T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANI TASARIMI VE BİR E–DEVLET PROJESİ KAPSAMINDA WEB-TABANLI CBS UYGULAMASI

Yalkın ÇAĞLAR

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

KONYA, 2006

T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANLARININ OPTİMİZASYONU VE BİR E–DEVLET PROJESİ KAPSAMINDA WEB-TABANLI CBS UYGULAMASI

Yalkın ÇAĞLAR

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Danışman
Prof. Dr. Ferruh YILDIZ

KONYA, 2006

T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİTABANLARININ OPTİMİZASYONU VE BİR E–DEVLET PROJESİ KAPSAMINDA WEB-TABANLI CBS UYGULAMASI

Yalkın ÇAĞLAR

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Bu tez 03 / 03 / 2006 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir

Danışman

Prof. Dr. Ferruh YILDIZ

Üye

Prof. Dr. Ahmet ARSLAN

Üye

Doç. Dr. Şirzat KAHRAMANLI

ÖZET

Yüksek Lisans Tezi

VERİTABANLARININ OPTİMİZASYONU VE BİR E–DEVLET PROJESİ KAPSAMINDA WEB-TABANLI CBS UYGULAMASI

Yalkın ÇAĞLAR
Selçuk Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Bölümü

Danışman: Prof. Dr. Ferruh YILDIZ

2006, 65 Sayfa

Jüri : Prof. Dr. Ahmet ARSLAN
Doç. Dr. Şirzat KAHRAMANLI

Bu çalışmada, jeodezik verilerin organizasyonu için yapılması gerekenler incelenmiştir. Bu bağlamda, Jeodezik Bilgi Sisteminin (JBS) çözümlenmesinde yapılan anketlerle oluşan Veri Akış Diyagramları (VAD) örneklenmiştir. Bunlar yardımıyla yüksek – seviye bir veri modeli olan Varlık – İlişki (Vİ) diyagramları oluşturulmuştur. Veri Tabanı Yönetim Sistemi (VTYS) olarak MS Access 2000 yazılımının seçiminden sonra, veri modeli – tablo dönüşümü yapılmış ve fiziksel veri tabanı kurulmuştur. Bu tablolar gereksinim çözümlemelerine göre indekslenerek sanal verilerle doldurulmuştur. Son adım olarak da bu verilerin e-ticaretine olanak sağlayan bir site hazırlanmıştır.

Anahtar Kelimeler: Jeodezik Bilgi Sistemi, Veri Modeli, Veri Tabanı Yönetim Sistemi, Veri Akış Diyagramları, Varlık – İlişki Diyagramları, e-Devlet, web-tabanlı CBS

ABSTRACT

MSc Thesis

OPTIMIZATION OF DATABASES AND A WEB-BASED GIS APPLICATION IN THE SCOPE OF AN E–GOVERNMENT PROJECT

Yalkın ÇAĞLAR
Selçuk University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Prof. Dr. Ferruh YILDIZ

2006, 65 Pages

Jury : Prof. Dr. Ahmet ARSLAN
Doç. Dr. Şirzat KAHRAMANLI

In this study, tasks to accomplish the organization of the geodetic data are held. Hence, data flow diagrams derived from interviews and used in analyzing Geodetic Information System (GdetIS) are exemplified. By means of them Entity – Relation (ER) diagrams (a high – level data model) are derived. After choosing MS Access 2000 as DataBase Management System (DBMS), ER to Relational Schema mapping is carried out to construct the physical database. According to results of the requirement analysis these tables are indexed and loaded by virtual data. As a last step a web site enabling the e-commerce of these data is prepared.

Key Words : Geodetic Database System, Data Model, Database Management System, Data Flow Diagrams, Entity-Relationship Diagrams, e-government, web-based GIS

ÖNSÖZ

"Veritabanı Tasarımı ve Bir e–Devlet Projesi Kapsamında Web-Tabanlı CBS Uygulaması" konulu bu çalışmada, tez yürütücülüğümü üstlenen, tezimin her aşamasında bana bilgi ve deneyimleriyle yardımcı olan sayın hocam Prof. Dr. Ferruh YILDIZ'a, ayrıca çalışmam sırasında sürekli yanımda olup her türlü desteğini benden esirgemeyen, yaptığı dilbilgisi kontrolleri ile tezimin daha anlaşılır hale gelmesini sağlayan sevgili eşim Serap'a ve çalışmalarım sırasında babalık görevimi ihmal etmeme rağmen bana karşı sabırlı ve anlayışlı olan kızım İlayda'ya ve oğlum Arda'ya teşekkür ederim.

Mart 2006 Yalkın ÇAĞLAR

KISALTMALAR LİSTESİ

4GL	: Fourth Generation Language
BS	: Bilgi Sistemi
BKİ	: Bilgi Kaynağı İdaresi
CBS	: Coğrafi Bilgi Sistemleri
CPU	: Central Processing Unit
CVT	: Coğrafi Veri Tabanı
DDL	: Data Definition Language
DEC	: Digital Equipment Corporation
DML	: Data Manipulation Language
DVTYS	: Dağıtık Veri Tabanı Yönetim Sistemi
f.k.	: foreign key
GVİ	: Gelişmiş Varlık – İlişki
JBS	: Jeodezik Bilgi Sistemi
SDL	: Storage Definition Language
SGİÇ	: Sıradüzensel Girdi İşlem Çıktı
STD / STÇ	: Sorun Tanımlama Dili / Sorun Tanımlama Çözümleyicisi
VAD	: Veri Akış Diyagramları
vd	: ve diğerleri
Vİ	: Varlık – İlişki
VTS	: Veri Tabanı Sistemi
VTY	: Veri Tabanı Yöneticisi
VTYS	: Veri Tabanı Yönetim Sistemi
YÇTY	: Yapısal Çözümleme ve Tasarım Tekniği

ŞEKİL LİSTESİ

Sayfa

Şekil 2-1 Büyük veritabanları için veritabanı tasarım aşamaları 5
Şekil 2-2 Bilgi akış modeli..... 9
Şekil 2-3 Bilgi akış iyileştirmesi 10
Şekil 2-4 Sıradüzensel veri modeli 13
Şekil 2-5 Ağ veri modeli 13
Şekil 2-6 İlişkisel veri modeli 14
Şekil 2-7 Nesneye yönelik veri modeli 14
Şekil 2-8 Varlık - İlişki gösterim işaretleri özeti..... 15
Şekil 2-9 Varlık – İlişki veri modeli 16
Şekil 2-10 Yukardan – Aşağı iyileştirme (Yeni bir varlık türü türetme) 18
Şekil 2-11 Yukardan – Aşağı iyileştirme (Varlık türünü ayrıştırma) 19
Şekil 2-12 Aşağıdan – Yukarı iyileştirme (Yeni bir ilişkinin eklenmesi)..... 19
Şekil 2-13 Birleştirmeden önce uyuşum için görünümlerin düzeltilmesi 21
Şekil 2-14 Görünüm birleştirmesi için farklı stratejiler 23
Şekil 3-1 JVT Veri akış diyagramları (nirengi) 32
Şekil 3-2 Bütünlük koşullarını içeren bir ilişkisel veritabanı şeması..... 35
Şekil 3-3a~m “Nirengi” tablosu tasarım görünümü (kurumsal) 36
Şekil 3-4a ~eTablolar (internet) 42
Şekil 3-5 TUTGA sayfası (ziyaretçi) 45
Şekil 3-6 Üyelik bilgi girişi..... 46
Şekil 3-7 Üye Girişi 46
Şekil 3-8 TUTGA Sayfası (kullanıcı tanımlı)..... 47
Şekil 3-9 TUTGA Nokta Seçimi..... 48
Şekil 3-10 TUTGA Nokta Bilgisi 48
Şekil 3-11 Nirengi Sayfası (kullanıcı tanımlı) 49
Şekil 3-12 Nirengi Nokta Seçimi 49
Şekil 3-13 Nirengi Nokta Bilgisi..... 50
Şekil 3-14 Nivelman Sayfası (kullanıcı tanımlı)..... 50
Şekil 3-15 Nivelman Nokta Seçimi 51
Şekil 3-16 Nivelman Nokta Bilgisi 51

İÇİNDEKİLER

ÖZET i
ABSTRACT ii
ÖNSÖZ iii
KISALTMALAR LİSTESİ iv
ŞEKİL LİSTESİ v
İÇİNDEKİLER vi
1. GİRİŞ 1
1.1 Bilgi Sistemi Hayat Çarkı 2
1.2 Veritabanı Uygulama Sistemi Hayat Çarkı 3
2. VERİTABANI TASARIM SÜRECİ 4
2.1 Aşama 1: Gereksinimlerin Derlenmesi ve Çözümlenmesi 7
2.1.1 Veri akış diyagramları 8
2.2 Aşama 2: Kavramsal Veritabanı Tasarımı 10
2.2.1 Aşama 2a: Kavramsal şema tasarımı 11
2.2.1.1 Veri modelleri 12
2.2.1.2 Kavramsal şema tasarımına yaklaşımlar 16
2.2.1.3 Şema tasarımı için stratejiler 17
2.2.1.4 Şema (görünüm) birleştirme 20
2.2.2 Aşama 2b: Veritabanı işlemi tasarımı 22
2.3 Aşama 3: Veritabanı Yönetim Sisteminin Seçimi 24
2.4 Aşama 4: Veri Modeli Dönüşümü 27
2.5 Aşama 5: Fiziksel Veritabanı Tasarımı 28
2.6 Aşama 6: Veritabanı Sisteminin Uygulanması 29
3. UYGULAMA (JEODEZİK VERİTABANI) 30
3.1 Aşama 1: Gereksinimlerin Derlenmesi ve Çözümlenmesi 30
3.2 Aşama 2: Kavramsal Veritabanı Tasarımı 31
3.3 Aşama 3: Veritabanı Yönetim Sisteminin Seçimi 33
3.4 Aşama 4: Veri Modeli Dönüşümü 33
3.5 Aşama 5: Fiziksel Veritabanı Tasarımı 36
3.6 Aşama 6: Veritabanı Sisteminin Uygulanması 45

4. SONUÇ 52
KAYNAKLAR 54
EKLER 57

1. GİRİŞ

Günümüzün teknolojik harikası olan bilgisayarın yalnızca bir hesaplama aracı olmayıp bir bilgi işlem ve bir bilgi iletişim aracı olduğu bir gerçektir. Genelde bilgi işlem olarak adlandırılan etkinliklerin hammaddesinin VERİ, ürününün BİLGİ (işlenmiş veri) olduğu da göz önüne alınırsa veri ve bilgilerin organize edilmesi sorunu ön plana çıkmaktadır. Nitekim pek çok projenin geliştirildiği ve uygulandığı büyük kuruluşlarda her projede pek çok yazılım ve her yazılımın kullandığı pek çok veri dosyaları mevcuttur. Veri ve bilgilerin depolandığı bu dosyalar her yazılımın kendi mantıksal yapısına göre şekillenmiştir. Ancak bu projeler ve yazılımlar birbirleriyle ilişkili ve bütünler nitelikte olabileceklerinden kullanılan veri dosyaları de ilişkili ve ortak olması gerekmektedir (Fry, J. ve Sibley, E., 1976). Bu dosyaların içeriğinin büyük bir bölümünün aynı olması ve birden fazla yerde saklanması yer ve performans kaybına neden olmaktadır. Öte yandan dosyalar işlendikçe (ekleme, silme, güncelleme vb.) aynı verileri içeren dosyalar arasında tutarsızlıklar ortaya çıkmaktadır. Kullanılan veri kümesinin çok yoğun olması durumunda bu tutarsızlıkların daha büyük boyutlarda olacağı açıktır. İşte Bilgi Sistemi (BS) ve Veritabanı Sistemi (VTS) bu tür sorunları gidermek üzere geliştirilmiştir. Veritabanı teknolojisi bir veri organizasyonunun bütün olarak işlenmesini ve farklı uygulamalar için kullanıcıların verilere daha ekonomik bir biçimde ulaşmasını sağlar (Date, C., 1990).

Coğrafi varlıklara ait jeodezik verilerin toplanması arazide yapılan ölçümlerle olur. Bu değerler doğrulanır, işlenir ve bir bütün olarak organize edilerek çok çeşitli amaçlar için kullanılır. Jeodezik Veritabanı (JVT)'nin temelini teşkil eden bu verilerin elde edilmesi uzun zaman aldığı gibi önemli bir harcamayı da gerektirmektedir (Korth, H. ve Silberschartz, A., 1991). Büyük bir ekonomik değere sahip bu verilerin hız – ekonomi – doğruluk optimizasyonu ile yönetilmelerinin gereği de kendiliğinden ortaya çıkmaktadır. Bu çalışmada bu tür büyük veri yığınlarının, bir veri yönetimi tekniği ile nasıl yönetileceği ve kullanıcılara sunulan hizmetin nasıl otomatize edileceği araştırılmaktadır.

1.1 Bilgi Sistemi Hayat Çarkı

Büyük bir kuruluştaki veritabanı sistemleri, çoğunlukla kuruluşun bilgi kaynaklarının yönetiminde kullanılan çok daha büyük bir bilgi sisteminin bir parçasıdır. Bir **bilgi sistemi**, kuruluş içerisindeki bilginin derlenmesini, yönetimini ve dağıtımını kapsayan tüm kaynakları içerir. Bilgisayarlı bir ortamda bu kaynaklar verinin kendisini, veritabanı yönetim sistemi yazılımını, bilgisayar sistemi donanımını ve depolama ortamını, veriyi kullanan ve yöneten personeli (veritabanı yöneticisi, son kullanıcılar, parametrik kullanıcılar, vb...), veriye ulaşıp güncelleyen uygulama yazılımını ve bu yazılımı hazırlayan yazılımcıları kapsar. Dolayısıyla veritabanı sistemi çok daha büyük bir örgütsel bilgi sisteminin sadece bir parçasıdır (Navathe, S. ve Kerschberg, L., 1986).

Genellikle bilgi sistemi hayat çarkına **makro hayat çarkı**, veritabanı sistemi hayat çarkına ise **mikro hayat çarkı**, adı verilir. Tipik bir **makro hayat çarkı**, şu aşamaları içerir (Elmasri, R., Larson, J. ve Navathe, S., 1986):

1. Olurluk çözümlemesi (analizi): Olası uygulama alanlarının çözümlenmesini, ön kar – maliyet çalışmalarını ve uygulamalar arası önceliklerin belirlenmesini içerir.
2. Gereksinimlerin belirlenmesi ve çözümlenmesi: Gereksinimlerin ayrıntıları, kullanıcılarla yapılan görüşmelerle derlenir.
3. Tasarım: Bu aşama iki dalda ilerler. Veritabanının tasarımı ve bu veritabanını kullanıp, işleyen uygulama sistemlerinin (yazılımlarının) tasarımı
4. Uygulama: Bilgi sistemi uygulanır, veritabanı yüklenir, veritabanı işlemleri uygulanır ve denenir.
5. Değerlendirme ve kabul testi: Sistem, kullanıcı gereksinimlerini karşılama ve başarı ölçütleri (kriterleri) anlamında değerlendirilir. Sistem, başarı ölçütlerine ve davranış şartlarına karşı denenir.
6. Kullanma: Bundan önce, eski sistemdeki kullanıcıların dönüşümü ve kullanıcı eğitimi gelebilir. Kullanma aşaması tüm sistem işlevleri kullanılabilir olduğunda başlar. Yeni gereksinim ve uygulamalar türedikçe sisteme katılıp kullanılmaya başlamadan önce, baştaki tüm aşamalardan geçer. Sistem başarısının gözlenmesi ve sistem bakımı kullanma aşamasının önemli etkinlikleridir.

1.2 Veritabanı Uygulama Sistemi Hayat Çarkı

Veritabanı Uygulama Sistemi (mikro) hayat çarkı ile ilgili etkinlikler aşağıdaki aşamaları içerir (Teorey, T., 1990):

1. Sistem tanımlaması: Veritabanı sisteminin kapsamı, kullanıcıları ve onun uygulamaları tanımlanır.
2. Tasarım: Bu aşamanın sonunda, seçilmiş veritabanı yönetim sistemi üzerinde tüm mantıksal ve fiziksel tasarım hazır hale gelmiş olur.
3. Uygulama: Bu aşama kavramsal, dış ve iç veritabanı tanımlarının yazılması, boş veritabanı dosyalarının yaratılması ve yazılımların uygulanmasını içerir.
4. Yükleme veya veri dönüşümü: Veritabanı, ya verilerin doğrudan yüklenmesi ya da hazır dosyaların veritabanı formatına dönüştürülmesi ile doldurulur.
5. Uygulama dönüşümü: Önceki sistemden kalma tüm yazılım uygulamaları yeni sisteme dönüştürülür.
6. Test ve değerlendirme: Yeni sistem denenir ve değerlendirilir.
7. Kullanım: Veritabanı sistemi ve uygulamaları kullanıma konur.
8. İzleme ve bakım: Kullanım aşaması boyunca sistem sürekli izlenir ve bakımı yapılır. Hem veri içeriğinde hem de yazılım uygulamalarında büyüme ve genişleme olabilir. Zaman zaman büyük değişiklikler ve yeniden yapılanmalar gerekebilir.

2, 3 ve 4 etkinlikleri daha büyük olan bilgi sistemin tasarım ve uygulama aşamalarının parçasıdır. Bir sonraki bölümde daha çok veritabanı tasarımını kapsayan ikinci etkinlikten bahsedilecektir. Kuruluşlardaki veritabanları tüm hayat çarkı etkinliklerinden geçerler. Dönüşüm adımları (4 ve 5) hem veritabanı hem de uygulamalar yeniyse uygulanabilir değildirler. Bir kuruluş, kurulu eski bir sistemden bir yenisine geçerken 4 ve 5 etkinlikleri en çok zaman alan aşamalardır ve bunları gerçekleştirmek için gereken çaba çoğunlukla azımsanır. Her aşamada sıkça yeni gereksinimler doğduğundan çeşitli adımlar arası geri beslemeler olur.

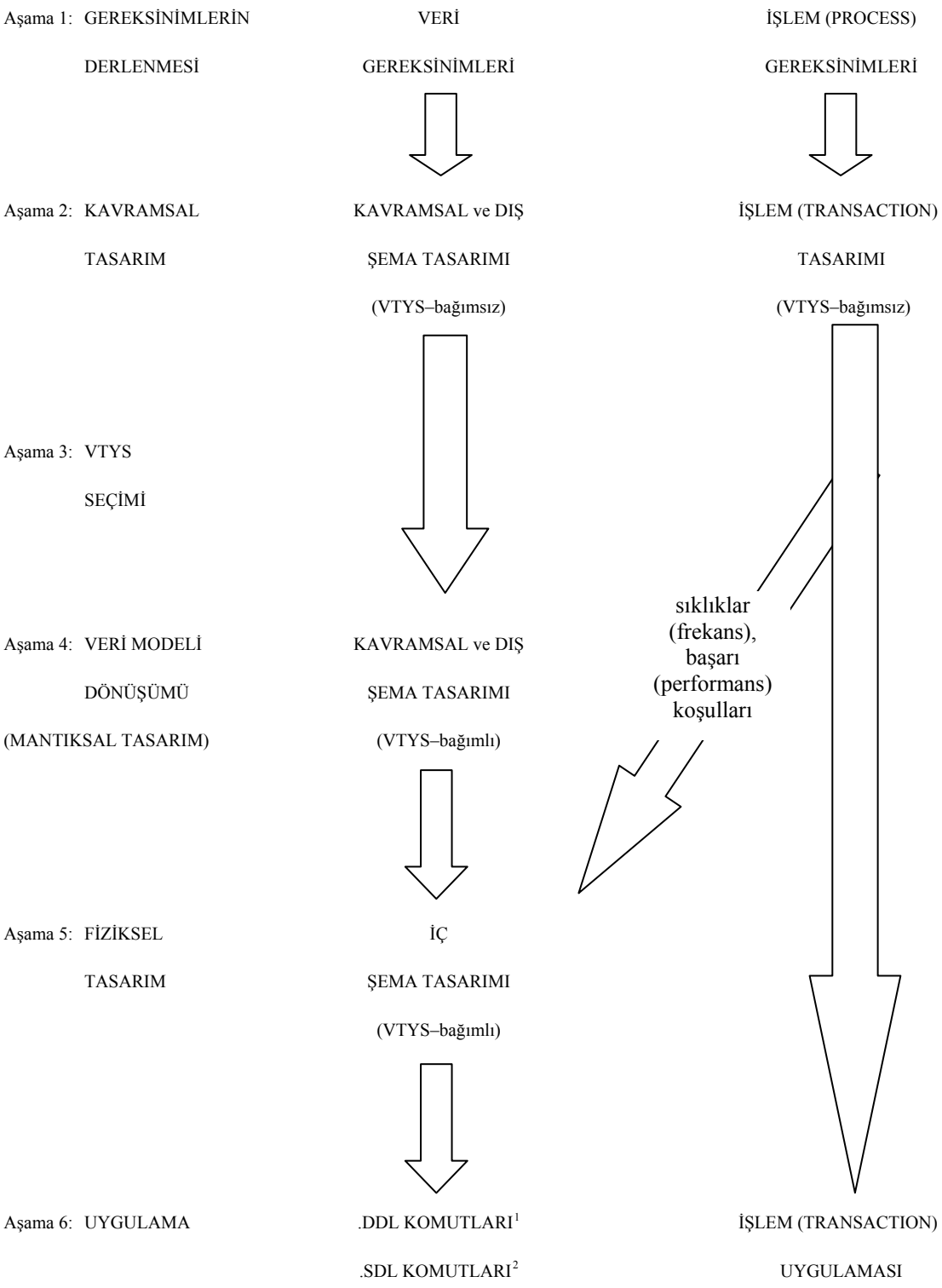
2. VERİTABANI TASARIM SÜRECİ

Veritabanı tasarım sorunu "Bir kuruluştaki kullanıcıların tanımlı bir uygulama kümesi için olan bilgi gereksinimlerini karşılamak üzere bir veya daha fazla veritabanının mantıksal ve fiziksel tasarımını yapmaktır." şeklin ifade edilebilir (Ceri, S., 1983).

Veritabanı tasarımının hedefleri çeşitlidir. Başlıca hedefler belirli kullanıcı ve uygulamaların bilgi gereksinimlerini karşılamak; doğal ve anlaması kolay bir bilgi yapılanması sağlamak; son olarak veri işleme gereksinimlerini ve cevaplama zamanı, işlem zamanı, depolama kapasitesi gibi başarı hedeflerini desteklemektir. Bunlar gerçekleştirilmesi ve ölçülmesi zor hedeflerdir. Veritabanı tasarım sürecinin yetersiz ve zayıf tanımlanmış hedeflerle başlaması, bu hedefleri ulaşılması daha zor bir hale sokar. Hâlbuki tasarım etkinliği, veritabanı uygulanmaya başladığında değiştirilmesi kolay olmayan, katı bir şekilde tanımlanmış bir veritabanı görünümüyle sonuçlanır. Veritabanı tasarım sürecinde altı ana aşama belirlenebilir:

1. Gereksinimlerin derlenmesi ve çözümlenmesi
2. Kavramsal veritabanı tasarımı
3. Veritabanı yönetim sisteminin seçimi
4. Veri modeli dönüşümü (mantıksal veritabanı tasarımı olarak da adlandırılır.)
5. Fiziksel veritabanı tasarımı
6. Veritabanı sisteminin uygulanması

Tasarım süreci Şekil 2-1'de gösterildiği gibi iki paralel etkinliği içerir. İlk etkinlik veritabanının **veri içeriği ve yapısı**nın tasarımını içerir. İkincisi ise veritabanının **işlenmesi ve yazılım uygulamaları** ile ilgilidir. Bu iki etkinlik oldukça iç içe geçmiştir. Örneğin, veritabanında tutulacak veriler veritabanı uygulamaları çözümlenerek belirlenebilir. Buna ek olarak, veritabanı dosyalarının depolama yapısının ve erişim yollarının seçildiği fiziksel veritabanı tasarım aşaması, bu dosyaları kullanacak olan uygulamalara bağlıdır. Diğer taraftan, veritabanı uygulamalarının tasarımı genellikle ilk etkinlik sırasında oluşturulan veritabanı şema yapılarına başvurularak hazırlanır. Bu iki etkinliğin birbirlerini çok güçlü bir şekilde etkilediği açıktır.



Şekil 2-1 Büyük veritabanları için veritabanı tasarım aşamaları

Bahsedilen altı adım kesin bir dizi içerisinde ilerlemeyebilir. Birçok durumda erken aşamalardaki tasarımların sonraki aşamalarda değiştirilmesi zorunlu olabilir. Aşamalar arasındaki ve içindeki bu geri besleme döngüleri veritabanı tasarımı sırasında yaygındır. Şekli karıştırmamak için Şekil 2-1'de bunlar gösterilmemiştir. Bu şekildeki Aşama 1 veritabanı kullanımına yönelik bilgi toplamayı içerirken, Aşama 6 veritabanı uygulaması ile ilgilidir. Aşama 1 ve 6 bazen veritabanı tasarımının bir parçası olarak değil de daha genel bir bilgi sisteminin hayat çarkının bir parçası olarak düşünülür. Veritabanı tasarım işleminin kalbi aşağıda kısaca özetlenecek olan Aşama 2, 4 ve 5'tir.

