AKDENİZ ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ

Sezgin IRMAK

İLİŞKİSEL VERİTABANI YÖNETİM SİSTEMLERİ VE AKDENİZ ÜNİVERSİTESİ SOSYAL BİLİMLER ENSTİTÜSÜ ÖĞRENCİ İŞLERİ OTOMASYONU UYGULAMASI

İşletme Anabilim Dalı
Yüksek Lisans Tezi

Antalya, 2004

AKDENİZ ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ

Sezgin IRMAK

İLİŞKİSEL VERİTABANI YÖNETİM SİSTEMLERİ VE AKDENİZ ÜNİVERSİTESİ SOSYAL BİLİMLER ENSTİTÜSÜ ÖĞRENCİ İŞLERİ OTOMASYONU UYGULAMASI

Danışman
Yrd. Doç Dr Can Deniz KÖKSAL

İşletme Anabilim Dalı
Yüksek Lisans Tezi

Antalya, 2004

Sosyal Bilimler Enstitüsü Müdürlüğü'ne,

Bu çalışma, jürimiz tarafından İşletme Anabilim Dalı Yüksek Lisans Programı'nda Yüksek Lisans tezi olarak kabul edilmiştir

Başkan : Prof. Dr. Orhan Kuruüzüm

Üye (Danışmanı) : Yrd. Doç. Dr. Cem Demiş

Üye : Yrd. Doç. Dr. Adil Korkmaz

Üye :

Üye :

Onay : Yukarıdaki imzaların, adı geçen öğretim üyelerine ait olduğunu onaylarım

12. /01/ 2004

Prof. Dr. Orhan KURUUZUM
Müdür

İÇİNDEKİLER

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

ÖZET

Günümüzde birçok organizasyon için iş süreçlerinde etkinliğin arttırılması önemli bir konudur. Bunu gerçekleştirmek için bilişim sistemlerinin olanaklarından yararlanılmakta ve birçok organizasyon kendilerine uygun yöntemlerle bünyelerinde bilişim sistemleri oluşturmaktadırlar. Bu çalışmada veritabanı yönetim sistemleri ve örnek bir sistemin Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü öğrenci işleri süreçlerine uygulanması ele alınmaktadır.

Çalışmanın birinci bölümünde bilişim sistemleri incelenmiştir. Sistem kavramı, bilişim sistemleri ve bileşenleri ele alınmış, bilişim sistemi kavramlarına, amaçlarına ve bilişim sistemlerinin önemine değinilmiştir. Daha sonra günümüzde işletmelerde kullanılan bilişim sistemleri hakkında bilgi verilmiş olup, sistem geliştirme yöntemleri ve aşamaları incelenmiştir.

İkinci bölümde veritabanlarının tasarımı, kurulumu ve yönetimi konuları ele alınmıştır. Veritabanı kavramları açıklanmış olup, operasyonel, dağıtık ve harici veritabanı türleri hakkında ve ayrıca hiyerarşik, ağ, ilişkisel, varlık-ilişki ve nesne-temelli veritabanı modelleri ile ilgili bilgi verilmiştir. Bundan sonra ilişkisel veritabanları ve veritabanı yönetim sistemleri incelenmiştir. Ayrıca ilişkisel veritabanlarının kabul görmüş sorgulama dili olan yapısal sorgulama dilinin (SQL) temel kavramları ve ilişkisel veritabanlarında kullanımı ile ilgili bilgiler verilmiştir.

Uygulama bölümü olan üçüncü bölümde Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü öğrenci işleri süreçleri için bir veritabanı yönetim sistemi tasarımı yapılmış ve bu doğrultuda uygulama yazılımı geliştirilerek sistemin otomasyona geçirilmesi amaçlanmıştır. Üçüncü bölümde sistem veritabanının ve otomasyon programının yapısının yanısıra, sistemin kayıt, belge, ders, vb., ana başlıkları altında toplanan işlemleri nasıl gerçekleştirdiği hakkında bilgiler de verilmeye çalışılmış olup, bunlara ek olarak otomasyon programında kullanılan form yapıları, veritabanı ilişkileri ve işlemleri, bu sırada çalışan sorgu tipleri ve örnek yazılım kodları hakkında da açıklayıcı bilgiler verilmiştir.

ABSTRACT

Increasing the effectiveness of business processes is an important matter for many organizations today. To achieve this, organizations use information systems' opportunities and develop information systems inside or outsource their information system requirements. Database management systems and the implementation of a sample system to Akdeniz University, Institute of Social Sciences' student affairs system were presented in this study.

Information systems were explained in the first section of the study. System concept and components, goal, and importance of information systems were mentioned. Also information systems used in business today, and systems development methods and stages were examined.

In the second section of the study, database design, implementation, and management subjects were studied. Database concepts, database types, such as operational, distributed, and external, and database models, such as hierarchical, network, relational, entity-relationship, and object-oriented were examined. In this section relational databases, relational database management systems and structured query language (SQL) were mainly explained.

In the last section, design and implementation of the sample database management system for the Institute of Social Sciences and the application program were explained. System database and application program's structure were examined. And also, how system operate processes, such as registration, reporting, course enrolling, etc. In addition, forms, database relations and transactions, queries, and sample software syntaxes were given in this section.

ÖNSÖZ

Önsöze başlarken öncelikle bana akademik ortamda çalışma fırsatını oluşturan ve desteğini her zaman hissettiren değerli hocam Prof. Dr. Ayşe KURUÜZUM'e teşekkür etmek isterim.

"İlişkisel Veritabanı Yönetim Sistemleri ve Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü Öğrenci İşleri Otomasyonu Uygulaması" başlıklı tez çalışmam süresince tezin şekillenmesinde önemli katkıları bulunan tez danışmanı hocam sayın Yrd. Doç. Dr. Can Deniz KÖKSAL'a, ayrıca bu konuyu öneren ve Sosyal Bilimler Enstitüsü öğrenci işlerinin süreçleri konusunda bilgilendirerek uygulamanın tasarım ve yazılım aşamalarını yönlendiren sayın hocam Prof. Dr. Orhan KURUÜZUM'e teşekkür ederim

Sezgin IRMAK

Ocak 2004, Antalya

GİRİŞ

Sanayi toplumundan bilgi toplumuna geçiş söylemlerinin oldukça sıklaştığı günümüzde bilginin, dolayısıyla bilginin oluşturulması, depolanması, işlenmesi ve erişilmesi kavramlarının da önem kazandığı bir gerçektir. Bilginin elektronik ortamlara girmesiyle birlikte bilgi üretimi ve paylaşımı oldukça hız kazanmış ve bu ortamlarda bulunan bilgi miktarı da oldukça büyük boyutlara ulaşmıştır.

Günümüzde bilgi teknolojileri günlük hayatı, iş dünyasını ve işin yapılma yöntemlerini etkiler ve değiştirir durumdadır. Vergi dairelerinde, haberleşmede, bankacılıkta, eğitim kurumlarında ve diğer birçok alanda bilişim sistemlerinin olanaklarından yararlanılmaktadır. Bilişim sistemleri organizasyonlarda insanların, donanımın, yazılımın, iletişim ağlarının ve veri kaynaklarının organize birlikteliğinden oluşmaktadır.

Bilişim sistemlerinde verinin organizasyonu için veritabanları kullanılmaktadır. Veritabanları operasyonel, dağıtık veya harici veritabanları gibi farklı türlerde ve kullanım biçiminde olabilirler. Veritabanlarında bulunan verinin erişilmesi, işlenmesi, bilgiye dönüştürülmesi ve dağıtılması gibi faaliyetler veritabanı yönetim sistemleri tarafından gerçekleştirilmektedir.

Birçok organizasyon, işlerin ve süreçlerin geleneksel yöntemlerle yürütülmesi veya yetersiz bilişim sistemleri uygulamaları nedeniyle işgücü yetersizliği, bilgi kaybı, yüksek hata oranı ve etkinliği sağlayamama gibi sorunlarla karşı karşıya gelmektedirler. Özellikle yüksek miktarlarda veri işlemlerinin ve bilgi iletişiminin gerçekleştiği organizasyonlarda verinin veritabanlarında depolanması ve veritabanı yönetim sistemleri yoluyla süreçlerin otomasyona geçirilmesi bir çözüm yolu olarak görülmektedir.

Bu çalışmada bilişim sistemleri, veritabanları ve veritabanı yönetim sistemleri ele alınmış ve Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü'nün öğrenci işleri süreçleri incelenerek bir veritabanı yönetim sistemi uygulaması geliştirilmiştir. Bu uygulama ile enstitü öğrenci işlerinin otomasyona geçirilmesi, oluşturulan veritabanı yönetim sistemiyle enstitü verilerinin organizasyonu ve bu yolla da süreçlerin etkinleştirilmesi amaçlanmıştır.

BİRİNCİ BÖLÜM

BİLİŞİM SİSTEMLERİ

1.1. Sistem Kavramı ve Bileşenleri

Sistem kavramı günümüzde birçok alanda farklı anlamlarda kullanılmaktadır. Her araştırmacı temel unsurları aynı kalmak koşuluyla kendi alanına ve kendi problemine özgü tanımlar geliştirebilir. Dolayısıyla sistem kavramı için tek bir tanımlamadan söz etmek zor olacaktır. Ancak bir sistemde yer alması gereken birden çok bileşen, bileşenler arası ilişkiler, bileşenlerin oluşturduğu bir bütün ve sistemin amacı gibi unsurları (Esen, s 10) vurgulamak açısından bir tanım kullanılabilir. Sistem bir veya daha çok amaca ve sonuca ulaşmak üzere aralarında ilişkiler olan fiziksel ve kavramsal birden çok bileşenin oluşturduğu bütündür (Esen, s.10). Bu çalışmada sistem kavramı bilişim sistemleri alanının bakışı açısından ele alınacaktır

Sistem, ortak amaçlar doğrultusunda girdileri kabul etmek ve düzenlenmiş işlem süreçleri içerisinde çıktılar üretmek üzere birlikte çalışan, birbirleriyle ilişkili bileşenler grubudur. Aynı zamanda dinamik sistem olarak da adlandırılan böyle bir sistem girdi, işlem ve çıktı olmak üzere etkileşimli üç temel bileşene veya fonksiyona sahiptir (O'Brien, 1997, s.18).

1.1.1. Girdi

Bütün sistemler bir çeşit veri girişine ihtiyaç duyarlar (Shelly vd , 2001, s 1.4). Örneğin bir arabanın gaz pedalına basıldığı zaman motora bir çeşit veri gönderilmiş olur veya evde televizyonun kumandasından ses ayar tuşlarına basıldığı zaman televizyon cihazı bir çeşit veri alır. Bütün sistemlerde sistemin işlem aşamasından önce veri girişi gereklidir, dolayısıyla girdi kavramını veri kavramıyla bütünleştirebiliriz.

1.1.2. İşlem

İşlem aşaması girdiyi çıktıya dönüştürmek için gerekli olan dönüştürme süreçlerini içermektedir (O'Brien, 1997, s.18). Bilişim sistemleri açısından baktığımızda girdi için veri çıktı için ise bilgi diyebiliriz. Bilgiye verinin işlenmiş ve kullanışlı bir yapıya dönüştürülerek çıktı olarak alınabilir hali diyebiliriz. İşlem aşamasına örnek olarak herhangi bir üretim

aşamasını, otomobil montaj hattını veya bir hesap sürecince matematik hesaplamaların yapılmasını verebiliriz.

1.1.3. Çıktı

Çıktı aşaması dönüştürme süreci tarafından üretilmiş unsurları son hedeflerine doğru transfer etme süreçlerini içermektedir (O'Brien, 1997, s.18). Bilişim sistemleri açısından bakıldığında çıktı için bilgi veya kullanışlı bilgi diyebiliriz. Herhangi bir sistemin amacının hedeflenen çıktıyı elde etmek olduğunu söyleyebiliriz. Sözü edilen çıktı buzdolabı gibi bir ürün, verilen bir hizmet, bazen de yönetimsel bilgi olabilir.

1.2. Bilişim Sistemlerinin Tanımı ve Amaçları

Bilişim sistemi, bir organizasyonda insanların, donanımın, yazılımın, iletişim ağlarının ve veri kaynaklarının organize birlikteliğinden oluşur ve organizasyonda bilgiyi toplar, işler ve dağıtır (O'Brien, 1997, s.4). Tanıma dayanarak bilişim sistemlerinin iki boyutundan söz etmek mümkün olabilir; bunlardan birincisi teknoloji boyutu, diğeri ise insan boyutudur. Bilişim sistemleri sosyoteknik sistemler olup, makinalar, cihazlar ve fiziksel teknoloji içerseler de aynı zamanda daha iyi çalışmaları için sosyal, entelektüel ve organizasyonel araştırmalara da ihtiyaç duyarlar (Karahoca ve Karahoca, 1998, s.16). Bilişim sistemlerinin gelişiminde teknoloji boyutunun gelişmesinin yanı sıra özellikle insan boyutunun niteliklerinin gelişmesi de göz ardı edilmemelidir.

İnsanlar uygarlığın doğuşundan bu yana değişik türlerde fiziksel araçları (donanım), bilgi işleme talimatlarını (yazılım), iletişim kanallarını (bilgisayar ağları) ve depolanmış verileri (veri kaynakları) kullanarak birbirleriyle iletişim kurabilmek için bilişim sistemlerine güvenmişlerdir (O'Brien, 1997, s.4). Bilişim sistemlerinin amacının bir organizasyonda, çalışanların veri toplama, işleme, yararlı bilgi elde etme ve bunları dağıtma ihtiyacını karşılamak olduğunu düşünebiliriz. Bu durumda teknolojik boyut organizasyonun bilişim ihtiyaçlarına göre değişen nitelikte olacaktır. Bazı durumlarda teknoloji kalem, kağıt ve yazılı evraktan ibaret olabilecekken bazı durumlarda ise karmaşık bilgisayar ağları, donanımları, yazılımları ve çok büyük boyutlarda veritabanları olabilecektir. Bu bölümde bilişim sistemleri bilgisayar tabanlı bilişim sistemleri düzeyinde ele alınacaktır.

1.3. Bilişim Sistemlerinin Önemi

Günümüzde organizasyonlar, verinin toplanmasını ucuzlatan ve kolaylaştıran teknoloji uygulamaları yoluyla çok büyük miktarlarda veri toplamaktadırlar. Dünyadaki veriler toplamı tahmini olarak her 20 ayda iki katına çıkmaktadır ve birçok büyük işletme günlük olarak terabaytlar (10^{12} bayt) düzeyinde veriyi yönetmektedirler (Watson, 2002, s 1). Dolayısıyla bilişim sistemleri, işletmeler ve diğer organizasyonlar için günümüzde hayati derecede önemli hale gelmiştir. Bu yüzden genel işletmecilik ve yönetim dallarında çalışılması gerekli bir alan özelliği kazanmıştır.

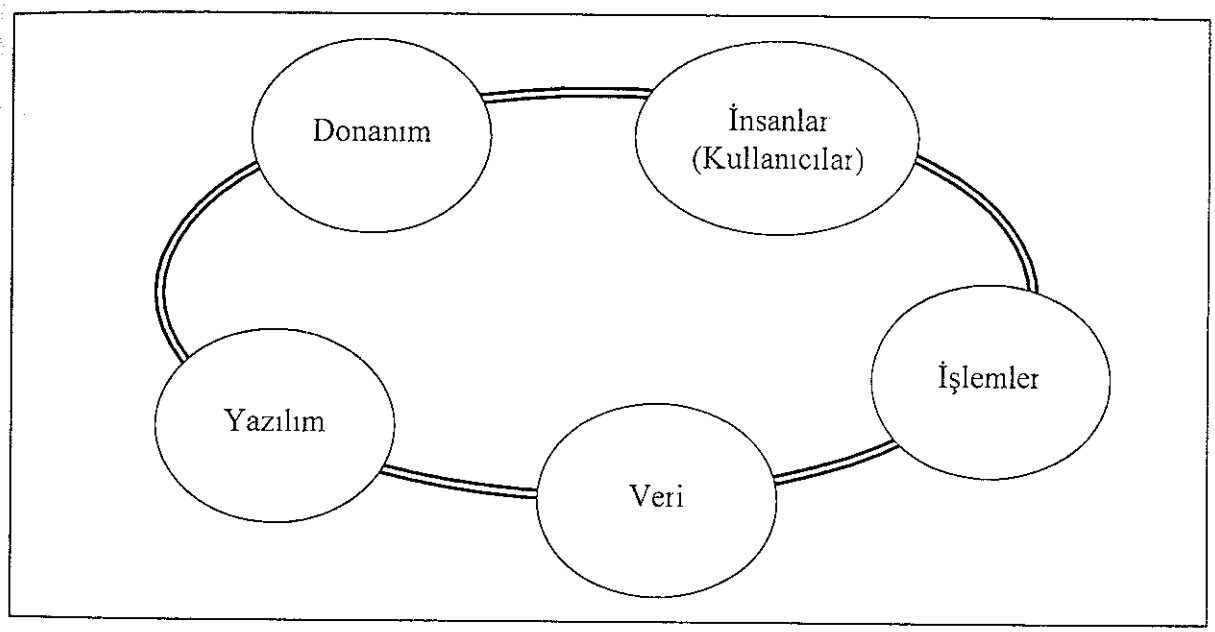
Özellikle son yirmi yılda bilginin tek başına önemli bir değer kazanmasıyla birlikte bilgiye ulaşma, veri toplama, işleme, elde edilen bilgiyi dağıtma ve bu işlemleri etkin bir şekilde yapabilme organizasyonlar için önem kazanmıştır. Bu fonksiyonu yerine getirebilmek için ayrıca bölümler oluşturulmuştur. Önceleri organizasyonların bilişim ihtiyaçları genellikle bilgi işlem bölümleri tarafından karşılanmaktaydı, ancak zamanla bilişim ihtiyaçlarının artması ve daha karmaşık olmaya başlamasıyla birlikte, bu ihtiyaçların daha geniş alanda uzmanlaşmış ve çok yönlü çözümler üretebilen fonksiyonlarca karşılanması gerekli hale geldi. Bu nedenle günümüzde işletmeler bünyelerinde, sözü edilen seviyede nitelikleri taşıyan genellikle IT (Information Technologies) olarak adlandırılan Bilgi Teknolojileri bölümleri oluşturmaktadırlar. Buna alternatif olarak da bilişim sektöründe faaliyet gösteren ve bilişim ihtiyaçlarını karşılamak amacıyla işletmelere çözümler üreten firmalarla ihtiyaçlarına göre değişen nitelik ve içerikte anlaşmalar yapmaktadırlar.

Stratejik bilişim sistemleri rakiplerinin üzerinde bir yer kazanmaları için organizasyonların amaçlarını, operasyonlarını, ürünlerini, hizmetlerini veya çevresel ilişkilerini değiştirmektedirler (Laudon ve Laudon, 1997, s.43).

Bir bilişim sistemi şirketlere satışlarını arttırmak ve pazarlama tekniklerini geliştirmek için veri üreterek rekabetçi avantaj sağlayabilir (Laudon ve Laudon, 1997, s 44). Müşterilerin hareketlerinin ve satın alma işlemlerinin incelenmesi sonucunda firmalar karlı müşterilerini ayırt edebilir ve bunlara yönelik daha fazla iş geliştirebilir. Aynı şekilde firmalar bu veriyi karlı olmayan müşterilerin tespit edilmesinde de kullanabilir (Clemons ve Weber, 1994, s 9).

1.4. Bilişim Sistemlerinin Bileşenleri

Bir bilişim sistemi veri kaynaklarını girdi olarak alan, bunları işleyerek bilgiye dönüştüren ve çıktı olarak bu bilgileri veren bir sistemdir (O'Brien, 1997, s.21). Bu süreci gerçekleştiren bir bilişim sistemi bazı bileşenler içermektedir. Bu bileşenler farklı kaynaklarda değişen ifadelerle yer alsalar da temelde aynıdırlar. Bir bilişim sistemi beş anahtar bileşene sahiptir, donanım, yazılım, veri, işlemler ve insanlar (Shelly, vd., 2001, s.1.5). Burada Şekil-1.1'de gösterilen beş temel bileşenden söz edeceğiz. Bunlar donanım, yazılım, veri, işlemler ve kullanıcılar olarak ifade edilebilir.



Şekil 1.1 Bir Bilişim Sisteminin Bileşenleri (Shelly, vd., 2001, s.1.5)

1.4.1. Donanım

Donanımı bir bilişim sisteminde yer alan tüm fiziksel araç, gereç ve malzemeler olarak nitelendirebiliriz. Donanım bir bilişim sisteminin fiziksel katmanıdır ve bilgisayarları, bilgisayar ağlarını, iletişim ekipmanlarını, tarayıcıları, dijital grafik yakalama kartlarını ve diğer teknoloji tabanlı altyapıyı içerir (Shelly, vd., 2001, s.1.5). Bilişim sistemlerinde yer alan donanım örnekleri;

- Bilgisayar sistemleri merkezi işlem birimlerini (CPU) ve buna bağlı çeşitli çevre birimlerini içerir. Ana bilgisayar sistemleri ve mikrobilgisayar sistemleri bunlara örnek olarak verilebilir.

- Bilgisayar çevre birimleri veri ve komut girişleri için klavye, elektronik fare, bilgi çıkışları için ekran veya yazıcı ve verileri saklamak için manyetik veya optik diskleri içerir. (O'Brien, 1997, s.23)

1.4.2. Yazılım

Yazılım için genel olarak bilgi işleme komutları kümesi diyebiliriz. Yazılım sistem yazılımları ve uygulama yazılımlarından oluşur (Shelly, vd., 2001, s.1.5).

- Sistem yazılımları; örneğin işletim sistemleri bir bilgisayar sisteminin işlemlerini kontrol eder ve destekler (O'Brien, 1997, s.23). Sistem yazılımları bilgisayarı kontrol eder ve işletim sistemini, donanımla iletişimi gerçekleştiren aygıt sürücülerini, verinin farklı bir formata dönüştürülmesini sağlayan görevleri üstlenen programları, virüs korumasını ve yedekleme işlemlerini içerir, ağ ortamında ise ağ işletim sistemi, veri akışını kontrol eder, güvenlik sağlar ve ağ hesaplarının yönetimini gerçekleştirir (Shelly, vd., 2001, s.1.5).
- Uygulama yazılımları; işletmelerin işletme fonksiyonlarını yerine getirmelerine olanak sağlayan ve kullanıcılara destek veren programlardan oluşur (Shelly, vd., 2001, s.1.5). Uygulama yazılımlarına örnek olarak, kelime işlemcilerini, hesap tablolarını, veritabanı yönetim sistemlerini, satış analizi, stok yönetimi gibi programları gösterebiliriz.

1.4.3. Veri

Veri işlenmemiş, özetlenmemiş ve analiz edilmemiş gerçeklerdir (Watson, 2002, s.25). Bu tanımı daha da geliştirecek olursak veri birçok farklı yapıda olabilir. Bunlar arasında geleneksel alfanümerik veri, metin verisi, görüntü verisi ve ses verisi gösterilebilir.

- Alfanümerik veri sayıların, alfabetik ve diğer karakterlerin bir araya gelmesinden oluşan işlemleri, diğer olayları ve fonksiyonları tanımlayan veridir.
- Metin verisi yazılı iletişimde kullanılan cümleler ve paragraflardan oluşan veridir.
- Görüntü verisi şemalar, şekiller ve grafik içeren veridir.
- Ses verisi insan sesi ve diğer seslerden oluşan veridir (O'Brien, 1997, s.24)

1.4.4. İşlemler

İşlemler veya süreçler kullanıcılar, yöneticiler veya BT çalışanları tarafından gerçekleştirilen görevleri tanımlamaktadır İşlemler mutlaka yazılı dokümanlarda, el kitaplarında veya online referans materyallerinde tanımlanmış olan özel işletme modellerini desteklemelidir (Shelly, vd., 2001, s.1.6).

1.4.5. Kullanıcılar

Bir bilişim sisteminin temel amacı işletme içinden veya işletme dışından kullanıcılara ve yöneticilere yararlı bilgi sağlamaktır (Shelly, vd., 2001, s.1.6). Kullanıcılar veya diğer bir deyişle son kullanıcılar, çalışanlar, müşteriler, satıcılar ve bilişim sistemiyle etkileşim halinde olan diğer kullanıcılar olarak tanımlanabilir (O'Brien, 1997, s.22) Kullanıcıları ise son kullanıcılar ve bilişim sistemleri uzmanları olarak iki gruba ayırmaktadır;

- Son kullanıcılar bilişim sistemini veya bunun ürettiği bilgiyi kullanan insanlardır. Bunlar muhasebeciler, satış uzmanları, mühendisler, memurlar, müşteriler veya yöneticiler olabilir. Günümüzde birçok insan bilişim sistemlerinin son kullanıcısıdır
- Bilişim sistemleri uzmanları bilişim sistemlerini geliştiren ve işleten insanlardır. Bunlar sistem analistleri, programcılar, bilgisayar operatörleri, yönetimsel, teknik ve memur seviyesinde bilişim sistemi personeli olabilir

1.5. İşletmelerde Bilişim Sistemleri

Bilişim teknolojileri iş dünyasının temellerini yeniden şekillendirmektedir Müşteri hizmetleri, işlemler, ürün ve pazarlama stratejileri ve dağıtım yoğun bir şekilde hatta bazen tam anlamıyla Bilişim Teknolojilerine (BT) bağımlıdır. Bilişim teknolojileri ve bunlara yapılan harcamalar iş-dünyasının olağan bir parçası haline gelmiştir. Bilişim teknolojileri tamamıyla işletmelere girmiş olup, Şekil-1.2'de işletmelerde bilişim sistemleri kullanımının temel nedenleri gösterilmiştir.

İşletmelerde bilişim teknolojileri kullanımının üç temel nedeninden söz edebiliriz. Bunlar piramidin aşağısından yukarıya işletme faaliyetlerinin desteklenmesi, yönetimsel karar

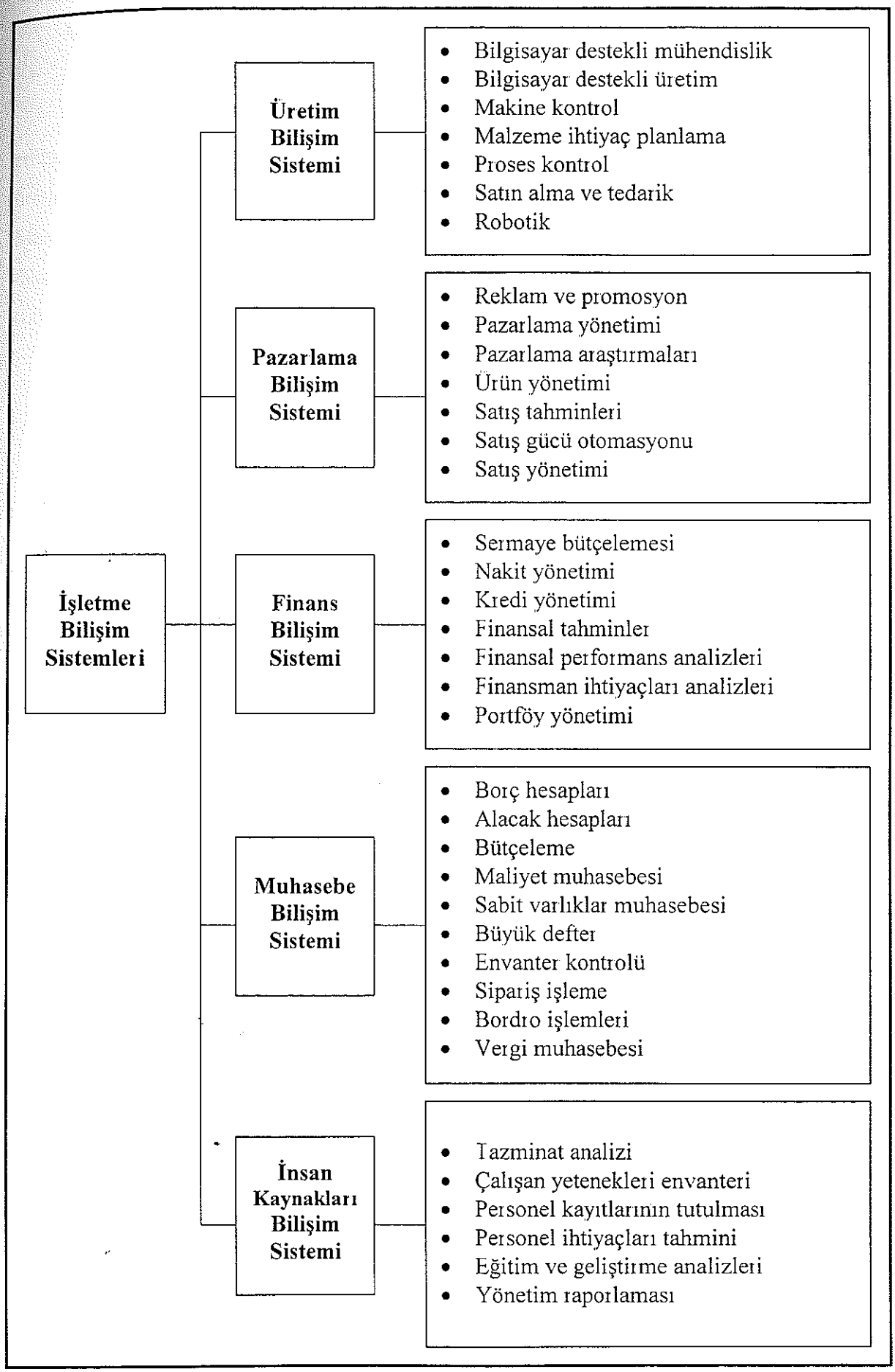
verme sürecinin desteklenmesi ve stratejik rekabet avantajı sürecinin desteklenmesi olarak sıralanabilir.

Şekil 1.2. İşletmelerde Bilişim Sistemleri Kullanımının Uç Temel Nedeni (O'Brien, 1997, s.11)

Herhangi bir işletmede müşteriler, işletme faaliyetlerini destekleyen bilişim sistemleriyle etkileşim halindedirler. Örnek olarak, birçok perakende şirketi müşterilerin alışveriş kayıtlarını tutmak, envanter hareketlerini izlemek, çalışanlarına ödeme yapmak, yeni mal almak ve satış trendlerini değerlendirmek için bilgisayar tabanlı bilişim sistemleri kullanmaktadırlar.

Bilişim sistemleri ayrıca yöneticilerin stratejik rekabet avantajı elde etme doğrultusunda daha iyi kararlar almalarına yardımcı olmaktadırlar. Bu yüzden stratejik bilişim sistemleri işletmelere rakipleri üzerinde stratejik rekabet avantajı sağlamak için stratejik ürün ve hizmetleri almada yardımcı olmaktadırlar (O'Brien, 1997, s.10).

İşletmelerde yapılması gereken faaliyetlere, gerçekleşmesi gereken işlemlere, çözülmesi gereken problemlere göre bilişim sistemlerinin kullanımının birçok farklı şekli olabilir. İşletmelerdeki fonksiyonel alanlarda kullanılan bilişim sistemi uygulamaları destekledikleri işletme fonksiyonuna göre adlandırılabilirler. Bunlar muhasebe bilişim sistemleri, pazarlama bilişim sistemleri, üretim bilişim sistemleri v.b. olabilir. Şekil-1.3'de bilişim sistemlerinin işletme fonksiyonları kategorilerine göre ne şekilde gruplandırılabileceği özetlenmektedir.



Şekil 1.3 İşletme Fonksiyonları Bilişim Sistemleri Örneği (O'Brien, 1997, s.240)

1.5.1. Üretim Bilişim Sistemleri

Üretim bilişim sistemleri, ürün veya hizmet üreten süreçlerin planlanması ve kontrol edilmesiyle ilgili bütün aktiviteleri içine alan üretim faaliyetlerini destekler. Bilgisayar tabanlı üretim bilişim sistemleri bilgisayarlı bütünleşik imalat sistemini (CIM – Computer-Integrated Manufacturing) desteklemek için birçok teknik kullanır. Bilgisayarlı bütünleşik imalat sistemi fabrika otomasyonunda bilgisayar kullanımının amaçlarını vurgulayan genel bir kavramdır (O'Brien, 1997, s.245).

Bu amaçlar;

- Otomasyon ve entegrasyon için önemli bir temel olan üretim süreçlerinin, ürün tasarımlarının ve fabrika organizasyonun değişim mühendisliği ile basitleştirilmesi.
- Üretim süreçlerinin ve bunları bilgisayarlar ve robotlarla destekleyen işletme fonksiyonlarının otomasyonu.
- Bütün üretim ve destek süreçlerinin bilgisayarlar ve iletişim ağları kullanılarak entegrasyonu.

olarak ifade edilebilir.

Bilgisayarlı bütünleşik imalat sisteminin bazı yararları şu şekilde sıralanabilir:

- İş basitleştirmesi ve otomasyonu, daha iyi üretim programı planlaması ve üretim iş yükü ile üretim kapasitesinin daha iyi dengelenmesi yoluyla artan etkinlik.
- Sürekli denetim, geribildirim ve fabrika faaliyetlerinin, ekipmanlarının ve robotlarının kontrolü sonucunda üretim imkanlarının daha etkin kullanımı, yüksek verimlilik ve daha iyi kalite kontrolü.
- Üretimin ve bitirilmiş ürün gereksinimlerinin daha iyi planlanması ve kontrolü, tam zamanında stok politikaları ve iş basitleştirmesi yoluyla üretim imkanlarına ve stoklarına yapılan yatırımda azalma.
- Stokta bulunmama durumlarındaki önemli düşüş ve müşteri ihtiyaçlarını daha iyi karşılayan yüksek kalitede ürünler üretilmesi yoluyla müşteri hizmetlerinde gelişme (O'Brien, 1997, s.246)

Üretim bilişim sistemleri adı altında birçok bilgisayar tabanlı sistem kullanılmaktadır. Bunlar imalat sisteminde yukarıda sıralanan faydaları sağlamaktadırlar. Tablo-1.1'de farklı amaçlara yönelik bilgisayar tabanlı bilişim sistemlerinin farklı örnekleri görülmektedir.

Tablo 1.1. Üretimde Kullanılan Bilgisayar Tabanlı Bilişim Sistemi Örnekleri (O'Brien, 1997, s 247)

Bilgisayar Destekli Tasarım (CAD)	Ürün ve imalat süreçleri modellerinin oluşturulması, simülasyonu ve değerlendirilmesi amacıyla kullanılır.
Bilgisayar Destekli Üretim (CAM)	Ürünlerin fabrikasyonu, montajı ve paketlenmesi için bilgisayarların ve robotların kullanılmasıdır.
Fabrika Yönetimi	Üretim faaliyetlerinin planlanması ve kontrolü, gelen siparişlerin ve ham madde ihtiyaçlarının, maliyet izleme ve kalite güvence programlarının koordinasyonu amacıyla kullanılır.
Kalite Yönetimi	Ürün ve proses özelliklerinin değerlendirilmesi, gelen malzemelerin ve giden ürünlerin test edilmesi, imalat süreçlerinin faaliyetteyken test edilmesi ve kalite güvence programlarının dizaynı
Lojistik	Malzemelerin satın alınması ve tedariği, malzemelerin kontrolü ve dağıtımı, envanterin kontrolü ve ürünlerin gönderilmesi.
Bakım	Makinelerin ve süreçlerin takibi ve ayarlanması, teşhislerin uygulanması, düzeltici ve koruyucu bakımın yapılması.

1.5.2. Pazarlama Bilişim Sistemleri

İşletmeler giderek artan bir oranda kendilerine günümüzün hızlı değişen iş ortamında önemli pazarlama fonksiyonlarını yerine getirmede yardımcı olmaları için bilgisayarlara yönelmeye başladılar. Bu aşamada bilgisayarlar, birçok pazarlama faaliyeti için gerekli olan bilgi akışının entegrasyonunu sağlayan pazarlama bilişim sistemleri için katalizör olmuşlardır. Pazarlama bilişim sistemleri pazarlama fonksiyonunun temel bileşenlerinin planlanması ve kontrolü için bilgi sağlarlar (O'Brien, 1997, s.241).

Tablo-1.2'de işletmelerin pazarlama fonksiyonlarında kullanılan bilgisayar tabanlı bilişim sistemlerinden bazıları ve bunların nerelerde kullanıldıkları gösterilmiştir

Tablo 1.2. Pazarlamada Kullanılan Bilgisayar-Tabanlı Bilişim Sistemi Örnekleri (O'Brien, 1997, s. 242)

Satış Yönetimi	Satış elemanlarının performanslarını ve ürün ve hizmetlerin satışlarını planlamak, takip etmek ve desteklemek amacıyla kullanılır.
Satış Gücü Otomasyonu	Satış elemanları tarafından gerçekleştirilen satış işlemlerinin kaydedilmesi ve raporlanması işlemlerinin ayrıca satış yönetiminden gelen satış desteği ve haberleşmenin otomasyonu
Ürün Yönetimi	Ürünlerin, ürün hatlarının ve markaların planlanması, izlenmesi ve desteklenmesi amacıyla kullanılır
Reklam ve Tutundurma	Medyanın ve tutundurma yöntemlerinin seçimine yardımcı olmak ve reklam ve tutundurma faaliyetlerinin sonuçlarını değerlendirmek amacıyla kullanılır
Satış Tahminleri	Kısa ve uzun vadeli satış tahminleri üretmek amacıyla kullanılır.
Pazarlama Araştırmaları	Pazar değişimleri, gelişmeleri ve trendleri hakkında dahili ve harici veri toplamak ve analiz etmek amacıyla kullanılır.
Pazarlama Yönetimi	Şirket hedeflerine, pazarlama araştırmaları ve satış faaliyetleri verilerine dayalı pazarlama stratejileri ve planları geliştirmek, ayrıca pazarlama faaliyetlerini izlemek ve desteklemek amacıyla kullanılır.

1.5.3. Finans Bilişim Sistemleri

Bilgisayar tabanlı finans bilişim sistemleri finans yöneticilerini iki alanda destekler, bunlardan birincisi işletmenin finansmanı diğeri ise işletme içindeki finansal kaynakların kontrolü ve dağıtımıdır. Temel finans bilişim sistemleri kategorileri, nakit ve teminat yönetimi, sermaye bütçelemesi, finansal tahminler ve finansal planlama olarak sayılabilir. Tablo-1.3 önemli finansal bilişim sistemleri örneklerini özetlemektedir.

Tablo 1.3. Finans Bilişim Sistemleri Örnekleri (O'Brien, 1997, s.257)

Nakit ve Menkul Kıymetler Yönetimi	Verileri kaydeder ve nakit tahsilatları ve harcamaları için tahminler üretir ve kısa vadeli menkul kıymetlere yapılacak yatırımları yönetir.
Sermaye Bütçelemesi	Planlanan sermaye harcamalarının finansal etkilerini ve karlılığı değerlendirir.
Finansal Tahmin	İş ve ekonomi trendlerini ve finansal gelişmeleri tahmin eder.
Finansal Planlama	Hali hazırdaki ve gelecekteki finansal performansı ve işletmenin finansman ihtiyaçlarını değerlendirir.

1.5.4. Muhasebe Bilişim Sistemleri

Muhasebe bilişim sistemleri işletmelerde en eski ve en yaygın kullanılan bilişim sistemleridir. Muhasebe bilişim sistemleri işletme faaliyetlerini ve diğer ekonomik hareketleri kaydeder ve raporlarlar. Bilgisayar tabanlı muhasebe sistemleri bir organizasyonda fonların akışını kaydeder, raporlar, bilanço ve gelir tablosu gibi önemli finansal tablolar üretirler. Bu sistemler ayrıca gelecekteki durumlar için tahminler de üretirler (O'Brien, 1997, s.253). Tablo-1.4 önemli muhasebe bilişim sistemi amaçlarını özetlemektedir.

