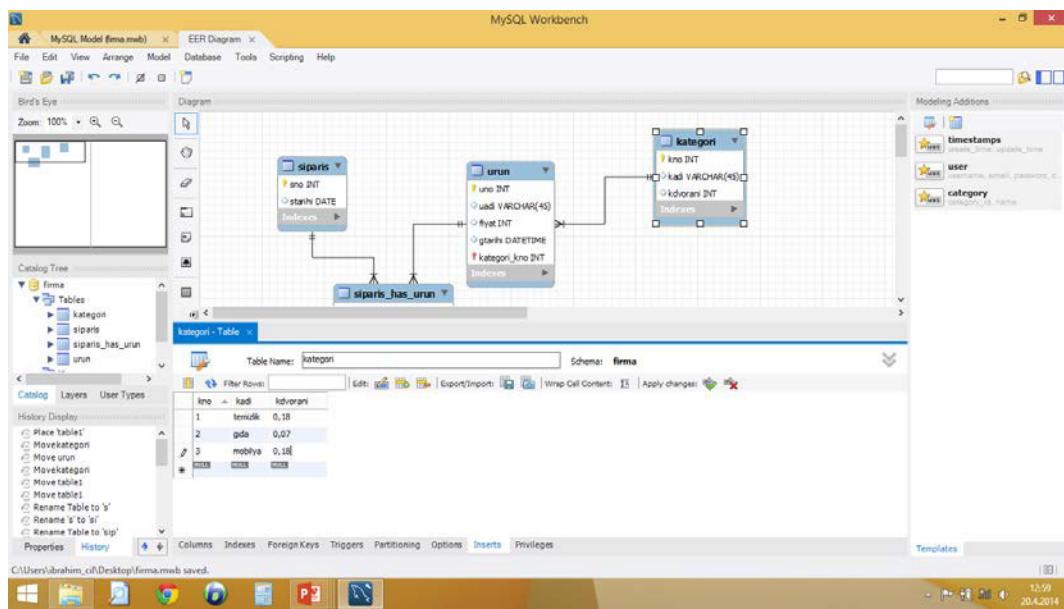


Şekildeki gibi çoka çok ilişki oluşturulmuş olur.

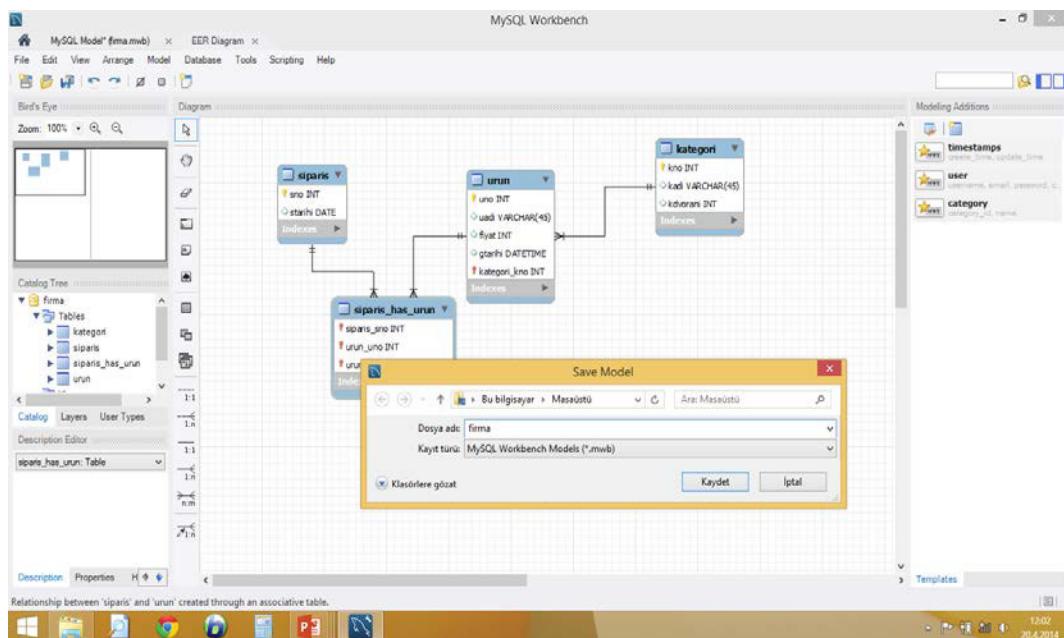


İlişkisi kurulmuş tablalarımızı görüyoruz. Şimdi bu tablolarımıza veri girişi yapalım.

Urun tablosuna tıkladığımızda aşağıdaki sekmlerden INSERTS yazan yere tıklıyoruz. Karşımıza çıkan boş kutucuklara verilerimizi giriyoruz. Aynı şekilde veri silme işlemlerinde buradan yapmaktayız.



File menüsünden oluşturulan modele bir isim verilerek kaydedilir.



9.2.4 MySQL Workbench Kullanarak Veri Aktarımı

Database Menüsü

“Manage Connections...” sekmesinde belirlediğiniz bağlantılar, bir daha ayar yapmanız gereklilikten kurtulmak için saklanır.

“Reverse Engineer...” veritabanındaki tabloları okuyup SQL kodlarını import etmeye yarar.

“Forward Engineer...” ise çalışma alanınızda bulunan diyagramların veritabanına export edilmesini sağlar.

“Synchronize Model...” çalışma alanınızdaki değişikliklerin veritabanıyla eşitlenmesini sağlar.

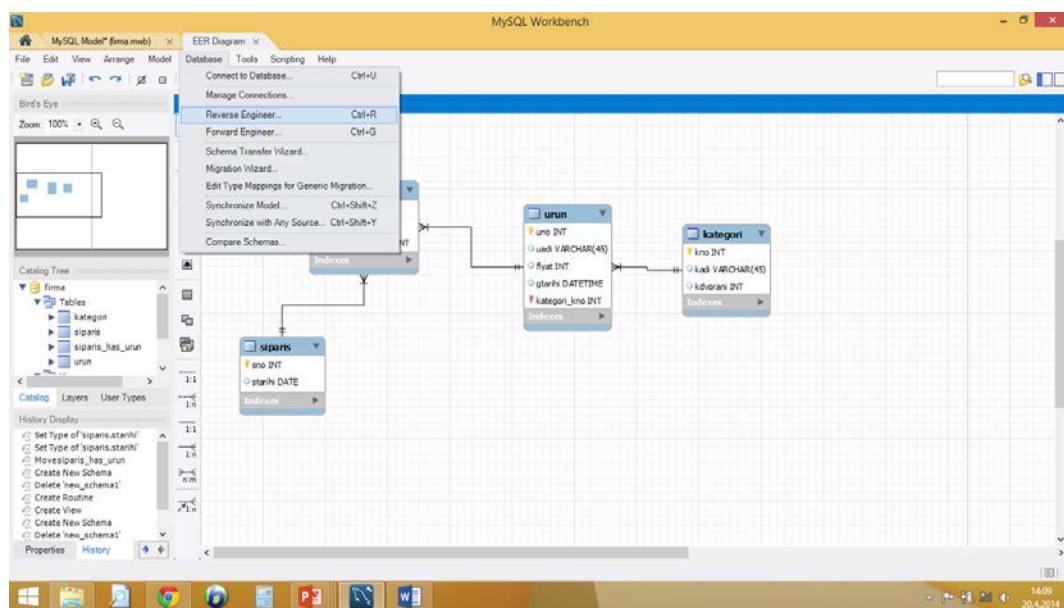
“Synchronize Model with Any Source...” ise herhangi bir kaynaktan alınan tabloları senkronize eder.

“Generate Catalog Diff Report...” iki kaynak arasındaki farklılıkları bulur. Bunlar iki SQL script dosyası veya iki veritabanının karşılaştırılması şeklinde olabilir.

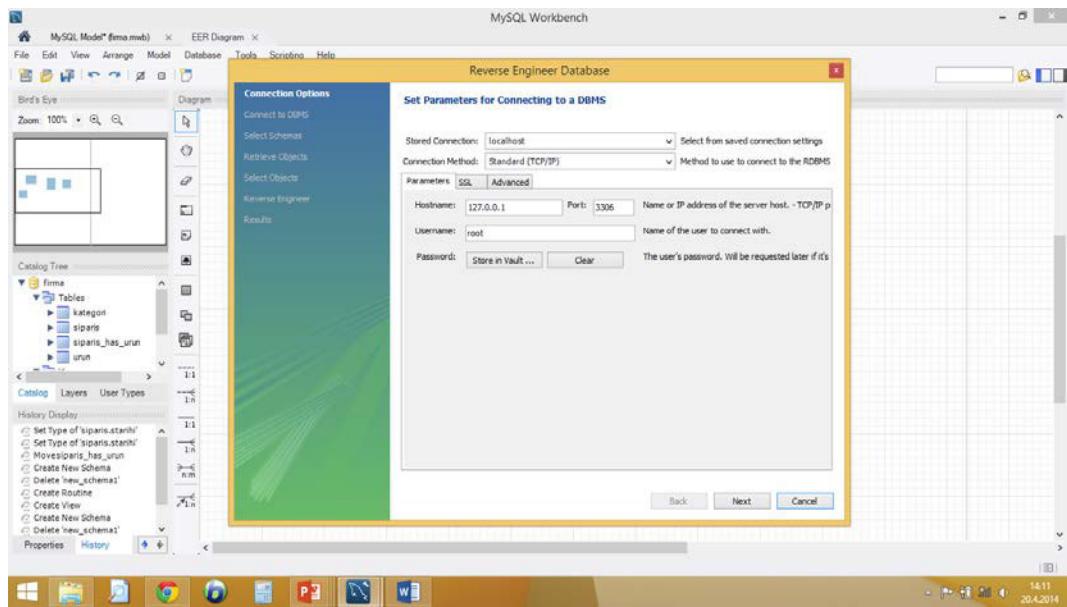
9.3 Wampserver Sunucusundaki Bir Veritabanını Modeli Olarak Model Ekranına Aktarma

Database sekmesinden “Reverse Engineer...” e tıklayarak Wampserver sunucusundaki auzef veritabanındaki tabloları okuyup model ekranına almak istiyoruz.

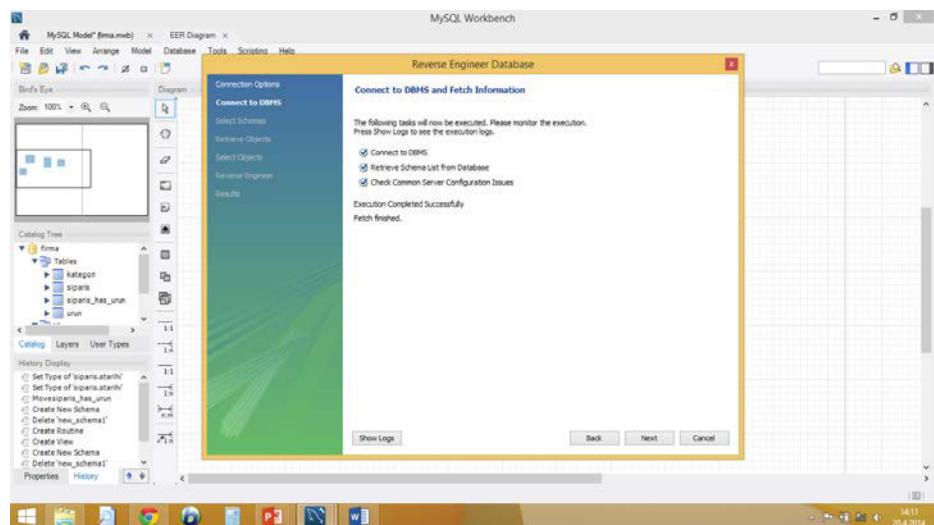
Wampserver dan oluşturmuş olduğumuz veri tabanını mysql workbenche İÇE AKTARIM yolu ile alalım. Bunun için DATABASE menüsünden REVERSE ENGINEER'e tıkliyoruz.



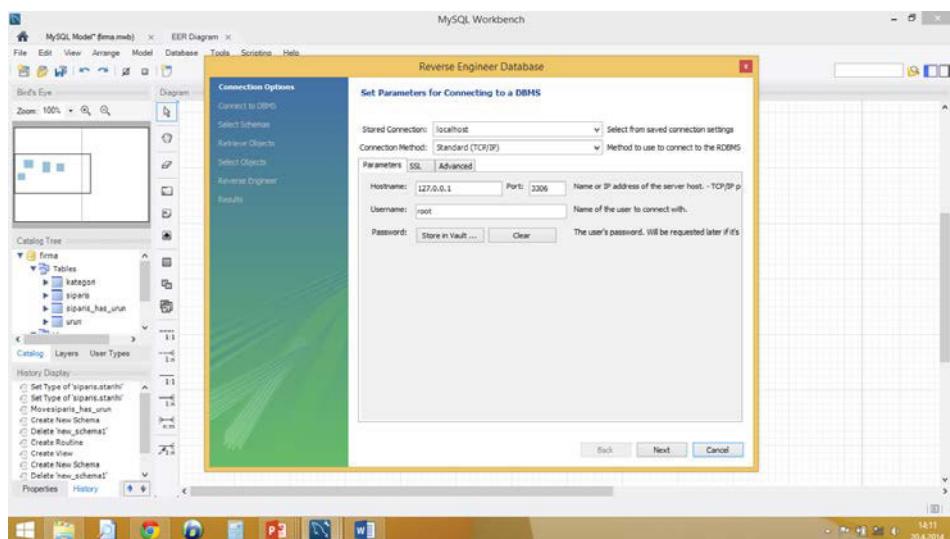
Gelen ekranda Next diyoruz.



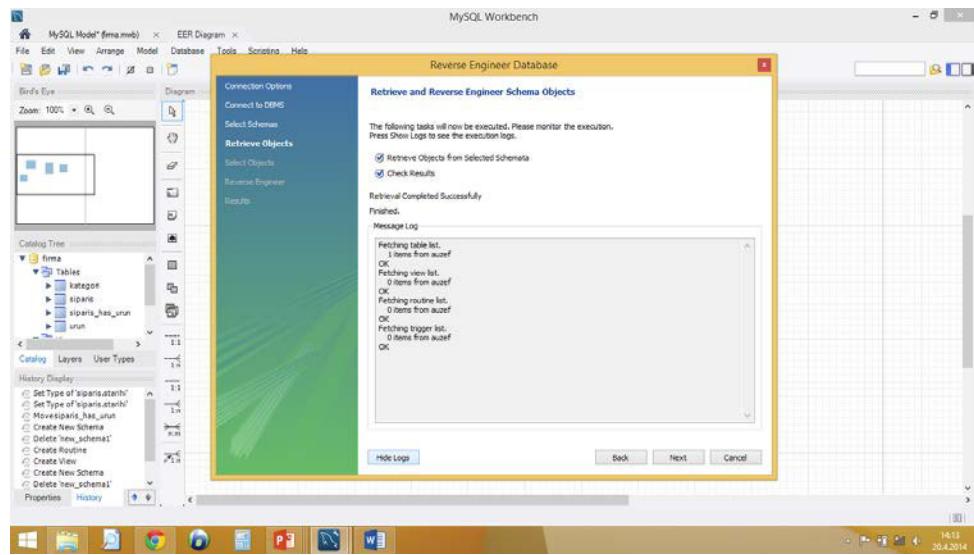
Gelen ekranda Next diyoruz.



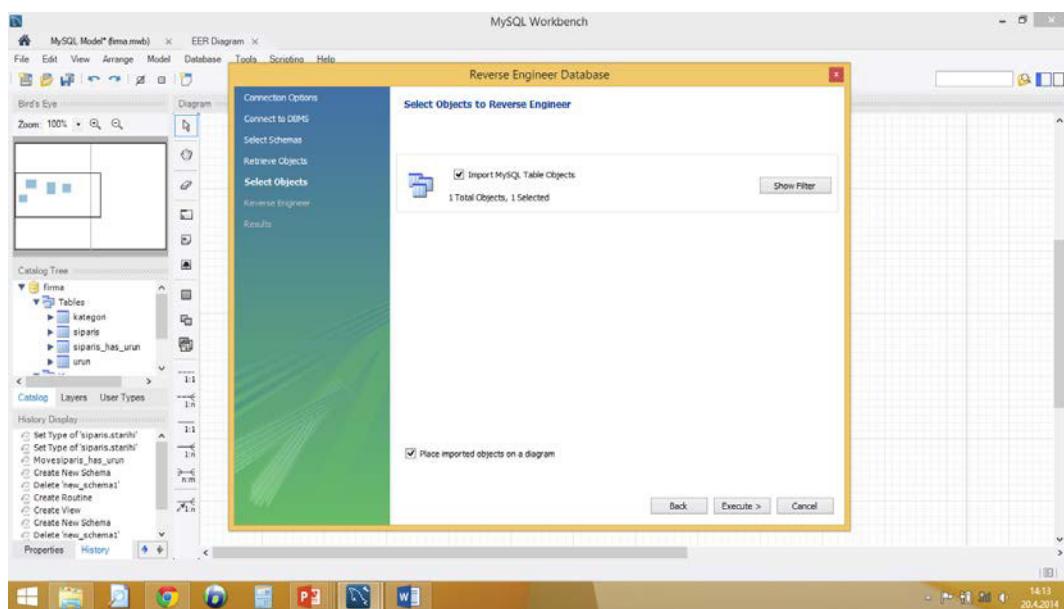
Gelen ekranda Next diyoruz.



