**VERİTABANI SİSTEMLERİ**

**ÖDEV 1**

**Ad: Emin Beyazıt POYRAZ**

**Ögr num: 23301071048**

**Veritabanının Tarihçesi**

Bir konuyla ilgili verilere hızlı ve kolay biçimde erişebilme isteği, verilerin düzenli biçimde

saklanması ihtiyacını doğurmuş, dolayısıyla veritabanı olgusunun bu adla olmasa

bile geçmişten günümüze kadar bir biçimde hayatımızda yer almasına neden olmuştur.

Geçmişle günümüz arasındaki fark kullanılan araç, gereç, teknik ve yaklaşımlarda ortaya

çıkmaktadır. Bilgisayar kullanımının özel işletmeler için uygun maliyetli bir seçenek olmaya başladığı

dönem olan 1960’larda veritabanlarının bilgisayarlar yardımıyla oluşturulması dönemi

de başlamıştır. İzleyen kesimde 1960’lı yıllardan itibaren veritabanı uygulamalarının

tarihsel gelişimine ilişkin kısa bilgilere yer verilmiştir

**2000’ler:** Internet endüstrisi 2000’lerin başında bir düşüş yaşamasına rağmen veritabanı

uygulamaları büyümeye devam etmiştir. Web tabanlı işletmelerin artması ve kullanımlarının

yaygınlaşması ile veritabanı yönetim sistemlerinin önemi giderek artmaya

devam etmekte ve yeni uygulamaların geliştirilmesine neden olmaktadır. Günümüzde,

veritabanı yönetim sistemi yazılımlarında lider kabul edilen üç işletme Microsoft, IBM ve

Oracle olarak sıralanmaktadır.

**VERİTABANINA İLİŞKİN TEMEL KAVRAMLAR**

• Veritabanı herhangi bir kurumda birden fazla uygulamada ortak olarak kullanılabilen

verilerden oluşur.

• Veritabanında sürekli niteliği olan veriler bulunur. Buna göre, girdi ya da çıktı verisi

olan ya da kurum için sürekli bir anlam ifade etmeyen geçici veriler veritabanında

yer almaz.

• Veritabanı, ortak kullanılan verilerin tekrarlanmasına izin vermeden çok amaçlı

kullanılmasına olanak verir.

• Veritabanında saklanan veriler durağan nitelikte değişmez veriler değildir. Ekleme,

silme ya da güncelleme işlemleri ile veritabanındaki veriler değiştirilebilir.

**GELENEKSEL DOSYA SİSTEMLERİ VE VERİTABANI YÖNETİM**

**SİSTEMLERİ**

Bilgisayar tabanlı bilgi sistemleri kapsamında kullanılan ve farklı uygulamalar tarafından paylaşılan

ortak verilerin düzenlenmesi, saklanması ve kullanılması amacına yönelik iki farklı yaklaşım vardır:

* Geleneksel dosya sistemi
* Veritabanı yönetim sistemi

Zaman içinde artan veri miktarı, bu verinin depolanması için gerekli olan kapasite,

veriye erişim ve işleme hızında yaşanan sıkıntılar geleneksel dosya sisteminin temel sınırlılıkları

olarak ortaya çıkmıştır. Dolayısıyla bu sıkıntılar modern veritabanı sistemlerinin

geliştirilmesini de gerekli hâle getirmiştir

**Gereksiz Veri Tekrarı ve Veri Tutarsızlığının Önlenmesi**

Önceki kesimde de belirtildiği gibi geleneksel dosya sistemlerinde her alt sistem ya da

uygulama için gerekli olan veriler ayrı dosyalar biçiminde düzenlenmektedir. Böylece verilerden

bir kısmı bilgisayarda aynı anda birden fazla dosya içinde yer alabilmekte dolayısıyla

verinin büyük ölçüde tekrar edilmesi sonucuyla karşılaşılmaktadır. Bu durum aynı

zamanda veri derleme işleminin de tekrarlı olmasına ve veri derleme için yapılan harcamaların

artmasına neden olabilmektedir. Diğer yandan veri tekrarı, verilerin depolanması

için kullanılan bellek kapasitelerinin dolayısıyla donanım harcamalarının artmasına da