Tablo 1.4. Yaygın Olarak Kullanılan Muhasebe Bilişim Sistemi Örnekleri (O'Brien, 1997, s.255)

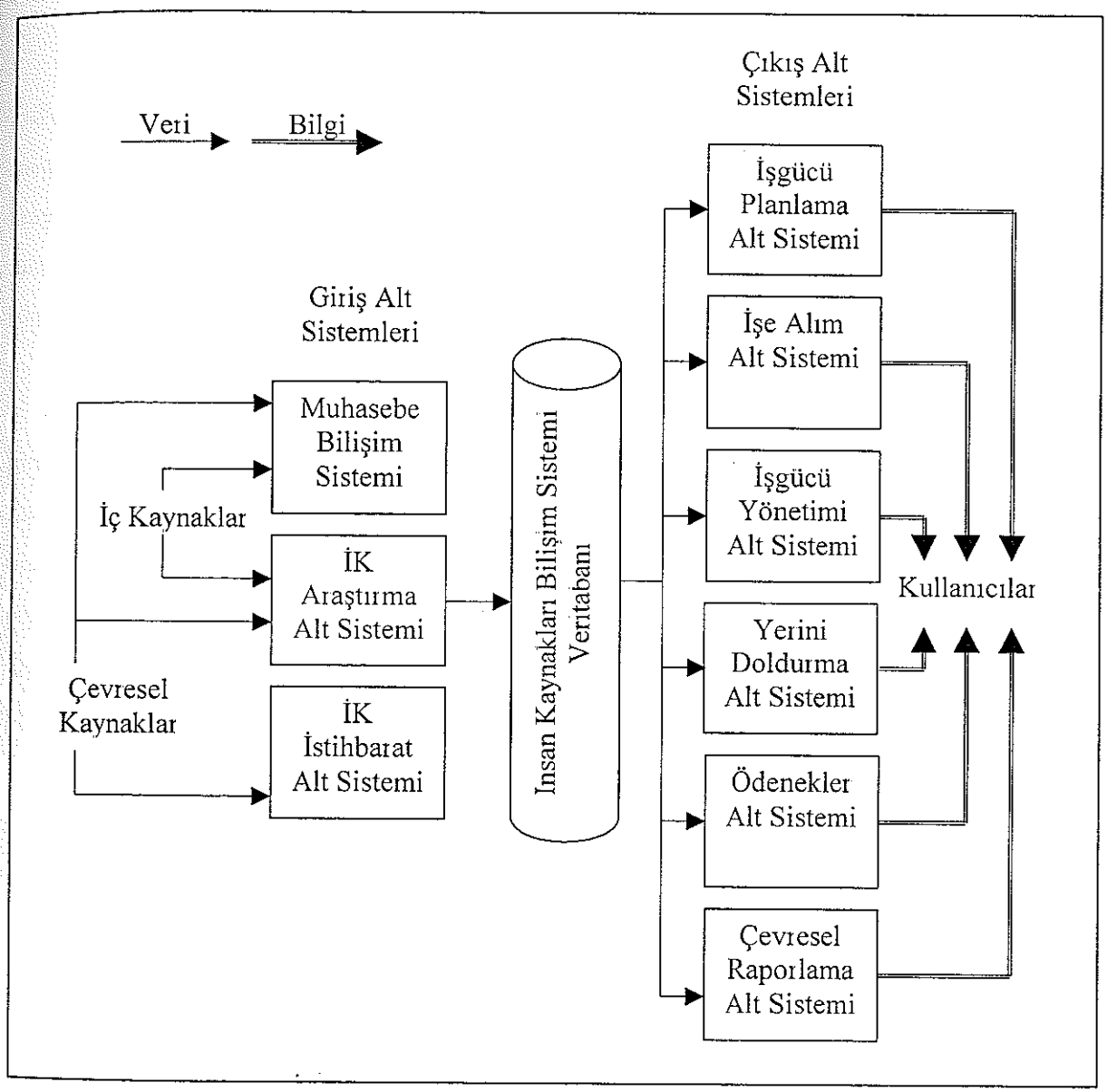
Sipariş İşleme	Müşteri siparişlerini alır, işleme koyar ve müşteri faturalarını düzenler.
Envanter Kontrolü	Envanterde oluşan değişiklikleri yansıtan veri üretir, ayrıca teslimat ve yeniden sipariş bilgileri üretir
Alacak Hesapları	Müşterilerde bulunan hesapları kaydeder ve aylık müşteri tabloları ve kredi yönetimi raporları üretir
Borç Hesapları	Alımları, ödenecek hesapları, tedarikçilere yapılacak ödemeleri kaydeder ve nakit yönetimi raporları hazırlar.
Bordro	İşçilerin çalışma ve tazminat verilerini kaydeder ve ödeme çeklerini, diğer bordro dokümanlarını ve raporları düzenler
Büyük Defter	Diğer muhasebe sistemlerinden gelen verileri birleştirir ve periyodik finansal tabloları ve işletme raporlarını düzenler.

1.5.5. İnsan Kaynakları Bilişim Sistemleri

İnsan kaynakları yönetimi fonksiyonu işe alım, yerleştirme, değerlendirme, yerini doldurma ve organizasyon çalışanlarının gelişimini içerir. İnsan kaynakları bilişim sistemi insan kaynakları yönetimi fonksiyonunu destekler. Bu işletme fonksiyonu, işletmenin personel ihtiyaçlarının planlanması, çalışanların potansiyellerinin geliştirilmesi ve bütün personel politikalarının ve programlarının denetlenmesi işlevleri üzerine odaklanmıştır (O'Brien, 1997, s.250). İnsan kaynaklarının yerine getirdiği dört temel faaliyetten söz edebiliriz (McLeod, Jr., ve Schell, 2000, s.443).

- İş ilanı ve işe alım
- Eğitim ve geliştirme
- Veri yönetimi
- İşten ayrılma ve tazminat yönetimi

İnsan kaynakları bilişim sistemini diğer işletme bilişim sistemlerinden ayıran bir özelliği gerçekleştirdiği uygulamaların alanının genişliğidir. Bu farklılık Şekil-1.4'de gösterilen modelde altı çıktı alt sistemi olarak yansıtılmıştır (McLeod, Jr., ve Schell, 2000, s.445).



Şekil 1.4. Bir İnsan Kaynakları Bilişim Sistemi Modeli (Mcleod, Jr., ve Schell, 2000, s.445)

1.6. Sistem Geliştirme Yöntemleri

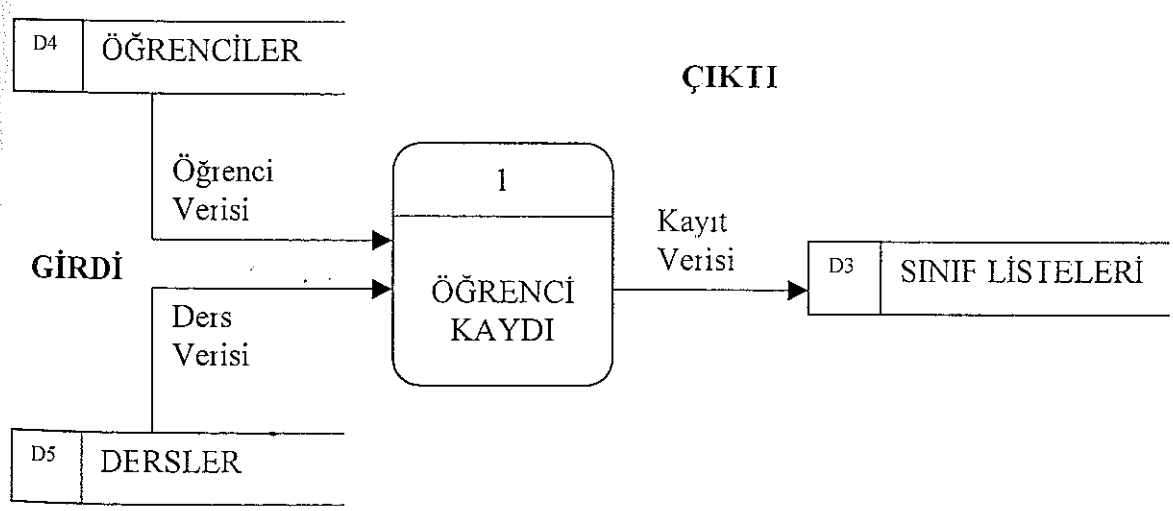
Bu bölümde bilgisayar tabanlı bilişim sistemlerinin geliştirilme yöntemlerine değinilecektir. Bu bağlamda iki farklı yöntemden söz edebiliriz. Sistem geliştirmede kullanılan geleneksel yöntem Yapısal Analiz Yöntemi olmakla birlikte günümüzde Nesne-Temelli (Object-Oriented) Analiz de yaygın olarak kullanılmaktadır. Hiçbir yöntem tek başına

en iyi çözümü sunmamaktadır. İhtiyaçlar ve imkanlara göre hangi yöntemin kullanılacağına veya birden fazla yöntemin kullanılmasına karar verilebilir.

1.6.1. Yapısal Analiz

Yapısal analiz sistem girişlerini, işlemleri ve çıkışlarını tanımlamak için çokça kullanılan yukarıdan aşağıya metodudur. Kullanışlı detay seviyelerini gösteren bir sistemi modül dizilerine paylaştıran bilgi akışının mantıksal grafik modelini sunar. Her bir modül ve arayüz arasında varolan işleri veya dönüşümleri belirtir. Bunun aracı da sistem bileşen işlemlerinin ve bileşenler arasındaki veri akışının bir grafik gösterimi olan veri akış şeması (data flow diagram)'dır (Karahoca ve Karahoca, 1998, s.494).

Yapısal analiz 1960'larda ayrı veri dosyalarının mainframe bilgisayarlarda işleme girmeleri temeline dayalı sistemlerin olduğu ortamda geliştirilmiştir. Bu analiz verinin kullanışlı bilgiye dönüştürülmesi işlemlerini tanımladığı için ayrıca işlem-merkezli teknik olarak da adlandırılmaktadır. İşlemlerin modellenmesine ek olarak, yapısal analiz veri organizasyonu ve yapısı, ilişkisel veritabanı tasarımı ve kullanıcı arayüzü konularını da içermektedir. Yapısal analiz bir bilişim sistemini desteklemek için Sistem Geliştirme Hayat Döngüsü olarak adlandırılan bir dizi aşama kullanır (Shelly, vd., 2001, s.1.16).



Şekil 1.5. Okul Kayıt Sistemi İçin Bir İşlem Modeli (Shelly, vd., 2001, s 1.16)

Şekil-1.5'de farklı iki kaynaktan giriş verisi kabul eden ve bunu bir çıkış verisine dönüştüren işlemi gerçekleştiren basit bir okul kayıt sistemi işlem modeli görülmektedir

1.6.2. Nesne Temelli Analiz

Yapısal analiz veri ve işlemleri ayrı bileşenler olarak ele alırken, nesne temelli analiz veriyi ve veriyi kullanan işlemleri nesnelere dönüştürerek birleştir (Shelly, vd., 2001, s.1.17). Nesne Temelli uygulamalar nesneleri kullanarak ve bunları birleştirerek hazırlanır. Bir nesne, bir banka müşterisinin tasarruf hesabına ilişkin veri seti ve faiz hesaplamaları gibi veri üzerinde gerçekleştirilecek işlemlerin tümü olabilir (O'Brien, 2001, s.158) Şekil-1.6'da banka müşterisinin tasarruf hesabına ait nesne örneği görülmektedir.

Şekil 1.6 Banka Tasarruf Hesabı Nesnesi Örneği (O'Brien, 2001, s 159)

Nesne teknolojileri ve teknikleri veri ve işlemler arasıdaki ayrımı elimine etmeyi amaçlamakta, veri ve bunlar üzerinde çalışan işlemler yerine bunları nesne olarak adlandırılan bloklarda birleştirmekte veya enkapsüle etmektedir. Bir nesnenin içerisindeki verinin oluşturulmasının, silinmesinin, değiştirilmesinin veya kullanılmasının tek yolu bu nesnenin enkapsüle edilmiş işlemlerinden birinin kullanılmasıdır. Bu işlemler metodlar olarak da adlandırılır (Hutchinson ve Sawyer, 2000, s.9.17). Nesne temelli analiz tekniklerini kullanan bir sistem analisti hali hazırda var olan nesnelerin yeni uygulamalar için yeniden kullanılma imkanını veya adaptasyonunun yapılıp yapılamayacağını değerlendirir ve işletme bilişim sistemi uygulamalarında hali hazırda var olan nesnelerle birleştirilecek yeni veya modifiye edilmiş nesneleri tanımlar (Whitten ve Bentley, 1997, s.126).

1.7. Sistem Geliştirme Hayat Döngüsü

Farklı organizasyonlar sistem geliştirme hayat döngüsüne farklı isimler verebilmektedirler. Bunlardan bazıları uygulama geliştirme döngüsü, sistem geliştirme döngüsü veya yapısal geliştirme hayat döngüsüdür. Bununla birlikte genel amaçlar aynıdır (Hutchinson ve Sawyer, 2000, s 9 5). Sistem geliştirme hayat döngüsünü (SGHD) organizasyonların bilişim sistemlerini kendi bünyelerinde geliştirmeleri için kullandıkları yöntemler bütünü olarak adlandırabiliriz. Günümüzde birçok organizasyon sistem geliştirme konusunda kim sorusuna kendi bünyelerinde ve ne sorusuna da cevap olarak sistem geliştirme hayat döngüsünü seçmektedirler (Haag, vd., 2002, s.272). Birincil olarak yapısal analizle birlikte tanımlanmasına rağmen, SGHD sistem geliştiricilerinin tipik olarak gerçekleştirdikleri faaliyet ve fonksiyonları bu faaliyet ve fonksiyonların hangi metodolojiye özgü olduğundan bağımsız olarak tanımlar (Shelly, vd , 2001, s.1.19).

Geleneksel sistem geliştirme hayat döngüsü, sistem geliştirmede bilgi teknolojileri uzmanları ile bilgi çalışanları arasındaki görev ayrımını sağlayan yapısal ve adım adım bir yaklaşımdır. SGHD'de bilgi çalışanları işletme faaliyetleri uzmanları ve kalite kontrol analistleri olup, bilgi teknolojileri uzmanları ise sistemin asıl tasarımı, uygulaması ve desteklenmesinden sorumludurlar (Haag, vd., 2002, s.272)

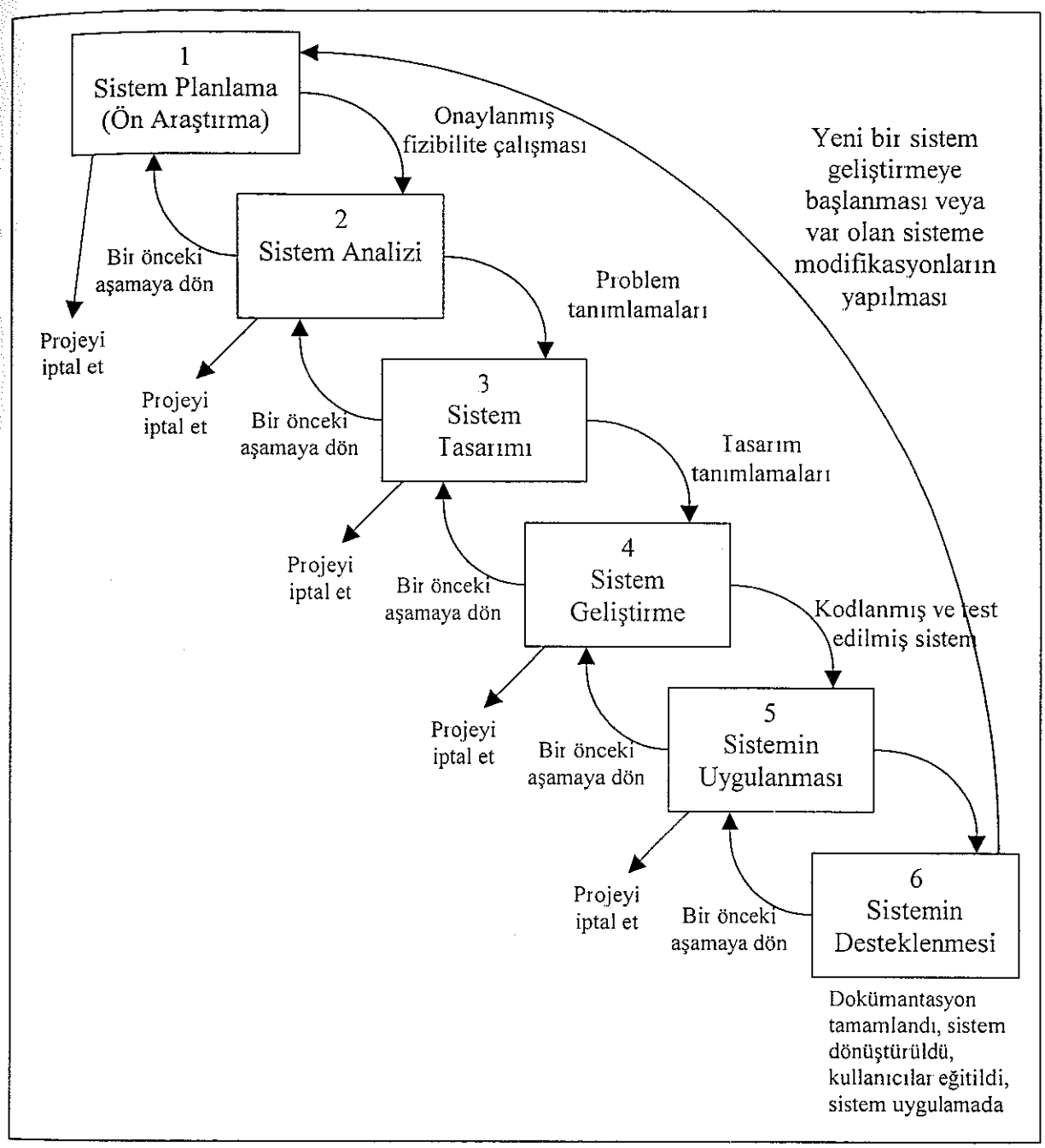
Sistem geliştirme hayat döngüsü altı aşamadan oluşur ancak bu döngüyü tamamlamak için gereken adım sayısı şirketlerin sistem geliştirmeyi hangi seviyede etkin olarak kontrol edeceklerine bağlı olarak değişebilir. Birbirinden ayrı altı SGHD aşamasından bahsediyor olsak da bir sonraki aşamaya geçilmesi için bir öncekinin tamamlanma zorunluluğu yoktur (Hutchinson ve Sawyer, 2000, s 9.5). Diğer bir deyişle aşamalar genellikle birbirinin üstüne geçişli olup, her aşamada bazı faaliyetler birbirleriyle ilişkili ve birbirlerine bağımlıdır. Bu faaliyetler kesişen-hayat-döngüsü faaliyetleri olarak adlandırılır. Bunlar problem tespiti, dokümantasyon ve sunum, tahmin ve ölçüm, fizibilite analizi, proje yönetimi ve süreç yönetimi gibi faaliyetleri kapsamaktadır (Whitten ve Bentley, 1997, s 99-100).

SGHD altı aşamadan oluşmaktadır ve bu aşamalar farklı kaynaklarda farklı biçimlerde adlandırılabilmektedir. Hutchinson ve Sawyer (2000, s.9.5)'a göre bu aşamalar şunlardır;

- Ön Araştırma
- Sistem Analizi

- Sistem Tasarımı
- Sistem Geliştirme
- Sistemin Uygulanması
- Sistemin Desteklenmesi

Literatürde birinci aşamanın farklı isimler aldığını görebiliriz. Bunlar arasında planlama (Haag, vd., 2002, s.272), sistem araştırması (O'Brien, 2001, s.406) ve sistem planlama (Shelly, vd., 2001, s.1.19) yer almaktadır. Temelde belirgin bir fark olmamasına rağmen burada sistem planlama ismi kullanılacaktır, çünkü ön araştırma planlama aşamasında iş fırsatları veya problemlerini açık olarak tanımlamak amacıyla gerçekleştirilen bir faaliyettir (Shelly, vd., 2001, s.1.20). Geleneksel olarak SGHD şelale modeliyle resmedilir (Shelly, vd., 2001, s.1.19). Şekil-1.7'de sistem geliştirme hayat döngüsü aşamaları ve bu aşamaların birbirleriyle ilişkileri görülmektedir.



Şekil 1.7. Sistem Geliştirme Hayat Döngüsü Aşamaları (Hutchinson ve Sawyer, 2000, s.9.6)

1.7.1. Sistem Planlama

Sistem planlama genellikle bilgi teknolojileri bölümüne gelen sistem talebi olarak adlandırılan ve bir işletme sürecinde veya bilişim sisteminde istenilen değişikliklerin veya problemlerin tanımlandığı resmi bir istek ile başlar. Bir sistem talebi üst düzey bir yöneticiden, planlama ekibinden, bölüm yöneticisinden veya bilgi teknolojileri bölümünün kendisinden gelebilir. Bu istek çok önemli veya nispeten daha küçük çaplı olabilir. Temel istek yeni bir bilişim sistemini veya güncel talepleri karşılayamayan mevcut sistemin

yenilenmesini içerebilir. Bunun tersine küçük istek bilişim sisteminde yeni bir özellik isteği veya kullanıcı arayüzlerinin değiştirilmesi olabilir (Shelly, vd , 2001, s.1.20).

Birinci aşamanın hedefleri; ön analizin yürütülmesi, alternatif çözümlerin önerilmesi, her çözümün maliyetlerinin ve yararlarının tanımlanması ve önerilerle birlikte bir ön planın hazırlanmasıdır (Hutchinson ve Sawyer, 2000, s.9.9).

- Ön analizin yürütülmesi: Bu süreç amaçların belirtilmesi, problemin özelliklerinin ve konusunun tanımlanmasını içermektedir.
- Alternatif çözümlerin önerilmesi: Sistemin tek başına bırakılması, daha etkin hale getirilmesi veya yeni bir sistem geliştirilmesi
- Her çözümün maliyetlerinin ve yararlarının tanımlanması: Burada maliyet yarar dengeleri tanımlanır.
- Ön hazırlık planının önerilerle birlikte gönderilmesi: Tüm bulgular yazılı bir rapor olarak sunulur.

Planlama aşamasının amacı sıklıkla fizibilite çalışması olarak da adlandırılan ön araştırmanın yapılarak problemin özelliklerinin ve konusunun açık bir şekilde saptanması ve tanımlanmasıdır Ön araştırma kritik bir adımdır, çünkü buradaki sonuçlar bütün sistem geliştirme sürecini etkiler Burada sonuç iş düşüncelerini tanımlayan, beklenen faydaları gözden geçiren ekonomik, teknik ve operasyonel faktörlere dayalı faaliyet yöntemi öneren bir rapordur (Shelly, vd , 2001, s.1.20).

Fizibilite çalışmalarının amacı alternatif sistem çözümlerinin değerlendirilmesi ve en yapılabilir ve istenilen bilişim sistemi uygulamasının geliştirme için önerilmesidir. Önerilen bir bilişim sisteminin yapılabilirliği Tablo-1.5'de gösterilen dört temel kategoride değerlendirilebilir (O'Brien, 2001, s.409).

Tablo 1.5. Organizasyonel, Ekonomik, Teknik ve Operasyonel Yapılabilirlik Faktörleri (O'Brien, 2001, s 409)

Organizasyonel Yapılabilirlik	Ekonomik Yapılabilirlik
• Önerilen sistem organizasyonun bilişim sistemi önceliklerini ne kadar iyi destekliyor olduğu	• Maliyet tasarrufları • Artan gelir • Azalan yatırım gereksinimleri • Artan kar marjları
Teknik Yapılabilirlik	Operasyonel Yapılabilirlik
• Donanım, yazılım ve bilgisayar ağları yeterliliği, güvenilirliği ve kullanılabilirliği	• Çalışan, müşteri ve tedarikçi kabulü • Yönetim desteği • Devletin istekleri ve diğer gereklilikler

1.7.2. Sistem Analizi

Sistem analizi aşamasının amacı işletme ihtiyaçlarının anlaşılması ve yeni sistemin mantıksal bir modelinin geliştirilmesidir. Bu aşamada ilk adım işletme süreçlerinin tanımlandığı ihtiyaçların modellenmesidir. Diğer adımlar olan veri modellemesi, süreç modellemesi ve nesne modellemesi süresince sistemin desteklemesi gereken işletme süreçlerinin mantıksal bir modeli geliştirilir. Model kullanılan metodolojiye bağlı olarak çeşitli tiplerde diyagramlar içerir (Shelly, vd., 2001, s 1 20).

Sistem analizi bir ön çalışma değildir. Bu aşama yeni bilişim sisteminin tasarımı için temel olarak kullanılacak fonksiyonel gereksinimleri çıkaran derin bir son kullanıcı bilişim ihtiyaçları çalışmasıdır. Sistem analizi geleneksel olarak aşağıdaki maddelerin detaylı çalışmasını içerir (O'Brien, 2001, s.410):

- Bir şirketin veya son kullanıcıların bilişim ihtiyaçları.
- Güncel olarak kullanılan bilişim sistemlerinin faaliyetleri, kaynakları ve ürünleri.
- Var olan bilişim ihtiyaçlarını ve kullanıcı ihtiyaçlarını karşılayacak bilişim sistemi yetenekleri.

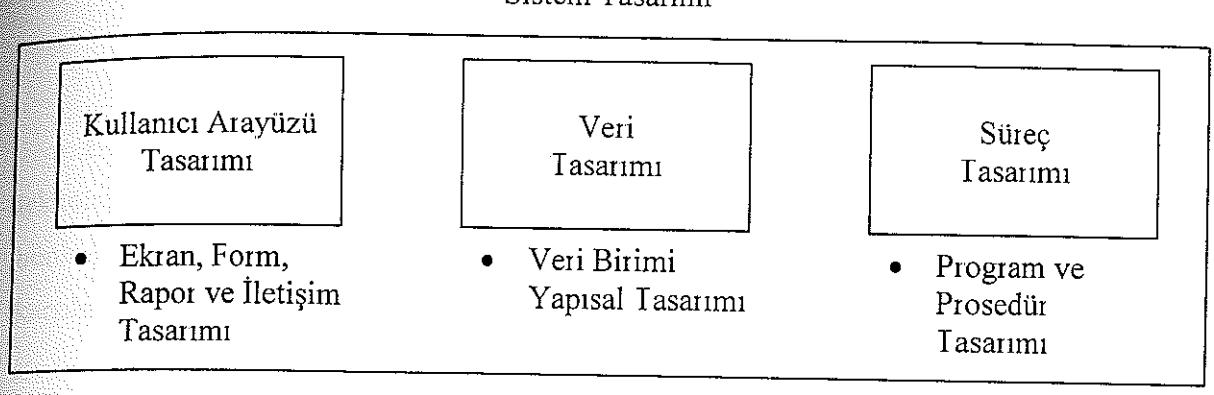
Sistem analizi aşaması veri toplanması, verinin analiz edilmesi ve rapor hazırlanması süreçlerini kapsar ve aşağıdaki gibi alt işlemlere ayrılabilirler (Hutchinson ve Sawyer, 2002, s.9.12-13,18):

1. Veri Toplanması: Veri toplanmasında sistem analistleri birçoğu teknik olmayan araçlar kullanırlar. Bunlar;
   a. Yazılı dokümanlar
   b. Yüzyüze görüşmeler
   c. Anketler
   d. Gözlemler
   e. Örnekleme
2. Verinin Analiz Edilmesi: Veri toplandıktan sonra bunların analiz edilmesine ve anlaşılmasına ihtiyaç vardır. Bunun için kullanılabilecek birçok analitik ve modelleme aracı vardır.
   a. Bilgisayar destekli yazılım mühendisliği araçları
   b. Veri akış diyagramları
   c. Sistem akış şemaları
   d. Bağlantı diyagramları
   e. Veri ilişki şemaları
   f. Karar tabloları
   g. Nesne-temelli analiz
3. Raporun Yazılması: Analizin tamamlanmasından sonra, bu aşamanın dokümante edilmesi gereklidir. Bu yönetim raporu üç kısımdan oluşmalıdır.
   a. Birinci kısım var olan sistemin nasıl çalıştığını açıklamalıdır.
   b. İkinci kısım var olan sistemin problemlerini açıklamalıdır.
   c. Son bölüm ise yeni sistemin gereksinimlerini tanımlamalı ve bunun için öneriler sunmalıdır.

1.7.3. Sistem Tasarımı

Sistem analizi kullanıcıların bilişim ihtiyaçlarını karşılamak için sistemin ne yapması gerektiğini tanımlarken sistem tasarımı sistemin bu amacı nasıl gerçekleştireceğini açıklar. Sistem tasarımı sistem analizi sürecinde geliştirilen fonksiyonel gereksinimleri karşılayan sistem tanımlamalarını üreten tasarım faaliyetlerini içerir. Şekil-1.8'de sistem tasarımının üç

temel ürününe odaklanan bir yaklaşım gösterilmektedir. Bu çerçevede sistem tasarımı üç faaliyet içermektedir; kullanıcı arayüzü, veri ve süreç tasarımı (O'Brien, 2001, s.413)



Şekil 1.8. Sistem Tasarımı Faaliyetleri (O'Brien, 2001, s.413)

Sistem tasarımının amacı dokümante edilmiş bütün ihtiyaçları karşılayan yeni sistem için bir tasarı hazırlamaktır. Sistem tasarımı sırasında bütün gerekli çıktılar, girdiler, arayüzler ve süreçler tanımlanır. Buna ek olarak sistemin güvenilir, doğru, desteklenebilir ve güvenli olmasını garanti eden bilgisayar-temelli ve manuel özellikleri içeren iç ve dış kontroller tasarlanır (Shelly, vd., 2001, s.1.20).

SGHD'nin üçüncü aşamasında önerilen bilişim sisteminin başlangıç taslağı ve sonrasında detaylı taslağı hazırlanır. Sistem tasarımı aşaması ön tasarımın yapılması, detaylı tasarımın yapılması ve rapor hazırlanmasını içerir (Hutchinson ve Sawyer, 2002, s.9.18,20-21).

1. Ön tasarımın yapılması: Ön tasarım önerilen bilişim sisteminin genel fonksiyonel yeteneklerini tanımlar. Ön tasarım ve detaylı tasarım aşamasında kullanılabilecek bazı araçlar şunlardır (Hutchinson ve Sawyer, 2002, s.9.18);
   a. Bilgisayar destekli yazılım mühendisliği araçları
   b. Proje yönetimi yazılımı
2. Detaylı tasarımın yapılması: Detaylı tasarım önerilen bilişim sisteminin ön tasarımda tanımlanan genel yetenekleri ne şekilde götüreceğini tanımlar. Detaylı tasarım genellikle sistemin aşağıdaki bölümlerini göz önüne alır (Hutchinson ve Sawyer, 2002, s.9.20).
   a. Çıktı gereksinimleri
   b. Girdi gereksinimleri

c. Depolama gereksinimleri
d. İşlem ve ağ gereksinimleri
e. Sistem kontrolleri ve yedekleme

3. Raporun yazılması: Ön ve detaylı tasarımın tüm işi bittikten sonra geniş ve detaylı bir rapor hazırlanır (Hutchinson ve Sawyer, 2002, s.9.21).

Bu aşamada sistem analistinin rolü kalite güvencedir. Sistem tasarımı aşamasında bilgi teknolojileri uzmanları görevlerinin birçoğunu tamamlarlar fakat, sistem analisti mutlaka bunların önerilerini gözden geçirmelidir ve önerilen teknik çözümün mantıksal gereklilikleri karşıladığından emin olmalıdır (Haag, vd., 2002, s. 280).

1.7.4. Sistem Geliştirme

Sistem geliştirme aşamasında sistem analisti veya organizasyondan bir yetkili yazılımın geliştirilmesi veya edinilmesi ve donanım edinilmesini gerçekleştirir ve sistem test edilir. Projenin büyüklüğüne bağlı olarak bu aşama önemli ölçüde para ve zaman harcanmasını gerektirebilir (Hutchinson ve Sawyer, 2002, s.9.21)

Sistemin test edilmesi birim testi ve sistem testi olmak üzere iki aşamada gerçekleştirilir:

- Birim testi: Birim testi aşamasında test programın bağımsız parçaları içinde yapılır. Eğer program birden fazla programcının işbirliği ile yazılmışsa programın her parçası ayrı ayrı test edilir.
- Sistem testi: Sistem testinde parçalar arasında bağlantılar oluşturulur ve test verisi parçaların birlikte çalışıp çalışmadığını görmek amacıyla kullanılır. Bu noktada gerçek organizasyon verisi sistemi test etmek amacıyla kullanılabilir. Sistem ayrıca çöküp çökmeyeceğini görmek amacıyla hatalı veri ile ve aşırı yoğun veri ile de test edilir (Hutchinson ve Sawyer, 2002, s.9.22).

Eğer sistem geliştirme hayat döngüsü süresince bilgisayar destekli yazılım mühendisliği araçları kullanılmışsa test aşaması minimize edilmiştir çünkü otomatik olarak üretilmiş herhangi bir program kodu çok büyük olasılıkla hatadan bağımsızdır (Hutchinson ve Sawyer, 2002, s.9.22)

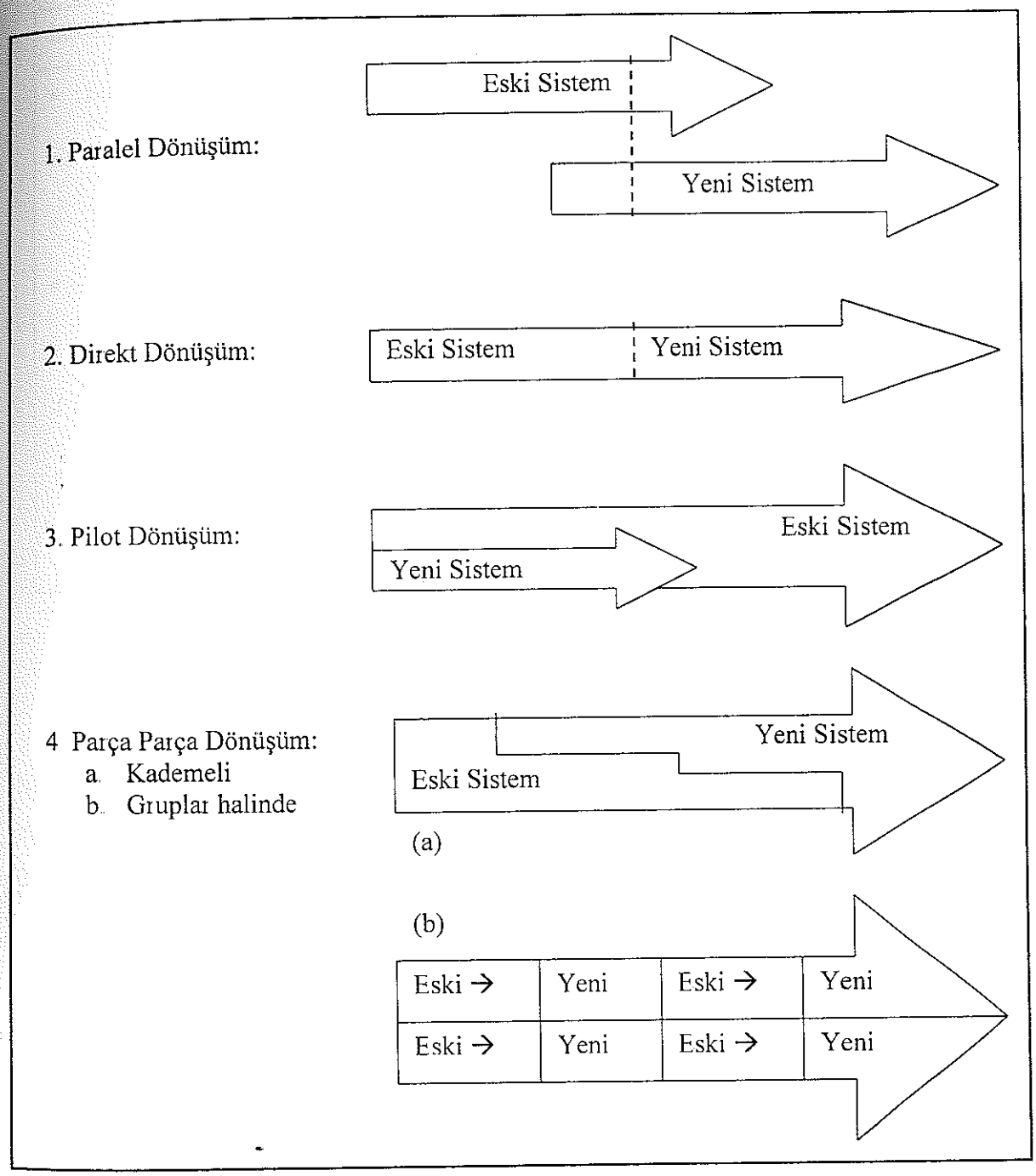
1.7.5. Sistemin Uygulanması

Sistemin uygulanması aşamasında yeni sistem yapılandırılır. Geliştiriciler yapısal analiz veya nesne-temelli yöntemlerden hangisini kullanıyor olurlarsa olsunlar süreçler aynıdır; programlar yazılır, test edilir, dokümantasyonu hazırlanır ve sistem kurulur. Eğer sistem bir paket olarak alınmışsa, sistem analisti gerekli modifikasyonları ve konfigürasyonları yapar. Uygulama aşamasının amacı tamamen fonksiyonel ve dokümantasyonu yapılmış bilişim sistemini teslim etmektir (Shelly, vd., 2001, s.1.21,22).

Uygulama aşaması süresince aşağıdaki anahtar görevler gerçekleştirilir (Haag, vd., 2002, s.282):

1. Programlama: Programlama gerekli tüm yazılım kodlarının yazılmasını içerir. Bilgi teknolojileri uzmanları bu fonksiyonu yerine getirirler ve yazılımın tamamlanması aylar ve hatta yıllar sürebilir.
2. Donanımın alınması ve kurulumu: Birçok yeni sistem yeni donanıma ihtiyaç duyar. Bu belki iş istasyonlarına dahili bellek eklenmesi kadar basit bir süreç belki de birkaç kent arasında geniş alan bilgisayar ağı kurmak kadar karmaşık olabilir.
3. Test: Yeni donanım ve yazılım yerini aldıktan sonra yeni sistemin mantıksal ihtiyaçları doğru bir şekilde karşılayıp karşılamadığından emin olunması için test edilmesi gereklidir.
4. Eğitim: Yeni sistemin doğru çalıştığından emin olunduktan sonra sistemi kullanacak kişilere eğitim verilmelidir.
5. Dönüşüm: Dönüşüm basitçe eski sistemden yeni sisteme geçiş olarak tanımlanabilir. Çeşitli dönüşüm yöntemleri vardır, Şekil-1.9'da dört farklı sistem dönüştürme stratejisi gösterilmektedir. Bu stratejiler:
   - Paralel dönüşüm; yeni sistemin doğru çalıştığı kesinleşinceye kadar eski ve yeni sistemlerin birlikte kullanılmasıdır.
   - Direkt dönüşüm; eski sistemin tamamen atılması ve bir an önce yeni sistemin kullanılmaya başlanmasıdır.
   - Pilot dönüşüm; yeni sisteme geçişin bölüm bölüm yapılmasıdır. Öncelikle bir kısım kullanıcı yeni sisteme geçer ve sistemin doğru çalıştığı görüldükçe geri kalanlar yeni sisteme geçerler.

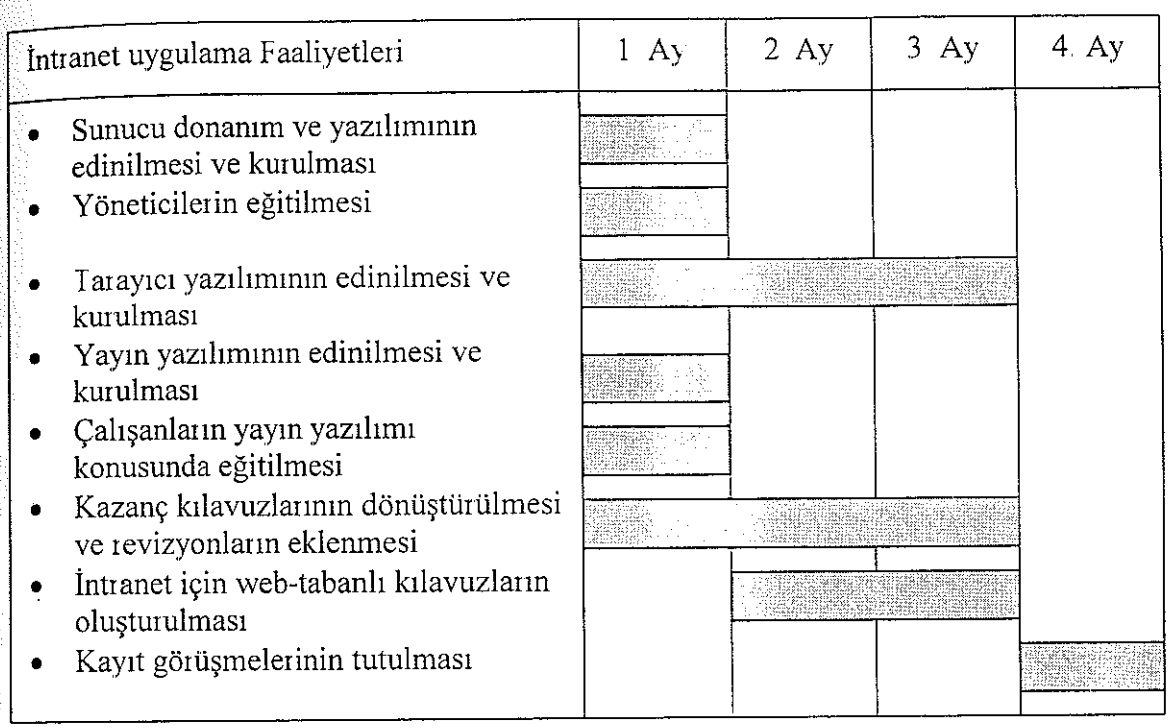
- Parça parça dönüşüm; yeni sistemin öncelikle bazı parçaları kullanıma girer ve doğru çalıştığı görüldükçe sistemin geri kalan parçaları da kullanılamaya başlanılır.



