Bölüm

9 Veritabanı Tasarımı

Öğrenme Hedefleri

Bu bölümü tamamladıktan sonra şunları yapabileceksiniz:

- **9-1** Başarılı bir bilgi sisteminin temeli olarak veritabanı tasarımının rolünü açıklayabileceksiniz.
- **9-2** Sistem Geliştirme Yaşam Döngüsü'nün (SDLC) beş aşamasını açıklayabileceksiniz.
- **9-3** Veritabanı Yaşam Döngüsü (DBLC) çerçevesindeki altı aşamayı kullanarak veritabanları tasarlayabileceksiniz.
- **9-4** SDLC ve DBLC çerçeveleri içerisinde değerlendirme ve revizyon yapabileceksiniz.
- **9-5** Veritabanı tasarımında yukarıdan aşağıya (top-down) ve aşağıdan yukarıya (bottom-up) yaklaşımlar arasındaki farkı ayırt edebileceksiniz.
- **9-6** Merkezi ve merkezi olmayan kavramsal veritabanı tasarımı arasındaki farkı ayırt edebileceksiniz.

ÖNİZLEME

Veritabanları, bilgi sistemi adı verilen daha büyük bir resmin parçasıdır. Bu gerçeği göz ardı eden veritabanı tasarımlarının başarılı olma olasılığı düşüktür. Veritabanı tasarımcıları, veritabanının kendi başına bir amaçtan ziyade, bir amaca ulaşmak için kritik bir araç olduğunu kabul etmelidir. Yöneticiler, veritabanının kendi yönetim ihtiyaçlarına hizmet etmesini ister, ancak pek çok veritabanı, yöneticileri veritabanı gereksinimlerine uymak için rutinlerini değiştirmeye zorluyor gibi görünmektedir.

Bilgi sistemleri kendiliğinden oluşmaz; dikkatlice aşamalandırılmış bir geliştirme sürecinin ürünüdürler. Sistem analizi, bir bilgi sistemine olan ihtiyacı belirlemek ve sınırlarını çizmek için kullanılır. Sistem analizi içinde, gerçek bilgi sistemi, sistem geliştirme olarak bilinen bir süreç aracılığıyla oluşturulur.

Bilgi sistemlerinin oluşturulması ve evrimi, bilgi sisteminin oluşturulması, bakımı, geliştirilmesi ve değiştirilmesinin sürekli bir süreci olan Sistem Geliştirme Yaşam Döngüsü (SDLC) adı verilen yinelemeli bir modeli izler. Veritabanları için de benzer bir döngü geçerlidir; veritabanı oluşturulur, bakımı yapılır, geliştirilir ve sonunda değiştirilir. Veritabanı Yaşam Döngüsü (DBLC), bu bölümde dikkatlice izlenir ve daha büyük Sistem Geliştirme Yaşam Döngüsü bağlamında gösterilir.

Bölümün sonunda, veritabanı tasarımına yönelik bazı klasik yaklaşımlar olan yukarıdan aşağıya (top-down) ve aşağıdan yukarıya (bottom-up) ile merkezi (centralized) ve merkezi olmayan (decentralized) yaklaşımlarla tanışacaksınız.

Veri Dosyaları *cengage.com'da* mevcuttur.

Not

Tamamen kavramsal olduğu için, bu bölüm herhangi bir veri dosyasına atıfta bulunmamaktadır.

9-1 Bilgi Sistemi

Temel olarak, bir veritabanı dikkatlice tasarlanmış ve oluşturulmuş bir gerçekler deposudur. Veritabanı, veri toplama, depolama, dönüştürme ve geri çağırma (erişim) imkanı sağlayan, bilgi sistemi (IS) olarak bilinen daha büyük bir bütünün parçasıdır. Bilgi sistemi ayrıca verilerin bilgiye dönüştürülmesine yardımcı olur ve hem verilerin hem de bilginin yönetilmesini sağlar. Bu nedenle, eksiksiz bir bilgi sistemi insanlardan, donanımdan, yazılımdan, veritabanı/veritabanlarından, uygulama programlarından ve prosedürlerden oluşur. Sistem analizi, bir bilgi sistemine olan ihtiyacı ve kapsamını belirleyen süreçtir. Bir bilgi sistemi oluşturma süreci, sistem geliştirme olarak bilinir.

Günümüz bilgi sistemlerinin en önemli özelliklerinden biri, küresel iş dünyası çağında bilginin stratejik değeridir. Bu nedenle, bilgi sistemleri her zaman stratejik iş misyonu ve hedefleriyle uyumlu olmalıdır; bağımsız ve soyutlanmış bilgi sistemleri görüşü artık geçerli değildir.