- Kavramsal veritabanı tasarımı (Aşama 2): Bu aşamanın hedefi veritabanı yönetim sisteminden bağımsız bir kavramsal veritabanı şeması üretmektir. Bu aşamada çoğunlukla Varlık – İlişki (Vİ) ya da Geliştirilmiş Varlık – İlişki (GVİ) gibi yüksek seviye bir veri modeli kullanılır. Ek olarak belli bir veritabanı yönetim sisteminden bağımsız bir gösterim (notasyon) kullanılmak suretiyle veritabanı uygulamalarının ve işlemleri belirlenir.
- Veri modeli dönüşümü (Aşama 4): Buna mantıksal veritabanı tasarımı da denir. Bu aşama sırasında Aşama 2'de kullanılan yüksek seviye veri modelinde oluşturulan kavramsal veri şeması, Aşama 3'te seçilmiş olan veritabanı yönetim sisteminin veri modeline dönüştürülür. Bu aşamaya uygulama veri modelinin seçiminden sonra belli bir veritabanı yönetim sisteminin seçimini beklemeden (örneğin ilişkisel veritabanı yönetim sistemi kullanılmasına karar verilmiş ama adı konmamış ise) başlanabilir. Buna **sistem – bağımsız** (ama veri modeli – bağımlı) mantıksal tasarım denir. Bu aşamanın sonucu seçilmiş bir veri modelinde **kavramsal şema**dır. Ek olarak bu aşama boyunca belli uygulamalar için dış şemalar sıklıkla yapılır.
- Fiziksel veritabanı tasarımı (Aşama 5): Bu aşama sırasında depolanmış (doldurulmuş) veritabanına ait fiziksel depolama yapısı, kayıt yapma ve erişim yolları anlamında şartlar tasarlanır.

---

[1] Veri Tanımlama Dili – Data Definition Language
[2] Depo Tanımlama Dili – Storage Definition Language

2.1 Aşama 1: Gereksinimlerin Derlenmesi ve Çözümlenmesi

Bir veritabanının etkin olarak tasarlanmasından önce kullanıcı beklentilerinin ve veritabanının kullanılma eğiliminin olabildiğince ayrıntılı olarak bilinmesi gerekmektedir. Kullanılma eğiliminin belirlenmesi ve çözümlenmesi işlemi gereksinimlerin belirlenmesi ve çözümlenmesi olarak da adlandırılır. Aşağıdaki etkinlikler bu aşamanın tipik bölümleridir:

- Veritabanını kullanacak ana uygulama alanları ve kullanıcı grupları belirlenir. Her bir grupta, gereksinim derlenmesi ve belirlenmesinin alt aşamalarında ana katılımcı olacak anahtar personel seçilir.
- Önce uygulamalar ile ilgili hazır belgeler (varsa) incelenir. Diğer belgeler, (kılavuzlar, formlar, raporlar ve örgütsel grafikler) gereksinim derleme ve belirleme işlemini etkileyip etkilemediğine ilişkin olarak gözden geçirilir.
- Mevcut işletim çevresi ve bilginin planlanan kullanılma amacı araştırılır. Bu adım işlem türlerinin ve sıklıklarının olduğu kadar sistemdeki veri akışının incelenmesini kapsar. İşlemlere girdi ve çıktı olan veriler belirlenir.
- Olası veritabanı kullanıcılarından soru kümelerine olan yazılı yanıtları (anketler) toplanır. Bu sorular, kullanıcıların çeşitli uygulamalara verdikleri önem ve öncelikleri kapsar. Anahtar personel ile bilginin değerinin biçilmesi ve önceliklerinin kurulması konularında yardım almak için görüşmeler yapılabilir.

Yukarıda bahsedilen gereksinim derleme yolları zayıfça yapılandırılmış ve informal gereksinimleri verir. Bunlar daha sonra daha formal **gereksinim belirleme tekniklerinden** biri kullanılarak daha iyi yapılı bir hale dönüştürülür. Bu teknikler şunlardır: Sıradüzensel Girdi İşlem Çıktı – SGİÇ (Hierarchical Input Process Output – HIPO), Yapısal Çözümleme ve Tasarım Tekniği – YÇTD (Structured Analysis and Design Technique – SAD), Veri Akış Diyagramları – VAD (Data Flow Diagrams – DFDs), Orr – Warnier Diyagramları, ve Nassi – Schneiderman Diyagramları. Tüm bunlar bilgi işleme gereksinimlerini düzenleme ve sunma için kullanılan diyagramsal yöntemlerdir. Bu diyagramlar yöntemine göre sıra düzenler, akış diyagramları, ardışık ve döngü yapıları vb. gösterimi kullanırlar. Metin, tablo, diyagram şeklindeki ilave belgeler ve karar gereksinimleri genellikle diyagramlara eşlik eder. Sadece bu aşamanın tartışıldığı kitaplardan bilgi alınabilir (Gane, C. ve Sarson, T., 1977),

(Pressman, Roger S., 1992), (Stevens, W. P., Myers, G. J. ve Constantine L. L., 1974). Bir sonraki bölümde bu çalışmada kullanılacak olan VAD hakkında bilgi verilecektir. Bu aşama oldukça zaman alıcı olabilir, fakat bilgi sisteminin gelecekteki başarısı için çok önemlidir.

Gereksinimlerin derlenmesi ve çözümlenmesi işi için bazı bilgisayar – yardımlı teknikler üzerinde bazı çalışmalar yapılmıştır. Bu teknikler koşulların bütünlüğünü ve kararlılığının denetimi için gereksinim çözümlemelerinde kullanılacak otomatik araçları içerir. Gereksinimler, genellikle tasarım veritabanı adı verilen tek bir ambarda depolanır ve tasarım ilerledikçe buradan görüntülenebilir ve güncellenebilir (Örn: SORUN TANIMLAMA DİLİ VE SORUN TANIMLAMA ÇÖZÜMLEYİCİSİ – STD / STÇ, PROBLEM STATEMENT LANGUAGE and PROBLEM STATEMENT ANALYZER – PSL / PSA (Eick, C. ve Lockemann, P., 1985).

2.1.1 Veri akış diyagramları

Veri akış diyagramları; sistemin karmaşıklık seviyesi ne olursa olsun o sistemin mantıksal seviyede anlaşılmasında ve model düzenlemelerinin yapılması amacıyla çözümleme safhasında kullanılan diyagramsal bir araçtır. Veri akış diyagramlarına kabarcık diyagramları (bubble chart) de denir.

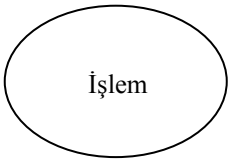
Veri akış diyagramları ;

- Sistemlerin alt sistemlere olan ayrışmasını,
- Sistem içindeki veri akışını,
- Veri depolarını ve buralara giren – çıkan verileri ve
- Sisteme göre dış varlık olarak bilinen kullanıcıları göstermektedir.

Veri akış diyagramlarında kullanılan işaretler şunlardır:

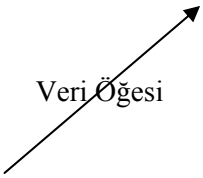


Modellenen sistem dışında kalan bir bilgi üreticisi veya tüketicisidir. Dörtgenin içerisine varlık ismi tekil olarak yazılır.



Modellenen sistem içinde kalan bir bilgi dönüştürücüsüdür. Otomatik veya elle yapılan bir işi veya işlemi gösterir. İşlem, kabarcığa giren verinin ya da bilginin kabarcıktan çıkan veri ya da

bilgiye dönüşümü işlevini anlatmaktadır. Kabarcık içine emir kipinde bir fiil ya da nesne yazılır. Veri akış diyagramlarına şekli itibariyle diğer ismini verir.

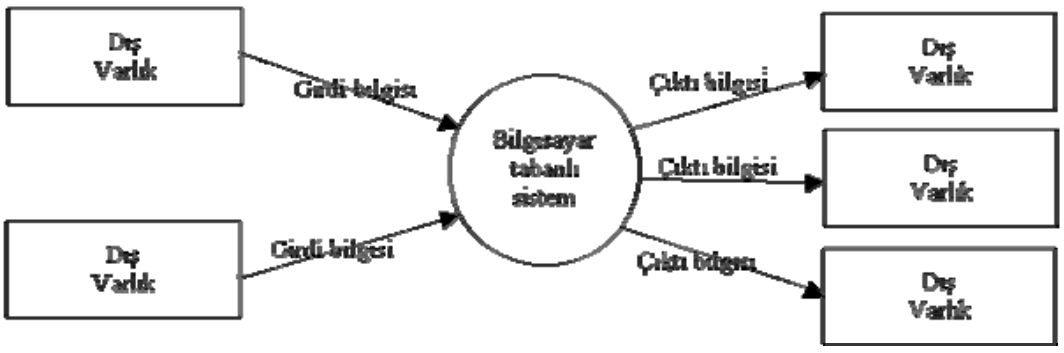


Bir veri öğesi ya da veri öğeleri topluluğudur. Ok verinin akış yönünü gösterir. Akan verinin ismi okun üzerine veya kesecek şekilde yazılır. Veri akışını gösteren oklar aynı zaman da sistemin sınırlarını gösterir.



Bir veya daha fazla işlemin kullanacağı verinin depolanacağı bir havuzdur. Bir kuyruk ya da arabellek kadar basit ya da ilişkisel bir veritabanı kadar karmaşık olabilir. İçlerine depoladığı veriyi en iyi şekilde anlatan tekil isim yazılır. Veriler bir işlemden bir işleme doğrudan nadiren gönderilir; depoların kullanımı burada ortaya çıkmaktadır. Depolar aslında verilerin bir işlemden bir işleme gönderilmeden önce sıralanmak veya başka işlemler yapılmak üzere alıkonuldukları havuzlardır. Dolayısıyla sistem başarımını yavaşlatan bir unsur olarak görülürüler. Depo işaretinin içine yazılacak olan isim giren – çıkan veriyi iyi bir şekilde tanımlıyorsa veri akışının üzerine giren – çıkan verinin ismi yazılmayabilir.

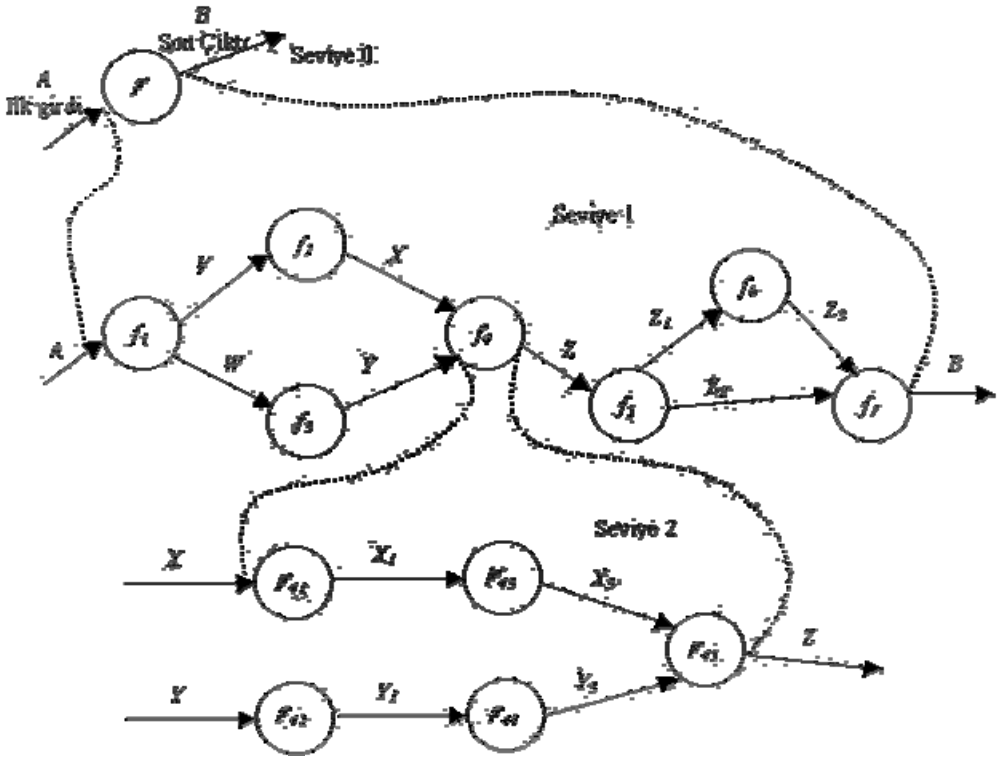
Veri akış diyagramları hazırlanırken genelde kaynak dış varlıklar, kağıdın solunda; hedef dış varlıklar, kağıdın sağında, işlemler ortada, depolar ise alt ve üst kısımda çizilir. Depo, ve dış varlıklar gerekirse birden fazla gösterilirler (***Şekil 2-2***).



Şekil 2-2 Bilgi akış modeli

Veri akış diyagramları içerdikleri anlam itibariyle üç kategoriye ayrılabilir (Şekil 2-3) (Bruyn, W. ve ark., 1988);

1. Ana Diyagram – Seviye 0: Düşünülen sistemin sınırlarını bir bütün olarak belirleyen diyagramdır. Bu diyagramda sistem ve dışında kalan kullanıcılar gösterilir.
2. Tasarım Diyagramı – Seviye 1: Sistem içinde kullanılan veri depolarını, giren ve çıkan tüm veri akışlarını ve sorgulamalarını gösterir. Ayrıca sistem; birden fazla alt sistemden oluşuyorsa her biri için ayrı bir tasarım diyagramı hazırlanabilir.
3. Detaylı Diyagram – Seviye 2: Tasarım diyagramlarındaki işlemlerin basitleştirilmesi amacıyla kullanılan alt – seviye diyagramlardır.



Şekil 2-3 Bilgi akış iyileştirmesi

2.2 Aşama 2: Kavramsal Veritabanı Tasarımı

Veritabanı tasarımının ikinci aşaması iki paralel etkinlik içerir. Bunların ilki olan kavramsal şema tasarımında, Aşama 1'den elde edilen veri gereksinimleri incelenir ve bir kavramsal veritabanı şeması üretilir. İkinci etkinlik olan işlem tasarımında, Aşama 1'de irdelenen veritabanı uygulamaları incelenir ve bu işlemler için yüksek seviyeli tanımlar üretilir.

2.2.1 Aşama 2a: Kavramsal şema tasarımı

Bu aşamada üretilen kavramsal şema çoğunlukla VTYS'den bağımsız yüksek seviyeli bir veri modeli ile yapıldığından doğrudan veritabanına uygulanamaz. VTYS – bağımsız bir kavramsal şema aşağıdaki nedenlerden dolayı önemlidir:

1. Kavramsal şema tasarımının hedefi; veritabanı yapısının anlamının (semantiğinin), ara ilişkilerin ve koşullarının tam olarak anlaşılmasıdır. Kavramsal şema tasarımı her veritabanı yönetim sisteminin kendine has teknik ve kısıtlamalarından kaynaklanabilecek etkileri bertaraf etmek için veritabanı yönetim sisteminden bağımsız olmalıdır.
2. Kavramsal şema tasarımı veritabanı içeriğinin sabit tanımlamalarını içermez. Veritabanı yönetim sistemi seçimi ve sonraki tasarım kararları VTYS – bağımsız kavramsal şema tasarımı değişimini gerektirmeden değişebilir.
3. Veritabanı kullanıcıları ve uygulama tasarımcıları için kavramsal şemanın iyi anlaşılması ve bunun sağlanması için veritabanı yönetim sisteminin veri modelinden daha genel ve açıklayıcı bir yüksek seviye veri modelinin kullanımı çok önemlidir.
4. Kavramsal şemanın diyagramsal tanımlaması veritabanı kullanıcıları, tasarımcıları ve çözümcüleri arasında mükemmel bir iletişim aracı olarak hizmet edebilir. Yüksek seviye veri modelleri, VTYS – bağımlı daha düşük seviyeli veri modellerine göre daha kolay anlaşılır kavramlara dayandığından, şema tasarımı ile ilgili herhangi bir iletişim, daha aracısız ve kesin olarak meydana gelir.

Veritabanı tasarımının bu aşamasında, yüksek seviyeli semantik veya kavramsal veri modeli de denilen aşağıdaki özelliklere sahip bir veri modelinin kullanılması önemlidir (McFadden F. ve Hoffer, J., 1988):

1. Anlamlılık: Veri modeli değişik tip verileri, ilişkileri ve koşulları ayrı anlamlarla ifade edebilmelidir.
2. Basitlik: Vasıfsız bir kullanıcının kavramlarını anlayıp kullanabileceği kadar basit olmalıdır.
3. Azlık (minimalite): Model, anlam olarak ayrı ve örtüşmeyen az sayıda temel kavrama sahip olmalıdır.

4. Diyagramsal gösterim: Model, kavramsal şemayı temsil için yorumlaması kolay, diyagramsal bir görünüme sahip olmalıdır.
5. Kurallılık: Bir veri modeli ile vurgulanan bir kavramsal şema belirsizliği bulunmayan kurallı bir veriyi temsil etmelidir, yani, model kavramları doğru ve belirsizliğe yer bırakmayacak şekilde tanımlanmalıdır.

Bu gereksinimler bazen birbiriyle çelişir. Özellikle ilk gereksinim diğerleriyle çelişir. Veritabanı tasarımı için birçok yüksek seviye kavramsal model önerilmiştir (Abrial, J., 1974), (Verheijen, G. ve VanBekkum, J., 1982), (Codd, E, 1974). Bölüm 2.2.1.1'de veri modelleri hakkında kısa bilgi verilmiştir. Bu çalışmada, Varlık – İlişki, Vİ (Entity – Relationship, ER) modeli kullanılmaktadır.

2.2.1.1 Veri modelleri

Veritabanlarının mantıksal yapıları belli biçimlerde modellendirilmektedir. Veri modeli, fiziksel depolama ve erişim yolları başta olmak üzere veritabanının birçok özelliklerini biçimlendirmekte ve buna göre her veritabanı yönetim sistemi yazılımı belli bir modeli esas almaktadır. Bu nedenle "Model" ve "Veri Modeli" kavramlarına açıklık getirmek gereklidir.

Model; gerçek dünyadaki varlıkların, olayların ve bunlar arasındaki ilişkilerin temsil ediliş şeklidir.

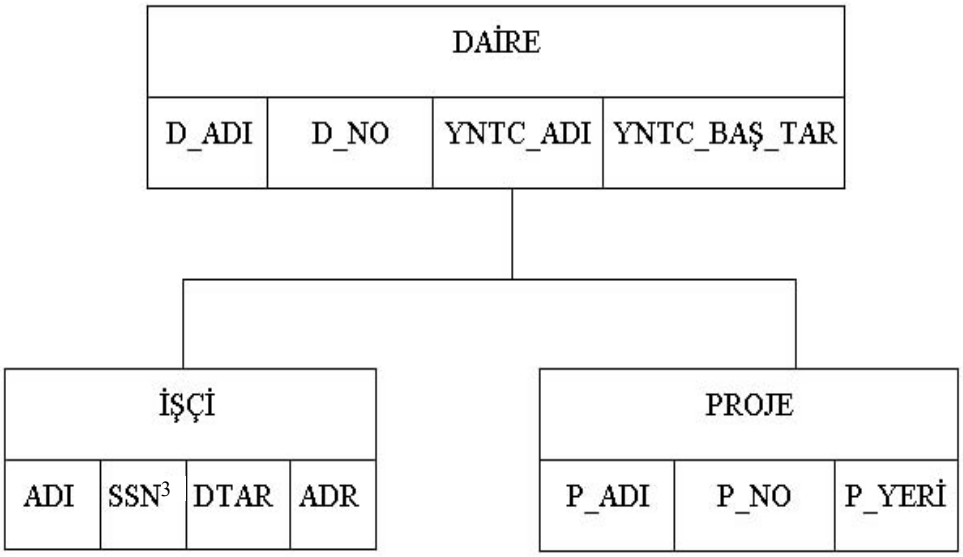
Veri modeli; gerçek dünyadaki varlıklar, olaylar, etkinlikler ve bunlar arasındaki ilişkiler hakkındaki verilerin temsil ediliş şeklidir.

Varlıklar ve olayların kendi aralarında üç çeşit ilişkisi vardır:

- Bire bir (1 : 1) ilişki (one – to – one relationship)
- Bire çok (1 : M) ilişki (one – to – many relationship)
- Çoka çok (M : N) ilişki (many – to – many relationship)

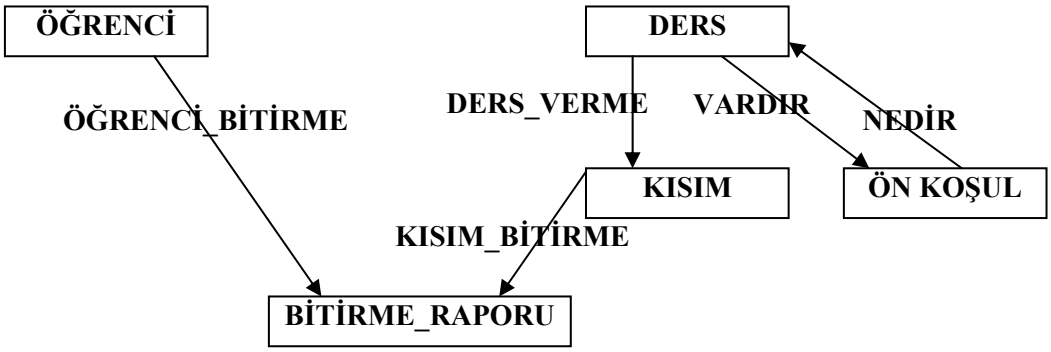
Bu ilişkiler ışığında şu veri modelleri tanımlanabilir;

- Sıradüzensel veri modeli: Varlıklar arası ilişkiler bire – çok şeklindedir. İlişkiler değişik düzeylerdeki bağlantılarla kurulur. Alt düzeydeki bir varlık sadece bir üst düzeydeki varlıklarla bağlantılıdır (***Şekil 2-4***).
- Ağ veri modeli: Varlıklar arası ilişkiler çoka – çok şeklindedir. İlişkiler değişik düzeylerdeki bağlantılarla kurulur. Alt düzeydeki bir varlık üst düzeydeki birden fazla varlık ile bağlantılı olabilir (***Şekil 2-5***).



Şekil 2-4 Sıradüzensel veri modeli

- İlişkisel veri modeli: Bu model, veritabanını her bir tablonun bir dosyada tutulduğu bir tablolar topluluğu olarak alır (***Şekil 2-6***).



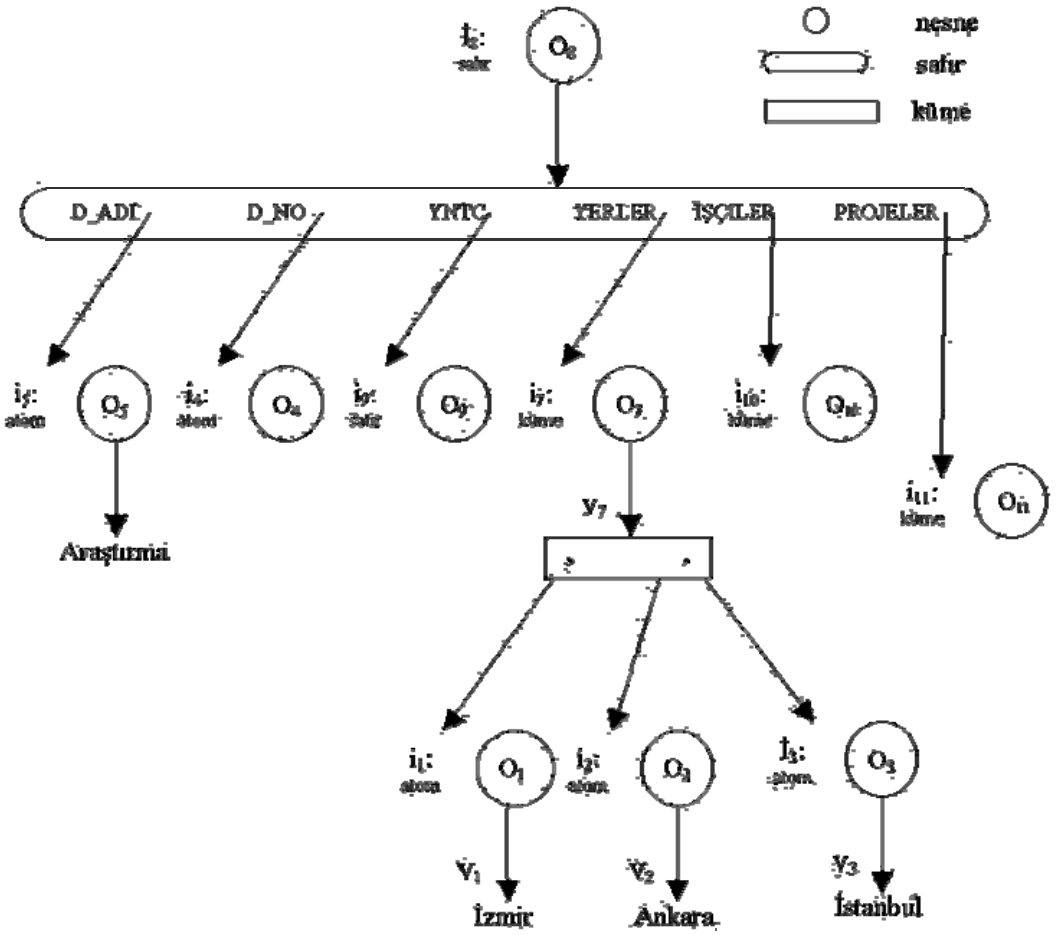
Şekil 2-5 Ağ veri modeli

[3] Sosyal Sigorta Numarası

- Nesneye – yönelik veri modeli: Bu model veritabanını nesneler, onların özellikleri ve onların işlemleri cinsinden tanımlar. Aynı yapı ve davranıştaki nesneler bir sınıfa aittir ve sınıflar sıradüzensel veya dönüşsel olmayan (acyclic) diyagramlar olarak düzenlenirler (Şekil 2-7).