Şekil 1.9. Dört Farklı Sistem Dönüştürme Stratejisi (Hutchinson ve Sawyer, 2000, s.9.24)

Sistemin uygulanması zor ve zaman alan bir süreç olabilir. Buna rağmen yeni geliştirilen bir sistemin başarısından emin olmak da oldukça önemlidir. Doğru bir şekilde uygulanmadığında mükemmel olarak geliştirilmiş bir sistem bile başarısızlıkla sonuçlanabilir (O'Brien, 2001, s.422). Şekil-1.10'da bir şirketin insan kaynakları bölümünde yeni çalışan

kazanç sistemi için bir intranet uygulamasındaki gerekli olabilecek faaliyetler ve zaman çizelgeleri gösterilmektedir (Hills, 1997, s.193)



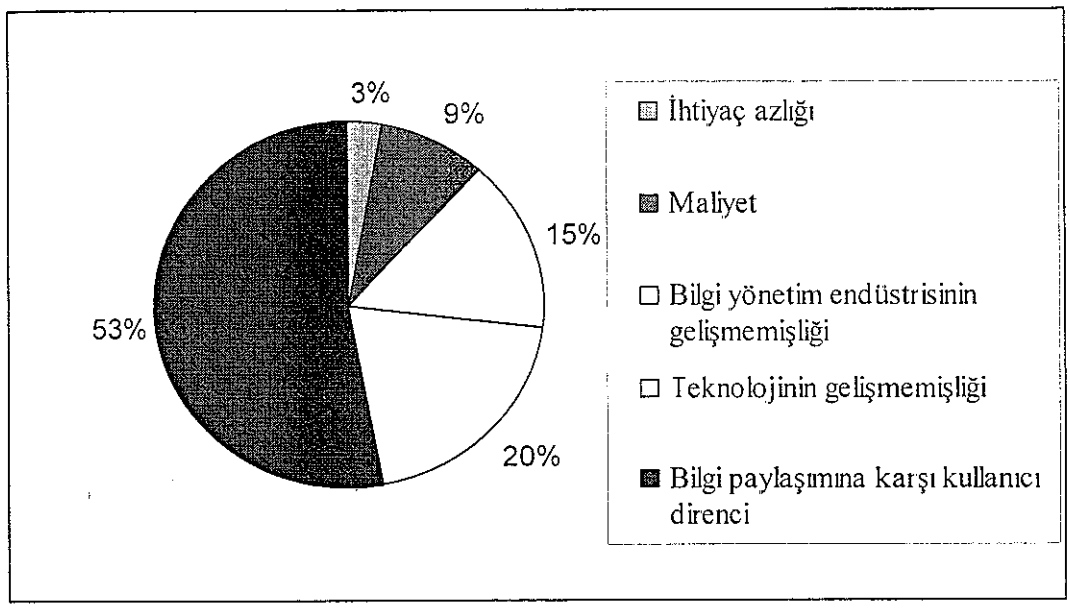
Şekil 1.10. Bir Uygulama Süreçleri Örneği (Hills, 1997, s.193)

Sistemin uygulanması aşamasının sonucunda, sistem kullanım için hazırdır. Son hazırlıklar organizasyon verisinin yeni sistem dosyalarına dönüştürülmesi, kullanıcıların eğitilmesi ve yeni sisteme asıl dönüşümün gerçekleştirilmesi süreçlerini kapsar (Shelly, vd., 2001, s.1.21).

1.7.6. Sistemin Desteklenmesi

Sistem tamamen uygulamaya geçirildikten ve işletme faaliyetlerinde kullanılmaya başlandıktan sonra sistem destek fonksiyonu başlar. Sistemin desteklenmesi operasyonel bilişim sistemlerinde gerekli veya istenilen geliştirmeleri yapmak için sistemin gözlenmesi, değerlendirilmesi ve değişikliklerin yapılmasıdır. Buna örnek olarak son kullanıcılardan gelecek geribildirimler sonucunda yapılabilecek değişiklikleri gösterebiliriz. Sistemi kullanan personel yeni sistemi çok iyi tanımadıkları için kolaylıkla hata yapabileceklerdir. Bu hatalar yeni sistemle deneyim kazanıldıkça azalacaktır ve kullanıcılar sistemin geliştirilebilecek alanlarını gösterebileceklerdir (O'Brien, 2001, s.431). Bununla birlikte karşımıza son kullanıcı direnci kavramı çıkmaktadır. Yeni bilişim sisteminin uygulanması değişikliğe karşı

çalışanlarda korkuya ve dirence neden olabilir (O'Brien, 2001, s 433). Örnek olarak Şekil-1 11'de işletmelerdeki bilgi yönetim sistemlerinin karşılaştığı önemli engeller gösterilmiştir. Burada kullanıcıların bilgi paylaşımına karşı gösterdikleri direnç en önemli engel olarak görülmektedir (Gomolski, 1997, s 6).



Şekil 1.11. Bilgi Yönetim Sistemlerinin Engelleri (Gomolski, 1997, s.6)

Sistemin desteklenmesi herhangi bir sistem geliştirme aşamasının son adımıdır. Bu adımın amacı sistemin belirtilen hedefleri karşılamaya devam ettiğinden emin olunmasıdır (Haag, vd., 2002, s.283)

Sistem yeni problemler oluşturacak kadar eskidiğinde, sistem geliştirme hayat döngüsü bir yenilenmeyi dizayn etmek ve geliştirmek için yeniden başlar. Büyük şirketlerde tipik bir SGHD süreci haftalar veya aylarla değil, yıllarla ölçülür (Hutchinson ve Sawyer, 2000, s.9.26)

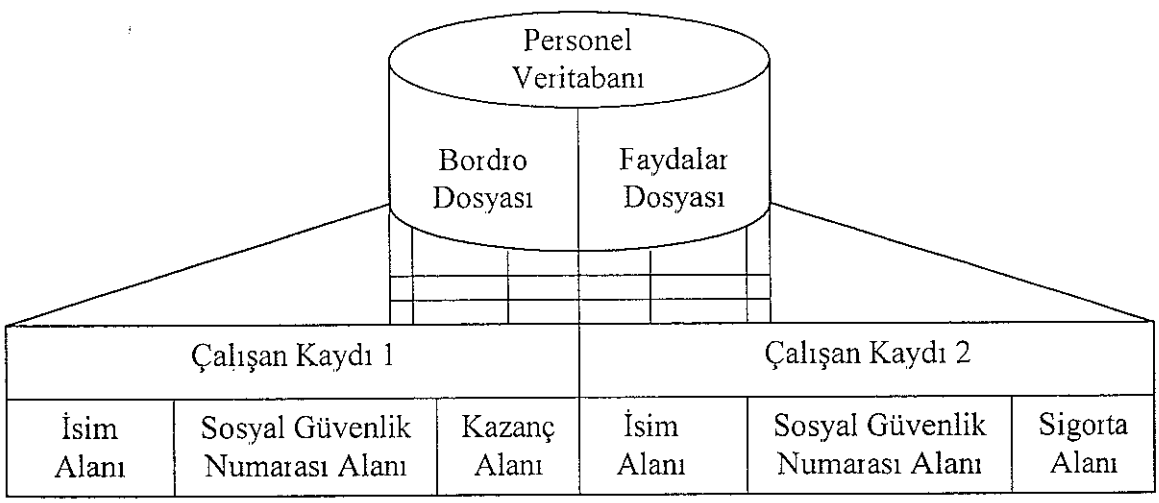
Bilişim sistemi geliştirilmesi daima ilerleyen bir çalışmadır. İşletme süreçleri büyük bir hızla değişmektedir ve birçok bilişim sistemi birkaç yıllık bir süreçten sonra yenilenmeye veya önemli oranda güncelleştirmeye ihtiyaç duymaktadır (Shelly, vd., 2001, s.1 21).

İKİNCİ BÖLÜM

VERİTABANLARI TASARIMI, KURULUMU VE YÖNETİMİ

2.1. Veritabanı Kavramları

Öncelikle bir bilişim sisteminde verinin nasıl organize edildiği ile ilgili kavramlara bakmak gereklidir. Verinin farklı birimleri veya gruplandırmaları arasında farklılık yaratan birkaç hiyerarşik seviye kurulmuştur. Bu nedenle, veri mantıksal olarak karakterlerle, alanlarla, kayıtlarla, dosyalarla ve veritabanlarıyla organize edilmiş olabilir. Bu tıpkı bir yazının harfler, kelimeler, cümleler, paragraflar ve belgelerle organize edilmesi gibidir. Bu mantıksal veri birimlerinin örnekleri Şekil-2.1'de gösterilmiştir (O'Brien, 2001, s.174).



Şekil 2.1 Bir Bilişim Sisteminde Mantıksal Veri Birimleri (O'Brien, 2001, s.174)

2.1.1. Karakter

En temel mantıksal veri birimi karakterdir. Karakter tek bir alfabetik, nümerik veya başka sembol içerir. Bir bit veya baytın daha temel veri birimi olabileceği düşünülebilir fakat bu kavramlar bilgisayar donanımı tarafından sağlanan fiziksel depolamanın birimleridir. Bundan dolayı kullanıcıların bakış açısından karakter görülebilecek ve değiştirilebilecek en temel veritabanı birimidir (O'Brien, 2001, s.174).

2.1.2. Alan

Verinin bir sonraki seviyesi alan veya veri parçasıdır. Alan bir karakter kümesinden oluşur. Örnek olarak bir kişinin ismini oluşturan alfabetik karakterler kümesi isim alanını veya satış miktarındaki sayı kümesi satış miktarı alanını oluşturur. Bir veri alanı bir varlığa (nesne, kişi, yer veya olay) ait bir özelliği temsil eder. Örnek olarak, bir çalışanın geliri işletmede çalışan birinin varlık özelliklerinden biridir ve bu tipik bir veri alanıdır (O'Brien, 2001, s.174-175).

2.1.3. Kayıt

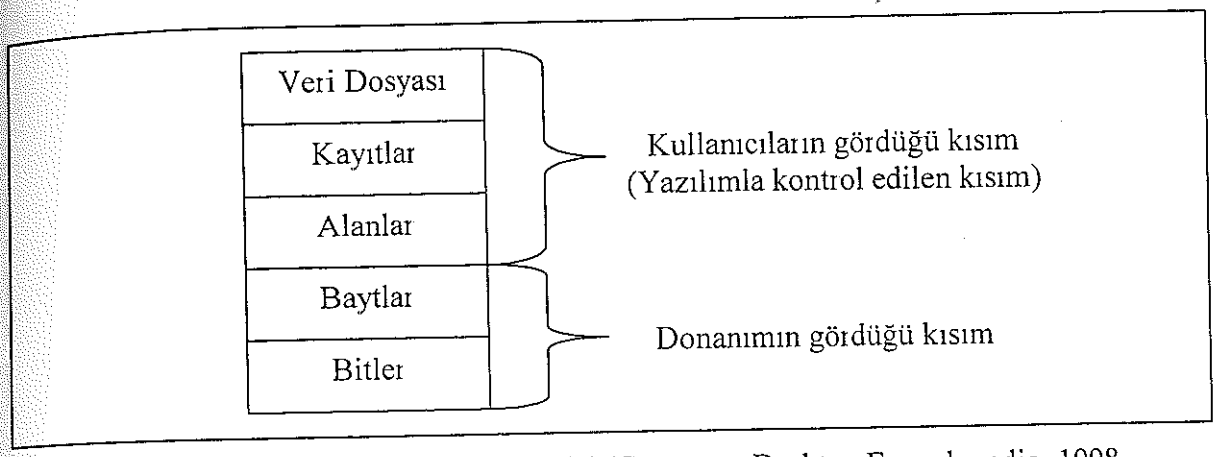
Kayıt, bir özne (ana kayıt) veya bir faaliyet (işlem kaydı) hakkında verinin saklandığı ilişkili alanlar kümesidir. Ana kayıtlar hem hesap numarası gibi kalıcı verileri hem de bilanço denkliği gibi değişken verileri içerir. İşlem kayıtları ise sadece ürün miktarı veya ürün kodu gibi kalıcı verileri içerir (http://www.techweb.com/encyclopedia/defineterm?term=record)

Verinin ilişkili alanları bir kayıt oluşturmak için gruplandırılır. Böylece bir kayıt varlığı tanımlayan özellikler toplamını temsil eder. Örnek olarak bir kişinin bordro kaydını verebiliriz. Bu kayıt kişinin adı, sosyal güvenlik numarası ve ödeme oranı gibi özellikleri tanımlayan veri alanlarından oluşur. Sabit-uzunluklu kayıtlar sabit sayıda ve sabit uzunluklu veri alanlarından oluşur. Değişken-uzunluklu kayıtlar ise değişen sayıda ve değişen uzunlukta veri alanları içerir (O'Brien, 2001, s 175).

2.1.4. Dosya

Veri dosyası bir veri kayıtları yığınıdır ve diğer dosyalardan faklı olarak kayıtlar ve alanlardan oluşur (http://www.techweb.com/encyclopedia/defineterm?term=data+file) İlişkili kayıtlar grubu veri dosyası veya tablo olarak tanımlanır. Bir çalışanlar dosyası, firmanın çalışanları ile ilgili kayıtları içerir. Dosyalar çoğunlukla bordro dosyası, envanter dosyası gibi birincil olarak kullanıldıkları uygulamalara göre sınıflandırılırlar, veya doküman dosyası, grafik dosyası gibi içerdikleri veriye göre sınıflandırılırlar. Ayrıca dosya sınıflandırmaları bunların sürekliliğine göre de yapılabilir. Bordro ana dosyası ve bordro haftalık işlem dosyası buna örnek olarak verilebilir. Buna dayanarak bir işlem dosyası bir süreçte gerçekleşen tüm

işlemlerin kayıtlarını içerir ve bu ana dosyada tutulan kalıcı kayıtları periyodik olarak güncellemekte kullanılabilir diyebiliriz (O'Brien, 2001, s 175).

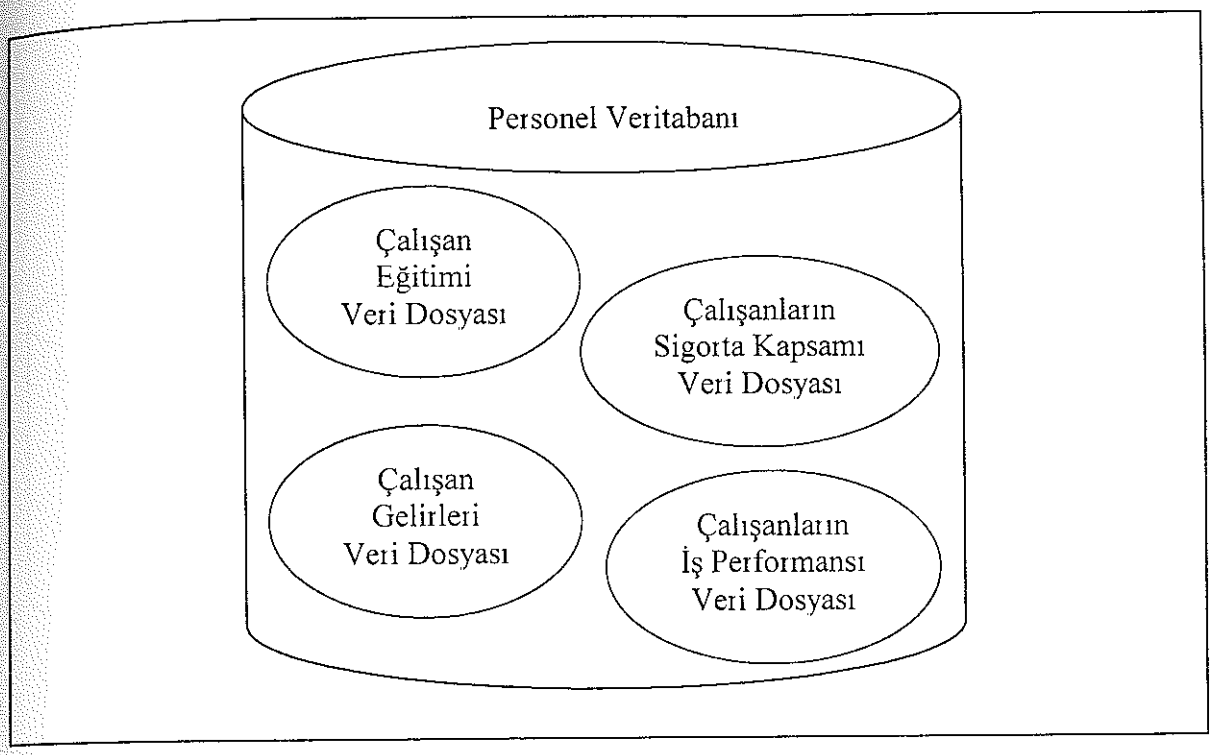


Şekil 2.2. Bir Veri Dosyası Hiyerarşisi (Computer Desktop Encyclopedia, 1998, http://www.techweb com/encyclopedia/defineterm?term=database)

2.1.5. Veritabanı

Veritabanı, bir veritabanı yönetim sistemi tarafından oluşturulan ve yönetilen ilişkili veri dosyaları setidir. Günümüzde veritabanı yönetim sistemleri metin, resim, ses ve video gibi herhangi bir formdaki veriyi yönetebilmektedir. Veritabanı ve dosya yapıları daima yazılım tarafından belirlenirken, donanım ise sadece bit ve baytlar ile ilgilenir (http://www.techweb.com/encyclopedia/defineterm?term=database)

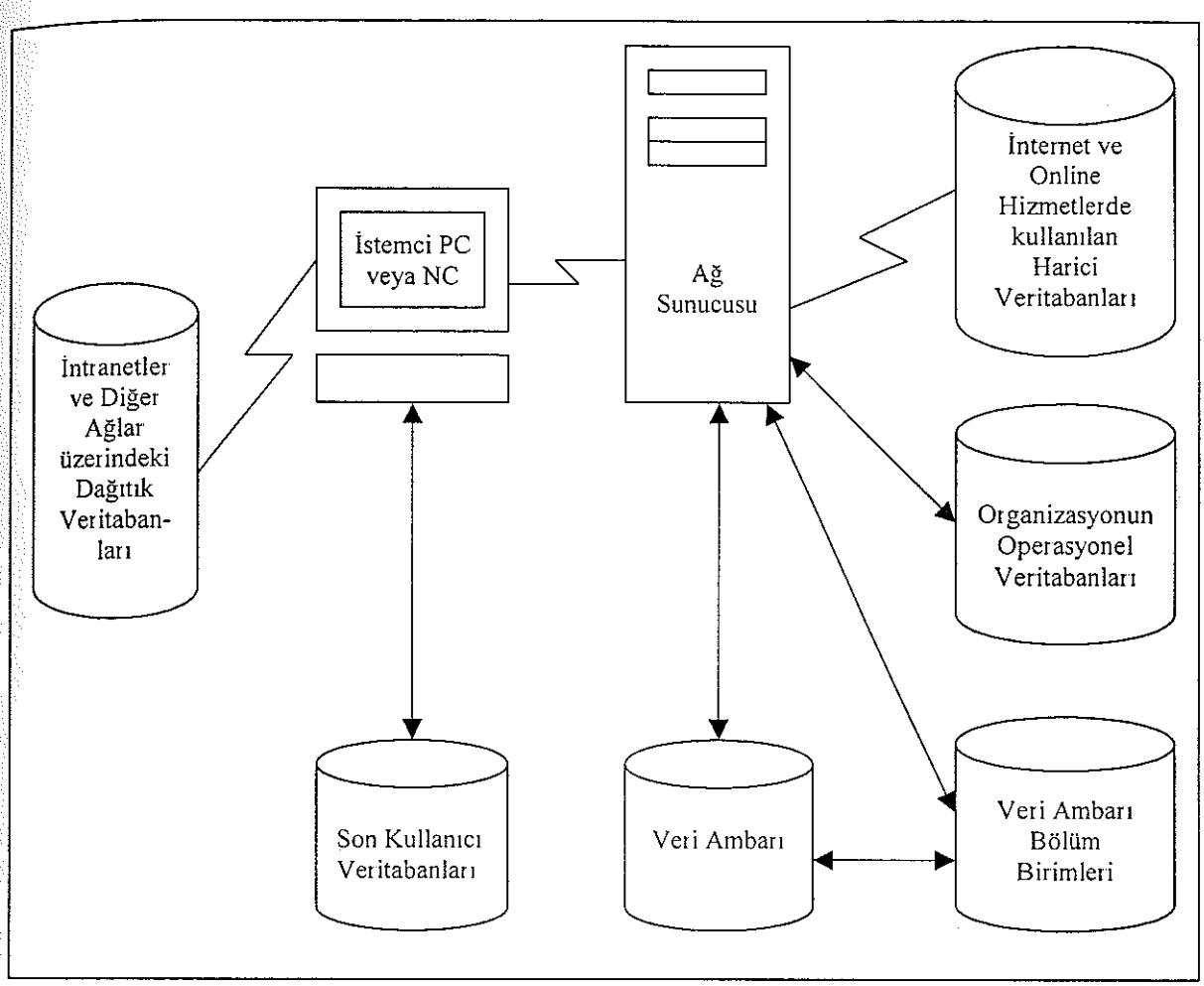
Bir veritabanı mantıksal ilişkisi olan kayıtların veya nesnelerin birbirlerine entegre toplamıdır. Nesne bir varlığın özelliklerini tanımlayan veri değerlerini ve buna ek olarak veri üzerinde gerçekleşebilecek işlemleri içerir. Bir veritabanı daha önceden farklı dosyalarda saklanan kayıtları birçok uygulamada kullanılmak üzere veri sağlamak amacıyla ortak bir veri kayıtları havuzunda toplar. Veritabanında saklanan kayıtlar onları kullanan uygulama programlarından ve üzerlerinde saklandıkları ikincil saklama aygıtlarından bağımsızdırlar (O'Brien, 2001, s.175). Şekil-2.3'de farklı dosyalardaki çalışan kayıtlarını toplayan bir personel veritabanı görülmektedir.



Şekil 2.3 Bir Personel Veritabanı Örneği (O'Brien, 2001, s.175)

2.2. Veritabanı Türleri

Bilişim teknolojilerindeki ve bunun işletme uygulamalarındaki devam eden gelişmeler birkaç temel veritabanı türü gelişimini sonuç vermiştir. Şekil-2.4 bilişim sistemi kullanan organizasyonlarda bulunabilecek veritabanlarının ana kavramsal kategorilerini göstermektedir (O'Brien, 2001, s.180-181). Kullanımda üç temel veritabanı türünden söz edebiliriz. Bunlar operasyonel, dağıtık ve harici veritabanlarıdır.



Şekil 2.4 Organizasyonlar ve Son Kullanıcılar Tarafından Kullanılan Temel Veritabanı Türleri Örnekleri (O'Brien, 2001, s.181)

2.2.1. Operasyonel Veritabanları

Operasyonel veritabanları iş süreçlerini ve elektronik ortamdaki işletme faaliyetlerini desteklemek için gereken detaylı veriyi saklar. Bu tip veritabanları ayrıca işlem veritabanları ve üretim veritabanları olarak da adlandırılır. Bunlara örnek olarak İnternet ve elektronik ticaret veritabanlarını gösterebiliriz, burada erişim eğilim verisi müşterilerin veya ziyaretçilerin şirketin web sitesindeki davranışlarını tanımlamaktadır (O'Brien, 2001, s 181).

2.2.2. Dağıtık Veritabanları

Birçok organizasyon veritabanlarının tamamını veya bazı kısımlarını kopyalayarak farklı sitelerdeki ağ sunucularına dağıtmaktadırlar. Dağıtık veritabanları internette, şirket intranet veya extranetlerinde, veya şirket ağlarında bulunan ağ sunucularında tutulabilir. Dağıtık veritabanları operasyonel veya analitik veritabanlarının, veya başka herhangi bir tür

veritabanının kopyaları olabilir. Veritabanlarının kopyalanması ve dağıtılması veritabanı performansı veya güvenliğinin arttırılması amacıyla yapılır. Dağıtık veritabanı yönetiminde en temel zorluk organizasyonun dağıtık veritabanlarında bulunan bütün verilerin sürekli ve uyumlu olarak güncellendiğinden emin olunmasıdır (O'Brien, 2001, s.181)

2.2.3. Harici Veritabanları

Harici veritabanlarından birçok bilgiye belirli ücretler karşılığında ticari online hizmetlerden, ücretli veya ücretsiz olarak özellikle world wide web olmak üzere internetteki birçok kaynaktan ulaşmak mümkündür. Web siteleri hipermedya veritabanlarında erişmek için neredeyse sonsuz farklılıkta hiperlink verilmiş çoklu ortam doküman sayfaları sağlamaktadır (O'Brien, 2001, s.181).

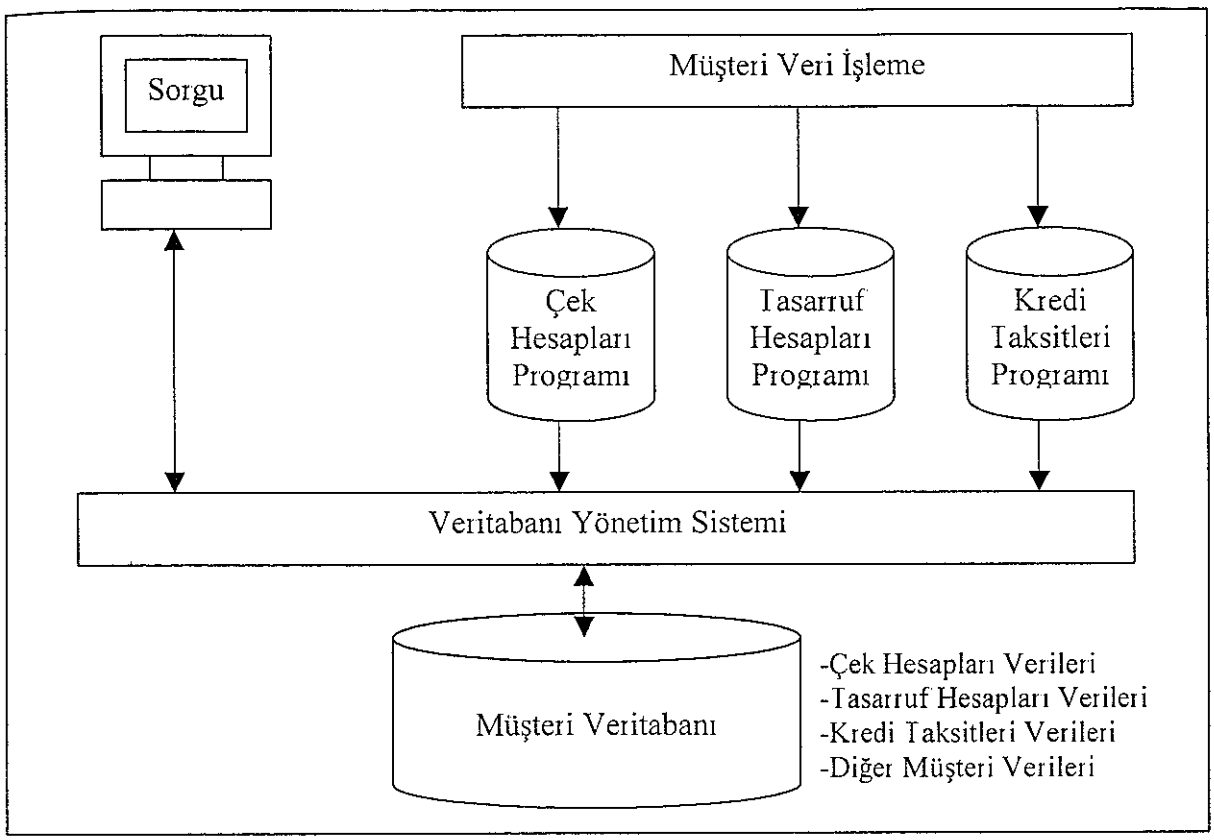
Harici veritabanlarına örnek olarak istatistiksel veri bankalarını, bilimsel veritabanlarını veya gazeteleri gösterebiliriz. İstatistiksel veri bankalarından ekonomik ve demografik faaliyetlerle ilgili istatistiksel veriler alınabilir. Bibliyografik ve tam metin veritabanlarından makalelerin, gazetelerin, dergilerin, haber bültenlerinin ve diğer yayınlanmış materyallerin özetlerini veya tamamını görüp kullanıcı bilgisayarına yüklemek mümkündür (O'Brien, 2001, s.182).

2.3. Veritabanı Yönetimi Yaklaşımı

Veritabanlarının ve veritabanı yönetimi yazılımlarının geliştirilmesi organizasyonel verinin yönetilmesinde modern yöntemlerin temelidir. Veritabanı yönetimi yaklaşımı birçok farklı uygulama programı tarafından erişilebilecek veri kayıtlarını ve nesneleri veritabanlarında birleştirir. Buna ek olarak, veritabanı yönetim sistemi olarak adlandırılan önemli bir yazılım paketi kullanıcılar ve veritabanları arasında bir yazılım arayüzü sunar. Bu arayüz kullanıcıların veritabanlarındaki kayıtlara kolaylıkla erişimini sağlar. Bu yüzden veritabanı yönetimi son kullanıcılar ve organizasyonları tarafından ihtiyaç duyulan bilgilerin sağlanması amacıyla veritabanlarının nasıl yaratılacağı, sorgulanacağı ve destekleneceğini kontrol eden veritabanı yönetimi yazılımının kullanımını da içerir (O'Brien, 2001, s.176).

Örnek olarak bankacılıkta farklı uygulamalar için ihtiyaç duyulan müşteri kayıtlarını ve çek işlemleri, ATM sistemleri, kredi kartları, tasarruf hesapları ve kredi taksitleri gibi diğer

ortak tiplerdeki veriyi gösterebiliriz. Bu veriler her uygulama için ayrı dosyalarda tutulmak yerine Şekil-2.5'de gösterildiği gibi ortak bir müşteri veritabanında birleştirilebilirler (O'Brien, 2001, s.176).



Şekil 2.5 Bankacılık Bilişim Sisteminde Bir Veritabanı Yönetimi Yaklaşımı Örneği (O'Brien, 2001, s.176)

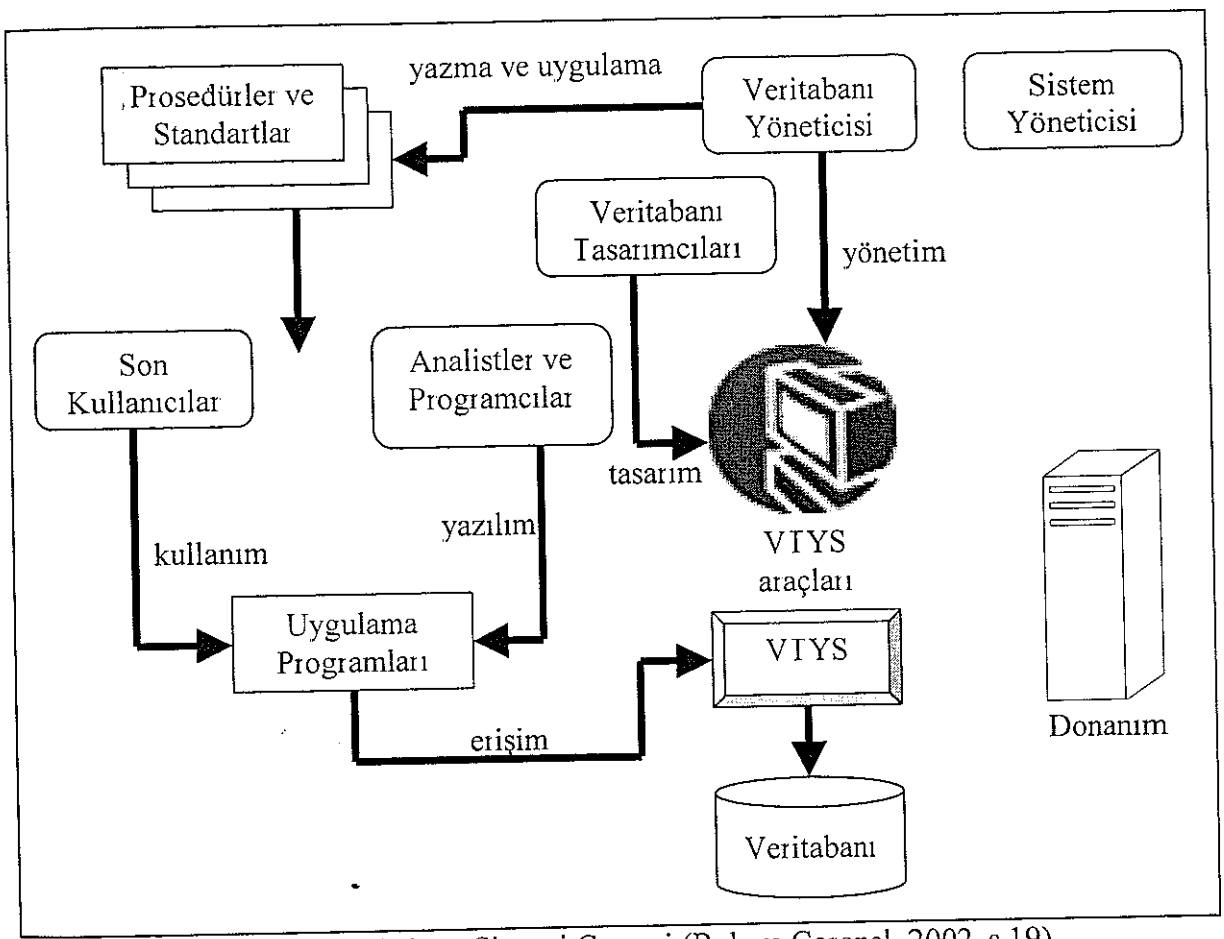
Veritabanı yönetimi yaklaşımının üç temel faaliyet içerdiğini söyleyebiliriz (O'Brien, 2001, s.176):

- Yeni işletme faaliyetlerini ve organizasyonun kayıtlarında değişiklik gerektiren diğer faaliyetleri göstermesi için ortak veritabanlarının güncellenmesi ve desteklenmesi.
- Ortak veritabanlarındaki veriyi paylaşan uygulama programlarını kullanarak her son kullanıcı uygulaması için ihtiyaç duyulan bilginin sağlanması. Bu veri paylaşımı veritabanı yönetim sistemi paketi tarafından sağlanan ortak bir yazılım arayüzü ile desteklenmektedir. Bu yüzden son kullanıcıların ve programcıların verinin fiziksel olarak nerede ve nasıl saklandığını bilmelerine gerek yoktur.
- Veritabanı yönetim sistemi yazılımı yoluyla sorgu/yanıt ve raporlama yeteneklerinin sağlanması ve böylece son kullanıcıların web tarayıcılarını, interneti

veya şirket intranetini kullanarak kolaylıkla veritabanı sorgularını gerçekleştirmeleri, raporlar üretmeleri ve bilgi için ad hoc isteklerine hızlı yanıt almalarının sağlanması.

2.4. Veritabanı Sistemleri

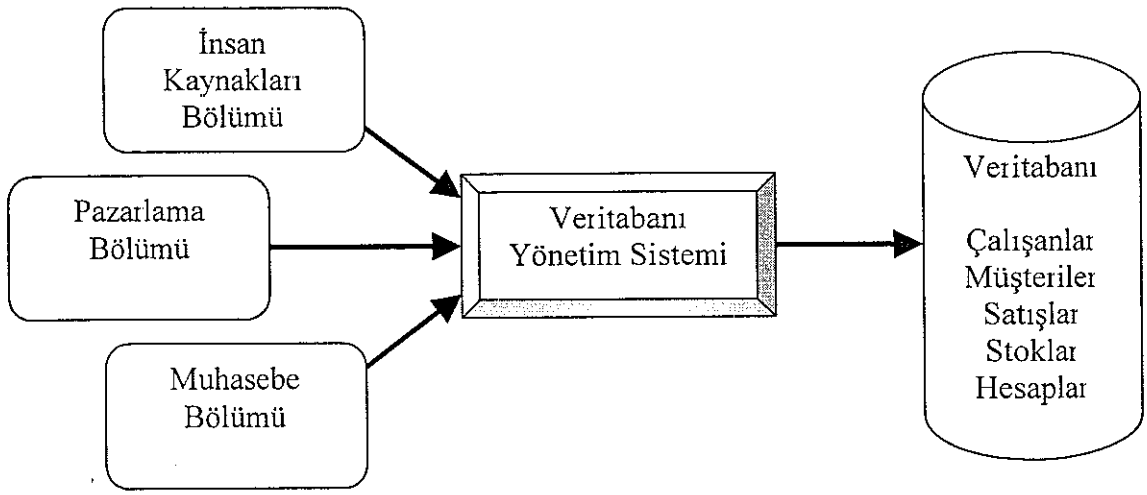
Dosya sistemlerinin birçok ayrı ve birbirleriyle ilişkili olmayan dosyalarının aksine veritabanı tek bir veri deposunda saklanan mantıksal olarak ilişkili verileri içerir. Bu yüzden veritabanı son kullanıcı verisinin saklanması, erişilmesi ve yönetilmesinde yöntem değişikliğini temsil eder (Rob ve Coronel, 2002, s.17). Şekil-2.6'da bir Veritabanı Sistemi Çevresi gösterilmiştir.



Şekil 2.6. Veritabanı Sistemi Çevresi (Rob ve Coronel, 2002, s.19)

Veritabanı Yönetim Sistemleri Şekil-2.7'de gösterildiği gibi dosya sistemi yönetimine göre birçok avantaj sağlar. Veritabanı Yönetim Sistemi (VTYS) dosya sistemi yönetiminde oluşabilecek veri tutarsızlığı, veri düzensizliği, veri bağımlılığı ve yapısal bağımlılık sorunlarını ortadan kaldırır. Yeni nesil Veritabanı Yönetim Sistemi yazılımları sadece veri

yapılarını saklamakla kalmamakta, merkezi bir konumda bu yapılar arasındaki ilişkileri ve erişim yollarını da tutmaktadır. Ayrıca yeni nesil VTYS yazılımları bu bileşenlere gereken erişim yollarının tanımlaması, saklaması ve yönetilmesini de gerçekleştirir (Rob ve Coronel, 2002, s.17)



Şekil 2.7. Bir Veritabanı Sistemi (Rob ve Coronel, 2002, s.17)

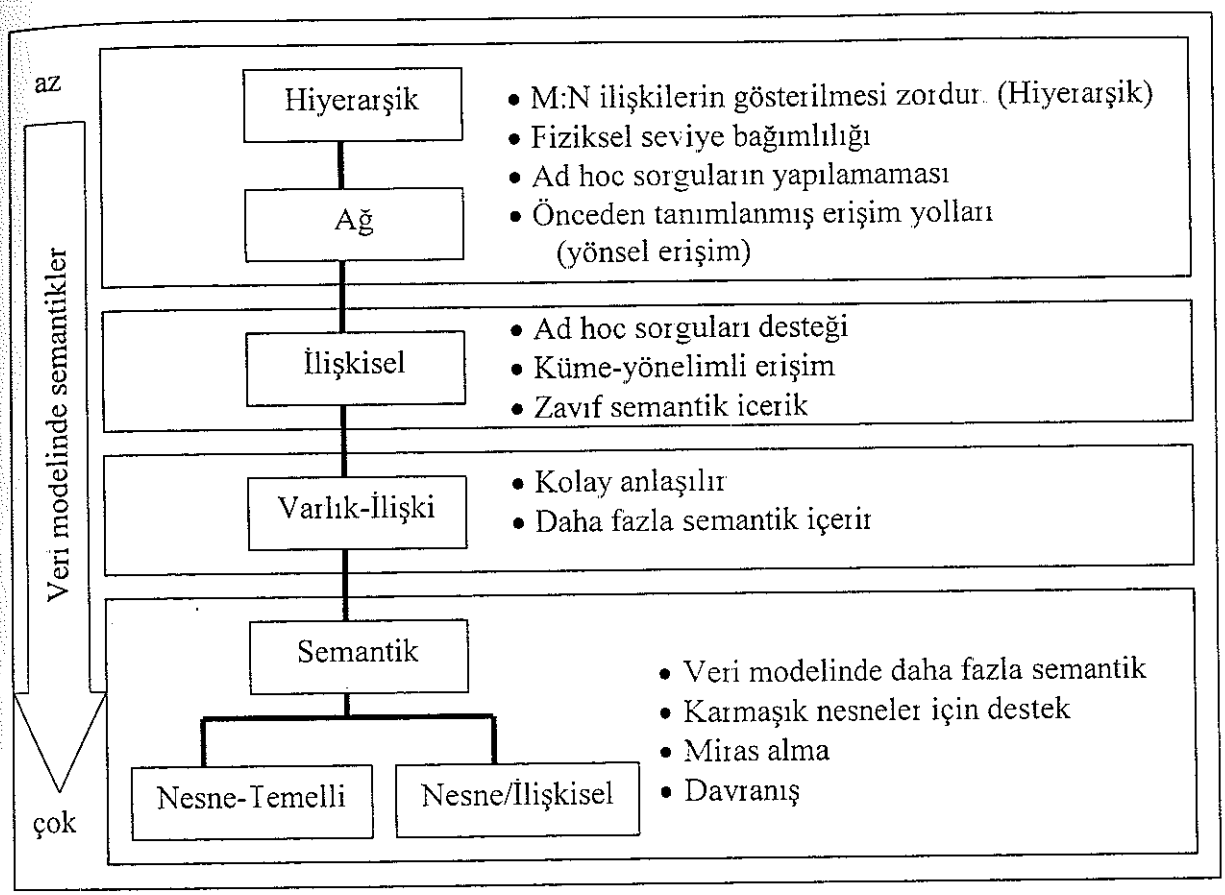
Veritabanı sistemi terimi bir veritabanı çevresinde verinin toplanmasını, saklanmasını, yönetimini ve kullanımını tanımlayan ve düzenleyen bileşenlerin organizasyonuna karşılık gelmektedir. Genel yönetim bakışı açısından bir veritabanı sistemi ilişkileri Şekil-2.6'da da gösterildiği gibi beş temel parçanın birleşmesinden oluşur. Bunlar donanım, yazılım, insanlar, prosedürler ve verilerdir (Rob ve Coronel, 2002, s.18).