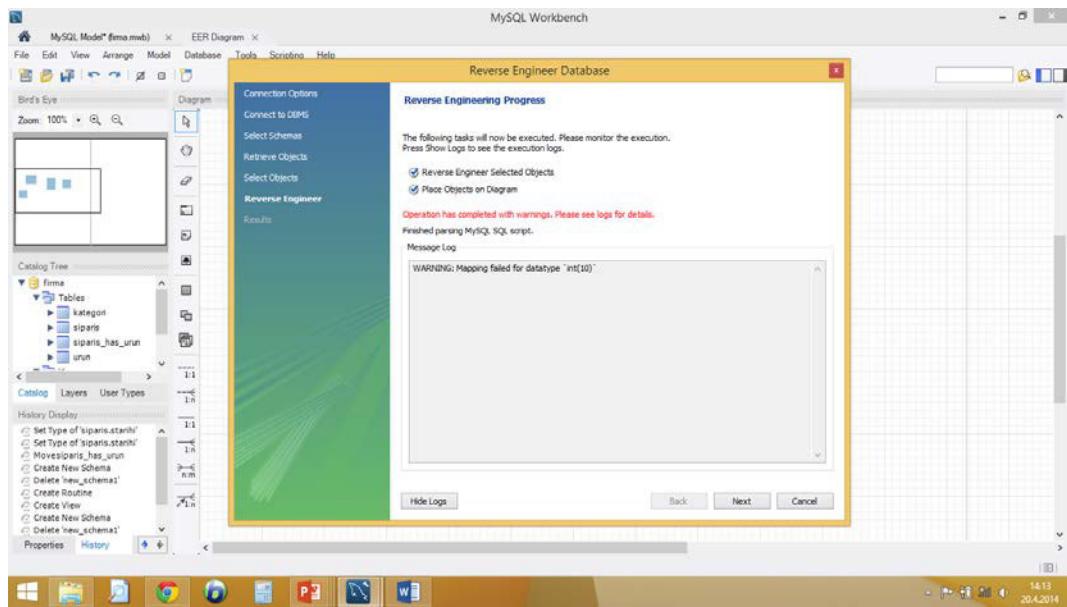
Gelen ekranda Next diyoruz.



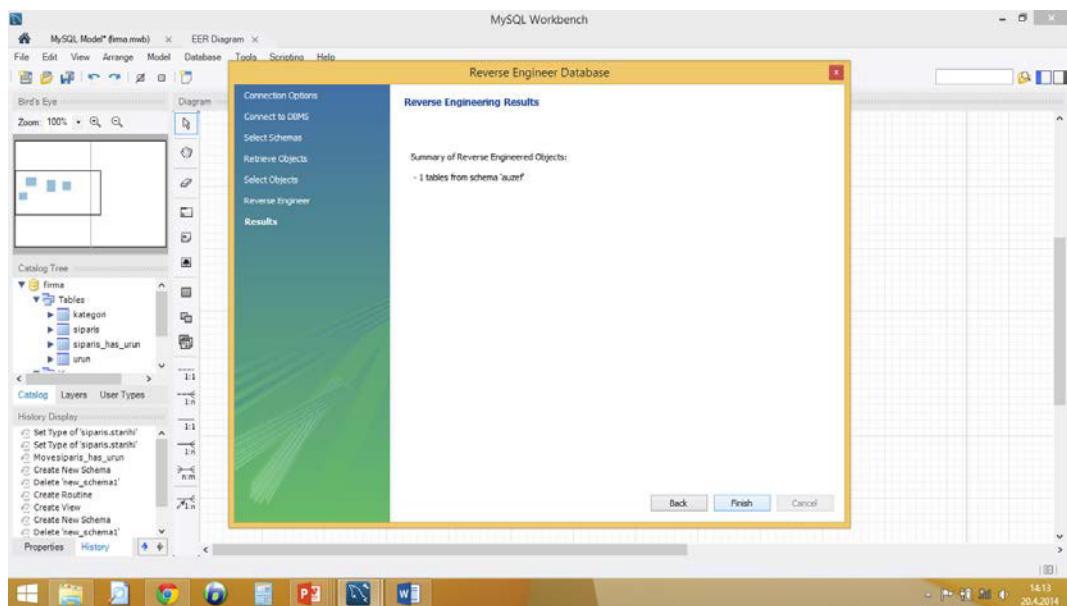
Gelen ekranda Execute diyoruz.



Gelen ekranda Next diyoruz.

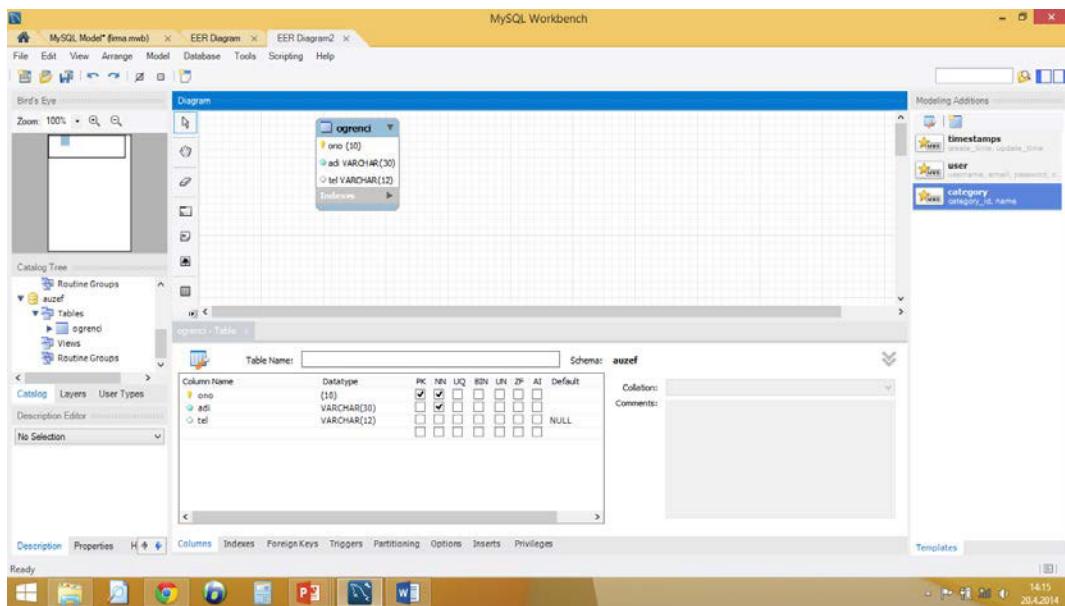


Bitir diyoruz.



Yeni bir model üzerinde öğrenci tablosu görülmektedir.

Sağ taraftaki bölümde de auzef veritabanı listelenmiştir.

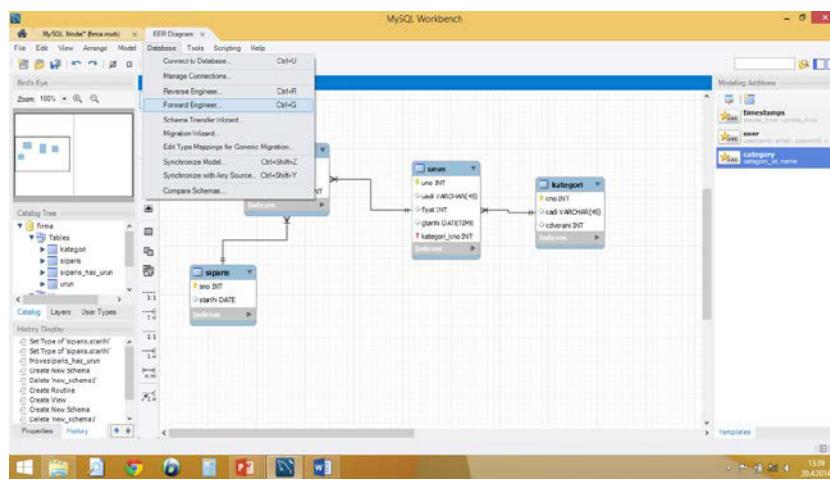


9.4 MySQL Workbench Kullanarak Veri Aktarımı

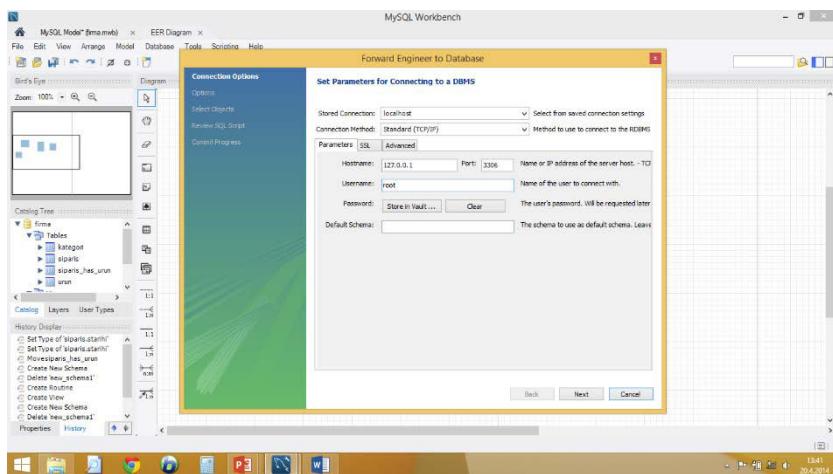
9.4.1 Hazırlanan Modeli Veritabanı Olarak Wampserver Sunucusuna Aktarma

Mysql workbench te hazırlamış olduğumuz veri tabanlarını MySQL veritabanı sunucusuna aktarmayı göreceğiz.

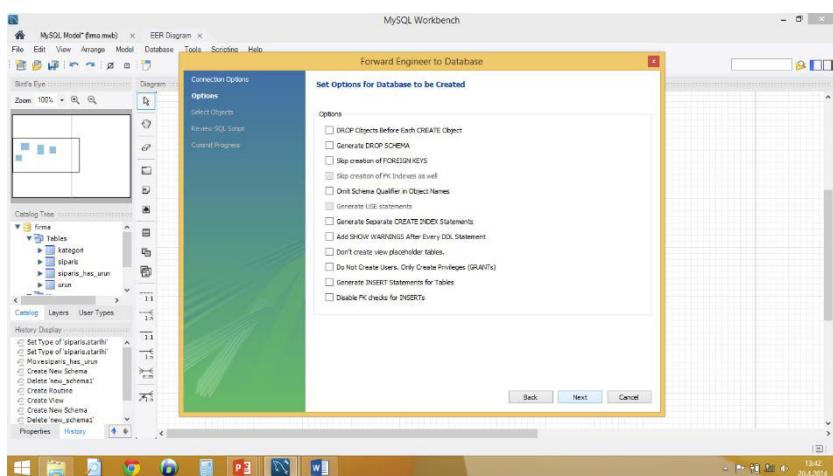
Bunun için DATABASE menüsünden FORWARD ENGINEER bağlantısına tıklıyoruz ve resimlerde görüldüğü üzere devam ediyoruz.



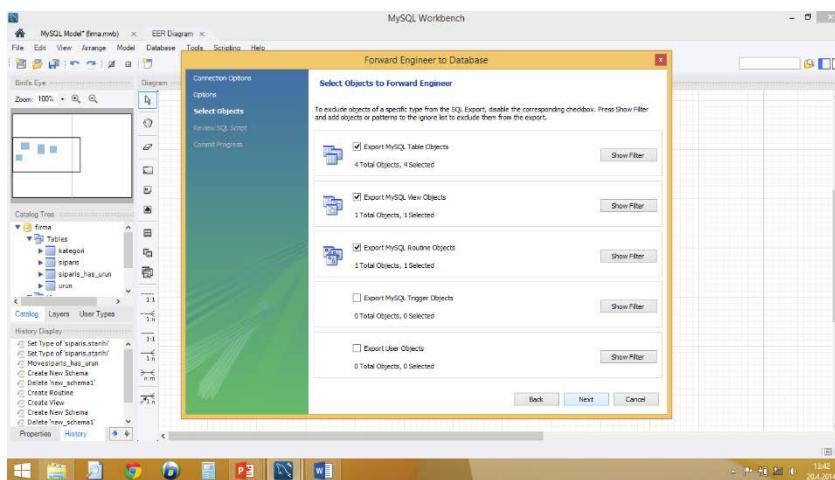
Gelen ekranda Next diyoruz.



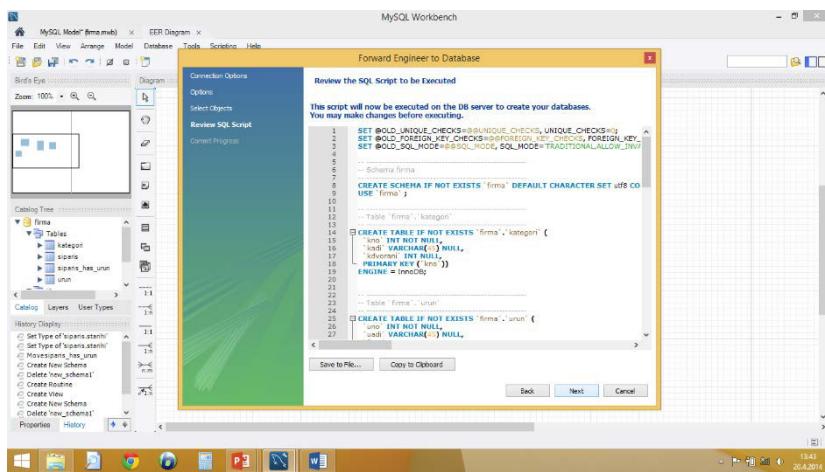
Gelen ekranada Next diyoruz.



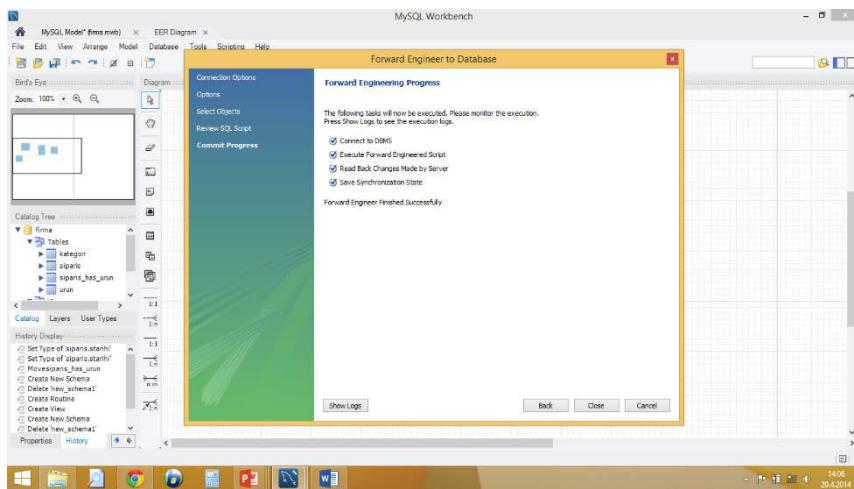
Gelen ekranada Next diyoruz.



Gelen ekranada Next diyoruz.



Aşağıdaki ekranda close tıklanarak kapatılır.



phpMyAdmin sekmesinden aktarılan verilen kontrol edilir. Firma veritabanına veriler aktarılmış olacaktır.

Uygulama Soruları

- 1)** MySQL Workbench kullanarak herhangi bir alanda bir veritabanı uygulaması geliştiriniz.

Bu Bölümde Ne Öğrendik Özeti

Bu bölümde bir uygulama için gerekli olan ve birden çok bilgisayarın erişebileceği veri tabanı yönetim sistemlerinin temel bilgileri ve kurulumları hakkında bilgi verildi. Özellikle MySQL ile çeşitli uygulamalar geliştirildi. Basit bir uygulamadan tutun da çok büyük kuruluşların verilerine kadar, ister bir web uygulaması olsun ister bir masaüstü uygulaması olsun, günümüzde bu gibi birçok alanda veri tabanı uygulamalarına ihtiyaç duyulmaktadır. Son yıllarda yapılan birçok proje çok sayıda bilgisayar tarafından kullanılabilecek şekilde tasarlanmaktadır. Bu yüzden, ağ ortamında birden fazla kullanıcı aynı proje üzerinde çalışabilmektedir.

Bölüm Soruları

1) Aynı verinin değişik yerlerde birkaç kopyasının bulunması, bir yerde değiştirilen verinin diğer yerde aynı kalması hangisine neden olur?

- a) Veri tutarsızlığı
- b) Veri Tutarlılığı
- c) Veri Paylaşımı
- d) Eş zamanlılık
- e) Veri Bağımsızlığı

2) Bir tabloda değişiklik yapılan verinin ilişkili olduğu diğer tablo veya tablolarda da aynı işlemin yapılması işlemeye ne denir?

- a) Veri Tutarlılığı
- b) Veri Bütünlüğü
- c) Veri Güvenliği
- d) Veri Bağımsızlığı

3) Aşağıdakilerden hangisi bir VTYS değildir?

- a) MySQL
- b) MSSQL
- c) Oracle
- d) XML

4) MySQL Workbench veri tabanı yönetim sisteminde Database menüsünde bulunan “Forward Engineer ...” ile aşağıdakilerden hangi işlem gerçekleştirilir?