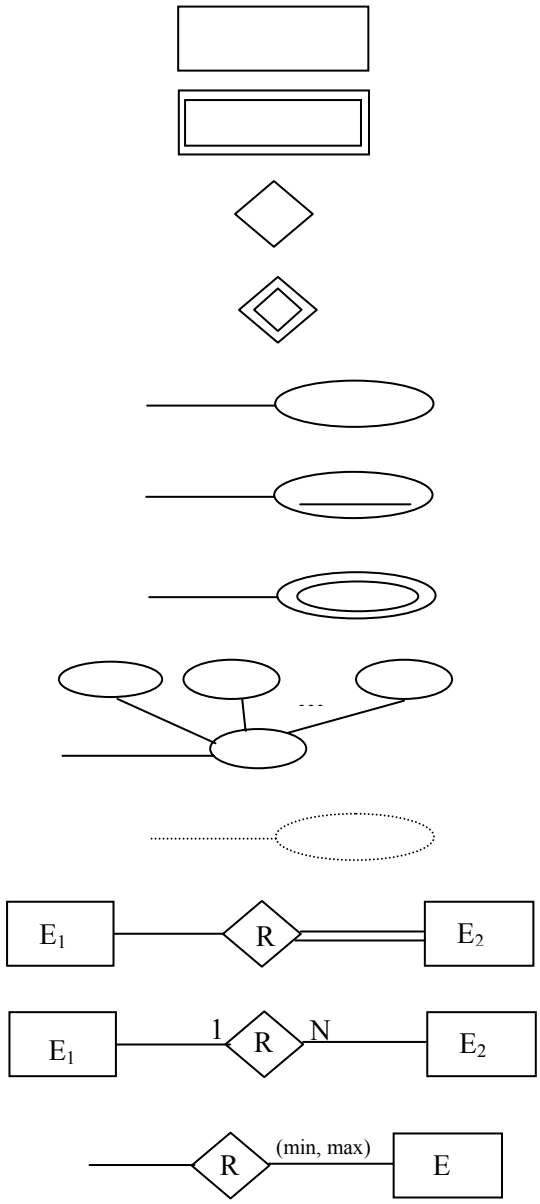
ÖĞRENCİ	Adı	ÖğrenciNo	Sınıf	Branş
	Erol	12	1	SOS
	Ebru	23	2	FEN

Şekil 2-6 İlişkisel veri modeli



Şekil 2-7 Nesneye yönelik veri modeli

- Yüksek seviye kavramsal veri modelleri (Veri – İlişki): Gösterim anlamları Şekil 2-8'de verilen bu model, veriyi varlık, ilişki ve öznitelik olarak tanımlar ve bunların belirlenmesinden meydana gelir (Şekil 2-9) (Chen, P., 1976), (ER Conference, 1992).



Varlık türü

Zayıf varlık türü

İlişki türü

Zayıf ilişki türü

Öznitelik

Anahtar öznitelik

Çoklu değerli öznitelik

Birleşik öznitelik

Türetilmiş öznitelik

E_2'nin R' de tam katılımı

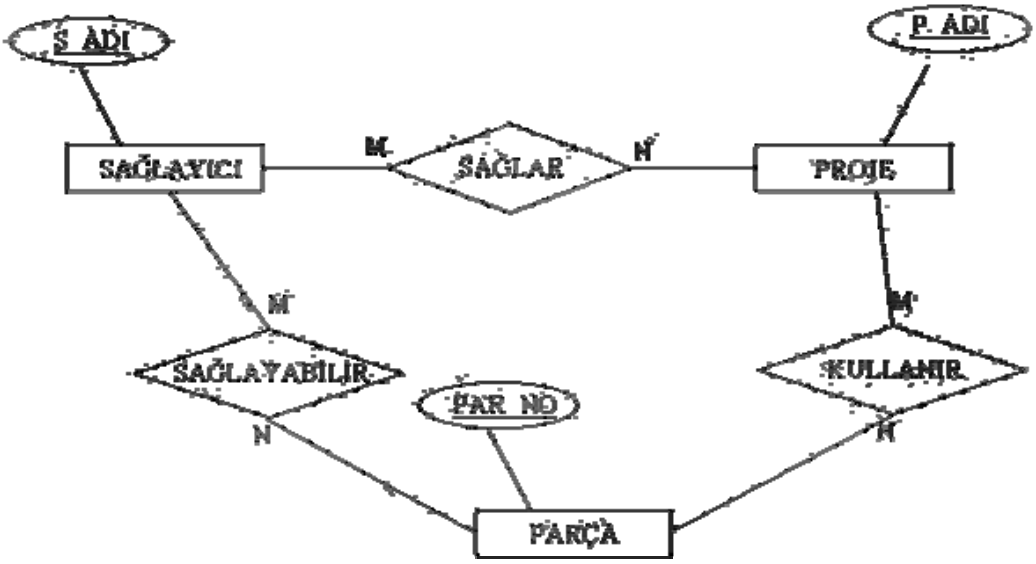
E_1 : E_2' nin R için sayal oranı 1 : N

E' nin R' deki katılımında (min, max) yapısal koşulu

Şekil 2-8 Varlık - İlişki gösterim işaretleri özeti

2.2.1.2 Kavramsal şema tasarımına yaklaşımlar

Kavramsal şema tasarımı için önce şemanın varlık tipleri, ilişki tipleri ve öznitelikleri gibi temel bileşenlerinin belirlenmesi gerekir. Ayrıca anahtar öznitelikler, ilişkilerdeki katılım ve önem koşulları, zayıf varlık türleri, sıradüzenlerin özelleştirilmesi / genelleştirilmesi işlemleri de tanımlanmalıdır. Bu tasarım Aşama 1'de derlenen gereksinimlerden türetilir.



Şekil 2-9 Varlık – İlişki veri modeli

Kavramsal şema tasarımına iki tür yaklaşım vardır (McFadden F. ve Hoffer, J., 1988): **Merkezi (tek – atım) şema tasarımı yaklaşımı** adı verilen ilk yaklaşımda, Aşama 1'den gelen farklı uygulama ve kullanıcı grupları *kavramsal şema tasarımı başlamadan* bir gereksinim kümesi içinde birleştirilir. Daha sonra birleştirilmiş bu gereksinim kümesine karşılık tek bir şema tasarlanır. Çok sayıda uygulama ve kullanıcı olduğunda tüm gereksinimleri birleştirmek zor ve zaman alıcı bir görev olabilir. Bu yaklaşımın ardındaki kabul merkezi bir otoritenin (veritabanı yöneticisi) farklı kullanıcı ve uygulamaların gereksinimlerinin nasıl birleştirileceğine karar vermekten ve tüm veritabanı için kavramsal şemayı tasarlamaktan sorumlu olmasıdır. Kavramsal şema tasarımı sona erdiğinde çeşitli kullanıcı grup ve uygulamaları için dış şemalar, veritabanı yöneticisi tarafından belirlenebilir.

Görünüm birleştirme yaklaşımı adı verilen ikinci yaklaşımda, gereksinimlerin birleştirilmesi yerine her bir grup veya uygulamanın sadece kendi ihtiyaçlarına dayalı bir şema (veya görünüm) tasarlanır. Sonuç olarak, her grup ve uygulama için tek bir yüksek seviye şema geliştirilir. Sonraki görünüm birleştirme aşaması sırasında, bu şemalar, tüm veritabanı için tek bir genel (global) kavramsal şemada birleştirilirler. Görünüm birleştirmesinden sonra her bir görünüm ayrı ayrı dış şemalar olarak yeniden oluşturulabilirler.

Bu iki yaklaşım arasındaki temel fark, çok sayıda kullanıcı ve uygulamaya ait çeşitli görünüm ve gereksinimlerin bağdaştırılması safhasındadır. Merkezi yaklaşımda, bağdaştırma şemaların tasarımından önce veritabanı yönetimi çalışanları tarafından elle yapılır ve Aşama 1'de derlenen gereksinimlere doğrudan uygulanır. Bu yaklaşım, kavramsal şema tasarımı uygulamaya konmadan genel gereksinimlerde açık ve anlaşılır tanımlar oluşturmak için kullanıcı grupları arasındaki farklılık ve çakışmaların bağdaştırılması konusunda veritabanı yönetimi çalışanlarına yıktığı yüke rağmen çoklukla kullanılmıştır. Bu görevin yerine getirilmesindeki güçlükler yüzünden Görünüm Birleştirme Yaklaşımı günümüzde daha çok kabul görmeye başlamıştır (Batini, C., Lenzerini, M. ve Navathe S., 1987).

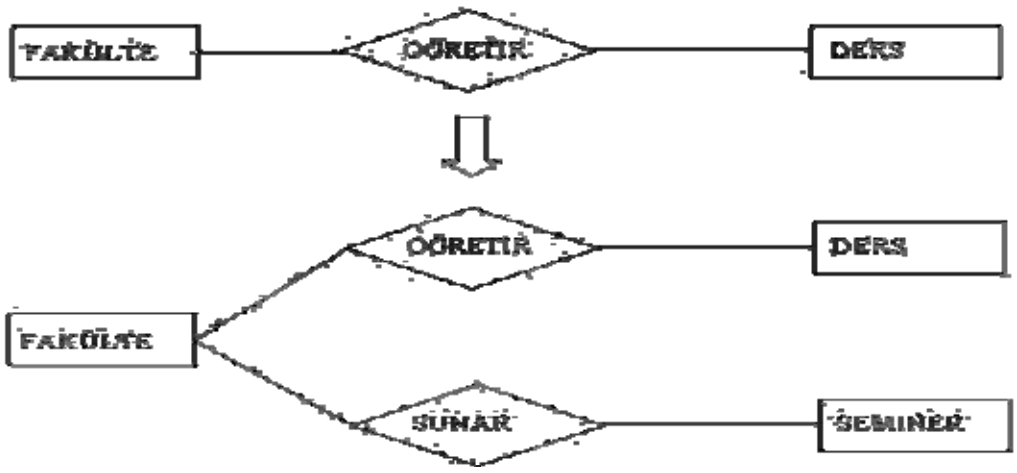
Görünüm birleştirme yaklaşımında, her bir kullanıcı grubu veya uygulama, aslında kendi gereksinimlerinde kendi Vİ diyagramını tasarlar. Daha sonra bu diyagramlar genel bir birleştirilmiş diyagram oluşturmak için veritabanı yöneticisi tarafından bir işleme tabi tutulur. Görünüm birleştirme elle yapılabilmesine rağmen, onlarca kullanıcı içeren bir veritabanına uygulanabilmesi bir yöntembilim ve birleştirmeyi kolaylaştıracak araçların kullanımını gerektirir. Özniteliklerin uygunluğu, çeşitli görünümlerdeki varlık ve ilişki türleri, birleşimden önce belirlenmelidir. Ek olarak, çakışan görünümlerin birleştirilmesi ve belirli diyagramlar arası uygunlukların kararlılığının doğrulanması gibi sorunlar çözülmelidir.

2.2.1.3 Şema tasarımı için stratejiler

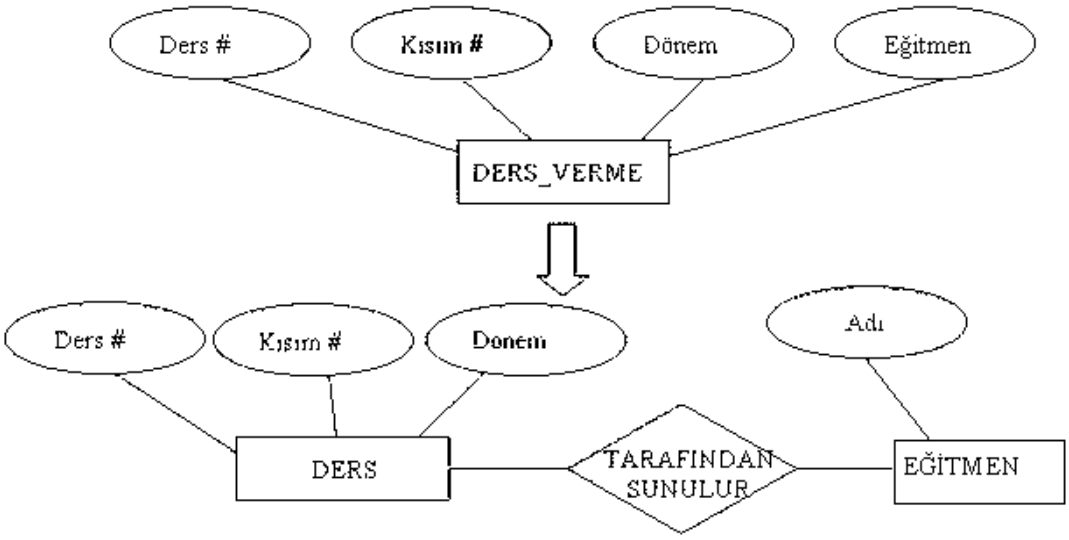
Tek bir kullanıcı için ya da bir kullanıcı grubu için bir gereksinim kümesi verildiğinde, bu gereksinimleri karşılayacak bir kavramsal şema yaratılmalıdır. Bu tür bir şemanın tasarımı için çeşitli stratejiler söz konusudur (Yao, S., 1985). Bu stratejilerin çoğu artan bir yaklaşım izlerler; yani, gereksinimlerden türetilen şema

yapıları ile başlarlar ve sonra artan bir şekilde bunları düzeltir, iyileştirir veya inşa ederler. Bu stratejiler şunlardır:

1. Yukardan – aşağı: Yüksek seviye kavramlar (abstractions) içeren bir şema ile işe başlanır ve sonra ardışık yukardan – aşağı iyileştirmeler uygulanır. Örneğin; sadece birkaç yüksek seviye varlık türünün belirlenmesi ile başlanabilir. Daha sonra, öznitelikler belirlenmeye başladıkça bunlar düşük – seviye varlık ve ilişkilere ayrıştırılabilir. Bir varlık türünü alt sınıflara ayrıştırarak iyileştirmek için yapılan belirleme işlemi de diğer bir yukardan – aşağı tasarım stratejisi örneğidir.
2. Aşağıdan – yukarı: Temel kavramlar içeren bir şemayla başlanır, sonra bunlar birleştirilir veya bunlara eklemeler yapılır. Örneğin; öznitelikler ile başlanabilir ve bunlar varlık türleri ve ilişkilerle gruplanabilirler. Alt sınıfları yüksek seviye genel sınıflara genelleştirme işlemi de diğer bir aşağıdan – yukarı tasarım stratejisi örneğidir.
3. İçten – dışa: Dikkatin en açık kavramlardan oluşan kümede yoğunlaştırıldığı özel bir aşağıdan – yukarı tasarım stratejisi uygulamasıdır. Modelleme daha sonra hazır kavramlar etrafındaki yeni kavramların ele alınmasıyla içeriden dışarıya genişler. Örneğin; şemada birkaç aşikar varlık türü belirlenip, her biri diğeriyle ilişkili diğer varlık türlerini ve ilişkileri bunlara ekleyerek devam edilebilir.
4. Karma: Tasarım boyunca özel bir strateji belirlemek yerine, gereksinimler yukardan – aşağı stratejiye göre bölümlenir ve her bir bölümün şema kısmı, aşağıdan – yukarı stratejiye göre tasarlanır. Değişik şema kısımları daha sonra birleştirilir.

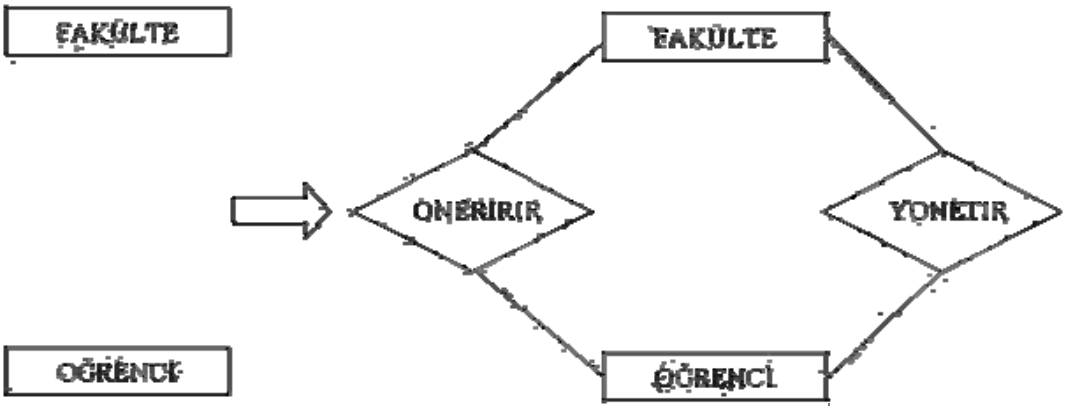


Şekil 2-10 Yukardan – Aşağı iyileştirme örnekleri (Yeni bir varlık türü türetme)



Şekil 2-11 Yukardan – Aşağı iyileştirme örnekleri (Bir varlık türünü iki varlık türü ve bir ilişki olarak ayrıştırma)

Şekil 2-10, Şekil 2-11 yukarıdan – aşağı ve aşağıdan – yukarı iyileştirmeleri göstermektedir. İlkel bir yukardan – aşağı iyileştirmeye örnek, bir varlık türünün birkaç varlık türüne ayrıştırılmasıdır. Şekil 2-10 DERS varlığının DERS ve SEMİNER varlıklarına, ÖĞRETİR ilişkisinin ÖĞRETİR ve SUNAR ilişkilerine ayrışmasını gösterir. Şekil 2-11 bir DERS_VERME varlığının aralarında bir ilişki olan iki varlık türüne ayrışımını gösterir. Şekil 2-12 ilkel bir aşağıdan – yukarı iyileştirme olan varlık türleri arasında ilişki kurmayı gösterir.



Şekil 2-12 Aşağıdan – Yukarı iyileştirme örneği (Yeni bir ilişkinin bulunması ve eklenmesi)

2.2.1.4 Şema (görünüm) birleştirme

Çok sayıda kullanıcı ve uygulamaya sahip büyük veritabanları için bireysel şemalar tasarlayıp birleştirmeden oluşan görünüm birleştirme yaklaşımı kullanılabilir. Bireysel görünümler nispeten küçük olduğundan, şema tasarımı basitleştirilmiş olur.

Bununla birlikte, görünümleri genel bir şemada birleştirmek için bir yöntembilime ihtiyaç vardır. Şema birleştirimi aşağıdaki alt görevlere bölünebilir (Navathe, S., Elmasri, R. ve Larson, J., 1986):

1. Şemalar arası benzerlik ve çakışmaların bulunması: Şemalar bireysel olarak tasarlandığı için, aynı gerçek – dünya kavramını temsil edecek şema yapılarının belirlenmesi gerekmektedir. Birleştirme ilerlemeden önce bu benzerlikler bulunmalıdır. Bu işlem sırasında şemalar arası şu türden çakışmalar bulunabilir:

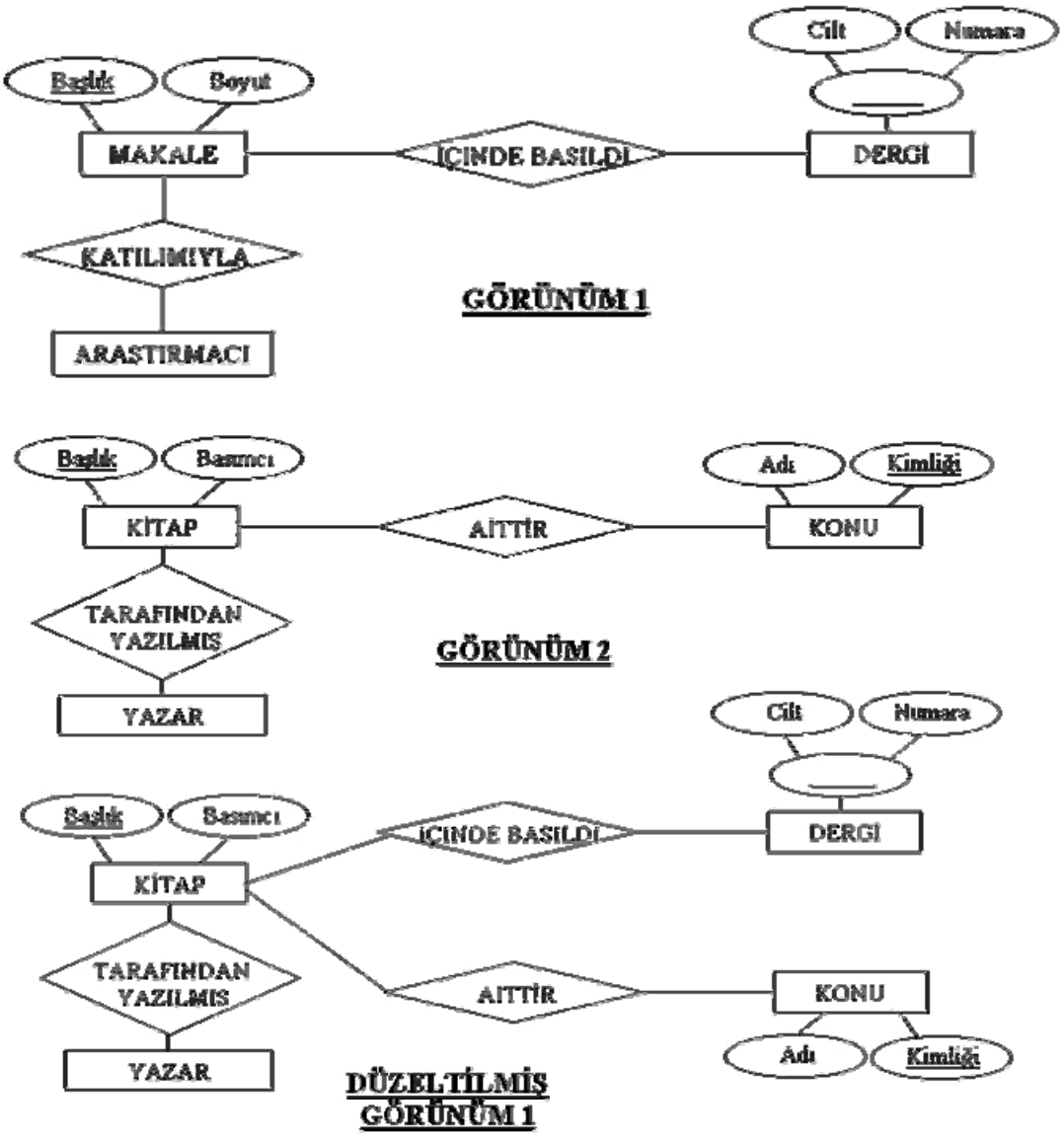
    a. İsimlendirme çakışmaları: Bunlar eşanlamlılar ve eşsesliler olmak üzere iki türdür. Eşanlamlılar, farklı şemalar aynı kavram için farklı isimler kullandığında ortaya çıkar. Örneğin; bir şemadaki MÜŞTERİ varlık türünün kastettiği anlam başka bir şemada ALICI varlık türünün kastettiği anlamla aynı olabilir. Eşseslilik ise farklı şemalarda farklı kavramlar için aynı isim kullanıldığında ortaya çıkar. Örneğin; PARÇA varlık türü bir şemada bilgisayar parçası anlamına gelirken bir başkasında mobilya parçası anlamına gelebilir.

    b. Tür çakışmaları: Aynı kavram farklı şemalarda farklı modelleme yapısı ile temsil edilebilir. Örneğin; ŞUBE kavramı bir şemada varlık türü iken bir başkasında öznitelik olabilir.

    c. Tanım kümesi çakışması: Bir öznitelik farklı şemalarda farklı tanım kümelerine sahip olabilir. Örneğin; SSN (Sosyal Sigorta Numarası) bir şemada tam sayı olarak kullanılırken bir başkasında karakter dizgisi olarak kullanılabilir. Bir şemada YÜKSEKLİK fit bir başkasında metre olarak temsil edildiğinde ölçü birimi çakışması meydana gelmiş olur.

2. Görünümleri birbirine uydurma: Bazı şemalarda, diğer şemalara daha çok uymaları için değişiklikler yapılabilir. Adım 1'de bahsedilen çakışmaların bazıları bu adımda giderilir.

3. Görünümleri birleştirme: Genel şema, bireysel şemaların birleştirilmesiyle oluşturulur. Birbirine benzeyen kavramlar genel şemada tek olarak temsil edilirler ve görünümlerin kendi aralarındaki ve genel görünümle aralarındaki eşlemeler belirlenir.
4. Yeniden yapılandırma: İsteğe bağlı son bir adım olarak, genel şema çözümlenebilir ve gereksizlikleri gidermek için yeniden yapılandırılabilir. Bu fikirlerin bazıları ***Şekil 2-13***'de oldukça basit örneklerle gösterilmiştir.



Şekil 2-13 Birleştirmeden önce uyuşum için görünümlerin düzeltilmesi

Şekil 2-13'de bir "Kaynakça Veritabanı" yaratmak için iki görünümün nasıl birleştirildiği gösterilmektedir. İki görünüm arasındaki benzerliklerin belirlenmesi sırasında, ARAŞTIRMACI ve YAZAR'ın, TARAFINDAN_DAĞITILIR ve KATILIMIYLA gibi eşanlamlı (bu veritabanı dikkate alındığında) olduğu bulunur. Sonra, GÖRÜNÜM 2'ye uyması için GÖRÜNÜM 1'e, MAKALE için bir KONU içermesi değişikliği yapılır.