Bilgi Sistemi (IS)
Veri toplama, depolama ve erişim sağlayan; verilerin bilgiye dönüştürülmesini kolaylaştıran; hem verileri hem de bilgiyi yöneten bir sistem. Bir bilgi sistemi donanım, DBMS ve diğer yazılımlar, veritabanı (ları), insanlar ve prosedürlerden oluşur.

Sistem Analizi
Bir bilgi sistemine olan ihtiyacı ve kapsamını belirleyen süreç.

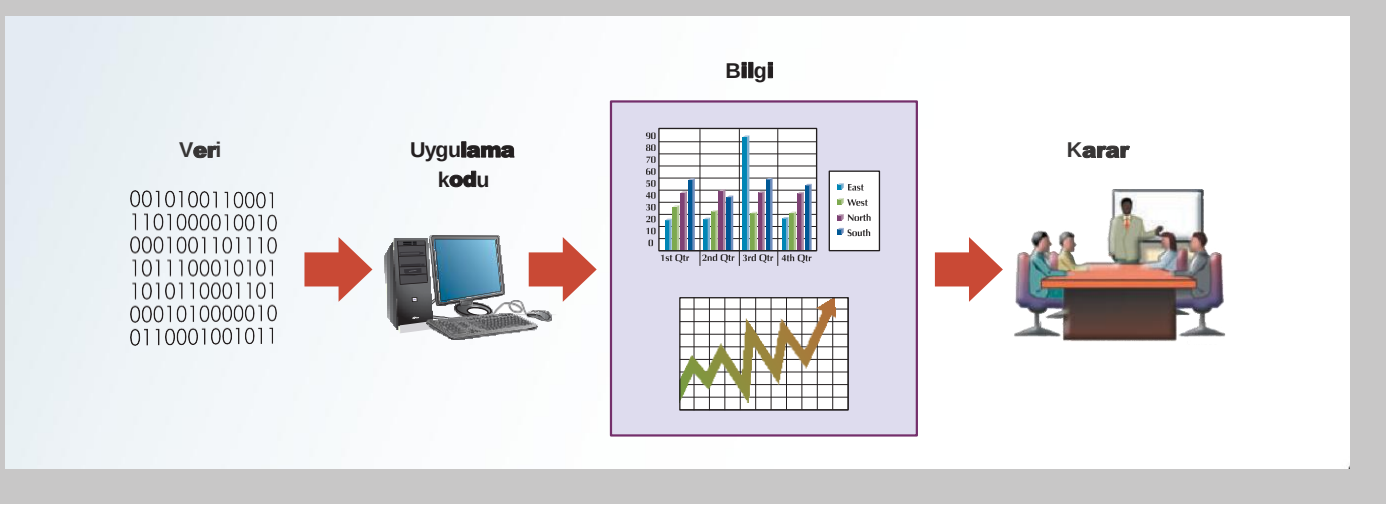
Sistem Geliştirme
Bir bilgi sistemi oluş-turma süreci.

Not

Bu bölüm, genellikle ayrı bir ders veya kitapta ele alınan sistem analizi ve geliştirmenin tüm yönlerini kapsamayı amaçlamamaktadır. Ancak bu bölüm, veritabanının kritik bir bileşen olduğu bilgi sisteminden etkilenen veritabanı tasarımı, uygulaması ve yönetimi ile ilgili konuları daha iyi anlamanıza yardımcı olacaktır.

Sistem geliştirme çerçevesinde, uygulamalar verileri karar vermenin temelini oluşturan bilgiye dönüştürür. Uygulamalar genellikle bilgiden içgörü üretmek için tasarlanmış resmi raporlar, tablolar ve grafik gösterimler oluşturur. Şekil 9.1, her uygulamanın iki bölümden oluştuğunu göstermektedir: veriler ve verilerin bilgiye dönüştürüldüğü kod (program talimatları). Veriler ve kod, gerçek dünyadaki iş işlevlerini ve faaliyetlerini temsil etmek için birlikte çalışır. Herhangi bir anda, fiziksel olarak depolanan veriler işletmenin bir anlık görüntüsünü temsil eder, ancak kod ile temsil edilen iş faaliyetlerinin anlaşılması olmadan resim tamamlanmış olmaz.

Şekil 9.1 Karar Verme için Bilgi Üretme



Bir bilgi sisteminin performansı üç faktöre bağlıdır:

- **Veritabanı tasarımı ve uygulaması**
- **Uygulama tasarımı ve uygulaması**
- **Yönetsel prosedürler**

Bu kitap, üçlünün veritabanı tasarımı ve uygulama bölümünü vurgulamaktadır - ki bu tartışmasız en önemli üç bölümden biridir. Ancak, diğer iki bölümün ele alınmaması, muhtemelen kötü işleyen bir bilgi sistemi ile sonuçlanacaktır. Sağlam bir bilgi sistemi oluşturmak zor iştir; sistem analizi ve geliştirme, tüm faaliyetlerin birbiriyle arayüz oluşturmasını, birbirini tamamlamasını ve zamanında tamamlanmasını sağlamak için kapsamlı planlama gerektirir.

