İÇİNDEKİLER

BÖLÜM 1: TEMEL KAVRAMLAR
1.1. GELENEKSEL DOSYALAMA SİSTEMLERİ 3
1.1.1.Sıralı Erişimli Dosyalar 3
1.1.2.Doğrudan Erişimli Dosyalar 4
1.2. VERİTABANI NEDİR? 5
1.3. VERİTABANI KAVRAMLARI 6
1.4. VERİTABANI YÖNETİM SİSTEMİ(VTYS) 12
1.5. İSTEMCİ-SUNUCU(CLIENT-SERVER) MİMARİSİ 13
1.6. VERİTABANI YÖNETİM SİSTEMLERİNİN SAĞLADIĞI YARARLAR 14
1.7. VERİTABANI KULLANICILARI 17
1.7.1.Veritabanı Yöneticisi(DBA-Database Administrator) 17
1.7.2.Uygulama Programcısı 19
1.7.3.Sorgu Dili Kullanıcıları 19
1.7.4.Son Kullanıcılar 19
1.8. VERİTABANI YÖNETİM SİSTEMİ YAZILIMLARI 20
1.8.1.Microsoft SQL Server 20
1.8.2.ORACLE 23
1.8.3.MySQL 25
1.8.4.Sybase 26
1.8.5.PostgreSQL 26
1.8.6.Microsoft Access 27
1.8.7.IBM DB/2 27
1.9. VERİ MODELLERİ 27
1.9.1.Sıradüzensel(Hiyerarşik) Veri Modeli 27
1.9.2.Ağ Veri Modeli 28
1.9.3.İlişkisel Veri Modeli 29
1.9.4.Nesneye Yönelik Veri Modeli 31
1.10.NEREDEN BAŞLAMALIYIZ? 32
1.11.BÖLÜM ÖZETİ 32
1.12.ÇALIŞMA SORULARI 33

BÖLÜM 2:İLİŞKİSEL VERİTABANI TASARIMI VE NORMALİZASYON
2.1. VERİTABANI TASARIMI 37
2.2. İLİŞKİSEL VERİTABANININ KAVRAMSAL TASARIMI 38
2.2.1.Varlık-İlişki Modelinin Tablolara Dönüştürülmesi 44
2.3. VERİTABANI NORMALİZASYON KURALLARI 50
2.3.1.Normalizasyonun Amaçları 51
2.3.2.Fonksiyonel Bağımlılık 51
2.3.3.Normalizasyon Kuralları 52
2.4. ÖRNEK VERİTABANI TASARIMI 65
2.5. BÖLÜM ÖZETİ 70
2.6. ÇALIŞMA SORULARI 71

BÖLÜM 3: ÖRNEK KÜTÜPHANE VERİTABANI
3.1. KÜTÜPHANE VERİTABANININ ÖZELLİKLERİ 75
3.2. ER(Entity-Relationship) Modeli ve Kavramları 76
3.3. Örnek Veritabanının ER Modeli İle İlk Tasarımı 78
3.4. İlişkilerin Belirlenmesi ve İlişki Tipleri 79
3.5. Örnek Kütüphane Veritabanına Ait ER Modeli 83
3.6. Veritabanı Şemasının Oluşturulması 84
3.7. Veritabanı Şemasının Tablolarla Gösterilmesi 89
3.8. Kütüphane Veritabanı İçin Örnek Durum 90

Scanned with CamScanner

BÖLÜM 4: İLİŞKİSEL CEBİR

4.1. Seçim(Selection) 98
4.2. Atma(Projection) 100
4.3. Kartezyen Çarpım(Cross-Product) 102
4.4. Birleşim(Union) 104
4.5. Küme Farkı(Set-difference) 106
4.6. Kesişim(Intersection) 107
4.7. Bölme(Division) 108
4.8. Birleştirme(Join) 110
4.9. Yeniden Adlandırma(Renaming) 115
4.10.Özetleme ve Gruplama 117
4.11.İlişkisel Cebir Sorgulama Örnekleri 119
4.12.Bölüm Özeti 123
4.13.Çalışma Soruları 124

BÖLÜM 5: SQL - YAPISAL SORGULAMA DİLİ

5.1. VERİ TANIMLAMA DİLİ(DDL) 130
5.1.1.CREATE İfadesi 130
5.1.2.ALTER İfadesi 148
5.1.3.DROP İfadesi 151
5.2. VERİ İŞLEME DİLİ(DML) 153
5.2.1.SELECT İfadesi 153
5.2.2.INSERT İfadesi 159
5.2.3.UPDATE İfadesi 162
5.2.4.DELETE İfadesi 164
5.3. VERİ KONTROL DİLİ(DCL) 166
5.3.1.GRANT İfadesi 166
5.3.2.DENY İfadesi 167
5.3.3.REVOKE İfadesi 168
5.3.4.VTYS 'ye Göre Veri Kontrol Dili 169
5.4. BÖLÜM ÖZETİ 170
5.5. ÇALIŞMA SORULARI 170

BÖLÜM 6: TEK TABLO İÇİNDE SORGULAMALAR

6.1. SORGU SONUCU TEKRARLI SATIRLARI ÖNLEMEK 177
6.2. SORGU SONUCUNU SIRALAMA 179
6.3. BİRDEN FAZLA SÜTUNA GÖRE SIRALAMA 181
6.4. SÜTUNLAR İÇİN TAKMA İSİM KULLANMA 184
6.5. SÜTUNLAR ÜZERİNDE MATEMATİKSEL İŞLEMLER 185
6.6. SÜTUNLARIN BİRLEŞTİRİLMESİ 187
6.7. KOŞULA BAĞLI SORGULAMALAR 188
6.7.1.Karşılaştırma Operatörleri 189
6.7.2.Tarih Türündeki Sütunlar İçin Koşul İfadesi 191
6.7.3.Birden Fazla Koşula Göre Sorgulama 191
6.7.4.IN Operatörü İle Veri Kümesi İçerisinde Arama 198
6.7.5.BETWEEN ... AND ... Operatörüyle Aralık Sorgulama 200
6.7.6.Karakter Kümesi İçeren Verileri Joker Karakter Kullanarak Sorgulama 201
6.7.7.NULL Değer Sorgulama 204
6.8. BÖLÜM ÖZETİ 205
6.9. ÇALIŞMA SORULARI 207

BÖLÜM 7: ARİTMETİKSEL OPERATÖRLER VE FONKSİYONLAR

7.1. ARİTMETİKSEL OPERATÖRLER 211
7.2. FONKSİYONLAR 214
7.2.1.Karakter Kümesi Fonksiyonları 214
7.2.1.1.İki Karakter Kümesini Birleştirmek 214
7.2.1.2.Verinin İstenilen Bölümünü Almak – SUBSTRING() 215

Scanned with CamScanner

7.2.1.3.Veriden İstenilen Kısmı Almak - LEFT() ve RIGHT() 218
7.2.1.4.Büyük-Küçük Harf Dönüşümü - LOWER() ve UPPER() 219
7.2.1.5.Boşlukları Kaldırmak – TRIM(), RTRIM() ve LTRIM() 220
7.2.1.6.Karakter Sayısını Bulmak – LEN() ve LENGTH() 223
7.2.1.7.İlk Harfi Büyük Yapmak – INITCAP() 225
7.2.1.8.İstenilen Bilgiyi Değiştirmek – REPLACE() 225
7.2.1.9.Karakter Arama – CHARINDEX() ve INSTR() 226
7.2.1.10.Tersten Yazdırma – REVERSE() 227
7.2.2.Sayısal Değerler İçin Kullanılan Fonksiyonlar 227
7.2.2.1.Mutlak Değer – ABS() 227
7.2.2.2.Aşağı ve Yukarı Yuvarlama – FLOOR() ve CEIL() 229
7.2.2.3.En Yakın Tamsayıya Yuvarlama – ROUND() 230
7.2.2.4.Bir Sayının Kuvvetini Alma – POWER() 231
7.2.2.5.Karekök Alma – SQRT() 231
7.2.2.6.Toplamı Bulma – SUM() 232
7.2.2.7.Ortalama Değeri Hesaplama – AVG() 235
7.2.2.8.En Büyük ve En küçük Değeri Bulma–MAX() ve MIN() 236
7.2.2.9.Satır Sayısını Bulma – COUNT() 237
7.2.2.10.Bölümden Kalanı Bulma – MOD() veya % 238
7.2.3.Tarih ve Saat Fonksiyonları 238
7.2.3.1.Güncel Tarih ve Saat Bilgisini Alma 238
7.2.3.2.MS SQL Server için DATENAME() Fonksiyonu 242
7.2.3.3.MS SQL Server ve MySQL için DATEDIFF() Fonksiyonu 242
7.2.3.4.MS SQL Server için DATEADD() Fonksiyonu 243
7.2.3.5.ORACLE için ADD_MONTHS() Fonksiyonu 244
7.2.3.6.MySQL için DATE_ADD() ve DATE_SUB() Fonksiyonu 244
7.2.4.Dönüşüm Fonksiyonları 245
7.2.4.1.MS SQL Server İçin Veri Tipi Dönüşümü 245
7.2.4.2.ORACLE İçin Veri Tipi Dönüşümü 248
7.2.4.3.MySQL İçin Veri Tipi Dönüşümü 251
7.3. BÖLÜM ÖZETİ 252
7.4. ÇALIŞMA SORULARI 253

BÖLÜM 8: GRUPLANDIRARAK SORGULAMA
8.1. Gruplandırma İşleminde Kullanılan Fonksiyonlar 260
8.2. BİRDEN FAZLA SÜTUNA GÖRE GRUPLANDIRMA 264
8.3. GRUPLANDIRMA İŞLEMLERİNDE KOŞUL İFADELERİ 267
8.4. BÖLÜM ÖZETİ 271
8.5. ÇALIŞMA SORULARI 272

BÖLÜM 9: BİRDEN FAZLA TABLO ÜZERİNDE SORGULAMA
9.1. Tablolarda Takma İsim Kullanma 277
9.2. Tabloların Birleştirilmesi(join) 278
9.2.1.WHERE İfadesi İle Birleştirme 280
9.2.2.Inner Join 281
9.2.3.Outer Join 290
9.2.3.1.Left Outer Join 291
9.2.3.2.Right Outer Join 292
9.2.3.3.Full Outer Join 293
9.2.4.Cross Join(Kartezyen Çarpım) 296
9.2.5.Natural Join(Doğal Birleştirme) 297
9.2.6.Self Join(Tabloyu Kendisiyle Birleştirme) 298
9.3. Ortak Sütunu Bulunmayan Tabloları Birleştirme 299
9.4. TABLOLARI KÜMELEME İFADELERİ İLE BİRLEŞTİRME 300
9.4.1.İki Sorgunun Kesişimi(INTERSECT) 300
9.4.2.İki Sorgunun Farkı(EXCEPT) 302
9.4.3.İki Sorguyu Birleştirme(UNION ve UNION ALL) 304

Scanned with CamScanner

9.5. BÖLÜM ÖZETİ 306
9.6. ÇALIŞMA SORULARI 307

BÖLÜM 10: ALT SORGULAR
10.1.ALT SORGULARDA DİKKAT EDİLECEK NOKTALAR 311
10.2.ALT SORGU OLUŞTURULMASI 312
10.3.ALT SORGULARDA IN VE NOT IN KULLANIMI 318
10.4.ALT SORGULARDA GROUP BY VE HAVING KULLANIMI 325
10.5.ALT SORGULARDA ANY VE ALL KULLANIMI 325
10.6.ALT SORGULARDA EXISTS VE NOT EXISTS KULLANIMI 328
10.7.SÜTUN İSİMLERİNDE ALT SORGU KULLANIMI 333
10.8.FROM İFADESİNDE ALT SORGU KULLANIMI 337
10.9.BÖLÜM ÖZETİ 339
10.10.ÇALIŞMA SORULARI 341

BÖLÜM 11: GÖRÜNÜMLER(views)
11.1.VIEW OLUŞTURMANIN YARARLARI 345
11.2.VIEW OLUŞTURMAK 345
11.3.VIEW YAPISINI DEĞİŞTİRMEK(ALTER VIEW) 359
11.4.VIEW'LERİ SİLMEK(DROP VIEW) 360
11.5.VIEW'LER İLE GÜVENLİĞİN SAĞLANMASI 360
11.6.MS SQL Server'da View'ler Hakkında Bilgi Toplama 360
11.7.BÖLÜM ÖZETİ 361
11.8.ÇALIŞMA SORULARI 362

BÖLÜM 12: INDEX KULLANIMI
12.1.INDEX NASIL ÇALIŞIR? 367
12.2.INDEX TİPLERİ 367
12.3.INDEX KULLANIMI 369
12.3.1.INDEX Ne Zaman Kullanılır? 369
12.3.2.INDEX Ne Zaman Kullanılmaz? 369
12.4.INDEX OLUŞTURMA 369
12.5.KARMA INDEX(Birden Fazla Sütun Üzerinde İndeks) 370
12.6.UNIQUE INDEX 371
12.7.ARTAN VEYA AZALAN SIRADA INDEX OLUŞTURMA 371
12.8.INDEX SİLME 372
12.9.BÖLÜM ÖZETİ 372
12.10.ÇALIŞMA SORULARI 373

EK A: MS SQL Server Kurulumu 377
EK B: Örnek Kütüphane Veritabanı İçin Gerekli SQL İfadeleri 389

Kaynakça 391

Scanned with CamScanner

BİRİNCİ BÖLÜM

1

1. TEMEL KAVRAMLAR

1.1. Geleneksel Dosyalama Sistemleri

1.2. Veritabanı Sistemleri

1.3. Veritabanı Kavramları

1.4. Veritabanı Yönetim Sistemi

1.5. İstemci-Sunucu(Client-Server) Mimarisi

1.6. Veritabanı Yönetim Sistemlerinin Sağladığı Yararlar

1.7. Veritabanı Kullanıcıları

1.8. Veritabanı Yönetim Sistemi Yazılımları

1.9. Veri Modelleri

1.10. Nereden Başlamalıyız?

1.11. Bölüm Özeti

1.12. Çalışma Soruları

Scanned with CamScanner

1. TEMEL KAVRAMLAR

1.1. GELENEKSEL DOSYALAMA SİSTEMLERİ

Bilgisayarlarda veri saklama işlemi bilgisayarın ilk bulunduğu yıllardan günümüze kadar uzanır ve her türlü kurum ve kuruluşlarda kullanılmaktadır.

Veri saklama işleminde geçmişten günümüze çok farklı yöntemler kullanılmıştır. Geleneksel dosyalama sistemleri veritabanı yönetim sistemleri öncesinde veri depolamak için kullanılan sistemlerdir. Bilgisayarların donanımsal veya yazılımsal gelişmelerinin en temel sebeplerinde birisi de ihtiyaçları en doğru ve en hızlı bir şekilde karşılamaktır. Zamanla geleneksel dosyalama sistemleri artan veri kapasitesi, veri işleme hızı ve kullanım alanı gibi birçok ihtiyaca cevap veremez duruma gelmiştir. Dolayısıyla artık yerini veritabanı yönetim sistemleri yazılımlarına bırakmıştır.