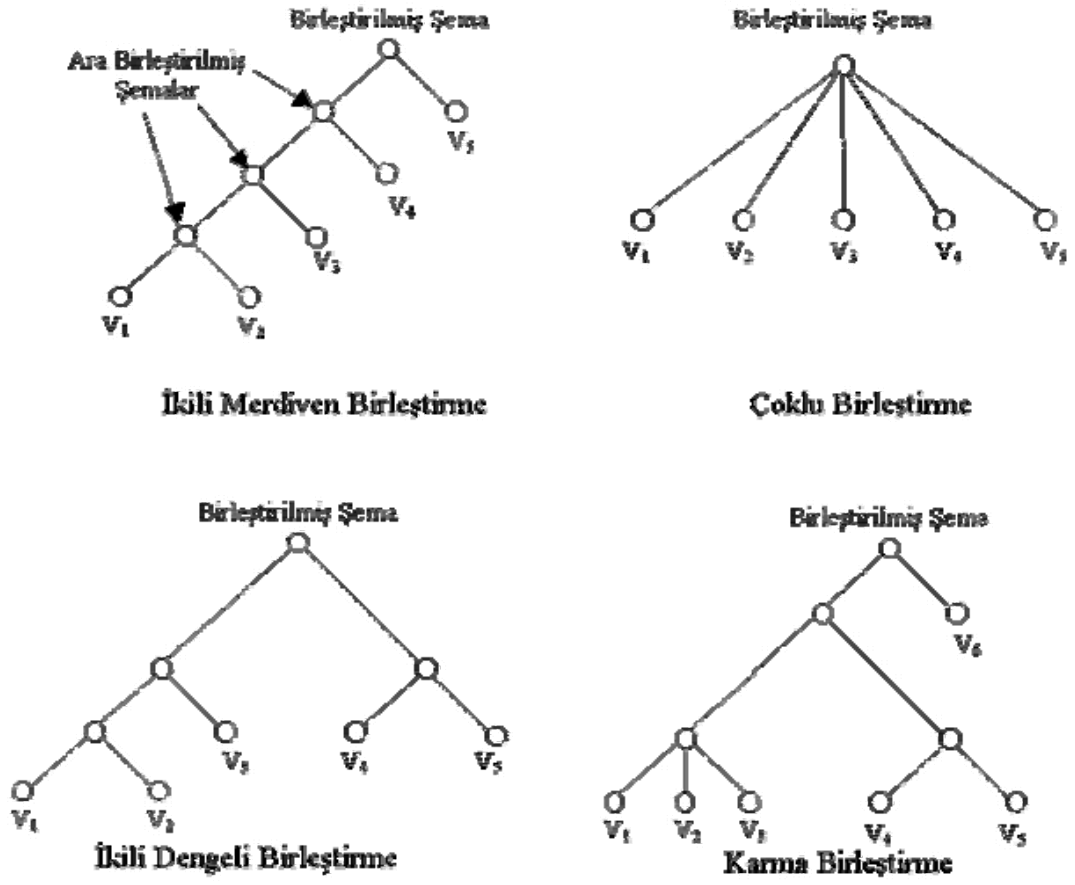
Görünüm birleştirme işlemi için çeşitli stratejiler önerilmiştir (Casanova, M. ve Vidal, V., 1982). ***Şekil 2-14***'de gösterildiği gibi bunlar şunlardır:

1. İkili merdiven birleştirmesi: İlk önce birbirine çok benzeyen iki şema birleştirilir. Sonra bu sonuç şema bir başkasıyla birleştirilir ve bu işlem tüm şemalar birleştirilene dek tekrar edilir. Birleştirme için şemaların sıralanmasında ölçüt olarak şema benzerliği alınabilir. Bu strateji izlediği adım – adım yaklaşımdan dolayı elle birleştirme için uygundur.
2. Çoklu birleştirme: Benzerliklerin çözümlenmesi ve belirlenmesinden sonra tüm görünümler tek bir işlemle birleştirilir. Bu strateji, büyük tasarım sorunlarından dolayı bilgisayarlı araçlar gerektirir.
3. İkili dengeli strateji: Önce şema çiftleri birleştirilir; sonra ileri birleştirmeler için sonuç şemalar eşleştirilir ve bu işlem sonuçta genel bir şema oluşturulana dek tekrar edilir (Teorey, T. ve Fry, J., 1982).
4. Karma strateji: Başta, şemalar benzerlikleri göz önüne alınarak gruplara bölünür ve her bir grup ayrı olarak birleştirilir. Ara şemalar tekrar gruplanır ve birleştirilir.

2.2.2 Aşama 2b: Veritabanı işlemi tasarımı

Aşama 2a ile paralel ilerleyen bu aşamanın amacı, bilinen veritabanı işlemlerinin özelliklerini VTYS – bağımsız bir şekilde tasarlamaktır. Bir veritabanı tasarlanırken tasarımcılar, veritabanı uygulamaya konduğunda veritabanı üzerinde çalışacak pek çok işlemin farkındadırlar. Bu işlemlerin işlevsel özelliklerinin tasarım işlemi sırasında belirlenmesi, veritabanı tasarımının önemli bir bölümünü oluşturur. Böylece, veritabanı şemasının bu işlemler tarafından gereken tüm bilgiyi içermesi garanti altına alınmış olur. Ek olarak, çeşitli işlemlerin nısbî önemlerinin bilinmesi ve bunların ümit edilen kullanılma oranları fiziksel veritabanı tasarımının (Aşama 5)

önemli bir bölümünü oluşturur. Genel olarak, tasarım sürecinde sadece bazı veritabanı işlemleri bilinir; veritabanı sistemi uygulanmaya başlandıktan sonra, yeni işlemler sürekli olarak tanımlanır ve uygulanır. Bununla birlikte, çoğu önemli işlemler sistemin uygulanmasından önce bilinir ve ilk evrelerde tanımlanmalıdır.



Şekil 2-14 Görünüm birleştirmesi için farklı stratejiler

Kavramsal düzeyde işlem belirlemenin yaygın bir tekniği onların girdi / çıktı ve işlevsel davranışlarının tanımlanmasıdır. Girdi verinin, çıktı verinin ve denetimin iç işlevsel akışının belirlenmesiyle tasarımcılar bir işlemi kavramsal ve sistem – bağımsız bir şekilde belirleyebilirler. İşlemler genellikle üç kategoride gruplanırlar: istemsel işlemler, güncelleme işlemleri ve karma işlemler. İstemsel işlemler ekranda gösterim ya da bir rapor üretimi için veri isteminde kullanılırlar. Güncelleme işlemleri, veritabanına yeni veri girişi veya varolan verinin değiştirilmesinde kullanılırlar. Karma işlemler, bazı istem ve güncellemeler yapan daha karmaşık

uygulamalarda kullanılır. Bir havayolları rezervasyon veritabanı tasarlandığı varsayılırsa; belli bir tarihte iki şehir arasındaki sabah uçuşlarının listesini almak istemsel işleme, bir müşteri için bir uçuşta yer ayırmak güncelleme işlemine örnek verilebilir. Karma işleme, herhangi bir uçuştaki müşteri rezervasyonunu görüntülenmesi gibi önce veri gösterimi sonra rezervasyonun silinerek iptal edilmesi gibi bir veri güncellemesi örnek gösterilebilir.

Gereksinim belirlemesi için çeşitli teknikler, birkaç işlemin birleşiminden oluşan **işlem süreçleri**nin belirlenmesi için işaret gösterimi içerirler. İşlem belirlemesindeki diğer öneriler ise TAXIS (Mylopolous, J., Bernstein, P. ve Wong, H., 1980), GALILEO (Albano, A., Cardelli, L. ve Orsini, R., 1985) ve GORDAS'dır (Elmasri, R., ve Wiederhold, G., 1981). İşlem tasarımı aynen şema tasarımı kadar önemlidir. Maalesef, yürürlükteki pek çok yöntembilimler birini öbürüne üstün görmektedirler. Aşama 2a ve Aşama 2b geri besleme döngüleri ve iyileştirmeler kullanılarak kararlı bir şema ve işlem tasarımına ulaşılana dek paralel olarak devam etmelidir (Elmasri, R. ve Navathe, S., 1995).

2.3 Aşama 3: Veritabanı Yönetim Sisteminin Seçimi

VTYS' nin seçiminde birkaç etmen vardır. Bazı etmenler teknik, bazıları ekonomik diğer bazıları ise kuruluş politikası ile ilgilidirler. Teknik etmenler, veritabanı yönetim sisteminin eldeki göreve uygunluğu ile ilgilenirler. Bu başlık altında şu hususlar değerlendirilir; veritabanı yönetim sistemi türü (ilişkisel, ağ, sıradüzensel, nesneye yönelik, vd.), veritabanı yönetim sisteminin desteklediği depo yapıları, erişim yolları, uygun kullanıcı ve yazılımcı ara yüzleri, yüksek seviye sorgu dili türleri vb. Bu bölümde veritabanı yönetim sistemi seçimini etkileyen ekonomik ve kuruluşla ilgili etmenler üzerinde durulacaktır. Veritabanı yönetim sistemi seçiminde aşağıdaki maliyetler dikkate alınmalıdır (Elmasri, R. ve Navathe, S., 1995):

1. Yazılım alım maliyeti: Dil seçenekleri, formlar, ekranlar gibi değişik ara yüzler, yedekleme / kurtarma seçenekleri, özel erişim yöntemleri ve belgeleme yetenekleri ile birlikte yazılımın alınmasındaki başlangıç maliyetidir.
2. Bakım maliyeti: Veritabanı yönetim sistemini güncel tutmak için satıcının sürekli sağlaması gereken standart bakım hizmetidir.
3. Donanım alım maliyeti: Hafıza, uçbirim (terminal), disk birimi veya özel veritabanı yönetim sistemi belleği gibi yeni donanım gereksinimi olabilir.

4. Veritabanı yaratımı ve dönüşümü maliyeti: Veritabanı sistemini sıfırdan yaratma ya da varolan sistemi yeni veritabanı yönetim sistemi yazılımına çevirme maliyetidir.
5. Personel maliyeti: Veritabanı yönetim sistemi yazılımının bir kuruluşa ilk alımını genellikle veri işleyen birimlerin yeniden örgütlenmesi izler. Veritabanı yönetim sistemini yeni edinen pek çok kuruluşta veritabanı yönetimi çalışanları için yeni kadrolar açılır.
6. Eğitim maliyeti: Veritabanı yönetim sistemlerinin çoğunlukla karmaşık sistemler olması nedeniyle veritabanı yönetim sistemini kullanmak ve programlamak için çoğu personelin eğitimi gerekir.
7. İşletim maliyeti: Veritabanı sisteminin sürekli işletim maliyeti, seçilen veritabanı yönetim sisteminden bağımsız olduğundan, genelde seçeneklerin değerlendirmesi gibi bir şey söz konusu değildir.

Veritabanı yönetim sistemi alımının yararları kolay ölçülüp nicelenemez. Veritabanı yönetim sisteminin geleneksel dosya sistemlerine göre kullanım kolaylığı, verinin daha yaygın kullanılabilmesi ve bilgiye daha hızlı ulaşım gibi, soyut avantajları vardır. Daha somut yararlar ise uygulama geliştirme maliyetinin azlığı, gereksiz (redundant) veri kullanımının azlığı ve daha iyi denetim ve güvenlik olarak sıralanabilir. Kuruluşlar yapacakları kar – maliyet çözümlemelerine göre veritabanı yönetim sistemine geçiş zamanına karar vermelidirler. Bu geçişi genellikle aşağıdaki etmenler etkiler;

- Veri karmaşıklığı: Veri ilişkileri karmaşıklaştıkça veritabanı yönetim sistemine olan ihtiyaç güçlenir.
- Uygulamalar arası paylaşım: Uygulamalar arası paylaşım arttıkça dosyalardaki gereksiz veri artar, dolayısıyla veritabanı yönetim sistemi gereksinimi artar.
- Dinamik olarak büyüyen ve gelişen veri: Veri sürekli olarak değişiyorsa veritabanı yönetim sistemi kullanarak bu değişikliklerle başa çıkmak dosya sistemlerine oranla çok daha kolaydır.
- Anlık veri gereksinimlerinin sıklığı: Dosya sistemleri anlık veri istemine kesinlikle uygun değildir.
- Veri hacmi ve denetim gereksinimi: Yalnızca veri hacmi ve denetim gereksinimi bile bazen veritabanı yönetim sistemini gerektirir.

Son olarak, bazı ekonomik ve kuruluşla ilgili etmenler veritabanı yönetim sistemleri arasındaki seçimi etkilemektedir. Aşağıda bu etmenler belirtilmektedir:

1. Veri yapısı: Saklanacak veri sıradüzensel bir yapı içindeyse sıradüzensel türde bir veritabanı yönetim sisteminin alınması düşünülmelidir. Pek çok ara ilişkisi olan veriler için ağ ya da ilişkisel sistem daha uygun olabilir. Karmaşık veri yapıları ve veri türleri için nesneye yönelik sistemler uygun olabilir.
2. Personelin sisteme aşinalığı: Kuruluştaki yazılımcılar belli bir veritabanı yönetim sistemine aşina iseler, eğitim maliyetini ve öğrenme zamanını azaltmak için o sistem tercih edilebilir.
3. Satış hizmetlerinin uygunluğu: Satış hizmet olanaklarının yakında olması sistemle ilgili sorunların çözümüne yardım açısından arzu edilir. VTYS olmayan bir çevreden VTYS olan bir çevreye geçiş genellikle büyük bir yüktür ve başlangıçta satıcı yardımını çokça gerektirir.

Bir veritabanı yönetim sistemi satın alınmadan önce, veritabanı yönetim sistemi için gereken donanım / yazılım konfigürasyonu ve veritabanı yönetim sisteminin farklı donanım türlerine uygunluğu dikkate alınmalıdır. Çok sayıda ticari veritabanı yönetim sistemi yazılımı farklı donanım / yazılım platformlarda çalışabilen sürümlere sahiptirler. Ayrıca yedekleme, kurtarma, başarım, bütünlük ve güvenlik için uygulama gereksinimleri de göz önünde tutulmalıdır. Çok sayıda veritabanı yönetim sistemleri hali hazırda kuruluşlardaki bilgi – işleme ve bilgi – kaynak yönetimi gereksinimlerine *toplam çözüm* olacak şekilde tasarlanmaktadır. Çoğu veritabanı yönetim sistemi satıcısı ürünlerini 4GL (fourth generation language, dördüncü kuşak diller) olarak kategorize ettikleri aşağıdaki seçeneklerle veya yerleşik (built – in) özelliklerle birleştirmektedirler:

- Metin düzenleyicileri ve göz atıcıları
- Rapor üreticiler ve listeleme kolaylıkları
- İletişim yazılımları
- Otomatik düzenleme yetenekleri olan form, ekran ve menüler gibi veri giriş ve görüntüleme özellikleri
- Grafik tasarım araçları

Bazı durumlarda bir veritabanı yönetim sistemi kullanmak uygun olmayabilir, bunun yerine uygulamalar için kuruluş içi bir yazılım geliştirilmesi tercih edilebilir. Bu ise

ancak uygulamalar önceden çok iyi tanımlı ve tümü biliniyorsa mümkündür. Böyle bir durumda, kuruluş içi, özel tasarım bir sistem bilinen uygulamaların en etkin şekilde gerçekleştirilmesi için uygun olabilir. Bununla birlikte, çoğu durumda tasarım sürecinde öngörülmeyen yeni uygulamalar sistem uygulanmaya başladıktan sonra ortaya çıkar. Veritabanı yönetim sistemlerinin yaygınlaşmasının asıl sebebi de budur; yeni uygulamaların hali hazır sisteme eklenmesini kolaylaştırırlar.

2.4 Aşama 4: Veri Modeli Dönüşümü

Veritabanı tasarımının sonraki adımı kavramsal bir şema ve seçili veritabanı yönetim sisteminin veri modeli cinsinden dış şemalar yaratma işlemidir. Bu da Aşama 2a'da üretilen kavramsal ve dış şemaların yüksek seviye veri modelinden veritabanı yönetim sisteminin veri modeline dönüştürülmesidir. Dönüşüm iki evrede ilerler:

1. Sistem – bağımsız dönüşüm: Bu adımda veritabanı yönetim sisteminin veri modeline dönüşüm sırasında, veri modelinin veritabanı yönetim sistemine uygulanmasında hiçbir özel durum veya özellik dikkate alınmaz. Sistemden bağımsız olara Vİ şemasından ilişkisel, nesneye yönelik, sıradüzensel veya ağ şemasına dönüşümü anlatan kaynaklar da mevcuttur (Elmasri, R. ve Navathe, S., 1995).
2. Şemaların belli bir veritabanı yönetim sistemine dönüşümü: Farklı veritabanı yönetim sistemleri özel modelleme özellikleri ve koşulları kullanarak veri modelini uygularlar. Adım 1'de elde edilen şemaları seçili veritabanı yönetim sisteminde kullanılan veri modelinin özelliklerine uydurmak için yeniden düzenlemek gerekebilir.

Dönüşüm sonucu elde edilen tablolar "Normalizasyon"a tabi tutulur. Bu aşamanın sonucu seçili veritabanı yönetim sistemi dilinden kavramsal ve dış düzeyde şemaları tanımlayan DDL (Veri Tanımlama Dili – Data Definition Language) komutları olmalıdır. Fakat DDL komutları bazı fiziksel tasarım ölçütleri içeriyorsa tam bir DDL tanımlaması, fiziksel veritabanı tasarımı evresi sonuçlanana kadar beklemelidir.

2.5 Aşama 5: Fiziksel Veritabanı Tasarımı

Fiziksel veritabanı tasarımı, çeşitli veritabanı uygulamalarında iyi bir başarıma ulaşabilmek için veritabanı dosyalarındaki depolama yapıları ve erişim yollarının seçilmesi işlemidir. Her veritabanı yönetim sistemi dosya organizasyonu ve erişim yolları için değişik seçenekler sunmaktadır. Bunlar genellikle dizinleme türleri, ilgili kayıtların disk üzerinde kümelenmesi, ilgili kayıtların işretçiler ile bağlanması ve çeşitli sıralama (hashing) türleridir. Belli bir veritabanı yönetim sistemi seçildikten sonra fiziksel veritabanı tasarımı işlemi veritabanı dosyalarının yapısı için veritabanı yönetim sisteminin sunduğu seçenekler içinden en uygun olanının seçimi ile sınırlıdır. Aşağıdaki ölçütler fiziksel veritabanı tasarım seçeneklerinin seçiminde kılavuzluk ederler:

1. Yanıtlama süresi: Bir veritabanı işleminin gönderilmesi ve yanıtın alınması arasındaki zamandır. Veritabanı yönetim sistemi denetimindeki yanıtlama süresindeki ana etki, işlemin başvurduğu veri öğeleri için veritabanı erişim zamanıdır. Yanıtlama zamanı veritabanı yönetim sistemi denetimde olmayan sistem yüklenmesi, işletim sistemi zamanlaması veya iletişim gecikmeleri gibi etmenlerden de etkilenir.
2. Alan kullanımı: Veritabanı dosyaları ve erişim yol yapıları tarafından kullanılan depolama alanı miktarıdır.
3. İşlem ürünü: Veritabanı sistemi tarafından dakikada işlenebilen ortalama işlem sayısıdır; havayolu rezervasyon ve banka gibi işlem sistemlerinde kritik bir ölçüttür.

Genel olarak, anlatılan ölçütlerdeki ortalama ve en kötü durum sınırları, sistem başarım gereksinimlerinin bir parçası olarak belirlenir. Prototiplerin oluşturulması ve benzetim gibi çözümsel veya deneysel teknikler, belirli başarım gereksinimlerinin karşılanıp karşılanmadığının belirlenmesi için çeşitli fiziksel tasarım kararlarında ortalama ve en kötü durum değerlerinin tahmininde kullanılırlar.

Başarı, kayıt büyüklüğüne ve dosyadaki kayıt sayısına bağlıdır. Bu yüzden her dosya için bu ölçütlerin tahmin edilmesi gerekir. Ek olarak, dosyanın tüm işlemlerden birikimli olarak güncelleme ve erişim şekilleri tahmin edilmelidir. Kayıtların seçimi için kullanılan özniteliklerin birincil erişim yolları ve ikincil dizinleri olmalıdır. Yeni

özniteliklerin eklenmesinden ya da kayıtların artmasından kaynaklanan dosya büyümesinin tahmini de fiziksel veritabanı tasarımı sırasında hesaba katılmalıdır.

Fiziksel veritabanı tasarımının sonucu veritabanı dosyalarına ait depo yapılarının ve erişim yollarının *ilk* belirlemesidir. Veritabanı uygulanmaya başladıktan sonra veritabanı sisteminin ölçülen başarımı temel alınarak, tasarımda ince ayarlar hemen hemen her zaman gereklidir. Pek çok sistem, sonraki çözümlemelerde kullanılmak üzere sistem katalog ve veri sözlüklerinde tutulan başarım istatistikleri toplamak için bir izleme kolaylık yazılımı içerir. Bu istatistikler önceden tanımlı işlem veya sorgulamaların çağrılma sayıları, dosyalardaki girdi / çıktı etkinlikleri, dosya sayfalarının ya da dizin kayıtlarının sayısı ve dizin kullanımının sıklığı gibi değerlerdir. Veritabanı sistemi gereksinimleri değiştikçe, bazı dosyaları yeni dizinler oluşturarak veya birincil erişim yöntemlerini değiştirerek yeniden düzenlemek gerekir.

2.6 Aşama 6: Veritabanı Sisteminin Uygulanması

Mantıksal ve fiziksel tasarımlar tamamlandıktan sonra, veritabanı sistemi uygulamaya konabilir. Seçili veritabanı yönetim sisteminin DDL (Veri Tanımlama Dili – Data Definition Language) ve SDL (Depo Tanımlama Dili – Storage Definition Language) komutları derlenir ve veritabanı şema ve (boş) veritabanı dosyalarını yaratmak için kullanılır. Bundan sonra veritabanı veriyle **yüklenebilir**. Eğer veri önceki bir bilgisayarlı sistemden dönüştürülecekse, veriyi yeniden düzenlemek ve yeni veritabanına yüklemek için **dönüşüm teknikleri** gerekebilir.

Bu evrede veritabanı işlemleri uygulama yazılımcıları tarafından uygulanmalıdır. İşlemlerin kavramsal tanımlamaları incelenir ve bunlara ilişkin gömülü DML (Veri İşleme Dili – Data Manipulation Language) komutları ile yazılım kodları yazılır ve sınanır. İşlemler hazır olduğunda ve veri yüklendiğinde, tasarım ve uygulama evresi bitmiş ve veritabanı sisteminin işletim evresi başlamış demektir.

3. UYGULAMA (JEODEZİK VERİTABANI)

Bu bölümde anlatılan aşamalara ait yapılan Jeodezik Veritabanı uygulamasından bahsedilecektir.

3.1 Aşama 1: Gereksinimlerin Derlenmesi ve Çözümlenmesi

Bir örneği aşağıda verilen ankete benzer anketler, anahtar personele verilmiş ve veri alışverişinde kullanılan formlar (EK – A) seçilen anketörlerle görüşülerek tespit edilmiştir:

1. Nirengilere ait konum bilgileri nelerdir?

Enlem ve boylam (coğrafi koordinatlar). Bunlar UTM (sağa, yukarı ve dilim orta meridyeni – dom) cinsinden de ifade edilir. Ayrıca yükseklik bilgisi de tutulur.

2. Bu bilgiler nasıl üretilir?

Arazide yapılan açı (yatay ve düşey) ve mesafe ölçümlerinin dengelenmesiyle üretilir.

3. Bu bilgileri hangi idari birim üretir?

Jeodezi Şubesi.

4. Bu bilgileri hangi idari birim sunar?

Jeodezik ve Jeofizik Değerler Şubesi (JJDŞ).

5. Bir nirengi noktası tek anlamlı olarak nasıl tanımlanır?

İçinde bulunduğu 1 / 100.000 ölçekli paftanın adı ve nokta numarası ile.

6. Bu bilgiler kimlere sunulur?

Fotogrametri Dairesi,Tapu ve Kadastro Gn. Md.lüğü, İller Bankası, firmalar, diğer bazı KİTler.

7. Nirengi değerleri ile ilgili konumdan başka ne tür bilgiler, hangi formatta tutulmalıdır?

   - Nokta numarası (101)

- İçinde bulunduğu pafta (Elbistan K36 –a1)
- Nokta derecesi (I, II, III, IV)
- Kot verilme türü (Trigonometrik, Nivelman)
- Tesis edilme türü (Beton Blok, Pilye, Bina Üstü, Kayada Bronz, Minare)
- Tesis yılı (1966)

8. Bu bilgileri hangi tarzda istenmektedir?

- Adı verilen bir pafta içine giren belirli nitelikte (örneğin; IInci derece, Beton Blok) noktalar
- 1 / 100.000 ölçekli pafta adı ve nokta numarası verilmek suretiyle istenen noktalar
- Köşe noktaları ile tanımlanan bir alan içine giren noktalar
- Konumu tanımlanan bir noktaya belli bir uzaklıktaki noktalar
- Pafta üzerinden görerek seçilen noktalar.

9. Bu bilgilerin sunumunda ne tür formlar kullanılır?

EK – A' de verilen formlar ve protokoller kullanılır.

10. Nirengilere ait sizin tutulması gerektiğini düşündüğünüz başka bilgi var mıdır?

Mevcut sistemde araziye çıkanlardan bazı noktaların tahrip olduğuyla ilgili bilgi gelmekte fakat bu bilgi saklanmamaktadır. Bu bilginin de tutulmasının yararlı olduğu değerlendirilmektedir.