Geniş anlamda, veritabanı geliştirme terimi, veritabanı tasarımı ve uygulama sürecini tanımlar. Veritabanı tasarımındaki temel amaç, eksiksiz, normalleştirilmiş, gereksiz olmayan (mümkün olan en büyük ölçüde) ve tamamen entegre kavramsal, mantıksal ve fiziksel veritabanı modelleri oluşturmaktır. Uygulama aşaması, veritabanı depolama yapısının oluşturulmasını, verilerin veritabanına yüklenmesini ve veri yönetiminin sağlanmasını içerir. Zaman içinde esnek ve ölçeklenebilir bir veritabanı tasarlamak ve uygulamak için dikkatli olunmalıdır. Çoğu tasarım tipik olarak mevcut sorunları çözmeye odaklansa da, gelecekteki değişikliklere (performans, boyut veya raporlama gereksinimleri gibi) uyum sağlayacak kadar esnek bir tasarım oluşturmak önemlidir.

Veritabanı Geliştirme
Veritabanı tasarımı ve uygulama süreci.

Bu bölümde ele alınan prosedürleri geniş çapta uygulanabilir hale getirmek için, bölüm tüm bilgi sistemlerinde ortak olan unsurlara odaklanmaktadır. Bu bölümde açıklanan süreç ve prosedürlerin çoğu, uygulanan veritabanının boyutuna, türüne veya karmaşıklığına bağlı değildir. Ancak, bir mahalle ayakkabı mağazası gibi küçük bir veritabanı tasarlamak için kullanılacak prosedürler, büyük bir şirket veya hatta böyle bir şirketin bir bölümü için bir veritabanı tasarlamak için gerekli prosedürlere tam olarak ölçeklenmez. Bir benzetme kullanmak gerekirse, küçük bir ev inşa etmek için bir plan gerekir, tıpkı Golden Gate Köprüsü'nü inşa etmenin gerektirdiği gibi, ancak köprü çok daha karmaşık planlama, analiz ve tasarım gerektirmiştir.

Sonraki bölümler, genel Sistem Geliştirme Yaşam Döngüsü'nü ve ilgili Veritabanı Yaşam Döngüsü'nü ele alacaktır. Bu süreç ve prosedürlere aşina olduktan sonra, yukarıdan aşağıya (top-down) ve aşağıdan yukarıya (bottom-up) ile merkezi (centralized) ve merkezi olmayan (decentralized) tasarım gibi veritabanı tasarımına yönelik çeşitli yaklaşımları öğreneceksiniz.

Not

Sistem Geliştirme Yaşam Döngüsü, bilgi sistemlerini geliştirmek ve sürdürmek için gereken faaliyetleri izleyebileceğiniz ve anlayabileceğiniz genel bir çerçevedir. Bu çerçeve içinde, SDLC'de belirtilen çeşitli görevleri tamamlamanın birkaç yolu vardır. Bu kitap, ER modellemesine ve ilişkisel veritabanı tasarımına ve uygulamasına odaklanmaktadır ve bu odak bu bölümde de korunmaktadır. Ancak, çeşitli alternatif metodolojiler de mevcuttur, örneğin:

• Birleşik Modelleme Dili (UML), bilgi sistemlerinin geliştirilmesiyle ilişkili görevleri desteklemek için nesne yönelimli araçlar sağlar. UML, www.cengage.com adresindeki Ek H, Birleşik Modelleme Dili (UML)'de ele alınmaktadır.

• Hızlı Uygulama Geliştirme (RAD)1, uygulama sistemlerini geliştirmek için prototipleri ve esnek yönetimi kullanan yinelemeli bir yazılım geliştirme metodolojisidir.

(devamı)

[1]*Bkz. Hızlı Uygulama Geliştirme, James Martin, Prentice-Hall, Macmillan College Bölümü, 1991.*

• Çevik Yazılım Geliştirme, işi daha küçük alt projelere bölerek daha kısa sürelerde ve daha iyi uyumla değerli çıktılar elde etmek için yazılım uygulamaları geliştirmeye yönelik bir çerçevedir. Bu yöntem, müşteri memnuniyetini artırmak amacıyla tüm kullanıcılar arasında yakın iletişimi ve sürekli değerlendirmeyi vurgular. Çevik metodolojilere örnekler:

• DevOps, bir organizasyondaki geliştirme ve operasyon grupları arasında iletişimi ve işbirliğini vurgulayan işbirlikçi bir yazılım geliştirme yaklaşımıdır.[3]

• SCRUM, öncelikli çıktılara yönelik sürekli iyileştirmeye odaklanan yinelemeli ve artımlı bir metodolojidir.[4]

Geliştirme metodolojileri değişebilse de, içinde kullanıldıkları temel çerçeve değişmez.