Çok fazla miktarda veri akışı olan sistemlerde veritabanı yönetim sistemleri kullanılmasına rağmen basit düzeyde ve fazla karmaşık olmayan verilerin saklanması için geleneksel dosyalama sistemleri kullanılmaya devam etmektedir.

Geleneksel dosyalama sistemlerinde, depolanacak veri grupları bağımsız olarak oluşturulmak zorundadır. Örneğin; kasa, banka, satış ve fatura bilgilerinin tutulması gerektiği durumlarda bu bilgilerin ayrı dosyalarda tutulması gerekir. Bu tür dosyalarda veri, sıralı veya doğrudan erişim yöntemi ile kullanılır.

1.1.1. Sıralı Erişimli Dosyalar

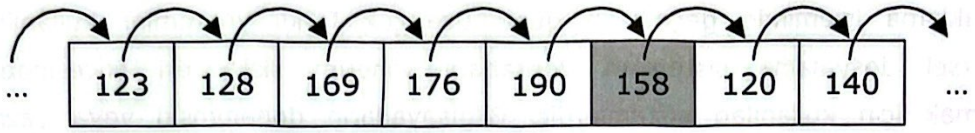
Bu tür dosyalarda veri sıralı erişim yöntemi ile işlenir. Sıralı erişimde bilgilerin tutulduğu dosya içeriği tarama yöntemiyle okunur. İstenilen kayda gelindiğinde işlem yapılır. Kullanılan programlama diline ve ihtiyaca bağlı olarak yeni eklenen veriler dosya sonuna, dosya başına veya imlecin bulunduğu konuma eklenebilir. Herhangi bir bilgi aranmak istendiğinde dosya başından itibaren okumaya başlanarak aranan kayıt bulunana kadar devam edilir, aranan bilgi yoksa okuma işlemi dosya sonuna kadar devam edecektir. Eskiden çoğunlukla kullanılmakta olan müzik kasetleri sıralı erişime güzel bir örnektir. Bir sanatçının kasetindeki 5 numaralı şarkıyı dinlemek için bundan önceki 4 şarkının ileri sarılması gerekmektedir.

Şekil 1.1. Sıralı erişimli dosya içeriği.

Scanned with CamScanner

Bu tür dosyalarda bir satırda tutulan farklı türdeki bilgileri birbirinden ayırt edebilmek için boşluk karakteri veya özel karakterler kullanılır.

Örneğin, 158 numaralı öğrencinin bilgileri dosyanın en sonunda ise, bu bilgilere ulaşmak istediğimizde buraya kadar olan tüm kayıtlar gereksiz yere okutulmuş olur.



Şekil 1.2. Sıralı erişim örneği.

Sıralı erişimli dosyaların en büyük dezavantajı istenilen bilgiye anında ulaşılamamasıdır. Ancak bazı durumlarda sıralı bir şekilde işlem gerektiren durumlar olabilir. Örneğin, müşteri bilgilerinin belirli bir id numarasına göre tutulduğu durumlarda, müşterilerin tamamının hesap dökümünü almak uzun bir iş olmasının yanı sıra sıralı bir iştir. Böyle bir döküm işlemi için sıralı erişim kullanılabilir çünkü müşteriler kendilerine verilen ıd bilgisine göre sıralı bir şekilde depolandığı için arama işlemi yapmadan sırasıyla döküm işlemi yapılacaktır.

1.1.2. Doğrudan Erişimli Dosyalar

Sıralı dosyalardaki en büyük sorun herhangi bir kayda ulaşmak için tüm bilgilerin okunmasıdır. Bu sorunun önüne geçmek için doğrudan erişimli dosyalar geliştirilmiştir.

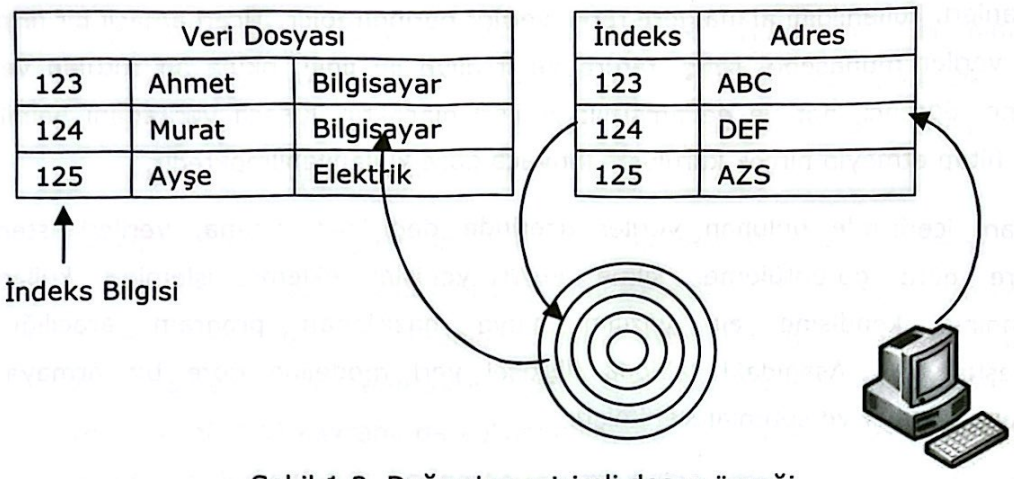
Bu tür dosyalarda, isminden de anlaşılacağı üzere istenilen bilgiye doğrudan erişim söz konusudur. Dosya başından itibaren istenilen bilgiye ulaşana kadar tüm bilgilerin okunmasına gerek yoktur.

Doğrudan erişimli dosyalar tuttuğu her satır bilgi için bir indeks numarasına sahiptir. Bu indeks numarası benzersiz değer içeren herhangi bir bilgi olabileceği gibi yazılımcı tarafından da oluşturulan sıralı değerler olabilir. Bilgilerin tutulduğu veri dosyasının yanı sıra indeks değerlerinin tutulması için indeks dosyası oluşturulur. İndeks dosyası, kitaplarda bulunan indeks bölümüyle aynı özelliğe sahiptir. Yani, her bir satırın sahip olduğu indeks değeri ile bu indeks değerine karşılık gelen disk adresini içerir.

Bir bilgiye ulaşılmak istendiğinde aranan bilginin karşılık geldiği indeks değeri indeksleri barındıran dosyadan bulunarak karşılık geldiği disk adresi bulunur ve bu disk adresi üzerinden aranan bilgiye ulaşılabilir.

Örneğin, 124 numaralı indeks değerine sahip kayda ulaşmak istendiğinde, bu indeks değerine sahip kaydın adresi tespit edilir ve bu adres üzerinden kayda ulaşılır.

Scanned with CamScanner



Şekil 1.3. Doğrudan erişimli dosya örneği.

1.2. VERİTABANI NEDİR?

Bilgisayarların ilk kullanılmaya başlandığı günden bu yana birçok alanda değişimler gerçekleşmiştir. Bu değişimlerin temelinde ihtiyaçlara daha iyi cevap vermek yatmaktadır. Eskiye oranla firmaların işlemiş olduğu veri miktarı da artmış ve karmaşıklaşmıştır. Geleneksel veri işleme teknikleri bu ihtiyaçlara cevap verememektedir.

Veritabanı kavramı günümüzde hemen hemen tüm alanlarda kullanılmaktadır. Veritabanı kullanılmayan sistemlerde girilen verilerin kalıcı olması mümkün değildir. Programcılığa ilk başladığınız günleri hatırlarsanız yazmış olduğunuz programlarda girmiş olduğunuz verilerin hiçbirisi kalıcı olmaz, program kapandığında kaybolurdu. O halde veritabanı istediğiniz bilgileri, istediğiniz türde, istediğiniz sürece saklanmasını sağlamaktadır.

Veritabanı, herhangi bir konuda birbiriyle ilişkili ve düzenli bilgiler topluluğudur. Günümüzde, basit müşteri bilgilerinin tutulmasından büyük kapsamlı şirket verilerin tutulmasına kadar veri depolaması gereken alanların tamamında veritabanları kullanılmaktadır. Örneğin, müşteri bilgilerinin, ödemelerin, taksitlerin, stokların, öğrenci bilgilerinin ve hastane otomasyon sisteminin verileri tutmak için veritabanları kullanılmaktadır. Veritabanlarında tutulan verilere erişim geleneksel dosyalama sistemlerine göre daha kolay ve hızlı bir şekilde gerçekleşir ve gereksiz veri tekrarlarının oluşmasını engeller.

Tabii ki, veritabanı kullanışlı olmasına rağmen kullanıcı kaynaklı yapılan hatalar veritabanının performansını düşürecek, gereksiz veri tekrarlarına sebep olacak ve istenilen verim alınamayacaktır. Bu nedenle, veritabanı kullanılması gereken durumlarda veritabanının organizasyonu hatasız bir şekilde yapılmalıdır.

Scanned with CamScanner

Veritabanları, kullanıldığı alana göre farklı veriler barındırabilir. Ticari amaçlı bir firmada tutulan veriler muhasebe, satış, fatura ve irsaliye ile ilgili, okula ait tutulan veriler öğretmen, öğrenci, not ve devamsızlık verileri olacaktır. Kısaca veritabanı belirli bir sektöre hitap etmeyip birçok kuruluşta ihtiyaca göre kullanılabilmektedir.

Veritabanı içerisinde bulunan veriler üzerinde değişiklik yapma, verileri istenilen kriterlere göre görüntüleme, silme veya yenisini ekleme işlemleri kullanılan veritabanının kendisine ait yazılımı veya hazırlanan program aracılığı ile gerçekleştirilebilir. Aşağıdaki tabloda ilişkisel veri modeline göre bir firmaya ait veritabanı, tablolar ve sütunlar verilmiştir.

Firma Data			
CARİ	**ÜRÜNLER**	**ALIM**	**SATIŞ**
Cari Kodu	*Ürün kodu*	*Ürün kodu*	*Ürün kodu*
Firma Adı	Ürün adı	Miktarı	Miktarı
Yetkili	Alış fiyatı	Toplam tutar	Toplam tutar
Telefon	Satış fiyatı	*Cari Kodu*	Cari Kodu
Faks	Birimi		
Adres			

Veritabanı(Database)

Tablolar(Tables)

Sütunlar/Alanlar

Şekil 1.4. Veritabanı Yapısı.

Burada "Firma Data" veritabanını, "Cari", "Ürünler", "Alım", "Satış" tabloları ve bunların altındakiler de tablolara ait sütunları göstermektedir. Alım ve Satış tablolarındaki ürün kodu sütunu ile ürünler tablosundaki ürün kodu, cari kodu sütunu ile cari tablosundaki cari kodu sütunu arasında ilişki vardır. Yani alım veya satış yaparken ürün kodu bilgisi veya cari kodu bilgisi rastgele girilemez, bu bilgiler cari ve ürünler tablosundaki bilgilerle bağlantılı olarak girilmelidir.

1.3. VERİTABANI KAVRAMLARI

Veritabanları, oluşturma ve kullanım aşamalarında çeşitli özellikleri içerisinde barındırmaktadır. Bu özellikler ve kavramlar veritabanının oluşmasını sağlamaktadır. Bu kavramlar aşağıda kısaca açıklanmıştır. Temel Veritabanı Kavramları bölümünde daha ayrıntılı olarak incelenecektir.

a) **Tablo:** Veritabanı içerisinde tutulacak verileri taşımak için kullanılır. Tablo, satır ve sütunlardan oluşan verilerin depolandığı veritabanı elemanıdır. Veritabanı, içerisinde barındırmış olduğu verileri tablolar halinde tutmaktadır. Bir veritabanı içerisinde birden fazla tablo kullanılabilir. Veritabanı içerisinde

Scanned with CamScanner

oluşturabileceğiniz tablo sayısı kullandığınız veritabanı sistemine göre değişiklik gösterebilmektedir. Veritabanı içerisinde oluşturulacak tablolar özellikle barındırılacak veri gruplarına göre belirlenir. Oluşturulan tablolar arasında çeşitli ilişkiler bulunabilir.

Örneğin, öğrenci bilgilerinin tutulduğu bir veritabanındaki öğrenci tablosu öğrencilerin no, ad, soyad gibi bilgilerini tutacaktır. Buradaki no, ad ve soyad bilgilerinin her biri bir sütunla gösterilecektir. Her bir sütunun barındırdığı veriye göre tipi belirlenecektir. Bazı kaynaklarda sütun yerine aynı anlama gelen alan(field) kavramı da kullanılmaktadır. Satır ise her bir öğrencinin tüm bilgileridir. Satır ifadesi yerine kayıt ifadesi de kullanılmaktadır. Tablo satırları kayıtları, sütunları ise alanları barındırmaktadır. Tablo içerisindeki her bir satır bir kaydın karşılığıdır. Yeni kayıtlar eklendikçe satır sayısı artacak veya kayıtlar silindikçe satır sayısı azalacaktır.

Hazırlanacak projeye başlamadan önce veritabanı tasarımı iyi yapılmalıdır. Daha sonradan tablolar veya alanlar üzerinde değişiklik yapıldığında projenin tamamı bundan etkilenir ve tekrar düzenlemek gerekir. Tablo içerisindeki satırlarda yani kayıtlarda değişiklik olması gayet doğaldır. Ama sütunlarda yani alanlarda değişiklik yapmamaya özen gösterilmesi, projenin sağlıklı çalışması açısından önemlidir.

Personel tablosundaki kayıtlar

per_no	adi	soyadi	gorevi
1995001	Ahmet	Akmaz	Personel Şefi
2000002	Ayşe	Güler	İşçi
2009003	Bekir	Duru	Hizmetli

Tablo içerisindeki her satır bir kaydı temsil eder. Ör. 1995001 nolu personelin bilgisi. Her kayıt içerisinde farklı alanlar vardır.

Tablo içerisindeki her sütun alanları gösterir. Tablomuzda, per_no, adi, soyadi, gorevi olmak üzere 4 adet alanımız vardır.

Şekil 1.5. Örnek tablo ve alanlar

b) Satır ve Sütun: Tabloyu oluşturan temel kavramlardır. Bir tablo satır ve sütunlardan oluşmaktadır. Sütun, tablo içerisinde tutulan her bir veri türüne verilen isimdir. Bir tablo içerisinde birden fazla kullanılmaktadır. Satır, tablo içerisinde sütuna ait veri grubudur. Örneğin; bir öğrencinin no, ad ve soyad

Scanned with CamScanner

bilgilerinin tamamı satırı ifade etmektedir. Çoğunlukla kayıt(record) olarak ifade edilir. Sütun içerisinde barındırılan veriler sütunun veri tipini (int, char, date v.b.) belirlemektedir.