2.5. Veritabanı Modelleri

Bir veritabanı modeli, veritabanı içerisinde bulunan veri yapılarını ve veri ilişkilerini temsil eden mantıksal yapıların bir araya gelmesiyle oluşur. Veritabanı modelleri kavramsal modeller ve uygulama modelleri olmak üzere iki kategoride gruplandırılabilir (Rob ve Coronel, 2002, s.23)

Kavramsal model verinin gösterilmesinin mantıksal ortamına odaklanmaktadır. Bu yüzden, kavramsal model veritabanında verinin nasıl gösterildiğinden çok neyin gösterildiği ile ilgilenir. Kavramsal modeller varlık-ilişki (E-R) modelini ve nesne-temelli modeli içerir. Kavramsal modelin tersine uygulama modeli veritabanında verinin nasıl gösterildiği veya neyin modellendiğini göstermek için veri yapılarının nasıl uygulandığı ile ilgilenir. Uygulama

modelleri de hiyerarşik, ağ, ilişkisel ve nesne-temelli veritabanı modellerini içerir (Rob ve Coronel, 2002, s.24). Aşağıdaki şekilde veritabanı modellerinin gelişimi gösterilmektedir.



Şekil 2 8 Veritabanı Modellerinin Gelişimi (Rob ve Coronel, 2002, s.43)

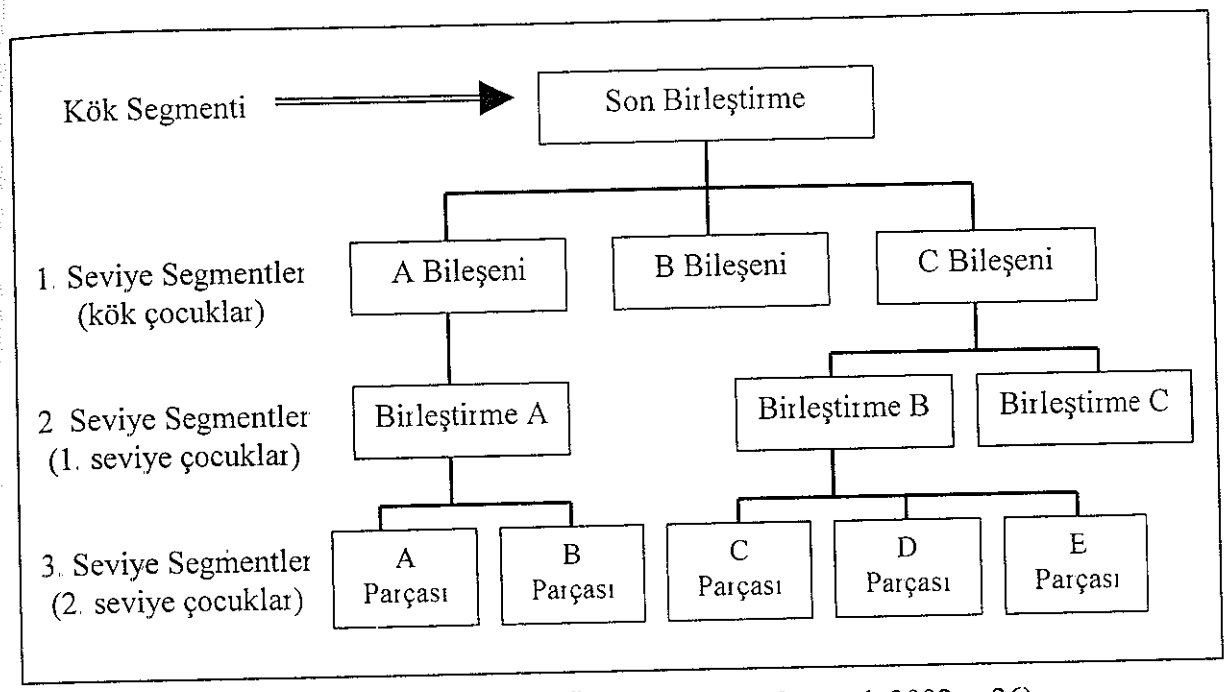
Kavramsal modeller veriler arasındaki ilişkileri tanımlarken üç tip ilişki kullanırlar: Bire çok, çoğa çok ve bire bir. Veritabanı tasarımcıları genellikle bunlar için kısaltmalar kullanırlar 1:M, M:N ve 1:1 M:N kısaltması M:M olarak da kullanılmaktadır (Rob ve Coronel, 2002, s.24)

2.5.1. Hiyerarşik Veritabanı Modeli

Hiyerarşik veritabanı modelinin var olan veritabanı pazarında çok önemli bir rolü olmamasına rağmen aşağıdaki nedenlerle bu modelin bazı karakteristikleri anlaşılmalıdır.

- Bu modelin temel kavramları sonraki veritabanı geliştirilme sürecinin temellerini oluşturur
- Modelin sınırlılıkları veritabanı tasarımına farklı bakış açısına yol açar.

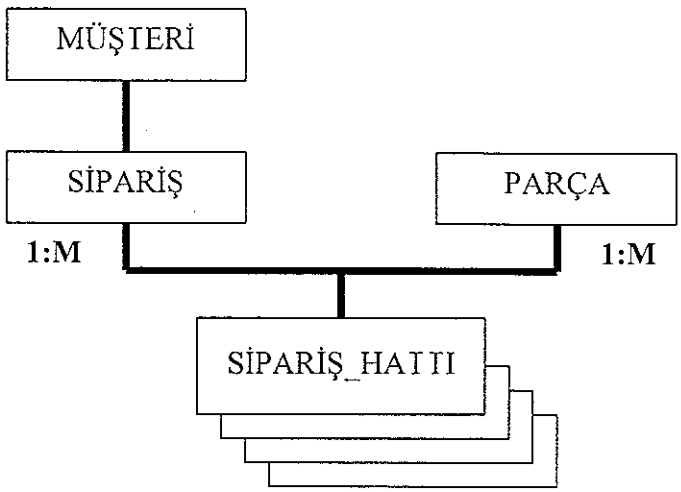
- Hiyerarşik modelin bazı temel kavramları günümüzdeki veritabanı modellerinde görülmektedir (Rob ve Coronel, 2002, s.25).



Şekil 2.9. Hiyerarşik Yapı Örneği (Rob ve Coronel, 2002, s 26)

Şekil-2.9'de de görüldüğü gibi kullanıcılar hiyerarşik veritabanını bir segmentler hiyerarşisi olarak görmektedir. Bir segment dosya sistemlerinin kayıt tipinin karşılığıdır. Diğer bir deyişle hiyerarşik veritabanı, baş aşağı duran ağaç yapısı oluşturacak şekilde mantıksal olarak organize edilmiş kayıtlar topluluğudur. Hiyerarşi içerisinde en üst seviye (kök) kendisinin altındaki segmentin ebeveyni olarak görülür. Örnek olarak Şekil-2.9'de kök segmenti birinci seviye segmentlerin ebeveynidir. Birinci seviye segmentler ise ikinci seviye segmentlerin ebeveynidir ve bu yapı bu şekilde devam eder (Rob ve Coronel, 2002, s.26).

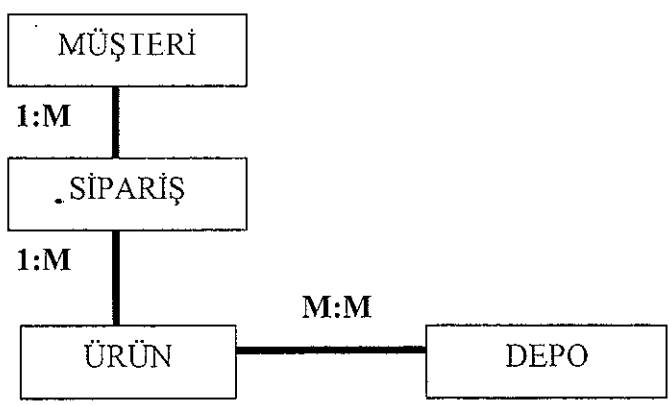
Hiyerarşik veritabanı modeli iki tip ilişkiye izin verir, bire-bir (1:1) ilişki ve bire-çok (1:M) ilişki (Senn, 1989, s.599). Hiyerarşik modelde 1:M ilişkileri kolaylıkla izlemek mümkündür, ancak bu veritabanı modeline M:M ilişkileri uyarlamak zordur. Ancak gerçek yaşamda karşılaşılabilecek bu tür yapılar çoklu ebeveynli çocuk (child with multiple parents) ilişki yapısı ile gösterilmektedir. Örnek olarak aşağıdaki Şekil-2.10'da gösterilen SİPARİŞ_HATTI iki ebeveyne sahiptir: SİPARİŞ ve PARÇA (Rob ve Coronel, 2002, s.28).



Şekil 2.10. Çoklu Ebeveynli Çocuk - Child with Multiple Parents (Rob ve Coronel, 2002, s.28).

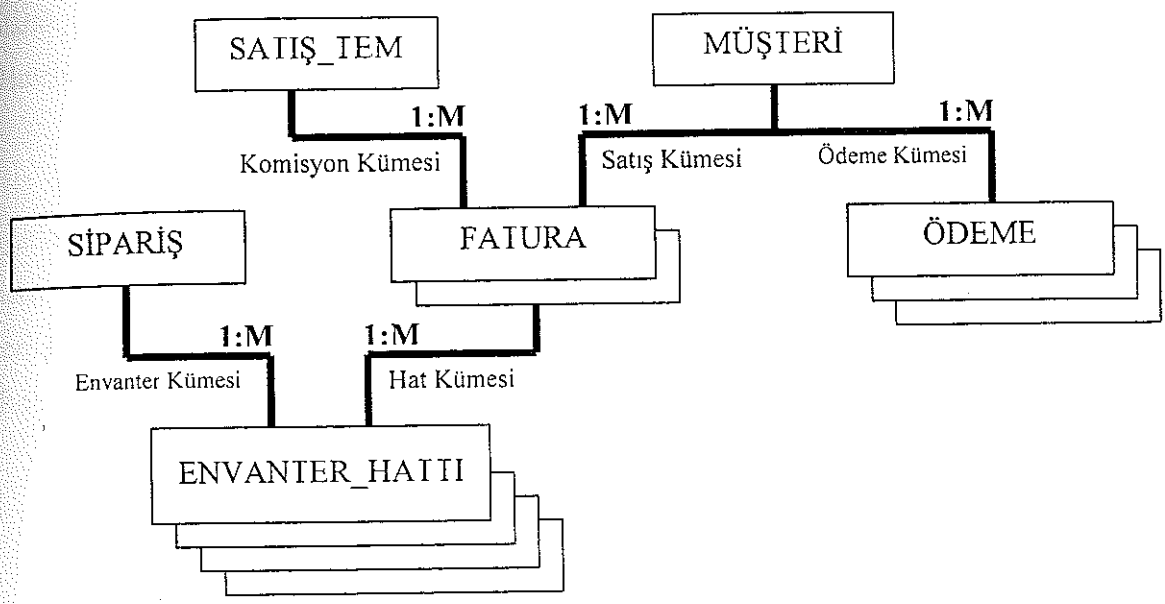
2.5.2. Ağ Veritabanı Modeli

Ağ (Network) veritabanı modeli, karmaşık veri ilişkilerini hiyerarşik modelin yapabildiğinden daha etkin bir şekilde gösterebilmek, veritabanı performansını arttırmak ve bir veritabanı standardı koymak için geliştirilmiştir. Birçok açıdan ağ modeli hiyerarşik modele benzemektedir. Örnek olarak hiyerarşik modelde olduğu gibi kullanıcı ağ veritabanı modelini 1:M ilişkilerine sahip kayıtlar topluluğu olarak algılar. Ancak, hiyerarşik modelin tersine ağ modeli bir kayıtın birden fazla ebeveyni olmasına izin verir. Bu yüzden Şekil-2.10'da gösterilen ilişkiler ağ veritabanı modeli ile kolaylıkla kullanılabilir (Rob ve Coronel, 2002, s 28-29).



Şekil 2.11. Ağ Modeli Örneği (Senn, 1989, s 602)

Şekil-2.11'da ağ modelinin sipariş sistemi örneği görülmektedir. Burada URUN ve DEPO arasında M:M ilişki vardır ve bir depo birden fazla ürünü depolayabilir, bir ürün ise birden fazla depoda saklanabilir. Yukarıda gösterilen ağ modeli hiyerarşik modelle de tasarlanabilir ancak bu durumda her bir ilişkinin tek tek tanımlanması gerekeceğinden tekrarlamalar artar ve sorgulamaların cevaplama süresi yavaşlar (Senn, 1989, s.602).



Şekil 2.12 Bir Ağ Veritabanı Modeli (Rob ve Coronel, 2002, s.30)

Ağ veritabanı terminolojisi kullanılırken bir ilişki küme (set) olarak adlandırılır. Her küme en az iki kayıt tipinden oluşur: hiyerarşik modelde ebeveyne karşılık olabilecek bir sahip (owner) kayıt ve hiyerarşik modelde çocuğa (child) karşılık olan bir üye (member) kayıt. Hiyerarşik model ile ağ modeli arasındaki fark ise bir üye kaydın birden fazla kümede görülebilmesidir. Diğer bir deyişle bir üyenin birden fazla sahibi olabilir. Bir küme, sahip ve üye arasındaki 1:M ilişkiyi temsil eder (Rob ve Coronel, 2002, s.30). Şekil-2.12'de bu ilişkilerin örneği tipik bir satış organizasyonu için gösterilmiştir.

2.5.3. İlişkisel Veritabanı Modeli

İlişkisel veritabanı modeli ilk olarak IBM'den E. F. Codd tarafından geliştirilmiştir (Senn, 1989, s.588) ve sonrasında esnek ve güçlü olduğu için popüler olmuştur (Shelly, vd., 2001, s.8.30).

Bir ilişkisel veritabanında veri tablolarda depolanır (Jacobson, vd., 1994, s. 276). Hiyerarşik ve ağ veritabanı modellerine göre daha esnek olan ilişkisel veritabanı modeli farklı dosyalardaki verileri anahtar alan veya ortak veri elemanı yoluyla ilişkilendirir veya birbirine bağlar. Bu düzenlemede herhangi bir hiyerarşi yoluyla yukarıdan aşağıya bir erişim yolu yoktur. Bunun yerine veri elemanları satırlar ve sütunlar şeklinde oluşmuş farklı tablolarda saklanırlar. Veritabanı tasarımcılarının teknik terminolojisinde tablolar ilişkiler (dosyalar) olarak, satırlar kayıtlar olarak ve sütunlar ise özellikler (alanlar) olarak adlandırılırlar (Hutchinson ve Sawyer, 2000, s 12.16)

İlişkisel veritabanı modeli çok gelişmiş ilişkisel veritabanı yönetim sistemleri yoluyla uyarlanmıştır. İlişkisel veritabanı yönetim sistemleri hiyerarşik ve ağ veritabanı yönetim sistemleri tarafından sağlanan temel fonksiyonları gerçekleştirmeye ek olarak ilişkisel veritabanı modelinin anlaşılması ve uygulanması için gereken fonksiyonların sunucusudur (Rob ve Coronel, 2002, s 32).

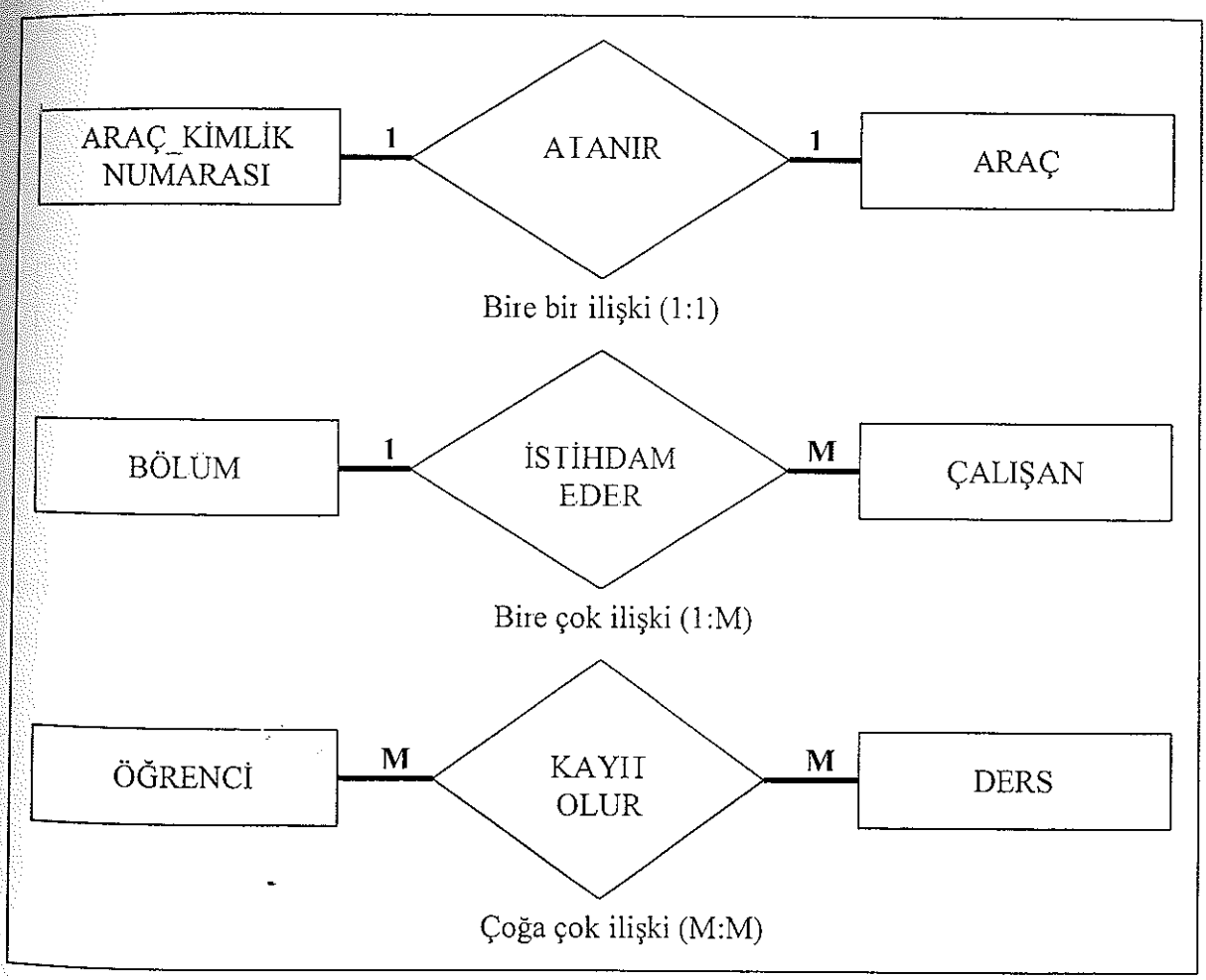
2.5.4. Varlık-İlişki Veri Modeli

İlişkisel veritabanı modelinin kavramsal sadeliği veritabanlarının alanlarının genişlemesini olanaklı hale getirmiştir. Bu yüzden ilişkisel veritabanı teknolojisinin sunumu daha fazla ve daha karmaşık işlem ve bilgi isteğini tetiklemiştir. Hızla artan işlem ve bilgi gereksinimleri daha karmaşık veritabanı uygulama yapıları ihtiyacını, dolayısıyla daha etkin veritabanı tasarım araçları ihtiyacını ortaya çıkardı. Varlık-ilişki modelleri normal olarak bir varlık-ilişki diyagramı (entity-relationship diagram, ERD) ile gösterilir. Varlık-ilişki diyagramı veritabanı bileşenlerini modellemek için grafik gösterimler kullanır (Rob ve Coronel, 2002, s.36).

Bir varlık-ilişki diyagramı bir bilişim sisteminin sistem varlıkları arasındaki ilişkileri gösteren grafik modeldir. Her varlık bir dikdörtgen ile ve varlıkları bağlayan ilişki de bir eşkenar dörtgen ile gösterilir. Varlıklar tekil isimlerle ve ilişkiler de aktif fiillerle etiketlendirilir (Shelly, vd , 2001, s.8.14)

Her varlık bir özellikler kümesi ile tanımlanır. Bir özellik varlığın karakteristiklerini ayrıntılı olarak tanımlar (Rob ve Coronel, 2002, s.37). Örnek olarak ÖĞRENCİ varlığı AD, SOYAD, DANIŞMAN, PROGRAM_ID ve ÖĞRENCİ_NUMARASI gibi özelliklere sahiptir.

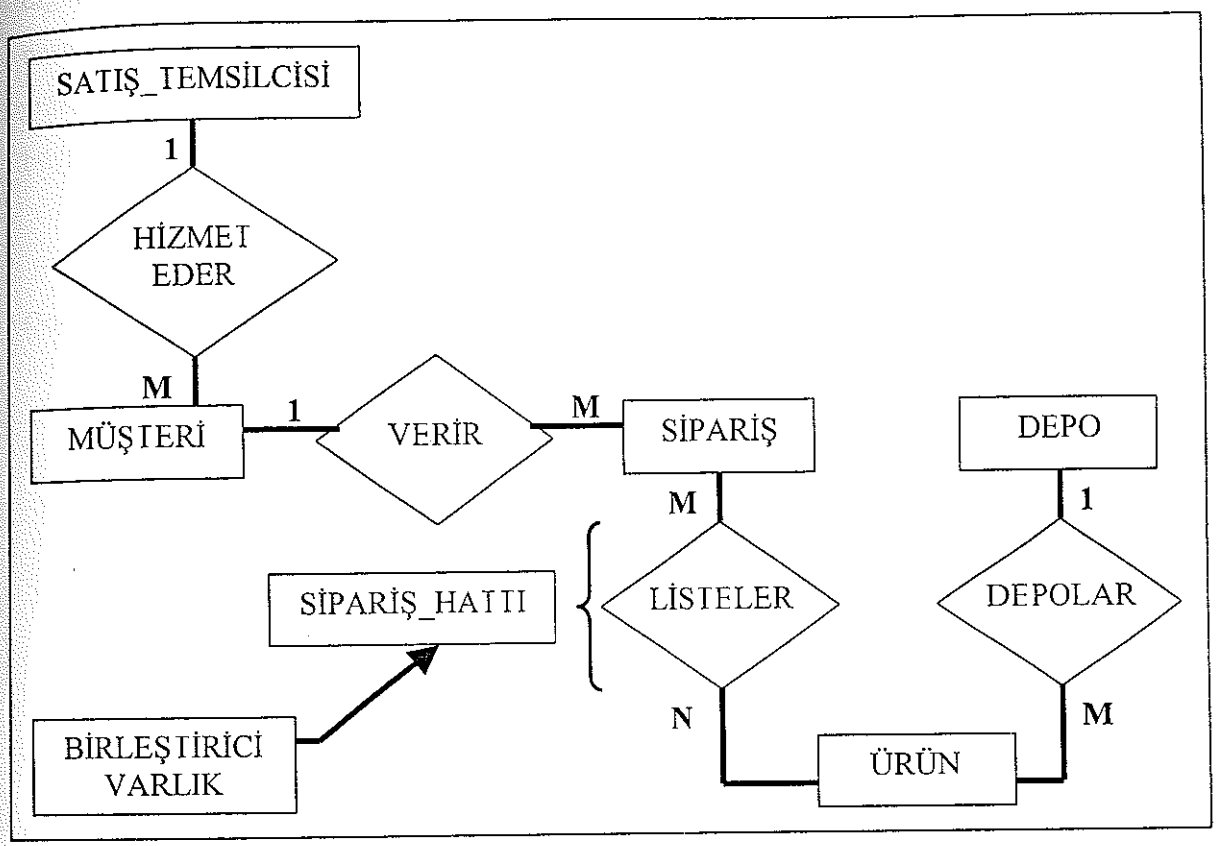
Varlıklar arasında üç temel tip ilişki vardır. Bire bir ilişki, 1:1 kısaltmasıyla gösterilir ve birinci varlığın her örneği için ikinci varlıkta 1 karşılık olduğunda vardır. Bire çok ilişki (1:M) birinci varlıktaki bir olay için ikinci varlıkta birden fazla olay olduğunda ve ikinci varlığın her olayının birinci varlıkta sadece bir olayla ilişkisi olduğunda vardır. Örnek olarak BÖLÜM ve ÇALIŞAN arasındaki ilişki bire çoktur. Bir bölümün birçok çalışanı olabilir, fakat her çalışan sadece bir bölümde çalışır. M:M veya M:N kısaltmalarıyla gösterilen çoğa çok ilişki birinci varlığın bir örneği ikinci varlıkta birden fazla örnekle ilişkili olduğunda ve ikinci varlıktaki bir örnek de birinci varlıkta birden fazla örnekle ilişkili olabildiğinde vardır. Örnek olarak ÖĞRENCİ ve DERS arasındaki ilişki çoğa çoktur. Bir öğrenci birden fazla derse kayıt olabilirken, bir ders için de birden fazla öğrenci kayıt olabilmektedir (Shelly, vd., 2001, s.8.14-15) Şekil-2.13'de bu ilişki tipleri varlık-ilişki diyagramları örnekleriyle gösterilmiştir.



Şekil 2.13 Varlık-İlişki Diyagramları (Shelly, vd., 2001, s.8.14-15)

M:M ilişki 1:1 veya 1:M ilişkilerden farklıdır, çünkü çoğa çok ilişkide iki varlığı birbirine bağlayan bir olay veya işlem aslında üçüncü bir varlıktır ve birleştirici varlık (associative entity) olarak adlandırılır. Birleştirici varlık kendi karakteristik ve özellik

kümesine sahiptir (Shelly, vd., 2001, s.8.15) Şekil-2.13'de üçüncü örnekte KAYIT_OLUR ilişkisi belirli bir derse kaydolan belirli bir öğrencinin her örneğini kaydeden KAYDOLMA varlığını temsil eder



Şekil 2.14 Bütünsel Bir Varlık-İlişki Diyagramı (Shelly, vd., s.8.16)

Bütünsel bir varlık-ilişki diyagramı sistemin bütün varlıklarını ve ilişkilerini gösterir. Şekil-2.14'de beş varlık gösterilmiştir: SATIŞ_TEMSİLCİSİ, MÜŞTERİ, SİPARİŞ, ÜRÜN ve DEPO (Shelly, vd., 2001, s.8.16) Burada SATIŞ_TEMSİLCİSİ ile MÜŞTERİ arasındaki ilişki 1:M olarak tanımlanmış ancak şirketlerin yapılanmasına göre bu bazı şirketlerde M:M olarak tanımlanabilir. Benzer şekilde DEPO ve ÜRÜN arasındaki ilişki her bir ürünün aynı depoda tutulduğunu gösterir şekilde 1:M, ancak eğer birden fazla ürünün birden fazla depoda tutulduğu durumlarda bu ilişki M:M olabilir.

2.5.5. Nesne Temelli Veritabanı Modeli

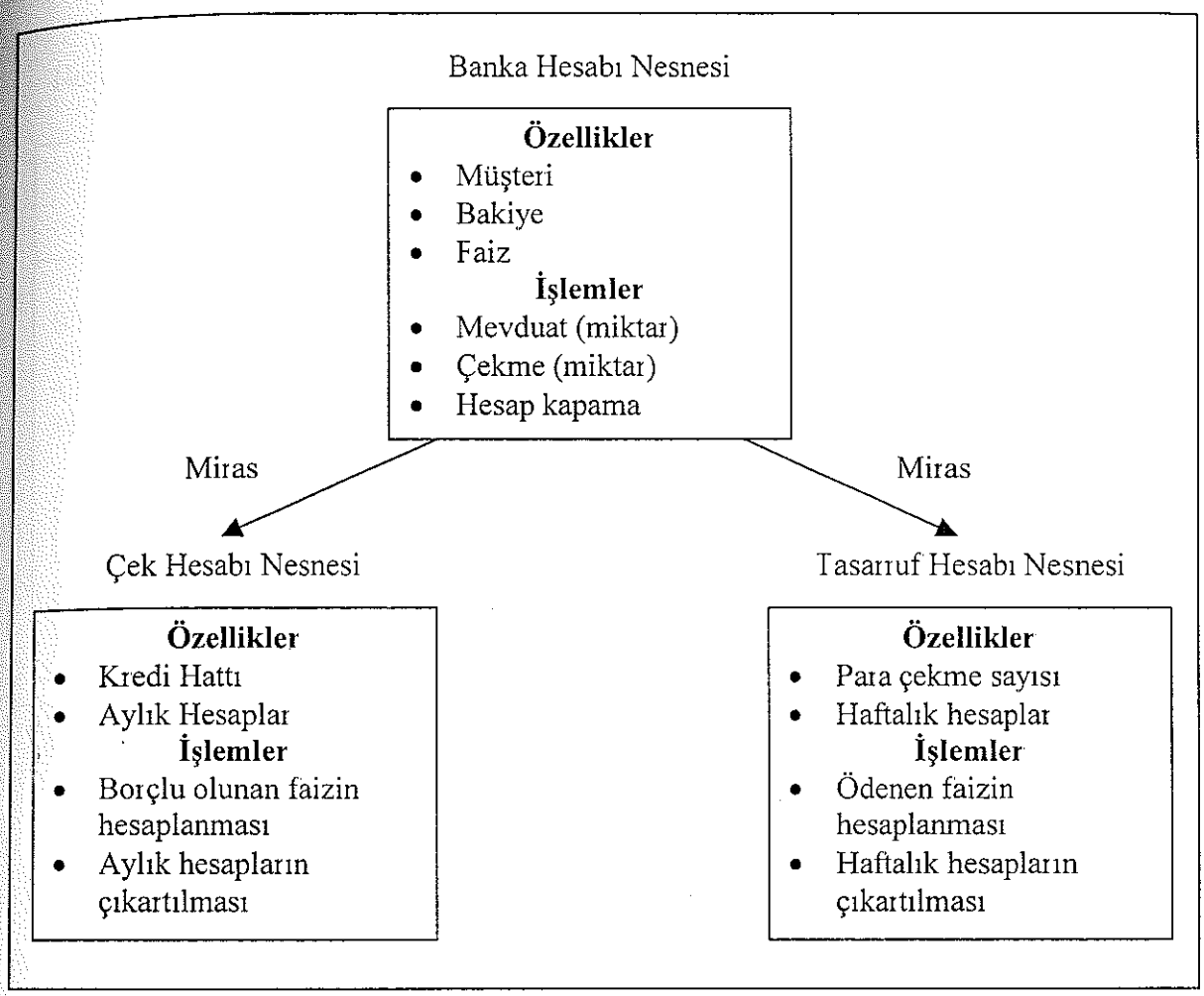
Günümüzde birçok sistem geliştirici nesne temelli veritabanı tasarımını, nesne temelli analiz sürecinin doğal bir uzantısı olarak kullanmaktadırlar. Bazı BT profesyonelleri nesne temelli veritabanlarının daha güçlü bir etkisinin olacağına ve sonuç olarak da ilişkisel yaklaşımın yerini alacağına inanmaktadırlar (Shelly, vd., 2001, s.8.32)

Nesne temelli veritabanı modeli varlıkların tanımlanması ve kullanılmasında çok farklı bir yolu yansıtmaktadır. İlişkisel veritabanı modelindeki varlık gibi, bir nesne kendisinin olgulara dayanan içeriği ile tanımlanır. Fakat varlıktan farklı olarak, bir nesne, diğer nesnelerle olan ilişkilerine ait bilgilerle birlikte, nesnenin kendi içinde yer alan olgular arasındaki ilişkiler hakkında da bilgiler içerir. Bu yüzden nesne içerisindeki olgular daha büyük anlama sahiptir (Rob ve Coronel, 2002, s.39).

Bir nesne temelli veri modeli en azından aşağıdaki bileşenlere sahip olmalıdır (Rob ve Coronel, 2002, s 40):

- Veri modelinin nesneleri gerçek yaşam varlıkları veya olaylarının çıkarımları olmalıdır
- Özellikler bir nesnenin niteliklerini tanımlamalıdır. Örnek olarak ÖĞRENCİ nesnesi AD, SOYAD, ÖĞRENCİ_NO ve ANABİLİM_DALI gibi özellikleri içermelidir.
- Benzer karakteristikleri paylaşan nesneler sınıflar (class) içerisinde gruplandırılmalıdır. Bir sınıf, paylaşılan yapı (özellikler) ve davranışları (metodlar) olan benzer nesneler topluluğudur. Genel bir bakışla sınıf, varlık-ilişki modelinin varlık setine benzemektedir. Ancak bir sınıf içerdiği işlemler kümesi (metodlar) ile bir varlık setinden farklılık göstermektedir.
- Sınıflar bir sınıf hiyerarşisi içerisinde organize edilmiştir. Sınıf hiyerarşisi her sınıfın sadece bir ebeveyninin olabileceği baş aşağı bir ağaç yapısına benzer. Örnek olarak bir MÜŞTERİ sınıfı ile bir ÇALIŞAN sınıfı ebeveyn KİŞİ sınıfını paylaşır. Bu açıdan hiyerarşik veritabanı modeli ile benzerlik gösterdiği söylenebilir.
- Bir nesne sınıf hiyerarşisi içerisinde kendisinin üstündeki sınıfların özelliklerini ve davranışlarını miras olarak alabilme yeteneğine sahiptir. Örnek olarak KİŞİ sınıfının altında yaratılacak MÜŞTERİ ve ÇALIŞAN sınıfları KİŞİ sınıfının bütün özellik ve davranışlarını alacaktır.

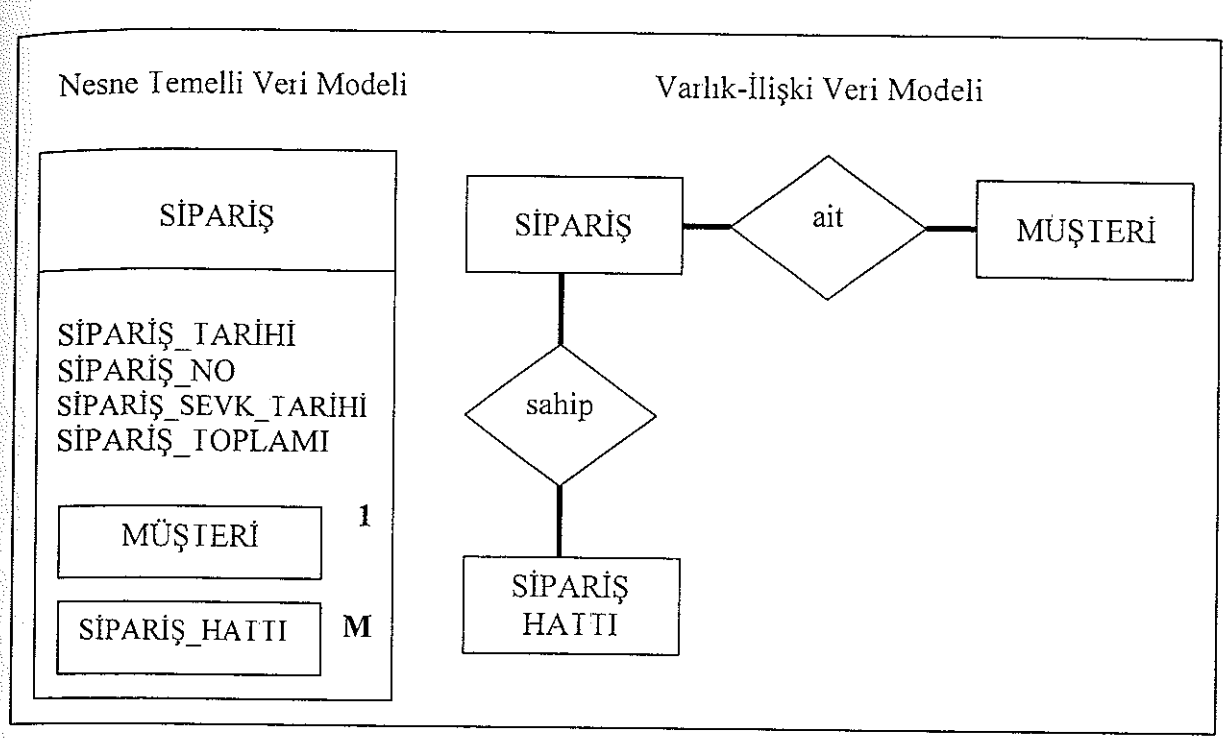
Aşağıdaki Şekil-2.15'de nesne temelli veritabanı modelinde üst sınıftan özellik ve davranışların miras alınmasına örnek olarak bir banka hesabı nesnesi örneği görülmektedir. Burada ÇEK_HESABI nesnesi ve TASARRUF_HESABI nesnesi kendilerinden bir üst sınıfta yer alan BANKA_HESABI nesnesinden ortak özellik ve işlemleri miras almaktadırlar.



Şekil 2.15 Nesne Temelli Modelde Üst Sınıftan Miras Alınması (Jacobsen, vd , 1995, s 65)

Günümüzde uygulamalar daha geniş ve daha karmaşık hale gelmektedir Uygulamaların boyutları ve karmaşıklığı arttıkça projelerin tamamlanması için gereken kişi-yıl sayısı da artmaktadır Bunun sonucu, projelerin geleneksel küçük programcı takımlarıyla yıllarca uzaması veya yeni koordinasyon ve entegrasyon sorunlarıyla birlikte takıma yeni programcılar eklenmesi olmaktadır Bu artan karmaşıklığı çözmenin kanıtlanmış yollarından biri karşımıza nesne temelli programlama olarak çıkmaktadır (Wang, 1996, s.129).

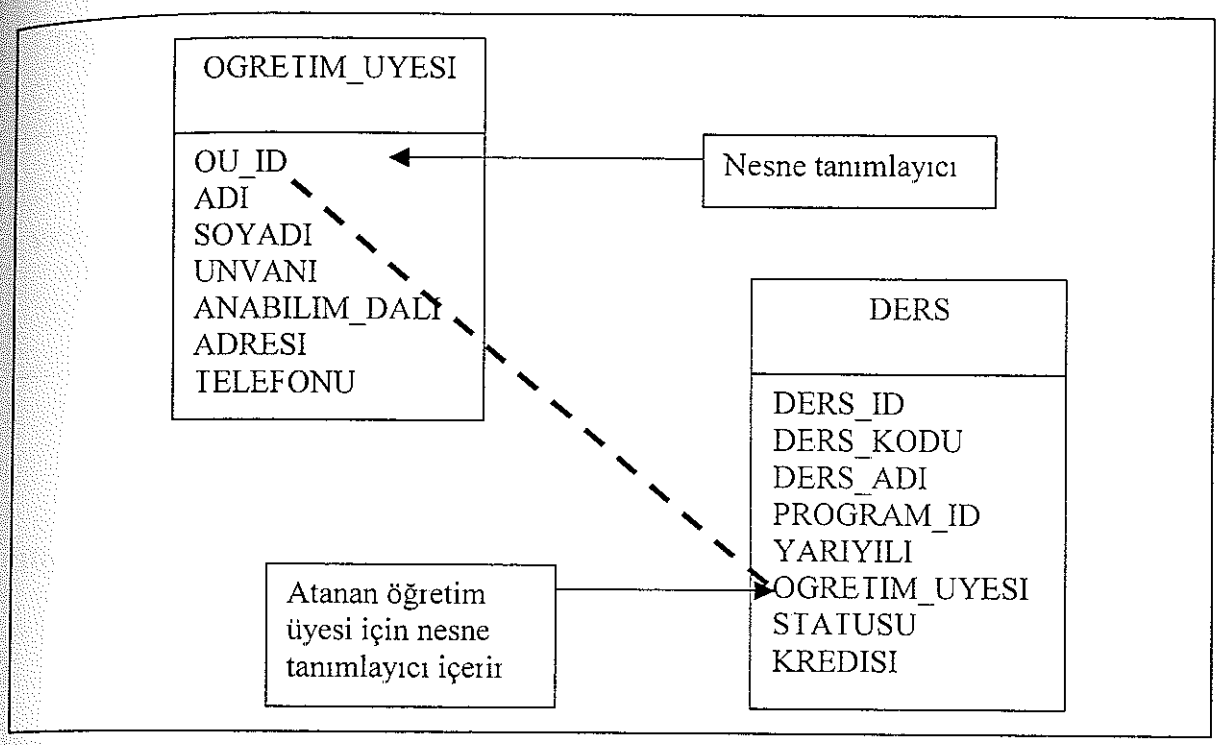
Nesne temelli veritabanı modeli ile varlık ilişki modeli arasındaki farklılıklar aşağıda Şekil-2.16'da gösterilmiştir



Şekil 2.16 Nesne Temelli Veri Modeli ile Varlık-İlişki Modelinin Karşılaştırılması (Rob ve Coronel, 2002, s 41).