- a) SQL komutlarıyla üzerinde denemeler yapmanızı sağlayan bir sekme açar.
- b) Çalışma alanınızda bulunan diyagramların veritabanına export (gönderilmesi) edilmesini sağlar
- c) Sekmesinde belirlediğiniz bağlantılar, bir daha ayar yapmanıza gerek kalmadan kullanılmak için saklanır.
- d) Veritabanındaki tabloları okuyup SQL kodlarını import (alınması) etmeye yarar ve E-R diyagramını oluşturur.
- e) Herhangi bir kaynaktan alınan tabloları senkronize eder

CEVAPLAR

1-A, 2-B, 3-D, 4-B,

10. WEB TABANLI VERİ TABANI SİSTEMLERİ

Bu Bölümde Neler Öğreneceğiz?

10.1 Web İle İlgili Temel Kavramlar

10.1.1 Internet'in Yapısı ve İstemci ve Sunucu Yaklaşımı

10.1.2 Sunucu (Server) Bilgisayarlar

10.1.3 Web Tarayıcılar (Web Browsers)

10.1.4 Web Üzerinde Veritabanı

10.1.5 Veri Tabanlarına Web'de Bağlanma

10.2 Web Teknolojileri ve Geliştirme Araçları

10.2.1 HTML

10.2.2 XML, Extensible Markup Language

10.2.3 Web Editörleri (Düzenleyiciler)

10.2.4 Web Tabanlı Veri tabanı Yönetim Sistemleri

10.3 PHP-MYSQL-APACHE Sunucu Üçlüsü

10.3.1 Apache Web Sunucusu

10.3.2 PHP

10.3.3 MYSQL

10.4 Web Uygulamaları

10.4.1 Web Uygulamalarının Çalışma Biçimi

10.4.1.1 Hazır Statik Sayfaların Web Sunucudan İstenmesi Durumu

10.4.1.2 Gelen Dinamik İsteği İşleyen Uygulama Sunucusunun Olduğu Durum

10.4.1.3 Gelen Dinamik İsteği İşleyen Uygulama Sunucusunun Veri Tabanı Sunucusıyla Birlikte Çalıştığı Durum

10.5 Web Uygulaması Geliştirme

10.5.1 APACHE-PHP-MYSQL Kurulumu

10.5.1.1 Apache Php Mysql Kurulumu (Hazır Programlar İle)

10.5.1.2 MySQL'de Veritabanı Oluşturma İşlemleri

10.5.1.3 PhpMyAdmin ile Veri tabanı Oluşturma

10.5.1.4 PhpMyAdmin ile Tablo Oluşturma

10.6 Dreamweaver Üzerinde Yeni Bir Site Açma

10.6.1 Local Info

10.6.2 Remote Info

10.6.3 Testing Server

10.7 Dreamweaver İle MYSQL Bağlantısı

10.7.1 Connection Name

10.7.2 Mysql Server

10.7.3 User Name

10.7.4 Password

10.7.5 Database

10.8 Dreamweaverde PHP ile Veritabanına Kayıt Ekleme

10.8.1 Form Oluşturma

10.8.2 Dreamweaver'de PHP ile Kullanıcı Adı-Şifre Korumalı Sayfalar Oluşturmak

10.8.3 İkinci Yol (Dreamweaver'de PHP İle Kullanıcı Adı-Şifre Korumalı Sayfalar Oluşturmak)

10.8.4 Dreamweaver'de PHP İle Veritabanından Kayıt Okuma ve Rapor Oluşturma

10.8.4.1 Dreamweaver'te Repeat Region Kullanımı

Bölüm Hakkında İlgi Oluşturan Sorular

- 1)** Web üzerinde Veri tabanı nasıl çalışır?
- 2)** Web tabanlı bir veri tabanı nasıl geliştirilir?
- 3)** Hangi konlarda web tabanlı uygulamaya gereksinim duyulur?
- 4)** Web formları nasıl hazırlanır?

Bölümde Hedeflenen Kazanımlar ve Kazanım Yöntemleri

Konu	Kazanım	Kazanımın nasıl elde edileceği veya geliştirileceği
	Apache-Php-Mysql Kurulumunu Yapabilir	Anlatım, Soru-Cevap, Tartışma
	Apache-Php-Mysql ile uygulama geliştirebilir	Alıştırma ve Uygulama, Örnek Olay
	Dreamweaver ile MySQL bağlantısını kurabilir	Bireysel Çalışma, Problem Çözme,
	Form Oluşturabilir	Proje Temelli Öğrenme

Anahtar Kavramlar

- Apache-Php-Mysql
- Form Oluşturma
- Dreamweaver
- Apache Web sunucusu
- PHP
- MYSQL
- WEB Uygulamaları

Giriş

Günümüzde bilişim sistemleri açısından tarihi bir değişim yaşanmakta ve bir Ağ ve Internet devrimi gerçekleşmektedir. Kurumların veri tabanına erişim için Web'in kullanımının sağladığı birçok avantaj vardır. İnternet donanım ve yazılım teknolojilerindeki ilerlemeler internet üzerinden veritabanı işlemleri yapmayı daha kolay hale getirmiştir. İnternette veritabanı uygulamaları ile Veritabanı yapıları ve içerikleri daha fazla geliştirilebilmiş ve çeşitlendirilebilmiş. Webde veritabanı uygulamaları veritabanı gelişimine kullanıcılar tarafından doğrudan etkileşimli olarak katkılar yapılmıştır. Veritabanında bulunan bilgilerin geniş kitlelere en ucuz ve en hızlı şekilde iletimi sağlanabilmiştir.

Internet bilgisayarlar arasındaki küresel bir bağlantı sistemidir. Bu sayede insanlar birbirleriyle veri transferi ve iletişim kurarlar. Internet sayesinde daha pek çok şey yapılmaktadır: İletişim (e-mail, video konferans.), Veriye erişim (veri tabanları), Başka bilgisayardan dosya transferi (ftp), İnternette ürün alım satımı vb. internette yaygın olarak kullanılan hizmetlerden yalnızca birkaç tanesidir. Internet dünyanın en büyük ve en çok kullanılan ağıdır. Kurumlar www'deki web sayfaları ile kendilerini dünyaya Internet ile tanıtmaktadır. Bazı örnekleri şu şekildedir: ticari (.com), kamu (.gov), eğitim (.edu) hem örgütSEL (.org), vb.

Web, HTTP standartını kullanarak bilgisayarların birbiriyle iletişimiminin sağlandığı bir ağdır. Web, bir ağ ortamında bilgiyi saklama, transfer etme, şekillendirme ve sunma işlemleri için evrensel olarak kabul edilmiş standartlar içeren bir sistemdir. World wide web Internete bağlı bilgisayarlardaki bilgiler için grafiksel kullanıcı arayüzüdür. WWW de başkalarına ulaşılacak istenen bilgiler HTML (Hyper Text Mark-Up Language) işaretleme dili ile kodlanarak HTTP (Hyper Text Transfer Protocol) protokolü ile talep edenlere aktarılır.

Bu derste verilen bilgiler doğrultusunda Web uygulamalarının nasıl geliştirildiği konusu ele alınacak. İnternet üzerinden erişilebilen ve yapılandırılabilen veri tabanı sistemlerinin nasıl oluşturulduğu anlatılacak. Hazırlanan MySQL veri tabanına internet üzerinden veri girmek ve veri tabanındaki kayıtlardan rapor almak için gerekli olan işlemlerin nasıl yapılacağı Dreamweaver editörü kullanılarak gerçekleştirilecek. Önce Yeni Bir Site Açılacek, sonra Dreamweaver ile MySQL bağlantısı gerçekleştirilecek, ardından oluşturulan formlarla veri tabanına veri girişi ve raporların nasıl alınacağı açıklanacaktır.

10. WEB TABANLI VERİ TABANI SİSTEMLERİ

Web Tabanlı Veri tabanı Yönetim Sistemleri, İnternet üzerinden erişilebilen ve yapılandırılabilen veri tabanı sistemleridir. Ortaya çıkışındaki temel etkenler: Farklı sunucu ve istemci mimarilerinde çalışabilecek yapıya sahip veri tabanı uygulamalarına duyulan ihtiyaç, Uzak bilgisayarlarda yer alan veri tabanı sistemlerini yapılandırma isteği (Web barındırma hizmetleri vb.) şeklinde sıralanabilir.

10.1 Web İle İlgili Temel Kavramlar

Internet: dünyadaki tüm bilgisayarların bir ağ üzerinde çalışmasıdır. İnternet terimi, international networks kelimelerinin birleşiminden oluşmaktadır.

Web (www): HTTP olarak bilinen iletişim standardını kullanarak bir ağ üzerindeki bilgisayarların bir birleriyle iletişimini sağlayan bilgisayar ağıdır.

Web bilgisi: Web sayfaları olarak adlandırılan dokümanlarda kayıtlı bilgidir.

Web server(sunucu): internet üzerinde lokal olarak depolanan bilgilerin paylaşım servisini sağlayan bir yazılım. Web sunucu örnekleri- Internet Information Server (IIS), Apache vb.

Web Client (web istemciler): bir web tarayıcı kullanarak web sayfasını okuyan internet üzerindeki bilgisayarlar.

Web Browser (tarayıcı): İnternet üzerindeki bilgileri görmemizi sağlayan bir yazılım yada web sayfalarını gösteren program. Web tarayıcı, bilgiyi görüntülemek (sunmak) için HTML de yazılmış bir dosyayı web serverdan talep eden yazılımdır. www üzerinde dolaşmaya yarayan ve bilgi transferine imkan sağlayan yazılımlara verilen ortak ad. Örnek: İnternet Explorer vb.

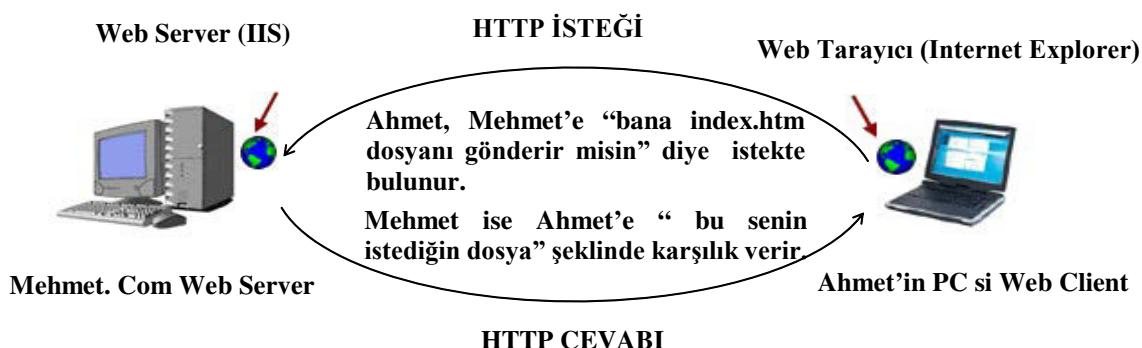
HTTP (Hyper Text Transfer Protocol): İnternet tarayıcıları web sunucuları üzerindeki bilgileri görüntüleyebilmek için çeşitli protokollere ihtiyaç duyar. HTTP, interneti kullanmamızı sağlayan ve web kurallarını belirleyen protokole verilen isimdir. HTTP, Web üzerinde metin, resim, ses, video gibi dosyaların karşılıklı alış verışı için kuralları ifade eder. Web tarayıcısı web sayfaları için HTTP isteğini gönderir. Web sunucuda HTTP cevabını geri web tarayıcıya gönderir.

Tablo 1 Web İçin Gerekli Olan Öğeler

Webe bağlanmak için	Internet servis sağlayıcısına bağlantı sağlayacak bir telekomünikasyon bağlantısı (Modem/Ethernet/Wireless) olmalıdır.
Web istemci programı olmalıdır.	Bir istemci programı (Internet Explorer, Netscape Navigator).
Web sunucu programı olmalıdır.	Bir web sunucu programı (IIS, Apache vb.) ve yüklenmiş dosyalar.
Web sayfalarının olması	HTML kodlu dosyalar.

10.1.1 Internet'in Yapısı ve İstemci ve Sunucu Yaklaşımı

Bugün, bilgisayarlar çoğunlukla artık tek başına değil Ağlar halinde kullanılmaktadır. Bir iletişim Ağrı ile birbirine bağlanmış birden çok bilgisayar birlikte kullanılmaktadır. Bütün işleri tek bir büyük bilgisayarın yaptığı “merkezi işlemin aksine, “dağıtık işlem” şeklinde dağıtilır. Yaygın olarak kullanılan bir “dağıtık işlem” şekli istemci/sunucu yaklaşımıdır. İstemci/Sunucu yaklaşımında işlemler “istemciler” ve “sunucular” arasında paylaştırılır. Günümüzde bilgi sistemlerinin tasarımları istemci/sunucu yapısındadır. Internet de yapısı itibariyle bir istemci/sunucu şeklidir. Yerel PC tarayıcıları, sunuculardan alınan HTML sayfalarını ve Java uygulamalarını işlemektedir. İstemci/sunucu modeli, istemcilerin web sunucusuna bağlı bir web tarayıcısının olduğu web merkezli bir yapıya doğru kaymaktadır. Hem istemci hem de sunucu aynı Ağdadır fakat her biri en iyi yaptığı işi yapmakla görevlidir. İstemci, normal olarak bir masaüstü ya da diz üstü PC olup, kullanıcı tarafını temsil eder. Kullanıcı genel olarak uygulamanın istemci kısmı ile muhatap olur ki, işi ya veri girmek ya da veri almaktır. Sunucu ise istemciye çeşitli hizmetler sunar (Şekil 1).



Şekil 1 İstemci/Sunucu Yapısında HTTP İstek ve Cevabı

10.1.1.1 Web Hosting Tanımı

Hosting yani türkçe karşılığı adıyla barındırma, internet sayfalarını oluşturan uygulamaların elektronik ortamda iletilebilmesi (internet'te yayınlanabilmesi) için gerekli olan alanın sunulması hizmetidir. Diğer bir ifadeyle hosting, bir web sitesinde yayınlanmak istenen sayfaların, resimlerin veya dokümanların internet kullanıcıları tarafından erişilebilecek bir server/bilgisayarda tutulmasıdır.

Bir web sitesi kurmak için, dosyalarınızı saklayacağınız bilgisayar ev veya iş yerlerimizde kullandığımız kişisel bilgisayarlarımız olamaz, çünkü gerekli program ve donanıma sahip olunsa bile bilgisayarın internet bağlantı hızı bu iş için genelde yetersiz kalacaktır. İnternette site yayınlamak için dosyaların, bu iş için özel olarak üretilmiş, hızlı internet bağlantısı olan bir bilgisayarda saklanması gereklidir. Web sitesine ait dosyalar için depo vazifesi gören ve bu dosyaları internet kullanıcılarının erişimine sunan, gelişmiş kapasiteli ve güçlü bilgisayar ağına web sunucusu (web server), web sitelerinin ihtiyaç duydukları bu veri saklama ve yayılama işlemini, 7 gün 24 saat kesintisiz olarak müşterilere kiralanması hizmetine ise de web hosting denir. Bu geniş kapasiteli sistemin belirli bir bölümünü belirli bir süreyle (örneğin 3 aylık, 2 yıllık gibi) internette web sitesini yayınlamak isteyen kişi yada kurumlara kiralayan şirktlere de web hosting hizmet sağlayıcısı denir.

10.1.2 Sunucu (Server) Bilgisayarlar

Sunucu bilgisayarlar özel olarak bir Ağ için kullanıcıların dosya, yazılım ve cihazları ya da diğer network kaynaklarını paylaşmalarına izin verecek şekilde optimize edilmişlerdir. Sunucular paylaşılan verileri saklar ve işler ve ayrıca kullanıcıların görmediği ağ faaliyetlerinin yönetimi gibi arka plan işlevlerini yerine getirir. Sunucu makinesi bir mainframe ya da masaüstü bilgisayar olabilir fakat genellikle daha gelişmiş özelliklere sahiptir. Büyük hafızaları ve disk saklama kapasiteleri, yüksek hızlı iletişim yetenekleri ve güçlü CPU'ları mevcuttur. Sunucular e-ticaret için de donanım platformu sağladığından, firmanın bilişim altyapısında önemli bir yere sahiptir. Büyük çaplı e-ticaret işletmeleri Web sitelerini IBM gibi şirketler tarafından üretilen ve birçok sunucudan oluşan sunucu merkezlerinde barındırmaktadır. Özel yazılımlar ile, satın alma ve satış etkileşimleri, şirket içi veri传递 ve Web sayfası barındırma gibi işleri yerine getirebilirler. Amazon.com ve Dell.com gibi büyük web siteleri de milyonlarca veriyi işlemek için bu tür sunucuları kullanmaktadır.

10.1.3 Web Tarayıcılar (Web Browsers)

Web sayfalarını görüntüleyebilmek için web tarayıcısına ihtiyaç vardır. Web tarayıcıları web sunucusuna ulaşarak bilgiyi alabilmemizi sağlayan bir programlardır. Örnek: Internet Explorer, Mozilla Firefox, Google Chrome, Safari, Opera. Web tarayıcıları Web sayfalarını görüntüleme, Web'de gezinme, ve diğer Web kaynaklarına ulaşmak için kullanılan kolay ve kullanışlı yazılım araçlarıdır. Ayrıca, yeni versiyonlarda kullanıcı bu paketler ile doğrudan kendi web sitesini hazırlayabilir.