Satır ve sütunları birbirinden bağımsız düşünmek mümkün değildir. Tablo içerisinde tutulan verilerden bir kayıt ifade edilirken hem satır hem de sütun ifade edilmiş olur.

c) **Veri Tipleri:** Oluşturulan veritabanında tutulan verilerin hepsi aynı türde değildir. Tutulacak veriye göre değişiklik gösterir. Örneğin isim bilgisi sayı ve karakter, no bilgisi sadece sayı, doğum tarihi bilgisi tarih bilgisini içerebilir. Bir veritabanı oluşturulurken önce tablolar sonra da tablolardaki alanların veri türlerinin ne olacağı tanımlanmak zorundadır. Ayrıca bir alanının uzunluğunun ne kadar olacağı veya en fazla kaç karakter yazılacağı belirlenmelidir.

İlerleyen konularda kullanılan veritabanına göre kullanılabilecek veri tipleri verilmiştir.

d) **Anahtar Kullanımı:** Tablolarda bulunan kayıtları birbirinden ayırt etmek için kullanılır. Ayırt etme işlemi bir sütun üzerinden yapılabileceği gibi birden fazla sütunda bu işlem için kullanılabilir. Kayıtlar içerisinde benzersiz olan sütun ilk anahtardır. Öğrenci bilgilerinin tutulduğu bir tabloda öğrencilerin birbirinden ayırt edilmesini sağlayan ne öğrencinin adı ne de sınıfıdır. Bizim için burada benzersiz olan bilgi öğrenci numarasıdır. Bir okuldaki öğrencilerin numaralarının aynı olma şansı yoktur. Bu örneğimiz için ilk anahtar sütun(Primary Key) öğrenci numarası olabilir. Tablolarda anahtar sütun tanımlama zorunluluğu yoktur ama kullanılması tavsiye edilmektedir.

Bir tablo içerisinde Birincil Anahtar(Primary Key), Tekil Anahtar(Unique Key) ve Yabancı Anahtar(Foreign Key) olmak üzere üç farklı anahtar tanımlanabilir.

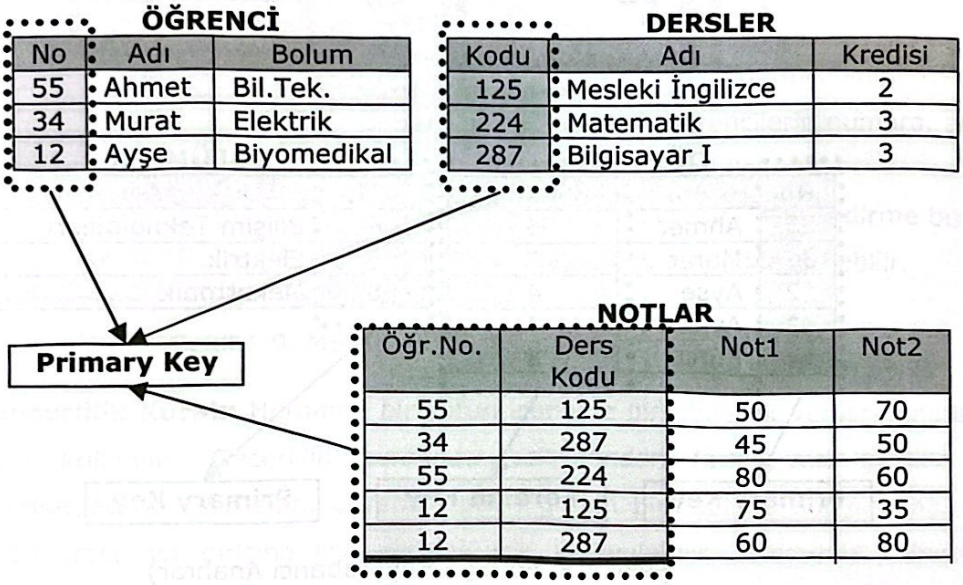
e) **Primary Key(Birincil Anahtar):** Tabloda tutulan verilerden benzersiz yani aynı değeri iki kez içermeyecek olan sütun birincil anahtar olarak belirlenir.

Birincil anahtar olan sütun NULL(boş) değerleri veya birbirinin aynı olan değerleri içeremez. Birincil anahtar birden fazla alanın birleşiminden oluşmuşsa alanların her biri aynı kaydı içerebilir ama alanların hiçbiri NULL değer içeremez. Birincil anahtar, kayıt silme ve düzeltme gibi işlemlerde çok büyük kolaylık sağlar. Silinmek istenen kaydın birincil anahtar olan sütun değeri verildiğinde başka sütunların kontrol edilmesine gerek yoktur. Örneğin, 115 numaralı öğrencinin kaydını silmek istersek, no sütunu birincil anahtar

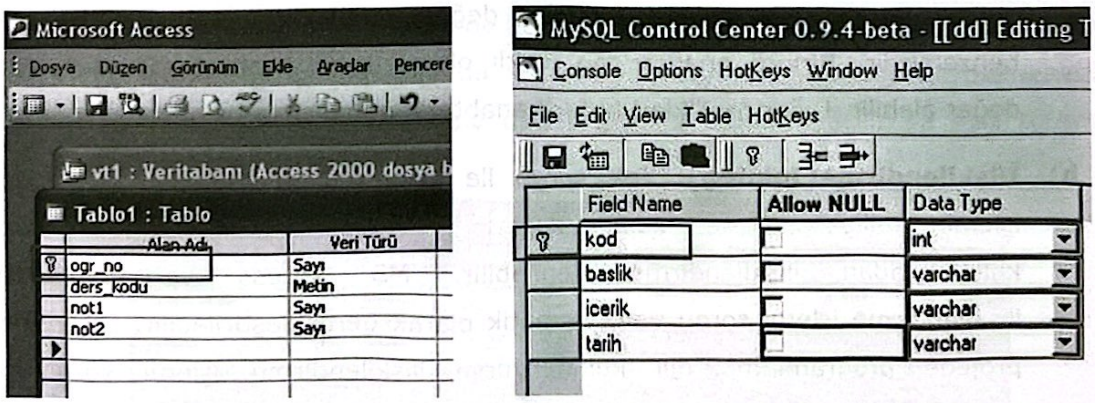
Scanned with CamScanner

olduğu için 115 nolu tek kayıt vardır ve yanlış kaydın silinmesi söz konusu değildir.

Örneğin, öğrenci tablosunda her bir öğrenci numarası tek bir öğrenciye ait olacağı için bu sütun birincil anahtar olarak belirlenebilir. Tablo içerisindeki sütunlardan bir tanesi veya birden fazla sütunun birleşimi birincil anahtar olarak kullanılabilir. Birincil anahtar olan sütuna aynı değer iki kez girilmek istendiğinde veya NULL(boş değer) değer içerdiğinde kayıt işlemi gerçekleşmeyecektir.



Şekil 1.6. Primary Key(Birincil Anahtar)



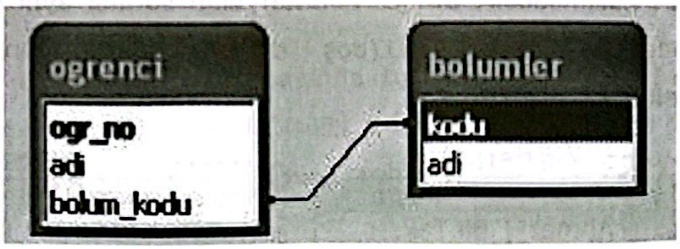
Şekil 1.7. Birincil anahtar sütunun yanında anahtar işareti bulunur.

f) **Foreign Key(Yabancı Anahtar):** Yabancı anahtar bir sütun veya birden fazla sütunun birleşiminden oluşabilir. Birincil anahtar veya Tekil Anahtarda olduğu gibi sütunu Yabancı Anahtar olarak belirleme söz konusu değildir. İlişki kurulmadığı sürece kullanılamaz. Yabancı Anahtar sütunu aynı tablo veya başka bir tablodaki birincil anahtar olan bir sütunla ilişkilendirilebilir. Birbiri arasında ilişki bulunan iki tablodan birisinden diğer tablodaki birincil anahtara

Scanned with CamScanner

başvuran sütun yabancı anahtar olarak adlandırılır. Yabancı anahtar olan sütun başvurduğu birincil anahtar olan sütunun değerlerini içermelidir ve NULL olmamalıdır ama aynı değeri içeren birden fazla kayıt olabilir.

Bir tabloya girilebilecek değerleri sınırlandırmak için kullanılır. Örneğin not tablosuna, tanımlanmamış bir öğrenci için not girişi bu yöntemle engellenebilir.



ÖĞRENGİ

No	Adı	Böl.Kodu
55	Ahmet	5
34	Murat	5
12	Ayşe	4
63	Ayhan	4
14	Burcu	8

BÖLÜMLER

Kodu	Adı
4	Bilişim Teknolojileri
5	Elektrik
8	Mekatronik

Primary Key | Foreign Key | Primary Key

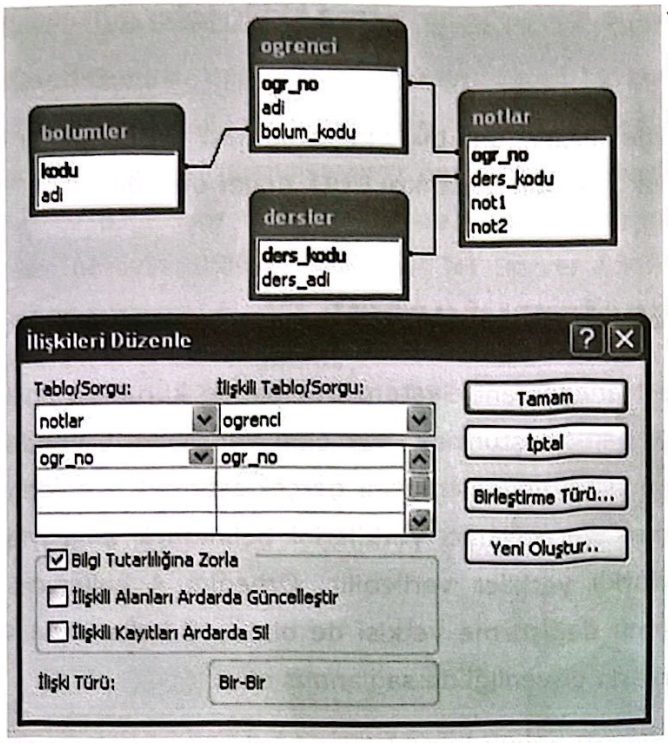
Şekil 1.8. Foreign Key(Yabancı Anahtar)

g) **Unique Key(Tekil Anahtar):** Tekil anahtar olarak tanımlanan sütunlar birincil anahtar da olduğu gibi bir değer yalnız bir kez girilebilir yani benzersizdir. Birincil anahtar'dan farklı olarak tekil anahtar sütunlar NULL değer alabilir. Diğer özellikleri birincil anahtarla aynıdır.

h) **İlişkilendirme(Joining):** Tek sorgu ile birden fazla tablodan bilgi alma işlemine ilişkilendirme denilir. Bazı veritabanı yazılımlarında sorgu kullanılmadan ilişkilendirme yapılabilir. MS Access veritabanlarında ilişkilendirme işlemi sorgu veya şematik olarak gerçekleştirilebilir. Hazırlanan projede programlama dili kullanılıyorsa ilişkilendirme işlemi sorgu ile yapılmalıdır. Öğrenci ve bölüm bilgilerinin ayrı ayrı tablolarda tutulduğu bir veritabanında öğrenci bilgileri ve öğrencinin kayıtlı olduğu bölümün adını öğrenmek için ilişki kurulması gerekmektedir.

İlişkilendirme ayrıca veri tutarlılığı sağlamak için de kullanılmaktadır. Örneğin, bölüm tablosunda olmayan bölüm kodunu öğrenci tablosunda kullanılmasının engellenmesi, öğrenci tablosundan öğrenci silindiğinde diğer tablolarda o öğrenciye ait tüm bilgilerin silinmesi veya değiştirilmesi için kullanılabilir.

Scanned with CamScanner



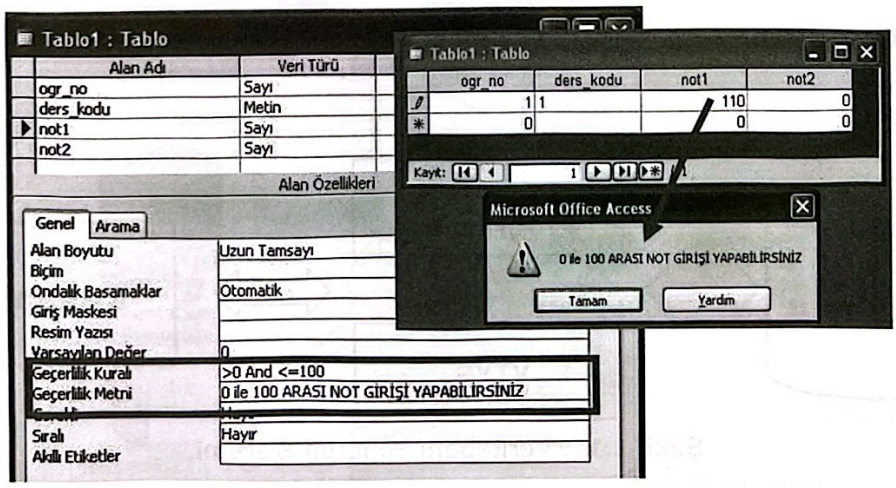
Yandaki şeklide MS Access'te hazırlanmış veritabanı ve öğrenci, notlar, dersler ve bölümler tabloları ile bunlar arasındaki ilişkiler gösterilmiştir. İstenirse ilişkiler düzenlenerek veri tutarlılığı sağlanabilir.

"Select ogrenci.ogr_no, ogrenci.adi, bolumler.adi from ogrenci,bolumler where ogrenci.bolum_kodu=bolumler.kodu " sorgusu öğrencilerin numara, ad ve bölüm adı bilgilerini göstermektedir. ilişkilendirme bu şekilde de gerçekleştirilebilir.

Şekil 1.9. MS Access İlişkilendirme Örneği

i) **Geçerlilik Kuralı:** Herhangi bir sütun içerisine girilebilecek verileri sınırlamak için kullanılır. Geçerlilik kuralında belirtilenden farklı veri girişine izin vermeyecektir. Örneğin, öğrenci notları girilen bir tabloda not alanına, 0 ile 100 arası not girişine izin vermek için kullanılabilir. İsterseniz kullanıcının NULL değer girişini de engelleyebilirsiniz.

Belirlenecek kurallar ile veritabanına yanlış bilgi girişi engellenmiş olur. Kullanıcı yanlış bilgi girdiğinde hata ile karşılaşır. Bazı veritabanı yazılımlarında yanlış veri girişi yapıldığında kullanıcıya verilecek hata mesajı belirlenebilir. Böylece kullanıcının rastgele değer girişinin önüne geçilir ve veri tutarlılığı sağlanmış olur.