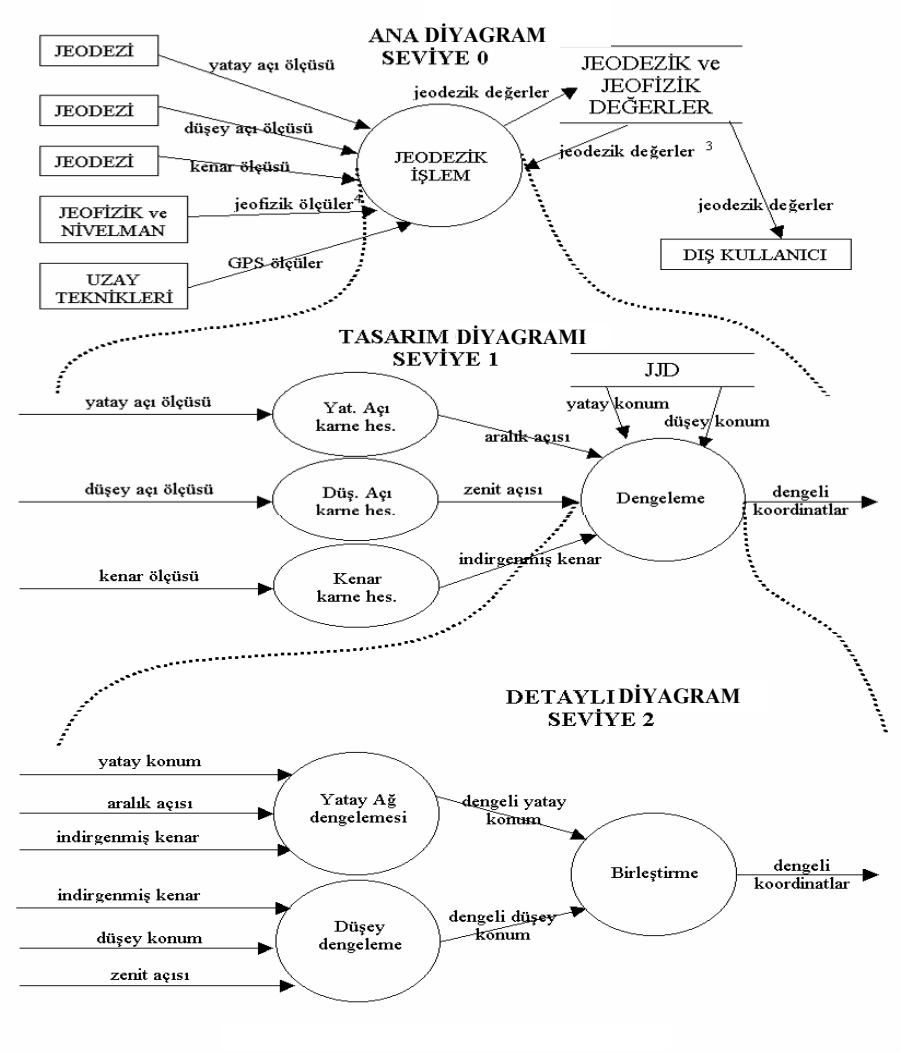
Bu kapsamda ***Şekil 3-1***' deki VAD'ları oluşturulmuştur.

3.2 Aşama 2: Kavramsal Veritabanı Tasarımı

Bölüm 2.2' de anlatıldığı gibi veritabanı tasarımının ikinci aşaması iki paralel etkinlik içerir. Bu bağlamda hazırlanan Varlık – İlişki diyagramları EK – B' de sunulmuştur. Aşama 2b'de değinilen işlem tasarımı için de istemsel işleme örnek teşkil edebilecek yukarıdaki anketin sekizinci sorusuna verilen cevaplardaki gereksinimlerin bazılarını karşılamak üzere aşağıdaki SQL komutları tasarlanmıştır:

- Adı verilen bir pafta içine giren belirli nitelikte (örneğin; 2. derece, Beton Blok) noktalar:

```
select  * from nirengi
where  paf250 = 'Elbistan' and
paf100 = 'K36' and
paf50 = 'a' and
paf25 = '1' and
tesis_türü = '1' and
nokder = '2'
```



Şekil 3-1 JVT Veri akış diyagramları (nirengi)

[4] Jeodezik değerler : yatay konum, GPS, yükseklik, gravite, manyetik değerleri
[5] Jeofizik ölçüler : yükseklik, gravite, manyetik ölçüleri

- 1/ 100.000 ölçekli pafta adı ve nokta numarası verilmek suretiyle istenen noktalar

  select * from nirengi where paf100 = 'K36' and nok_no = '101'

3.3 Aşama 3: Veritabanı Yönetim Sisteminin Seçimi

Oracle 10g VTYS yazılımının gerçek uygulamada kullanılması ilgili kurumca tasarlanmaktadır. Ancak geliştirilen uygulamanın taşınabilir olması ve e–Devlet kapsamında kullanılan verilerin inernetten çalınma riski de göz önüne alınarak güvenliği sağlamak için bu çalışmada veritabanı seyreltilip değerleri kabalaştırılarak Microsoft ACCESS 2000'e dönüştürülmüştür.

3.4 Aşama 4: Veri Modeli Dönüşümü

Bu aşamada Vİ diyagramlarından tablolar üretilmiştir. Bunun için aşağıdaki adımlar uygulanmıştır Burada f.k. (foreign key) yabancı anahtar, TUTGA Türkiye Ulusal Temel GPS Ağı noktası anlamına gelmekte olup birincil anahtarların da altı çizilidir. Anlaşılırlığını arttırabilmek için gerekli görülen özniteliklerin yanına alabilecekleri örnek değerler tırnak içinde ("değer") şeklinde yazılmıştır.:

1. Zayıf olmayan her bir varlık için tüm özniteliklerini içeren bir tablo yaratılır ve birincil anahtar olacak öznitelik belirlenir.

- Personel [sicil, adı, soyadı] ● Bölgesi [kod, bölge]
- Nirengi [paf100 "İ29", nokno "101", paf250 "ANKARA", paf50 "b", paf25 "2", sağa "412 321,12", yukarı "4 255 237,63", dom "33", eder "36", edak "12", esan "21.1234", bder, bdak, bsan, yük, tesyıl, nokder "3"]
- Nivelman [hatno "403A", nokno "62", paf250, paf100, paf50, paf25, eder, edak, esan, bder, bdak, bsan, yük, tesyıl, nokder "düğüm", gravite "9,8061113"]
- TUTGA [nokno, paf250, paf100, paf50, paf25, eder, edak, esan, bder, bdak, bsan, yük, X, Y, Z, tesyıl, nokder]
- Kot Türü [kod, tür] ● Nir. Tes. Türü [kod, tür] ● GPS Tes. Türü [kod, tür]

- Datum [kod, datum] ● Nitelik [kod, niteliği]

2. Her bir zayıf varlık için tüm özniteliklerini içeren ayrıca sahibinin birincil anahtarını yabancı anahtar olarak alan bir tablo yaratılır. Sahibinin ve varsa kendisinin birincil anahtarlarının birleşimi tablonun birincil anahtarı olur.

   - NirÖlçü [PerSicil(f.k), Tarih, YDurNok, YBakNok1, YBakNok2, AraAçı, DDurNok, DBakNok, Zenit, KDurNok, KBakNok, Mesafe]
   - NivÖlçü [PerSicil(f.k), Tarih, Yükseklik, Mesafe]
   - GPSÖlçü [PerSicil(f.k), Tarih, Sinyal, AletYüksekiği]
   - Protokol [Anahtar, NirPaf100(f.k.), Nirnokno(f.k.), NivHatNo(f.k.), Nivnokno(f.k.), GPSNokNo, tarif, kroki, kaba değer]

3. Tüm bire bir ilişkilerde ilişkinin bir tarafındaki varlığa K diğerine H denirse (tam katılımı olana K demek daha doğru olur), H'nin birincil anahtarı K'ye yabancı anahtar yapılır.

   - Nirengi [paf100, nokno, paf250, paf50, paf25, sağa, yukarı, dom, eder, edak, esan, bder, bdak, bsan, yük, tesyıl, drc, KotKod(f.k.), TesKod(f.k.)]
   - Nivelman [HatNo, nokno, paf250, paf50, paf25, eder, edak, esan, bder, bdak, bsan, yük, tesyıl, drc, BölgeKod(f.k.)]
   - TUTGA [nokno, paf250, paf100, paf50, paf25, eder, edak, esan, bder, bdak, bsan, yük, X, Y, Z, tesyıl, drc, DatumKod(f.k.), NitelikKod(f.k.), TesKod(f.k.)]

4. Tüm bire çok ilişkilerde ilişkiye çoklu (N) olarak katılan varlığa K diğerine H denirse, H'nin birincil anahtarı K'ya yabancı anahtar olarak eklenir.

   - Protokol [Anahtar, NirPaf100(f.k.), Nirnokno(f.k.), NivHatNo(f.k.), Nivnokno(f.k.), GPSNok_no, tarif, kroki, kabadeğer, PerSicil(f.k.)]

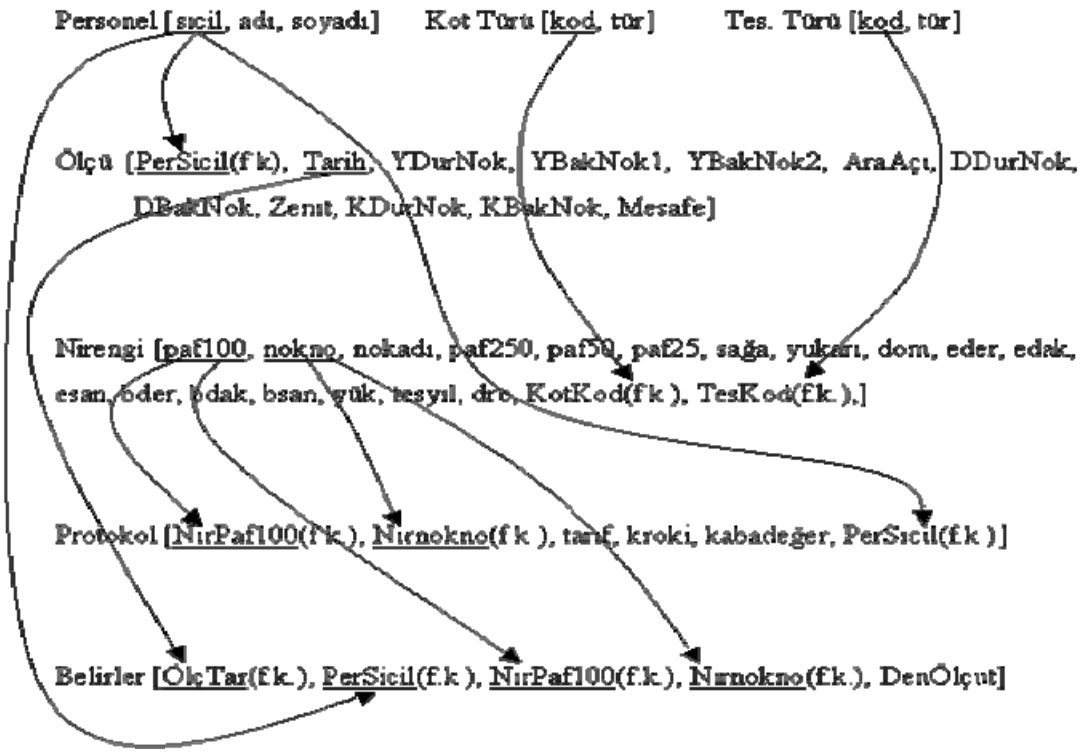
5. Tüm çoka çok ilişkiler için bir tablo yaratılarak ilişkiye katılan tüm varlıkların birincil anahtarları bu tabloya yabancı anahtar olarak eklenir ve bunların birleşimi tablonun birincil anahtarı olur. Varsa ilişkinin öznitelikleri de bu tabloya eklenir.

   - NirBelirler [ÖlçTar(f.k.), PerSicil(f.k.), NirPaf100(f.k.), Nirnokno(f.k.), DenÖlçüt]

- NivBelirler [ÖlçTar(f.k.), PerSicil(f.k.), NivHatNo(f.k.), Nivnokno(f.k.), DenÖlçüt]
- GPSBelirler [ÖlçTar(f.k.), PerSicil(f.k.), GPSnokno(f.k.), DenÖlçüt]

6. Tüm çoklu değerli öznitelikler için bir tablo yaratılır. Çoklu değerin ait olduğu tablonun birincil anahtarı ve çoklu değerin kendisi bu tabloya öznitelik olarak girer. Bunların birleşimi de birincil anahtar olur. Kullanılan örnekte çoklu değer bulunmamaktadır.
7. Tüm üçlü ilişkiler için bir tablo yaratılarak ilişkiye katılan tüm varlıkların birincil anahtarları bu tabloya yabancı anahtar olarak eklenir ve bunların birleşimi tablonun birincil anahtarı olur. Varsa ilişkinin öznitelikleri de bu tabloya eklenir. Kullanılan örnekte üçlü ilişki bulunmamaktadır.

Nirengi için oluşturulan bu tablolara ait başvurusal bütünlük koşullarını (referential integrity constraints) da içeren bir ilişkisel veritabanı şeması ***Şekil 3-2***'de görülmektedir.



Şekil 3-2 Bütünlük koşullarını içeren bir ilişkisel veritabanı şeması

3.5 Aşama 5: Fiziksel Veritabanı Tasarımı

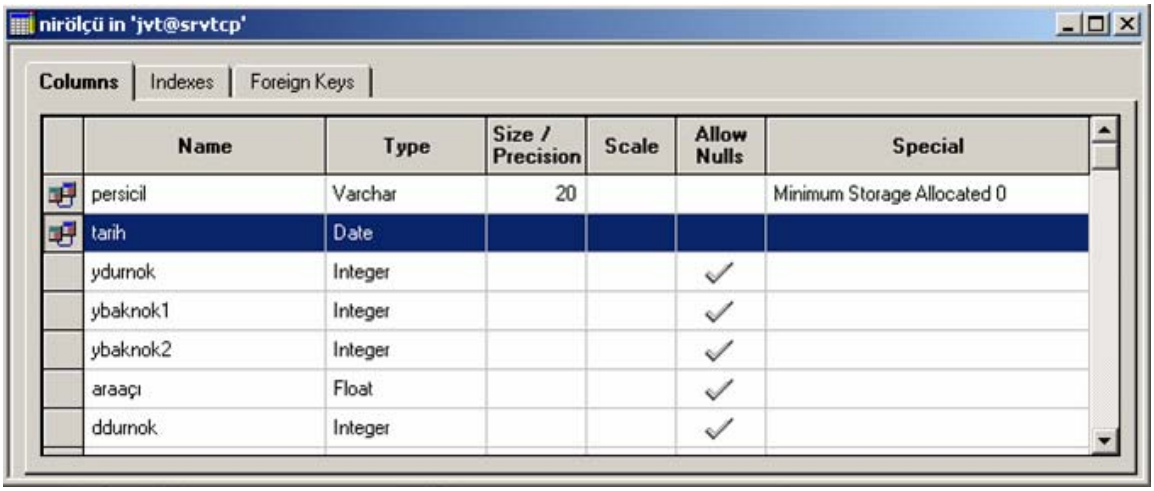
Bölüm 3.4'te oluşturulan kavramsal şema ve tablolar bu aşamada yerel (kurum bünyesindeki kullanılacak olan) uygulamalar için Oracle 10g kullanılarak fiziksel ortamda hazırlanmıştır. Bunlar ***Şekil 3-4***'te görülmektedir. Ancak bu çalışma kapsamında, daha sonra internet ortamında da kurumca kullanılacağı değerlendirilen uygulama tarafından kullanılan verilerin güvenlik sorunu olması nedeniyle en azından başlangıçtaki test süresince, farklı yapı ve uydurma içeriklerde MS ACCESS2000'de hazırlanmış tablolar kullanılmıştır. Bu tablolar da ***Şekil 3-3***'te görülmektedir. Sık kullanılacağı yapılan anketlerle belirlenen sorgulamalara yanıtlama süresini azaltmak için tablolarda indeksleme yapılmıştır. Örneğin tüm tablolardaki paf250, paf100, paf50, paf25 öznitelikleri indekslenmiştir.

nirengi in 'jvt@srvtcp'

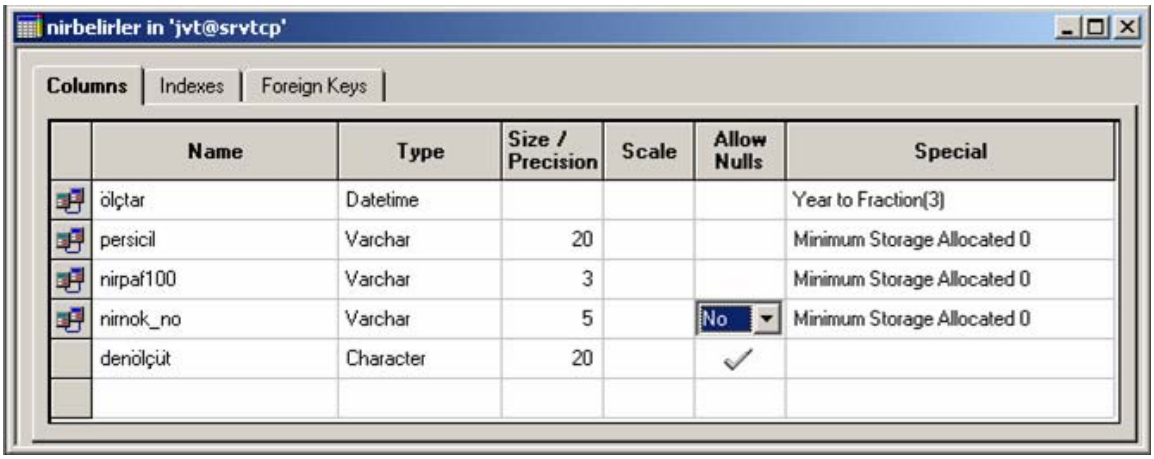
Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	nok_no	Integer				
	paf250	Varchar	20		✓	Minimum Storage Allocated 0
	paf100	Varchar	3			Minimum Storage Allocated 0
	paf50	Varchar	1		✓	Minimum Storage Allocated 0
	paf25	Varchar	1		✓	Minimum Storage Allocated 0
	saga	Float			✓	
	yukarı	Float			✓	
	dom	Integer			✓	
	eder	Integer			✓	
	edak	Integer			✓	
	esan	Float			✓	
	bder	Integer			✓	
	bdak	Integer			✓	
	bsan	Float			✓	
	yük	Float			✓	
	tesyıl	Datetime			✓	Year to Fraction(3)
	kotkod	Integer			✓	
	teskod	Integer			✓	

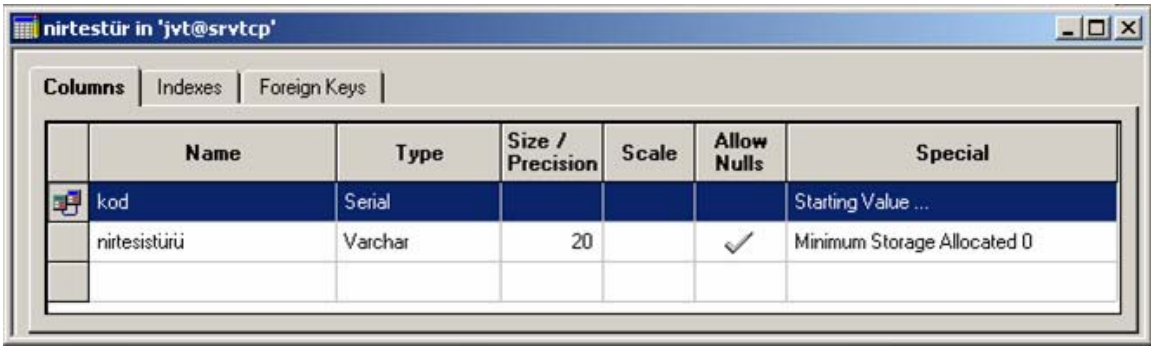
Şekil 3-3a "Nirengi" tablosu tasarım görünümü (kurumsal)



Şekil 3.3b "Nirölçü" tablosu tasarım görünümü (kurumsal)



Şekil 3.3c "Nirbelirler" tablosu tasarım görünümü (kurumsal)



Şekil 3.3ç "Nirtestür" tablosu tasarım görünümü (kurumsal)

kotürü in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	kod	Serial				Starting Value ...
	tür	Varchar	20		✓	Minimum Storage Allocated 0

Şekil 3.3d "Kottürü" tablosu tasarım görünümü (kurumsal)

nivelman in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	hat_no	Integer				
	nok_no	Integer			No	
	paf250	Varchar	20		✓	Minimum Storage Allocated 0
	paf100	Varchar	3		✓	Minimum Storage Allocated 0
	paf50	Varchar	1		✓	Minimum Storage Allocated 0
	paf25	Varchar	1		✓	Minimum Storage Allocated 0
	eder	Integer			✓	
	edak	Integer			✓	
	esan	Float			✓	
	bder	Integer			✓	
	bdak	Integer			✓	
	bsan	Float			✓	
	yük	Float				
	tesyıl	Datetime			✓	Year to Fraction(3)
	drc	Varchar	4		✓	Minimum Storage Allocated 0
	bölgekod	Integer			✓	

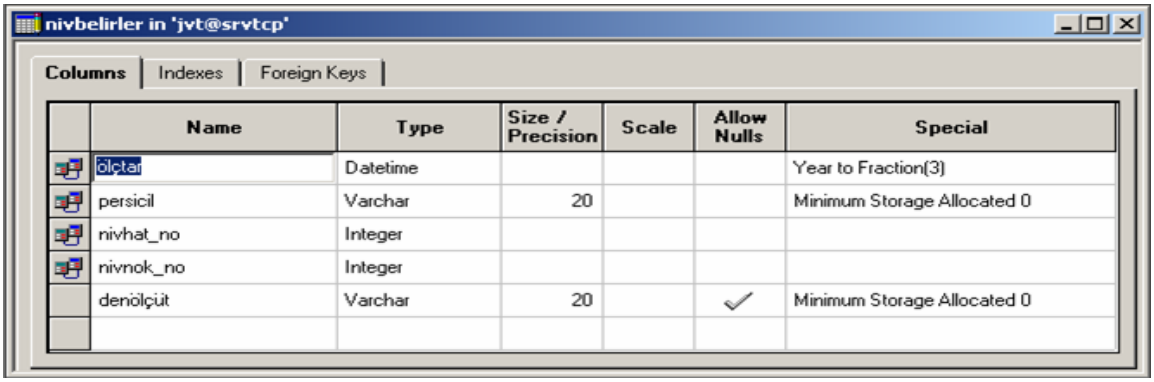
Şekil 3.3e "Nivelman" tablosu tasarım görünümü (kurumsal)

nivölçü in 'jvt@srvtcp'

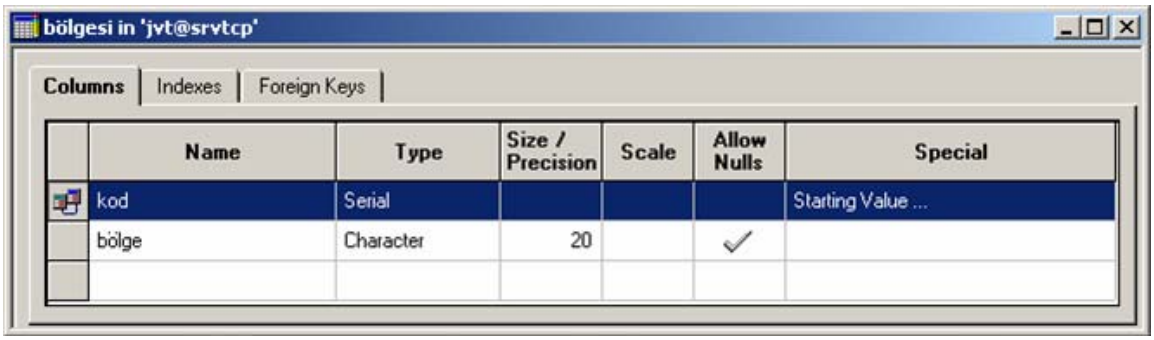
Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	persicil	Character	20			
	tarih	Date				
	yükseklik	Float				
	mesafe	Float			✓	

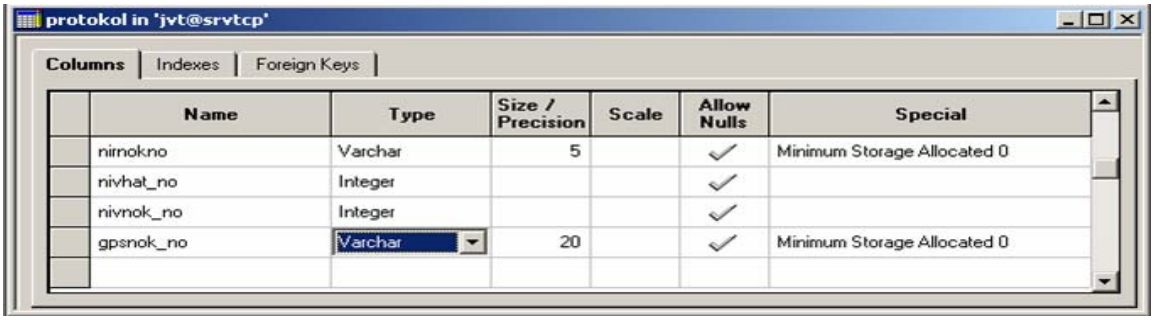
Şekil 3.3f "Nivölçü" tablosu tasarım görünümü (kurumsal)



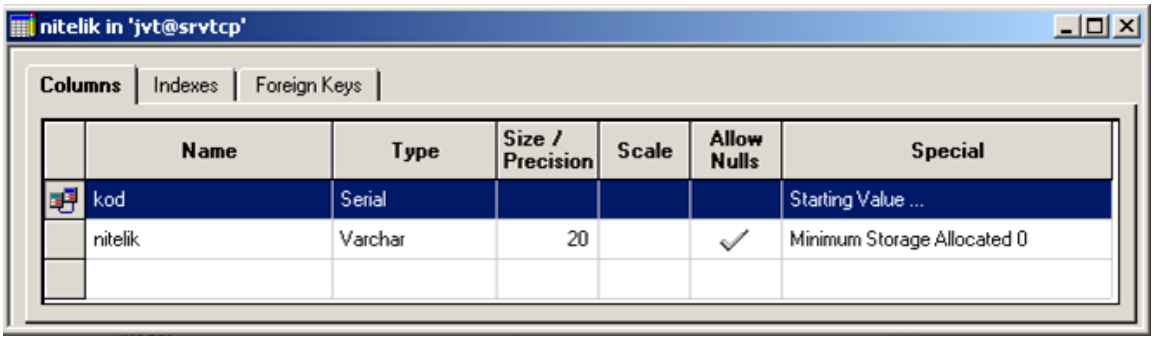
Şekil 3.3g "Nivbelirler" tablosu tasarım görünümü (kurumsal)



Şekil 3.3ğ "Bölgesi" tablosu tasarım görünümü (kurumsal)