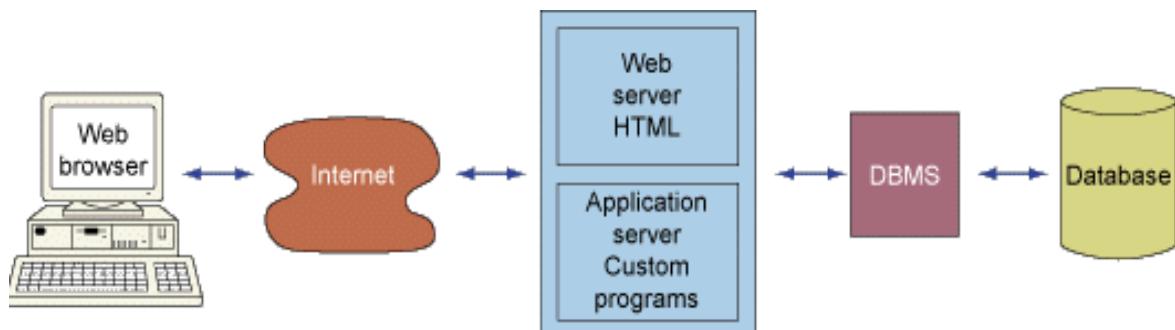
10.1.4 Web Üzerinde Veritabanı

Veri tabanı teknolojisi bilgi kaynaklarının WWW'de bulunabilirliğini sağlamada önemli bir rol oynar. Web siteleri **hypermedia** veri tabanlarını kullanarak bilgileri metin, ses, video ve grafik içeren ve birbirine bağlı bir dizi sayfalar halinde saklar. **Hypermedia** veri tabanı yaklaşımında, bilgi parçaları **linklerle** birbirine bağlı **node'lar** şeklinde saklanır. Kullanıcılar web sitelerindeki istedikleri bilgilere **istedikleri sırada** erişebilirler.

Hypermedia Veri tabanı: Kullanıcı bir node'dan diğerine kendi yolunu çizebilir. Her bir node metin, grafik, ses, video ve uygulama programları içerebilir.

10.1.5 Veri Tabanlarına Web'de Bağlanma

Kullanıcıların web üzerinden işletme verilerine erişmelerini sağlamak için bir dizi middleware ve yazılım ürünleri geliştirilmiştir. Örneğin, web'de bir müşteri bir işletmenin ürünlerinin fiyatları için on-line bilgi almak isteyebilir. Şekildeki gibi bir düzenek ile müşteri işletmenin içsel veri tabanına web'den erişerek istediği bilgilere erişebilir (Şekil 2).



Şekil 2: Veri Tabanlarına Web'de Bağlanma

Çalışma Biçimi: Kullanıcının Web tarayıcı yazılımı organizasyonun web sunucusu ile HTML komutları vasıtası ile iletişim kurarak veri tabanından gerekli veriyi ister. Çoğu veri tabanı programları HTML komutlarını algılayamadığından, Web sunucu bu istekleri HTML kodunu SQL'e çevirebilen özel bir yazılıma aktarır. Veri tabanı, SQL komutlarını alınca istenen bilgileri temin eder. İstenen bilgiler organizasyonun içsel veri tabanından tekrar geriye web sunucuya transfer edilir. Oradan da bilgiler müşterinin web tarayıcısına iletilir. Web sunucu ile Veri tabanı arasında çalışan yazılım ya bir uygulama sunucusu yada özel bir programdır. Uygulama sunucusu bir organizasyonun tarayıcıya dayalı bilgisayarları ile veri tabanları arasındaki işlemleri ve veri akışını kontrol eden bir yazılım programıdır. "Uygulama sunucusu" istekleri Web sunucudan alır, bu isteklere göre programı yürütür ve organizasyonun veri tabanına bu istekleri ileterek cevap alır.

10.2 Web Teknolojileri ve Geliştirme Araçları

Web projesi geliştirmede HTML, XML, XHTML, Java gibi Web geliştirme dilleri (İşaretleme dilleri~Markup Languages) ve PHP, ASP, JAVA SCRIPT, JAVA APPLET ve FLASH gibi çeşitli Destekleyici Programlardan yararlanılır.

10.2.1 HTML

HTML (HyperText Markup Language / Hareketli-Metin İşaretleme Dili) basitçe, tarayıcılarla görebileceğimiz, internet dökümanları oluşturmaya yarayan bir çeşit dildir. HTML metinleri ya da verileri biçimlendirmek, düzenlemek için kullanılan komutlar dizisidir. HTML, Web sayfası oluşturmak için kullanılan bir sayfa tanımlama dilidir. Metin, şekil ve bağlantıların (link) sayfada nasıl görüneceğini belirlemek için “tag” (etiket) adı verilen komutları kullanır.

HTML dili kendi başına kullanılabileceği gibi, artık otomatik olarak HTML kodu üreten kullanımı kolay yazılımlar da mevcuttur. HTML editorleri Web sayfası oluşturmada oldukça güçlü programlardır. Web bilgileri web sayfası denen dokümanlarda saklanır. Yazdığınız kodların tarayıcı tarafından alınıp yorumlanabilmesi için, dosyalarınızın uzantısının ".htm" veya ".html" olması gereklidir.

HTML dosyasının Head ve Body Bölümleri

Head (Baş) Bölümü

```
<html>
<head>
<title>İlk Sayfam</title>

<meta name="description" content="anahtar kelimeler">

</head>
```

Body (Gövde) Bölümü

```
-----
<body>
Sayfama Hoşgeldiniz <br>

<font face="tahoma" size="5" Ilkbahar</font>

<a href="mailto: yasemin@aku.edu.tr">mail atın</a>



</body>

</html>
```

10.2.2 XML, Extensible Markup Language

XML, web dokümanlarının yararlılığını artırmak için tasarlanmış yeni bir spesifikasyondur.

Aslında XML, HTML'nin daha geliştirilmiş halidir. HTML sadece metin ve şekillerin web sayfasında nasıl görüneceğini belirler iken, XML bu sayfalardaki verilerin ne anlama geldiğini de tanımlar. XML web sayfasındaki bilgilerin program içinde kullanılabilmesini sağlar. XML'de, bir sayı sadece bir sayı olmayıp, XML etiketi (tag) bu sayının bir fiyatı mı, bir tarihi mi, yada bir ZIP kodunu mu temsil ettiğini belirtir. Örneğin, bir elbisenin fiyatını web sayfasında koyu yazdırma için gerekli HTML etiketi `b.$39,/b.` şeklindedir. Aynı işi yapan bir XML etiketi bu 39 doların fiyatını gösterdiğini belirtmek için `price.$39,price` ifadesini kullanır.

10.2.3 Web Editörleri (Düzenleyiciler)

Bu araçlar HTML sayfası hazırlanırken ihtiyaç duyulan kodlama işlemini kolaylaştıran programlardır. Bu tip programlar ile site içinde yer olması istenen resimleri, yazıları ve bağlantıları istediğiniz biçimde bir Web Tarayıcı tarafından izlenecek şekilde tasarlayaharak biraraya getirebiliriz.

Piyasadaki bazı düzenleyiciler;

- Macromedia Dreamweaver
- Microsoft Frontpage
- Allaire Homesite,
- HotDog Pro
- Netscape Composer
- Adobe PageMill, gibi birçok program alternatif bulunuyor.

Bu tür birçok web editörü size HTML dilini bilmeden, HTML kodlarını kullanarak web sitesi yapma imkânı veriyor. Bu tip editörler sitenizde yaptığınız her adımı, otomatik olarak HTML koduna çeviriyor.

10.2.4 Web Tabanlı Veri tabanı Yönetim Sistemleri

Günümüzde yaygın olarak kullanılan popüler Web tabanlı veri tabanı sistemleri:

- MySQL – www.mysql.com
- PostgreSQL - www.postgresql.org
- MS SQL - www.microsoft.com/sql/default.mspx

- Oracle – www.oracle.com

Bu sistemler içerisinde PHP ile en uyumlu çalışan ve yüksek performans gösteren iki veri tabanı sistemi bulunmaktadır:

- MySQL
- PostgreSQL

Gelişirilen uygulamalara ve yaygın kullanımına bakıldığındá ise MySQL tercih olarak ön plana çıkmaktadır.

10.3 PHP-MYSQL-APACHE Sunucu Üçlüsü

Bugün Apache, MySQL, PHP üçlüsü Internet dünyasında vazgeçilmez bir paket haline geldi.

Bu derste PHP-MySQL-Apache üçlüsü ile Web geliştirme konusu üzerinde durulacak. Web tasarımda ve uygulama geliştirilmesinde kullanılan teknolojiler açık kaynak kodlu yazılım teknolojileridir. Bunlar istemci/sunucu (client/server) tipi uygulamalar geliştirmek için tasarlanmışlardır.

Açık kaynak kodlu yazılım teknolojileri maliyet bakımından oldukça avantajlıdır.

Kullanılan teknolojiler aşağıdaki şekilde dir;

- PHP Uygulama sunucusu teknolojisi
- Apache HTTP Sunucusu
- MySQL veri tabanı sunucusu
- SQL-(Structured Query Language [Yapısal Sorgu Dili])
- Macromedia Dreamweaver web(html) editörü ve uygulama geliştirme aracı.

10.3.1 Apache Web Sunucusu

Apache, [açık kaynak](#) kodlu bir web [sunucusu](#) programıdır. World Wide Web'in genişlemesinde ve yayılmasında anahtar rol oynamıştır. 1996'dan bugüne Apache İnternet'teki en yaygın web sunucusu olmuştur.

10.3.2 PHP

PHP ile her türlü işlevselliğe ait program yazılabilir. PHP web sunucuya bir takım işler yaptmak için kullanılan program yazma dilidir. PHP kodları, HTML sayfaları arasında HTML etiketleri arasında kendi özel ayıraçları arasında yazılır.

```
<html>  
    <?php  
        echo("merhaba");  
    ?>  
</html>
```

Ekrana 'merhaba' şeklinde yazı gelir

PHP, "Hypertext Önİşlemci" (PHP:Hypertext Preprocessor), sözcüklerinin baş harfleriyle temsil edilen HTML içine gömülebilir açık-kodlu bir betik dilidir. Sunucu taraflı bir teknolojidir. Dil yapısının önemli bir kısmını C, Java ve Perl gibi dillerden almış, kendisine has özelliklerle bu yapıyı pekiştirmiştir. Dilin ana amacı, web geliştiricilerinin dinamik sayfalar yapmasını yani sayfa içeriklerini dinamik hale getirmesini sağlamaktır.

PHP hemen hemen her platformda çalışabiliyor. (PHP aynı kod temelini kullandığı için, UNIX ve Windows gibi 25 platformda derlenip kurulabilir. Kodlar aynı olduğundan script'ler platformdan bağımsız olarak çalışacaktır)

PHP pek çok veri tabanı arayüzü bulunduruyor. PHP, MySQL, MSSQL, Oracle, Informix, PostgreSQL ve diğerleriyle doğrudan çalışabiliyor.

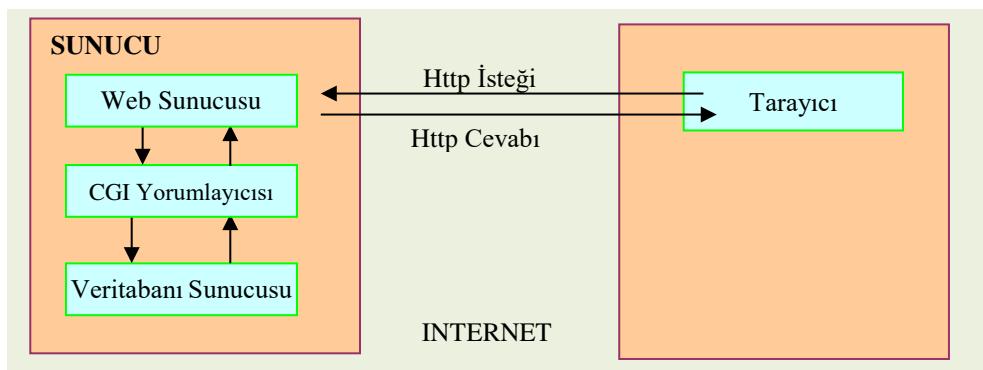
Bir PHP kullanıcısı herhangi bir kütüphane için arayüz oluşturmakta zorluk çekmez. Pek çok kullanıcı bu yolu seçmiş, grafik rutinleri, PDF dosyaları, Flash Movie'leri, Cybercash cetvelleri, XML, IMAP, POP ve diğerleriyle ilgili modüller bulabilmiştir.

PHP bir açık kod uygulamasıdır ve pek çok profesyonel kullanıcı için çok şey ifade etmektedir. Basitçe açıklamaya çalışırsak PHP kullanıcısı, çalışmayan uygulamalar için üretici firmayı keyfini beklemekten, her yıl sistemini belli paralar ödeyerek güncelleme zorunluluğundan kurtarmaktadır.

PHP Çalışma Şekli: Web Sunucular htm ve html uzantılı dosyalarını sabit diskinde bulundurur ve ziyaretçiye sunar, ama PHP uzantılı dosyalar da ise PHP derleyicilerin çağrılması gereklidir. PHP derleyiciler ise kendine ulaşan metin dosyayı içinde <? veya <?PHP Kelimelerine rastlayana kadar tüm HTML kodunu web sunucusuna aynen gönderiri. <?PHP ve ?> arasındaki PHP kodlarını seçerek alır ve gereği neyse onu yapar. Bu ayıraçların içinde kalan kodlar, bizim yapılmasını istediğimiz işlemin komutlarıdır.

HTML ile bir web sunucusundaki bir veri tabanı dosyasını açıp okuyamayız, düz yazı dosyalarını da okuyamayız ve bu disklere dosya yazamayız. PHP de “**<?PHP ve ?>**” ayıraçları içindeki kodları icra eder ziyaretçi bu kodların içini göremez ve okuyamaz.

PHP bir CGI programlama dilidir. Bu dille web sunucusu ile web ziyaretçisi arasında buluşma noktası olan CGI ‘da bilgi alış veriş yapar. Sunucuda bulunan başka programlar çalıştırılabilir ve web sayfamıza HTML sınırlamasının dışında hareket ve ziyaretçi arasında bilgi alışverişi sağlar (Şekil 3).



Şekil 3 CGI Yapısı

Tarayıcı tarafından kullanıcı sunucuya istediği sayfanın url adresini yazarak yada link’ini tıklayarak belirtir. Sunucu bu talebi alınca x.php dosyasını php işlemcisince işlemesi gerektiğini bildiği için bu dosyayı bulur ve php işlemcisine gönderir. php işlemcisi bu dosyayı işler ve çıktıyı HTML haline getirir. Bu HTML dosyası istemciye geri gönderilerek talep ettiği web sayfasının görünmesi sağlanır. Diğer bir anlatımla Apache web sunucu php dosyasını php yorumlayıcısına gönderir. Php yorumlayıcısı kodları yorumlar ve saf HTML koduna çevirip kullanıcının tarayıcısına gönderir. Dosyanın uzantısı.php bile olsa kullanıcıya html komutları gelir. Tüm internet programlama dilleri bu mantıkla çalışır.

10.3.3 MYSQL

MySQL veri tabanı yönetim sistemi, yüksek performans, yüksek güvenilirlik ve kullanım kolaylığı nedeniyle dünyanın en popüler açık kaynak kodlu veri tabanı yönetim sistemi haline gelmiştir. MySQL 20’den fazla platform üzerinde çalışabilmektedir. MySQL’dir.

En önemli özellikler:

- Açık kaynak kodludur. (Open Source)
- Ücretsizdir.
- Tüm platformlarda (Windows/Linux/Unix) sorunsuzca çalışır. Bir ilişkisel veri tabanı yönetim sistemidir.
- Tüm verileri tek bir ambara yığmak yerine farklı tablolarda ve veri tabanlarında düzenli bir biçimde saklar.
- Veri tabanlarına erişmek için SQL standart dil ile işlemler yapıyor.
- MySQL AB isimli (eski ismi TCX) bir İsveç firması tarafından geliştiriliyor.
- Apache ve PHP ile beraber web-veri tabanı uygulamalarında çok yaygın olarak kullanılır. Apache-PHP-MySQL üçlüsü için hazırlanmış çok geniş bir yazılım yelpazesi bulunuyor.
- Birden fazla CPU ile kolaylıkla çalışabiliyor.
- 60000'in üzerinde tablo, 5 milyarın üzerinde satır ile çalıştığı söylenen MySQL sistemler bulunuyor.
- Tabloların kontrolü, optimizasyonu ve tamiri hızlı bir biçimde yapılabiliyor.
- Farklı karakter setlerini (iso8859-9,...) ve onlara göre sıralama yapılmasını destekliyor, farklı dillerde hata mesajları verebiliyor.
- Özellikle internet ortamında önem kazanan, çok esnek ve güçlü bir kullanıcı erişim kısıtlama/yetkilendirme sisteme sahip.

10.4 Web Uygulamaları

Dinamik web sayfalarının ve genellikle bir veri tabanının kullanıldığı sistemlere web uygulamaları denir. Başta arama motorları olmak üzere, forum, elektronik ticaret, eğlence, belge yönetimi, kütüphane hizmetleri vs. gibi birçok site web uygulaması şeklinde dizayn edilirler.

Arayüz olarak web sitelerinin kullanıldığı uygulamalar web uygulaması olarak adlandırılır. Yaygın olarak kullanılan arayüzler php, asp, cgi uygulaması, java ya da.net kullanılarak tasarılanırlar. Temelde yaptıkları iş bakımından birbirlerinden farklı olmasalar bile php'nin diğer programlama dillerine göre daha fazla kullanıldığını söylemek yanlış olmayacağından emin olabiliriz.