Sekil 1.10. MS Access Gecerlilik Kuralı

Scanned with CamScanner

j) **NULL Değer:** NULL değer, tablo içerisinde bir sütuna hiçbir değer girilmediği durumlarda oluşur.

Sütun içerisinde görünürde hiçbir bir bilginin olmaması NULL değer olduğunu göstermez. Boşluk tuşunun kullanımı NULL değer değildir.

1.4. VERİTABANI YÖNETİM SİSTEMİ(VTYS)

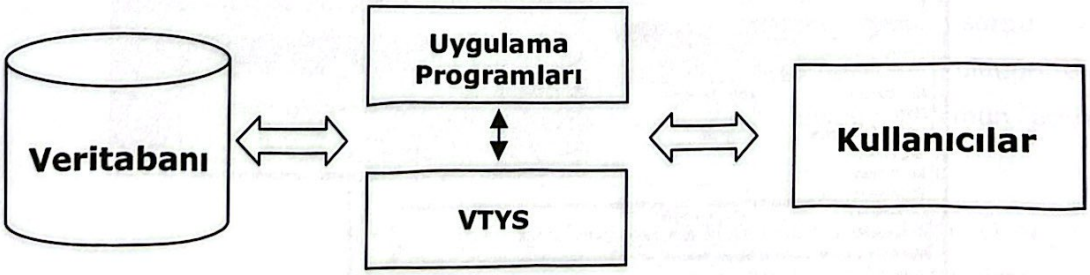
Veritabanı yönetim sistemi (Database Management System-DBMS) ise kuruluşla ilgili verileri tutmak amacıyla yeni bir veritabanı oluşturmak, var olan veritabanı üzerinde değişiklik yapmak, veritabanının bakımını ve yedeklemesini gerçekleştirmek, verilerin tutulduğu tablolar arası ilişkileri kurmak ve kullanıcı yetkilerini belirlemek amacıyla kullanılan yazılımlardır. Kullanıcılara farklı yetkiler verilebilir. Örneğini A kullanıcısı sadece verileri görebilsin, B kullanıcısının değiştirme yetkisi de olsun, C kullanıcısının tüm yetkileri olsun gibi. Bu sayede verilerin güvenliği de sağlanmış olur.

VTYS basit bir web sayfasından tutunda çok büyük kuruluşların verilerinin tutulmasına kadar birçok alanda kullanılmaktadır. Veritabanı yönetim sistemleri işletim sistemleri kadar popüler ve olmazsa olmazlardandır.

Farklı isimler altında da olsa aynı prensiple çalışan birçok VTYS yazılımı mevcuttur. Bunların bir kısmı lisans gerektirirken bir kısmı da ticari olmamak kaydıyla ücretsiz olarak kullanılabilir.

VTYS 'ler sadece verinin tutulması ve yönetilmesi için kullanılmazlar. Aynı zamanda farklı kullanıcıların farklı yetkilerle veritabanına bağlantısını sağlar. Yani kullanıcı izinlerini de kontrol eder. Günümüzde yaygın olarak ilişkisel veritabanları kullanılmaktadır. İlişkisel kavramı hakkında ilerleyen konularda ayrıntılı bilgi verilecektir.

MySQL, Oracle, MS SQL, Firebird ve MS Access yaygın olarak kullanılan VTYS yazılımlarıdır.

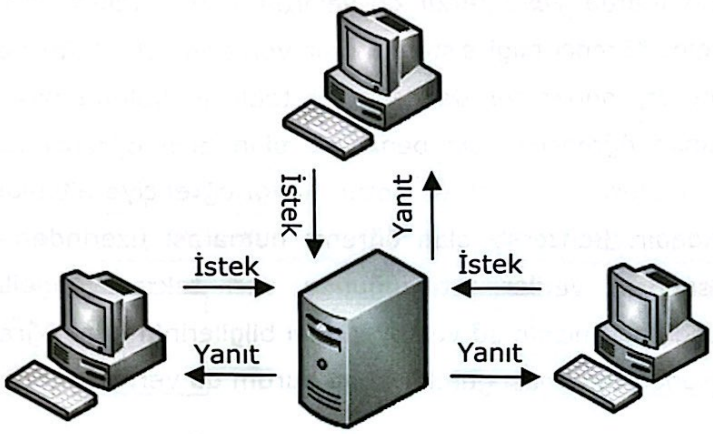


Şekil 1.11. Veritabanı Yönetim Sistemi.

Scanned with CamScanner

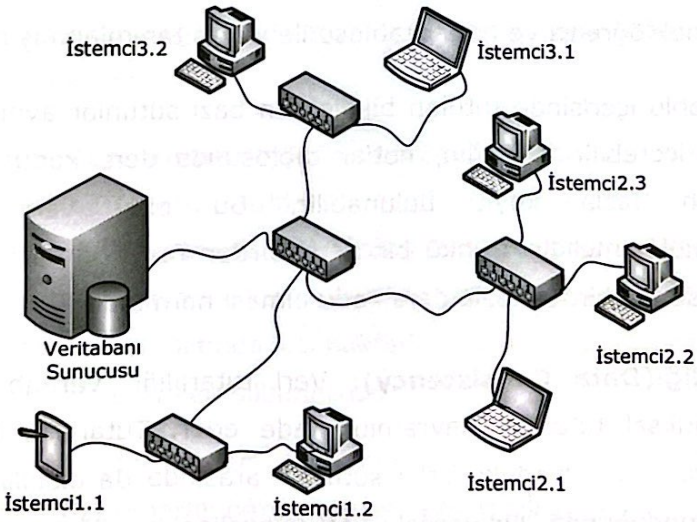
1.5. İSTEMCİ-SUNUCU(CLIENT-SERVER) MİMARİSİ

İstemci-Sunucu mimarisinde, hizmet veren bir sunucu ve hizmet alan bir veya birden fazla istemci bulunur. İstemciler sunucuya bağlanarak vermiş olduğu hizmetlerden yararlanır. Sunucu bilgisayarın performansı istemcilerden daha iyi olmak zorundadır. Sunucu bilgisayar dosya, e-posta, web hizmetleri, veritabanı gibi görevleri sunmak üzere görevlendirilirler. Windows NT Server / Windows 2000 Server / Windows 2003 Server / Windows 2008 Server, Linux, Novell Netware işletim sistemleri sunucularda çoğunlukla kullanılmaktadır.



Şekil 1.12. İstemci-sunucu mimarisi.

Günümüzde kullanılan veritabanı sistemlerinin büyük çoğunluğu ve özellikle web tabanlı sistemler istemci-sunucu mimarisini kullanmaktadır. Örneğin, bir muhasebeci bünyesinde birden fazla bilgisayar kullanıyorsa her bilgisayarda tutulan verilerin aynı olması gerekir. Bu durumda verilerin merkezi bir yerde depolanması gerekir. Bilgilerin depolandığı merkezi bilgisayar sunucu olarak hizmet vermektedir. Diğer kullanıcı bilgisayarları veritabanından bilgi okumak, veritabanına yeni bilgi yazmak gibi işlemleri sunucu üzerinden gerçekleştirecektir ve bu bilgisayar istemci ismini alacaktır.



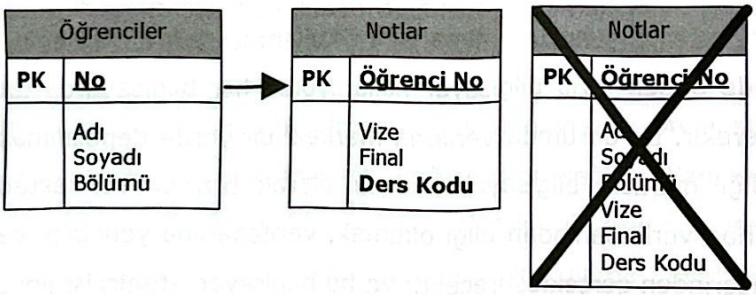
Şekil 1.13. Örnek bir öğrenci bilgi sisteminin veritabanı alt yapısı.

Scanned with CamScanner

1.6. VERİTABANI YÖNETİM SİSTEMLERİNİN SAĞLADIĞI YARARLAR

- **Veri Tekrarı(Data Redundancy):** Hazırlanan uygulamaların işlediği bilgilerin tamamı doğrudan veya dolaylı olarak birbiriyle ilişkili olduğu için uygulamaya ait veriler tek bir veritabanında birden fazla tablo içerisinde tutulur. Böylece bir tablo içerisine girilen bir bilginin diğer tabloya tamamen girilmesi engellenerek veri tekrarı önlenmiş olur. Aynı verinin farklı bilgisayarlarda veya farklı dosyalarda tekrar tekrar tutulmasını engeller. Örneğin, öğrenci bilgi sistemine ait verilerin tutulduğu veritabanı içerisinde öğrenciler, notlar ve dersler gibi tablolar bulunmaktadır. Bu veritabanı içerisinde öğrenciler için benzersiz olan bilgi öğrenci numarasıdır. Ayrıca notlar tablosunda belirtilen notun hangi öğrenciye ait olduğu da belirtilmek zorundadır. Benzersiz olan öğrenci numarası üzerinden öğrenci ve notlar tablosundaki veriler tutulduğunda veri tekrarı engellenmiş olur. Aksi durumda öğrencinin ad ve soyad gibi bilgilerinin hem öğrenci hem de notlar tablosunda tutulması gerekir ki bu durum da veri tekrarına sebep olacaktır.

  Tablolar içerisinde veri tekrarının önlenmesi aynı zamanda veri bütünlüğünün sağlanmasına da yardımcı olur.



Şekil 1.14. Örnek öğrenci ve notlar tablosu ile yanlış tanımlanmış notlar tablosu.

Bir tablo içerisinde tutulan bilgilerden bazı sütunlar aynı değerleri birden fazla içerebilir. Örneğin, notlar tablosunda ders kodu bilgisi aynı olan birden fazla kayıt bulunabilir. Bu bilgi veri tekrarı olarak düşünülmemelidir. Çünkü bir dersi birden fazla öğrenci aldığı için notlar tablosunda birden fazla ders kodu olması normaldir.

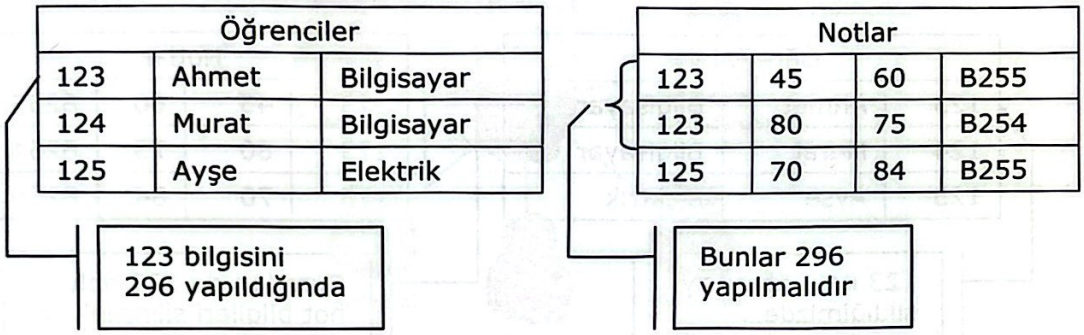
- **Veri Tutarlılığı(Data Consistency):** Veri tutarlılığı, veritabanı sistemlerinde verilerin mantıksal tutarlılık kavramını ifade eder. Tutarlılık, tablolar arasında olabileceği gibi aynı tablodaki farklı sütunlar arasında da olabilir. Aynı veri farklı tablolarda bulunduğunda, bir yerdeki veri güncellenirse diğer tablodaki verinin de

Scanned with CamScanner

güncellenmesi gerekir. Bu işlem gerçekleşmezse veritabanı içerisinde veri tutarsızlığı oluşacaktır.

Örneğin, öğrenci bilgi sisteminde bulunan öğrenci tablosundan bir öğrencinin no bilgisi güncellendiğinde, öğrenci numarasını barındıran tüm tablolardaki no bilgisi de güncellenmelidir. Aksi takdirde daha sonraki erişimlerde öğrenci bilgilerine erişilemeyebilir.

Şekil 1.15'de gösterildiği gibi öğrenci numarası 123 olan öğrencinin numarası 296 yapıldığında notlar tablosunda ki 123 olan öğrenci no bilgileri 296 yapılmalıdır.



Şekil 1.15. Veri tutarlılığı.

- **Veri Paylaşımı(Data Concurrency):** Veritabanı yönetim sistemleri istemci sunucu mimarisini kullanarak tek bir veritabanını ağ üzerinden birden fazla istemci için kullanıma açabilir ve bağlantı kuracak istemci sayısı istenilirse sınırlandırılabilir.

  VTYS kullanılmadan veritabanına aynı anda birden fazla erişim sağlanmak istendiğinde veriye sıralı erişim sağlanır ve birden fazla kullanıcı aynı veriye aynı anda erişemez. VTYS kullanıldığında ise aynı anda birden fazla kullanıcı aynı veriye erişebilir. VTYS aynı anda birden fazla kullanıcı aynı veri üzerinde değişiklik yapmak istediğinde önceliği yetki üstünlüğüne göre veya bağlantı önceliğine göre belirler diğerini bekletir.

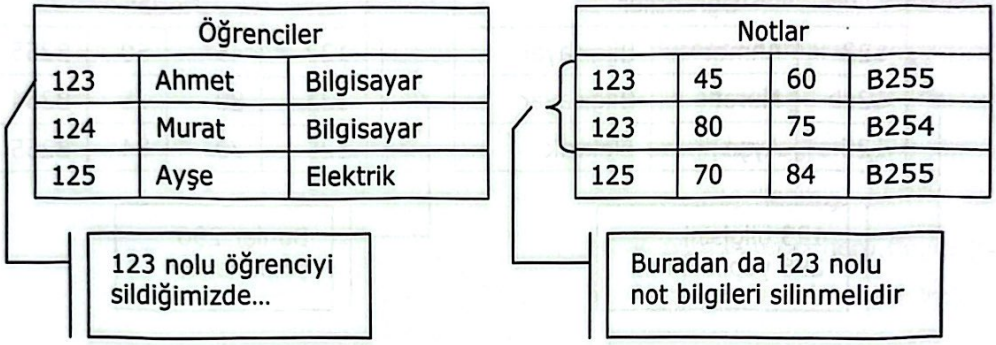
- **Veri Bütünlüğü(Data Integrity):** Veritabanlarında bir veri birden fazla tabloda tutulabilir. Örneğin, öğrenci numarası hem öğrenci hem de notlar tablosunda kullanılır. Veri bütünlüğü, bir verinin birden fazla parçaya bölündüğü durumlarda, tüm verilerin bir bütün olarak kullanılmasını sağlar. Veri bütünlüğü özellikle kayıt silme işlemlerinde tehdit altında kalmaktadır. Bir veri silindiğinde o verinin ilişkili olduğu tüm tablolardan o veri silinmelidir.