yol açar. Veri tekrarı aynı zamanda tekrarlanan verilerin farklı dosyalardaki değerlerinin

de farklı olabilmesine ve dolayısıyla veri tutarsızlığı ile uygulamalarda sorunlar yaşanmasına

neden olabilir

**Veri Bütünlüğünün Sağlanması**

Veri bütünlüğü, veritabanında yer alan bir verinin farklı uygulamalarda kullanımı sözkonusu

olduğunda veri üzerinde yapılacak bir değişimin verinin kullanıldığı diğer uygulamalara

da yansıtılması anlamına gelir. Bu, özellikle verinin silinmesi durumunda önem

kazanmaktadır. Buna göre bir veri silindiğinde verinin ilişkili olduğu tüm uygulamalardan

o verinin silinmesi gereklidir. Geleneksel dosya sistemlerinde veri bütünlüğünün sağlanması

güç iken veritabanı yönetim sistemlerinde, sisteme girilen kısıtlamalar ve kurallar

yardımıyla veri bütünlüğü rahatlıkla sağlanabilmektedir.

**VERİTABANI KULLANICILARI**

Veritabanı ile herhangi bir şekilde etkileşimde olan kişi ya da kişiler veritabanı kullanıcısı

olup aşağıdaki gibi sınıflandırılabilirler:

• Veritabanı Sorumluları

• Veritabanı Yöneticisi

• Veritabanı Tasarımcısı

• Son Kullanıcılar

• Standart Kullanıcılar

• Sıradan ya da Parametrik Kullanıcılar

• Gelişmiş Kullanıcılar

• Bağımsız Kullanıcılar

• Sistem Analistleri ve Uygulama Programcıları vb

**VERİTABANI YÖNETİM SİSTEMLERİNİN MİMARİSİ**

Veritabanı yaklaşımı **veri soyutlama** özelliği ile farklıkullanıcıların verileri tercih

ettikleri ayrıntı seviyesinde algılamalarına, gereksiz ayrıntıları

görmemelerine olanak sağlar.

Temel **veri modeli** işlemleri;veritabanı üzerinde ekleme, silme,değiştirme, veriyi geri çağırma

gibi genel işlemleri içerir

**Veri Modellerinin Sınıflandırılması**

**Fiziksel veri modelleri,** kayıt biçimi, kayıt sırası ve erişim yolu bilgilerini göstermek suretiyle

verilerin bilgisayarda dosya olarak nasıl saklandığını belirler.

Yüksek düzeyli ya da kavramsal veri modelleri (conceptual data models), kullanıcıların

*veri algılama biçimiyle* ilişkili kavramları kapsar.

Düşük düzeyli ya da **fiziksel veri modelleri** (physical data models), *verinin bilgisayar*

*ortamında nasıl depolanacağına* ilişkin ayrıntıları tanımlayan kavramları kapsar.

Fiziksel veri modelleri kavramları genellikle son kullanıcılar için değil, bilgisayar uzmanları

için geliştirilir.

**Şemalar, Örnekler ve Veritabanının Durumu**

Veritabanının herhangi bir veri modeliyle tanımlanması *veritabanı şeması* (database schema)

olarak adlandırılır. Veri modellerinin çoğunda şemaların bir diyagram

olarak gösterilebilmesi amacıyla kullanılan belirli kuralları vardır. Şema görünümleri *şema*

*diyagramı* (schema diagram) olarak adlandırılır.

Şema diyagramları şemaların yalnızca bazı yönlerini gösterir. Veritabanında her şema yapısı kendi oluşum

kümesine sahiptir. Örneğin “öğrenci” veritabanındaki bireysel öğrenci kayıtları oluşumlara

örnektir. Veritabanı durumu, belirli bir şema ile ilişkilendirilerek yapılandırılır. Her zaman

yeni bir kayıt ekleme, silme ya da bir kayıttaki verinin değerini değiştirmek mümkündür

(böylece veritabanının durumu bir durumdan diğerine değiştirilebilir).