Bir nesne temelli veritabanı yönetim sistemi (OODBMS) nesneleri veritabanı dosyaları içindeki elemanlar olarak kullanır. Bir nesne metin, ses, video, resim ve veri üzerinde gerçekleştirilecek işlemler için komutlar (algoritmalar) formundaki veriden oluşur. Bir hiyerarşik veya ağ veritabanı bir öğrenci hakkında sadece nümerik veya metin veri tutabilirken, bir nesne temelli veritabanı öğrencinin fotoğrafı, ses kaydı veya kısa video görüntüsü gibi çoklu ortam verisini de tutabilir. Buna ek olarak nesne, nesnelerin kendi kendilerine işlem yapabilmek için kullanacakları davranışları (metodlar) veya programları da saklayabilir (Hutchinson ve Sawyer, 2000, s.12.17)

Nesne temelli veritabanları henüz geleneksel işletmelerde kullanılmamaktadır. Bu veritabanlarının geliştirilmesi çok pahalı olduğu için birçok organizasyon milyarlarca bayt veriyi ilişkisel veritabanlarında saklamaktadırlar. Bu yüzden bu şirketler hali hazırda kullandıkları veritabanlarını nesne temelli veritabanı yönetim sistemi formatına dönüştürmenin risk ve maliyetlerini göze alamamaktadırlar. Buna karşın veritabanı uzmanları nesne temelli veritabanı yönetim sistemlerinin çoklu ortam yeteneklerinin büyük faydalar sunduğu bazı alanlarda geniş kullanım alanı bulacağını öngörmektedirler (Hutchinson ve Sawyer, 2000, s.12.17).



Şekil 2 17 Nesnelere Birbirleriyle İletişim Kurduran Nesne Tanımlayıcılar

Nesne temelli veritabanlarında her nesne tek ve özel bir nesne tanımlayıcıya sahiptir. Bunlar ilişkisel veritabanlarındaki birincil anahtarlara benzer Tanımlayıcı Şekil-2 17'de gösterildiği gibi nesnelerin diğer nesneler ve form ilişkileri ile etkileşimine olanak sağlar. Programcılar geleneksel varlık-ilişki diyagramlarındaki ilişkilere benzeyen nesneden nesneye ilişkileri uygulamak ve tanımlamak için C++ gibi nesne temelli dilleri kullanırlar Nesneler diğer nesneler ile 1:1, 1:M veya M:M ilişkilere sahip olabilirler (Shelly, vd , 2001, s.8 33)

2.5.6. Farklı Veritabanı Modellerinin Avantajları ve Dezavantajları

Veritabanı modelleri dosya sistemlerinin doğal zayıflıklarından yola çıkılarak geliştirilmiştir. Veriyi farklı dosyalarda tutmak yerine veritabanları veriyi tek bir veri deposunda tutarlar. Bu da veritabanı yönetim sistemlerine (VTYS-DBMS) veritabanı aktiviteleri üzerinde sıkı kontrol imkanı tanır. Üç temel veritabanı modeli (hiyerarşik, ağ ve ilişkisel) ticari başarı elde etmişlerdir. Her üçü de veritabanı yöneticilerine görevlerini daha etkin yapabilmelerine olanak tanıyan veritabanı yönetim sistemleri kullanırlar. Ancak veritabanı yönetim sistemleri faaliyetlerinin kapsamı ve bunların kullanıcı-dostu yazılımları çeşitli veritabanı sistemlerinde farklılık göstermektedir (Rob ve Coronel, 2002, s 45). Aşağıdaki tabloda yukarıda geçen beş farklı veritabanı modelinin avantaj ve dezavantajları özetlenmiştir.

Tablo 2.1. Çeşitli Veritabanı Modellerinin Avantajları ve Dezavantajları (Rob ve Coronel, 2002, s.48)

VERİTABANI MODELİ	VERİ BAĞIMSIZLIĞI	YAPISAL BAĞIMSIZLIK	AVANTAJLARI	DEZAVANTAJLARI
Hiyerarşik	Evet	Hayır	1 Veri paylaşımı yüksektir 2.Ebeveyn/Çocuk ilişkisi ile kavramsal sadelik artar 3. Ebeveyn/Çocuk ilişkisi veritabanı bütünlüğünü arttırır 4. 1:M sabit ilişkilerle etkindir	1. Yönsel sistem, karmaşık tasarım, uyarlama, uygulama geliştirme, kullanım ve yönetimi sonuç verir 2. Uygulama sınırlıdır (M:M veya çoklu ebeveyn ilişkileri yok) 3 VTYS'de veri tanımlama veya veri işlem dili yok 4.Standart eksikliği
Ağ	Evet	Hayır	1 Hiyerarşik modele eşit kavramsal basitlik 2 Daha fazla ilişki tipini kullanır M:M veya çoklu ebeveyn gibi 3 Sahip/Üye ilişkileri veritabanı bütünlüğü sağlar 4 Standartlara uygunluk 5.VTYS'de veri tanımlama ve veri işleme dilleri içerir	1 Sistem karmaşıklığı etkinliği sınırlandırır (hala yönsel sistem) 2. Yönsel sistem, karmaşık tasarım, uyarlama, uygulama geliştirme, kullanım ve yönetimi sonuç verir
İlişkisel	Evet	Evet	1.Çizelge görünümü kavramsal sadeliği önemli ölçüde arttırır,böylece veritabanı tasarımı, uygulaması, yönetimi ve kullanımı kolaylaşır 2. SQL'e dayalı ad hoc sorgu yeteneği 3 Güçlü veritabanı yönetim sistemi uygulama ve yönetim sadeliğini geliştirir	1. Sistemin kullanımını kolaylaştıran İlişkisel Veritabanı Yönetim Sistemi çok fazla donanım ve sistem yazılımı gerektirir 2 Kavramsal sistem sadeliği nispeten az eğitimli insanların iyi bir sistemi etkin kullanamamasına yol açar
Varlık-İlişki	Evet	Evet	1 Görsel modelleme olağanüstü kavramsal sadelik sağlar 2. Görsel sunum sistemi etkin bir iletişim aracı haline getirir 3. Baskın ilişkisel veritabanı modeli ile entegredir	1 Sınırlı kısıt sunumu 2. Sınırlı ilişki sunumu 3 Veri işlem dili yok 4. Bilgi içeriğinde kayıplar, çünkü özellikler genellikle kalabalık görüntü oluşmaması için çıkarılır
Nesne-Temelli	Evet	Evet	1. Semantik içerik ekler 2 Görsel sunum semantik içeriği de kapsamaktadır 3. Miras alma veritabanı bütünlüğünü arttırır	1. Standart eksikliği 2. Karmaşık yönsel sistem 3 Dik öğrenme eğrisi 4. Yüksek sistem yükü işlemleri yavaşlatır

2.6. İlişkisel Veritabanları ve Veritabanı Yönetim Sistemleri

İlişkisel veritabanı modeli, tasarımcının fiziksel depolama detayları yerine verinin mantıksal sunumuna odaklanmasını mümkün kılmaktadır. Bunun anlamı ilişkisel modelin veriyi fiziksel olarak değil mantıksal olarak görmeyi sağlamasıdır. Mantıksal görünümün uygulamadaki farkı veri depolamasına ilişkin daha yalın bir dosya kavramının akılda tutulmasıdır. Tablo kullanımı dosyalardan farklı olarak, yapısal bağımsızlık ve veri bağımsızlığı avantajına sahiptir. Büyük orandaki mantıksal sadelik daha kolay ve daha etkin veritabanı tasarımı sonucunu vermektedir (Rob ve Coronel, 2002, s 58)

Bir ilişkisel veritabanında veri, benzer satır ve sütun yapısı olan bir veya daha fazla tabloda tutulan kümelere ayrılmıştır. İlişkisel veritabanları farklı tablolarda saklanan ayrı veri birimlerini hızla alabilir ve bunları kullanıcıya veya uygulama programına sonuç olarak adlandırılan bütünleşik veri topluluğu şeklinde getirebilir (McGeever, 2000, s 70)

2.6.1. Veritabanı Yönetim Sistemi Özellikleri

Bir veritabanı yönetim sistemi Tablo 2.2'de de gösterilen bazı bileşenlere sahiptir:

Tablo 2.2. Veritabanı Yönetim Sisteminin Bazı Önemli Özellikleri (Hutchinson ve Sawyer, 2000, s.12.18)

Bileşen	Tanımı
Veri Sözlüğü	Verinin alanları ve dosyalarını tanımlar
Programlar	Veri girişlerini yaratarak, düzenleyerek ve görüntüleyerek veritabanlarının devamlılığını sağlar
Sorgulama Dili	Kullanıcıların veritabanını sorgulamasını ve seçilen kayıtların alınabilmesini sağlar
Rapor Üreteci	Uzman olmayan kişilerin kayıtlardan görüntülü veya basılı raporlar üretmesini sağlar
Erişim Güvenliği	Kullanıcı erişim haklarını tanımlar
Veri Kurtarma	Sistem hatasından sonra veritabanı içeriğinin kurtarılmasını sağlar

Veritabanı yönetim sistemi (VTYS) ile işlem yapmak bazı avantajlar sağlar. Yazılım zinciri içerisindeki veritabanı yönetim sisteminin konumu veri bağımsızlığı sağlar. Bir veritabanı yönetim sistemi ile veri sadece bir konumda saklanır ve birçok farklı sistem veya bölüm tarafından erişilebilir. Bu yapıda gereğinden fazla veri tutulmasını engeller. VTYS içerisindeki yazılım maskesi veri bütünlüğü sağlar. İyi eğitilmiş uzman bir personel grubu tarafından desteklenen merkezi bir veritabanı yönetim sisteminin kurulmasıyla ölçek ekonomisi sağlanabilir (Ronald ve Anjard, 1994, s.11)

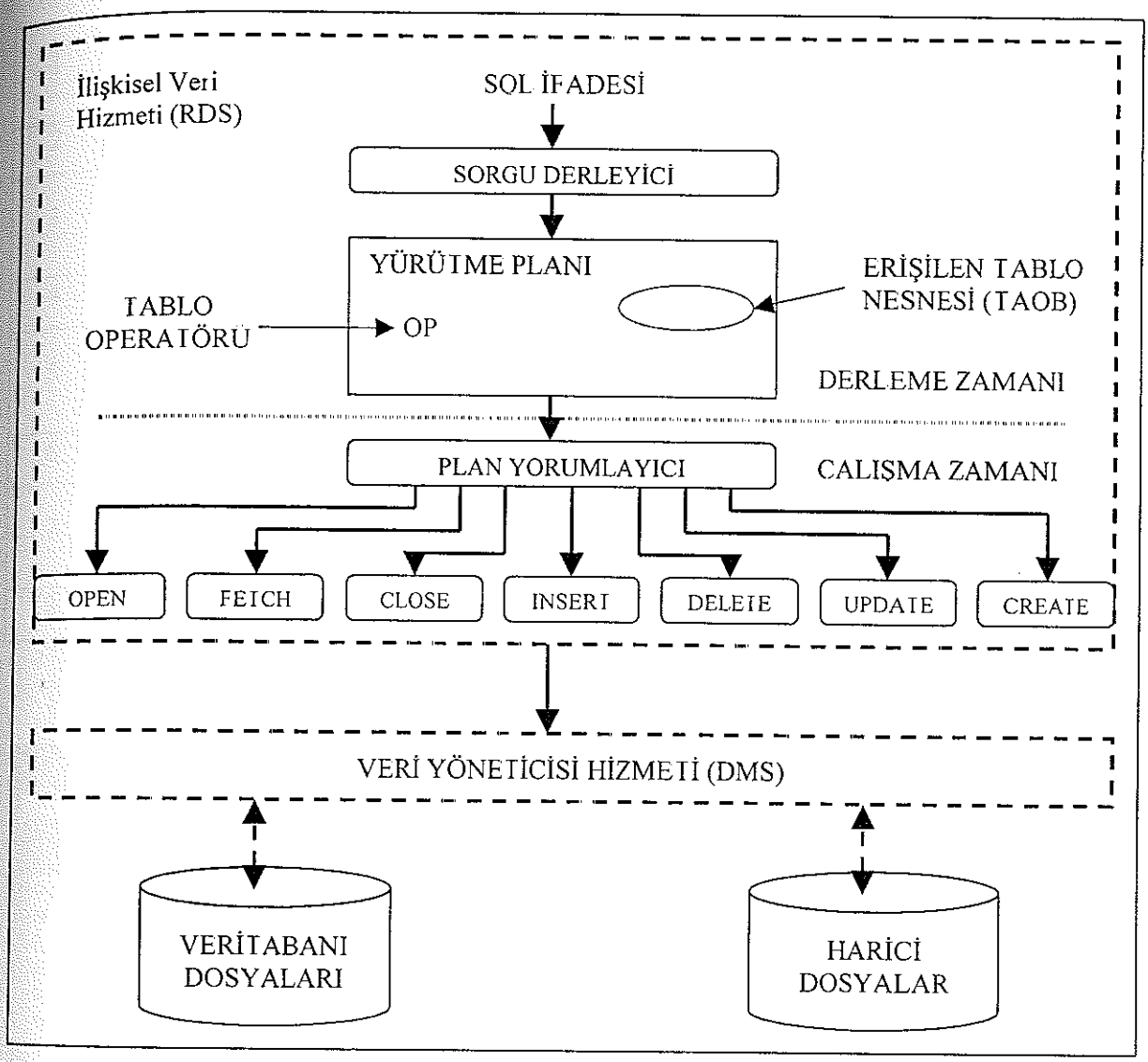
Tablo 2.3. Veritabanı Yönetim Sisteminin Avantajları ve Dezavantajları (Ronald ve Anjard, 1994, s 12)

Avantajları	Dezavantajları
• Veri bağımsızlığı • Planlanmayan gereksiz veri yok • Tutarlı veri • Veri güvenliği • Veri bütünlüğü • Ölçek ekonomisi	• Büyüklüğü çok fazla • Karmaşık • Yüksek maliyet • Artan donanım gereksinimleri • Büyük hata etkisi

2.6.2. İlişkisel Veritabanı Yönetim Sistemi Mimarisi

İlişkisel veritabanı yönetim sistemleri ileri veri sistemlerinin gelişmesiyle birlikte teknolojinin temel unsurlarından biri olmuştur (Bertino ve Martino, 1994, s.2). Tablolar ve tablolar üzerindeki işlemler ilişkisel modelin merkezidir ve geliştirildiğinden bu yana Yapısal Sorgulama Dili'nin (SQL) çekirdeğidir. Sorgular, tabloları girdi işlemi olarak kabul eden ve başka tabloları da çıktı olarak üreten işlemleri tanımlar. Bir ilişkisel veritabanı yönetim sistemi (RDBMS) motorunda (engine) sorgu değerlendirmesi kısıtlama (restriction), gösterim (projection), birleştirme (join) gibi tablo kayıtlarını işleyen ilişkisel operatörlere dayalıdır (Fuh, vd., 1998, s.539).

Tipik bir ilişkisel veritabanı yönetim sistemi motoru çalışma zamanında (run time) veritabanının mantıksal görünümü için bir ilişkisel veri hizmeti (relational data service – RDS) bileşeni ve veritabanının fiziksel (ham veri) görünümü için de bir veri yöneticisi hizmeti (data manager service – DMS) bileşeni içerir. Aşağıda Şekil-2.18'de bir ilişkisel veritabanı yönetim sisteminin ayrıntılı mimarisi görülmektedir (Fuh, vd., 1998, s.544).



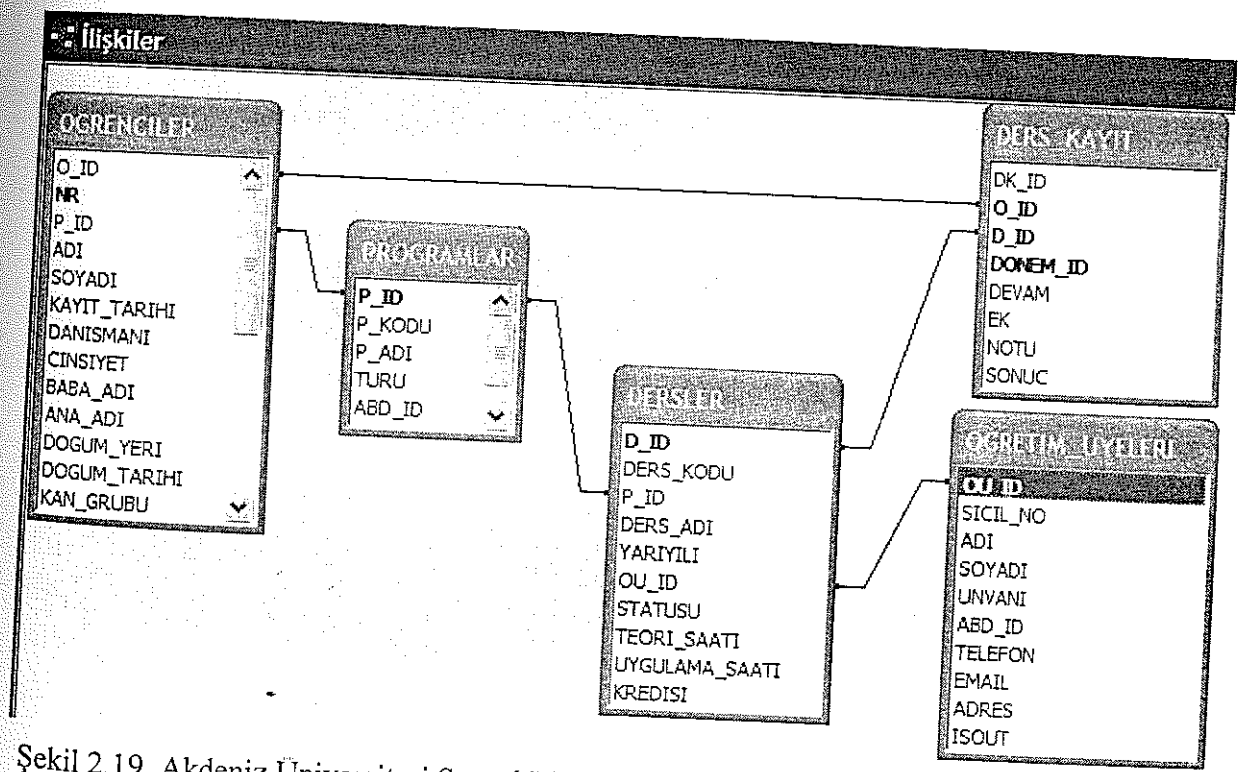
Şekil 2.18. İlişkisel Veritabanı Yönetim Sistemi Mimarisi (Fuh, vd., 1998, s.544)

2.6.3. İlişkisel Veritabanı Tabloları ve Özellikleri

İlişkisel veritabanlarının mantıksal görünümü mantıksal yapılara (tablo) dayalı veri ilişkilerinin yaratılması ile kolaylaşmaktadır. Bir tablo satır ve sütunlardan oluşan iki boyutlu bir yapı olarak algılanır. Tablonun kullanıcısı tablonun ilişkili varlıklar grubu yani bir varlık kümesi (entity set) içerdiğini de düşünebilir. Bu yüzden varlık kümesi ve tablo terimleri birbirinin yerine geçebilecek şekilde kullanılabilir (Rob ve Coronel, 2002, s 59). Tablo ayrıca ilişki olarak da adlandırılabilir, çünkü ilişkisel modelin yaratıcısı E. F. Codd ilişki terimini tablo ile eşanlamlı olarak kullanmıştır (Codd, 1970, s.377).

Tablo 2.4 Bir İlişkisel Tablonun Karakteristikleri (Rob ve Coronel, 2002, s.59)

1	Bir veritabanı tablosu satır ve sütunlardan oluşan iki boyutlu bir yapı olarak algılanır.
2	Her tablo satırı, varlık seti içerisinde tek bir varlık olayını temsil eder
3	Her tablo sütunu bir özelliği temsil eder ve her sütun farklı ve belirgin bir isme sahiptir.
4	Her satır/sütun kesişimi tek bir veri değerini temsil eder
5	Her tablo mutlaka her satırı özel olarak tanımlayan bir özelliğe veya özellikler kombinasyonuna (anahtar) sahip olmalıdır
6	Bir sütundaki bütün değerler mutlaka aynı veri formatına uymalıdır
7	Her sütun özellik domain'i (attribute domain) olarak bilinen belirli değerler aralığına sahip olmalıdır.
8	Satır ve sütunların sırası Veritabanı Yönetim Sistemi (DBMS) için önemsizdir.



Şekil 2.19 Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü Veritabanı Tablo İlişkileri Örneği

Şekil-2.19'da Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü Veritabanında yer alan bazı tabloların aralarındaki ilişkiler gösterilmiştir. Şekilde öğrenci tablosu, öğrencilerin kayıtlı olduğu programlar tablosu programlarda açılan dersler tablosu, derslere atanan öğretim üyeleri tablosu ve öğrencilerin dönem içerisinde hangi derslere kayıt olduğunun tutulduğu ders kayıt tablosu yer almaktadır. Burada OGRENCILER ve PROGRAMLAR arasında 1:M

ilişki vardır. Bu her öğrenci sadece bir programa kayıtlı olabilirken, bir programın birden fazla öğrenci kontenjanına sahip olduğu anlamına gelir. PROGRAMLAR ve DERSLER arasında, her ders ve o dersin kodu sadece bir programa özel olduğu ve bir programda birden fazla ders olduğu için 1:M ilişki vardır. Ayrıca DERSLER ve OGRETIM_UYELERI arasında 1:M, DERSLER ve DERS_KAYIT arasında 1:M ve OGRENCILER ve DERS_KAYIT arasında M:N ilişki vardır, öğrenciler birden fazla derse kayıt olabildiği gibi, bir ders için birden fazla öğrenci kayıt olabilmektedir.

İlişkisel veritabanı tablolarında mutlaka en az bir anahtar sütun bulunur. Bir anahtar diğer özellikleri saptamak için bir veya daha fazla özellik içerir. Örnek olarak bir sipariş numarası sipariş tarihi, müşteri adı, vb gibi bütün sipariş özelliklerini tanımlar. İlişkisel veritabanlarında birkaç farklı tür anahtar kullanılmaktadır. Bunlardan en önemlisi birincil anahtardır (primary key). Kullanılan diğer anahtar türleri ise; süper anahtarlar (superkeys), aday anahtarlar (candidate keys), ikincil anahtarlar (secondary keys) ve dış anahtarlardır (foreign keys) (Rob ve Coronel, 2002, s 62)

2.6.4. Veritabanı Tablolarının Normalizasyonu

İyi bir ilişkisel veritabanı yazılımına sahip olmak veri fazlalığı tehlikesinden uzak durmak için yeterli olmayabilir. Eğer veritabanı tabloları bir dosya sistemindeki dosyalar olarak düşünülürse, kullanılan ilişkisel veritabanı yönetim sistemi (RDBMS) kendisinin yüksek veri yönetimi yeteneklerini gösterme şansına sahip olamaz (Rob ve Coronel, 2002, s.176).

İlişkisel model veriyi tablolarda yapılandırmaktadır ve bu hemen bütün yapılandırmalarda veri işleme için esneklik tanımaktadır. İlişkisel model veri fazlalığını azaltmakta ve veri bütünlüğünü arttırmaktadır. Buna ek olarak ilişkisel veritabanları normalize edilmektedir. Normalizasyon, karmaşık veri gruplarından kararlı veri yapıları yaratma ve aşırı veriyi ayrı tablolara ayırarak bunların ilişkilerini yapılandırma sürecidir. Normalizasyonun altı aşaması vardır ve altıncı normal formda olan bir veritabanı olabilecek en mükemmel forma yakındır. Bir veritabanı ne kadar normalize edilmişse o kadar esnek ve daha geniş alanda sorgu ve işlemlere izin verir durumdadır (Tribunella, 2002, s.69-70).

Normalizasyon normal formlar olarak adlandırılan bir aşamalar serisi yoluyla çalışır. İlk üç aşama birinci normal form (1NF), ikinci normal form (2NF) ve üçüncü normal form (3NF)

olarak adlandırılır. Yapısal bakış açısından 2NF 1NF'dan daha iyidir ve 3NF da 2NF'dan daha iyidir. Birçok işletme veritabanı tasarımı, üçüncü normal formu (3NF) normalizasyon işleminde gidilmesi gereken en ileri aşama olarak amaçlamaktadır. Bazı özel uygulamalar dördüncü normal form (4NF) veya daha ileri aşamalara giden bir normalizasyon işlemini gerektirebilir. Bu uygulamalar genellikle istatistiksel araştırmaların gereksinimleri için tasarlanan programlar olabilir. Bu tür programlar birçok işletme faaliyetinin dışında olduğundan normalizasyon süreci genellikle üçüncü normal formdan ileriye götürülmez (Rob ve Coronel, 2002, s.176).

Normalizasyon çok önemli bir veritabanı tasarım aşaması olmasına rağmen, veritabanı tablolarının olabilecek en yüksek seviyede normalize edilmesi her zaman istenilen bir durum olmayabilir. Genellikle daha ileri bir normal form tanımlanan sonucun alınması için daha fazla birleştirme gerektirir. Bu da veritabanının son kullanıcı isteklerine daha yavaş cevap vermesine neden olur. Başarılı bir tasarım hızlı performans için mutlaka son kullanıcı isteklerini göz önüne almalıdır. Bu yüzden veritabanı tasarımcısından performans gereksinimlerinin karşılanması için bazı tasarım bölümlerini denormalize etmesi beklenebilir. Denormalizasyon işlemi var olan normal formu daha düşük seviye bir normal forma indirger (Rob ve Coronel, 2002, s 176)

2.7. Yapısal Sorgulama Dili (SQL)

Yapısal sorgulama dili (Structured Query Language - SQL) bir ilişkisel veritabanına bilgi girilmesi ve seçilen kayıtların geri getirilmesi için tasarlanmış bir programlama dilidir (McGeever, 2000, s.70). Sorgulama dilleri, kullanıcıların herhangi bir yordamsal dilde geleneksel programlama komutları yazmalarını gerektirmeyecek veritabanlarına bir arayüz sağlamak amacıyla geliştirilmiştir. Tipik olarak bir sorgu, bir cümle veya SELECT, DELETE ve MODIFY gibi basit kelimeleri kullanan İngilizce'ye yakın bir komut formundadır. Her biri kendi kelime haznesi ve prosedürleri olan birkaç farklı sorgulama dili vardır. Bunların en yaygın olanı SQL'dir. (Hutchinson ve Sawyer, 2000, s. 12.18)

İdeal olarak bir veritabanı dili, veritabanı ve tablo yapıları oluşturmayı, temel veri yönetimi işlerini (ekleme, silme ve değiştirme) gerçekleştirmeyi ve ham veriyi kullanışlı bilgiye dönüştürmek için tasarlanan karmaşık sorguları gerçekleştirmeyi sağlamalıdır. Buna ek olarak bazı temel fonksiyonları minimum kullanıcı çabasıyla yerine getirebilmeli ve dilin komut yapısı ve cümle diziminin öğrenilmesi kolay olmalıdır. Son olarak da bir veritabanı

yönetim sisteminden diğerine geçişte bazı temel standartları sağlamalıdır (Rob ve Coronel, 2002, s.210).

SQL ideal veritabanı dili gereksinimlerini iyi bir şekilde karşılamaktadır ve SQL aşağıdaki iki geniş kategoriye uymaktadır (Rob ve Coronel, 2002, s.210):

- SQL bir veri tanımlama dilidir. Veritabanı tablo yapıları oluşturmak ve veritabanına erişim haklarını tanımlamak için gereken komutları içermektedir.
- SQL bir veri işleme (manipulation) dilidir. Veritabanı tabloları içerisinde verinin yerleştirilmesi (insert), güncellenmesi (update), silinmesi (delete) ve bulunup getirilmesi (retrieve) için gereken komutları içermektedir.

SQL nispeten öğrenilmesi kolay bir dildir. Komut seti olarak sayısı 100'ün altında temel kelimelere sahiptir. Ayrıca SQL yordamsal olmayan bir dildir. Bunun anlamı kullanıcı yalnızca neyin yapılacağının komutunu yazdığı zaman bunun nasıl yapılacağını bilmesine gerek olmamasıdır (Rob ve Coronel, 2002, s 210)

Başlangıçta IBM tarafından kendi ana bilgisayar sistemleri için geliştirilen SQL, Amerikan Ulusal Standartlar Enstitüsü (American National Standards Institute – ANSI) tarafından ilişkisel veritabanı sorgu standardı olarak tanınmıştır. SQL'in son standardı SQL:1999 ilişkisel modelin nesne-temelli uzanımları için de destek sağlamaktadır (Dietrich, 2001, s 65). Yazılım tasarımcıları bu standardı kullanıcıların veri aramalarını yapabilecekleri veritabanı sorgu arayüzleri tasarlamak için kullanmaktadırlar. Bu nedenle SQL, içerisinde ORACLE, SYBASE, DBASE, Paradox ve Microsoft Access gibi birçok ticari veritabanı yönetim sistemi ürünü tarafından kullanılan bir veri erişim dilidir (Hutchinson ve Sawyer, 2000, s.12.18).

Günümüzde SQL, veritabanı içeren bütün uygulamaları başarıyla sonuçlandırmak için çok yaygın ve temel bir geliştirme aracıdır. Bu yüzden veritabanı içeren uygulamaları geliştirmek için kullanılan bir görsel programlama aracının nesne-temelli bir grafik arayüzü olsa bile programlama bittiği zaman sistem temelini oluşturan tüm veritabanı çağrılarını ve komutlarını SQL'e dönüştürür. Bu özellikle çok katlı istemci/sunucu uygulamalarında ön-uç (front-end) ve arka-uç (back-end) sistemlerin entegrasyonunu kolaylaştırmaktadır (McGeever, 2000, s.70).

2.7.1. SQL Nasıl Çalışır?

SQL'in gücü içerisinde uygulama geliştiricilerin, veritabanı yöneticilerinin, yöneticilerin ve son kullanıcıların olduğu bütün kullanıcı türlerine faydalar sağlıyor olmasıdır. Teknik olarak SQL bir alt dildir. SQL'in amacı ORACLE veya SQL Server gibi bir ilişkisel veritabanına arayüz sağlamaktır ve bütün SQL ifadeleri veritabanına yönelik komutlardır. Bu nedenle SQL, C gibi genel amaçlı programlama dillerinden ayrılmaktadır. SQL'in bazı özellikleri şunlardır (Oracle9i SQL Reference Manual, 2001, s.1.2):

- Veri kümelerini tek birimler olarak değil gruplar olarak işler.
- Veri üzerinde otomatik yönelim sağlar
- Tek başlarına karmaşık ve güçlü olan dolayısıyla tek başlarına yeterli olan sözdizimleri kullanır

SQL kullanıcının veri ile mantıksal seviyede çalışmasını sağlar. Kullanıcı veri üzerinde işlem yapmak istediğinde sadece uygulama detayları ile ilgilenmek durumundadır. Örnek olarak bir tablodan satır kümeleri almak için bu satırları filtreleyecek koşullar tanımlanmalıdır. Tanımlanan koşulları sağlayan bütün satırlar tek bir adımda alınır ve bir birim olarak kullanıcıya, başka bir SQL ifadesine veya bir uygulama programına gönderilebilir. Kullanıcı satırlarla tek tek ilgilenmek durumunda veya bunların fiziksel olarak nasıl saklandığı veya alındığını bilmek durumunda değildir. SQL çeşitli görevler için ifadeler içermektedir ve tek bir tutarlı dilde aşağıdaki görevleri birleştirmektedir (Oracle9i SQL Reference Manual, 2001, s 1.3):

- Verinin sorgulanması
- Bir tabloda satırların eklenmesi, güncellenmesi ve silinmesi
- Nesnelerin yaratılması, konumlandırılması, değiştirilmesi ve silinmesi
- Veritabanına ve veritabanı nesnelerine erişimin kontrol edilmesi
- Veritabanı tutarlılığının ve bütünlüğünün garanti edilmesi

2.7.2. SQL Kullanılarak Veritabanı Sorguları Yaratılması

Bir sorgu veritabanında tutulan bir veri tarafından cevaplanabilecek bir sorudur. Örnek olarak "Öğrenci Mehmet Demir Sosyal Bilimler Enstitüsü'ne ne zaman kayıt olmuştur?" sorusunun yanıtını alabiliriz. SQL kullanılarak bir veritabanı tablosundan veri alabiliriz veya

dış anahtar (foreign key) kullanarak çoklu veritabanı tablolarını birbiriyle birleştirip (join) farklı tablolardan ilişkili verileri alabiliriz (Morrison ve Morrison, 2000, s.52).

Yazılım programları yapısal sorgulama diline üç yolla erişmektedirler (Hoske, 2002, s.37):

- Gömülmüş (Embedded) SQL, SQL ifadelerinin C veya COBOL gibi bir sunucu (host) dilde gömülü olmasıdır
- SQL modülleri, SQL ifadelerinin bir veritabanı yönetim sisteminde (VTYS-DBMS) derlenmesi (compile) ve bir sunucu dilden çağırılmasıdır
- Çağrı seviyesi arayüzü (Call level interface – CLI), SQL ifadelerinin veritabanı yönetim sistemine (DBMS) geçişi ve DBMS'ten sonuçların getirilmesi için çağırılan fonksiyonları içerir.

Aşağıda Şekil-2.20'de basit bir SQL sorgusu örneği görülmektedir. Bu sorgu şekilde görülen OGRENCILER tablosundan P_ID alanı 2 olan İşletme Yüksek Lisans Programı'na kayıtlı öğrencileri seçerek öğrenci numaraları (NR), isimlerini (ADI), soyadlarını (SOYADI) ve kayıt tarihlerini (KAYIT_TARIHI) listelemektedir.

OGRENCILER : Tablo

O_ID	NR	P_ID	ADI	SOYADI	KAYIT_TARIHI	DANISMANI
34	008504030	2	Recep	Irmak	10.09.2000	1
36	008504032	2	Mehmet	Demir	10.09.2000	7
38	018502002	5	Çağla	Aktaş	15.09.2001	42
37	018504033	2	Özlem	Şahin	14.09.2001	12
35	038506070	1	Hasan	Yılmaz	10.06.2003	34

PROGRAMLAR : Tablo

P_ID	P_KODU	P_ADI	TURU
1	804025	İşletme Bilimsel Hazırlık	BH
2	804025	İşletme Yüksek Lisans	YL
3	804026	İşletme Doktora	D
4	8040252	İşletme Tezsiz Yüksek Lisans	TYL
5	804015	İktisat Yüksek Lisans	YL
6	804045	Gıda Ekonomisi ve İşletmeciliği Yüksek Lisans	YL
7	804035	Kamu Yönetimi Yüksek Lisans	YL

```
SELECT    NR, ADI, SOYADI, KAYIT_TARIHI
FROM      OGRENCILER
WHERE     P_ID = 2
```

Şekil 2.20. Basit Bir SQL Sorgusu Örneği

Yukarıdaki sorgu sonucunda aşağıdaki gibi bir sonuç dönecektir:

008504030	Recep	Irmak	10.09.2000
008504032	Mehmet	Demir	10.09.2000
018504033	Özlem	Şahin	14.09.2001

Tablo 2.5. Bazı SQL Terimleri ve Anlamları (Hayes ve Hunton, 2001, s.40)

SQL İfadesi	Yan İfade	Açıklama
SELECT		Sorguda yer alacak sütunları listeler
	FROM	Sütunların seçileceği tabloları tanımlar
	WHERE	Satır seçimi için koşulları içerir
	AS	Sorguda yaratılan yeni veriyi tanımlar
	ORDER BY	Bir veya daha fazla sütun değerlerine göre satırları azalan veya artan sırayla sınıflandırır
	GROUP BY	Ortak sütun değerlerine göre satırları gruplar
INNER JOIN		Ortak sütunlardaki değerlerin eşitliğine göre iki veya daha fazla tabloyu bir dinamik kümede (sanal tablo) birleştirir
	ON	Bir dinamik kümede iki veya daha fazla tabloyu birleştirmek için kullanılacak ortak sütunları tanımlar

Tablo 2.5'de bazı SQL terimleri ve açıklamaları gösterilmiştir. Bir SQL sorgusunun bir veritabanı tablosundan bütün satırları getiren temel formatı aşağıdaki şekildedir:

```
SELECT    [sütun1, sütun2, .... ]
FROM      [tablo adı]
```

SELECT ifadesi yapılan sorgu sonucu görüntülenmek istenen sütunları listeler. Her sütun kendisinden sonraki ile bir virgül ile ayrılır (Morrison ve Morrison, 2000, s.53). Genellikle bir tablodan belirli sütunlar getirilmek istenir. Bunun için sorguya aşağıdaki genel formatta bir arama koşulu eklenmelidir (Morrison ve Morrison, 2000, s.57)

```
SELECT    [sütun1, sütun2, .... ]
FROM      [tablo adı]
WHERE     [arama koşulu]
```

Arama koşulunun da genel kullanım biçimi aşağıdaki şekildedir:

WHERE [ifade] [karşılaştırma operatörü] [ifade]

Her arama koşulu mutlaka TRUE veya FALSE olarak değerlendirilebilmelidir. Tablo 2.6'da SQL ifadelerinde yaygın olarak kullanılan karşılaştırma operatörleri listelenmektedir (Morrison ve Morrison, 2000, s.57).

Tablo 2.6 Sorgu Koşulu Karşılaştırma Operatörleri

Operatör	Açıklama
=	Eşittir
>	Büyüktür
<	Küçüktür
>=	Büyüktür veya eşittir
<=	Küçüktür veya eşittir
<>	Eşit değildir veya farklıdır

2.7.3. Sorgu Sonuçlarının Sıralanması

Veritabanı sorgu sonuçları görüntülendiği zaman bunlar veritabanı tablosunda olduğu sıra ile aynı sırada görüntülenmektedir. Genellikle bu verinin görüntülenmesi için çok tercih edilen yol değildir (Jones, 2001, s.6).

SQL sorgusu çıktıları SELECT cümlesinin sonuna ORDER BY yan ifadesi eklenerek ve SQL'in veriyi sıralamak için esas alacağı sütun olan sıralama anahtarı tanımlanarak belirli bir sırada görüntülenebilir. ORDER BY yan ifadesinin genel kullanımı ORDER BY [sütun adı] şeklindedir. Varsayılan olarak veri kayıtları artan sırayla sıralanırlar. Verilerin artan sıraya göre dizilmeleri net olarak belirtmek istenirse ORDER BY ifadesinin sonuna ASC komutu yazılır. Eğer verinin azalan sırayla dizilmesi istenirse DESC komutu yazılır (Morrison ve Morrison, 2000, s.62).

DERSLER : Tablo

D_ID	DERS_KODU	P_ID	DERS_ADI	YARIYILI	OU_ID
1	80402501	1	İşletme Yönetimi	Güz	9
2	80402505	1	İstatistik	Güz	8
3	80402507	1	Genel Muhasebe	Güz	10
4	80402509	1	Genel Ekonomi	Güz	1
5	80402511	1	Yöneylem Araştırması	Güz	1
6	80402502	1	Davranış Bilimleri	Bahar	5
7	80402504	1	Pazarlama İlkeleri	Bahar	6
8	80402506	1	Uretim Yönetimi	Bahar	3
9	80402514	1	Finansal Yönetim	Bahar	7
10	80402512	1	İş Hukuku	Bahar	1

```
SELECT    DERS_KODU, DERS_ADI, YARIYILI
FROM      DERSLER
WHERE     OU_ID = 1
ORDER BY  DERS_KODU DESC;
```

Şekil 2.21. Sıralama Sorgusu Örneği

Şekil-2.21'de görülen sorguda DERSLER tablosunda yer alan derslerden Öğretim Üyesi ID'lerine göre (OU_ID) seçim yapılıp ders kodlarına göre (DERS_KODU) azalan sırayla görüntülenecektir. Bu sorgunun koşturulması sonucu aşağıdaki veri sırası görüntülenecektir:

80402512	İş Hukuku	Bahar
80402511	Yöneylem Araştırması	Güz
80402509	Genel Ekonomi	Güz

2.7.4. SQL Kullanarak Veri Eklenmesi, Güncellenmesi ve Silinmesi

Veritabanı tablolarına veri eklenmesi, verinin güncellenmesi ve silinmesi işlemlerini yapan komutlar faaliyet sorguları (action queries) yaratırlar. Bu sorgular veritabanında saklanan veriyi değiştirirler. Faaliyet sorgularını kullanarak veri kayıtlarının nasıl yönetileceğinin bilinmesi önemlidir (Morrison ve Morrison, 2000, s.77)

SQL'in INSERT komutu iki yolla kullanılabilir; bir kayıtta bulunan bütün alanlara bir işlemde veri eklemek için veya bazı seçilmiş alanlara veri eklemek için. INSERT komutunun temel formatı bir kayıttaki tüm alanlara veri eklemek için kullanılanıdır

```
INSERT INTO    [tablo adı]
VALUES         ([sütun1 değeri, sütun2 değeri, ...]);
```

Sadece bir INSERT komutunu kullanarak aynı anda sadece bir tabloya kayıtlar girilebilir. Bir kayıtın tüm alanları için değer girilirken, INSERT komutunun VALUES yan ifadesi mutlaka tüm alanlar için değer içermelidir. Eğer alanın değeri bilinmiyor veya tanımlı değilse değer yerine NULL ifadesi kullanılmalıdır. Sütun değerleri tablo yaratılırken belirlenen sıralama ile aynı sırada olmalıdır (Morrison ve Morrison, 2000, s 77)

INSERT komutu aynı zamanda bir kayıtta sadece seçilen alanlara değer girilmesi için de yapılandırılabilir. Seçilen tablo alanına veri girişi için kullanılan temel INSERT komutu formatı aşağıdaki gibidir:

```
INSERT INTO     [tablo adı] ([sütun1, sütun2, . . .])
VALUES          ([sütun1 değeri, sütun2 değeri, . . .]);
```

Sütun adları herhangi bir sıra içerisinde verilebilir, ancak sütun değerleri INSERT INTO ifadesinde yer alan sütun adları sıralaması ile aynı sırada olmalıdır (Morrison ve Morrison, 2000, s.77).