  Örneğin, öğrenci tablosundan 110 numaralı bir öğrenci silindiğinde notlar tablosundan da 110 numaralı öğrenciye ait bilgiler silinmelidir.

Scanned with CamScanner

Örneğin, ürün ve stok bilgilerinin tutulduğu bir veritabanında ürünler tablosundan A45 kodlu ürün silindiğinde, stok tablosundan da A45 koda sahip stok bilgileri silinmelidir. Aksi durumda zamanla veritabanı gereksiz yere aşırı şekilde büyüyecek ve beraberinde depolama, erişim hızı gibi problemleri doğuracaktır.

Veritabanlarında veri bütünlüğünü sağlamak için ya programlama aşamasında ya da veritabanı içerisinde gerekli düzenlemeler yapılır. Programlama aşamasında, her yapılan veritabanı işlemi kodlarla denetlenerek bütünlük sağlanır. VTYS sistemlerde ise çeşitli kısıtlayıcılar(birincil anahtar, yabancı anahtar, tekil anahtar, geçerlilik kuralı vb.) ile veri bütünlüğü sağlanır.



Şekil 1.16. Veri bütünlüğü.

Veri bütünlüğü ve veri tutarlılığı teorik olarak farklı düşünülse de her ikisinin de temel amacı veritabanı içerisindeki verilerin sağlıklı bir şekilde depolanması için veri bütünlüğünü sağlamaktır.

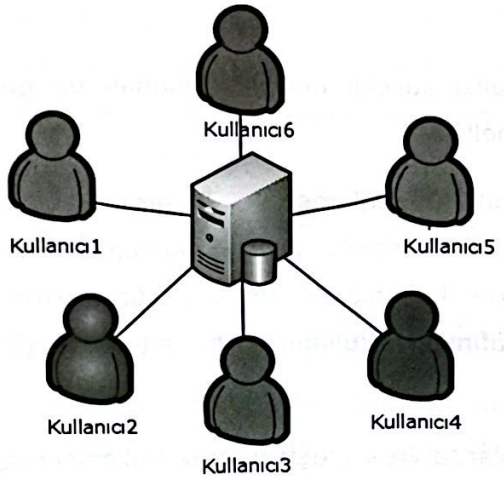
Örnek olarak verilen öğrenci ve notlar tablosunun tam tersi durumu yani notlar tablosundan 123 nolu bilgiler silindiğinde öğrenciler tablosundan da 123 nolu öğrencinin silinmesi veri bütünlüğü sağlamak yerine veri kaybına sebep olacaktır. Bu gibi durumlarda mantıksal veya ilişkisel bağlantılar dikkate alınmalıdır.

- **Veri Güvenliği(Data Security):** VTYS veritabanı yönetiminin yanı sıra kullanıcıların veritabanlarına erişimlerini de düzenleyerek verinin güvenliğini sağlamaktadır. Kullanıcılar verilen yetki ve sorumluluk dahilinde verileri görebilme, yetkisiz kişilerin girişini engelleme, yedekleme ve bakım işlemlerini gerçekleştirebilir. Örneğin öğrencinin sadece notlarını görmesini sağlayıp diğer bölümlere girişi engellenebilir veya ABC kullanıcısının sadece kayıtları görme yetkisi verilip diğer yetkileri engellenebilir.

Şekil 1.17'de verilen kullanıcı ve veritabanı resminin örnek yetkileri aşağıda verilmiştir.

Scanned with CamScanner

Kullanıcı Adı	Okuma	Ekleme	Değiştirme	Silme
Kullanıcı1	Evet	Hayır	Hayır	Hayır
Kullanıcı2	Evet	Evet	Evet	Evet
Kullanıcı3	Hayır	Evet	Hayır	Hayır
Kullanıcı4	Evet	Evet	Evet	Hayır
Kullanıcı1	Hayır	Hayır	Hayır	Hayır
Kullanıcı6	Evet	Hayır	Evet	Hayır



Şekil 1.17. Örnek kullanıcı ve veritabanı şeması.

Tabloda verilen kullanıcı yetkilendirmeleri en genel hatlarıyla gösterilmiştir. İstenilirse, kullanıcının belirli tabloları görmesini sağlayıp diğerlerini görmesi engellenebilir veya bir tablo içerisinde belirli sütunları görmesi sağlanarak yetkilendirmeler daha özelleştirilebilir.

- **Veri Bağımsızlığı(Data Independence):** Veritabanın fiziksel yapısı kullanıcılardan gizlenir. VTYS, yüksek düzey programlama dilleri gibi davranarak veritabanının yapısını kullanıcıya göstermeden çeşitli terimlerle veritabanı üzerinde işlem yapılmasını sağlar. Böylece kullanıcı, karmaşık veri gösterimleri ve algoritmalarla ilgilenmeden isteklerini kolayca gerçekleştirebilir.

1.7. VERİTABANI KULLANICILARI

1.7.1. Veritabanı Yöneticisi(DBA-Database Administrator)

Kullanılacak olan veritabanının tasarımı, oluşturulması ve işletiminden birinci derecede sorumlu olan ve veritabanı üzerinde en fazla yetkiye sahip olan kullanıcıdır.

Veritabanı yöneticisinin sürekli olarak gerçekleştirmesi gereken çeşitli görevleri vardır. Bunlar;

Scanned with CamScanner

- Veritabanı Tasarımı: DBA veritabanı oluştururken, oluşturacağı veritabanın hem kavramsal şemasını hem de fiziksel olarak veritabanını oluşturabilmelidir.

- Veritabanı Performans Analizi: Performans, veritabanını kullanan kurum veya kuruluşun beklentilerini sağlayacak düzeyde olmalıdır. DBA veritabanı ayarları, donanım ve yazlım ihtiyaçlarını belirleyerek veritabanından en yüksek performansın alınmasını sağlamalıdır. Ayrıca performans artışını sağlamak için gerekli indeksleme işlemlerini, ilişkilendirmeleri ve gerekli belleğin ayrılmasını da sağlamalıdır.

  Performans analizi sürekli olarak yapılmalı ve gereken durumlarda anında müdahale edilmelidir.

- Erişim Yetkilerini Düzenleme ve Erişimi Sağlama: Hazırlanan veritabanı kullanıma açıldıktan sonra gerekli kullanıcıların veritabanına erişebilmesi sağlanmalıdır. Her kullanıcının her kaynağa erişmesi söz konusu olamayacağı için DBA tarafından kullanıcıların erişebileceği kaynaklar ve yetkileri belirlenmelidir.

  Gereken durumlarda view oluşturularak kullanıcı erişimleri de belirlenebilir. Bu sayede erişim yetkileri belirlenirken veritabanının yetkisiz erişimlere karşı da güvenliği sağlanmış olur.

- Yedekleme ve Geri Yükleme: DBA'nın en önemli görevlerinden birisi de tutulan verilerin her türlü soruna karşı güvenliğini sağlamaktır. Güvenliği tehdit edici unsurlar kullanıcı kaynaklı olabileceği gibi yazılımsal sorunlarda olabilir. Bunun için yedekleme ve geri yükleme süreçleri belirlenmelidir.

  Sistem yedeklerinin alınması, yedekleme ortamlarının güvenliğinin sağlanması ve güvenilirlikten emin olunmalıdır. Alınan yedekler farklı bir sistem üzerinde geri yüklenerek çalışabilirliği test edilmelidir.

- Veri Bütünlüğünü Sağlama: DBA veritabanını tasarlarken veri kaybı oluşmaması için gerekli kuralları ve ilişkileri belirlemelidir. Ayrıca, kullanıcıdan kaynaklanabilecek veri bütünlüğünü tehdit edici unsurlar dikkate alınmalıdır.

  Veri bütünlüğünün sağlanması için gerekli kurallar ilerleyen bölümlerde daha ayrıntılı verilecektir.

- Sistemin Sürekliliğini Sağlama: Veritabanı, VTYS veya kullanılan sistem üzerinde meydana gelebilecek sorunlara karşı hazırlıklı olunmalıdır. Veritabanının bulunduğu sistem üzerinde meydana gelebilecek bir sorun veritabanını kullanan kurum veya kuruluştaki tüm sistemi doğrudan

Scanned with CamScanner

etkileyecektir. Bu yüzden, sistemde bir sorun meydana geldiğinde sistemin sürekliliğini sağlamak DBA 'nın ve sistem yöneticisinin sorumluluğundadır. Ayrıca, DBA oluşabilecek sorunları daha önceden tespit edebilmelidir. Genellikle bu tür sorunlara karşı hazırlıklı olan kurum veya kuruluşlar sorun durumunda devreye girebilecek yedek bir sistem bulundururlar.

1.7.2. Uygulama Programcısı

Uygulama programcıları veritabanlarını doğrudan kullanmazlar. Temel görevleri, veritabanlarını kullanabilecek son kullanıcılara yönelik yazılımlar geliştirmektir. Kullanıcılar veritabanlarına doğrudan erişmek yerine hazırlanan yazılımlar yardımıyla gerekli giriş çıkış işlemlerini gerçekleştirirler.

Uygulama programcıları temel olarak veri işleme dilini ve yazılım geliştirme araçlarını kullanır.

1.7.3. Sorgu Dili Kullanıcıları

Sorgu dili kullanıcıları, mevcut veritabanı sorgulamaları dışında ihtiyaç duyulan sorguları hazırlamak ve işletmekten sorumludur. Bu işlem için SQL ifadeleri kullanılır.

SQL veritabanından bilgi almak, veritabanına bilgi eklemek veya silmek, veritabanı yapısı üzerinde değişiklikler yapmak veya kullanıcı yetkilerini belirlemek için kullanılan bir yapısal sorgulama dilidir.

Bu kullanıcılar veritabanı üzerinde her türlü sorgu işletme yetkisine sahip değildir. Örneğin, kullanıcı yetkilerini belirlemek için kullanılan SQL ifadeleri bu kullanıcılar tarafından kullanılamaz. Ama veritabanı üzerinde yapısal değişiklikler(tablo silme, tablo oluşturma vb.) yapmak için kullanılan SQL ifadeleri kısmen de olsa kullanabilirler.

Bir sorgu dili kullanıcısı, üzerinde sorgular çalıştıracağı veritabanının yapısını(tablolar, sütunlar, indeksler, ilişkiler vb.) bilmek zorundadır. Çünkü sorgulama işlemlerinde bu tür bilgiler kesinlikle kullanılır.

1.7.4. Son Kullanıcılar

Son kullanıcılar, uygulama programcıları tarafından geliştirilen ve veritabanı üzerinde çeşitli işlemler(kayıt ekleme, silme, raporlama vb.) gerçekleştiren yazılımları kullanır. Bu kullanıcıların veritabanı yapısı hakkında bilgi sahibi olmasına ve üst düzey bilgisayar kullanıcısı olmasına gerek yoktur. Örneğin, bir müşteri takip programı tarafından kullanılan müşteri veritabanına program aracılığıyla bilgi giriş çıkış yapan çalışan, faturaları sisteme giren muhasebe personeli veya öğrenci bilgi sisteminden derslerini ve notlarını gören öğrenci.

Scanned with CamScanner

1.8. VERİTABANI YÖNETİM SİSTEMİ YAZILIMLARI

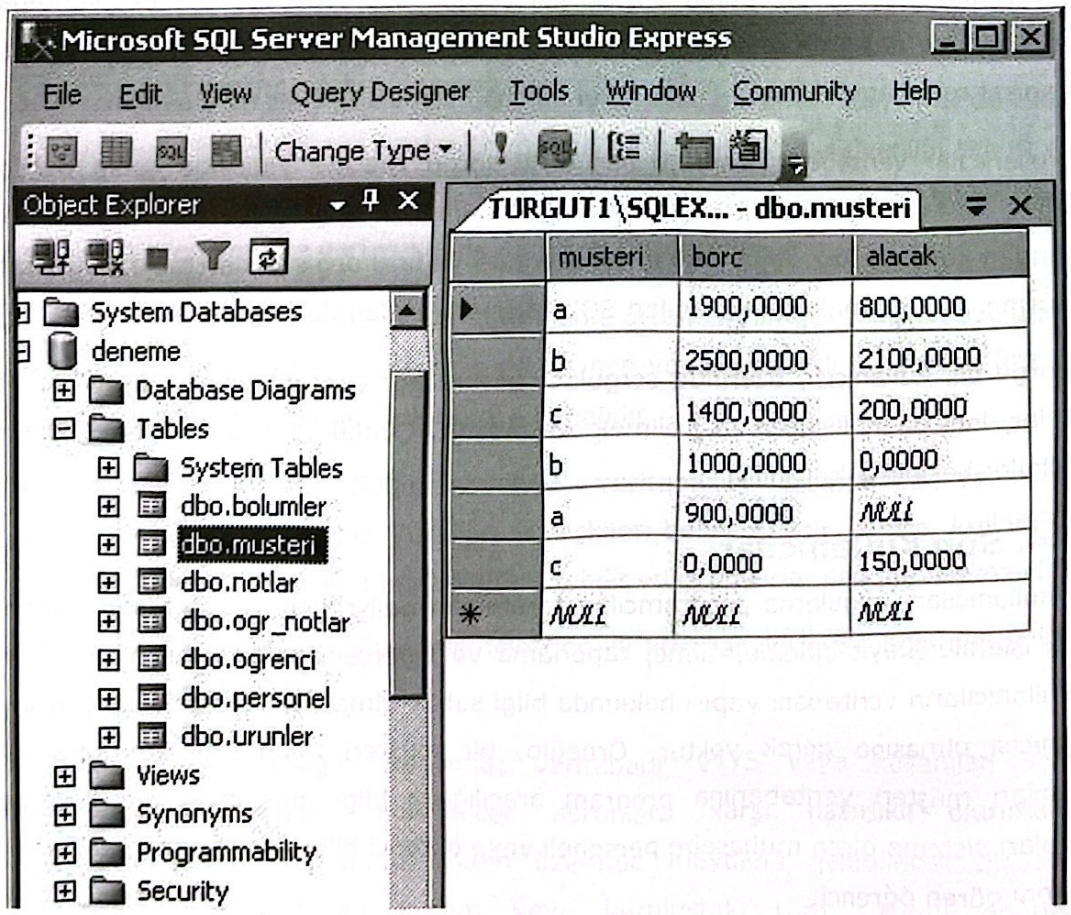
1.8.1. Microsoft SQL Server

Microsoft firmasına ait veritabanı sunucu yazılımıdır. İlişkisel veri tabanı yönetim sistemidir. Veri depolama, gruplama, analiz etme işlevlerine sahiptir ve bu işlevleri kısa sürede gerçekleştirir. Verilerin organizasyonunu merkezi olarak gerçekleştirir.