9-2 Sistem Geliştirme Yaşam Döngüsü (SDLC)

Sistem Geliştirme Yaşam Döngüsü (SDLC)
Bir bilgi sisteminin geçmişini izleyen döngü. SDLC, veritabanı tasarımı ve uygulama geliştirmenin haritalandırılıp değerlendirilebileceği büyük resmi sağlar.

Sistem Geliştirme Yaşam Döngüsü (SDLC), **bir bilgi sisteminin geçmişini izler. Sistem tasarımcısı için belki de daha önemlisi, SDLC, veritabanı tasarımı ve uygulama geliştirmenin haritalandırılıp değerlendirilebileceği büyük resmi sağlar.**

Şekil 9.2'de gösterildiği gibi, geleneksel SDLC beş aşamaya ayrılır: planlama, analiz, detaylı sistem tasarımı, uygulama ve bakım. SDLC, sıralı bir süreçten ziyade yinelemeli bir süreçtir. Örneğin, fizibilite çalışmasının detayları ilk değerlendirmenin iyileştirilmesine yardımcı olabilir ve SDLC'nin kullanıcı gereksinimleri bölümünde ortaya çıkarılan detaylar fizibilite çalışmasının iyileştirilmesine yardımcı olabilir.

Veritabanı Yaşam Döngüsü, SDLC'ye uyduğu ve ona benzediği için, SDLC'nin kısa bir açıklaması yerinde olacaktır.