Web uygulamalarından önce server-client yazılımlarının kullanıldığı sistemler bulunuyordu. Bir uygulamanın kullanılması, sunucu üzerinde çalışan bir programa, istemci bilgisayara yüklenen özel bir program sayesinde ulaşılması şeklinde oluyordu. Ancak web uygulamaları ile sadece sunucu taraflı bir uygulama ile internet ya da intranet üzerinden tarayıcı aracılığıyla buna ulaşan bir istemci yeterli olmaktadır. İstemci bilgisayar üzerinden yapılan her istek, sunucu tarafından yorumlanır, sonuç html kodları içeren dinamik sayfa yardımıyla tarayıcı'a gönderilir.

10.4.1 Web Uygulamalarının Çalışma Biçimi

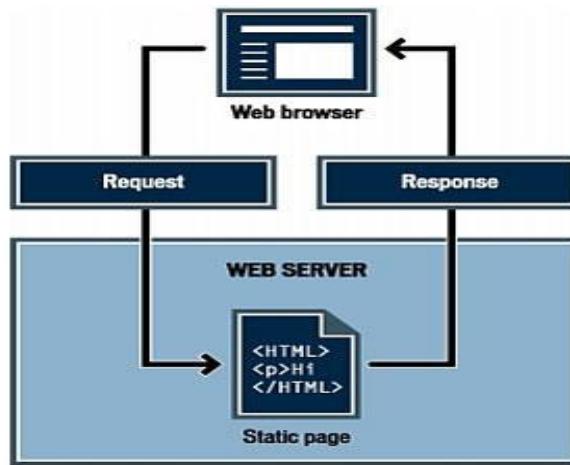
Bir web uygulaması, dinamik ve statik sayfaların bir derlemesidir. Statik sayfalar kullanıcının isteğine göre değişmezler; sunucu web sayfası üzerinde herhangi bir değişiklik yapmadan sayfayı kullanıcının kullandığı tarayıcıya gönderir. Dinamik sayfalar ise statik sayfaların tam tersine tarayıcı tarafından yapılan isteğe uygun olarak sunucu tarafından üretilen web sayfalarıdır (Tablo 1). Bu sayfalara dinamik denmesinin sebebi de bu özellikleridir.

Tablo 1 Tarayıcı-Sunucu İlişkisi: Dinamik ve Statik HTML Sayfaları

STATİK HTML SAYFALARI	DİNAMİK HTML SAYFALARI
Kullanıcı statik sayfa olan linke tıklar.	Kullanıcı tarayıcıdan bir linke tıklar. (HTTP ile tarayıcıdan sunucuya statik bir web sayfası için istek gönderilir)
Sunucu disk üzerinde (stocks.php) sayfasını bulur.	Sunucu diskte istenilen sayfayı bulur ve istemciye gönderir.
Web sunucusu sayfayı CGI yorumlayıcısına (php) gönderir.	Tarayıcı sayfayı alır ve gösterir.
CGI-yorumlayıcısı programı çalıştırır, bu sırada program veri tabanı sunucusuna da bağlanabilir-XML/HTML sayfası oluşturulur.	Kullanıcı tarayıcıdan bir linke tıklar. (HTTP ile tarayıcıdan sunucuya statik bir web sayfası için istek gönderilir)
XML/HTML dosyası web sunucusuna gönderilir.	Sunucu diskte istenilen sayfayı bulur ve istemciye gönderir.
Web sunucusu dosyayı tarayıcıya gönderir.	

10.4.1.1 Hazır Statik Sayfaların Web Sunucudan İstenmesi Durumu

Statik bir web sitesi, içinde bir takım html sayfalarını barındıran ve web sunucu hizmeti bulunan bir sunucu bilgisayar üzerinde bulunur. Web sunucu hizmeti, tarayıcıdan yapılan istek üzerine ilgili sayfaları gösteren bir yazılımdır. Tasarımcı tarafından hazırlanan sayfaların, kullanıcının yaptığı istek üzerine değiştirilmesi söz konusu değildir. Sayfa üzerinde çeşitli flash (swf) veya gif animasyonlarının olması teknik açıdan sayfanın statik olduğu sonucunu değiştirmez.

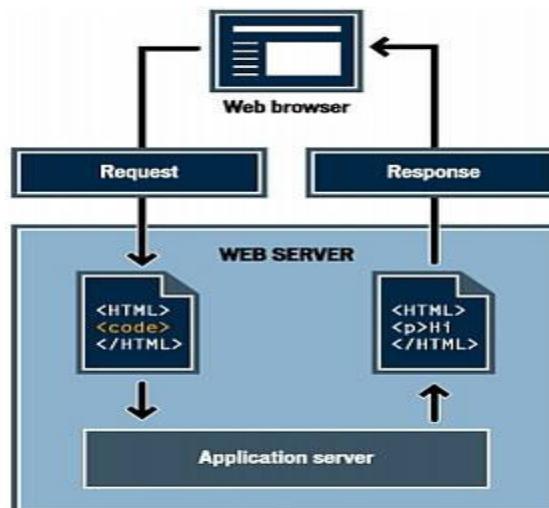


Şekil 5

Yukarıdaki şekilde statik bir web sayfası isteğinde bulunan web tarayıcıya, web sunucu hizmeti tarafından ne şekilde yanıt verildiği gösterilmiştir (Şekil 5). Web tarayıcı tarafından istenen statik sayfa, sunucu tarafından bulunur ve istemciye gönderilir.

10.4.1.2 Gelen Dinamik İsteği İşleyen Uygulama Sunucusunun Olduğu Durum

Statik bir sayfa isteğinde bulunan web tarayıcıya ilgili sayfa doğrudan gönderilir. Web sunucusuna dinamik bir sayfa isteği gelmişse durum bundan daha karmaşıktır. İstek yapılan sayfanın gönderilebilmesi için isteği işlemekten sorumlu özel yazılımın bu sayfayı üretmesi gereklidir. Burada bahsi geçen özel yazılım uygulama sunucusudur (application server) (Şekil 6). Uygulama sunucu dinamik sayfa üzerindeki kodları okur, buradaki emirleri yerine getirir ve en sonunda bu sayfa içerisinde bulunan kodları kaldırır. Sonuçta karşı tarafa gönderilen bir statik sayfadır. Dolayısıyla istekte bulunan tarayıcı sadece HTML kodlarını görebilir. (Web tarayıcılar bilindiği gibi sadece HTML kodlarını yorumlayabilirler.)



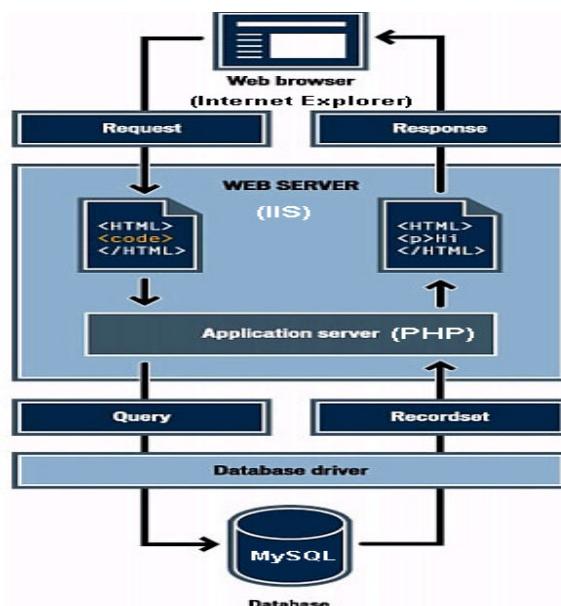
Şekil 6 Gelen Dinamik İsteği İşleyen Uygulama Sunucusu (Application Server)

10.4.1.3 Gelen Dinamik İsteği İşleyen Uygulama Sunucusunun Veri Tabanı Sunucusuyla Birlikte Çalıştığı Durum

Bir uygulama sunucusu, sunucu taraflı (server-side) kaynaklarla çalışır. Örneğin, dinamik bir sayfa uygulama sunucusuna, veri tabanındaki bilgileri buradan alıp HTML belgesi içeresine yazma komutu verebilir. Veri tabanındaki bilgileri alma emri veri tabanı sorgusu olarak adlandırılır. Bir sorgu SQL olarak adlandırılan veri tabanı arama kriterlerinden oluşur. SOL sorgusu sayfaya sunucu taraflı scriptler ya da taglar şeklinde yazılır.

Uygulama sunucusu veri tabanına doğrudan erişemez. Çünkü farklı üreticilere ait veri tabanlarının aynı yolla sorgulanması mümkün değildir. Bunu bir word belgesinin not defteri uygulamasında açılamamasına benzetebiliriz. Uygulama sunucusunun veri tabanına erişimi bir aracı yazılım ile mümkün olabilir. Bu aracı yazılım veri tabanı sürücüsü olarak adlandırılır. Veri tabanı sürücüsü veri tabanı ve uygulama sunucusu arasında yorumlayıcı vazifesi görür. Sürücü bağlantıyi sağladiktan sonra, veri tabanı üzerinde çalıştırılan sorgu bir kayıt dizisi (recordset) oluşturur. Kayıt dizisi (recordset), veri tabanı içinde bulunan bir ya da daha fazla tablodan çıkarılmış dataların serisidir. Kayıt dizisi uygulama sunucusuna sayfayı oluşturabilmesi için gönderilir.

SQL için basit bir sorgu aşağıdaki gibidir; SELECT soyad, ad, yas FROM calisanlar



Şekil 7 Sistemin Teknolojik Açıdan Genel Akış Şeması

Yukarıdaki anlatım üç sütundan oluşan bir kayıt dizisi oluşturur. Bu üç sütundaki her satır soyad, ad ve yas alanlarında bulunan kayıtları içerir. Veri tabanı bağlantısı bulunan bir web uygulamasının çalışma şekli, Şekil 7'de gösterilmektedir. Bir web uygulaması için hemen hemen bilinen tüm veri tabanları kullanılabilir. Daha açık bir ifadeyle veri tabanı sürücüsü bulunan tüm veri tabanı yazılımlarını kullanmak mümkündür. Güçlü web uygulamaları için sağlam bir veri tabanı planlaması yapılmalıdır. Bu tür bir plan içerisinde Microsoft SOL Server,

Oracle yada MySOL veri tabanı seçenekleri mutlaka göz önünde bulundurulmalıdır. Web uygulaması ile veri tabanının aynı sunucuda bulunması gibi bir şart olmamakla birlikte, eğer böyle bir yönteme başvurulacaksa iki sunucu arasında hızlı bir bağlantının olmasına dikkat edilmelidir.

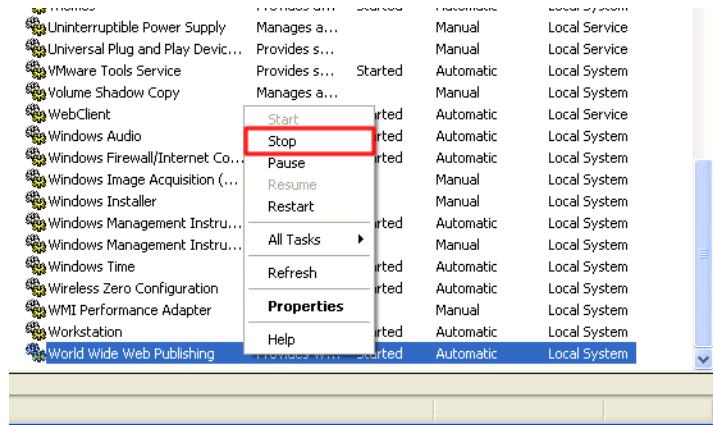
Şekil 7'de görüldüğü gibi kullanıcı web tarayıcı sistemimizde Internet Explorer aracılığı ile bir istek (request) yapar. Bu istek HTML kodları ile web sunucuya (web sunucusu- sistemimizde Apache) gönderilir. Daha sonra web sunucu bu isteği kendisine bağlı bulunan Uygulama Sunucusu'na (Application Server- sistemimizde PHP) yollar ve isteği bundan sonra uygulama sunucusu işlemeye devam eder. İstek öncelikle SQL sorguları yoluyla Veri tabanı Sürücüsüne (Database Driver-Uygulama sunucusundan gelen istekleri SQL Veri tabanı sunucusu diline çevirerek arada bir çevirmen rolü üstlenen bir yazılım) aktarılır. Veri tabanı sürücüsü uygulama sunucusunun istediği bilgileri veri tabanından elde etmek için gelen komutları veri tabanına özel komutlara çevirir ve bunları veri tabanı sunucusuna yollar. Daha sonra veri tabanı istenilen işlemleri yaparak istenilen bilgileri bulur ve bunları veri tabanı sürücüsüne iletir. Veri tabanı sürücüsü bu kez de önce yaptığı işlemin tersini yapar ve veri tabanından gelen verileri uygulama sunucusunun anlayacağı dile çevirir ve daha sonra bir kayıt seti oluşturur. Bu kayıt setinin içerisinde veri tabanından gelen ve sadece sorguda istenilen veriler bulunur. Böylece istenilen Sorgu (Query) nun Veri tabanı Sürücüsü(Database Driver) tarafından Recordset (Kayıt Seti) haline getirilmesi işlemi tamamlanmış olur. Daha sonra bu kayıt seti uygulama sunucusuna ilettilir ve uygulama sunucusu da bunları web sunucusuna iletir. Web sunucusu da bu ham verileri düzenleyip, biçimlendirerek HTML kodlarına dönüştürür ve kullanıcının web tarayıcısına response (yanıt) olarak iletir. Kullanıcı istediği bilgiye artık ulaşmıştır. Böylece sistem bir işlemi tamamlamış olur.

10.5 Web Uygulaması Geliştirme

10.5.1 APACHE-PHP-MYSQL Kurulumu

PHP web geliştiricileri genel olarak veritabanı tercihlerini daha çok MySQL'den yana kullanırlar. Özellikle PHP-MySQL-Apache üçlüsü web sitelerinde performans isteyenler için tercih edilir. Web programlamaya yeni başlayanlar için Apache-PHP-Mysql kurulumu hakkında kısa bir bilgi vermek gerekirse. Kurulum için 2 alternatifiniz var; her programı ayrı ayrı kurup ayarlarını elle yapmak ya da hazır kurulum programlarını kullanarak tek adımda tüm kurulumu gerçekleştirmek. Yeni başlayanlar büyük ihtimalle hazır kurulum programlarını tercih edecektir (Tavsiye edilir). Ama bu üçluğun kendi aralarındaki ilişkiyi anlamak ve deneyim kazanmak adına en az bir kez, her birini ayrı ayrı kurup ayarlarını elle yapmanızda fayda var. Eğer Windows XP Professional kullanıyorsanız IIS (Internet Information Server) adlı diğer bir web sunucu yazılımı makinanızda çalışıyo olabilir. Apache'yi kurabilmek IIS'yi kapatmanız gerekmekte.

Bunun için Denetim Masası penceresine gidip oradan Administrative Tools (Yönetimsel Araçlar) altından Services (Hizmetler) adlı simgeye çift tıklayarak bilgisayarınızda yüklü olan servisleri görüntüleyebilirsiniz. Karşınıza gelen hizmet listesinin sonlarına doğru gelip World Wide Web Publishing servisine sağ tıklayarak Stop'u seçelim. Böylelikle IIS kapanmış olacaktır.



Şekil 1. IIS Kapatılması

10.5.1.1 Apache Php Mysql Kurulumu (Hazır Programlar İle)

1. WAMP
2. EasyPHP
3. PhpTriad
4. Apache2Triad
5. XAMPP
6. NuSphere Tech Platform
7. AppServ
8. Uniform Server

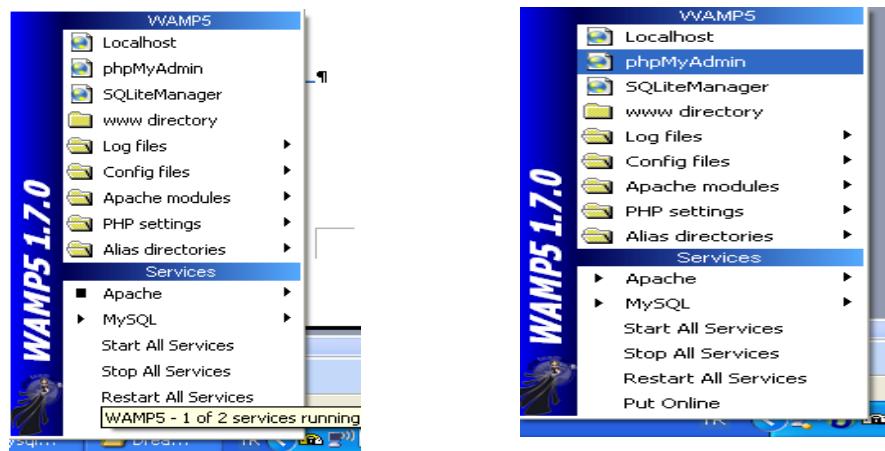
Biz wamp5_1.7.0.exe dosyasını çalıştırarak bilgisayarımıza PHP-MySQL-Apache üçlüsünü kuracağız. Bu işlem WampServer kurulum Sihirbazı takip edilerek kolayca kurulur.



Şekil 2 Wamp'in Kurulumu

Wamp5_1.7.0.exe programını kurduktan sonra Başlat > Tüm Programlar > WampServer çalıştırılır.

Çalıştırıldıktan sonra her iki sunucu’inde çalışır olması gereklidir. Bazı durumlarda yalnızca MySql çalışmamaktadır. Apache sunucu çalıştırılmak için Yönetimsel araçlar>hizmetlerden IIS’in durdurulması gerekmektedir. Bu yapıldığında Apache sunucusu de çalışacaktır. Ve simge beyaz görünümde olacaktır.