İstemci(client) uygulamaları ile sunucu üzerindeki verilere ulaşılabilir. Sunucu veya istemci olarak kurulumu gerçekleştirilebilir.

Microsoft SQL Server Management Studio Express programı yazılımını kullanarak Microsoft SQL Server'a kolaylıkla bağlanılabilir ve temel veritabanı işlemleri sorgu yazmadan görsel olarak gerçekleştirilebilir.

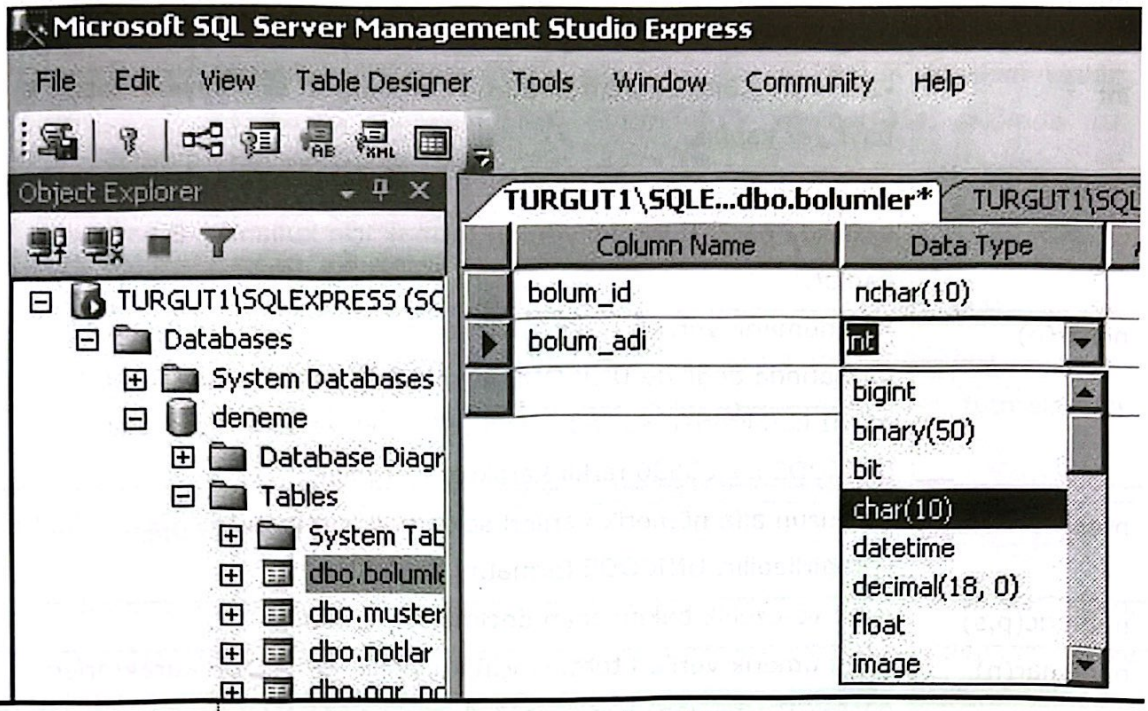
Orta ve büyük ölçekli işlemler için kullanılır. Sadece Windows işletim sistemlerinde çalışır. Maliyeti yüksektir. Tablo başına 4TB'a kadar veri depolayabilir. Veritabanı üzerindeki standart işlemler(seçme, güncelleme, ekleme vb.) SQL ile gerçekleştirilebilirken daha ayrıntılı (koşullu ifadeler, özel fonksiyonlar vb.) işlemler için de T-SQL(Transact-SQL) kullanılabilir. Aşağıdaki şekilde Microsoft SQL Server Management Studio Express programının ekran görüntüsü verilmiştir.



Şekil 1.18. Microsoft SQL Server Management Studio Express

Scanned with CamScanner

Microsoft SQL Server Veri Tipleri



Veri Tipi	Açıklama
bigint	-9.223.372.036.854.775.808 ile 9.223.372.036.854.775.807 arasındaki tam sayıları tutar. 8 bayt yer kaplar.
binary	8000 byte'a kadar sabit uzunlukta binary veri alabilir.
bit	0 veya 1 tam sayı değeri alır. Yani geriye TRUE veya FALSE değeri döndürür. Örneğin cinsiyet sütunu için erkek-bayan, medeni hal sütunu için evli-bekar gibi iki değerli durumlar için kullanılabilir.
char(n)	Alfa nümerik verileri tutar. n karakter sayısını belirler. Bu veri tipi içine girilecek olan karakter sayısı bilindiği olduğu zaman kullanılır. Örneğin, ad sütunu için char(25) şeklinde bir tanımlama ad sütununa en fazla 25 karakter bilgi girilebileceği anlamına gelir. 10 karakterlik bilgi girilirse geriye kalan 15 karakter boşluk karakteri ile doldurulur.
cursor	Tablo yapısında yer almaz. Kayıtlara satır satır erişim için kullanılır.
datetime	01.01.1753 00:00:00 ve 31.12.9999 23:59:59 arasında tarih ve saat tutabilir. 8 bayt yer kaplar.
decimal(p,s)	38 basamak uzunluğuna kadar tam sayı veya virgüllü sayıları tutar. Örneğin; decimal(3,2) olarak tanımlanan bir alanın tam sayı kısmı 3 ondalık kısmı 2 basamaklı olacak demektir.
float(n)	-1,79E + 308 'den 1,79E + 308 'e kadar olan değerleri alabilir. Kapladığı alan "n" değerine göre değişir. (1 ≤n ≤ 53)

Scanned with CamScanner

Veri Tipi	Açıklama
image	2147483647 byte uzunluğunda resim, sunu, excel dosyası gibi ikili formatta veri saklamak için kullanılır.
int	-2.147.483.648 ile 2.147.483.647 arasındaki tam sayıları tutar. 4 bayt yer kaplar.
money	-922,337,203,685,477.5808 ile 922,337,203,685,477.5807 arasında parasal içerikli verileri tutmak için kullanılır. 8 bayt yer kaplar.
nchar(n)	Alfa nümerik verileri tutar. Char formatından farkı verileri ANSI formatında değil de UNICODE formatında saklar. ANSI formatı ASCII karakterleri sakladığı için 256 farklı karakter içerebilir. UNICODE ise 65536 farklı karakter içerebilir.
ntext	Çok uzun alfa nümerik verileri saklamak için kullanılır. 2GB'a kadar veri girilebilir. UNICODE formatını destekler.
numeric(p,s)	İşlev ve özellik bakımından decimal ile aynıdır.
nvarchar(n)	Alfa nümerik verileri tutmak için kullanılır. UNICODE karakterleri de(ASCII olmayan) içerebilir. n değeri yerine MAX yazılırsa 536870912 karakter veya 2GB veri tutabilir.
real	Float(24) ile aynı özelliğe sahiptir. -3.40E+38 ile 3.40E+38 arasındaki sayıları tutar.
rowversion	Veritabanı içerisindeki benzersiz bir numaradır. Bir satır güncellendiğinde bu değerde kendini günceller.
smalldatetime	01.01.1900 00.00.00 ile 31.12.2079 23:59:59 arasında olan tarih ve saat bilgisini tutar. 4 bayt yer kaplar.
smallint	-32.768 ile 32.767 arasındaki tam sayıları tutar. 2 bayt yer kaplar.
smallmoney	–214748.3648 ile 214748.3647 arasında parasal içerikli verileri tutmak için kullanılır.
sql_variant	8000 bayta kadar gireceğimiz değerin çeşidini bilmediğimiz sütunlarda kullanabileceğimiz bir veri tipidir. Örneğin, resim veya veri türünün bilinmediği durumlar.
text	Çok uzun alfa nümerik verileri saklamak için kullanılır. 2GB'a kadar veri girilebilir.
tinyint	0 ile 255 arasındaki tam sayı tutar. 1 bayt yer kaplar.
unique-identifier	Her çalışmada benzeri olmayan 16 byte uzunluğunda hexadecimal sistemde sayı üretir. Benzersiz değerler elde etmek için kullanılabilir.
varbinary	8000 byte'a kadar değişken uzunlukta binary veri alabilir.

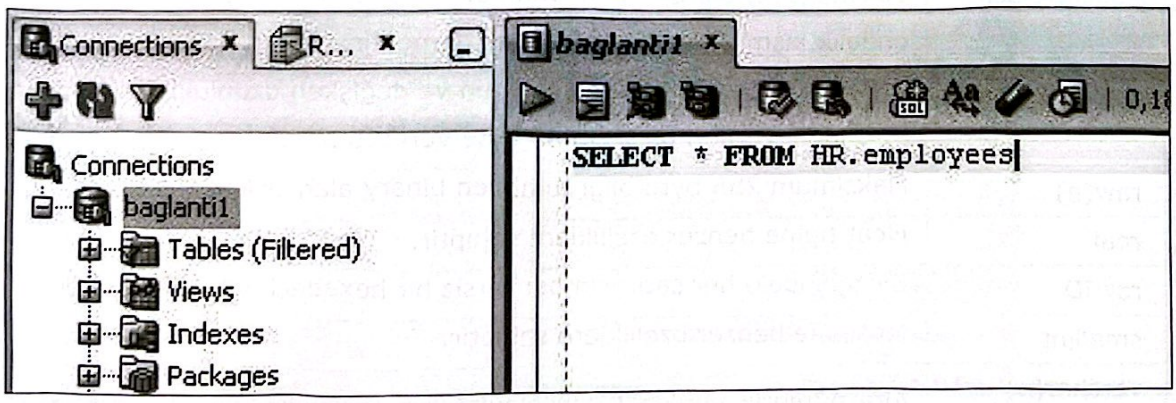
Scanned with CamScanner

varchar(n)	Alfa nümerik verileri tutan alanlar için kullanılır. char formatından farkı uzunluğun(n) değişken olmasıdır. Bu veri tipi girilen değerin uzunluğu değişken olduğunda kullanılır. n değeri maksimum uzunluk değeridir. Veri daha kısa ise uzunluğun gereken kadarı kullanılır. Örneğin, ad sütunu için varchar(25) şeklinde bir tanımlama yapılır ve veri olarak 15 karakterlik bir veri girilirse 25 yerine 15 karakterlik değer kullanılır. Yani, char formatında geriye kalan 10 karakter boşluk ile doldurulurken bu veri tipinde 15 karakterden sonra boşluk eklenmez. n yerine MAX ifadesi kullanılırsa 1 073 741 824 karakter veya 2GB veri tutabilir.
XML	En fazla 2GB uzunluğunda XML türünde verileri tutmak için kullanılır.

1.8.2. ORACLE

Oracle firmasına ait veritabanı sunucu yazılımıdır. Gelişmiş bir ilişkisel veritabanı yönetim sistemidir. Büyük miktardaki verileri çok kullanıcılı ortamlarda depolamayı ve güvenli bir şekilde erişimi yönetir. Özellikle kurumsal alanlarda kullanılmaktadır. Dünyanın en güçlü ve en güvenilir veritabanı olarak gösterilmektedir. Çok yüksek ölçekli uygulamalar için tercih edilir. Maliyet çok yüksektir. Birçok işletim sistemi üzerinde kullanılabilir. Oluşturulabilecek tablo sayısı sınırsızdır. Veritabanı üzerindeki işlemler PL/SQL sorgulama dili ile gerçekleştirilir. PL/SQL(Procedural Language/Structured Query Language), oracle tarafından geliştirilen oracle veritabanı sistemlerine özel bir dildir. Temel SQL komutlarının yanında programlamada akış kontrollerini ve değişkenleri kullanmaya olanak sağlayan yani yapısal dillere ait özelliklerin standart SQL'e eklenmesi sonucu oluşan bir dildir.

Oracle veritabanına erişebilmek için Apex, Oracle SQL Developer, JDeveloper gibi istemci uygulamaları kullanılır.



Şekil 1.19. Oracle SQL Developer ekran görüntüsü.

Scanned with CamScanner

ORACLE Veri Tipleri

ORACLE Veri Tipi	Açıklama
bfile	4GB boyutunda binary veri tutmak için kullanılır.
char(n) character(n)	Alfa nümerik verileri tutar. n karakter sayısını belirler. Sabit uzunluktaki alfa nümerik verileri tutar. Bu veri tipi içine girilecek olan karakter sayısı kesin belli olduğu zaman kullanılır. Örneğin, ad sütunu için char(25) şeklinde bir tanımlama ad sütununa en fazla 25 karakter bilgi girilebileceği anlamına gelir. 10 karakterlik bilgi girilirse geriye kalan 15 karakter boşluk karakteri ile doldurulur. Oracle 7 ve daha önceki sürümler için bu alanın uzunluğu en fazla 255 karakter olabilir. Oracle 8 ve daha sonrasında 2000 karakter uzunluğundadır.
date	Tarih ve saat bilgileri tutulabilir. Tarih formatları Oracle yüklenirken seçilen dile göre değişir. 01.01.4712(M.Ö) 00:00:00 ile 12.31.9999(M.S) 00:00:00 arasında tarih saat bilgisi alabilir.
integer(n)	38 basamağa kadar işaretli ve işaretsiz tamsayıları tutar.
long	2 GB'a kadar bilgi tutabilen karakter alanlar için kullanılır. Bir tabloda bu tipten ancak bir adet alan tanımlanabilir. Long veri tipine sahip alanlar için index oluşturulamaz.
long raw	2 GB'a kadar binary bilgi tutabilen alanlar için kullanılır.
nchar(n)	2000 byte sabit uzunlukta ve UNICODE karakterler içerebilir.
number(p,s) numeric(p,s)	Tamsayı ve virgüllü sayılar için kullanılan sayısal veri tipidir. Tam kısım(p) en fazla 38 basamak olabilir. Ondalık kısmın(s) basamak sayısı da -84 ile 127 arasında değişmektedir. Örneğin; decimal(3,2) olarak tanımlanan bir sütunun tam sayı kısmı 3 ondalık kısmı 2 basamaklı olacak demektir.
nvarchar2(n)	UNICODE karakterleri destekleyen ve değişken uzunlukta karakter türündeki verileri tutar. 4000 byte veri tutabilir.
raw(n)	Maksimum 255 byte bilgi tutabilen binary alanlar için kullanılır.
real	Float tipine benzer özelliklere sahiptir.
rowID	Bir tablodaki her satır için benzersiz bir hexadecimal değer sunar.
smallint	int tipine benzer özelliklere sahiptir.
varchar(n)	Alfa nümerik verileri tutan alanlar için kullanılır. char formatından farkı uzunluğun(n) değişken olmasıdır. Bu veri tipi girilen değerin