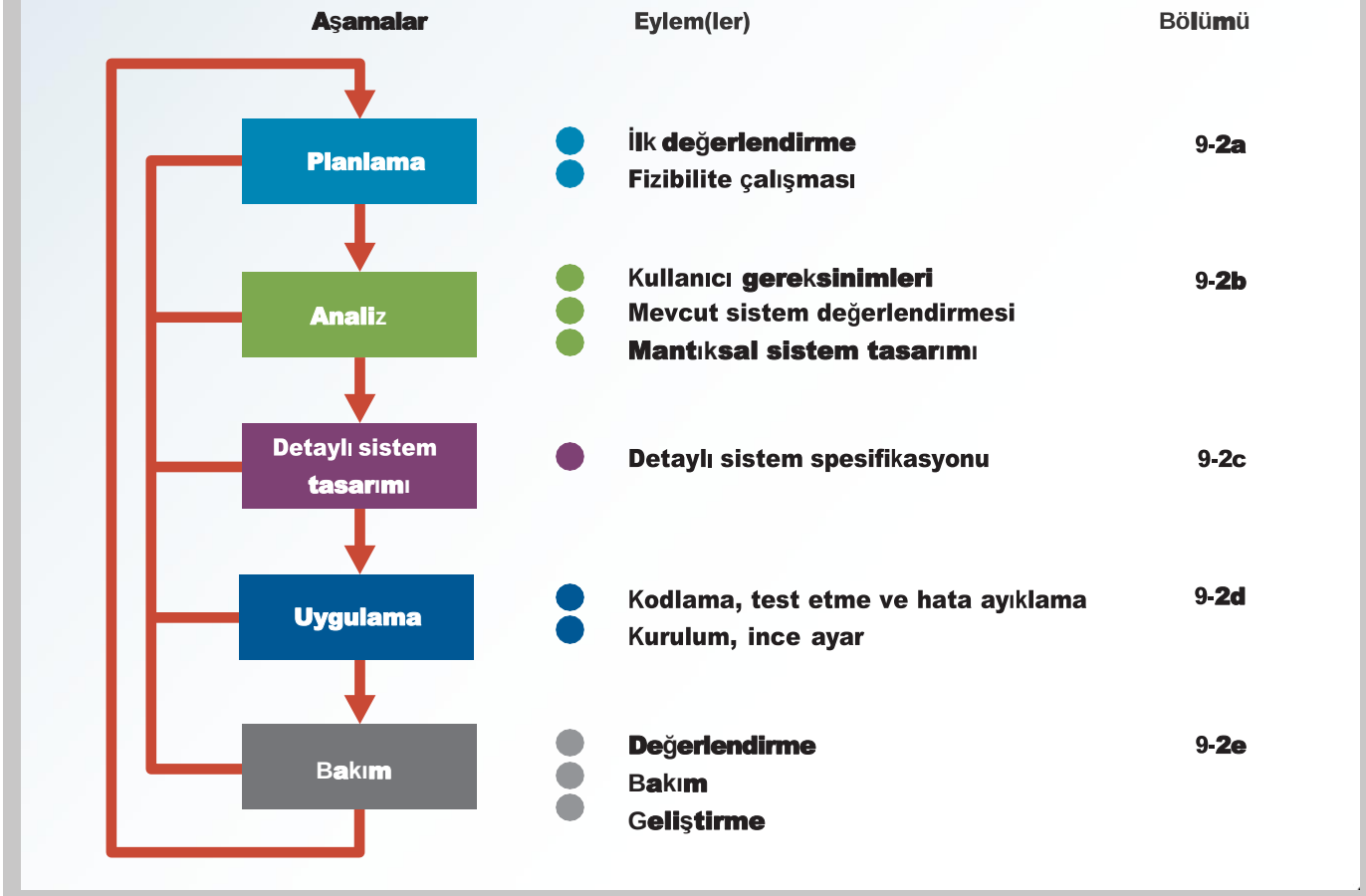
9-2a Planlama

SDLC planlama aşaması, şirketin ve hedeflerinin genel bir özetini sunar. SDLC'nin bu keşif aşamasında bilgi akışı ve kapsam gereksinimlerinin ilk değerlendirmesi yapılmalıdır. Böyle bir değerlendirme, bazı önemli soruları yanıtlamalıdır:

- ***Mevcut sistem devam ettirilmeli mi? Eğer bilgi üreteci işini iyi yapıyorsa, onu değiştirmenin veya yerine yenisini koymanın bir anlamı yoktur. Eski bir deyişle, "Bozuk değilse, tamir etme."***
- ***Mevcut sistem değiştirilmeli mi? Eğer ilk değerlendirme, bilginin kapsamı ve akışında eksiklikler gösteriyorsa, küçük (veya hatta büyük) değişiklikler gerekebilir. Değişiklikler düşünülürken, ilk değerlendirmeye katılanlar istekler ve ihtiyaçlar arasındaki farkı unutmamalıdır.***
- ***Mevcut sistem değiştirilmeli mi? İlk değerlendirme, mevcut sistemin kusurlarının düzeltilemeyecek kadar kötü olduğunu gösterebilir. Yeni bir sistem oluşturmak için gereken çaba göz önüne alındığında, istekler ve ihtiyaçlar arasındaki ayrım, bu durumda sistemi değiştirirken olduğundan belki de daha önemlidir.***

[2]Çevik Yazılım Geliştirme hakkında daha fazla bilgi için, www.agilealliance.org adresine gidin.
[3]DevOps'a ilişkin iyi bir genel bakış https://aws.amazon.com/devops/what-is-devops/ adresinde bulunabilir
[4]SCRUM çerçevesinin bir açıklaması için www.scrum.org/resources/what-is-scrum adresini ziyaret edin.

Şekil 9.2 Sistem Geliştirme Yaşam Döngüsü (SDLC)



SDLC'nin ilk değerlendirmesine katılanlar, alternatif çözümleri incelemeye ve değerlendirmeye başlamalıdır. Eğer yeni bir sistem gerekliyse, bir sonraki soru bunun fizibil olup olmadığıdır. Fizibilite çalışması aşağıdakileri ele almalıdır:

- *Donanım ve yazılım gereksinimlerinin teknik yönleri. Kararlar henüz satıcıya özel olmayabilir, ancak donanım gereksinimlerinin (masaüstü bilgisayar, ana bilgisayar, süper bilgisayar veya mobil cihaz) ve yazılım gereksinimlerinin (tek kullanıcılı veya çok kullanıcılı işletim sistemleri, veritabanı türü ve yazılımı, uygulamalar tarafından kullanılacak programlama dilleri vb.) doğasını ele almalıdır.*
- *Sistem maliyeti. Kabul etmek gerekir ki "Bunu karşılayabilir miyiz?" gibi sıradan bir soru çok önemlidir. Cevap, ilk değerlendirmenin dikkatli bir şekilde gözden geçirilmesini zorlayabilir. Bin dolarlık bir soruna bir milyon dolarlık bir çözüm savunulamaz. Bir noktada, karar "şirket içinde" bir sistem kurmak veya üçüncü taraf bir satıcı sistemini satın almak (ve özelleştirmek) arasında olabilir. Uzun vadede, kuruluşun ihtiyaçlarına (mevcut ve gelecekteki) en iyi şekilde hizmet eden, maliyet açısından verimli bir çözüm bulmanız gerekir.*
- *İşletme maliyeti. Şirket, sistemi çalışır durumda tutmak için insan, teknik ve finansal kaynaklara sahip mi? Fizibilite çalışması, bu sistemin başarısını sağlamak için gerekli operasyonel prosedürleri uygulamak için gereken yönetim ve son kullanıcı desteğinin maliyetini içermeli mi? Bu yeni sistemin şirketin kültürü üzerindeki etkisi ne olurdu? İnsanların değişime karşı direnci asla küçümsenmemelidir.*[5]