Şekil 3.3h "Protokol" tablosu tasarım görünümü (kurumsal)



Şekil 3.3ı "Nitelik" tablosu tasarım görünümü (kurumsal)

tutga in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	nok_no	Varchar	20			Minimum Storage Allocated 0
	paf250	Varchar	20		✓	Minimum Storage Allocated 0
	paf100	Varchar	3		✓	Minimum Storage Allocated 0
	paf50	Varchar	1		✓	Minimum Storage Allocated 0
	paf25	Varchar	1		✓	Minimum Storage Allocated 0
	eder	Integer			✓	
	edek	Integer			✓	
	esan	Float			✓	
	bder	Integer			✓	
	bdak	Integer			✓	
	bsan	Float			✓	
	x	Float			✓	
	y	Float			✓	
	z	Float			✓	
	tesyıl	Datetime			✓	Year to Fraction(3)
	drc	Varchar	4		✓	Minimum Storage Allocated 0
	datumkod	Integer			✓	
	nitelikod	Integer			✓	
	teskod	Float			✓	

Şekil 3.3i "TUTGA" Tablosu Tasarım Görünümü (kurumsal)

gpsölçü in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	persicil	Varchar	20			Minimum Storage Allocated 0
	tarih	Date				
	sinyal	Character	20		✓	
	aletyüksekligi	Float			✓	

Şekil 3.3j "GPSölçü" Tablosu Tasarım Görünümü (kurumsal)

gpsbelirler in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	ölçtar	Datetime				Year to Fraction(3)
	persicil	Varchar	20			Minimum Storage Allocated 0
	gpsnok_no	Varchar	7			Minimum Storage Allocated 0
	denölçüt	Varchar	20		✓	Minimum Storage Allocated 0

Şekil 3.3k "GPSbelirler" Tablosu Tasarım Görünümü (kurumsal)

gpstestür in 'jvt@srvtcp'

Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	kod	Serial				Starting Value ...
	gpstesistürü	Character	20		✓	

Şekil 3.3l "GPStestür" Tablosu Tasarım Görünümü (kurumsal)

datum in 'jvt@srvtcp'

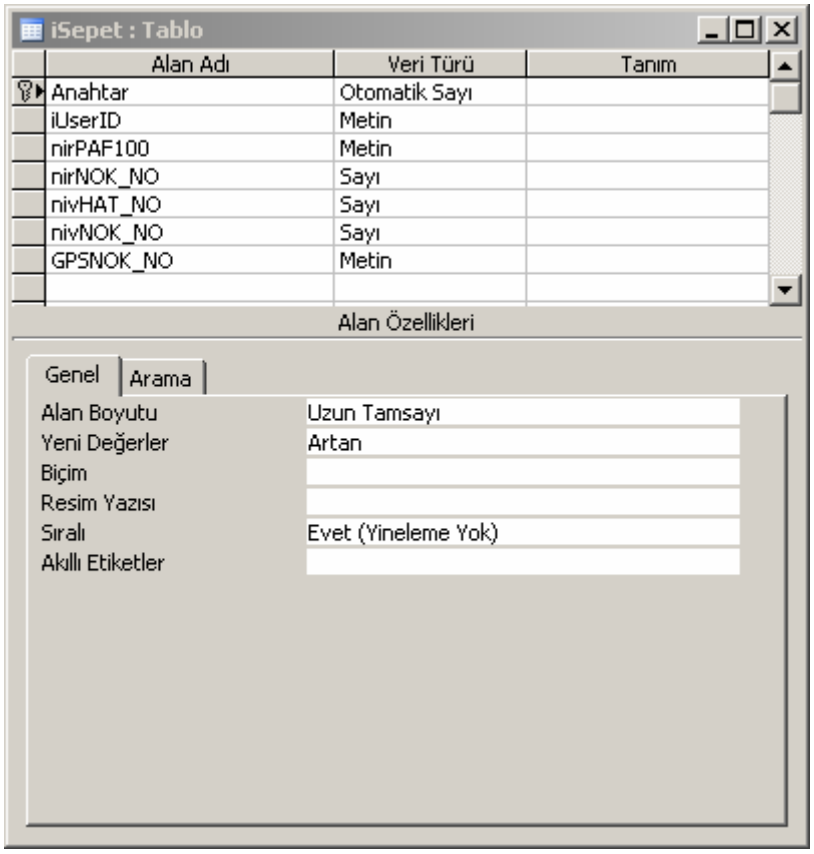
Columns | Indexes | Foreign Keys

	Name	Type	Size / Precision	Scale	Allow Nulls	Special
	kod	Serial				Starting Value ...
	datum	Varchar	20		✓	Minimum Storage Allocated 0

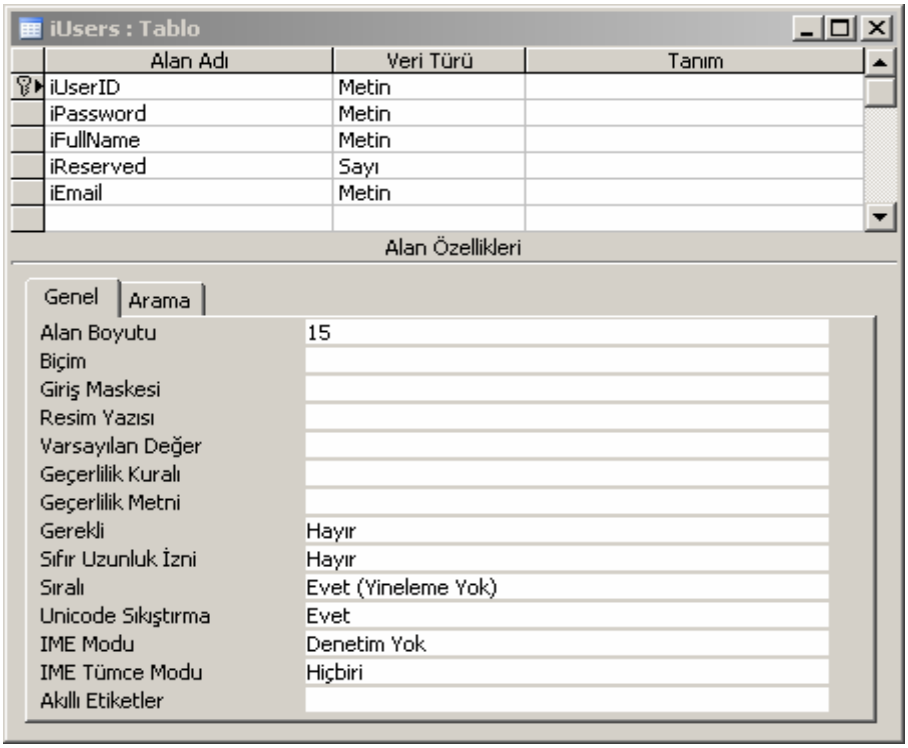
Şekil 3.3m "Datum" Tablosu Tasarım Görünümü (kurumsal)



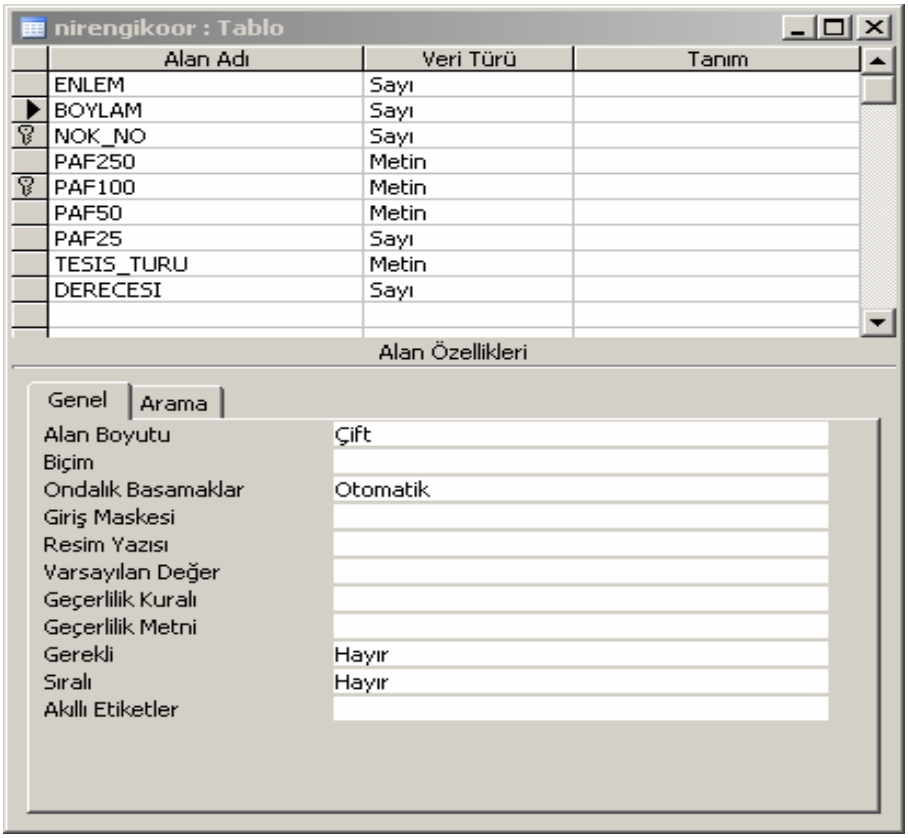
Şekil 3-4a Tablolar (internet)



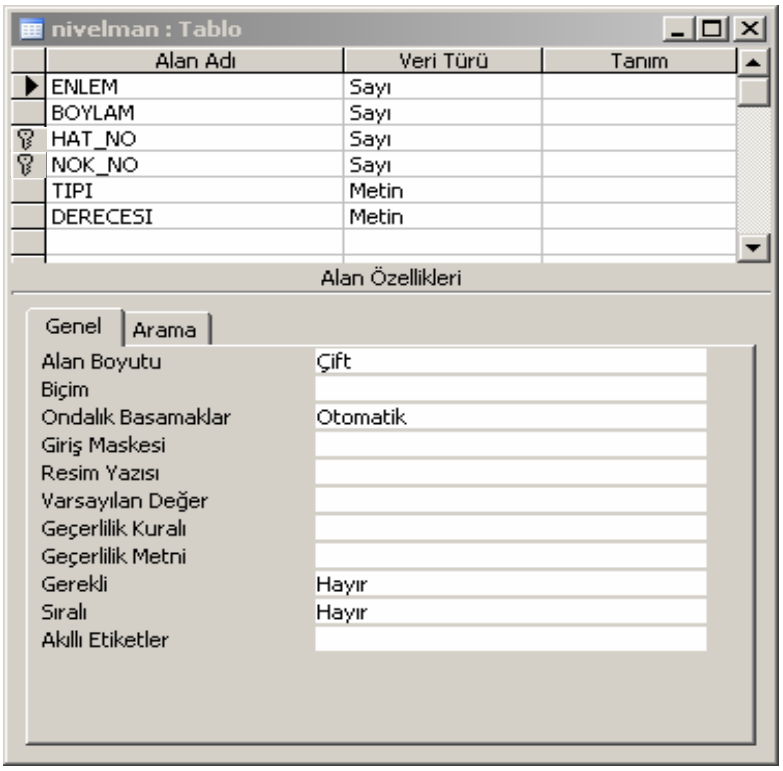
Şekil 3.4b "iSepet" tablosu tasarım görünümü (internet)



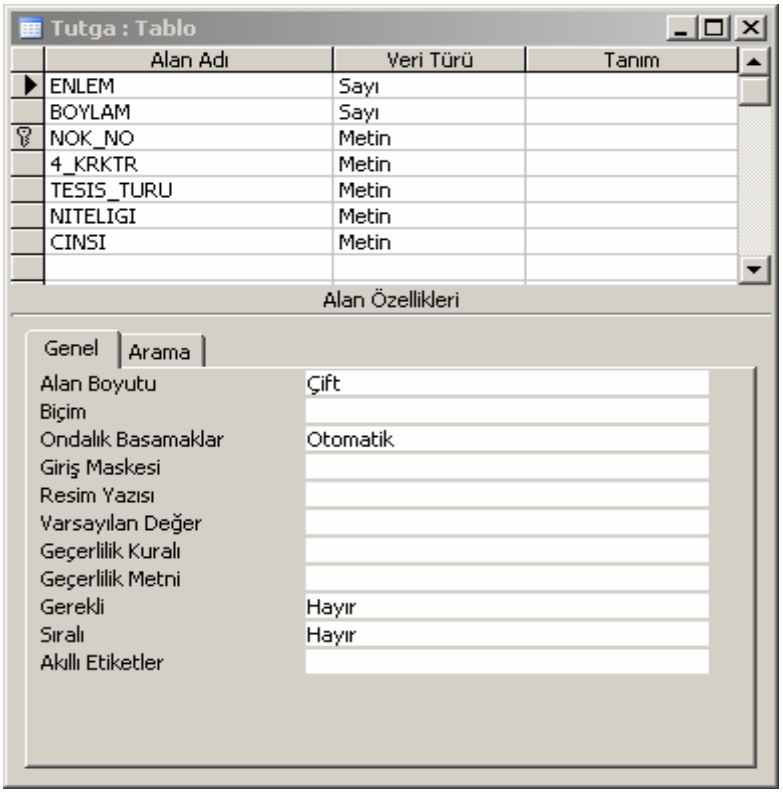
Şekil 3.4c "iUsers" tablosu tasarım görünümü (internet)



Şekil 3.4ç "Nirengikoor" tablosu tasarım görünümü (internet)



Şekil 3.4d "nivelman" tablosu tasarım görünümü (internet)

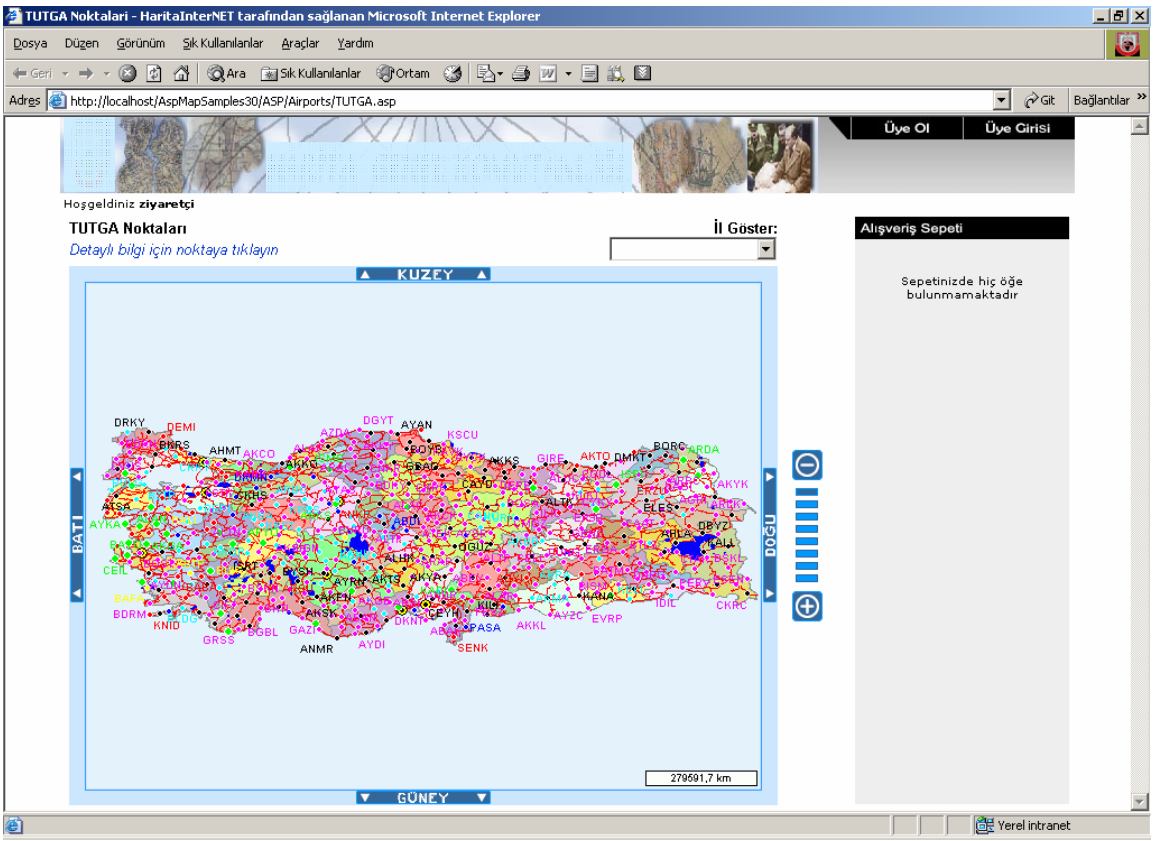


Şekil 3.4e "TUTGA" tablosu tasarım görünümü (internet)

3.6 Aşama 6: Veritabanı Sisteminin Uygulanması

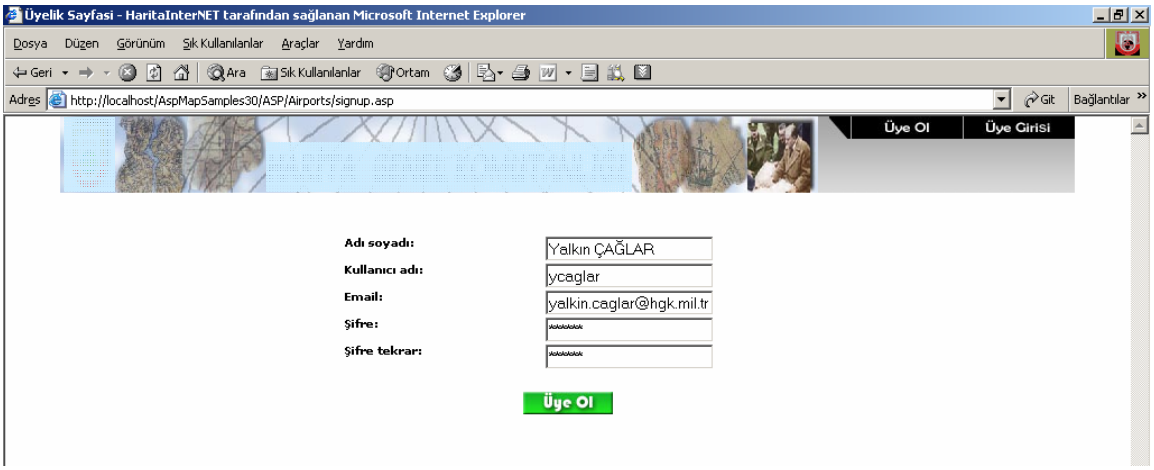
Tüm aşamalar göz önünde bulundurularak web mimarisinde, internetten satışa da hizmet edebilecek yapıda ki kullanıcı ara yüzlerinin ASP ve AspMap 3.0™ kullanılarak hazırlanmış ve uygulama bu şekilde tamamlanmıştır.

Sayfa tasarımlarında kurumun ana sayfasından nirengi, nivelman ve TUTGA değerlerine ayrı ayrı erişilecek bağlar olacağı öngörülmüştür. Tüm sayfalarda il_alan, karayolu, il_merkezi, göl katmanları, ayrıca haritanın büyütme seviyesine bağlı olarak ilce_sinirlari ve ilce_merkezleri katmanları kullanılmış, noktalar bu katmanlar üzerine veritabanından atılmıştır.



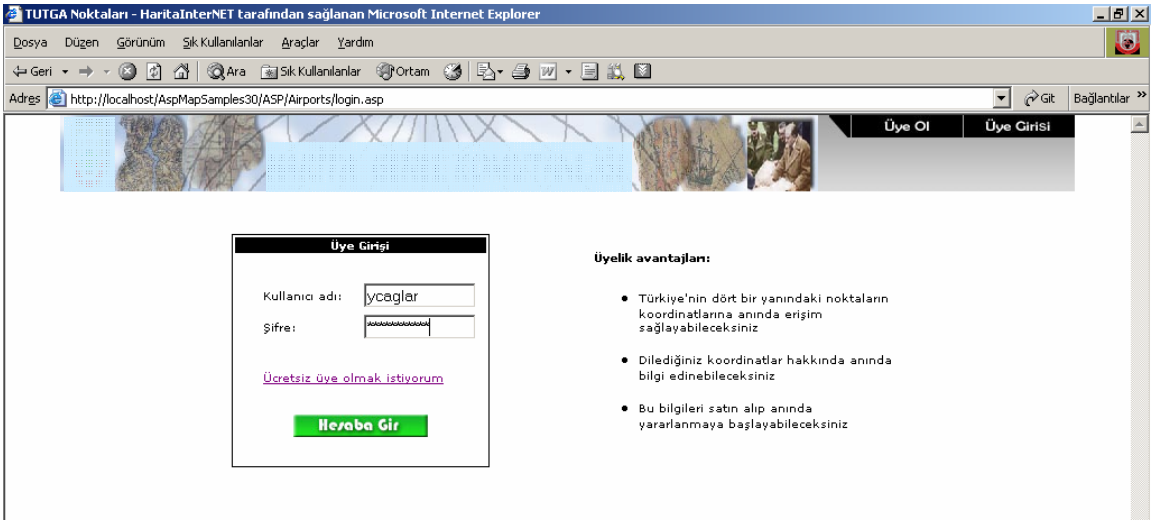
Şekil 3-5 TUTGA sayfası (ziyaretçi)

Yukarıdaki sayfada (***Şekil 3-5***) TUTGA noktaları görülmektedir. Site tasarımında nokta satın alabilmek için üyelik sistemi zorunlu kılınmıştır.



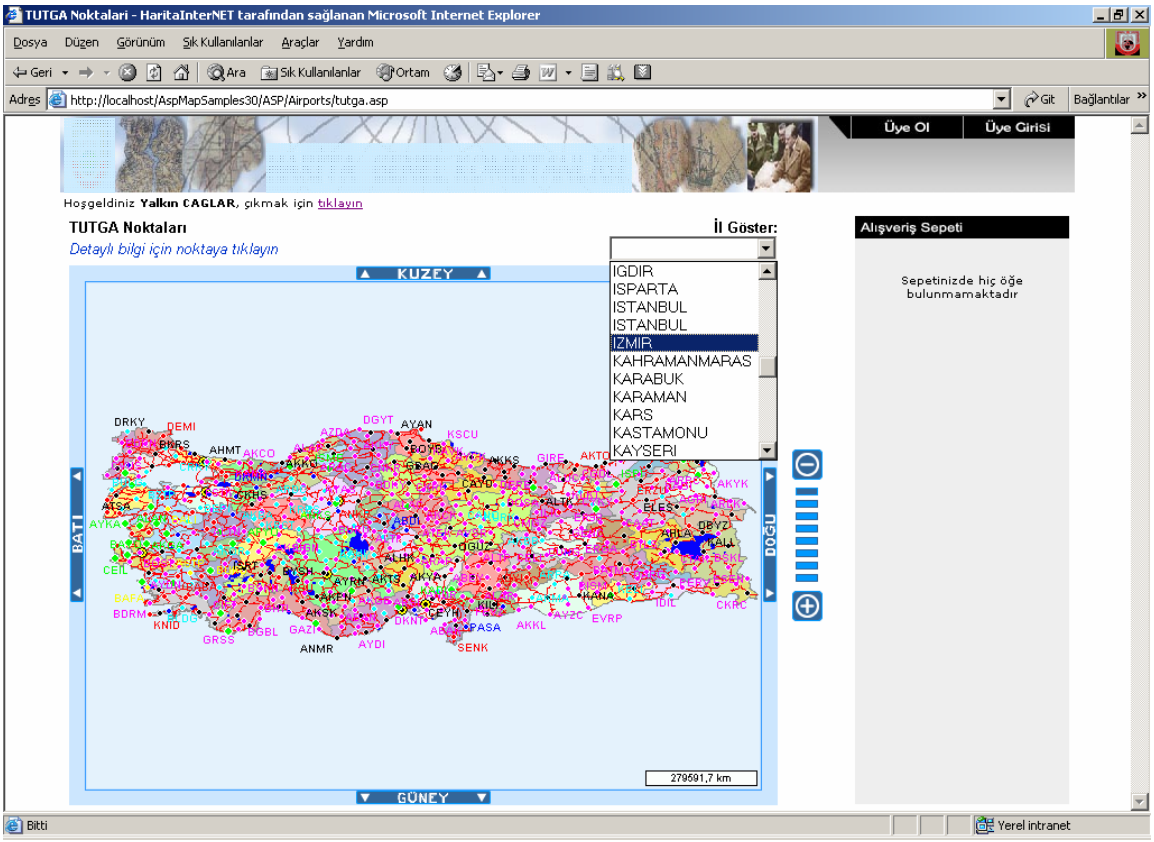
Şekil 3-6 Üyelik bilgi girişi

"Üye Ol" ile gelen "Üyelik bilgi girişi" sayfasında (Şekil 3-6) kullanıcının çeşitli bilgiler girmesi istenmektedir. Bu bilgileri tanımlayan kullanıcı siteye üye olmaktadır.