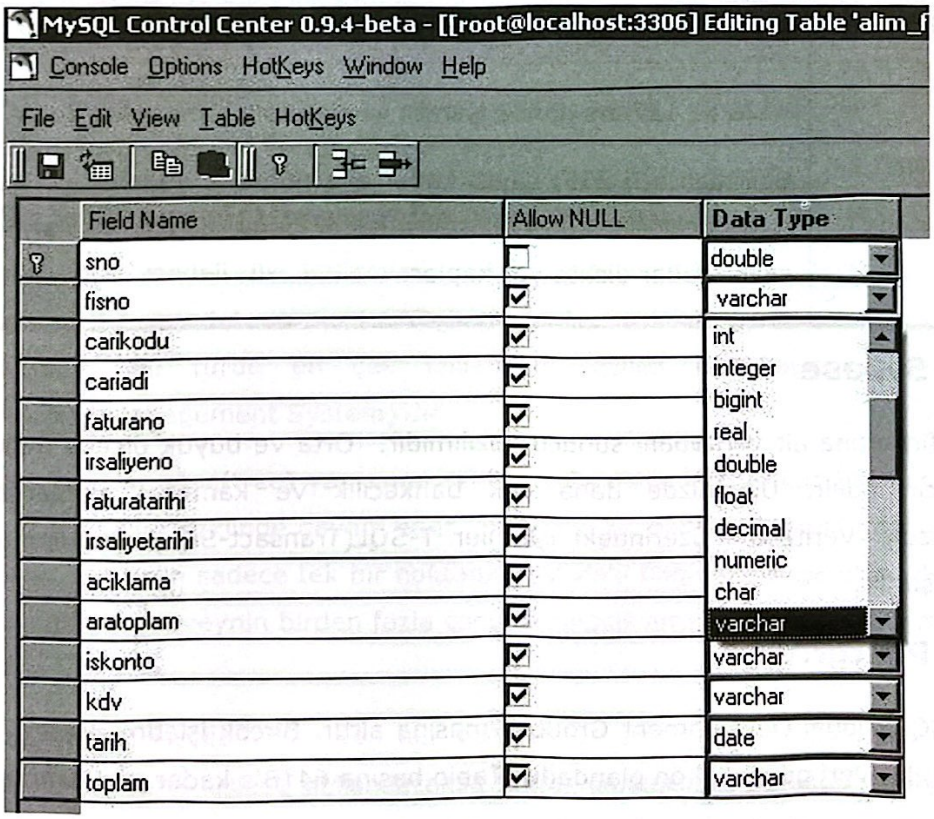
Scanned with CamScanner

	uzunluğu değişken olduğunda kullanılır. n değeri maksimum uzunluk değeridir. Veri daha kısa ise uzunluğun gereken kadarı kullanılır. Örneğin, ad sütunu için varchar(25) şeklinde bir tanımlama yapılır ve veri olarak 15 karakterlik bir veri girilirse 25 yerine 15 karakterlik değer kullanılır. Yani, char formatında geriye kalan 10 karakter boşluk ile doldurulurken bu veri tipinde 15 karakterden sonra boşluk eklenmez.
varchar2	4000 karakter ve değişken uzunlukta karakter tutar.
XML	En fazla 4GB uzunluğunda XML türünde verileri tutmak için kullanılır.

1.8.3. MySQL

Açık kaynak kodlu bir veritabanı yönetim sistemidir. Unix, OS/2 ve Windows işletim sistemleri için ücretsiz dağıtılmakla birlikte ticari lisans kullanmak isteyenler içinde lisanslı seçeneği mevcuttur. Linux işletim sistemlerinde daha iyi performans sağlar. Ayrıca ODBC sürücüleri de mevcut olduğu için birçok geliştirme platformunda kullanılabilir. Web sunucularında çoğunlukla kullanılmaktadır. Tablo başına 4TB'a kadar veri depolayabilir.

MySQL Veri Tipleri



Scanned with CamScanner

Veri Tipi	Açıklama
char(n)	Belirtilen n(0-255) sayısı kadar alfa nümerik ifadeler için kullanılır. n 'de belirtilen karakter sayısından daha az karakter yazıldığında bile n 'de belirtilen karakter sayısı kadar diskte yer kaplar.
date	1000-01-01'den 9999-12-31'e kadar değişebilen tarih bilgisi tutar.
mediumblob mediumtext	En fazla 16.777.215 karakter uzunluğunda metin veya binary veri tutar.
mediumint	-8388608 ile 8388607 aralığında işaretli ve işaretsiz tamsayıları tutar.
nchar(n)	UNICODE karakterler içerebilir. Özelliği diğer VTYS'ler ile aynıdır.
numeric(p,s), decimal(p,s)	Tamsayı ve virgüllü sayılar için kullanılan sayısal veri tipidir. Tam kısım(p) ve ondalık kısmın(s) ile belirlenir. Örneğin; numeric(3,2) olarak tanımlanan bir sütunun tam sayı kısmı 3 ondalık kısmı 2 basamaklı olacak demektir.
nvarchar(n)	En fazla 255 karakter ve değişken uzunlukta karakterler tutar. UNICODE içerebilir.
smallint	-32768 ile 32767 aralığında işaretli ve işaretsiz tamsayıları tutar.
text	En fazla 65.535 karakter alabilen metin verisi tutar.
timesstamp	01.01.1970 00:00:00 ile 12.31.2037 23:59:59 arasında Yıl + Ay + Gün + Saat + Dakika + Saniye biçimindeki zaman bilgisidir.
tinyblob tinytext	255 karakter veya daha az uzunlukta metin veya binary veri tutar.
tinyint	-128 ile 127 aralığında işaretli ve işaretsiz tamsayıları tutar.
varchar(n)	Belirtilen n(0-255) sayısı kadar alfa nümerik ifadeler için kullanılır. n 'de belirtilen karakter sayısı kadar değil, bilgide yazılan karakter sayısı kadar diskte yer kaplar.

1.8.4. Sybase

Sybase firmasına ait veritabanı sunucu yazılımıdır. Orta ve büyük ölçekli uygulamalar için tercih edilir. Ülkemizde daha çok bankacılık ve kamusal alanlarda tercih edilmektedir. Veritabanı üzerindeki işlemler T-SQL(Transact-SQL) sorgulama dili ile gerçekleştirilir.

1.8.5. PostgreSQL

PostgreSQL Global Development Group firmasına aittir. Birçok işletim sistemi üzerinde kullanılabilir. Veri güvenliği ön plandadır. Tablo başına 64TB'a kadar veri depolayabilir.

Scanned with CamScanner

1.8.6. Microsoft Access

Microsoft firmasına ait Microsoft Office ürünüdür. Küçük ölçekli uygulamalar için tercih edilir. Tablo başına 2GB'a kadar veri depolayabilir. Aynı anda 255 bağlantıya izin verilir. Windows işletim sistemleri dışında kullanılamaz.

1.8.7. IBM DB/2

Access ve MySQL'e göre daha performanslı, ancak küçük işletmelere göre yüksek maliyete sahiptir. Unix ve Windows işletim sistemlerinde çalışabilir.

1.9. VERİ MODELLERİ

Veritabanlarının temel görevi istenilen bilgileri saklamak olmasına rağmen yapısal olarak farklılıklar içermektedir. Veri modeli, verilerin depolanması, işlenmesi, veriler arası ilişkilerin kurulması biçimidir. Verileri mantıksal düzeyde düzenlemek için Her VTYS belirli bir veri modelini kullanır.

Bugüne kadar birçok veri modeli geliştirilmiştir. Bu veri modellerini dört grupta toplamak mümkündür.

- Sıradüzensel veri modeli(Hierarchical Data Model).
- Ağ veri modeli(Network Data Model).
- İlişkisel veri modeli(Relational Data Model).
- Nesneye yönelik veri modeli(Object-Oriented Data Model).

İlişkisel veri modeli yaygın olarak kullanılmaktadır. Günümüzde kullanılan VTYS'lerin hemen hemen tümü ilişkisel veri modeline dayalıdır.

1.9.1. Sıradüzensel(Hiyerarşik) Veri Modeli

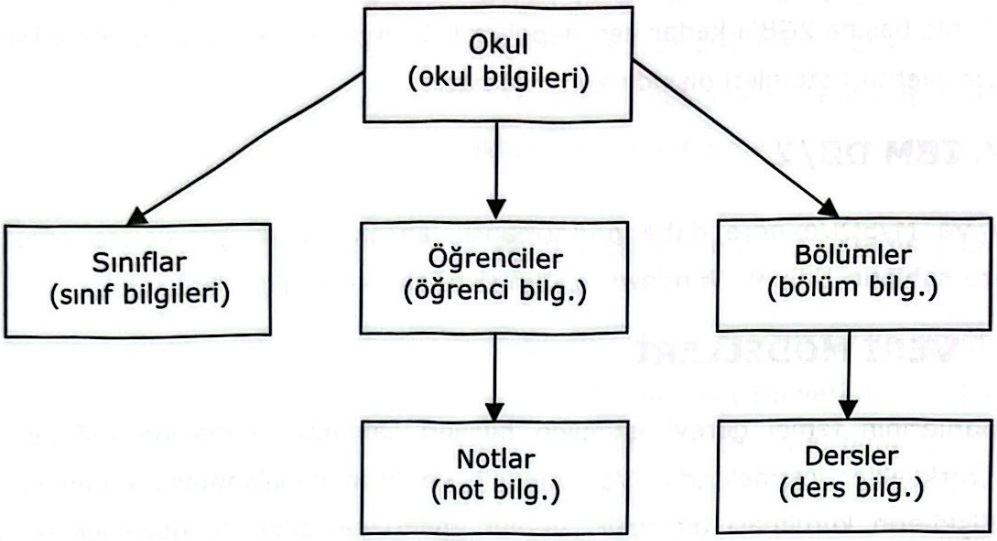
Sıradüzensel veri modeli ilk kullanılan veri modelidir. 1960 ve 1970 'li yıllarda kullanılmıştır. Bu model sunucu bilgisayarlarda çalışan yazılımlar tarafından kullanılmaktadır. Bu türde en çok kullanılan yazılım IBM tarafından üretilen IMS(Information Management System)'dir.

Bu modelde kayıtlar ağaç(tree) yapısında saklanır. En üstte kök ve kökün dalları ve bu dala bağlı diğer dallar şeklinde devam eder. Her dalın alt dalları sayesinde dallanma ve çeşitlilik artar. Alt dalın sadece tek bir noktadan üst dala bağlanması gerekir. Ebeveyn-çocuk ilişkisine(bir ebeveynin birden fazla çocuğu olabilir ama bir çocuğun bir ebeveyni olur) dayanır.

Varlıklar arasındaki ilişkileri kurarken her varlığın tek bir varlığa bağlanması gerekir. Aynı varlık birden fazla ilişkide kullanılacaksa tekrar oluşturmak gerektiği için gereksiz veri tekrarına neden olur. Herhangi bir bilgi bulunmak istediğinde arama en tepe

Scanned with CamScanner

noktadan yani kökten başlar dallara doğru devam eder. Dallardan herhangi biri silindiğinde o dala bağlı tüm kayıtlar silinir.

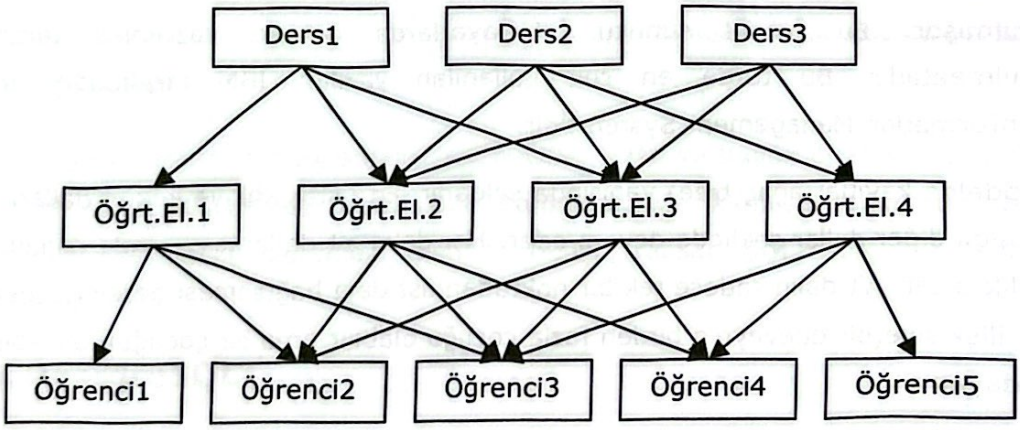


Şekil 1.20. Örnek Sıradüzensel Veri Modeli Diyagramı

1.9.2. Ağ Veri Modeli

1969 yılında çıkartılan bu model 1970 'li yıllarda ve 1980 'li yılların ilk yarısında kullanılmıştır. Sıradüzensel veri modelinin yetersiz kalmasından dolayı geliştirilmiştir. Sıradüzensel veri modeline benzer ancak bu modelde bir veri birden fazla veri ile ilişki kurabilmektedir. Verilerin birbirlerine ağ şeklinde bağlandığı varsayılır. Ortak bilgi alanları tanımlanarak birden fazla veri ile ilişki kurabilir. Ebeveyn-çocuk ilişkisinden biraz farklıdır. Her kaydın birçok ebeveyn ve birçok çocuk kaydı bulunabilir.

Aşağıdaki diyagramda da görüldüğü gibi her öğrenci birden fazla öğretim elemanına, her öğretim elemanı birden fazla derse bağlı olabilir.