[5] "Zappos'ta 210 çalışan 'patronsuz' çalışmak yerine ayrılmaya karar veriyor," Jena McGregor, The Washington Post, 8 Mayıs 2015.

"İnşa etmek" yerine "satın almayı" seçseniz bile, başarılı olması için sistem uygulamasının dikkatlice planlanması gerekir. Seçilen seçenek ne olursa olsun (inşa et veya satın al), maliyeti ve kültür değişikliklerini en aza indirirken değeri en üst düzeye çıkaracak şekilde çözümü kuruluş genelinde uygulamak için bir analiz yapılmalıdır. SDLC, sağlam planlama ve uygulama için bir çerçeve sağlar.

9-2b Analiz

Planlama aşamasında tanımlanan sorunlar, analiz aşamasında daha ayrıntılı olarak incelenir. Hem bireysel ihtiyaçların hem de organizasyonel ihtiyaçların makro bir analizi yapılmalı ve aşağıdaki gibi sorular ele alınmalıdır:

- Mevcut sistemin son kullanıcılarının gereksinimleri nelerdir?
- Bu gereksinimler, genel bilgi gereksinimlerine uyuyor mu?

SDLC'nin analiz aşaması, aslında, kullanıcı gereksinimlerinin kapsamlı bir denetimidir. Mevcut donanım ve yazılım sistemleri de analiz aşamasında incelenir. Analizin sonucu, sistemin işlevsel alanlarının, mevcut ve potansiyel sorunların ve fırsatların daha iyi anlaşılması olmalıdır.

Son kullanıcılar ve sistem tasarımcısı/tasarımcıları, süreçleri tanımlamak ve potansiyel sorunlu alanları ortaya çıkarmak için birlikte çalışmalıdır. Bu tür bir işbirliği, yeni sistemin değerlendirilebileceği uygun performans hedeflerini tanımlamak için hayati öneme sahiptir.

Kullanıcı gereksinimleri ve mevcut sistemlerin incelenmesinin yanı sıra, analiz aşaması aynı zamanda mantıksal bir sistem tasarımının oluşturulmasını da içerir. Mantıksal tasarım, uygun kavramsal veri modelini, girdileri, süreçleri ve beklenen çıktı gereksinimlerini belirtmelidir.

Mantıksal bir tasarım oluştururken, tasarımcı sistem akış şemaları, veri akış diyagramları (DFD'ler), hiyerarşik girdi işlem çıktı (HIPO) diyagramları gibi süreç diyagramları, bağlam diyagramları, kullanım senaryoları, varlık-ilişki (ER) diyagramları ve hatta bazı uygulama prototipleri gibi araçlar kullanabilir. Veritabanı tasarımının veri modelleme faaliyetleri, bu noktada tüm varlıkları ve özniteliklerini ve veritabanı içindeki varlıklar arasındaki ilişkileri keşfetmek ve tanımlamak için gerçekleşir.

Mantıksal sistemi tanımlamak, aynı zamanda veritabanı ortamındaki her bir süreç için sistem bileşenlerinin (modüllerin) işlevsel açıklamalarını da sağlar. Tüm veri dönüşümleri (süreçler), DFD'ler gibi sistem analizi araçları kullanılarak açıklanır ve belgelenir. Kavramsal veri modeli, bu süreçlere göre doğrulanır.

9-2c Detaylı Sistem Tasarımı

Detaylı sistem tasarımı aşamasında, tasarımcı sistem süreçlerinin tasarımını tamamlar. Tasarım, sistemi daha verimli bir bilgi üreteci haline getirmeye yardımcı olabilecek ekranlar, menüler, raporlar ve diğer cihazlar için gerekli tüm teknik özellikleri içerir. Eski sistemden yeni sisteme geçiş için adımlar belirlenir. Eğitim ilkeleri ve metodolojileri de planlanır ve yönetimin onayına sunulmalıdır.

Not

Çözümler geliştirmeye çalışırken, veritabanı tasarımcısı sorunların kaynağını aramalıdır. Birçok veritabanı sistemi, sorunların kaynağından ziyade belirtilerini tedavi etmek için tasarlandıklarından, son kullanıcıları memnun etmede başarısız olmuştur.