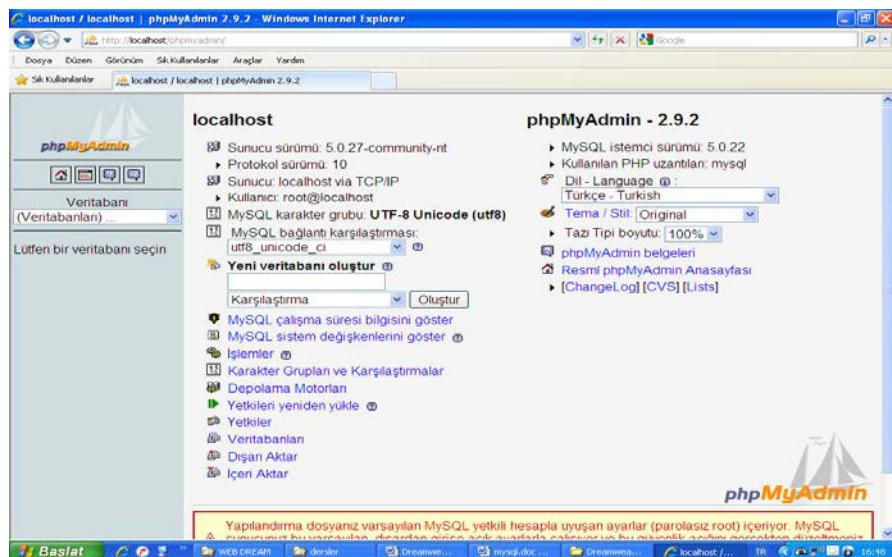
Şekil 4. Phpmymadminin Çalıştırılması

10.5.1.2 MySQL'de Veritabanı Oluşturma İşlemleri

Ne kadar profesyonel olursanız olun, hazır yazılımlar kullanarak yapılan MySQL yönetimi hem daha hızlı hem de daha sıkıntısız olacaktır. PHP ile geliştirilmiş olan PhpMyAdmin yazılımı kullanılarak tarayıcı penceresi üzerinden MySQL’le ilgili birçok işlemi gerçekleştirebilmek mümkündür. PhpMyAdmin’ı kullanabilmek için Apache ve MySQL çalıştırılmış olmalıdır.

PhpMyAdmin wamp sunucu simgesinin üzerine sol klik edilerek aşağıdaki gibi çalıştırılır.

PhpMyAdmin açıldığında ise aşağıdaki gibi gözükecektir.



Şekil 5. PhpMyAdmin ile Veri tabanı Oluşturma

10.5.1.3 PhpMyAdmin ile Veri tabanı Oluşturma

Yeni veritabanı oluştur kısmının altına bir veri tabanı adı girilerek veri tabanı oluşturulur. Bu örnekteki veri tabanının adı **ebuveritabani** olsun. Oluşturulması istenilen veri tabanı adı写字楼 olarak oluşturtuşuna basılsrsa veri tabanı oluşturulacaktır. Fiziksel olarak elimizde şu an bir veri tabanı olmuş olacaktır.

10.5.1.4 PhpMyAdmin ile Tablo Oluşturma

ebuveritabani veritabanında yeni tablo oluştur yazan bölümde tablo adı ve kaç alandan oluşacağı yazılarak kaç alandan oluşacaksa belirtilerek sihirbaz (wizard) mantığı ile tablo oluşturulur. Bu örnkte Tablonun adı **uyekaydi** ve 9 alandan oluşmaktadır. Oluşturulan tablo Şekil 2 de gözükmektedir. Login tablosu ve alanları ise aşağıda gözükmektedir.

	Alan	Türü	Karşılaştırma	Öznitelikler	Bos	Varsayılan	Ekstra
<input type="checkbox"/>	kullaniciadi	varchar(20)	latin1_swedish_ci		Hayır		
<input type="checkbox"/>	sifre	varchar(20)	latin1_swedish_ci		Hayır		

Buraya kadar Mysql üzerinde veri tabanı ve tablolarını oluşturduk Şimdi Dreamweaveri kurup çalıştırarak önce yeni bir sitenin oluşturulması ve siteyle ilgili tanımlamaların yapılması yapılacak ardından da oluşturduğumuz site üzerinde çeşitli dinamik uygulamalar geliştirilecek.

10.6 Dreamweaver Üzerinde Yeni Bir Site Açıma

MySQL'de Veritabanı Oluşturma İşlemlerinin ardından web arayüzlerinin oluşturulması için önce yeni bir site oluşturmamız gereklidir. Bunu; Site>New Site menüsünden öncelikle kendimize yeni bir site tanımlayalım. Eğer bu menu "Basic mod" ile açılırsa bu menüyü advanced özellikleri görebileceğimiz şekilde yukarıdaki sekmeden değiştirelim. Sırası ile sitemizi tanımlayalım.

10.6.1 Local Info

"Site name" den site ismimizi ve sitemizin local adresini belirliyoruz.

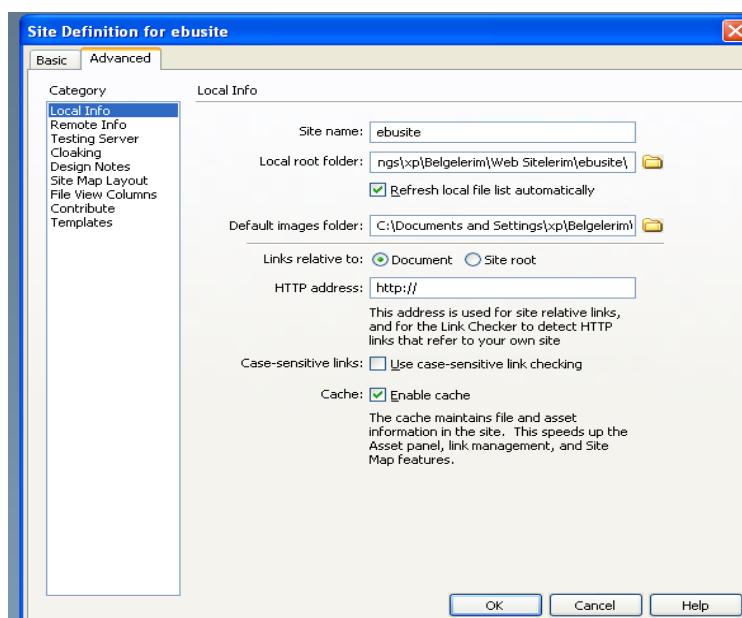
Örnek

Site Name:ebusite

Local Root Folder: C:\Documents and Settings\xp\Belgelerim\Web Sitelerim\ebusite\

Default Images Folder: C:\Documents and Settings\xp\Belgelerim\Web Sitelerim\ebusite\images\

"Default Images Folder" alanına site klasörünüz içerisindeki resimleri saklamak istediğiniz klasörü gösterebilirisiniz. bunu yaparsanız site dışından seçtiğiniz resimler otomatik olarak bu klasörün içine kopyalanacaktır. Bunu isterseniz D:\Medyasoft\images olarak belirleyin. Ancak bu ayarı yapma zorunluluğunuza olmadığını da bilmenizde yarar var.



Şekil 6 Local Info Verilerinin Girilmesi

10.6.2 Remote Info

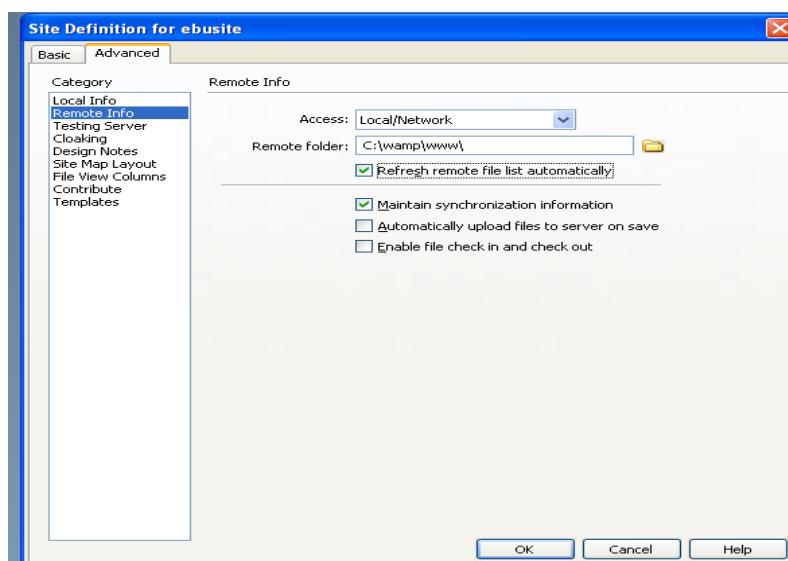
Bu menüden "Access" drop down menüsünden "Local/Network" özelliğini seçelim. Bu özelliği seçtiğimizde bizden "Remote Folder" alanının belirtilmesini isteyecektir. Yandaki klasör işaretine basarak "Remote Folder" alanımızı belirleyelim. Remote folder sunucumuzun uygulamaları yorumlayabildiği bir klasör altında olmalıdır. Örneğimizi yaptığımız bilgisayarın yorumladığı klasör altına koyuyoruz.

Örnek:

Access Local/Network

Remote Folder: C:\wamp\www\

Eğer Apache üzerinde PHP çalıştırıyorsanız Remote Folder: "C:\Apache\htdocs\Medyasoft" şeklinde olabilir. Not: burada uzak sunucuyu bulup seçmek yeterli C:\wamp\www\ nin içerisine yeni bir klosör oluşturmaya gerek yok.



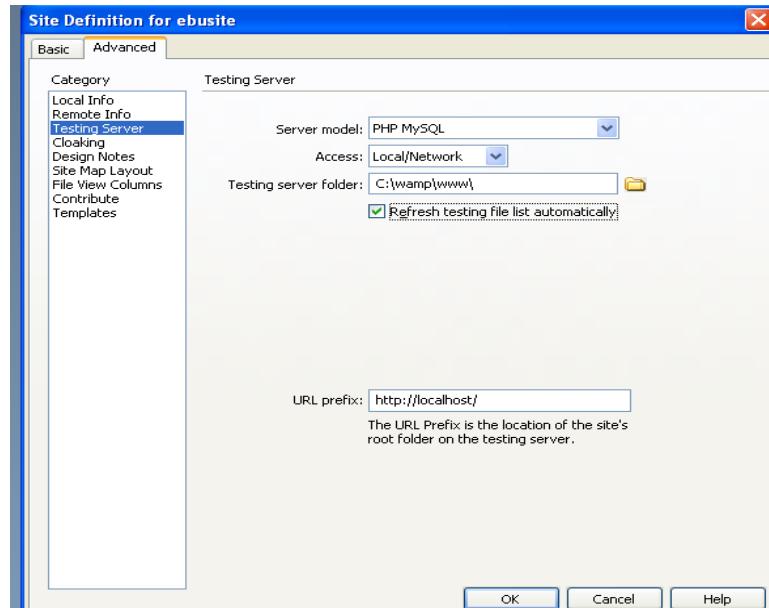
Şekil 7 Remote Info Verilerinin Girilmesi

10.6.3 Testing Server

"Server Model" açılır menüsünden PHP/MySQL'i seçiyoruz. "Access Type" alanından ise "Local/Network" seçiyoruz. Bu ayarlamaları yaptıktan sonra daha önceki ayarlamalarımıza dayanarak Dreamweaver gerekli alanları bize belirtiyor ve otomatik olarak dolduruyor.

Server Model: PHP/MySQL

Access Type: Local/Network



Şekil 8 Testing Server Verilerinin Girilmesi

Diğer kısımlarda şimdilik bir ayarlama yapmamıza gerek yok. Şimdi "Ok" tuşuna basıp sitemizin tanımlamasını bitiriyoruz.

10.7 Dreamweaver İle MYSQL Bağlantısı

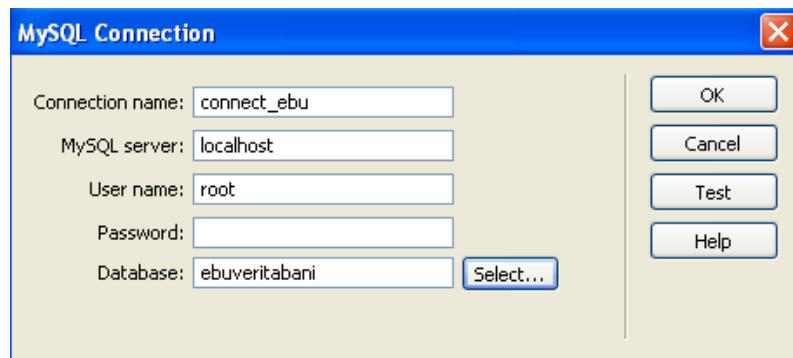
Bu kısımda Dreamweaver ile MySQL bağlantısı yapmak hakkında bilgi edineceksiniz.

Site Map menüsünden şu anda default olarak seçili sitenin az önce oluşturduğumuz site olup olmadığı kontrol ettikten sonra Mysql ile bağlantı kuracağımız "Application" menüsüne geçiyoruz. Burada öncelikle "Database" menüsünden veri tabanımız ile iletişime geçmemiz gerekiyor. Bu menü kapalı ise Windows>Database menüsünden açabilirsiniz.

Database menüsünde + işaretine basarak yeni bir database bağlantısı kuracağız. Eğer henüz "Document Type" belirlemediyseniz bu menüdeki "Document Type" alanının üstüne tıklayarak döküman tipimizi PHP olarak belirleyelim. Database menüsünde sitemiz için gerekli tanımlamaların neler olduğu ve bunlardan başarılı olanlarında yanında "Check" işaretini ile belirtilmiştir. Database menümüzde + işaretine basıyoruz ve tek bağlantı yapabileceğimiz database olan Mysql'i seçiyoruz.

10.7.1 Connection Name

Bu bizim database ile bağlantı kuracağımız bağlantının adıdır. Bu örnekte `connect_ebu` adında bir bağlantı adı verildi. Bu ada sahip bir dosya sitemizin "Connections" klasörünün altında oluşturulacaktır. Bu sebepten dolayı ileride sorumlara sebebiyet vermemek için bu dosya adında Türkçe karakter kullanmamamız gerekiyor. Daha sonra sitemizi host firmamıza upload ettiğimizde makinamızdaki connection bilgileri ile sitemizde connection bilgileri genelde aynı olmadığından "Connections" klasöründeki bu dosyayı notepad ile açıp gerekli değişikleri yapmamız sitemizin host sitesinde de sorunsuz çalışmasını sağlayacaktır.



Şekil 9. MYSQl Bağlantısının Oluşturulması

10.7.2 Mysql Server

Buradaki değere genelde localhost yazılır. Buraya yapılacak değer sizin tamamen ISS veya Apache'yi nasıl kurduğunuz ve yaptığınız ayarlamalar ile ilgilidir. Eğer normal veya standart bir kurulum gerçekleştirmiş ve bununla ilgili extra ayarlamalar yapmadısanız bu değer "localhost" olarak tanımlanmalıdır. Tabiki " " işaretlerini yazmıyoruz.

10.7.3 User Name

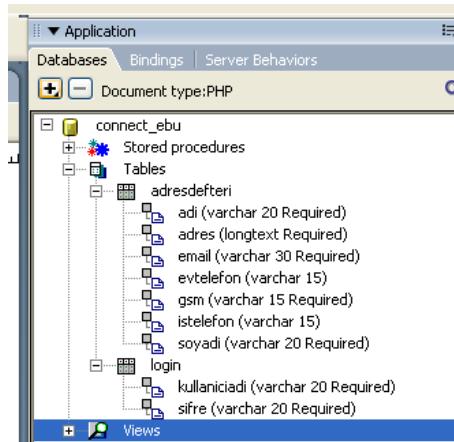
MySQL'e bağlanmak için gerekli olan kullanıcı adınızı bu bölüme yazıyoruz. Eğer local'de MySQL kullanıcı adı tanımlamadıysanız boş bırakılmalıdır.

10.7.4 Password

MySQL'e bağlanmak için gerekli olan şifrenizi bu bölüme yazıyoruz. Eğer local'de MySQL kullanıcısı için şifre tanımlamadıysanız boş bırakılmalıdır.

10.7.5 Database

Yukarıdaki alanları eksiksiz ve tam olarak doldurduğunuz zaman size mysql'de kullanabileceğimiz veritabanlarını browse tuşu ile belirleyebilirsiniz. Bu işlemler sonrasında mysql veritabanımız kullanıma hazırlıdır. Artık Database menüsünde veritabanınızı görebilirsiniz.



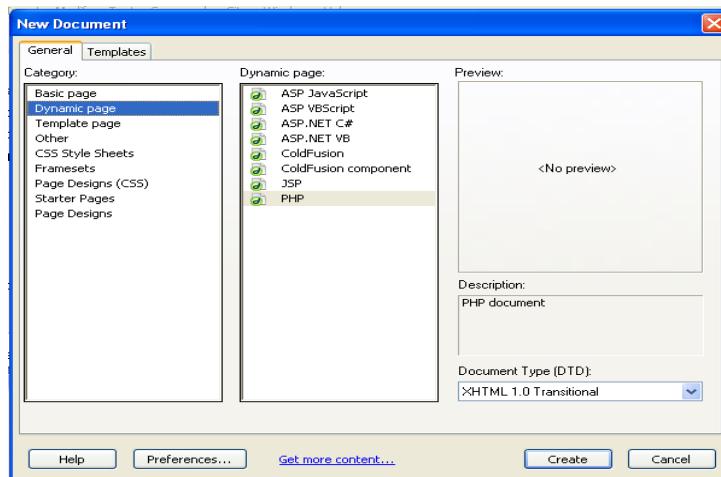
10.8 Dreamweaverde PHP ile Veritabanına Kayıt Ekleme

Örnek olarak küçük bir adres defteri uygulaması geliştirelim. Veri tabanındaki tablomuzda; isim, soyisim, adres, ev telefonu, iş telefonu, GSM ve e-mail alanları yeterli olacaktır.