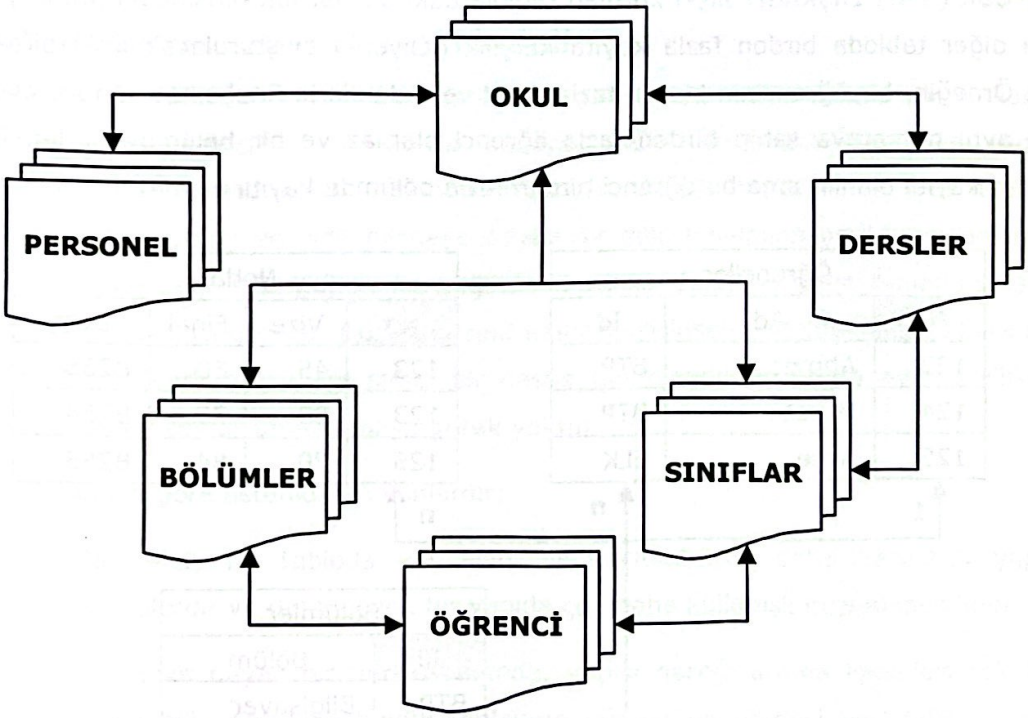
Şekil 1.21. Örnek Ağ Veri Modeli Diyagramı

Scanned with CamScanner

1.9.3. İlişkisel Veri Modeli

1970 'li yılların sonlarında kullanılmaya başlanmış ve 1985 'den sonra yaygınlaşmıştır. 1990 'lı yıllarda hemen hemen tüm sistemlerin ilişkisel tabanlı olduğu söylenebilir. Şu anda kullanılan veritabanlarının çoğu ilişkisel veri modeline daha çok destek verirler. Bu modelde ortak özelliğe sahip veriler tablolar aracılığı ile tutulurlar. Tablolar satır ve sütunlardan oluşur. Her sütun ayrı bir veri tutar. Ve buraya alan(field) denilir. Her alanın tutacağı verinin tipi belirlenmelidir. Aynı satırdaki veriler aynı kaydı ifade eder. Veriler ve ilişkiler tablolar üzerinde tanımlanır ve tüm bilgiler görülebilecek şekildedir. İlişkiler kurulurken birincil anahtar(Primary Key) ve yabancıl anahtar(Foreign Key) kullanılır. Anahtar alanlar sayesinde indeksleme yapıldığı için ilişkisel modelde işlemler daha hızlı gerçekleştirilir. Birden fazla tablodaki bilgiler ortak alanlar tanımlanarak birbirleriyle ilişkilendirilir. Böylelikle istenilen her türlü raporlama sağlanır.

Aşağıdaki şekil, tablolar ve bu tablolar arasındaki ilişkileri göstermektedir. Buradaki oklar tablolar arasındaki ilişkileri gösterir. Örnek olarak öğrenci tablosunun alanları ve türleri verilmiştir.



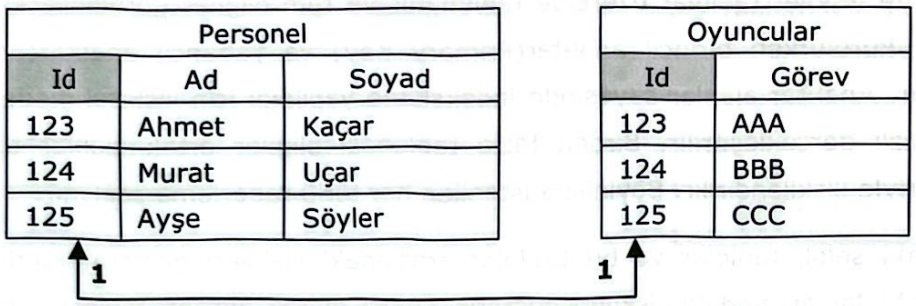
Ör. Öğrenci(no: int, isim: string, bolum: string, tcno: integer)

Şekil 1.22. Örnek İlişkisel Veri Modeli Diyagramı

Scanned with CamScanner

Tablolar arasındaki ilişkiler tablonun yapısına ve içerdiği bilgilere göre değişiklik gösterebilir. İlişkiler Bire-Bir(1-1), Bire-Çok(1-n), Çoğa-Bir(n-1) ve Çoğa-Çok(n-m) olabilir.

Bire-Bir (1-1) İlişkiler: İlişki kurulan tablolardaki kayıtların aynısından ikincisi olmayacaksa kurulan ilişkidir. İlişki kurulan tablolardaki alanlardan birisindeki primary key alanı diğer tabloda sadece bir kayıta karşılık geliyorsa oluşturulacak ilişki bire-bir olur.

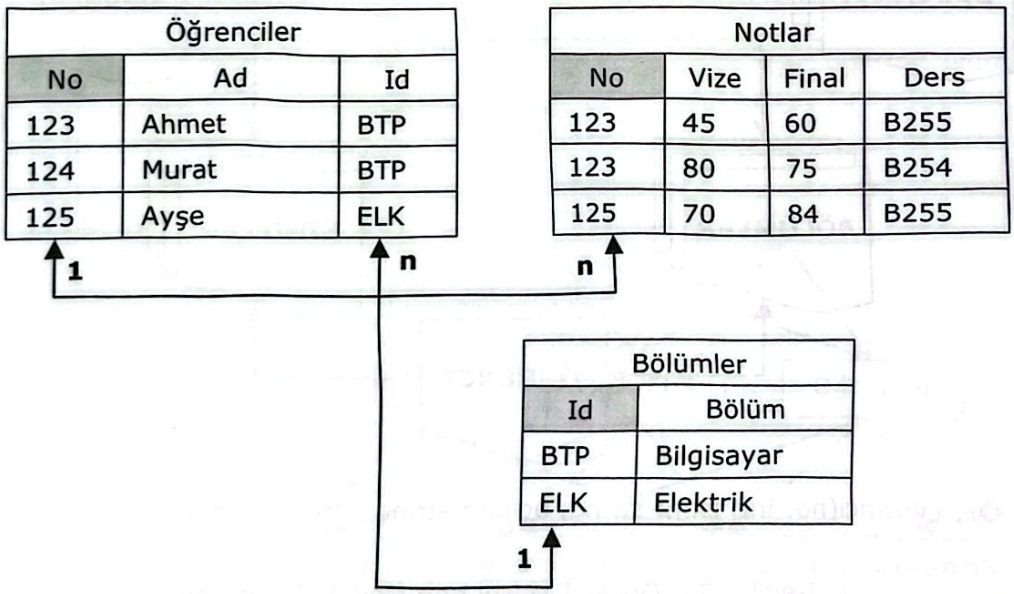


Personel		
Id	Ad	Soyad
123	Ahmet	Kaçar
124	Murat	Uçar
125	Ayşe	Söyler

Oyuncular	
Id	Görev
123	AAA
124	BBB
125	CCC

Şekil 1.23. Bire-Bir ilişki örneği

Bire-Çok (1-n) İlişkiler: İlişki kurulan tablolardaki alanlardan birisindeki primary key alanı diğer tabloda birden fazla kayıta karşılık geliyorsa oluşturulacak ilişki bire-çok olur. Örneğin, bir öğrencinin birden fazla dersi ve o derslerin final sınav sonucu olabilir ama aynı numaraya sahip birden fazla öğrenci olamaz ve bir bölümde birden fazla öğrenci kayıtlı olabilir ama bir öğrenci birden fazla bölümde kayıtlı olamaz.



Öğrenciler		
No	Ad	Id
123	Ahmet	BTP
124	Murat	BTP
125	Ayşe	ELK

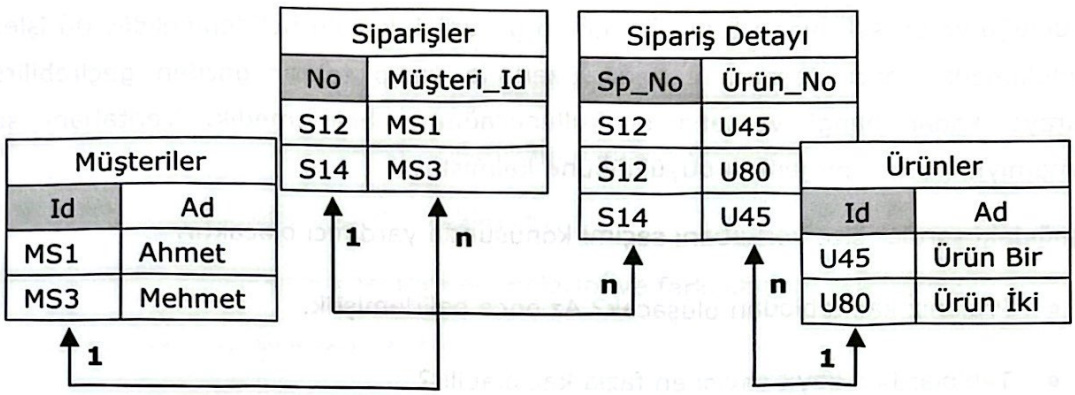
Notlar			
No	Vize	Final	Ders
123	45	60	B255
123	80	75	B254
125	70	84	B255

Bölümler	
Id	Bölüm
BTP	Bilgisayar
ELK	Elektrik

Şekil 1.24. Bire-Çok ilişki örneği.

Scanned with CamScanner

Çoğa-Çok (n-m) İlişkiler: İlişki kurulan tablolardan birindeki birçok kaydın değeri diğer tabloda birden fazla kayda karşılık geliyorsa oluşturulacak ilişki çoğa-çok olur. Örneğin; ürünler, siparişler ve müşteriler tablosu olan bir veritabanında bir müşteri birden fazla sipariş verebilir, bir sipariş birden fazla ürün içerebilir ve bir ürün birden fazla siparişte yer alabilir. Veya birden fazla ders bir öğrenciye ait olabilir veya birden fazla öğrenci aynı dersi alabilir.



Şekil 1.25. Çoğa-Çok ilişki örneği.

1.9.4. Nesneye Yönelik Veri Modeli

Son on yıldır gündemdedir. Günümüzde yaygın olarak kullanılmasa da kullanımı giderek yaygınlaşmaktadır. Son yıllarda nesneye yönelik veri modeli ilişkisel veri modeli ile birlikte VTYS 'lerde kullanılmaktadır. Nesneye yönelik model C++ gibi nesneye dayalı bir dille yazılan ve yine nesneye dayalı bir dille kullanılan veritabanı anlamına gelir. Bu modelde bir sorgunun karşılığında mutlaka daha önceden tanımlanmış bir nesne kümesi olması gerekir. Buradaki sınıf kavramı ilişkisel modeldeki ilişki kavramına karşılık gelir. Belirli özelliklere sahip bir nesne tanımlandıktan sonra aynı özelliklere sahip işlemlerde tekrar tanımlamaya gerek yoktur.

İlişkisel modele göre üstünlükleri şunlardır;

- Nesneler, bir tabloda yer alan bir kayıttan çok daha karmaşık yapıya sahiplerdir ve daha esnek bir yapıda çok daha kullanışlı düzenlenebilirler.
- Nesneye dayalı bir 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.

Bu bahsettiğimiz özellikler tamamen nesneye yönelik model kullanıldığında geçerlidir. İlişkisel veri modeli ile kullanıldığında nesneye yönelik model kendini ilişkisel modele uydurarak gerçekleştirir.

Scanned with CamScanner

1.10. NEREDEN BAŞLAMALIYIZ?

Hazırlayacağımız proje her ne olursa olsun başlamadan önce kendimize bir yol haritası çizmeliyiz. İlk olarak veritabanı tasarımını yapmalıyız. Başlangıçta veritabanınızın tasarımını iyi yapamazsanız ileride gereksiz bilgi yığınlarıyla dolu bir veritabanıyla karşılaşırsınız. Veritabanı tasarlarken öncelikle hangi tabloların kullanılacağı belirlenmelidir. Daha sonra tabloların içerisindeki sütunlar, sütunların türleri, sütun uzunluğu ve bu sütunların tutacağı verinin geçerlilik koşulu belirlenmelidir. Bu işlemleri uygulamadan önce tasarımını kağıt üzerinde yapıp tekrar gözden geçirebilirsiniz. Buraya kadar hangi veritabanını kullanacağımızı belirlemedik. Veritabanı seçimi tamamıyla size ve projenizin büyüklüğüne kalmıştır.

Aşağıdaki sorular size veritabanı seçimi konusunda yardımcı olacaktır.

- Projeniz kaç tablodan oluşacak? Az önce belirlemiştik.
- Tablolardaki kayıt sayısı en fazla kaç olabilir?
- Projenizi aynı anda kaç kişi kullanacak?
- Projenizin için ne kadar alan ayırdınız?
- Projeniz için ne kadar bütçe ayırdınız?
- Projenizde güvenlik önemli mi?
- Projeniz hangi tür platformlarda kullanılacak?

1.11. BÖLÜM ÖZETİ

Veritabanı sistemleri günümüzdeki projelerin tamamında kullanılmaktadır. Birbiriyle ilişkili verilerin tek bir ortamda tutulmasını sağlar. Veritabanları içerisinde barındıracağı verilere göre gruplandırılarak tablolar ve sütunlara bölünür. Veritabanı Yönetim Sistemleri kullanılan veritabanlarının oluşturulması, yönetimi, analizi gibi işlemlerin tamamının kolaylıkla yapılmasını sağlamaktadır. Veritabanları kişisel bilgisayarlarda tek kullanıcı olarak kullanılabileceği gibi istemci sunucu mimarisi yardımıyla birden fazla kullanıcı tarafından da kullanılabilir. Veritabanı kullanıcıları görevlerine göre veritabanı yöneticisi, uygulama programcısı, sorgu dili kullanıcısı ve son kullanıcı olarak gruplandırılır. Veritabanlarının tamamının temel görevi verileri depolamaktır ama bu işlemi yapma biçimine göre sıradüzensel veri modeli, ağ veri modeli, ilişkisel veri modeli ve nesneye yönelik veri modeli olarak gruplandırılır. Günümüzde kullanılan veritabanlarının çok büyük bir bölümünü ilişkisel veri modelini kullanmaktadır.

Scanned with CamScanner

1.12. ÇALIŞMA SORULARI

1. Geleneksel dosyalama sistemlerinin çalışma şeklini açıklayarak bildiğiniz bir programlama dili yardımıyla txt dosyadan veri okumak ve veri yazmak için bir uygulama geliştiriniz.

2. Geleneksel dosyalama sistemleriyle veritabanı yönetim sistemlerinin benzerlik ve farklılıklarını açıklayınız.

3. VTYS'nin geleneksel sisteme göre üstün özelliklerini açıklayınız.

4. Veritabanının görevini açıklayarak kullanıldığı alanlara örnekler veriniz.

5. Tablo, satır ve sütun kavramlarını açıklayınız.

6. Birincil anahtar ve yabancıl anahtar kavramlarını ve farklılıklarını açıklayınız.

7. Veritabanı kullanıcı türleri nelerdir?

8. Örnek bir veritabanı için kullanıcılar belirleyerek yetkilendirmelerini şematik olarak gösteriniz.

9. Veritabanı ile VTYS'nin farkını açıklayınız.

10. İlişkisel veri modelini açıklayınız.

11. Bir öğrenci bilgi sistemi için kullanılacak veritabanı, tablolar ve tablolar arasındaki ilişkileri belirleyiniz.

Scanned with CamScanner