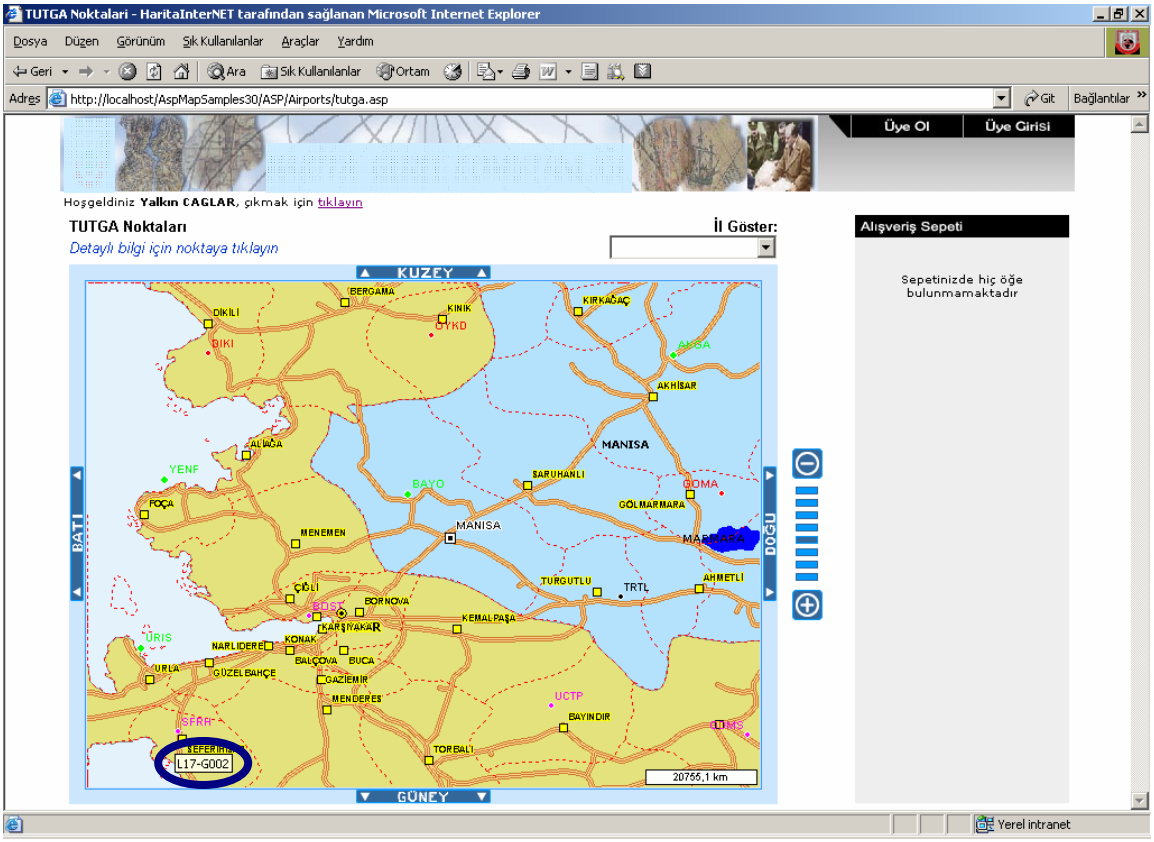
Şekil 3-7 Üye Girişi

Kullanıcılar daha sonra bu bilgilerini kullanarak (Şekil 3-7) siteye girebilmektedirler. Şekil 3-8'de kullanıcı olarak girildiğinde TUTGA sayfası görülmektedir. Bu sayfa üzerinden istenilen İl seçilip o bölgedeki noktalar görüntülenebilmektedir.

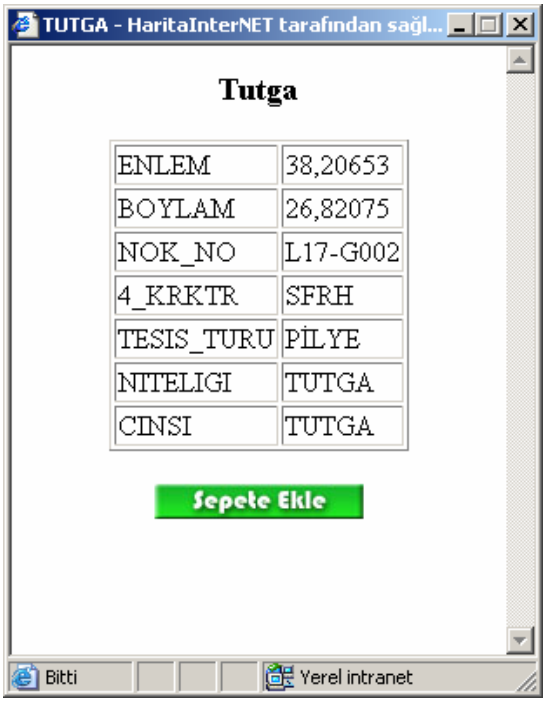


Şekil 3-8 TUTGA Sayfası (kullanıcı tanımlı)

Seçilen bölgedeki TUTGA noktaları görüntülenirken büyütme ve küçültme işlemleri yapılabilmektedir. İstenilen bölge görüntülendiğinde fare ile noktanın üzerine gelindiğinde ipucu olarak noktanın adı belirmekte (Şekil 3-9) ve noktaya tıklanarak nokta hakkındaki ayrıntılı bilgiye ulaşılabilmektedir (Şekil 3-10).

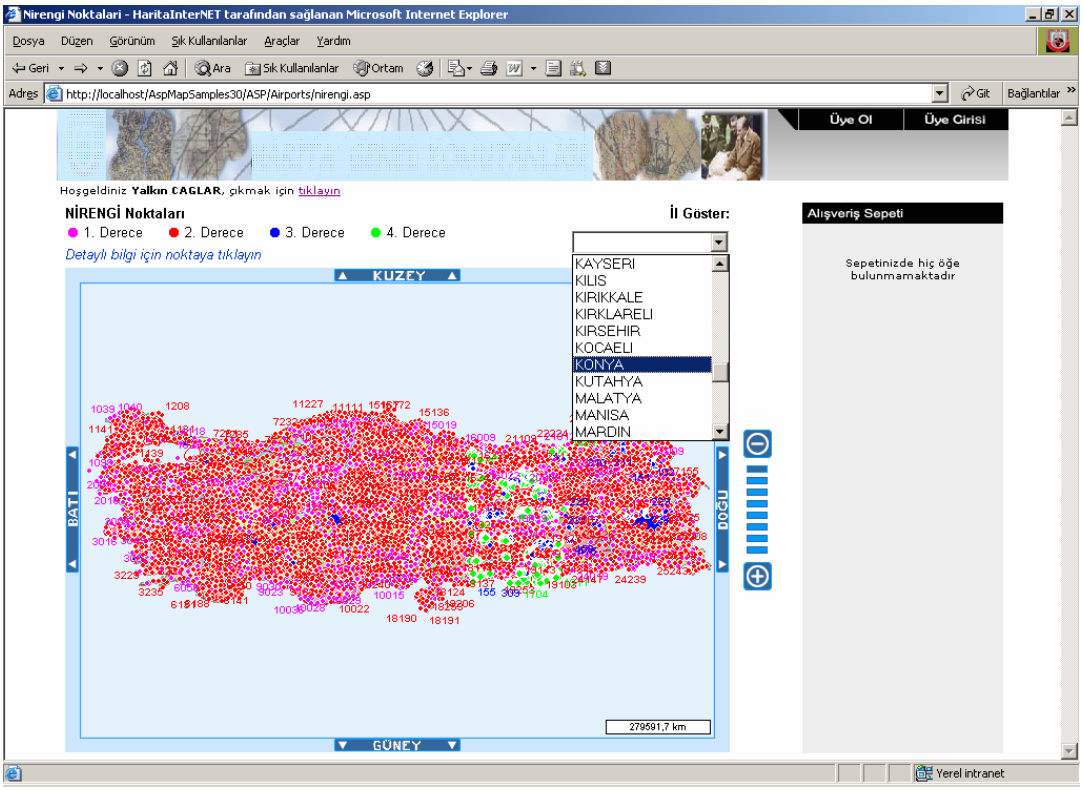


Şekil 3-9 TUTGA Nokta Seçimi

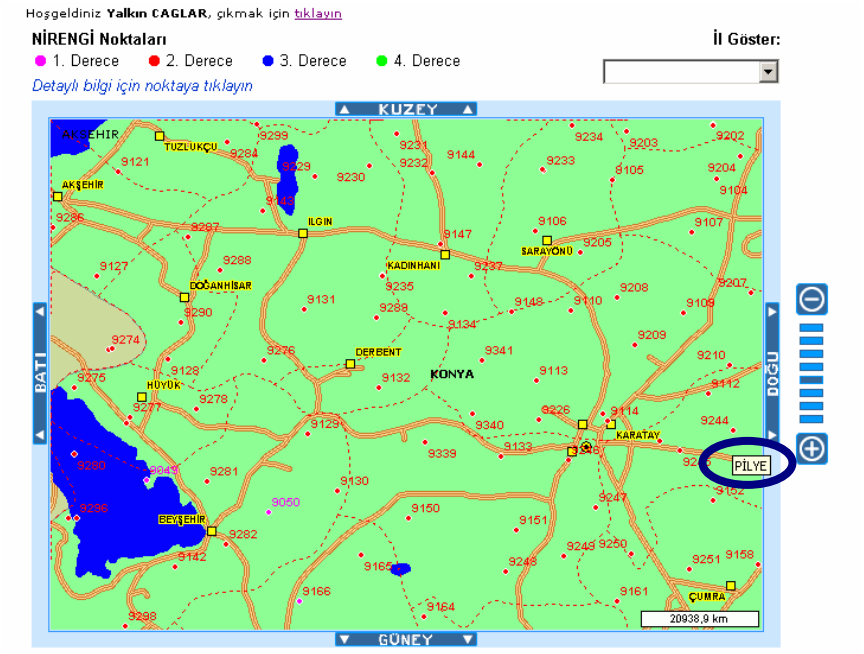


Şekil 3-10 TUTGA Nokta Bilgisi

Seçimi tamamlanan ürünler “Sepete Ekle” ile satın alınmak üzere seçilebilmektedir. Aynı işlemler nirengi ve nivelman noktaları için de yapılabilmektedir. Bu sayfalar aşağıda gösterilmiştir (***Şekil 3-11*** ~ ***Şekil 3-16***).



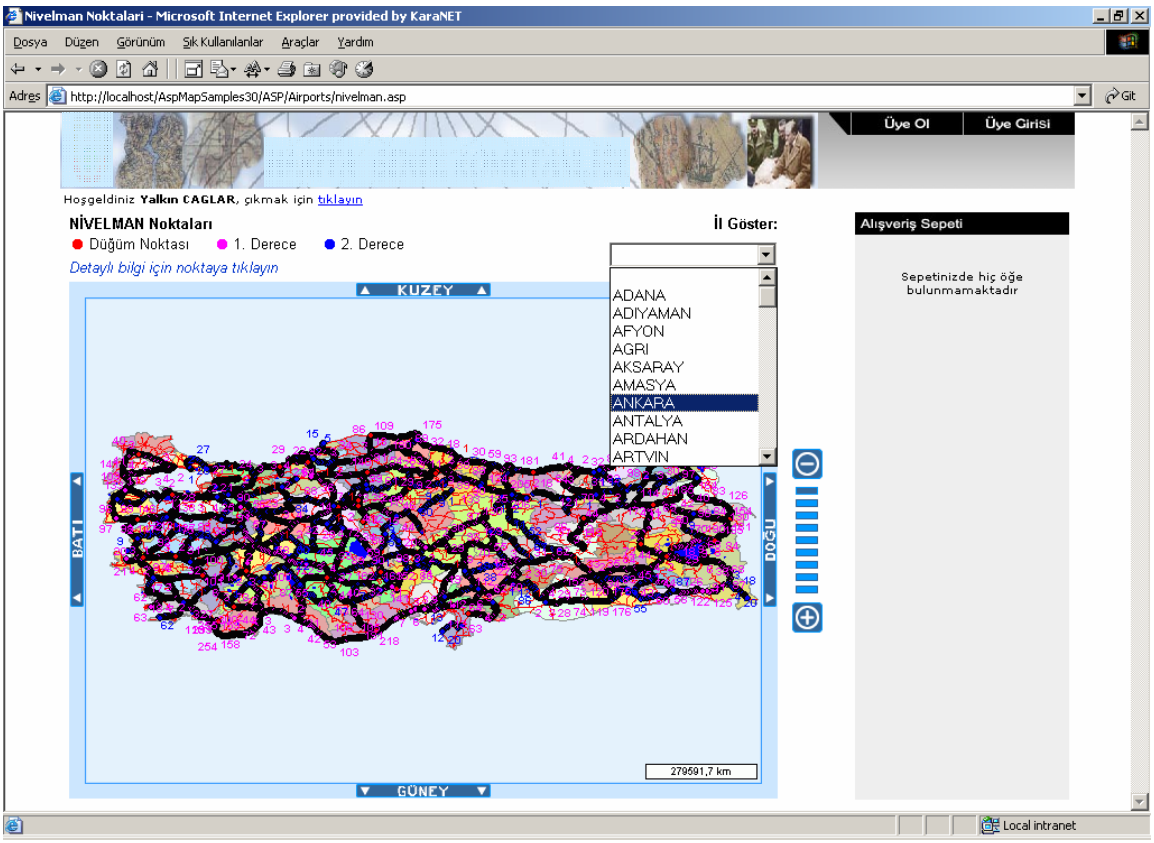
Şekil 3-11 Nirengi Sayfası (kullanıcı tanımlı)



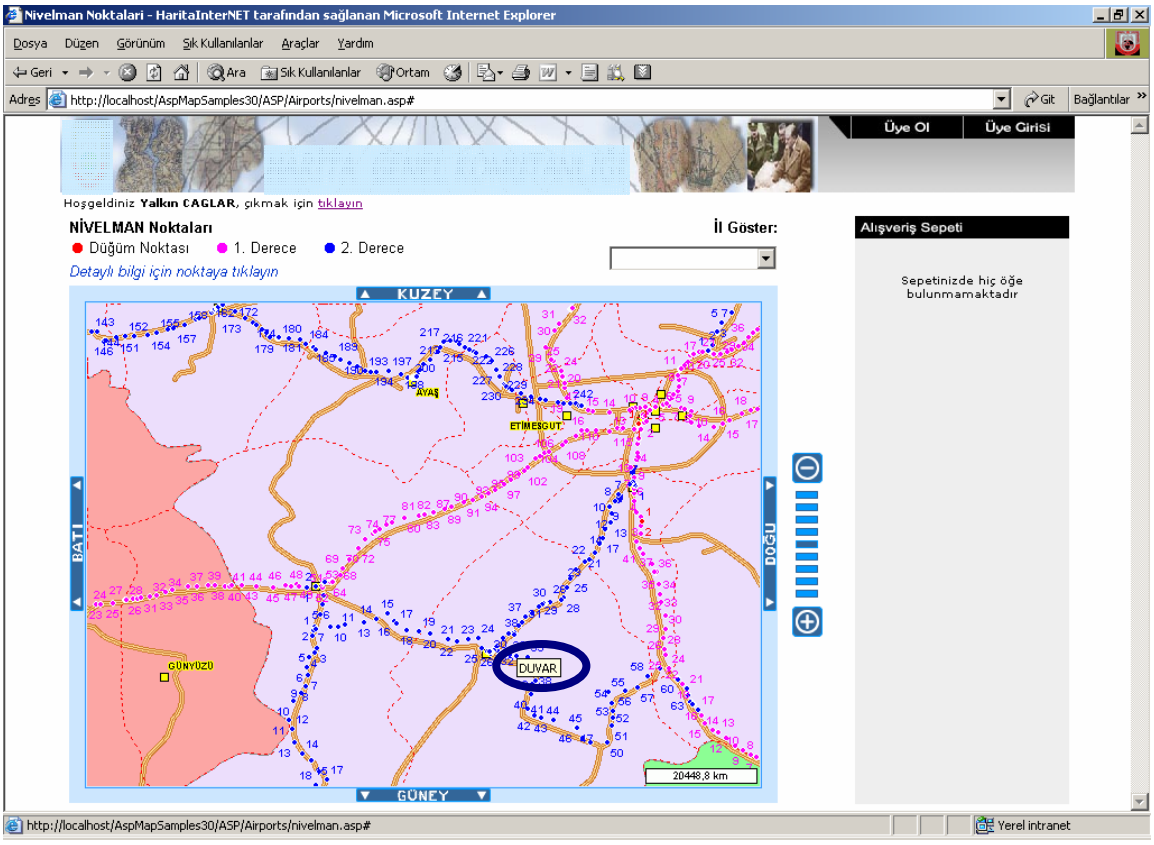
Şekil 3-12 Nirengi Nokta Seçimi



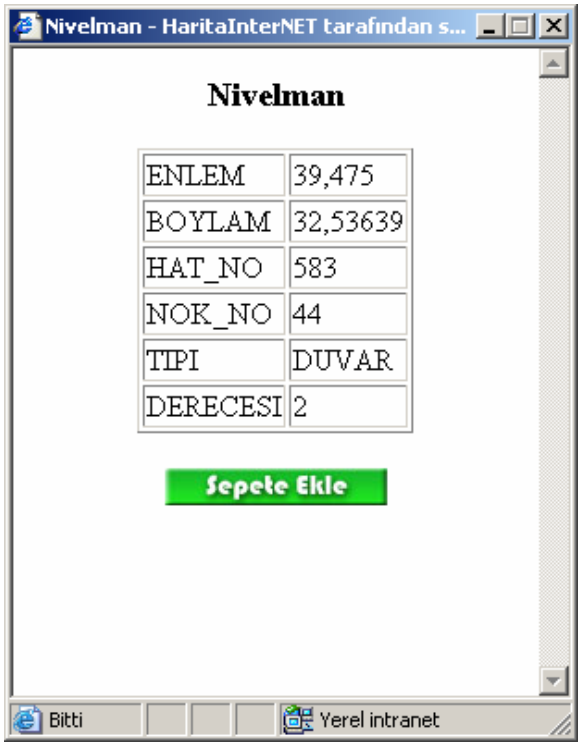
Şekil 3-13 Nirengi Nokta Bilgisi



Şekil 3-14 Nivelman Sayfası (kullanıcı tanımlı)



Şekil 3-15 Nivelman Nokta Seçimi



Şekil 3-16 Nivelman Nokta Bilgisi

4. SONUÇ

Bu çalışmada veri tabanı tasarım sürecindeki aşamalar Jeodezik Veri Tabanı için ele alınmış ve bir e-Devlet projesi kapsamında bu değerlerin satışını yapabilmek üzere web-tabanlı bir Coğrafî Bilgi Sistemi uygulaması hazırlanmıştır. Bu uygulama sonucunda aşağıdaki sonuçlar çıkarılabilir:

1. Çözümlemenin başında kapsamın çok geniş tutulması ve bunun için yeterli personel ve zamanın iyi planlanmaması, sistem çözümleme uygulamasının sonuçlandırılamama riskini arttırmaktadır.

2. Verilerin bir VTS içerisinde tutularak kullanıcılara sunulması sırasında şu zorluklarla karşılaşılmıştır:

    a. Sistem çözümlemesinin doğasında olan geri beslemeler (örneğin mantıksal veritabanı tasarımı aşamasında yeni gereksinimlerin ortaya çıkması) fazlalaştığında geriye dönmekten ilerleme sağlanamamakta bu da çözümcüler arasında bıkkınlık ve yılgınlığa neden olmaktadır. Bunu önlemek için baştan sistemin sınırlarının ve ara hedeflere ulaşmadaki zamanların iyi belirlenmesi gerekmektedir.

    b. Son kullanıcılarla çözümcüler çok farklı alanlardan geldiğinde bir iletişim kopukluğu ortaya çıkmakta, bu da mevcut sorunların teşhisinde yanlışlıklara neden olmaktadır. Bunu önlemek için bu iki grubun mümkün olduğunca birbirlerini anlayabilecek insanlardan seçilmesi doğu sonuçları ulaşılma oranını yükseltecektir.

    c. Uygulamanın ömrünün uzaması başta kullanılmasına karar verilmiş donanım ve yazılımların demode olmasına ve son gereksinimleri karşılamakta yetersiz kalmasına sebep olmaktadır. Bunu için daha önce de değinildiği gibi uzun zaman alacak mükemmel bir sistem çözümlemesine ulaşmak yerine kısa zamanda ulaşılabilecek ara hedefler belirlenerek ulaşılan her hedefin kullanıma sunulmasının daha uygun olduğu değerlendirilmektedir.

    d. İnternet ortamında veri paylaşımı devlet kurumlarını her zaman rahatsız etmekte bu da sonucun alınmasında sıkıntılarla

karşılaşılmasına neden olmaktadır. Bu yüzden karar vericileri ikna etmek bazen eldeki olanakların bir bölümünün kullanımından vazgeçmekle mümkün olmaktadır.

3. Tüm bunların yanı sıra gerçekleştirilen uygulama ile şu yararlar sağlanmaktadır:

    a. Bu sayfaların kullanıma açılmasıyla Türkiye'nin her köşesinden kullanıcılar en güncel ve doğru jeodezik değeri satın alabilecekler ve resmi/sivil işlem ve hesaplamalarında kullanabileceklerdir.

    b. Veriler arasındaki karmaşık ilişkilerin modellenmesi ve verilerin ortak bir alanda korunması ile veri güncellemesi kolaylaşmaktadır.

    c. Çeşitli bilgi kaynakları birleştirilerek karmaşık kararlar almak daha kolay ve hızlı olarak yapılabilmeye başlanmaktadır.

    d. Uygulamalar arası veri bağımsızlığı sağlandığından verilerin bozulması ve kaybolması riski ortadan kaldırılmaktadır.

    e. Veriler arası ilişkiler kurulup bunlardan görünümler oluşturulduğundan disk hacminden büyük tasarruf sağlanmaktadır.

    f. Yeni uygulama geliştirimi çok hızlı ve ucuza yapılabilir hale gelmektedir.

Sonuç olarak bilgi sisteme geçiş ile her sistem daha kararlı bir duruma gelir. İnternetin kullanımı ile de hayatı kolaylaştırmak hem devlet kurumlarının hem de özel / tüzel kişilerin hedefi olmalıdır. Toplum yararı için kurulmuş kurumlar bu görevlerini interneti kullanarak daha hızlı ve etkin olarak yerine getirmelidirler.

KAYNAKLAR

Abrial, J., 1974. Data Semantics, North – Holland.

Albano, A., Cardelli, L. ve Orsini, R., 1985 GALILEO: A Strongly – Typed Interactive Conceptual Language, Association for Computing Machinery Transaction on Database Systems (journal), **Vol.10: No.2**.

Batini, C., Lenzerini, M. ve Navathe S., 1987. A Comparative Analysis of Methodologies for Database Schema Integration, Association for Computing Machinery Computing Surveys, **Vol.18: No.4**.

Bruyn, W. ve ark., 1988. ESML: An Extended Systems Modeling Language Based on the Data Flow Diagram, Association for Computing Machinery Software Engineering Notes, **Vol.13: No.1**, pp. 58 – 67.

Casanova, M. ve Vidal, V., 1982. Toward a Sound View Integration Method, Proceedings of the Association for Computing Machinery Symposium on Principles of Database Systems.

Ceri, S., 1983. Methodology and Tools for Database Design, North – Holland.

Chen, P., 1976. The Entity – Relationship Model – Toward a Unified View of Data, Association for Computing Machinery Transaction on Database Systems (journal), **Vol.1: No.1**.

Codd, E, 1974. Extending the Database Relational Model to Capture More Meaning, Association for Computing Machinery Transaction on Database Systems (journal), **Vol.2: No.4**.

Date, C., 1990. An Introduction to Database Systems, Volume 1, Fifth Edition, Addison – Wesley.

Eick, C. ve Lockemann, P., 1985. Acquisition of Terminological Knowledge Using Database Design Techniques, SIGMOD.

Elmasri, R., Larson, J. ve Navathe, S., 1986. Schema Integration Algorithms for Federated Databases and Logical Database Design, Honeywell CSDD, Technical Report CSC-86-9: 8212.

Elmasri, R. ve Navathe, S., 1995. Fundamentals of Database Systems, Second Edition, Benjamin / Cummings Publishing Co. Inc.

Elmasri, R., ve Wiederhold, G., 1981, GORDAS: A Formal, High – Level Query Language for the Entity – Relationship Model, ER Conference 1981, Washington DC.

ER Conference, 1992. Proceedings of the Eleventh International Conference on Entity – Relationship Approach, Pernul, G. and Tjoa, A. (editors), Karlsruhe, Germany.

Fry, J. ve Sibley, E., 1976. Evolution of Database Management Systems, Association for Computing Machinery Computing Surveys, **Vol.8: No.1**.

Gane, C. ve Sarson, T., 1977. Structured System Analysis: Tools and Techniques, Improved System Technologies Inc.

Korth, H. ve Silberschartz, A., 1991. Database System Concepts, Second Edition, McGraw – Hill Inc.

McFadden F. ve Hoffer, J., 1988. Database Management, Second Edition, Benjamin / Cummings Publishing Co. Inc.

Mylopolous, J., Bernstein, P. ve Wong, H., 1980. A Language Facility for Designing Database Intensive Applications, Association for Computing Machinery Transaction on Database Systems (journal), **Vol.5: No.2**.

Navathe, S., Elmasri, R. ve Larson, J., 1986. Integrating User Views in Database Design, Computer Magazine (journal) of Institute of Electrical and Electronics Engineers (IEEE), **Vol.19: No.1**.

Navathe, S. ve Kerschberg, L., 1986. Role of Data Dictionaries in Database Design, Information and Management, **Vol.10: No.1**.

Pressman, Roger S., 1992. Software Engineering: A Practitioners Approach, McGraw – Hill Inc.

Stevens, W. P., Myers, G. J. ve Constantine L. L., 1974. Structured Design, IBM Systems Journal, **Vol.13: No.2**, pp. 115 – 139.

Teorey, T., 1990. Database Modeling and Design: The Entity – Relationship Approach, Morgan Kaufmann.

Teorey, T. ve Fry, J., 1982. Design of Database Structures, Prentice – Hall.

Verheijen, G. ve VanBekkum, J., 1982. NIAM: An Information Analysis Method, North – Holland.

Yao, S., 1985. Principles of Database Design, Volume 1: Logical Organizations, Prentice – Hall.