The screenshot shows the 'adresdefteri' table structure in Dreamweaver. The table has seven columns: Alan (Field), Türü (Type), Karşılaştırma (Comparison), Öznitelikler (Attributes), Boş (Null), Varsayılan (Default), and E (Edit). The fields are: adi (varchar(20)), soyadi (varchar(20)), adres (longtext), evtelefon (varchar(15)), istelefon (varchar(15)), gsm (varchar(15)), and email (varchar(30)). The 'Boş' column for 'evtelefon' and 'istelefon' is 'Evet' (Yes), and their 'Varsayılan' values are 'NULL'. The other fields have 'Hayır' (No) in the 'Boş' column and 'Hayır' in the 'Varsayılan' column. A note at the bottom says 'Tümünü İşaretle / Hiçbirini Seçme Seçilileri: [checkbox icons]'

Alan	Türü	Karşılaştırma	Öznitelikler	Boş	Varsayılan	E
adi	varchar(20)	utf8_general_ci		Hayır		
soyadi	varchar(20)	utf8_general_ci		Hayır		
adres	longtext	utf8_general_ci		Hayır		
evtelefon	varchar(15)	utf8_general_ci		Evet	NULL	
istelefon	varchar(15)	utf8_general_ci		Evet	NULL	
gsm	varchar(15)	utf8_general_ci		Hayır		
email	varchar(30)	utf8_general_ci		Hayır		

Tablomuzu oluşturduktan sonra oluşturduğumuz sitemizde Veri tabanında tutacağımız kayıtların girişini sağlayacak yeni bir sayfa açarak orada kayıt formumuzu oluşturalım. Dreamweaver açıp; File>Yeni>Dinamic Pge>PHP sekülerini seçip sayfamızı oluştururuz.



Şekil 10. Kayıt Formunu Oluşturmak İçin Sayfa Açılması

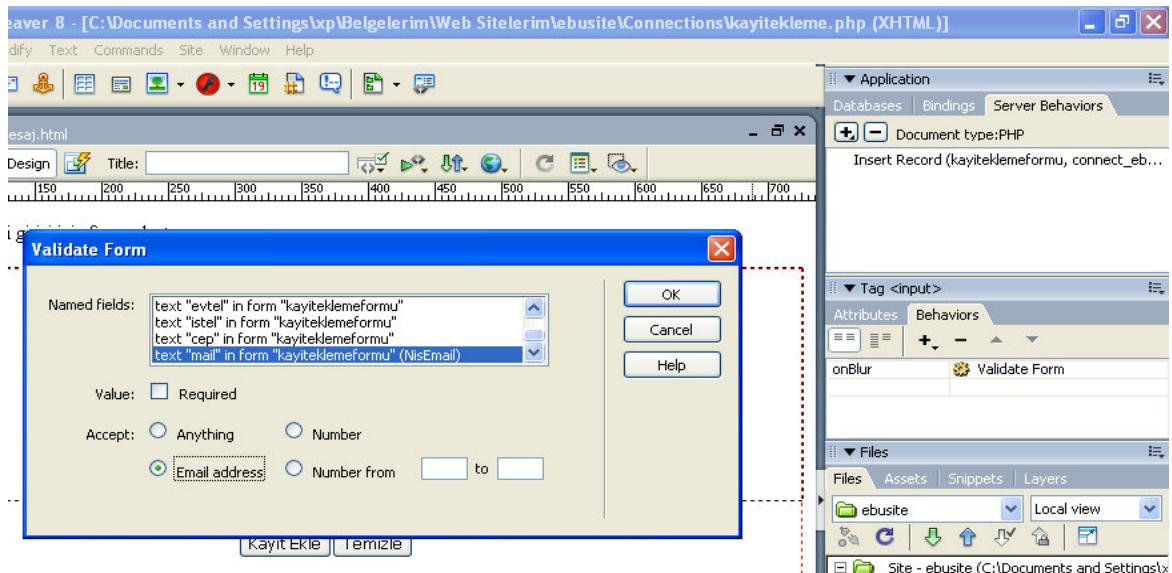
Veri tabanındaki her sütun için bir form alanı olması gerekmektedir. Form alanında oluşturacağımız nesnelerin adı veri tabanındaki alan adlarıyla aynı olursa insert record davranışında alanların eşleşmesi daha kolay olur. Bir form içerisinde; isim, soyisim, adres, ev-telefonu, iş-telefonu, gsm ve email adlarında form nesnelerini ve en sonda da bir Ekle butonu oluşturuyoruz.

10.8.1 Form Oluşturma

Sayfamızda basit bir form oluşturalım. Bu form içerisinde temel iletişim bilgileri yer alınsın. Oluşturduğumuz formun sayfamızdaki görünümü aşağıdaki gibi olmalıdır. Oluşturduğumuz form içerisindeki "Text Field" lara Properties bölümünden mantıklı isimler verelim. Bu işlemin yapılması uyarı mesajlarında bu isimlerin kullanılmasından dolayı önem taşımaktadır.

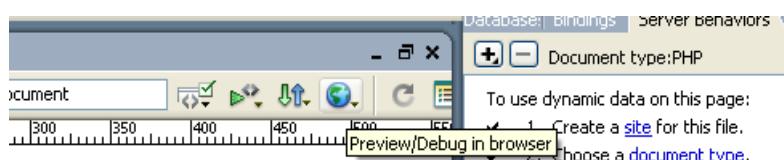
Şekil 10. Form Oluşturulması

İsimlendirmeyi yaptıktan sonra Behaviors bölümünden Validate Form'u seçtiğinizde aşağıdaki iletişim kutusu açılır.



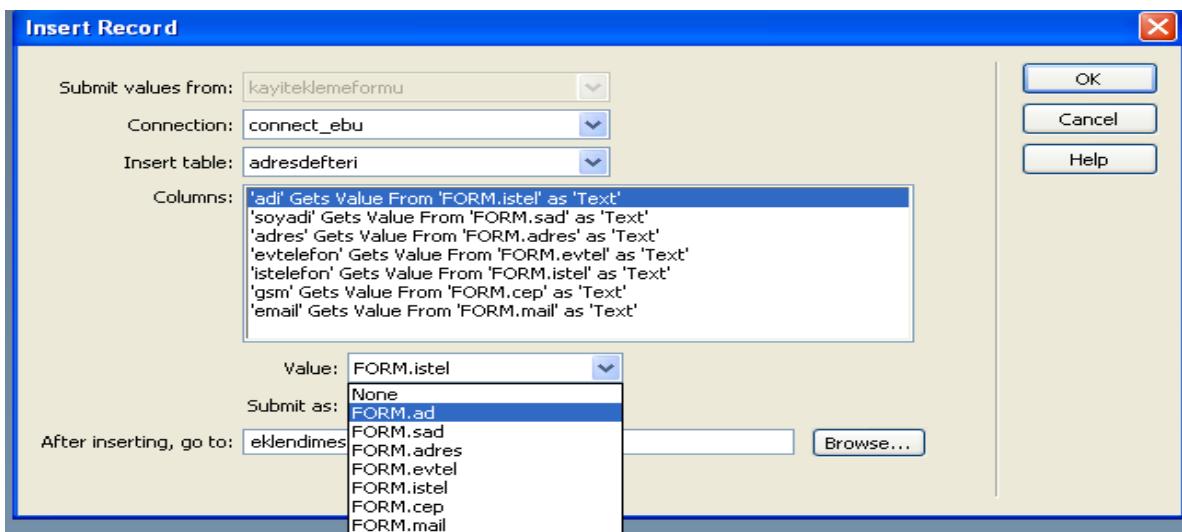
Şekil 11. Geçerliliğin sağlanması

Her alan için uygun olan denetleme kriterleri belirlenir. Bilgi girilmesi zorunlu alanlar "Required" olarak ayarlanır. Ad ve Soyad alanlarını "Anything" olarak belirlemek, Telefon alanını "Number" olarak belirlemek, E-Posta alanını ise "Email Address" olarak belirlemek doğru olacaktır. Formu oluşturduktan sonra sayfamıza bir isim verip kaydediyoruz. F12 Yada Preview simgesinden nasıl göründüğü kontrol edebiliriz.



Oluşturulan form ve veri tabanı arasında bağlantının kurulması

Applications>Server Behaviors>Insert Record sekmesini seçip öncelikle hangi formdan veri ekleneceğini kontrol ediyoruz, daha sonra bağlantımızı ve hangi tabloya ekleneceğini seçip alta gelen form alanlarını veri tabanındaki sütunlarla eşleştiriyoruz.



Şekil 12. Oluşturulan Form ve Veri Tabanı Arasında Bağlantının Kurulması

En alttaki bölümünden veri eklendikten sonra gelecek sayfayı belirliyoruz.

OK'e bastığımızda gerekli olan kod php sayfamıza yerleşmiş olur.



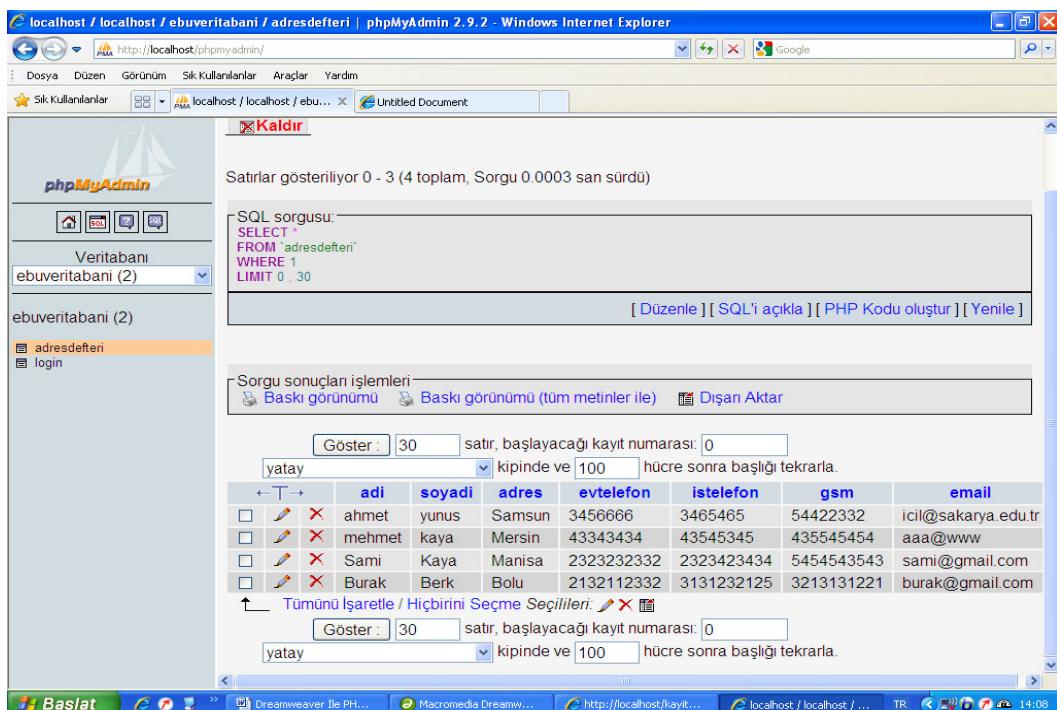
F12 Yada Preview simgesine basıp. Yeni bir kayıt girişi yapıyoruz.

Adı	Burak
Soyadı	Berk
Adresi	Bolu
Ev Telefonu	2132112332
İş Telefonu	3131232125
GMS	3213131221
e-mail	burak@gmail.com

Kayıt Ekle **Temizle**

Şekil 13 Yeni Bir Kayıt Girişinin Yapılışı

Girilen kaydın kontrolünü PhpMyadminden kontrol edebiliriz.



Şekil 14. Girilen Kaydın Kontrolünü

10.8.2 Dreamweaver'de PHP ile Kullanıcı Adı-Şifre Korumalı Sayfalar Oluşturmak

MySQL veri tabanında kullanıcılar isimli bir tablo ve bu tabloda da; kullanıcı adı, şifre, seviye alanlarını oluşturarak kullanıcılar eklemelisiniz. Tablo aşağıdaki şekilde olacaktır.

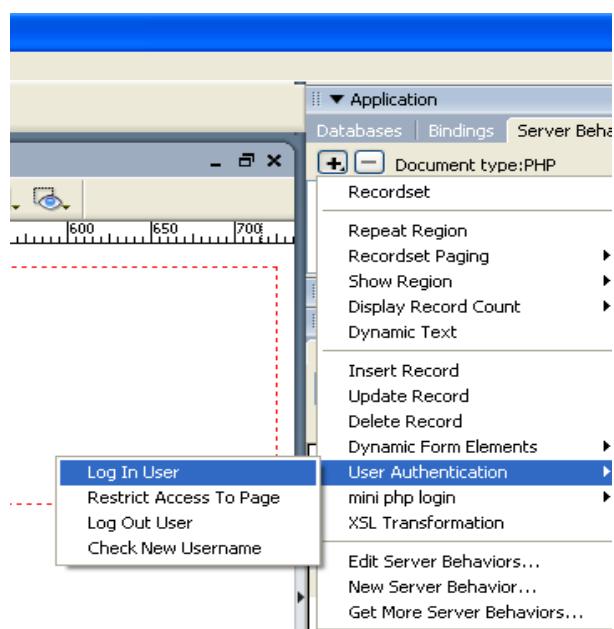
Daha sonra `INSERT INTO login VALUES ('icil', '123');` komutuyla kullanıcı yada kullanıcılar eklenir.

Öncelikle kullanıcıların siteye erişim iznini sağlayabilecekleri bir login.php oluşturmamız gerekmektedir. Bunun için Dreamweaverde bu sayfayı oluşturup, login formu hazırlanır.

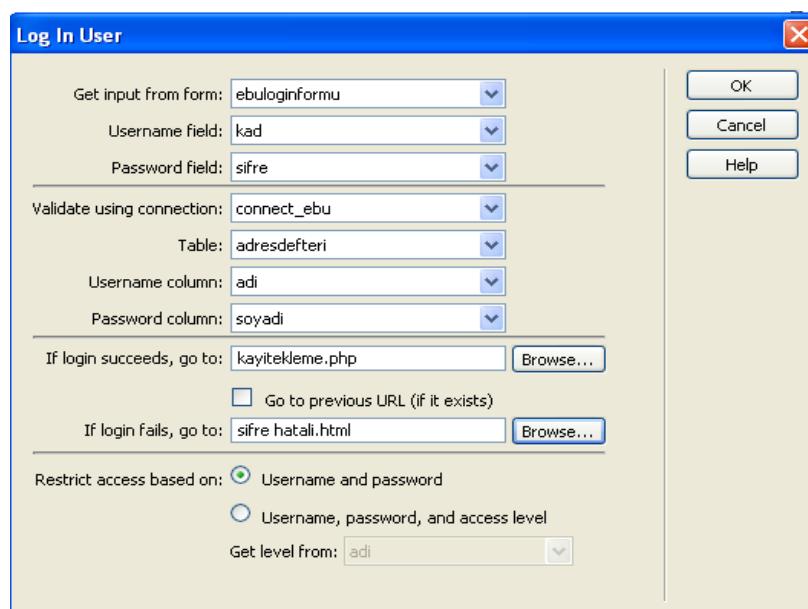


Form hazırlandıktan sonra

Application>Server Behaviors>User Authentication>Log in User Seçilir

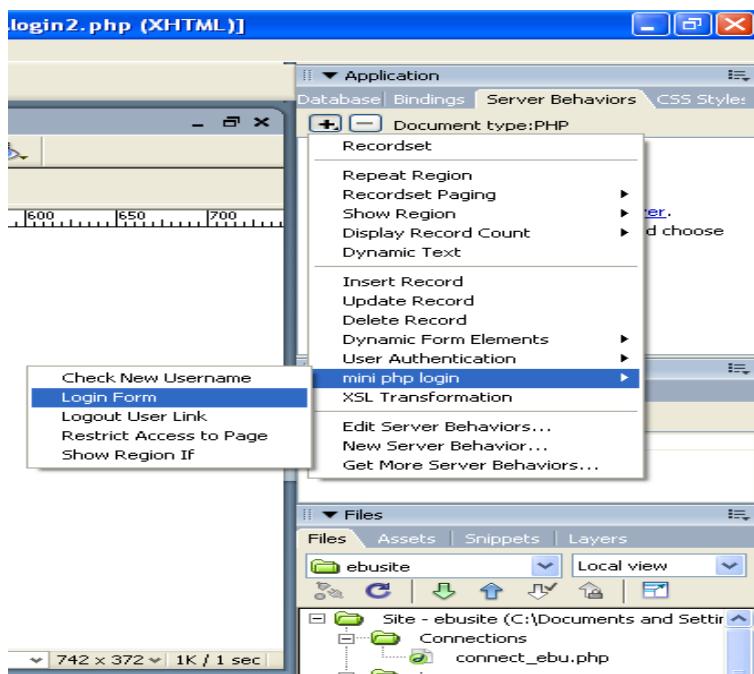


Karşımıza çıkan iletişim kutusu aşağıdaki gibidir.