Önemli bir veri koruma işlemi var olan veri kayıtlarının güncellenmesidir. Bir veritabanı kaydının SQL ile güncellenmesinde UPDATE komutu kullanılır. UPDATE ifadesinin genel formatı aşağıdaki gibidir

```
UPDATE      [tablo adı]
SET         [sütun1] = [yeni veri değeri_1], [sütun2] = [yeni veri değeri_2], . . .
WHERE       [arama koşulu];
```

Bir UPDATE komutu kullanılarak veritabanında aynı anda sadece bir tabloda veriler güncellenebilir. Ancak aynı tabloda sadece bir UPDATE komutu kullanarak WHERE ifadesinde yer alan aramą koşulunu sağlayan birçok kayıt ve bu kayıtlardaki birçok alan güncellenebilir (Morrison ve Morrison, 2000, s 81).

Diğer bir veri koruma işlemi de kayıtların silinmesidir, bu işlem için SQL'in DELETE komutu kullanılır. Sadece bir DELETE komutu kullanarak sadece bir tablodan bir veya daha fazla kayıt silinebilir. DELETE komutunun genel biçimi aşağıdaki gibidir.

```
DELETE FROM     [tablo adı]
WHERE           [arama koşulu];
```

Bir tablodan kayıt silinmesi işleminde mutlaka WHERE ifadesinden sonra bir arama koşulu kullanılmalıdır. Arama koşulunun atlanması sonucunda bütün tablo kayıtları silinecektir. Ayrıca sadece doğru kaydın silinebilmesi için arama koşulunun doğru belirlenmesi gereklidir (Morrison ve Morrison, 2000, s.83).

ÜÇÜNCÜ BÖLÜM

AKDENİZ ÜNİVERSİTESİ SOSYAL BİLİMLER ENSTİTÜSÜ ÖĞRENCİ İŞLERİ OTOMASYONU UYGULAMASI

2.8. Sosyal Bilimler Enstitüsü'nün Yapısı

Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü (S.B.E.) 1992 yılında kurulmuştur İçinde bulunduğumuz 2003-2004 eğitim öğretim yılı itibariyle Enstitü Müdürü ve 3 idari personel ile lisansüstü eğitim hizmetlerini vermektedir. Enstitünün kuruluşundan bu yana anabilim dalı, program, öğrenci, ders ve öğretim üyesi sayısı giderek artmaktadır S.B.E. 2003-2004 eğitim öğretim yılında 17 Anabilim Dalında, yüksek lisans seviyesinde 26 ve doktora düzeyinde 8 programla eğitim öğretim faaliyetlerine devam etmektedir. Enstitüye bağlı Anabilim Dalları ve bunların bünyesinde yer alan programlar aşağıda Tablo-3.1'de verilmiştir

Tablo 3.1. Akdeniz Üniversitesi S.B.E. Anabilim Dalları ve Bunlara Bağlı Programlar

Anabilim Dalı	Bilimsel Hazırlık	Yüksek Lisans	Tezsiz Yüksek Lisans	Doktora veya Sanatta Yeterlilik
Arkeoloji	Var	Var		Var
Tarih		Var		Var
Eskiçağ Dilleri ve Kültürleri		Var		Var
Turizm İşletmeciliği ve Otelcilik		Var		
İktisat		Var - İktisat - Gıda Ekonomisi ve İşletmeciliği		Var
İşletme	Var	Var	Var	Var
Kamu Yönetimi		Var		Var
Halkla İlişkiler ve Tanıtım		Var	Var	
Eğitim Yönetimi ve Denetimi	Var	Var	Var	
Orta Öğretim Sosyal Alanlar Eğitimi			Var - Tarih Öğr - Türk D. ve Ed. Öğr.	
Sosyoloji		Var		
Felsefe		Var		
Özel Hukuk		Var		
Kamu Hukuku		Var		
Spor Yöneticiliği		Var		
Resim		Var		Var
Grafik		Var		Var

2.9. Otomasyona Geçme İhtiyacı

Yıllar itibariyle enstitü bünyesindeki anabilim dalı, program, öğrenci ve ders sayısının artmasına rağmen bu artışa orantılı olarak personel sayısının artmaması nedeniyle artan iş yükü otomasyona geçme ihtiyacını gündeme getirmiştir. Gelişen bilişim teknolojilerinin ve özellikle donanım ihtiyaçlarının ucuzlaması nedeniyle Sosyal Bilimler Enstitüsü öğrenci işlerinin veritabanı yönetim sistemi kullanılarak otomasyona geçirilmesi olanaklı hale gelmiştir. Aşağıda 2003-2004 eğitim öğretim yılı rakamlarıyla enstitü bünyesinde yer alan öğrenci, öğretim üyesi, program, ders ve idari personel sayıları verilmiştir.

Tablo 3.2 S.B.E. Program, Ders, Öğrenci, Öğretim Üyesi ve İdari Personel Sayıları

Sosyal Bilimler Enstitüsü	Yüksek Lisans	Doktora	Toplam
Program Sayısı	26	8	34
Ders Sayısı	365	130	495*
Öğrenci Sayısı	397	45	442
Öğretim Üyesi Sayısı	114		114
İdari Personel Sayısı	3		3

*Ders sayılarına özel konular ve danışmanlıklar dahil değildir.

Yukarıdaki rakamlara baktığımızda Sosyal Bilimler Enstitüsünde 34 yüksek lisans ve doktora programında toplam 442 öğrencinin lisansüstü eğitimini sürdürdüğü görülmektedir. Bu eğitim faaliyetlerinde enstitü bünyesinden 114 öğretim üyesi görev almakta ve yıl boyunca özel konular ve danışmanlıklar haricinde 495 dersin açılma olasılığı bulunmaktadır

Bu hizmetler için sadece 3 idari personelin görevli olması öğrenci işleri süreçlerinin geleneksel yöntemlerle etkin olarak gerçekleştirilmesinin zorluğunu göstermektedir Bu nedenle S.B.E. bünyesinde veritabanı yönetim sisteminin geliştirilmesi ve uygulanmasına ihtiyaç duyulduğunu söyleyebiliriz.

2.10. Sistem Veritabanı

Sosyal Bilimler Enstitüsü Veritabanı Yönetim Sistemi'nde maliyetler ve Enstitü ihtiyaçları göz önüne alınarak ilişkisel veritabanı Microsoft® Access 2002, sürüm 10.0 kullanılmıştır. Tablo-3.3'de sistemin X-kartı ile veritabanında bulunan tablolar, tablolardaki alanlar ve ilişkili alanlar verilmiştir. Tabloda koyu **X** işareti ile belirtilen alanlar tablolar arasındaki ilişkileri oluşturan anahtar alanları belirtmektedir.

Tablo 3.3. Sistemin X-Kartı

Tablo Adı / Alan Adı	ABD	AKTIF_DERSLER	ARGOR	ASKERLIK	DERS_KAYIT	DERSLER	DONEMLER	ESDEGER_DERSLER	KAYIT_DONDURMA	KULLANICILAR	OGRENCI_TEZ	OGRENCILER	OGRETIM_UYELERI	PROGRAMLAR	YKK
ABD_ID	X		X										X	X	
ABD_ADI	X														
P_ID						X						X		X	
P_KODU														X	
P_ADI														X	
TURU														X	
OU_ID		X				X							X		
SICIL_NO			X										X		
ADI												X	X		
SOYADI												X	X		
UNVANI													X		
TELEFON												X	X		
EMAIL												X	X		
ADRES												X	X		
ISOUT													X		
O_ID		X	X	X	X				X		X	X			
NR												X			
KAYIT_TARIHI												X			
DANISMANI												X			
CINSIYET												X			
BABA_ADI												X			
ANA_ADI												X			
DOGUM_YERI												X			
DOGUM_TARIHI												X			
KAN_GRUBU												X			
IL												X			
ILCE												X			
MAHALLE_KOY												X			
CILT_NO												X			
AILE_NO												X			
SIRA_NO												X			
BIT_UNI												X			
BIT_BOL												X			
BASLAMA_DONEMI												X			
MEZUNIYET_TARIHI												X			
AG_ID			X												
SICIL_NO			X										X		
ATAMA_TARIHI			X												
DONEM_ID		X			X		X		X						
YILI							X								
YARIYILI						X	X								
ISAKTIF							X								

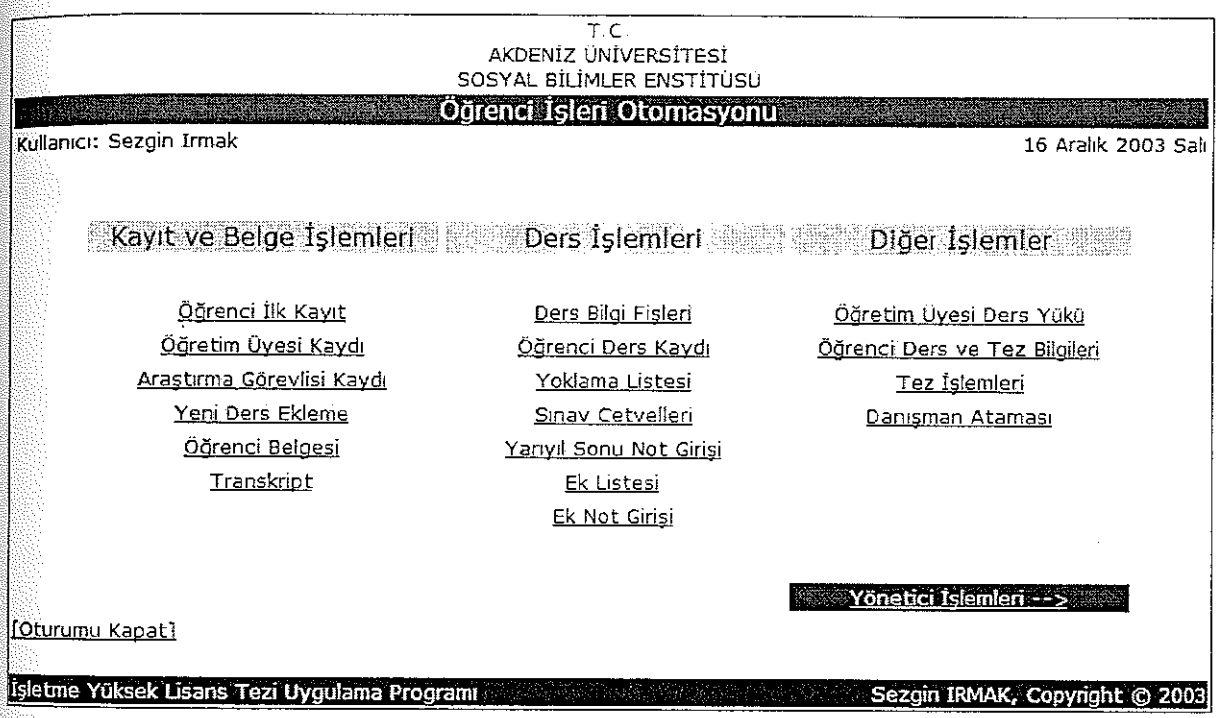
Alan Adı \ Tablo Adı	ABD	AKTIF_DERSLER	ARGOR	ASKERLIK	DERS_KAYIT	DERSLER	DONEMLER	ESDEGER_DERSLER	KAYIT_DONDURMA	KULLANICILAR	OGRENCI_TEZ	OGRENCILER	OGRETIM_UYELERI	PROGRAMLAR	YKK
USER_ID										X					
KULLANICI										X					
SIFRE										X					
ADI_SOYADI										X					
ISADMIN										X					
ACIKLAMA										X					
BLOCK										X					
D_ID		X			X	X		X							
DERS_KODU						X									
DERS_ADI						X									
STATUSU						X									
TEORI_SAATI						X									
UYGULAMA_SAATI						X									
KREDISI						X									
DK_ID					X										
DEVAM					X										
EK					X										
NOTU					X										
SONUC					X										
AD_ID		X													
OGRENCI_SAYISI		X													
ASKERLIK_SUBESI				X											
TECIL_TARIHI				X											
OT_ID											X				
TEZ_DURUMU											X				
YKK_ID								X	X						X
YKK_NO															X
KONUSU															X
ES_ID								X							
ES_D_ID								X							
KD_ID									X						

2.11. Otomasyon Programının Yapısı

S.B.E. Öğrenci İşleri Otomasyonu programı Enstitü Veritabanı üzerinde çalışan son kullanıcının veritabanında ihtiyaç duyulan veri işlemlerini yapmasını sağlayan ve temel olarak Aktif Sunucu Sayfaları (Active Server Pages-ASP) diliyle yazılmış bir programdır. Program web tabanlı olup Microsoft Internet Information Services (IIS) sunucusu altında çalışmaktadır. Program tasarımında ayrıca veri tanımlama ve veri işleme dili olarak SQL

yapısal sorgulama dili, formlar ve kullanıcı arayüzleri için HTML, VB Script ve Java Script kullanılmıştır. Program geliştirme ortamı olarak Microsoft Visual InterDev 6.0 kullanılmıştır.

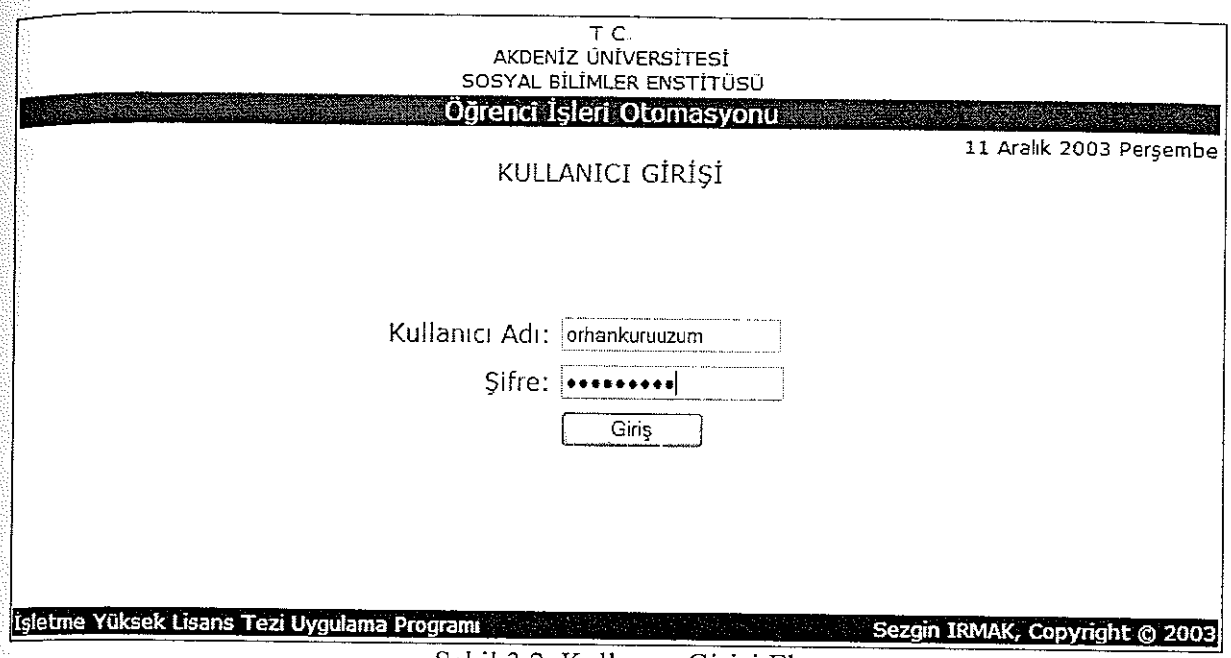
Aşağıda programın ana ekranı (sbemain.asp) görülmektedir. Programda oturum açabilmek için öncelikle sistemde kullanıcı olarak tanımlı olmak gereklidir. Ana ekranda otomasyon sisteminin yazılımları fonksiyonlarına göre gruplandırılmıştır.



Şekil 3.1. Akdeniz Üniversitesi S.B.E. Öğrenci İşleri Otomasyonu Ana Ekranı

2.12. Programın Başlatılması ve Erişim Kontrolü

Program, Sosyal Bilimler Enstitüsü sunucusuna bağlanarak wwwroot klasörü altında yer alan login.asp dosyasının çalıştırılmasıyla başlamaktadır. Login.asp dosyası Şekil-3.2'de görülen kullanıcı girişi ekranında kullanıcıya sistemde tanımlı kullanıcı adını ve şifresini sormaktadır



Şekil 3.2. Kullanıcı Girişi Ekranı

Burada login.asp dosyası kullanıcı adı ve şifresini sorduktan sonra girilen veriyi log_control.asp dosyasına gönderir. Log_control.asp dosyası veritabanında KULLANICILAR tablosundan bu kullanıcının olup olmadığını kontrol eder, eğer varsa şifresinin doğru girilip girilmediğini kontrol eder. Eğer kullanıcı tanımlı değilse ASP, response nesnesinin end deyimi (response.end) ile programı durdurur. Aşağıda login kontrolünün nasıl yapıldığının örneği verilmiştir.

```
set conn=Server.CreateObject("ADODB.Connection")
connstr="Provider=Microsoft.Jet.OLEDB.4.0;DATA SOURCE=sbe.mdb"
conn.Open connstr
        KULLANICIA=Request.Form("KULLANICI_ADI")
        SIFREA=Request.Form("SIFRE_ADI")
set rs=server.CreateObject("ADODB.Recordset")
rsSQL="SELECT USER_ID, SIFRE, ADI_SOYADI "&_
        FROM KULLANICILAR "&_
        WHERE KULLANICI='"&KULLANICIA&"'"
rs.Open rsSQL,conn

if rs.EOF then
    Response.Write("Kullanıcı sistemde kayıtlı değil ! ")
    Response.End
else
        if rs("SIFRE")=SIFREA then
                Session("USER")=rs("USER_ID")
                Response.Redirect("sbemain.asp")
        else
                Response.Redirect("login.asp?control=hatalisifre")
        end if
end if
rs.Close
conn.Close
```

Yukarıda görülen program kodları, form nesnesinden gelen veriyi veritabanındaki KULLANICILAR tablosundaki veri ile karşılaştırmakta ve doğru ise sbemain.asp dosyasına, hatalı şifre girilmişse control=hatalisifre querystring'i ile birlikte login.asp dosyasına response.redirect ile yönlendirmektedir

2.13. Kayıt ve Belge İşlemleri

Otomasyon sisteminde Kayıt ve Belge İşlemleri başlığı altında altı fonksiyon bulunmaktadır. Bunlar Enstitüdeki bir programa yeni öğrenci kaydı, öğrenci belgesi hazırlanması, transkript hazırlanması, yeni öğretim üyesi bilgilerinin girilmesi, yeni atanan araştırma görevlisi bilgilerinin girilmesi ve enstitü programlarına yeni ders eklenmesi olarak gruplandırılmışlardır.

2.13.1. Yeni Öğrenci Kaydı

S.B.E.'deki bir enstitü programa öğrenci alınması için başvuran kişinin açılan sınavı kazanmış olması, bankaya ilk harcını yatırması, banka dekontunu ve enstitü tarafından istenilen diğer belgeleri getirmesi gerekmektedir. Bu işlemler sonucunda öğrenci işleri otomasyonunda öğrenci kaydı için aşağıdaki form kullanılarak kayıt süreci başlayacaktır

T.C.
AKDENİZ ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ

Öğrenci İşleri Otomasyonu

11 Aralık 2003 Perşembe

ÖĞRENCİ KAYIT FORMU

Öğrenci Numarası	028604101	Programı	İşletme Doktora
Adı	Emre	Soyadı	İPEKÇİ ÇETİN
Baba Adı	İbrahim Lütfü	Ana Adı	Günnur
Doğum Yeri	Lübeck	Doğum Tarihi	24.05.1977
N K O İl	Sivas	İlçe	Gemerek
Mahalle-Köy	Çepni Ataybey	Cilt No	0012
Aile Sıra No	00131	Sıra No	0016
Cinsiyeti	○ Erkek ◉ Kız	Kan Grubu	A Rh+
		Kayıt Tarihi	30.09.2002
Bitirdiği Üniversite	Akdeniz Üniversitesi	Bitirdiği Bölüm	Matematik
Askerlik Şubesi		Adresi	131 Sk. No:9/1 Antalya
Telefonu	02423101839	E-Mail	ecetin@akdeniz.edu.tr

Öğrenciyi Kaydet

S.B.E. Ana Safya

Copyright © 2003

Şekil 2.3. Öğrenci Kayıt Formu

Öğrenci bilgileri kayıt formuna girildikten sonra form dosyası ogr.asp bu bilgileri ogr_db.asp dosyasına gönderir. Ogr_db.asp, VB AddNew metodunu kullanarak bilgileri veritabanında OGRENCILER tablosuna yeni bir kayıt ekleyerek kaydeder. Bu bilgiler iki farklı yolla veritabanına kaydedilebilir: SQL'in INSERT INTO komutunu kullanarak, ve Visual Basic AddNew metodunu kullanarak. Burada aşağıda örneği verilen AddNew metodu kullanılmıştır.

```
set rsOgr=Server.CreateObject("ADODB.Recordset")
SQLstrOgr="SELECT top 1 * FROM OGRENCILER"
rsOgr.Open SQLstrOgr, conn,1,3

rsOgr.AddNew
        rsOgr("NR")=request.Form("OGRNO")
        rsOgr("P_ID")=request.Form("PROGRAMI")
        rsOgr("ADI")=request.Form("ADI")
        rsOgr("SOYADI")=request.Form("SOYADI")
        rsOgr("KAYIT_TARIHI")=request.Form("KAYIT_TARIHI")
        rsOgr("DANISMANI")=request.Form("DANISMANI")
        rsOgr("BABA_ADI")=request.Form("BABA_ADI")
        rsOgr("ANA_ADI")=request.Form("ANA_ADI")
        rsOgr("DOGUM_YERI")=request.Form("DOGUM_YERI")
        rsOgr("DOGUM_TARIHI")=request.Form("DOGUM_TARIHI")
        rsOgr("KAN_GRUBU")=request.Form("KAN_GRUBU")
        rsOgr("IL")=request.Form("IL")
        rsOgr("ILCE")=request.Form("ILCE")
        rsOgr("MAHALLE_KOY")=request.Form("MAHALLE_KOY")
        rsOgr("CILT_NO")=request.Form("CILT_NO")
        rsOgr("AILE_NO")=request.Form("AILE_NO")
        rsOgr("SIRA_NO")=request.Form("SIRA_NO")
        rsOgr("CINSIYET")=request.Form("CINSIYET")
        rsOgr("BIT_UNI")=request.Form("BIT_UNI")
        rsOgr("BIT_BOL")=request.Form("BIT_BOL")
        rsOgr("ADRES")=request.Form("ADRES")
        rsOgr("TELEFON")=request.Form("TELEFON")
        rsOgr("EMAIL")=request.Form("EMAIL")
        rsOgr("BASLAMA_DONEMI")=rsDONEM("DONEM_ID")
rsOgr.update

if conn.Errors.Count > 0 then
        for each error in conn Errors
        Response.Write(Error.Number&" : "&Error.Description)
        next
end if
```

Bu işlem sonucunda şekilde görülen kayıt OGRENCILER tablosuna yeni bir kayıt olarak girilecektir

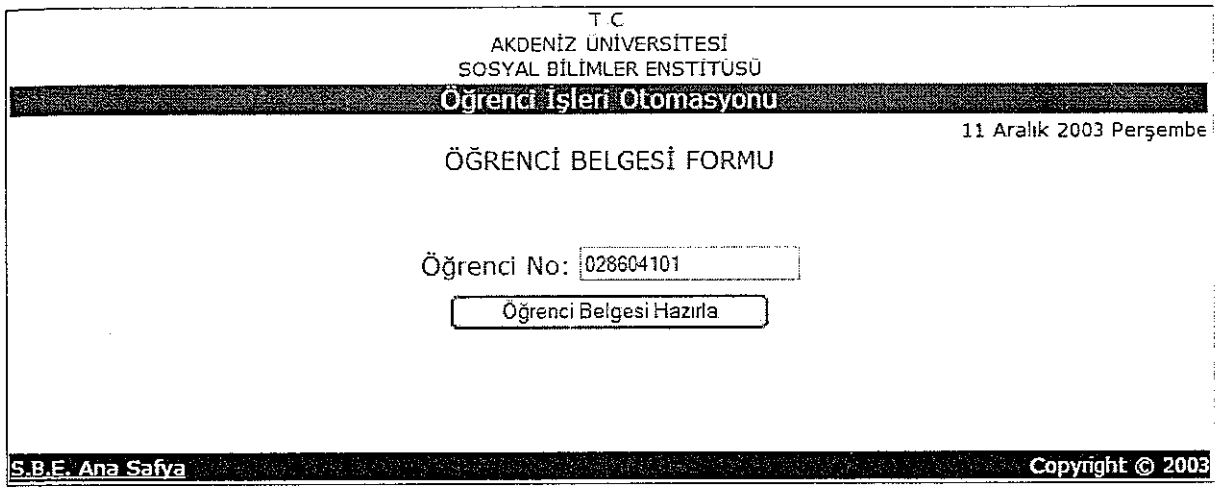
OGRENCILER : Tablo

O_ID	NR	P_ID	ADI	SOYADI	KAYIT_TARIHI	DANISMANI	CINSIYET	BABA_ADI	ANA_ADI	DOGUM
1	028604101	14	Emre	İPEKÇİ ÇETİN	30 09 2002		26 K	İbrahim Lütfü	Günnur	Lübeck

Şekil 3.4. S.B.E. Veritabanındaki Örnek Öğrenci Kaydının Görünümü

2.13.2. Öğrenci Belgesi

Otomasyon sisteminde öğrenci belgesi veritabanından gereken alanların alınmasıyla belirlenen formatta bir ekran çıktısı şeklinde hazırlanır. Bu ekran çıktısına göre A4 formatında yazıcı çıktısı da alınabilir. Kayıtlı bir öğrenciye öğrenci belgesi hazırlanması için aşağıdaki form kullanılır.



Şekil 3.5. Öğrenci Belgesi Form Ekranı

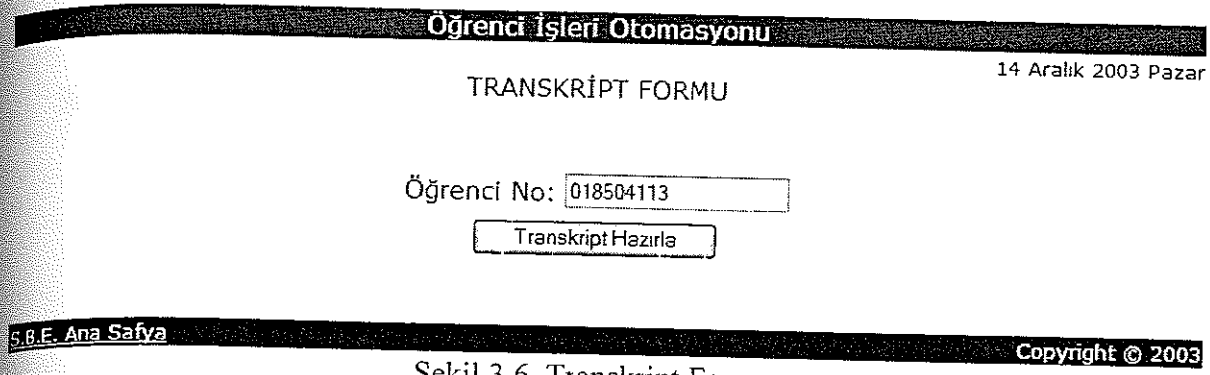
Öğrenci No alanına öğrencinin numarası girildikten sonra ogrenci_belgesi.asp dosyasında aşağıdaki sorgu çalışır ve getirilen veri bir kayıt setine alınır.

```
set rs=Server.CreateObject("ADODB.Recordset")
rsSQL="SELECT * FROM OGRENCILER WHERE NR='"&request.Form("OGR_NO")&"'"
rs.Open rsSQL, conn
```

Kayıt setine alınmış veri çıktı formunda belirlenen biçimde öğrenci belgesi oluşturmak için HTML kodları arasında kullanılarak çıktı belgesi oluşturulur.

2.13.3. Transkript

Bir öğrencinin transkriptinin çıkartılması için kayıt ve belge işlemleri altında yer alan transkript formu kullanılır. Şekil-3.6'da bu formun örneği görülmektedir.



Şekil 3.6. Transkript Formu

Transkript almak için transkript.asp dosyasında forma öğrenci numarası girildikten sonra yazılım bu bilgiyi aynı dosyaya sc=goster querystring'i ile birlikte gönderir. Girilen öğrenci numarasından OGRENCILER tablosunda öğrenci kimlik numarası (O_ID) seçilir ve DERS_KAYIT tablosunda bu ID ile kayıtlı ders ve bu derslere ait sonuç bilgileri alınır. Bu bilgiler aşağıdaki düzende bir ekran çıktısına dönüştürülür. Aynı zamanda bu çıktı yazıcı çıktısı olarak da alınabilir.

Öğrenci No: 018504113
Adı ve Soyadı: Serdar YAMAN
Programı: İşletme Yüksek Lisans

Ders Kodu	Ders Adı	Alınan Dönem	Başarı Notu	Sonuç	Alınan Kredi
80402520	Stratejik Yönetim	2001-2002 Bahar	80	Geçti	3
80402522	Sayısal Araştırma Yöntemleri	2001-2002 Bahar	60	Başarısız	0
80402524	Üretim Kontrol Sistemleri	2001-2002 Bahar	85	Geçti	3
80402542	Finansal Tablolar Analizi	2001-2002 Bahar	75	Geçti	3
80402526	Finansal Yönetim	2001-2002 Bahar	90	Geçti	3
80402550	Business English II (İşletme İngilizcesi II)	2001-2002 Bahar	90	Geçti	0
80402529	Örgütsel Davranış	2002-2003 Güz	70	Geçti	3
80402523	Pazarlama Yönetimi	2002-2003 Güz	85	Geçti	3
80402525	Yönetim Ekonomisi	2002-2003 Güz	75	Geçti	3
80402527	Maliyet ve Yönetim Muhasebesi	2002-2003 Güz	75	Geçti	3
80402521	Organizasyon Teorisi	2002-2003 Güz	80	Geçti	3
				Alınan Kredi Toplamı:	27

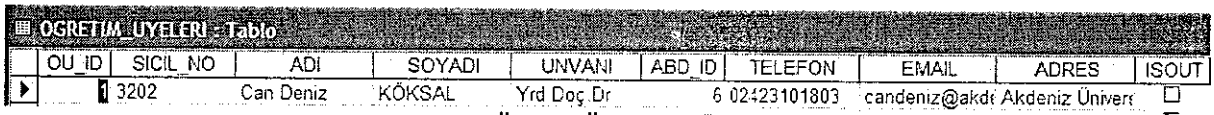
Şekil 3.7. Transkript Çıktısı Örneği

Transkriptte alınan toplam kredi hesaplanırken başarısız olunan dersler ve kredisiz dersler için alınan kredi sıfır olarak hesaplanır.

2.13.4. Yeni Öğretim Üyesi Bilgileri Girilmesi

Şekil 3.8. Yeni Öğretim Üyesi Kayıt Formu

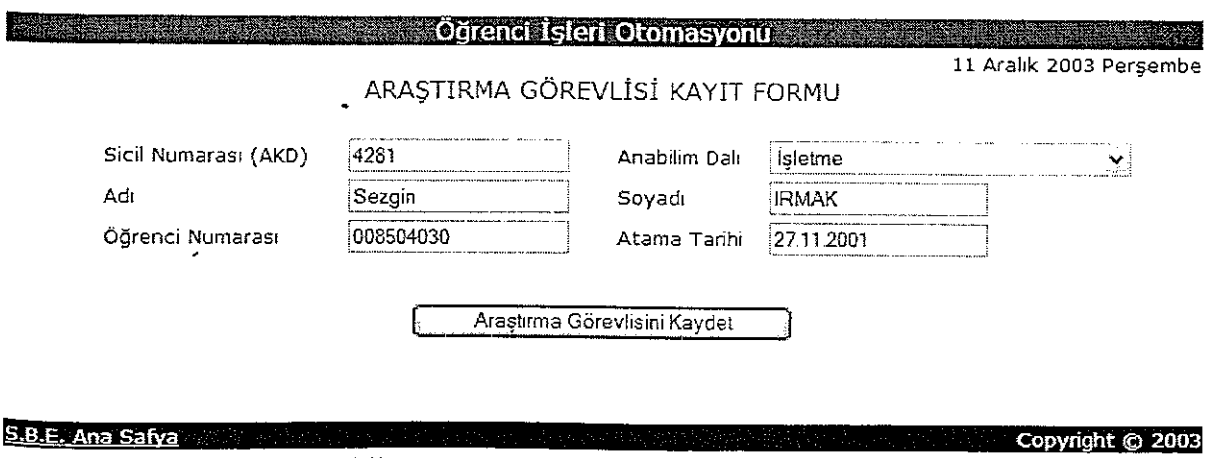
Bu form enstitü bünyesinde görev alacak öğretim üyelerinin bilgilerinin veritabanına girilmesi amacıyla kullanılır. Eğer bir öğretim üyesi enstitü programlarında enstitü dışından görev alıyorsa "Öğretim Üyesi Sosyal Bilimler Enstitüsü Dışından" seçeneği işaretlenir "Öğretim Üyesini Kaydet" butonuna basıldığında girilen bilgiler OGRETIM_UYELERI tablosuna kaydedilir.



Şekil 3.9. Örnek Öğretim Üyesi Kaydı

OGRETIM_UYELERI tablosunda OU_ID alanı, öğretim üyeleri için veritabanı işlemlerinde kullanılacak olan özel ve tek kimlik numarasıdır. Bu veritabanında kayıt sırasında otomatik olarak verilir.

2.13.5. Yeni Araştırma Görevlisi Bilgileri Girilmesi



Şekil 3.10. Araştırma Görevlisi Kayıt Formu

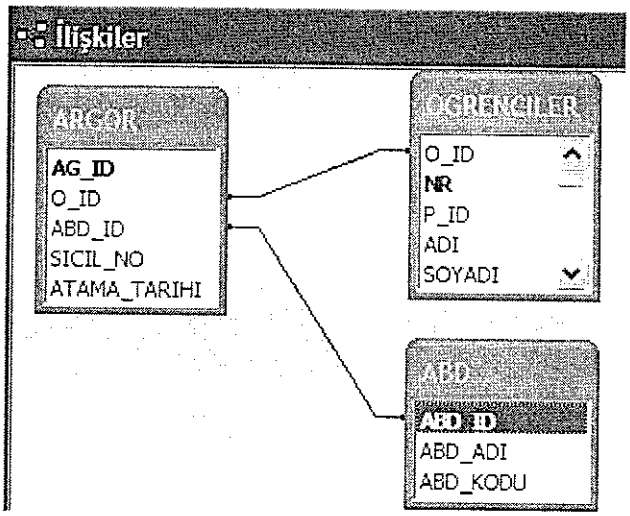
Bu form enstitüye bir araştırma görevlisi atandığı zaman bilgilerinin girilmesi amacıyla kullanılır. Enstitüye atanan araştırma görevlisinin anabilim dallarından birisinde öğrenci olması gerektiği için burada kişinin öğrenci numarası alınarak OGRENCILER tablosunda O_ID alanı ile ilişkilendirilmektedir. Aşağıda şekilde de görüleceği gibi veritabanında ARGOR tablosunda AG_ID ve O_ID tutulmaktadır.

ARGOR : Tablo

AG_ID	O_ID	ABD_ID	SICIL_NO	ATAMA_TARIHI
1	12	6	4281	27.11.2001

Şekil 3.11. Örnek Araştırma Görevlisi Kaydı

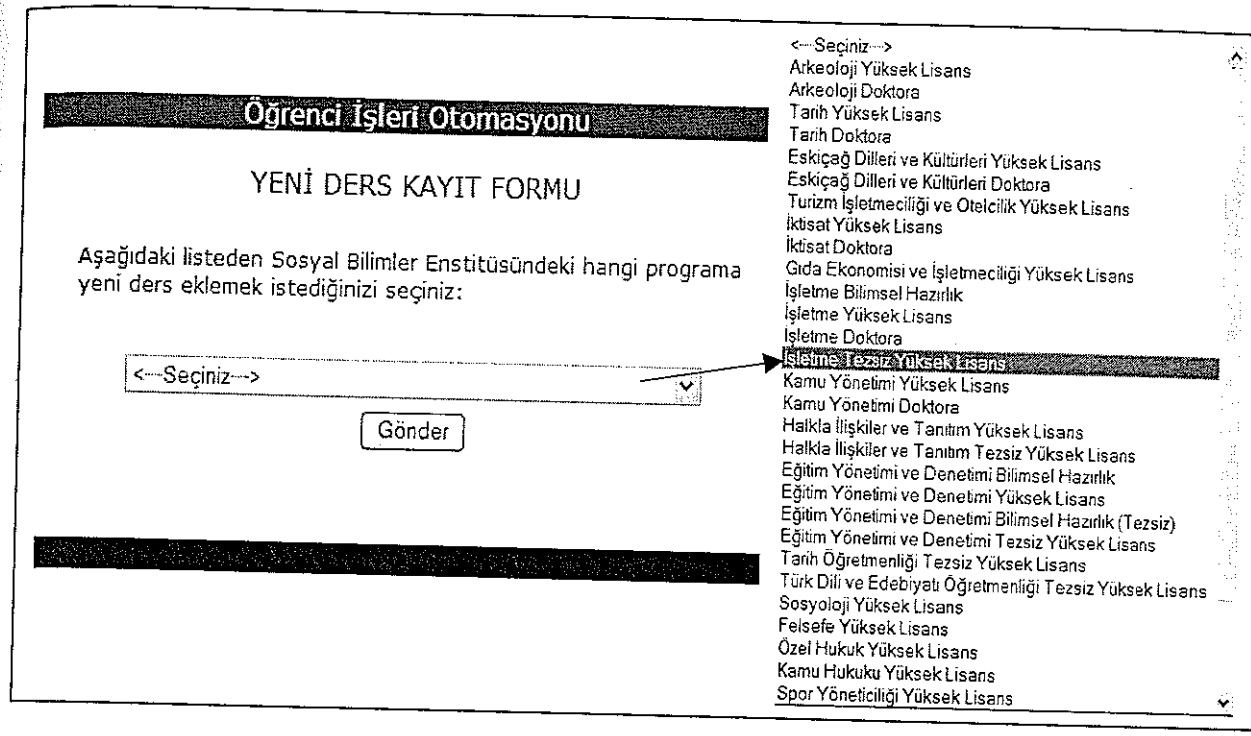
Atanan araştırma görevlisi ile ilgili tüm bilgiler aşağıda verilen ilişki yapısı doğrultusunda OGRENCILER tablosundan alınacaktır



Şekil 3.12 Araştırma Görevlisi Tablosu İlişkileri

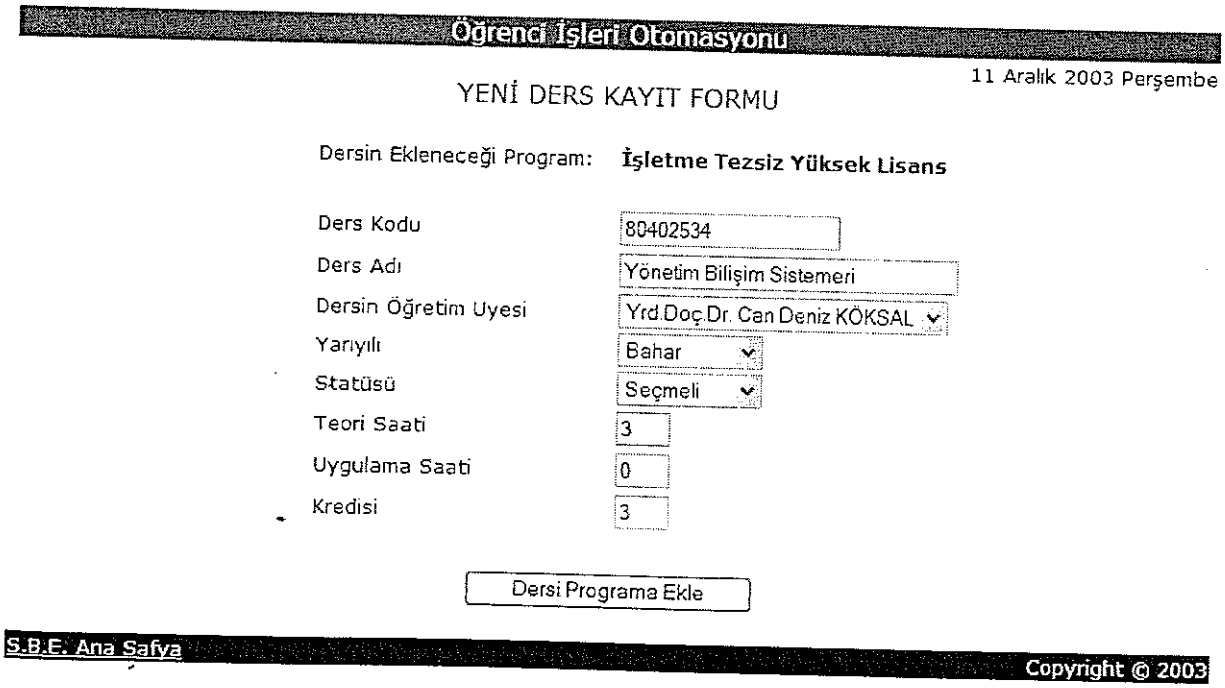
2.13.6. Bir Enstitü Programına Yeni Ders Eklenmesi

Otomasyon sisteminin bu fonksiyonu enstitü programlarından birine yeni bir ders ekleneceği zaman kullanılır. Yeni ders kayıt formunda kullanıcıya öncelikle hangi programa ders ekleyeceği sorulur. Açılır menüde listelenen programlardan biri seçildiğinde yazılım (yeni_ders.asp) program kimlik numarasını (P_ID) alır



Şekil 3 13. Yeni Ders Kayıt Formu Giriş Ekranı

Yeni_ders.asp prog=P_ID ve action=kayitform querystringlerini kendi içinde gönderir Action değişkeni değeri kayitform olduğunda seçilen program için aşağıdaki form oluşur.