VTYS şema yapılarını ve kısıtlarını (meta veri) veritabanı içinde tanımlar ve VTYS

yazılımı ne zaman ihtiyaç duysa bu şemaya başvurur.

**Üç Şema Mimarisi**

Veritabanı yaklaşımının önemli karakteristikleri aşağıdaki biçimde sıralanabilir:

1. Veritabanı tanımlarını (şema) depolamak için kendi kendine tanım yapabilmeyi

sağlayan bir katalog kullanır.

2. Veri ve program izolasyonunu sağlar (program – veri bağımsızlığı; program – işlem

bağımsızlığı)

3. Çoklu kullanıcıyı destekler

Üç şema mimarisi, yukarıda sıralanan bu özelliklerin gerçekleştirilmesine ve anlaşılmasına

yardımcı olacaktır.

Üç şema mimarisinin amacı kullanıcı uygulamalarını fiziksel veritabanından ayırt etmektir.

Bu mimaride şemalar izleyen üç düzeyde tanımlanır:

**1. İçsel (fiziksel) düzey** (internal level), veritabanının *fiziksel depolama yapısını* tanımlayan

içsel şemayı içerir. İçsel şema, veriyi depolama ayrıntılarının tamamını

ve veritabanına erişim yollarını tanımlayan fiziksel veri modelini kullanır.

**2. Kavramsal düzey** (conceptual level), kullanıcı topluluğu için tüm veritabanının

yapısını tanımlayan kavramsal şemayı içerir. Kavramsal şema fiziksel depolama

yapısının ayrıntılarını gizler ve veritabanında yer alan verilerin tipine, veriler arası

ilişkilere, kullanıcı işlemlerine ve kısıtlara ilişkin tanımlara yoğunlaşır. Veritabanı sistemlerinde,

uygulama veri modelleri kavramsal şemayı tanımlamak için kullanılır.

Bu kavramsal şema, yüksek düzeyli kavramsal veri modelleri kullanılarak tasarlanır.

**3. Dışsal (görünüm) düzey** (external level), bir dizi dışsal şema ya da kullanıcı görünümü

içerir. Her dışsal şema bir grup kullanıcının ilgilendiği bazı veritabanı

bölümlerini tanımlar. Böylece veritabanının diğer kısmı bu kullanıcı grubundan

gizlenir. Her dışsal şema, bir yüksek düzey veri modelinde tasarlanan dışsal şema

tabanlı uygulama veri modeli kullanır.

**Veri Bağımsızlığı**

**Veri bağımsızlığı**, herhangi bir düzeydeki şema değiştiğinde bir üst düzeydeki şemanın değişmeden kalmasını bununla birlikte iki düzey arasındaki eşleştirmelerin değişmesini sağlar.

Bu eşleştirme sonucunda bir üst düzeydeki şemaya başvuran uygulama programları değişiklik ihtiyacı göstermez.

**Veritabanı Yönetim Sistemlerinde Kullanılan Diller**

Veritabanı tanımları **Veri Tanımlama Dili** kullanılarakoluşturulur.

Daha sonra butanımlar **Veri Tanımlama Dili Derleyicisi** kullanılarakçözümlenir ve gerektiğindekullanıcıya iletilmek üzereVTYS tarafından uygun yapılara

dönüştürülerek saklanır.

**VERİTABANI TÜRLERİ**

Daha önce de söz edildiği üzere her veritabanı yönetim sistemi bir veri modeli kullanır.

Veritabanında yer alacak veriler ve veriler arasında kurulacak ilişkiler mantıksal olarak

ilgili veri modeline göre yapılandırılır ve veritabanları da buna göre sınıflandırılır.