EKLER

EK-A KARNELER

EK-A1 DİSTOMAT (KENAR) ÖLÇÜ KARNESİ

DISTOMAT UZUNLUK ÖLÇÜSÜ VE İNDİRGEME HESABI

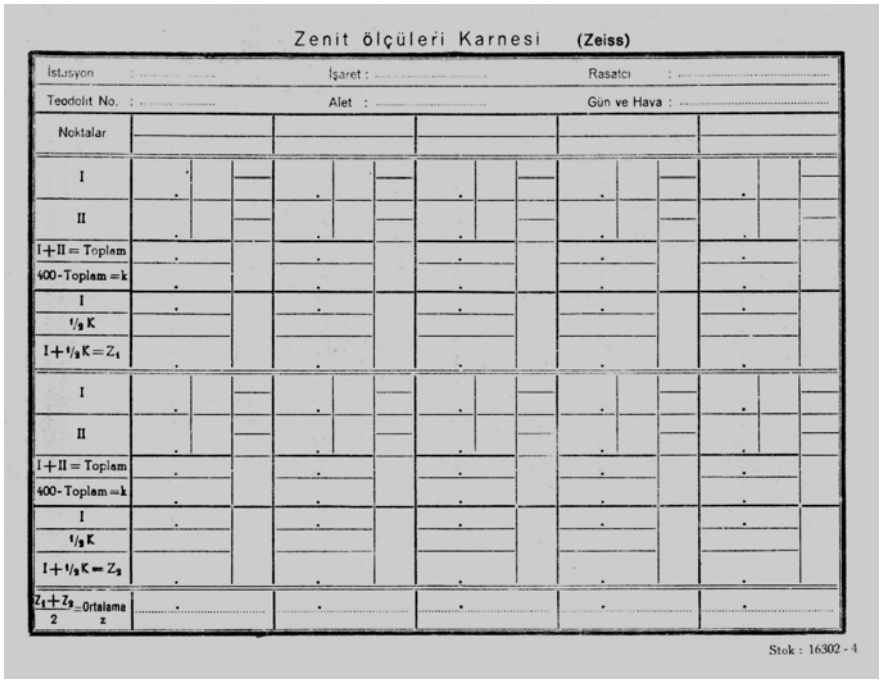
GÖZLEMCİ :	TARİH :	I			
		II			
İSTASYON :	HEDEF :	I+II=T			
		400-T=K			
ALET YÜK :	İŞARET YÜK:	I			
		1/2K'			
		I+1/2K=Z_1			
ÖLÇÜLER		I			
t = °C	S_1 = , m	II			
p = mb	S_2 = , m	I+II=T			
		400-T=K			
ppm =	S_3 = , m	I			
		1/2K			
	S_{ort} = , m	I+1/2K=Z_2			
		Z = $(Z_1+Z_2)/2$			

FORMÜLLER	DEĞERLER	AÇIKLAMA
S_{ort}		Meteorolojik düzeltme almış uzunluk (Birimi = Metre)
$\Delta H = H_a - H_b$ ★		Alet ve hedef yüksekliği farkı (Birimi = Metre)
$H_m = \frac{H_a+H_b}{2}$		Alet ve hedef yükseklikleri ortalaması (Birimi = Metre)
R		Yeryuvarlağı eğrilik Yarıçapı (Birimi = Metre)
$K_1 = (\frac{\Delta H^2}{2S_{ort}} + \frac{\Delta H^4}{8S^3_{ort}})$		Yatay düzleme indirgeme düzeltmesi
$K_2 = -\frac{H_m}{R} S_{ort}$		Deniz Yüzeyine (veya elipsoide) indirgeme düzeltmesi
$K_3 = \frac{S^3_{ort}}{24\,R^2}$		Deniz Yüzeyindeki (veya elipsoiddeki) Kirişten yay uzunluğuna indirgeme düzeltmesi
[K]		$[K] = K_1 + K_2 + K_3$
$S_o = S_{ort} + K$		Deniz (veya elipsoid) Yüzeyindeki yay uzunluğu
$ds = \frac{1}{6R^2}(y_1^2 + y_1y_2 + y_2^2)\, S_o$		Gauss - Kruger Yüzeyine indirgeme düzeltmesi
$S = S_o + ds$		Gauss - Kruger Yüzeyindeki uzunluk

★ Denizden olan Yükseklik alınırsa, deniz seviyesine ;
elipsoidden olan yükseklik alınırsa, elipsoid yüzeyine indirgeme yapılmış olur.

NOT : Bu indirgemeler geçicidir.

EK-A2 ZENİT (DÜŞEY AÇI) ÖLÇÜ KARNESİ



Zenit ölçüleri Karnesi (Zeiss)

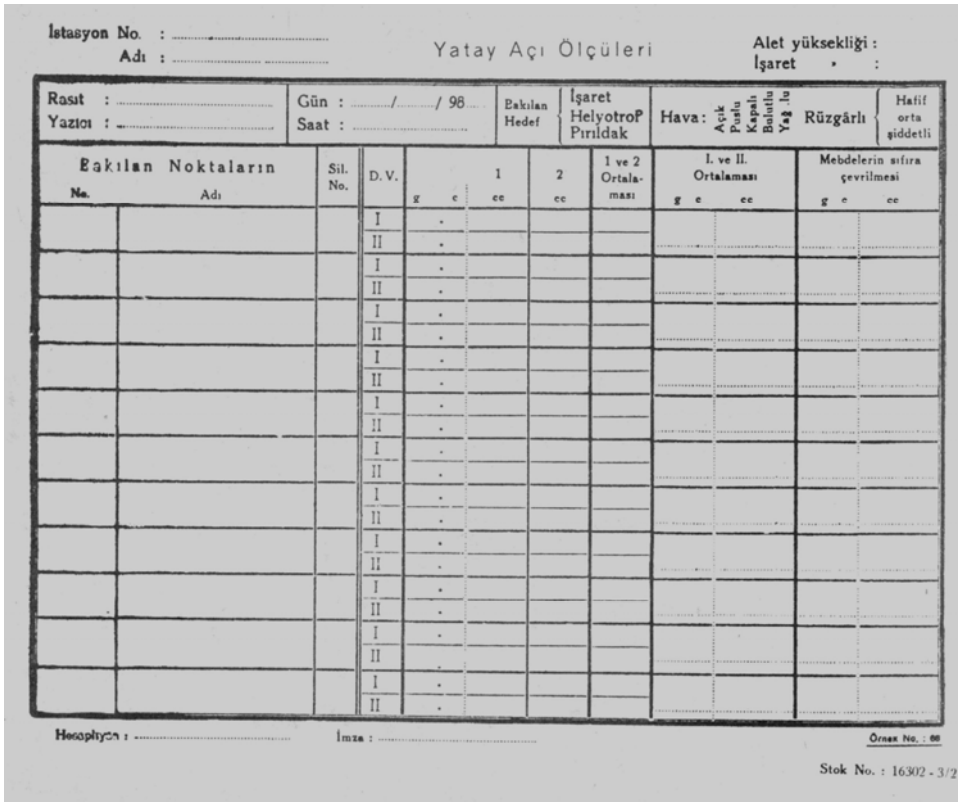
İstasyon : ..... İşaret : ..... Rasatcı : .....

Teodolit No. : ..... Alet : ..... Gün ve Hava : .....

Noktalar					
I	.	.	.	.	.
II	.	.	.	.	.
I+II = Toplam	.	.	.	.	.
400 - Toplam = k	.	.	.	.	.
I	.	.	.	.	.
1/3 K					
I + 1/3 K = Z_1	.	.	.	.	.
I	.	.	.	.	.
II	.	.	.	.	.
I+II = Toplam	.	.	.	.	.
400 - Toplam = k	.	.	.	.	.
I	.	.	.	.	.
1/3 K					
I + 1/3 K = Z_2	.	.	.	.	.
$\frac{Z_1+Z_2}{2}$ = Ortalama z	.	.	.	.	.

Stok : 16302 - 4

EK-A3 YATAY AÇI ÖLÇÜ KARNESİ – 1



İstasyon No. : .....
Adı : .....

Yatay Açı Ölçüleri

Alet yüksekliği : .....
İşaret : .....

Rasat : .....
Yazıcı : .....

Gün : ... / ... / 98 ...
Saat : .....

Bakılan Hedef | İşaret Helyotrop Pırıldak

Hava: Açık Pusu Kapalı Bulutlu Yağ..lu

Rüzgârlı | Hafif orta şiddetli

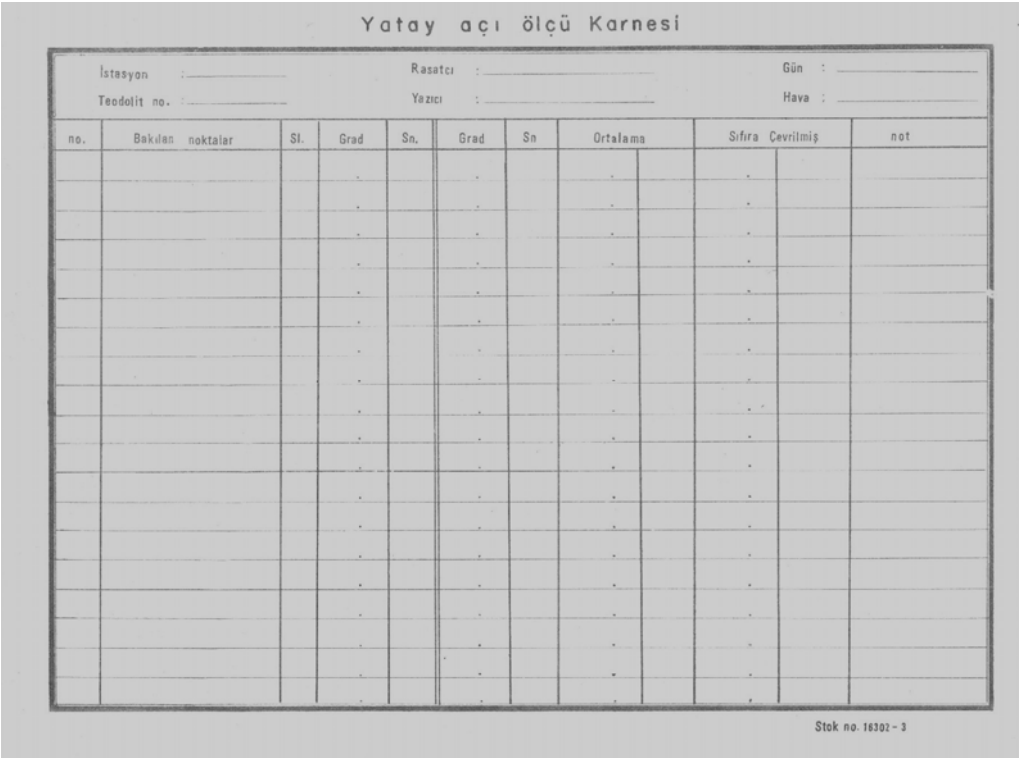
Bakılan Noktaların No.	Adı	Sil. No.	D. V.	1 g	1 c	1 cc	2 cc	1 ve 2 Ortalaması	I. ve II. Ortalaması g	I. ve II. Ortalaması c	I. ve II. Ortalaması cc	Mebdelerin sıfıra çevrilmesi g	Mebdelerin sıfıra çevrilmesi c	Mebdelerin sıfıra çevrilmesi cc
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										
			I	.										
			II	.										

Hesaplayan : ..... İmza : .....

Örnek No. : 66

Stok No. : 16302 - 3/2

EK-A4 YATAY AÇI ÖLÇÜ KARNESİ – 2



Yatay açı ölçü Karnesi

İstasyon : Rasatcı : Gün :

Teodolit no. : Yazıcı : Hava :

no.	Bakılan noktalar	Sl.	Grad	Sn.	Grad	Sn	Ortalama		Sıfıra Çevrilmiş		not

Stok no. 16302 - 3

EK-A5 DEĞER CETVELİ – 1

DEĞER CEDVELİ

Paftası : BANDIRMA-618-C2

Dilim orta meridyeni $= \lambda_0 =$ 27°

Dilim genişliği 6°

Küçültme faktörü 0.9996

NOKTALARIN No.	NOKTALARIN Adı	NOKTALARIN KOORDİNELERİ Sağa (Metre)	NOKTALARIN KOORDİNELERİ Yukarı (Metre)	Kot Yer (Metre)	NOT
329	Boz tepe	352 805.65	4 248 638.11	25.48	
342	Akbisar	385 256.17	4 361 241.27	32.15	

Yazıldığı gün : 24.09.2001

Yazanın adı : Ayten Gökçe

İmzası : AG

Kontrol eden : Ahmet Yetiş

İmzası : AG

Stok No. : 15310 - A

EK-A6 DEĞER CETVELİ – 2

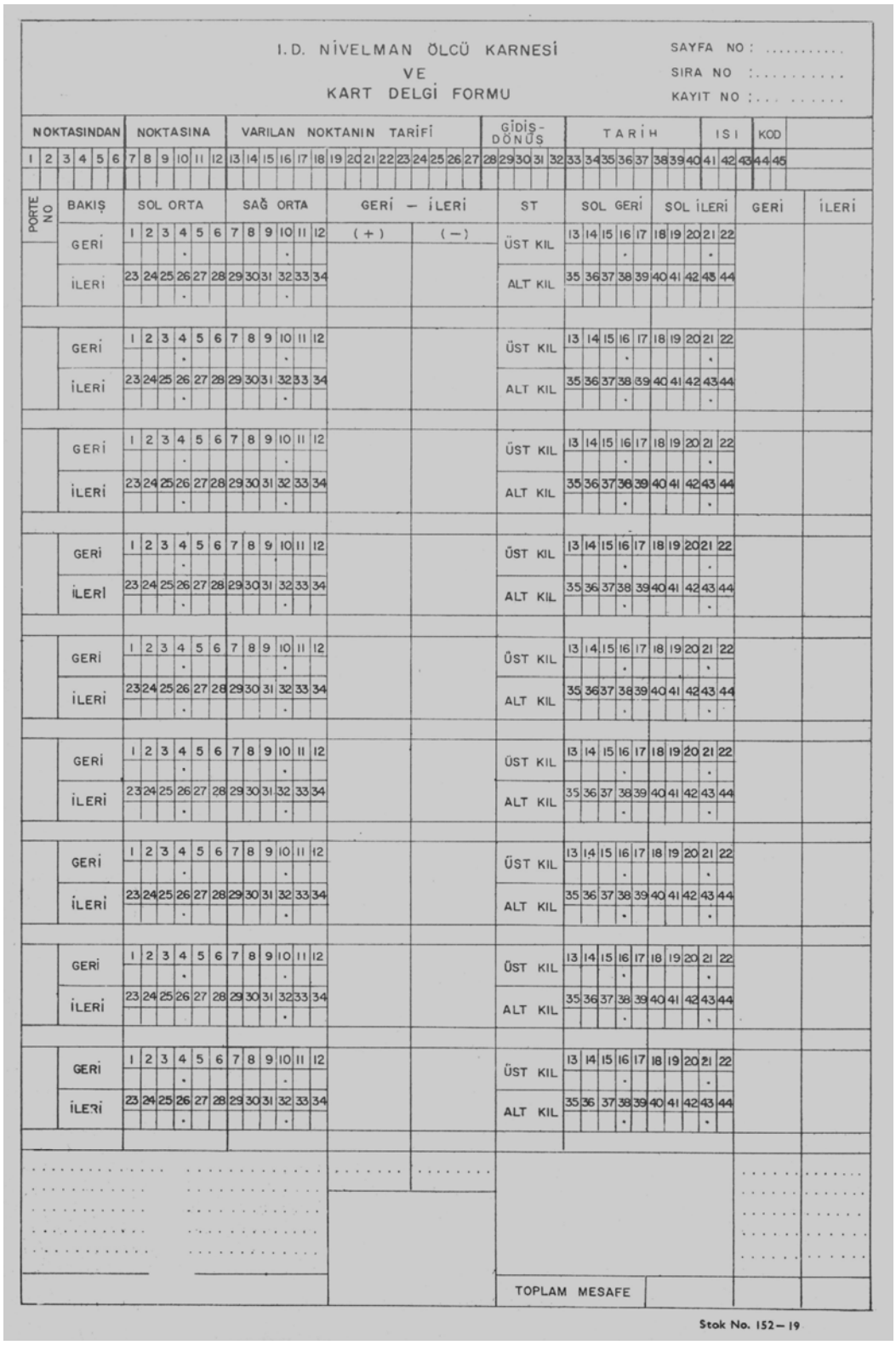
DEĞER CETVELİ

PAFTASI : ELBİSTAN K38-a1
DİLİM ORTA MERİDYENİ : 39°
DİLİM GENİŞLİĞİ : 3°
KÜÇÜLTME FAKTÖRÜ : 1.0000

NOKTA NO	NOKTA ADI	SAĞA DEĞER (m.)	YUKARI DEĞER (m.)	YER KOTU (m.)	NOT
101	Höyük	544 436.99	4 578 767.00	985.97	
106	Alaca	532 654.85	4 473 666.77	623.41	
ELBİSTAN K38-a2					
126	Aras tepe	556 634.77	4 499 523.53	875.91	

YAZILDIĞI TARİH : 24.09.2001 **KONTROL EDEN :Ahmet TAN**
HAZIRLAYAN :Ayten Gökçe **İMZASI :**
İMZASI :

EK-A7 NİVELMAN (YÜKSEKLİK) ÖLÇÜSÜ KARNESİ



I.D. NİVELMAN ÖLCÜ KARNESİ
VE
KART DELGİ FORMU

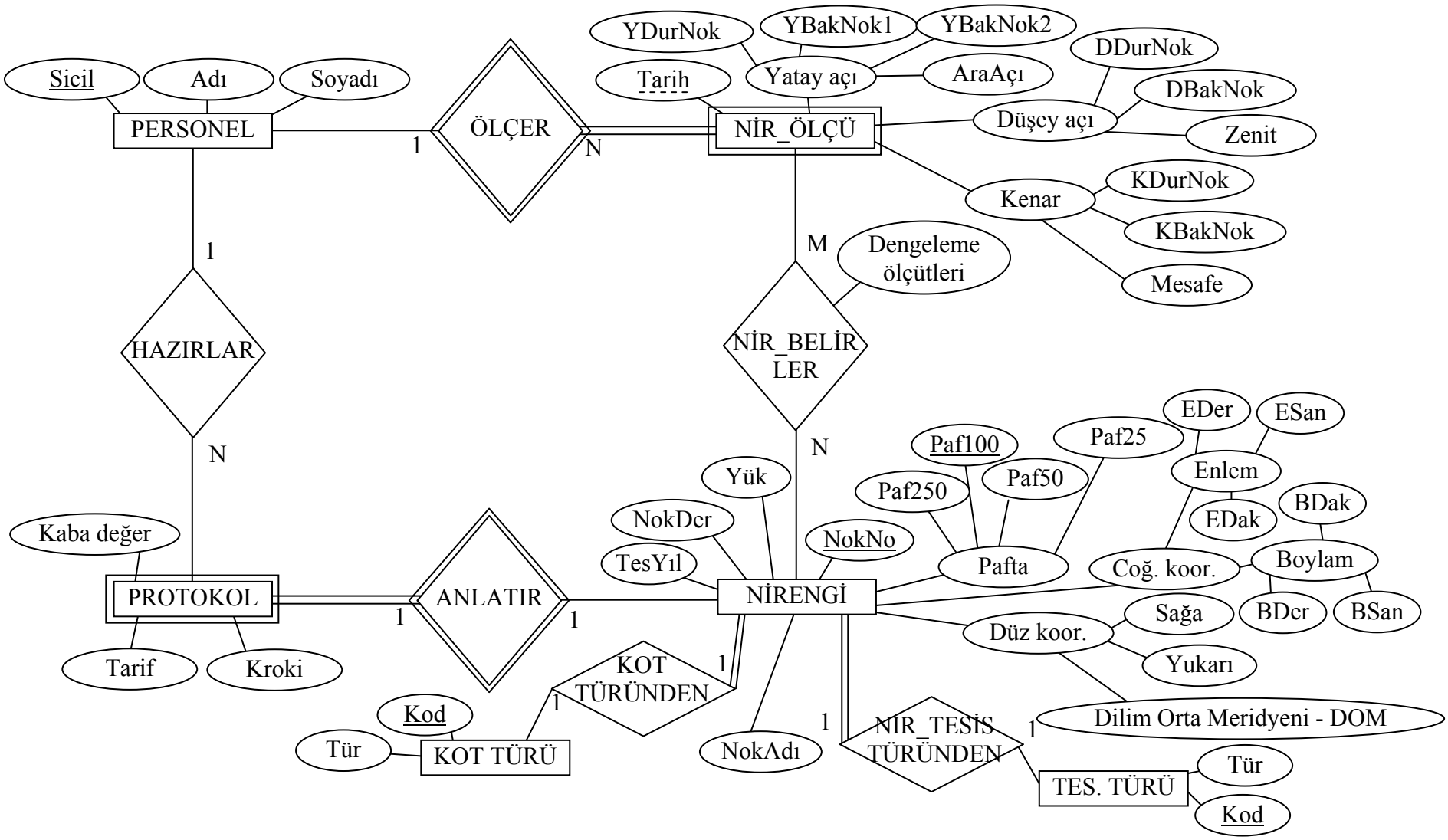
SAYFA NO : .....
SIRA NO : .....
KAYIT NO : .....

NOKTASINDAN	NOKTASINA	VARILAN NOKTANIN TARİFİ	GİDİŞ-DÖNÜŞ	TARİH	ISI	KOD
1 2 3 4 5 6	7 8 9 10 11 12	13 14 15 16 17 18 19 20 21 22 23 24 25 26 27	28 29 30 31 32	33 34 35 36 37 38 39 40 41	42 43	44 45

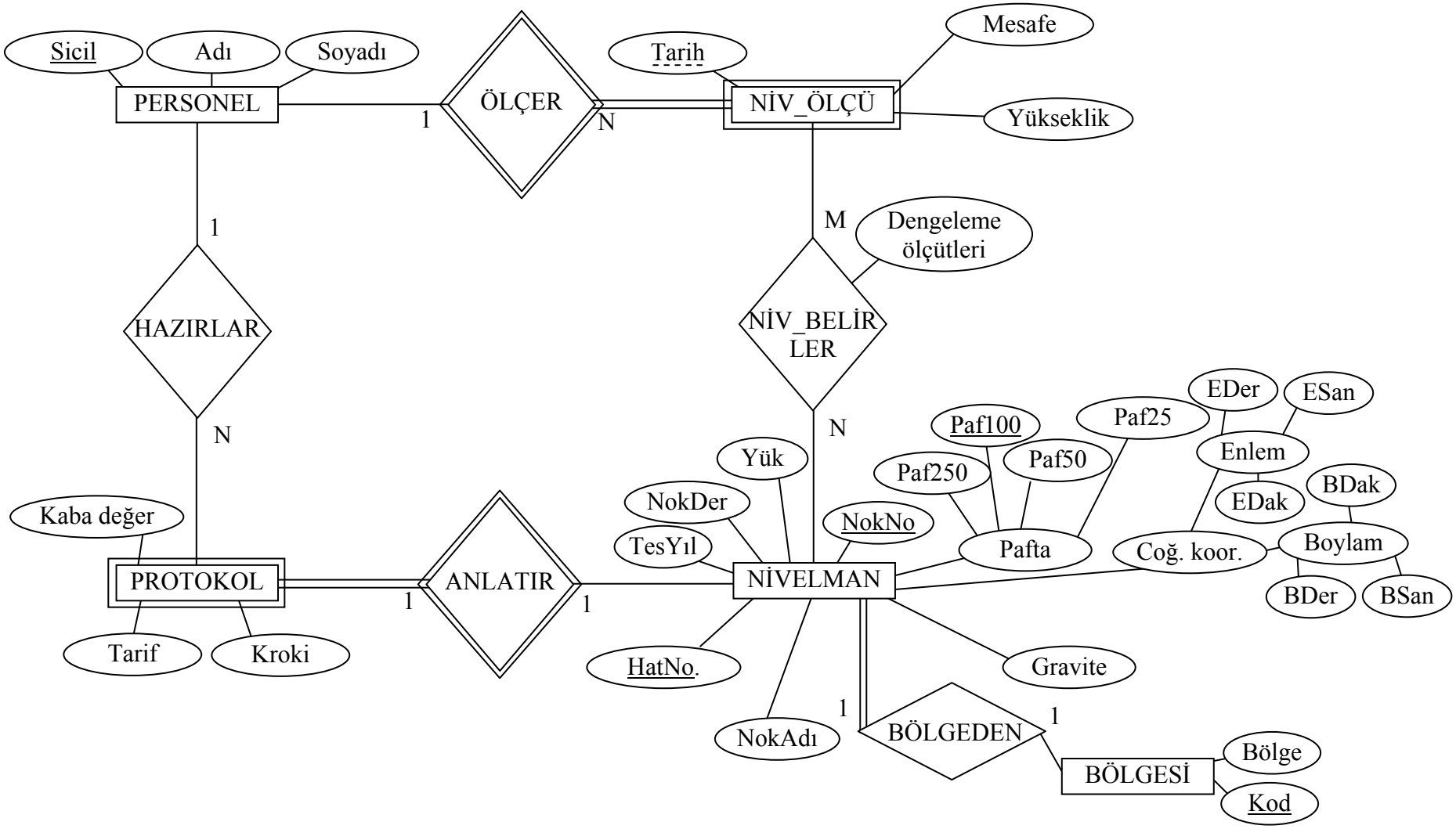
PORTE NO	BAKIŞ	SOL ORTA	SAĞ ORTA	GERİ – İLERİ		ST	SOL GERİ	SOL İLERİ	GERİ	İLERİ
				(+)	(–)					
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
	GERİ	1 2 3 4 5 6	7 8 9 10 11 12			ÜST KIL	13 14 15 16 17	18 19 20 21 22		
	İLERİ	23 24 25 26 27 28	29 30 31 32 33 34			ALT KIL	35 36 37 38 39	40 41 42 43 44		
						TOPLAM MESAFE				

Stok No. 152–19

EK – B1: Varlık – İlişki Diyagramı (Nirengi)



EK – B2: Varlık – İlişki Diyagramı (Nivelman)



EK – B3: Varlık – İlişki Diyagramı (TUTGA)