Şekil 3 14. Yeni Ders Kayıt Formu Ekranı

Bu formda derse ait bilgiler girildikten sonra veri yeni_ders_db asp dosyasına gönderilir ve burada gerekli veritabanı bağlantısı kurularak AddNew metodu kullanılarak DERSLER tablosuna yeni kayıt olarak girilir.

DERSLER : Tablo

D_ID	DERS_KODU	P_ID	DERS_ADI	YARIYILI	OU_ID	STAT	TEORI	UYGU	KREDIS
639	80402528	13	İnsan Kaynakları Yönetimi	Bahar	34	S	3	0	3
640	80402530	13	Toplam Kalite Yönetimi	Bahar	56	S	3	0	3
641	80402532	13	International Marketing (Uluslararası Pazarl	Bahar	12	S	3	0	3
642	80402534	13	Yönetim Bilişim Sistemleri	Bahar	15	S	3	0	3

Şekil 3.15. Eklenen Yeni Dersin Veritabanı Kaydı

2.14. Ders İşlemleri

Sosyal Bilimler Enstitüsü Öğrenci İşleri Otomasyonunda Ders İşlemleri başlığı altında altı fonksiyon vardır. Bunlar; öğrencilerin derse kaydı, ders yoklama listelerinin çıkartılması, sınav cetvellerinin hazırlanması, yarıyıl sonu not girişi, ek süre alan öğrenci listelerinin çıkartılması ve ek süre bitiminde notların girilmesi olarak gruplandırılmışlardır

2.14.1. Öğrenci Ders Kaydı

Sosyal Bilimler Enstitüsü Öğrenci İşleri Otomasyonu'nun en önemli fonksiyonlarından biri öğrencilerin ders seçme işlemleridir. Öğrencinin içinde bulunulan döneme ait derslere kayıt yaptırması için aşağıdaki form kullanılır

Öğrenci İşleri Otomasyonu

12 Aralık 2003 Cuma

ÖĞRENCİ DERS KAYIT FORMU

Öğrenci No: 018504113

Program Derslerini Listele

Farklı Bir Programdan Ders Kaydı İçin Tıklayınız-->

S.B.E. Ana Safya

Copyright © 2003

Şekil 3.16. Öğrenci Ders Kayıt Formu Giriş Ekranı

Öğrenci ders kayıt formunda öğrenci kendi programının o yarıyıla ait derslerini seçebilir veya farklı bir programdan bir derse de kaydolabilir. Kendi programından ders seçmek için

öğrenci numarasını ilgili alana girdikten sonra bu bilgi form dosyasından ders_form.asp dosyasına sc=list querystring'i ile birlikte gönderilir. Bu işlem sonucunda Şekil-3.17'de görüldüğü gibi Ders_form.asp dosyası öğrencinin kayıtlı olduğu programın içinde bulunulan yarıyıla ait derslerini listeler.

Öğrenci No: 018504113
Adı ve Soyadı: Serdar YAMAN
Programı: İşletme Yüksek Lisans
Dönem :2002-2003 Bahar

Ders Kodu	Ders Adı	Öğretim Üyesi	Statüsü	Kredisi	Seç
80402520	Stratejik Yönetim	Prof.Dr. Fulya SARVAN	Z	3	☐
80402522	[illegible]	Yrd.Doç.Dr. Ayşe KORUÜZÜM			☐
80402524	Üretim Kontrol Sistemleri	Prof.Dr. Orhan KURUÜZÜM	Z	3	☐
80402542	Finansal Tablolar Analizi	Prof.Dr. Ayten ERSOY	Z	3	☐
80402526	Finansal Yönetim	Yrd.Doç.Dr. Mehmet ŞEN	S	3	☐
80402528	İnsan Kaynakları Yönetimi	Yrd.Doç.Dr. Nilgün ANAFARTA	S	3	☐
80402530	Toplam Kalite Yönetimi	Prof.Dr. Fulya SARVAN	S	3	☐
80402532	International Marketing (Uluslararası Pazarlama)	Dr. Nancy Thomas WARD	S	3	☐
80402534	Yönetim Bilişim Sistemleri	Yrd.Doç.Dr. Can Deniz KÖKSAL	S	3	☐
80402536	Esnek Üretim Sistemleri	Prof.Dr. Orhan KURUÜZÜM	S	3	☐
80402538	Menkul Kıymetler ve Portföy Yönetimi	Öğr.Gör.Dr. Hakan ER	S	3	☐
80402540	Uluslararası Finans	Öğr.Gör.Dr. Aslıhan ERSOY BOZCUK	S	3	☐
80402544	Uygulamalı Stokastik Süreçler	Prof.Dr. Orhan KURUÜZÜM	S	3	☐
80402546	Hizmet Pazarlaması	Doç.Dr. Şafak AKSOY	S	3	☐
80402548	İmalat Kaynaklarını Planlama	Prof.Dr. Orhan KURUÜZÜM	S	3	☐
80402552	Crisis Management (Kriz Yönetimi)	Dr. Nancy Thomas WARD	S	3	☐
80402550	Business English II (İşletme İngilizcesi II)	Dr. Nancy Thomas WARD	S	0	☐

Ders Kayıt

Şekil 3.17. Program Dersleri Listesi

Ders seçim ekranında kırmızı alt rengi ile görülen ders öğrencinin bir önceki yıl alıp başarısız olduğu bir dersi belirtir ve dersi bu yıl yeniden seçmesi zorunludur. Eğer bu dersin bir eşdeğer dersi sistemde tanımlı ise öğrenci bunun yerine o dersi de alabilir. Dersin adı üzerine tıklanarak veritabanında ESDEGER_DERSLER tablosundan bu dersin eşdeğer dersi olup olmadığına bakılır, eğer varsa listelenir. Eşdeğer dersler tablosunda o dersin ve eşdeğer dersin özel kimlik numaraları (D_ID ve ES_D_ID) ve ilgili Yönetim Kurulu Kararı bulunur.

Ders seçimi yapıldıktan sonra seçilen ders ve öğrenci bilgileri ders_form_db.asp dosyasına gönderilir ve bu yazılım veritabanında AKTIF_DERSLER ve DERS_KAYIT tablolarına gerekli kayıtları girer.

DERS_KAYIT : Tablo

DK_ID	O_ID	D_ID	DONEM_ID	DEVAM	EK	NOTU	SONUC
1	9	634	6	☑	☐	80	☑
2	9	635	6	☑	☐	60	☐
3	9	636	6	☑	☐	85	☑
4	9	637	6	☑	☐	70	☑
5	9	638	6	☑	☐	90	☑
6	9	650	6	☑	☐	90	☑
7	9	634	8	☐	☐	0	☐
8	9	635	8	☐	☐	0	☐
9	9	636	8	☐	☐	0	☐
10	9	637	8	☐	☐	0	☐
11	9	638	8	☐	☐	0	☐
12	9	639	8	☐	☐	0	☐
13	9	640	8	☐	☐	0	☐
14	9	642	8	☐	☐	0	☐
18	9	784	8	☐	☐	0	☐

Şekil 3.18. DERS_KAYIT Tablosunda Tutulan Öğrenci Ders Kayıtları

DERS_KAYIT tablosunda her öğrencinin (O_ID) her dönemde (DONEM_ID) aldığı her ders (D_ID) için bir kayıt tutulur. Ayrıca bu derslere ait devam durumu (DEVAM), ek süre (EK), not (NOTU), ve başarı durumu (SONUC) bilgileri tutulur. Bu yolla öğrencinin bir dersi iki kez tekrarlaması durumunda hem önceki dönem bilgileri hem yeni bilgileri tutulur.

AKTIF_DERSLER : Tablo

ID	DONEM_ID	D_ID	OU_ID	OGRENCI_SAYISI
7	8	634	4	8
8	8	635	1	6
9	8	636	3	15
10	8	637	5	4
11	8	638	8	7
12	8	639	12	5
13	8	640	4	6
14	8	642	2	8
16	8	784	1	1

Şekil 3.19 AKTIF_DERSLER Tablosunda Tutulan Ders Kayıtları

AKTIF_DERSLER tablosu her dönem açılan derslere ait bilgilerin tutulması amacıyla kullanılır. Eğer o dönemde bir ders açılmamışsa o dersin bilgisi bu tabloda yer almaz. Bir öğrenci derse ilk kayıt yaptırdığı zaman aktif dönem için o ders eklenir. Eğer ders zaten açıksa sadece OGRENCI_SAYISI bilgisi SQL UPDATE ile güncellenir. AKTIF_DERSLER tablosunda ders kimlik numarası (D_ID), dersin hangi dönem açıldığı (DONEM_ID), o dönem hangi öğretim üyesi tarafından verildiği (OU_ID) ve kaç öğrencinin dersi aldığı (OGRENCI_SAYISI) bilgileri tutulur. Aşağıda AKTIF_DERSLER tablosunda yeni ders açılmasına veya açık dersin öğrenci sayısının güncellenmesine karar veren ders_form_db.asp dosyasındaki kod verilmiştir.

```
set rsAKTIF=Server.CreateObject("ADODB.Recordset")
rsAKTIFSQL="SELECT * FROM AKTIF_DERSLER "&_
        WHERE DONEM_ID="&DONID&" AND D_ID="&DER&""
rsAKTIF.Open rsAKTIFSQL,conn,1,3

if rsAKTIF.EOF then
        rsAKTIF.AddNew
        rsAKTIF("DONEM_ID")=DONID
        rsAKTIF("D_ID")=DER
        rsAKTIF("OU_ID")=request.Form("OUID"&i)
        rsAKTIF("OGRENCI_SAYISI")=1
        rsAKTIF.Update
else
        OGR_SAY=rsAKTIF("OGRENCI_SAYISI")+1
        SQLAKTIFupdate="UPDATE AKTIF_DERSLER "&_
                SET OGRENCI_SAYISI="&OGR_SAY&" "&_
                WHERE DONEM_ID="&DONID&" AND D_ID="&DER&""
        conn.Execute SQLAKTIFupdate
end if
rsAKTIF.Close
```

Ders kayıt işlemleri tamamlandığı zaman kullanıcıya Şekil-3.20'de örneği görülen bir rapor verilir.

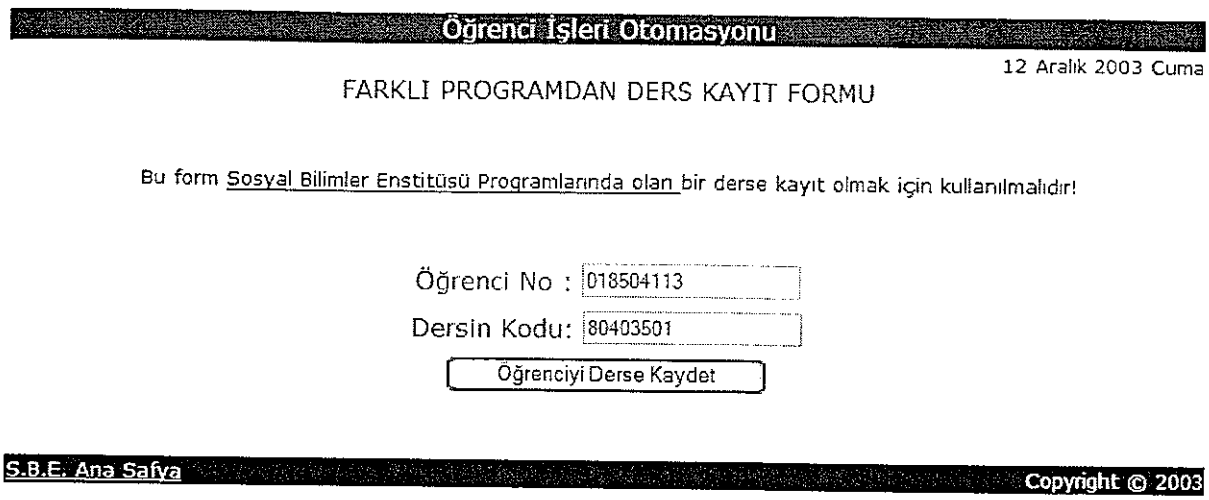
Öğrenci ders kayıt bilgileri veritabanına kaydedildi!

Öğrenci	Ders Adı
Serdar YAMAN	Stratejik Yönetim
Serdar YAMAN	Sayısal Araştırma Yöntemleri
Serdar YAMAN	Üretim Kontrol Sistemleri
Serdar YAMAN	Finansal Tablolar Analizi
Serdar YAMAN	Finansal Yönetim
Serdar YAMAN	İnsan Kaynakları Yönetimi
Serdar YAMAN	Toplam Kalite Yönetimi
Serdar YAMAN	Yönetim Bilişim Sistemleri

Tamam Yeni Kayıt Farklı Programdan Ders Seçimi-->

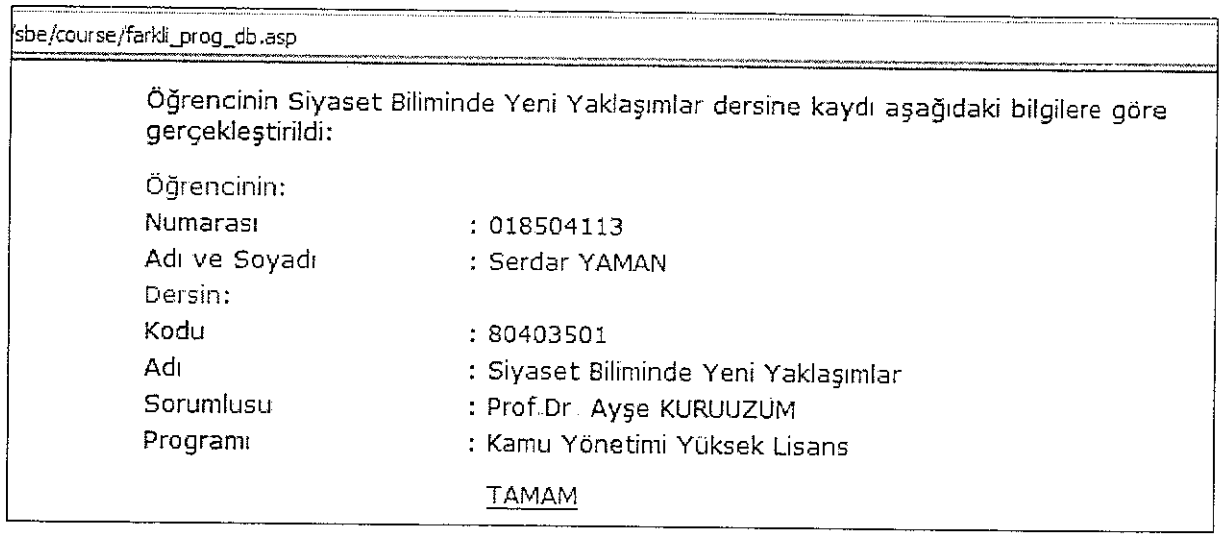
Şekil 3.20 Ders Kayıt İşleminin Tamamlandığını Belirten Ekran Çıktısı

Bir öğrencinin kendi programından farklı bir programdan ders seçebilmesi için, farklı programdan ders kayıt formu kullanılır. Burada öğrenci numarası ve seçilen dersin kodu girilerek farkli_prog_db.asp dosyasına gönderilir. Bu yazılım yukarıda anlatılan ders kayıt işlemlerine benzer bir yol izleyerek öğrencinin derse kayıt işlemini gerçekleştirir.



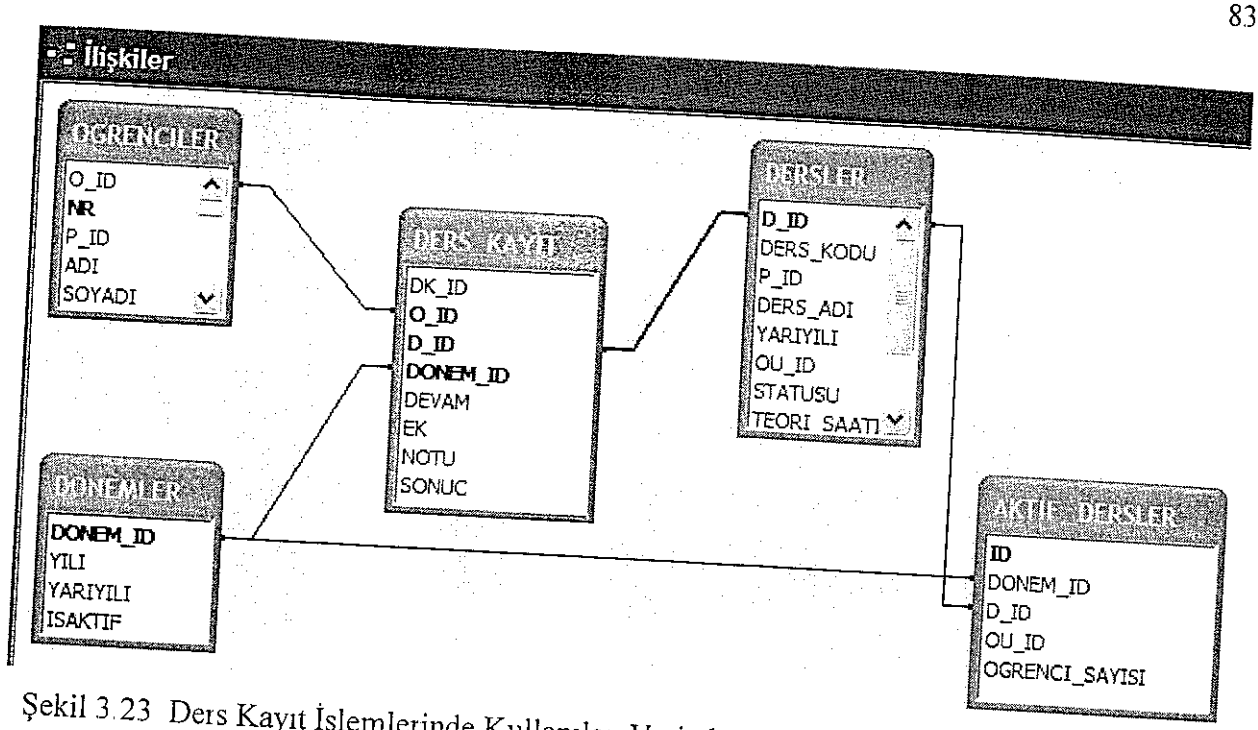
Şekil 3.21 Farklı Programdan Bir Derse Kayıt Formu

Öğrencinin farklı programdan bir derse kayıt işlemi tamamlandıktan sonra aşağıdaki şekilde bir ekran çıktısı görüntülenir



Şekil 3.22 Farklı Programdan Ders Kayıt İşleminin Tamamlandığını Belirten Ekran Çıktısı

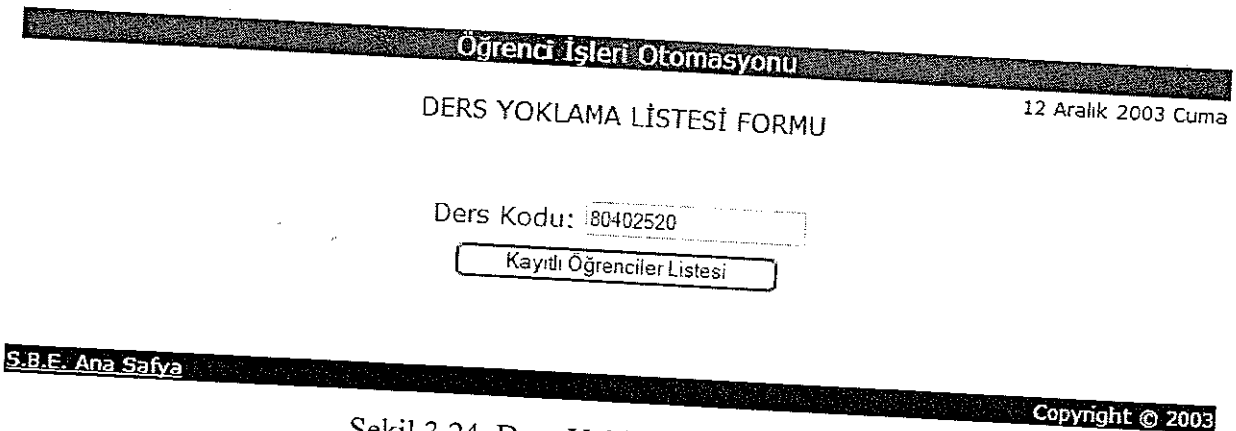
Ders kayıt işlemleri sırasında kullanılan veritabanı tabloları ve aralarındaki ilişkiler aşağıdaki şekilde gösterilmiştir Burada bütün tablolar arasında bire-çok (1:M) ilişki vardır OGRENCILER, DERSLER ve DONEMLER tabloları ana dosyalar, DERS_KAYIT ve AKTIF_DERSLER tabloları ise işlem dosyalarıdır



Şekil 3.23 Ders Kayıt İşlemlerinde Kullanılan Veritabanı Tabloları ve Aralarındaki İlişkiler

2.14.2. Ders Yoklama Listesi ve Sınav Cetveli Hazırlanması

Derslere ait yoklama listeleri ve sınav cetvelleri aynı mantıkla hazırlanmaktadır. Aşağıda örnek bir ders yoklama listesi formu ve örnek bir sınav cetveli ekran çıktısı görülmektedir. Ders yoklama listelerinin ve sınav cetvellerinin yazıcı çıktısı da alınabilmektedir.



Şekil 3.24. Ders Yoklama Listesi Formu

Ders yoklama listesi formuna ders kodu girildikten sonra bu bilgi ilgili yazılıma gönderilerek DERS_KAYIT tablosundan o dönem derse kayıtlı öğrenci bilgileri ve ilişkili tablolardan gerekli bilgiler alınarak ekran çıktısı ve yazıcı çıktısı alınabilmektedir. Aşağıda bu işlem için gereken örnek ASP kodları ve SQL sorguları verilmiştir.

```
kod=TRIM(request.Form("DERS_KODU"))
set rs=Server.CreateObject("ADODB.Recordset")
rsSQL="SELECT P_ID, D_ID, DERS_KODU, DERS_ADI, OU_ID
        FROM DERSLER WHERE DERS_KODU='"&kod&"'"
rs Open rsSQL, conn
dersid=rs("D_ID")
set rsDK=Server.CreateObject("ADODB.Recordset")
rsDKSQL="SELECT O_ID FROM DERS_KAYIT
        WHERE D_ID="&dersid&" AND DONEM_ID="&rsDONEM("DONEM_ID")&""
rsDK.Open rsDKSQL,conn
```

2002-2003 Bahar Yarıyılı Sonu Sınav Cetveli

Dersin Kodu: 80402520
Dersin Adı: Stratejik Yönetim
Öğretim Üyesi: Prof Dr. Fulya SARVAN
Anabilim Dalı ve Programı: İşletme Yüksek Lisans

S.No	Öğrenci No	Adı ve Soyadı	Devam Durumu	Başarı Notu	Başarı Notu (Yazı ile)	Sonuç
1	018504113	Serdar YAMAN				
2	998504110	Nilgün ÇELİK				
3	028504123	Uğur ÖZTAŞ				
4	018504007	Mehmet BOZDEMİR				

Şekil 3.25. Sınav Cetveli Ekran Çıktısı Örneği

2.14.3. Yarıyıl Sonu Not Girişi

Yarıyıl sonunda notların girilmesi otomasyon programının bir diğer önemli fonksiyonudur. Not girişi için öncelikle ders kodu girilerek (Şekil-3.26) içinde bulunulan dönem için o derse kayıt yaptırmış öğrenciler listelenir (Şekil-3.27).

Öğrenci İşleri Otomasyonu

12 Aralık 2003 Cuma

YARIYIL SONU NOT GİRİŞ FORMU

Ders Kodu: 80402522

Kayıtlı Öğrenciler Listesi

S.B.E. Ana Sayfa Copyright © 2003

Şekil 3.26. Yarıyıl Sonu Not Giriş Formu

2002-2003 Bahar Dönemi Sınav Sonuçları

Dersin Kodu: 80402522
Dersin Adı: Sayısal Araştırma Yöntemleri
Öğretim Üyesi: Prof Dr. Ayşe KURUÜZÜM
Anabilim Dalı ve Programı: İşletme Yüksek Lisans

S.No	Öğrenci No	Adı ve Soyadı	Ek	Devam	Başarı Notu	Başarılı
1	018504113	Serdar YAMAN	☑			
2	998504110	Nilgün ÇELİK	☑			
3	028504123	Uğur ÖZTAŞ	☐	☑	70	☑
4	018504007	Mehmet BOZDEMİR	☐	☑	75	☑

Notları Kaydet

Şekil 3 27 Yayıl Sonu Not Girişi Öğrenci Listesi

Öğrencilerin ek süre, devam, başarı notu ve başarı durumu bilgileri enstitü süreçleri içerisinde öğretim üyelerinden gelen sınav cetvellerindeki bilgilere göre girilir. Öğrenci listesini ve not girişi formunu oluşturan not_girisi asp dosyası bu bilgileri not_giris_db.asp dosyasına gönderir

```
counter=request.Form("counter")
set rsDK=Server.CreateObject("ADODB.Recordset")
rsDKSQL="SELECT DEVAM, EK, NOTU, SONUC "&_
                FROM DERS_KAYIT "&_
                WHERE D_ID="&DERSID&" AND DONEM_ID="&DONID&""
rsDK.Open rsDKSQL,conn,1,3
for i=1 to counter
                OGRID  =Request.Form("OGRID"&i)
        if Request.Form("DEVAM"&i)="on" then
                DEVAMA=Request.Form("DEVAM"&i)
        else
                DEVAMA="off"
        end if
EKA     =Request.Form("EK"&i)
NOTUA =Request.Form("NOTU"&i)
        if Request Form("SONUC"&i)="on" then
                SONUCA=Request.Form("SONUC"&i)
        else
                SONUCA="off"
        end if
while not rsDK.EOF
        if Request.Form("EK"&i)="on" then
                SQLDKupdate="UPDATE DERS_KAYIT SET EK="&EKA&" "&_
                WHERE O_ID="&OGRID&" AND D_ID="&DERSID&"  "&_
                AND DONEM_ID="&DONID&""
        else
                SQLDKupdate="UPDATE DERS_KAYIT  "&_
                SET DEVAM="&DEVAMA&", NOTU="&NOTUA&", SONUC="&SONUCA&"  "&_
                WHERE O_ID="&OGRID&" AND D_ID="&DERSID&"  "&_
                AND DONEM_ID="&DONID&""
        end if
        conn.Execute SQLDKupdate
rsDK.MoveNext
wend
rsDK.MoveFirst
next
rsDK.Close
```

Not_girisi_db.asp yazılımı yukarıda örneği verilen kod yardımıyla öğrencilerin ek süre alıp almadığının kontrolünü yapar. Eğer ek süre almışsa ek süre sonunda not girişine izin verecek şekilde ek bilgisini, eğer almamışsa sonuçları veritabanındaki DERS_KAYIT tablosuna kaydeder. Aşağıdaki şekilde DERS_KAYIT tablosunda yukarıda ders kayıt formunda örneği görülen bilgilerin kaydedilmiş durumu görülmektedir.

DERS_KAYIT : Tablo

DK_ID	O_ID	D_ID	DONEM_ID	DEVAM	EK	NOTU	SONUC
1	9	634	6	☑	☐	80	☑
2	9	635	6	☑	☐	60	☐
3	9	636	6	☑	☐	85	☑
4	9	637	6	☐	☑		☐
5	9	638	6	☑	☐	90	☑
6	9	650	6	☑	☐	90	☑

Şekil 3.28. DERS_KAYIT Tablosuna Sonuçların Kaydedilmesi

2.14.4. Ek Süre Alan Öğrenci Listesi Oluşturulması

Ek süre alan öğrencilerin listesi içinde bulunulan dönem için DERS_KAYIT tablosundaki EK bilgisine göre oluşturulur. Aşağıda bunu işlemi gerçekleştiren örnek SQL sorgusu verilmiştir.

```
set rsDK=Server.CreateObject("ADODB Recordset")
rsDKSQL="SELECT O_ID FROM DERS_KAYIT
WHERE D_ID="&dersid&" AND EK=True AND DONEM_ID="&rsDONEM("DONEM_ID")&""
rsDK.Open rsDKSQL,conn
```

Yukarıdaki sorguya göre DERS_KAYIT tablosunda ek süre almış öğrenci kimlik numaraları (O_ID) ve ilişkili tablolardan öğrenci ve ders bilgileri alınarak aşağıdaki form çıktısı oluşturulur.

2002-2003 Bahar Yarıyılı Sonu Ek Süre Alanlar Listesi

Dersin Kodu: 80402522
Dersin Adı: Sayısal Araştırma Yöntemleri
Öğretim Üyesi: Prof Dr. Ayşe KURUÜZÜM
Anabilim Dalı ve Programı: İşletme Yüksek Lisans

S.No	Öğrenci No	Adı ve Soyadı	Devam Durumu	Başarı Notu	Başarı Notu (Yazı ile)	Sonuç
1	018504113	Serdar YAMAN				
2	998504110	Nilgün ÇELİK				

Şekil 3.29. Ek Süre Alanlar Listesi

2.14.5. Ek Süre Sonu Not Girişi

Ek süre sonunda DERS_KAYIT tablosundan ek süre alan öğrencilerin listesi oluşturularak aşağıdaki form dosyasına not girişleri yapılır. Bu formda artık ek süre seçeneği bulunmamaktadır.

2002-2003 Bahar Dönemi Sınav Sonuçları

Dersin Kodu: 80402522
Dersin Adı: Sayısal Araştırma Yöntemleri
Öğretim Üyesi: Prof.Dr. Ayşe KURUÜZÜM
Anabilim Dalı ve Programı: İşletme Yüksek Lisans

S.No	Öğrenci No	Adı ve Soyadı	Devam	Başarı Notu	Başarılı
1	018504113	Serdar YAMAN	☐		☐
2	998504110	Nilgün ÇELİK	☐		☐

Notları Kaydet

Şekil 3.30. Ek Süre Sonu Not Giriş Formu

Yukarıdaki forma (ek_not_girisi.asp) öğretim üyelerinden gelen sonuçlar girildikten sonra ek_not_girisi_db.asp dosyasına gönderilir. Bu yazılım aşağıdaki kodlar yardımıyla veritabanında DERS_KAYIT tablosunu günceller. Ek süre seçeneği kaldırılarak sonuç bilgileri UPDATE edilir.

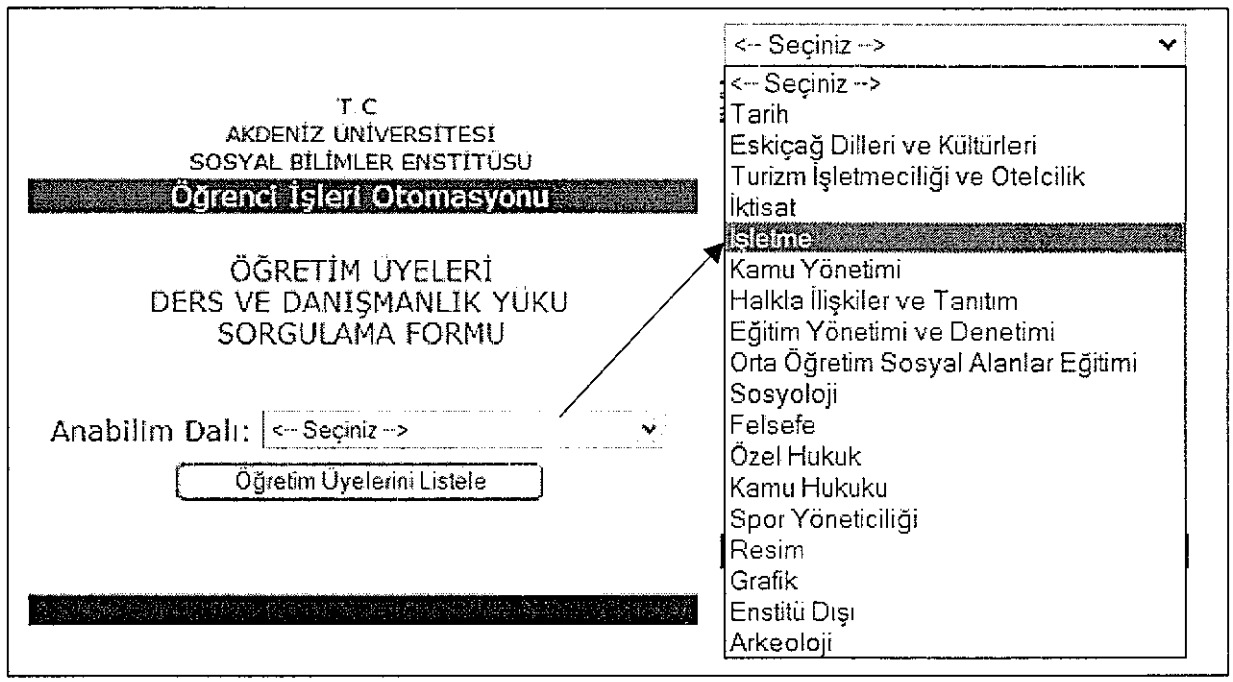
```
for i=1 to counter

if Request.Form("DEVAM"&i)="on" then
        DEVAMA=Request.Form("DEVAM"&i)
else
        DEVAMA="off"
end if
EKA             ="off"
NOTUA           =Request.Form("NOTU"&i)
if Request.Form("SONUC"&i)="on" then
        SONUCA=Request.Form("SONUC"&i)
else
        SONUCA="off"
end if

while not rsDK.EOF

SQLDKupdate="UPDATE DERS_KAYIT "&_
SET DEVAM="&DEVAMA&", EK="&EKA&", NOTU="&NOTUA&", SONUC="&SONUCA&" "&_
WHERE O_ID="&OGRID&" AND D_ID="&DERSID&" AND DONEM_ID="&DONID&""

conn.Execute SQLDKupdate
rsDK.MoveNext

wend

rsDK.MoveFirst
next
```

2.15. Diğer Öğrenci ve Öğretim Üyesi İşlemleri

Bu bölümde otomasyon programının iki temel fonksiyonu olan anabilim dallarına göre öğretim üyelerinin ders ve danışmanlık yüklerinin raporlanması ile kayıtlı olunan programlara göre öğrenci ders ve tez bilgilerinin raporlanması açıklanmaktadır.

2.15.1. Öğretim Üyeleri Ders ve Danışmanlık Yüklerinin Raporlanması

Öğretim üyelerinin içinde bulunulan dönemde kaç ders açtıklarının, bu derslerin toplam kredisinin ne olduğunun ve o dönemde üzerlerinde kaç danışmanlık bulunduğunun raporlanması için öğretim üyeleri ders ve danışmanlık yükleri sorgulama formu kullanılır.



Şekil 3.31 Öğretim Üyeleri Ders ve Danışmanlı Yükleri Sorgu Formu

Burada öğretim üyeleri üzerindeki yükler anabilim dallarına göre listelenir. İstenen anabilim dalı seçildikten sonra bu bilgi ou_ders_yuku_liste asp dosyasına gönderilir. Burada OGRETIM_UYELERI tablosundan anabilim dalı kimlik numarası (ABD_ID) alanı seçilen anabilim dalı olan öğretim üyeleri bilgileri alınır. Öğretim üyesi kimlik numarası (OU_ID) yoluyla da ilişkili tablolar olan DERS_KAYIT tablosundan ders bilgileri, verilen özel konular dersi kodları yoluyla danışmanlık sayıları hesaplanır. Aşağıda İşletme Anabilim Dalı için örnek Öğretim Üyeleri Ders ve Danışmanlık Yükleri raporu görülmektedir.

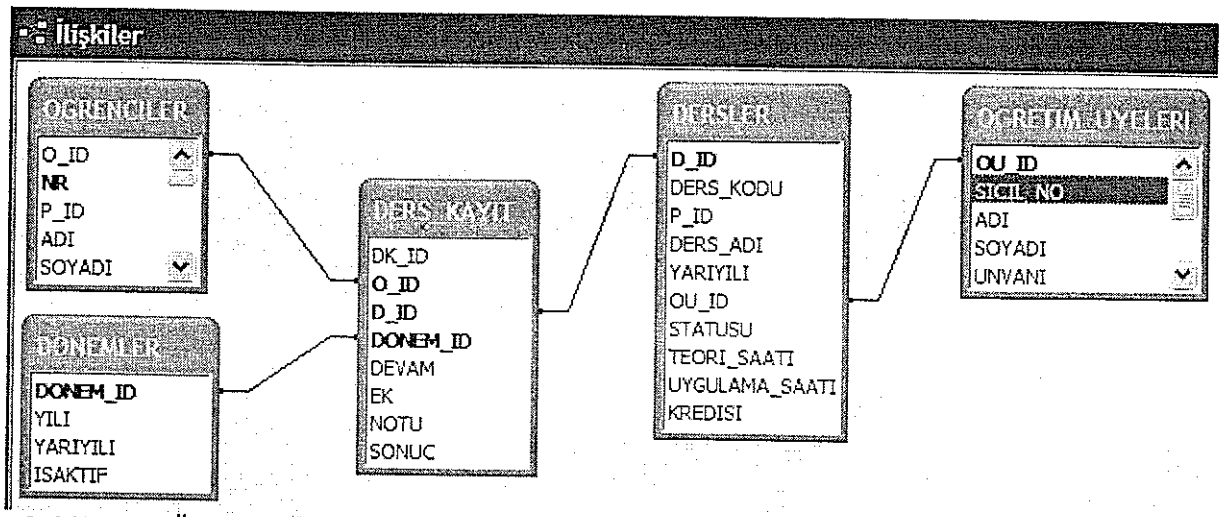
İşletme Anabilim Dalı Öğretim Üyeleri Ders ve Danışmanlık Yükleri			
Öğretim Üyesi	Ders Sayısı	Toplam Kredi	Danışmanlık Sayısı
Prof.Dr. Ayşe KURUÜZÜM	3	9	3
Yrd.Doç.Dr. Can Deniz KÖKSAL	2	6	2
Prof.Dr. Orhan KURUÜZÜM	1	3	4
Prof.Dr. Fulya SARVAN	4	9	2
Prof.Dr. Ayten ERSOY	3	9	3
Prof.Dr. Ferda ERDEM	3	6	3
Doç.Dr. Şafak AKSOY	2	6	2
Yrd.Doç.Dr. Mehmet ŞEN	2	6	3
Yrd.Doç.Dr. İrfan ÖZCAN	1	3	2
Öğr.Gör.Dr. Hakan ER	1	3	1
Öğr.Gör.Dr. Aslıhan ERSOY BOZCUK	1	3	1
Yrd.Doç.Dr. Nilgün ANAFARTA	1	3	2

Şekil 3.32. Öğretim Üyeleri Ders ve Danışmanlık Yükleri Çıktı Örneği

Bu tablo için hesaplamalar aşağıda örneği verilen kod yardımıyla yapılır

```
DYsql="SELECT * FROM AKTIF_DERSLER "&_
        WHERE OU_ID="&rs("OU_ID")&" AND DONEM_ID="&rsDONEM("DONEM_ID")&""
set rsDY=conn.Execute(DYsql)

while not rsDY.EOF

        KREDIsql="SELECT KREDISI, DERS_KODU FROM DERSLER "&_
                        WHERE D_ID="&rsDY("D_ID")
        set rsKREDI=conn.Execute(KREDIsql)

        derssay=derssay+1
        topkredi=topkredi+rsKREDI("KREDISI")
        if right(rsKREDI("DERS_KODU"),2)>59 and right(rsKREDI("DERS_KODU"),2)<82 then
                topdanismanlik=topdanismanlik+rsDY("OGRENCI_SAYISI")
                OZKON_ID=rsDY("D_ID")
        end if
        rsKREDI.Close
        set rsKREDI=nothing

rsDY.MoveNext
wend
```

Derslere ve kredilerine ait bilgiler de DERS_KAYIT tablosunda ve DERSLER tablosundaki anahtar alan olan ders kimlik numaraları alanı (D_ID) yoluyla DERSLER tablosundan alınır. Aşağıda OGRENCILER, DERS_KAYIT, OGRETIM_UYELERI ve DERSLER tabloları arasındaki ilişkiler gösterilmiştir.