Geçmişten günümüze kadar geliştirilmiş olan çok sayıda veri modeli, kullandıkları teknikler

açısından dört temel başlıkta incelenir. Bu dört veri modelinden hangisini kullandığına

bağlı olarak veritabanları da aşağıda verilen dört başlık altında sınıflandırılabilir:

1. Hiyerarşik veritabanı (Hierarchical database)

2. Ağ veritabanı (Network database)

3. İlişkisel veritabanı (Relational database)

4. Nesneye yönelik veritabanı (Object oriented database).

**Hiyerarşik Veritabanı**

Hiyerarşik veritabanı, en eski veri modeli olan hiyerarşik veri modelini temel alıp 1960 ve

1970’li yıllarda yaygın olarak kullanılmıştır. Bu tür veritabanlarında kullanılan veri modelinde

kayıtlar, ilişkileri temsil eden ve ağaç yapısına benzeyen kök ve dallar biçiminde hiyerarşik

bir yapıda oluşturulur. Bu yapı, başlangıç noktası ağacın kökü, bağlantılı kılınacak

noktalar dallar ve ana dallara bağlı alt dallar olarak düşünülebilen bir yapı biçimindedir.

Buna göre veriler arasındaki ilişkilerde hiyerarşinin üst bölümünde olan dallar alt bölümde

birden çok dal ile bağlantılı olabilirken alt bölümünde olan dallar üstte kalan dallara

yalnızca tek bir noktadan bağlantılı olabilirler.

**Ağ Veritabanı**

Ağ veritabanı, 1970’li yıllar ile 1980’li yılların ilk yarısında kullanılan ve ağ veri modelini

temel alan veritabanı türüdür. Hiyerarşik veri modelindeki ebeveyn-çocuk ilişkisinin

yetersizliği ağ veri modeliyle giderilmeye çalışılmıştır. Buna göre her bağlantı noktası düğüm

olarak ifade edilirse hiyerarşik yapıdan farklı olarak ağ veri modelinde, her düğümün

birden fazla ebeveyn ve birden fazla çocuk düğümü ile bağlantısı olabilir.

**İlişkisel Veritabanı**

Bu tür veritabanı ilişkisel veri modelini temel almış ve ilk olarak 1970 yılında ortaya atılmıştır.

1970’li yılların sonunda kullanılmaya başlanmış ve 1985 yılından sonra kullanımı

yaygınlaşmıştır. Bu yapıda ilk iki veri modelinden farklı olarak birden çok ilişki biçimi

kullanılabilir. Günümüzde kullanılan veritabanı yönetim sistemlerinin hemen hemen

hepsinde tercih edilen model ilişkisel veri modelidir. Bu nedenle modele ilişkin ayrıntılı

bilgiler Ünite 3’te ele alınmıştır.

**Nesneye Yönelik Veritabanı**

Günümüzde kullanılan ve gelecekte de kullanılacak pek çok uygulamada yalnızca harf,

rakam ya da çeşitli karakterler kullanılarak yapılandırılmış verileri değil aynı zamanda

multimedya (çeşitli çizim, fotoğraf, görüntü, ses ya da video gibi nesneleri) de içeren veritabanı

yönetim sistemlerine ihtiyaç duyulmaktadır. Bu nedenle bu eksikliği gidermek

amacıyla nesneye yönelik veri modelleri geliştirilmiştir. Sözkonusu bu modelleri kullanan

veritabanları da nesneye yönelik veritabanı olarak adlandırılmaktadır

Önek akış diyagramı :

Bire bir - bire çok çoka çok birer tane şema oluşturuldum

Bire bir ilişki

1 1

**aittir**

TC. NO

|  |
| --- |
| Birinci anahtar |

|  |
| --- |
| Adı |
| soyadı |
| TC. |

|  |
| --- |
| adı |
| soyadı |
| TC.no |

|  |
| --- |
| Yabancı anahtar |

|  |
| --- |
| adres |

Bire çok ilişki

Anne

**İlişki**

Çocukları

1 n

|  |
| --- |
| Adı |
| Soyadı |
| Adres |

|  |
| --- |
| Birinci anahtar |

|  |
| --- |
| Adı\_ |
| Soydı |
| Adres\_no |

|  |
| --- |
| Yabncı anahtar |

Çoka çok ilişki

Öğrenciler

Dersler

**Kayıt olur**

m

n m

|  |
| --- |
| Birinci anahtar |

|  |
| --- |
| Ögrenci |
| Üniv\_no |
| Ad |
| Adres |

|  |
| --- |
| Öğrenci no |
| Dersler no |
| Adı |
| Adres.no |

|  |
| --- |
| Yabancı anahtar |