9-2d Uygulama

Uygulama aşamasında, donanım, DBMS yazılımı ve uygulama programları kurulur ve veritabanı tasarımı uygulanır. Uygulama aşamasının ilk evrelerinde, sistem teslim edilmeye hazır olana kadar bir kodlama, test etme ve hata ayıklama döngüsüne girer. Gerçek veritabanı oluşturulur ve sistem, tabloların ve görünümlerin, kullanıcı yetkilendirmelerinin vb. oluşturulmasıyla özelleştirilir.

Veritabanı içeriği, çeşitli yöntemler ve araçlar kullanılarak etkileşimli olarak veya toplu iş modunda yüklenebilir:

- Özelleştirilmiş kullanıcı programları
- Veritabanı arayüz programları
- Toplu iş programları, bir veritabanı yardımcı programı veya her ikisini kullanarak verileri farklı bir dosya yapısından içe aktaran dönüştürme programları

Sistem, kullanıma hazır olana kadar kapsamlı testlere tabi tutulur. Geleneksel olarak, yeni bir sistemin uygulanması ve test edilmesi, toplam geliştirme süresinin yüzde 50 ila 60'ını alırdı. Ancak, gelişmiş uygulama oluşturucuların ve hata ayıklama araçlarının ortaya çıkışı, kodlama ve test süresini önemli ölçüde azaltmıştır. Test tamamlandıktan sonra, nihai belgeler gözden geçirilir ve yazdırılır ve son kullanıcılar eğitilir. Bu aşamanın sonunda sistem tam olarak çalışır durumdadır, ancak sürekli olarak değerlendirilecek ve ince ayar yapılacaktır.

9-2e Bakım

Sistem faaliyete geçer geçmez, son kullanıcılar sistemde değişiklik talep etmeye başlar. Bu değişiklikler, üç türde gruplandırılabilecek sistem bakım faaliyetlerini oluşturur:

- *Sistem hatalarına yanıt olarak düzeltici bakım*
- *İş ortamındaki değişiklikler nedeniyle uyarlanabilir bakım*
- *Sistemi geliştirmek için mükemmelleştirici bakım*

Her yapısal değişiklik talebi SDLC adımlarının yeniden izlenmesini gerektirdiğinden, sistem anlamda her zaman SDLC'nin bir aşamasındadır bir .

Her sistemin önceden belirlenmiş bir operasyonel ömrü vardır, ancak gerçek ömrü algılanan faydasına bağlıdır. Belirli sistemlerin operasyonel ömrünü kısaltmak için çeşitli nedenler vardır. Hızlı teknolojik değişim, özellikle işlem hızına ve genişletilebilirliğe dayalı sistemler için bir nedendir. Bir diğer yaygın neden ise bir sistemin bakım maliyetidir.

Sistemin bakım maliyeti yüksekse, değeri şüpheli hale gelir. System Architect veya Relational Rose gibi bilgisayar destekli yazılım mühendisliği (CASE) araçları, makul bir süre içinde ve makul bir maliyetle daha iyi sistemler üretilmesine yardımcı olur. Buna ek olarak, CASE tarafından üretilen uygulamalar daha yapılandırılmış, daha iyi belgelendirilmiş ve özellikle standartlaştırılmıştır; bu da sistemlerin güncellenmesini ve bakımını daha kolay ve daha ucuz hale getirerek operasyonel ömrünü uzatma eğilimindedir.

bilgisayar destekli yazılım mühendisliği (CASE)
Bilgisayar destekli sistem mühendisliği olarak da bilinen, Sistem Geliştirme Yaşam Döngüsünün bir kısmını veya tamamını otomatikleştirmek için kullanılan araçlar.

Veritabanı Yaşam Döngüsü (DBLC)
Bir bilgi sistemi içindeki bir veritabanının geçmişini izleyen bir döngü. Döngü altı aşamaya ayrılır; ilk çalışma, veritabanı tasarımı, uygulama ve yükleme, test ve değerlendirme, işletme ve bakım ve evrim.