Şekil 3.33. Öğretim Üyeleri Yükleri Çıktısında Kullanılan Tablolar ve Aralarındaki İlişkiler

Yukarıda Şekil-3.32'de gösterilen raporda her öğretim üyesi için ders sayısı alanındaki sayının üzerine tıklandığında öğretim üyesinin o dönem açtığı dersler, derslerin programları ve bu dersi alan toplam öğrenci sayısı listelenmektedir. Aşağıda Şekil-3.24'de bunun örneği görülmektedir. Bu tabloda öğretim üyesinin adı, dersin açıldığı dönem ve dersin o döneme ait bilgileri yer almaktadır.

Öğretim Üyesi: Prof.Dr. Orhan KURUUZUM

Dönem: 2002-2003 Güz

Sıra	Ders Kodu	Ders Adı	Programı	Öğrenci Sayısı
1	80402525	Yönetim Ekonomisi	İşletme Yüksek Lisans	11

Şekil 3.34. Öğretim Üyesinin Verdiği Derslerin Listelenmesi

Aynı raporda danışmanlık sayısı alanındaki sayıya tıklandığında da öğretim üyesinin hangi programlarda hangi öğrencilere danışmanlık yaptığının listesi benzer bir mantıkla görüntülenebilmektedir (Şekil-3.35). Burada aynı form içerisinde action=danismanliklar, OUID=[öğretim üyesi kimlik no] ve OZKONID=[öğretim üyesine kayıtlı özel konular dersi kimlik no] querystring olarak gönderilmektedir.

Öğretim Üyesi: Prof.Dr. Ayşe KURUUZUM

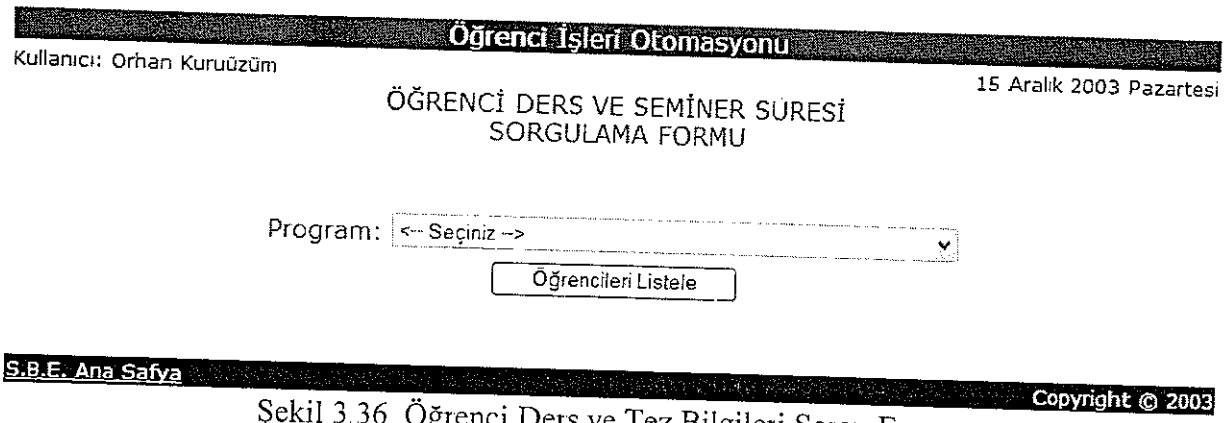
Dönem: 2003-2004 Güz

Sıra	Öğrenci No	Adı ve Soyadı	Programı
1	008504030	Sezgin Irmak	İşletme Yüksek Lisans
2	028604101	Emre İpekçi Çetin	İşletme Doktora

Şekil 3.35. Öğretim Üyesinin Danışmanlıklarının Listelenmesi

2.15.2. Öğrencilerin Ders ve Tez Bilgilerinin Raporlanması

Öğrencilerin ders ve tez durumlarının içinde bulunulan döneme göre raporlanması için aşağıdaki form kullanılır. Burada öğrenciler kayıtlı oldukları programlara göre listelenmektedirler. Otomasyon sisteminde öğretim üyeleri bağlı oldukları anabilim dalına göre öğrenciler ise kayıtlı olduklara programlara göre değerlendirilmektedirler.



Şekil 3.36 Öğrenci Ders ve Tez Bilgileri Sorgu Formu

İstenilen program formda seçildikten sonra program kimlik numarasına (P_ID) göre OGRENCILER tablosundan öğrenciler belirlenir. Bu tablodaki öğrenci kimlik numarası (O_ID) alanı ile ilişkili DERS_KAYIT, AKTIF_DERSLER ve OGRENCI_TEZ tablolarından öğrencilerin önceden aldığı dersler, almakta olduğu dersler ve tez bilgileri alınır ve rapor hazırlanır. Eğer öğrenci kayıt dondurmuşsa bunun bilgisi de bu raporda yer alır. Aşağıda İşletme Yüksek Lisans Programı için örnek bir rapor görülmektedir.

İşletme Yüksek Lisans Programı Öğrenci Ders ve Tez Bilgileri

Adı ve Soyadı	Dönemi	Aldığı Kredi	Almakta Olduğu Kredi	Tez Durumu	Kayıt Dondurma
Uğur ÖZTAŞ	1	0	12	Yok	-
Mehmet BOZDEMİR	1	0	12	Yok	-
Nilgün ÇELİK	4	24	0	YL Tez	-
Serdar YAMAN	2	12	15	Yok	-
Fatma Bozdağ	1	0	12	Yok	-

Şekil 3.37 Öğrenci Ders ve Tez Bilgileri Ekran Çıktısı Örneği

2.16. Yönetici İşlemleri

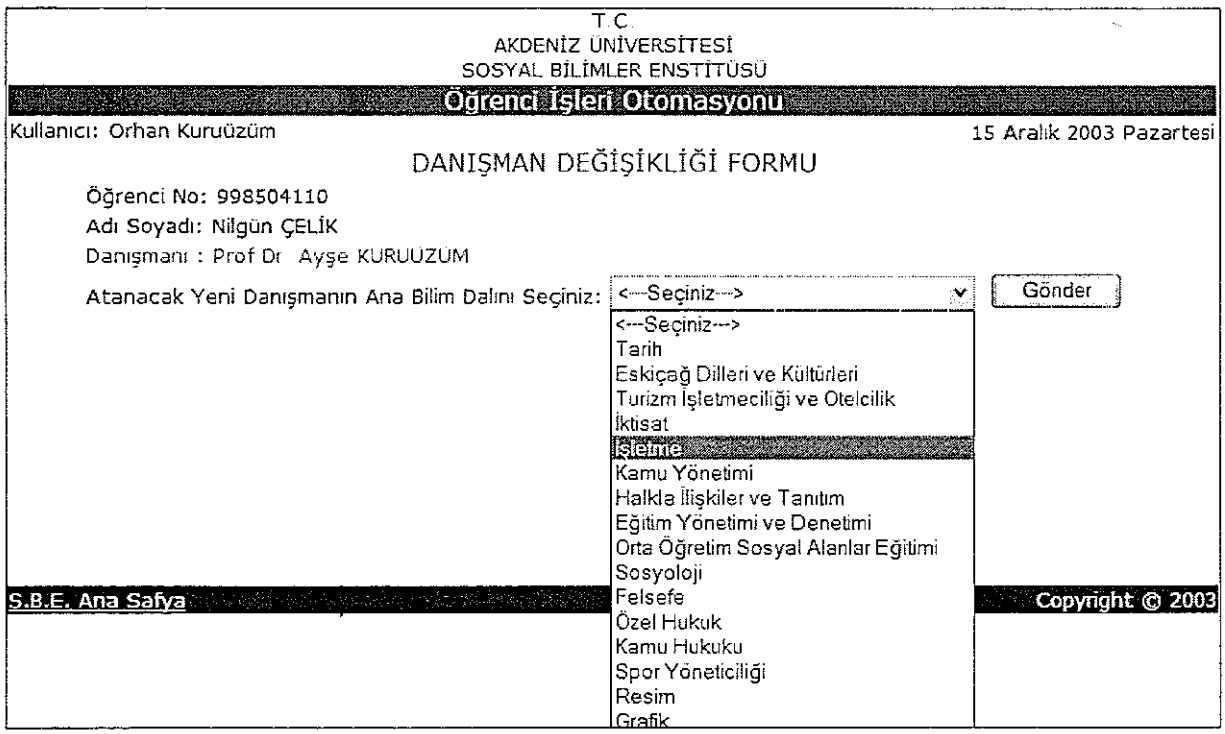
Yönetici işlemleri bölümü sadece yönetici özelliği olan kullanıcılar için aktif olan bir bölüm olarak tasarlanmıştır. Bunun amacı yazılımın çalışmasını düzenleyen veya enstitü öğrenci işlerinin kritik işlemlerini herhangi bir kullanıcının erişiminden uzak tutmaktır. Programın başlangıcında kullanıcı girişi yapılırken KULLANICILAR tablosundan giriş yapan

kullanıcının yönetici (ADMIN) olup olmadığı kontrol edilir. Eğer kullanıcının yönetici özelliği varsa açılan oturum için Yönetici İşlemleri aktif olur.

Bu bölümde danışman değişikliği, kayıt dondurma, ders bırakma, bir derse başka bir dersi eşdeğer olarak atama ve aktif olan dönemin değiştirilmesi fonksiyonları yer almaktadır.

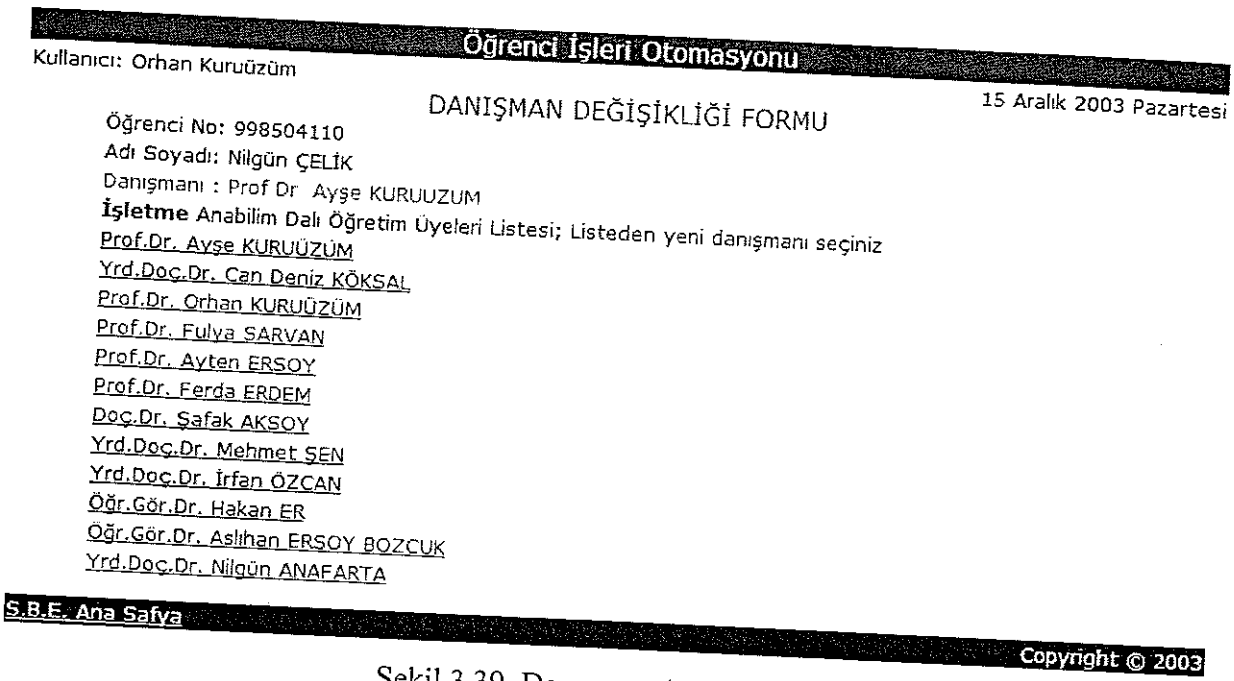
2.16.1. Danışman Değişikliği

Bir öğrencinin danışmanının değiştirilmesi için danışman değişikliği formu kullanılır. Öğrenci numarası girildikten sonra veritabanından öğrenci ve danışmanına ait bilgiler alınır ve aşağıdaki form ekrana gelir. Burada atanacak yeni danışmanın anabilim dalının seçilmesi istenir. Bunun amacı öğrenciye başka bir anabilim dalından veya enstitü dışından bir danışman atanmasına olanak tanımaktır.



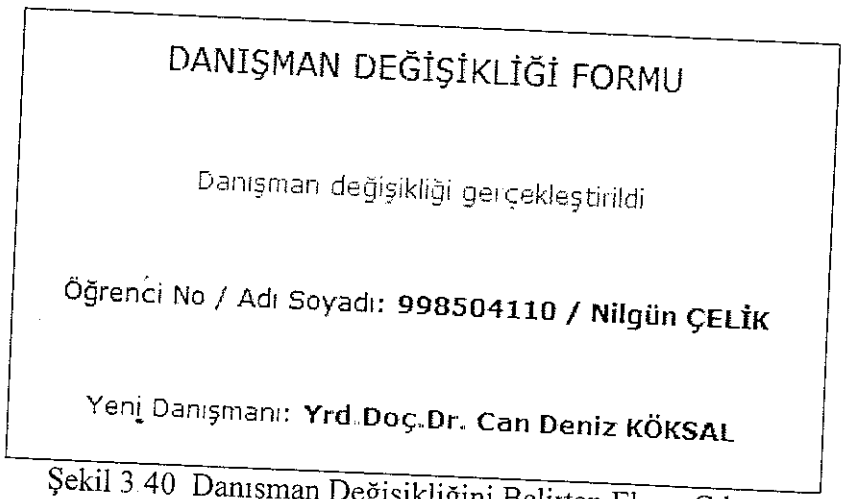
Şekil 3.38 Danışman Değişikliği Formu

Atanacak danışmanın anabilim dalı seçildikten sonra bu anabilim dalındaki öğretim üyeleri aynı form içerisinde listelenir ve kullanıcının öğretim üyelerinden birisini seçmesi istenir.



Şekil 3.39. Danışman Atama Listesi

Öğrencinin yeni danışmanını seçmek için listedeki öğretim üyesinin üzerinde tıklanması yeterli olacaktır. Bu bilgi aynı form içerisinde OUID=[öğretim üyesi kimlik no] querystring olarak gönderilir ve veritabanında gerekli alan SQL UPDATE komutu ile güncellenir. Eğer yukarıda örneği görülen listede Yrd. Doç. Dr. Can Deniz KÖKSAL seçilirse OGRENCILER tablosunda DANISMANI alanı seçilen öğretim üyesinin kimlik numarası (OU_ID) olarak UPDATE edilir ve aşağıdaki gibi bir ekran çıktısı görüntülenir

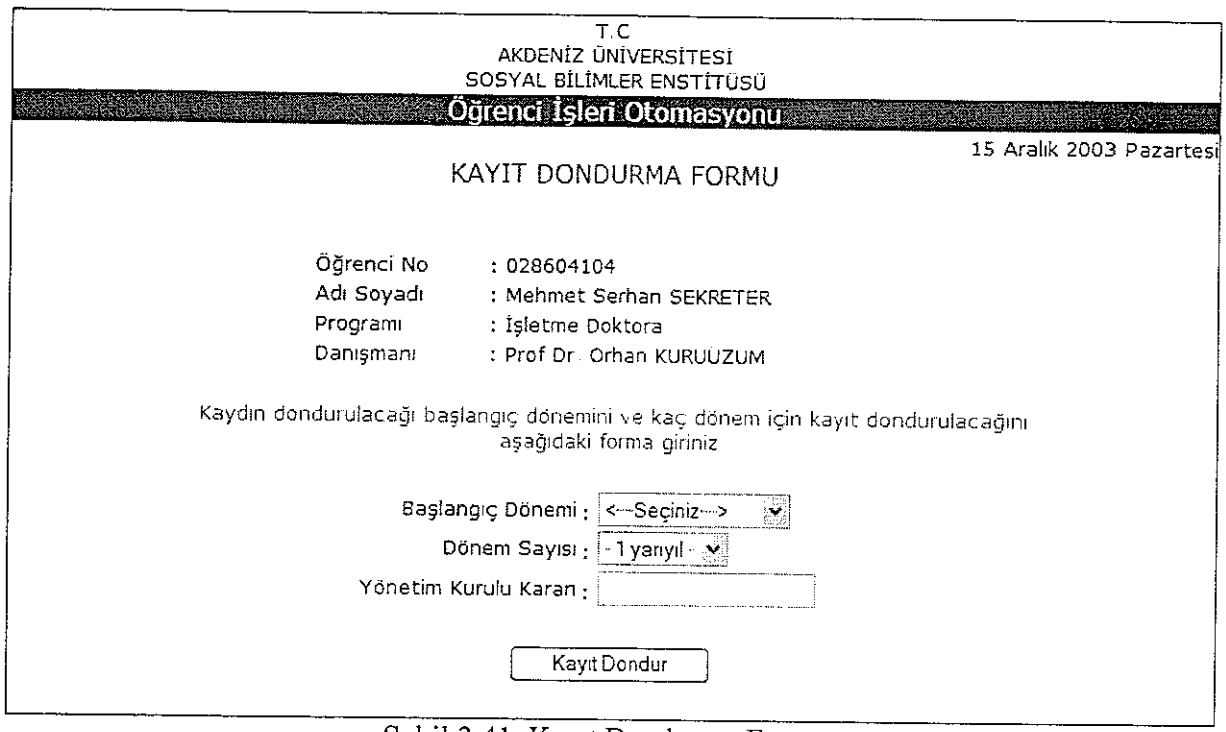


Şekil 3.40 Danışman Değişikliğini Belirten Ekran Çıktısı

2.16.2. Kayıt Dondurma

Kayıt dondurma işlemini gerçekleştirmek için öğrencinin numarası girilir ve öğrenci bilgileri veritabanından alınarak aşağıdaki form ekrana getirilir. Burada öğrencinin hangi dönemden itibaren kaç yarıyıl kayıt donduracağı ve bunun kabul edildiği yönetim kurulu

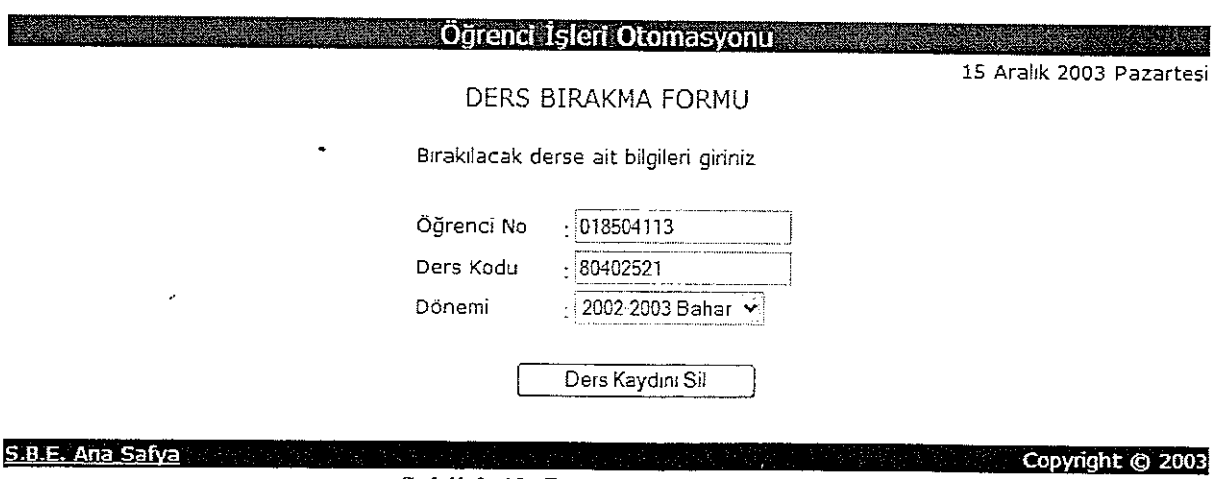
kararı bilgisinin girilmesi istenir. Bu bilgiler girildikten sonra veritabanında KAYIT_DONDURMA tablosu güncellenir.



Şekil 3.41 Kayıt Dondurma Formu

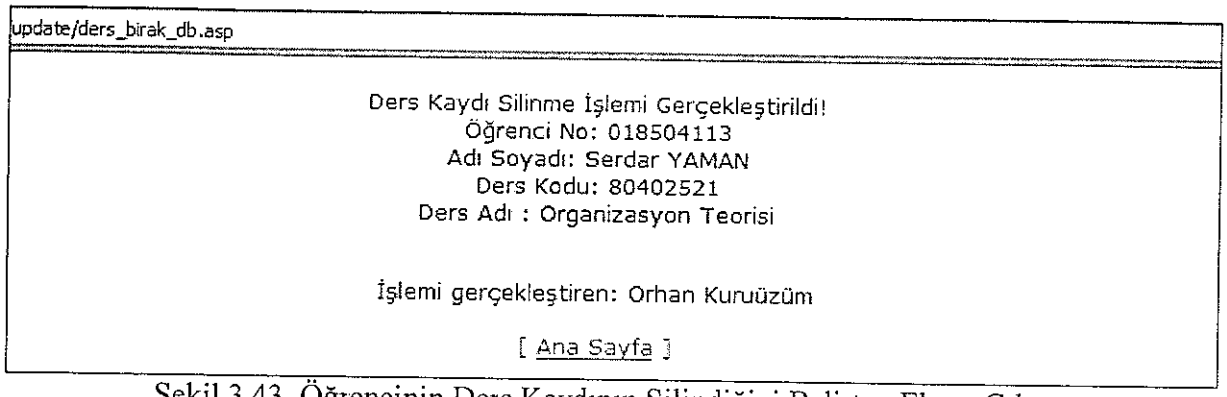
2.16.3. Ders Bırakma

Ders bırakma işlemi bir öğrencinin önceden bir derse kayıt yaptırmış olması ve sonradan bu dersten kaydını sildirmek istemesi durumunda kullanılır. Öğrencinin dersi bırakabilmesi için onay verildikten sonra programda ders bırakma formu kullanılarak öğrencinin dersteki kaydı silinir. Bunun için aşağıda görülen forma öğrenci numarası, ilgili dersin kodu ve doğabilecek herhangi bir hataya karşı dersin alındığı dönemin seçilmesi istenir. Eğer doğru dönem seçilmezse öğrencinin ders kaydı silinmez.



Şekil 3.42. Ders Bırakma Formu

Doğru bilgi girildikten sonra öğrencinin seçilen dönemde belirtilen dersteki kaydı silinir ve aşağıdaki ekran raporu görüntülenir.



Şekil 3.43. Öğrencinin Ders Kaydının Silindiğini Belirten Ekran Çıktısı

Burada hem DERS_KAYIT tablosunda hem de AKTIF_DERSLER tablosunda değişiklikler yapılması gereklidir. DERS_KAYIT tablosundan ilgili kayıtlar SQL DELETE komutu ile silinir.

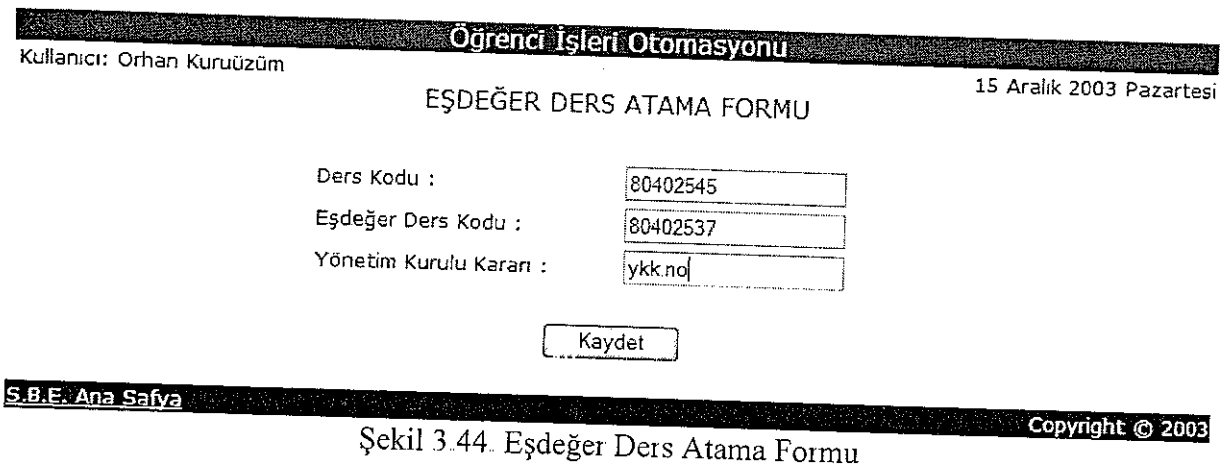
```
DKsql="DELETE FROM DERS_KAYIT "&_
WHERE O_ID="&RSOGR("O_ID")&" AND D_ID="&RSDERS("D_ID")&" AND "&_
DONEM_ID="&DONID&""
conn.Execute(DKSQL)
```

AKTIF_DERSLER tablosunda ise eğer o dersi sadece o öğrenci seçmişse DELETE ile ilgili kayıt silinir. Eğer dersi seçen başka öğrenciler de varsa UPDATE ile dersi alan öğrenci sayısı bir azaltılır. Bunu gerçekleştirmek için aşağıdaki yazılım kodları kullanılır.

```
ADsql="SELECT * FROM AKTIF_DERSLER "&_
WHERE D_ID="&RSDERS("D_ID")&" AND DONEM_ID="&DONID&""
set rsAD=conn.Execute(ADsql)

if rsAD("OGRENCI_SAYISI")=1 then
        ADSILsql="DELETE FROM AKTIF_DERSLER "&_
        WHERE D_ID="&RSDERS("D_ID")&" AND DONEM_ID="&DONID&""
        conn.Execute(ADSILsql)
else
        OGRSAY=rsAD("OGRENCI_SAYISI")-1
        ADUPsql="UPDATE AKTIF_DERSLER SET OGRENCI_SAYISI="&OGRSAY&" "&_
        WHERE D_ID="&RSDERS("D_ID")&" AND DONEM_ID="&DONID&""
        conn.Execute(ADUPsql)
end if
```

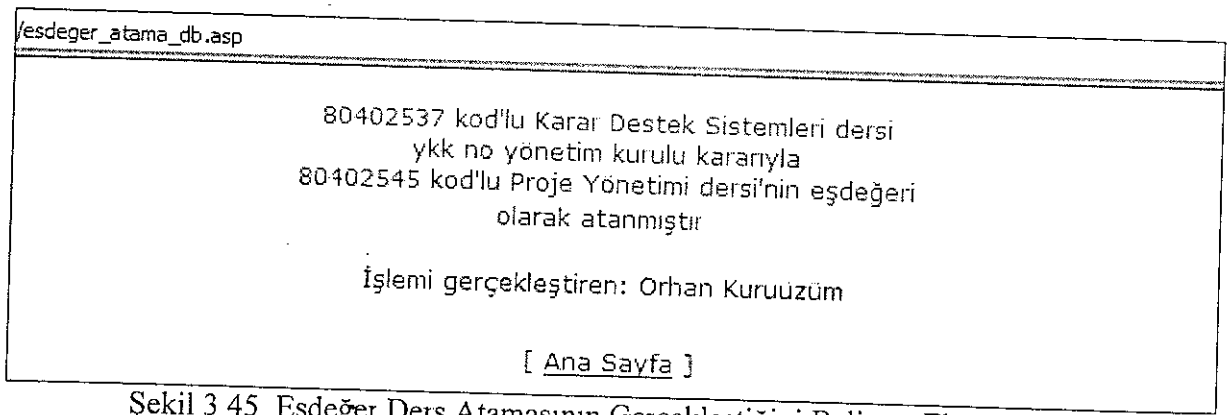
2.16.4. Eşdeğer Ders Atama

Eşdeğer ders atama fonksiyonu Enstitü tarafından birbirinin eşdeğeri olarak kabul edilmiş derslerin veritabanı yönetim sisteminde de eşdeğer olarak görülerek işlem yapılabilmesi için ESDEGER_DERSLER tablosuna bu bilgilerin girilmesi amacıyla kullanılır.



Şekil 3.44. Eşdeğer Ders Atama Formu

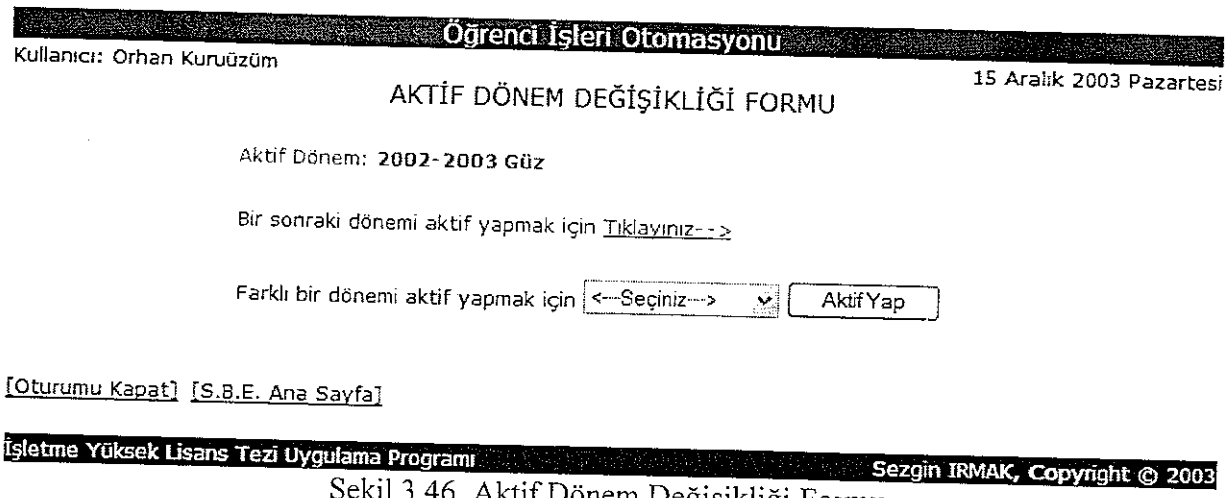
Eşdeğer ders atama formunda dersin kodu, bu derse eşdeğer olan dersin kodu ve ilgili yönetim kurulu kararı girilerek esdeger_atama_db.asp dosyasına gönderilir. Bu yazılım girilen bilgileri veritabanına kaydeder ve aşağıdaki gibi bir ekran çıktısı görüntülenir. Bu ekran çıktılarında işlemi gerçekleştiren kullanıcı da belirtilmektedir.



Şekil 3.45 Eşdeğer Ders Atamasının Gerçekleştiğini Belirten Ekran Çıktısı

2.16.5. Aktif Dönem Değişikliği İşlemi

Aktif dönem burada kullanılan yazılım için en kilit özelliği olan veridir. Bütün veritabanı işlemleri içinde bulunulan döneme, dolayısıyla DONEMLER tablosunda seçili aktif döneme göre yapılmaktadır. Yönetici olan bir kullanıcının aktif dönemi değiştirme hakkı bulunmaktadır.



Şekil 3 46. Aktif Dönem Değişikliği Formu

Bu formda program için seçili aktif dönem görüntülenmekte ve kullanıcının hangi dönemi aktif yapmak istediği sorulmaktadır. Aktif dönemi değiştirmek için birinci yol belirtilen linke tıklanarak bir sonraki dönemin aktif yapılmasıdır. Burada veri action=birsonraki querystring'i ve dönem kimlik numarası (DONEM_ID) da querystring olarak gönderilmektedir. İkinci yol ise açılır menüden istenilen dönemin seçilmesidir. Burada action=degisken querystring olarak DONEM_ID ise form verisi olarak gönderilir.

SONUÇ

Günümüzde artık bilgi teknolojilerinin hayatın her alanında söz sahibi olduğunu söylemek mümkündür. Birçok organizasyon ihtiyaçları oranında çeşitli büyüklüklerde bilişim sistemlerine sahiptir ve bilişim sistemlerinin iş süreçlerini geleneksel yollar kullanılarak yapılamayacak boyutta etkinleştirdiği söylenebilir.

Bu çalışmada Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü öğrenci işleri süreçleri için bir ilişkisel veritabanı yönetim sistemi tasarlanmıştır. Enstitü kurulduğu 1992 yılından bu yana bünyesindeki anabilim dalı sayısı, bu anabilim dallarının açtığı yüksek lisans ve doktora programlarının sayısı ve dolayısıyla öğrenci, öğretim üyesi ve ders sayısı bugüne kadar sürekli artmıştır. Az sayıda personel ve geleneksel yöntemlerle yürütülen işlerin, bir veritabanı yönetim sistemi uygulaması ile etkinleştirilmesi planlanmıştır.

Veritabanlarının ve veritabanı yönetimi yazılımlarının geliştirilmesi organizasyonel verinin yönetilmesinde modern yöntemlerin temelini oluşturmaktadır. Veritabanı yönetimi yaklaşımı birçok farklı uygulama programı, sistem veya bölüm tarafından erişilebilecek veriyi bir veritabanında birleştirmekte olup veritabanı yönetim sistemi (VTYS) ile işlem yapılması veri bağımsızlığı sağlamaktadır. Bu yapı gereğinden fazla veri tutulması engellemekte ve ilişkisel veritabanı modeli tasarımcının fiziksel depolama detayları yerine verinin mantıksal sunumuna odaklanmasını mümkün kılmaktadır.

İlişkisel veritabanları farklı tablolarda depolanan ayrı veri birimlerini hızla alabilmekte ve bunları kullanıcıya veya uygulama programına sonuç olarak adlandırdığımız bütünleşik veri topluluğu şeklinde getirebilmektedirler. İlişkisel veritabanlarına bilgi girilmesi ve seçilen kayıtların getirilmesi amacıyla tasarlanmış en önemli ve yaygın olarak kullanılan dil kısaca SQL olarak adlandırılan yapısal sorgulama dilidir. SQL bütün ifadeleri ilişkisel veritabanlarına yönelik komutlar olan bir alt dildir ve kullanıcının veri ile mantıksal seviyede çalışmasını sağlamaktadır.

Akdeniz Üniversitesi S.B.E. Öğrenci İşleri Otomasyonu uygulamasında veritabanı olarak Microsoft Access 2002, sürüm 10.0, uygulama programının geliştirilmesinde platform olarak Microsoft Visual InterDev 6.0, programlama dili olarak ASP, veri tanımlama ve veri işlem dili olarak SQL, ayrıca form tasarımları ve istemci tarafı işlemleri için HTML ve Java

Script kullanılmıştır. Uygulama programı web tabanlı olup, Microsoft Internet Information Services (IIS) sunucusu altında çalışmaktadır.

Uygulama yazılımı web tabanlı olduğu için herhangi bir istemci tarafı yazılımına ihtiyaç duyulmadan bir web tarayıcısı kullanılarak programa erişilebilmektedir. Güvenlik sorunları dikkate alınarak program başlangıçta bir intranet uygulaması olarak tasarlanmıştır. Ancak güvenlik sağlayıcı önlemler alındığı taktirde internete de açılmaya uygun bir şekilde tasarlanmıştır.

Programın uygulanması aşamasında özellikle veri girişleri önemli bir sorun oluşturmuştur. Sisteme, yeni öğrencilerle birlikte, halen kayıtlı olan 34 farklı programdan ve farklı aşamalarda bulunan öğrencilerin de girecek olması geçmişe dönük verilerin de veritabanına girilmesi zorunluluğunu beraberinde getirmiştir. Enstitü yönetimi dışında çalışanlarının bilgi teknolojileri konusunda yeterliliğinin az olması ve yığın verilerin sistematik girişinin büyük bir iş yükü getirmesi nedeniyle veri girişi işlemleri oldukça yavaş gerçekleşmiştir.

S.B.E. veritabanı yönetim sistemi uygulaması öğrenci, öğretim üyesi, anabilim dalı, yüksek lisans ve doktora programı ve ders bilgilerinin ilişkisel model doğrultusunda veritabanında sistematik olarak depolanmasını, ayrıca belge verilmesi işlemlerini, öğretim üyelerinin ders ve danışmanlık yüklerinin raporlanmasını, öğrencilerin ders ve tez durumlarının raporlanmasını ve özellikle ders kaydından yıl sonu sonuçlarının girilmesine kadar olan ders işlemleri süreçlerini etkinleştirmektedir.

KAYNAKÇA

Bertino, E., Martino, L., Object-Oriented Database Systems: Concepts and Architectures, Addison-Wesley 1994

Clemons, E. K., Weber, B. W., Segmentation, Differentiation, and Flexible Pricing: Experiences with Information Technology and Segment-Tailored Strategies, Journal of Management Information Systems Vol. 11 No. 2, Fall 1994

Codd, E. F., A Relational Model of Data for Large Shared Data Banks, Communications of the ACM, Association for Computing Machinery Inc., June 1970

Dietrich, S. W., Understanding Relational Database Query Languages, Prentice Hall, Upper Saddle River, 2001

Esen, H. Ö., İşletme Yönetiminde Sistem Yaklaşımı, 2. Baskı, İstanbul Üniversitesi İşletme Fakültesi Yayınları

Fuh, G. Y., Chow, J. H., Mattos, N. M., Tran, B. T., Truong, T. C., On the Linkage of Dynamic Tables in Relational DBMSs, IBM Systems Journal, Vol 37. No 4., 1998

Gomolski, B. C., Users Loathe to Share Their Know-How, Computerworld, Computerworld Inc., Framingham, MA, November 17, 1997

Haag, S., Cummings, M., McCubbrey, D. J., Management Information Systems for the Information Age 3/E, Irwin/McGraw-Hill, 2002

Hayes, D. C., Hunton, J. E., When Querying Databases You've Got to Ask the Right Question, Journal of Accountancy, February 2001

Hoske, M. T., How to Get the most from a Database, Control Engineering, June 2002

http://www techweb.com/encyclopedia/defineterm?term=record (24/10/2003 10:40), Tech Encyclopedia, CMP Media Inc., 1998.

http://www.techweb.com/encyclopedia/defineterm?term=database (24/10/2003 10:48), Tech Encyclopedia, CMP Media Inc., 1998.

Hutchinson, S. E., Sawyer, S. C., Computers, Communications and Information7/E, Irwin/McGraw-Hill, 2000

Jacobson, I., Christerson, M., Jonsson, P., Övergaard, G., Object-Oriented Software Engineering, Addison-Wesley, 1994

Jacobsen, I., Ericsson, M., Jacobsen, A., The Object Advantage: Business Process Reengineering with Object Technology, New York: ACM Pres, 1995

Jones, M. D., Provide Dynamic Data Extraction with SQL Statements, Inside Microsoft Visual Basic, July 2001

Karahoca, D., Karahoca, A., Yönetim Bilişim Sistemleri ve Uygulamaları, Beta Basım Yayım Dağıtım A.Ş., 1998

Larson, J. A., Database Directories, Upper Saddle River, NJ: Prentice Hall PTR, 1995

Laudon, K. C., Laudon, J. P., Essentials of Management Information Systems: Organization and Technology 2/E, Prentice Hall, 1997

McGeever, C., Structured Query Language, Computerworld, May 15, 2000

Morrison, M., Morrison, J., Database-Driven Web Sites, Thomson Learning Web Warrior Series, 2000

O'Brien, J. A., Introduction to Information Systems 8/E, Irwin/McGraw-Hill, 1997

O'Brien, J. A., Introduction to Information Systems: Essentials of the Internetworked E-Business Enterprise, Irwin/McGraw-Hill, 2001

Oracle9i SQL Reference Manual Release 1 (9.0.1), Oracle Corporation, RedWood City, CA, 2001

Rob, P., Coronel, C., Database Systems: Design, Implementation and Management 5/E, Course Technology/Thomson Learning, Boston, Massachusetts, 2002

Ronald, P., Anjard, Sr , The Basics of Database Management Systems (DBMS), Industrial Management & Data Systems, Vol. 94 No. 5 MCB University Pres Limited, 1994

Senn J. A , Analysis and Design of Information Systems, McGraw-Hill Publishing Company, Singapore, 1989

Shelly, G. B., Cashman, T J., Rosenblatt, H J , Systems Analysis and Design 4/E, Course Technology/Thomson Learning, 2001

Tribunella, T., Designing Relational Database Systems, The CPA Journal, July 2002

Wang, G , The Programmer's Job Handbook, New York: McGraw-Hill, 1996

Watson, R T., Data Management: Databases and Organizations 3/E, John Willey & Sons, 2002

ÖZGEÇMİŞ

Adı ve SOYADI	: Sezgin IRMAK
Doğum Yeri ve Tarihi	: Serik, 04 01 1978
Medeni Durumu	: Bekar

Eğitim Durumu

Mezun Olduğu Lise	: Antalya Teknik Lisesi, Bilgisayar Bölümü
Lisans Diploması	: Marmara Üniversitesi, Elektronik ve Bilgisayar Öğretmenliği Bölümü (İngilizce)
Tez Konusu	: İlişkisel Veritabanı Yönetim Sistemleri ve Akdeniz Üniversitesi Sosyal Bilimler Enstitüsü Öğrenci İşleri Otomasyonu Uygulaması

İş Durumu

2001 –	: Akdeniz Üniversitesi S.B.E., Araştırma Görevlisi, Antalya
Adres	: Akdeniz Üniversitesi, İ.İ.B.F. İşletme Bölümü, Kampüs ANTALYA
Telefon	: 0 242 310 18 45