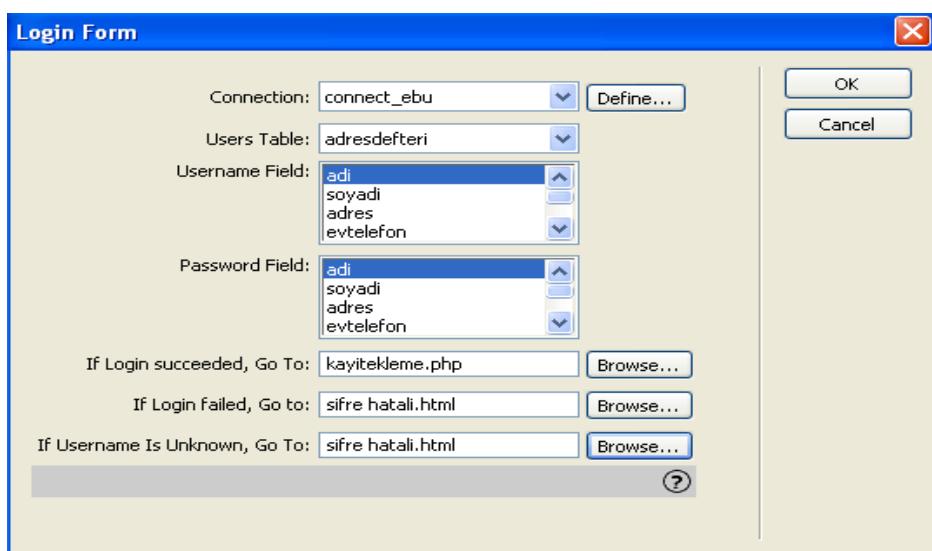


10.8.3 İkinci Yol (Dreamweaver'de PHP İle Kullanıcı Adı-Sifre Korumalı Sayfalar Oluşturmak)

PHP'de kullanıcı oturumuna dayalı sayfalar oluşturmak için MiniPHPLogin.mxp isimli dosyayı Macromedia Extension Manager aracılığı ile yüklemiş olmanız ve Dreamweaver ile MySQL bağlantısı yapmış olmanız gerekmektedir. Bir php dökümanı oluşturup bunun adını login.php verdikten sonra Application panelinden Server Behaviors>mini php login>Login Form davranışını seçilmelidir.



Karşımıza çıkan iletişim kutusu aşağıdaki gibidir.



İletişim kutusundaki alanların açıklamaları aşağıda yer almaktadır:

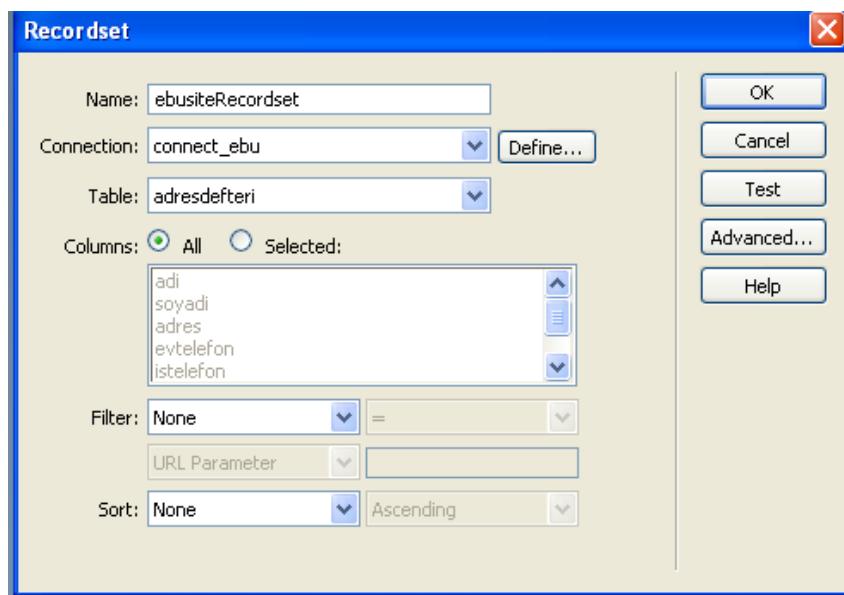
Connection: Bağlantı ayarlarının olduğu dosya. Users Table: Kullanıcı verilerinin alınacağı tablo. Username Field: Kullanıcı adlarının alınacağı alan. Password Field: Şifrelerin alınacağı alan.

"Login Succeeded" alanı kullanıcı adı ve şifre doğru ise gidilecek olan sayfayı, "Login Failed" alanı ise kullanıcı adı ve şifre yanlış ise gidilecek olan sayfayı belirler. Son alan ise kullanıcı adı bulunamaz ise gidilecek sayfayı belirler. OK'e tıkladığınızda gerekli olan kod PHP sayfasına eklenecektir.

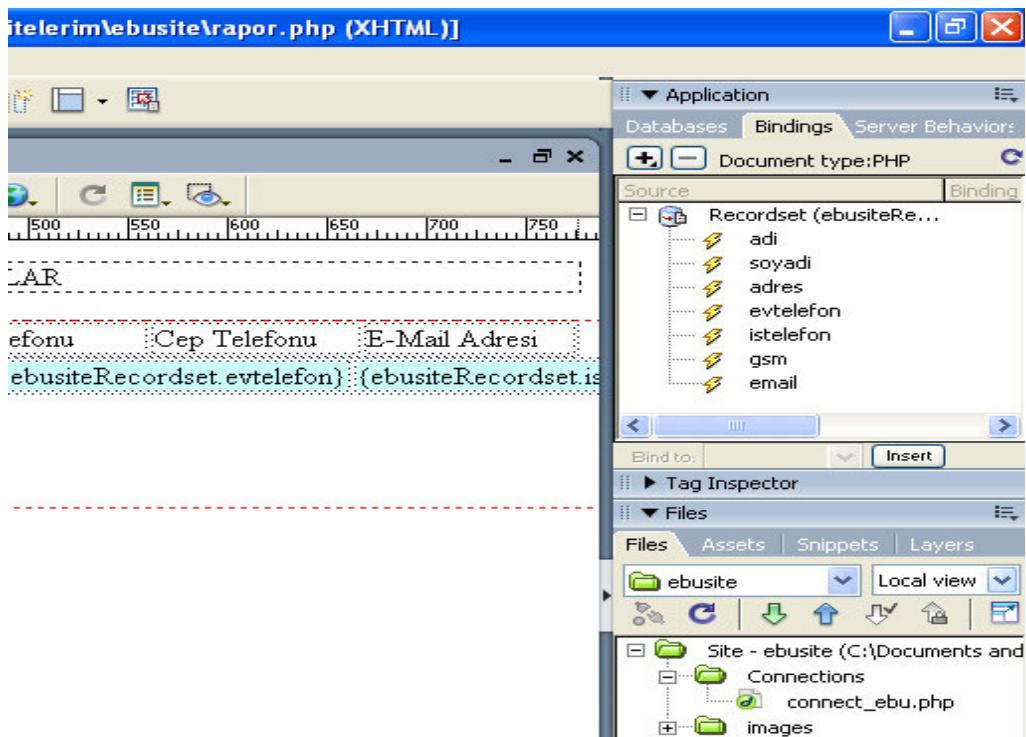
User:	<input type="text"/>
Password:	<input type="password"/>
<input type="button" value="Submit"/>	<input type="button" value="Reset"/>

10.8.4 Dreamweaver'de PHP İle Veritabanından Kayıt Okuma ve Rapor Oluşturma

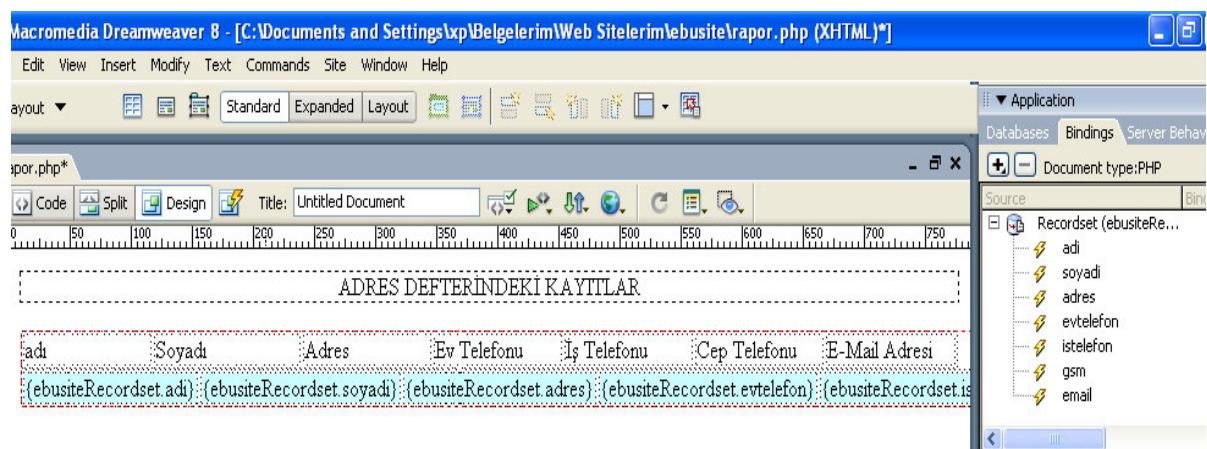
Bu dersteki örnek uygulamamız adres defterindeki verileri sayfaya yazdırma olacaktır. Application panelindeki Bindings sekmesinden RecordSet'i oluşturalım. RecordSet'i tanımlayıcı bir isim verelim. Connection açılır menüsünden daha önce oluşturulan bağlantıyı seçerek Table alanına veri tabanındaki tablo isimlerinin gelmesini sağlayalım. Buradan verinin okunacağı tablo seçelim.



OK'e basarak RecordSet'in oluşumunu tamamlayalım.



Binding panelinde beliren alan adlarını sayfamızda istediğimiz yere sürükle bırak mantığı ile ekleyerek bu verilerin görüntülenmesini sağlayabiliriz. Bu işlemleri gerçekleştirdikten sonra sayfanın görünümü alttaki gibi olmalıdır.



F12 ile sayfanızı test edebilirsiniz.

NOT: Dinamik içerik kullanılan sayfalarda, sayfanın sunucu tarafında güncellendiğinden emin olunuz.

10.8.4.1 Dreamweaver'te Repeat Region Kullanımı

Veri tabanındaki kayıtların tamamını yada belirli bir kısmını sayfada gösterebilmek için kayıt görüntüleme ilgili dersimizde oluşturduğumuz yapıya Repeat Region davranışını eklemeliyiz. Repeat Region kullanılmadığı zamanlarda veri tabanına son girilen kayıt gösterilebilmektedir.



Repeat Region davranışının sağlıklı kullanılabilmesi için verilerin tablo içerisinde bulunması önem taşımaktadır.

Aşağıda yer alan örnek tabloda verinin eklendiği satır seçilerek Repeat Region davranışı uygulanır.

Gerekli olan çalışma yapıldıktan sonra sayfanın tarayıcıdaki görüntüsü aşağıdaki gibi olacaktır.

Uygulama Soruları

Bir adres defteri veri tabanı hazırlayarak web uygulamasını geliştiriniz.

Bu Bölümde Ne Öğrendik Özeti

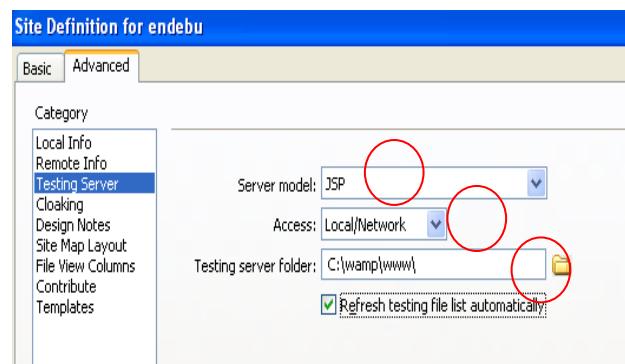
Web Tabanlı Veri tabanı Yönetim Sistemleri, İnternet üzerinden erişilebilen ve yapılandırılabilen veri tabanı sistemleridir. Ortaya çıkışındaki temel etkenler: Farklı sunucu ve istemci mimarilerinde çalışabilecek yapıya sahip veri tabanı uygulamalarına duyulan ihtiyaç ve Uzak bilgisayarlarda yer alan veri tabanı sistemlerini yapılandırma isteği şeklinde sıralanabilir. Bu derste bu teknolojilerin nasıl çalıştığı anlatıldı. Özellikle veri tabanı uygulamalarını ilgilendiren Web ile ilgili temel bilgiler açıklandı.

Bu derste verilen bilgiler doğrultusunda Web uygulamalarının nasıl geliştirildiği öğrenilmiş olundu. Hazırlanan MySQL veri tabanına internet üzerinden veri girmek ve veri tabanındaki kayıtlardan rapor almak için gerekli olan işlemlerin nasıl yapıldığı adım adım uygulamalı olarak anlatıldı.

Bölüm Soruları

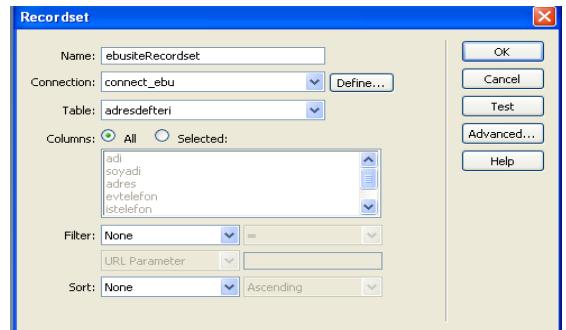
1) Numaralı sekmlerde hangisi hatalıdır?

- a) 1 hatalıdır.
- b) 2 hatalıdır.
- c) 3 hatalıdır.
- d) Hepsi hatalıdır.
- e) Hiçbiri hatalı değildir.



2) Dreamweaver de bu arayüz niçin kullanılır?

- a) Form bağlantısı için.
- b) MySQL connection ile veritabanı bağlantısı için.
- c) Veri tabanından kayıt okumak için.
- d) Veritabanına kayıt girmek için.
- e) Forma aracılığında bilgi girmek için.



3) Application>Server Behaviors>User Authentication>Log in User

Sekmesi ile aşağıdakilerden hangisi gerçekleştirilebilir?

- a) Kullanıcı adı ve şifre korumalı sayfalar oluşturulur.
- b) Web sitesinde ilk sayfa hazırlanır.
- c) İsim ve şifre doğrulaması yapılır.
- d) Oluşturulan formun davranışları girilir.
- e) Hiçbiri

4) Bindings sekmesindeki Recordset aşağıdaki uygulamalardan hangisi için gereklidir?

- a) Form
- b) Rapor
- c) Server Behaviour
- d) Veri Tabanı Bağlantısı
- e) Şifre ve Simge Doğrulama

5) Internet üzerinden erişilebilen ve yapılandırılabilen veri tabanı sistemlerine ne denir?

- a) Veri Tabanı Yönetim Sistemleri
- b) Veri Yönetimi
- c) Web Uygulamaları
- d) E-Ticaret
- e) Web Programcılığı

6) Apache nedir?

- a) Web Sunucusu
- b) Uygulama Sunucusu
- c) Veri Tabanı Sunucusu
- d) Program Sunucusu
- e) Local Sunucu

7) Aşağıdakilerden hangisi MySQL'in en önemli özelliklerinden biri değildir?

- a) Ücretsizdir.
- b) Açık kaynak kodludur. (Open Source).
- c) Tüm platformlarda (Windows/Linux/Unix) sorunsuzca çalışır.
- d) Internet Information Server (IIS) sunucu yazılımı ile uyumlu çalışır.
- e) Özellikle internet ortamında önem kazanan, çok esnek ve güçlü bir kullanıcı erişim kısıtlama/yetkilendirme sistemine sahip.

8) Aşağıdakilerden hangisi bir web sunucu yazılımıdır?

- a) CuteFTP
- b) WAMP
- c) PHP
- d) HTTP
- e) ASP

9) Aşağıdaki hizmetlerden hangisi PhpMyAdmin yerine getirdiği işlerden biri değildir?

- a) MySQL veri tabanının Web ya da yerel ağ üzerinden yönetilmesini sağlar.
- b) Wamp Serveri çalıştırılmasını sağlar.
- c) IIS'in durdurulmasını sağlar.
- d) RecordSet'in oluşturulmasını sağlar.
- e) PHP ile Veri tabanından Kayıt Okumayı sağlar

10) “Bir ağ üzerindeki istemci ve sunucunun birbiriyle iletişimini kurulmasını sağlayan kuralları tanımlayan metoda.....denir.”

Yukarıdaki boşluğa aşağıdakilerden hangisi gelmelidir?

- a) Protokol
- b) Sistem Mimarisi
- c) Web Tarayıcı
- d) HTTP
- e) CuteFTP

11) Aşağıdakilerden hangisi MySQL’dirin en önemli özelliklerindendir?

- a) Açık kaynak kodludur. (Open Source)
- b) Ücretsizdir.
- c) Tüm platformlarda (Windows/Linux/Unix) sorunsuzca çalışır.
- d) PC bazlı ortamlarda önem kazanan bir sistemdir.

CEVAPLAR

1-A, 2-B, 3-A, 4-B, 5-C, 6-A, 7-D, 8-B, 9-C, 10-A, 11-D