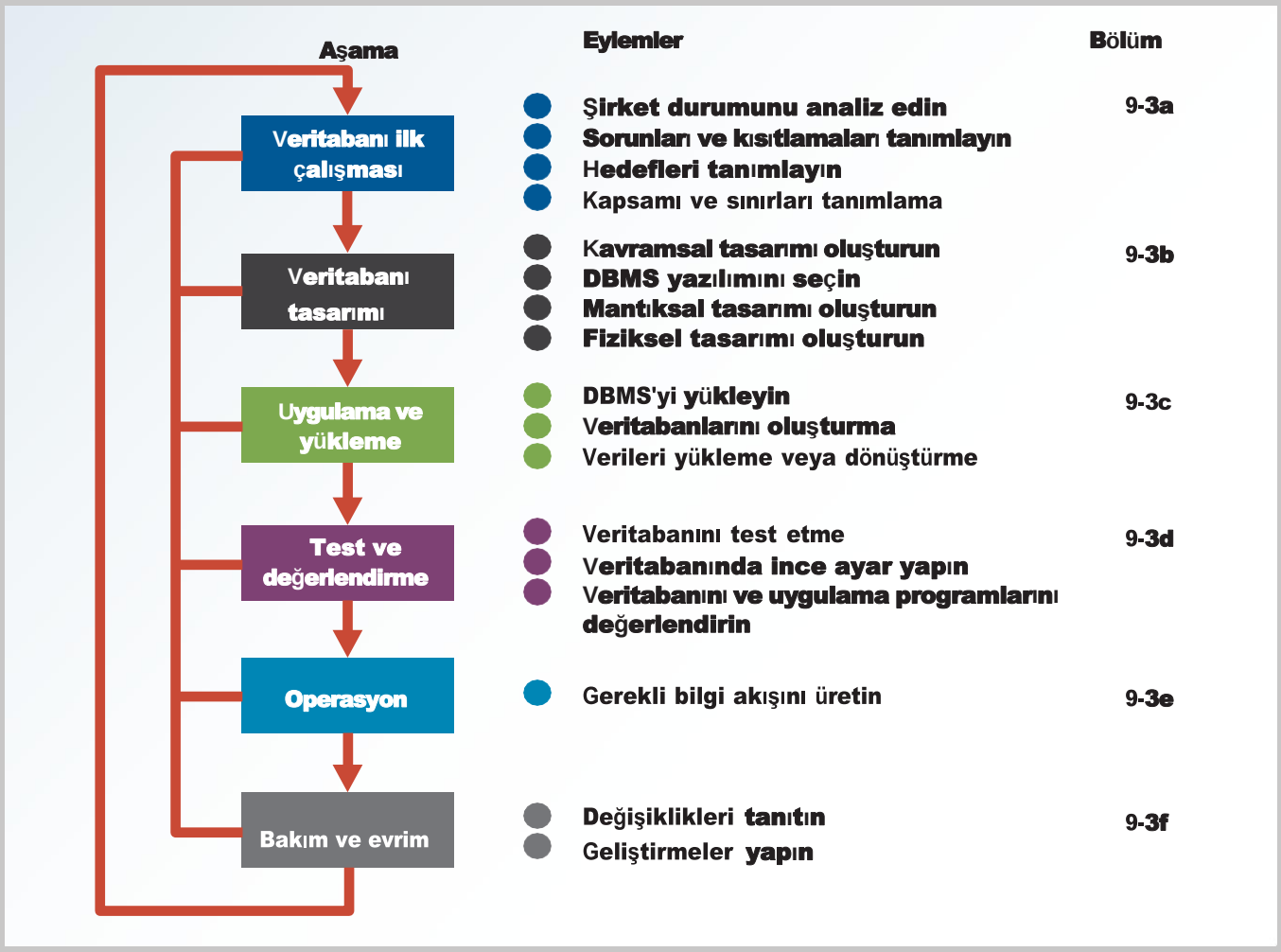
9-3 Veritabanı Yaşam Döngüsü

Daha büyük bilgi sistemi içinde, veritabanı da bir yaşam döngüsüne tabidir.

Döngüsü (DBLC) Şekil 9.3'te gösterildiği gibi altı aşamadan oluşur; veritabanı

Veritabanı Yaşam başlangıç çalışması, veritabanı tasarımı, uygulama ve yükleme, test ve değerlendirme, işletim ve bakım ve geliştirme.

Şekil 9.3 Veritabanı Yaşam Döngüsü (DBLC)



9-3a Veritabanı İlk Çalışması

Bir tasarımcı çağrıldıysa, mevcut sistemin şirket tarafından hayati önem verilen işlevleri yerine getirememiş olma ihtimali vardır. (Borular sızdırmadıkça tesisatçıyı aramazsınız.) Bu nedenle, mevcut sistemin şirket içindeki işleyişini incelemenin yanı sıra, tasarımcı mevcut sistemin nasıl ve neden başarısız olduğunu belirlemelidir. Bu, son kullanıcılarla konuşmak ve onları dinlemek için çok zaman harcamak anlamına gelir. Veritabanı tasarımı her ne kadar teknik bir iş olsa da aynı zamanda insan odaklıdır. Veritabanı tasarımcıları mükemmel iletişimciler olmalı ve ince ayarlanmış kişilerarası becerilere sahip olmalıdır.

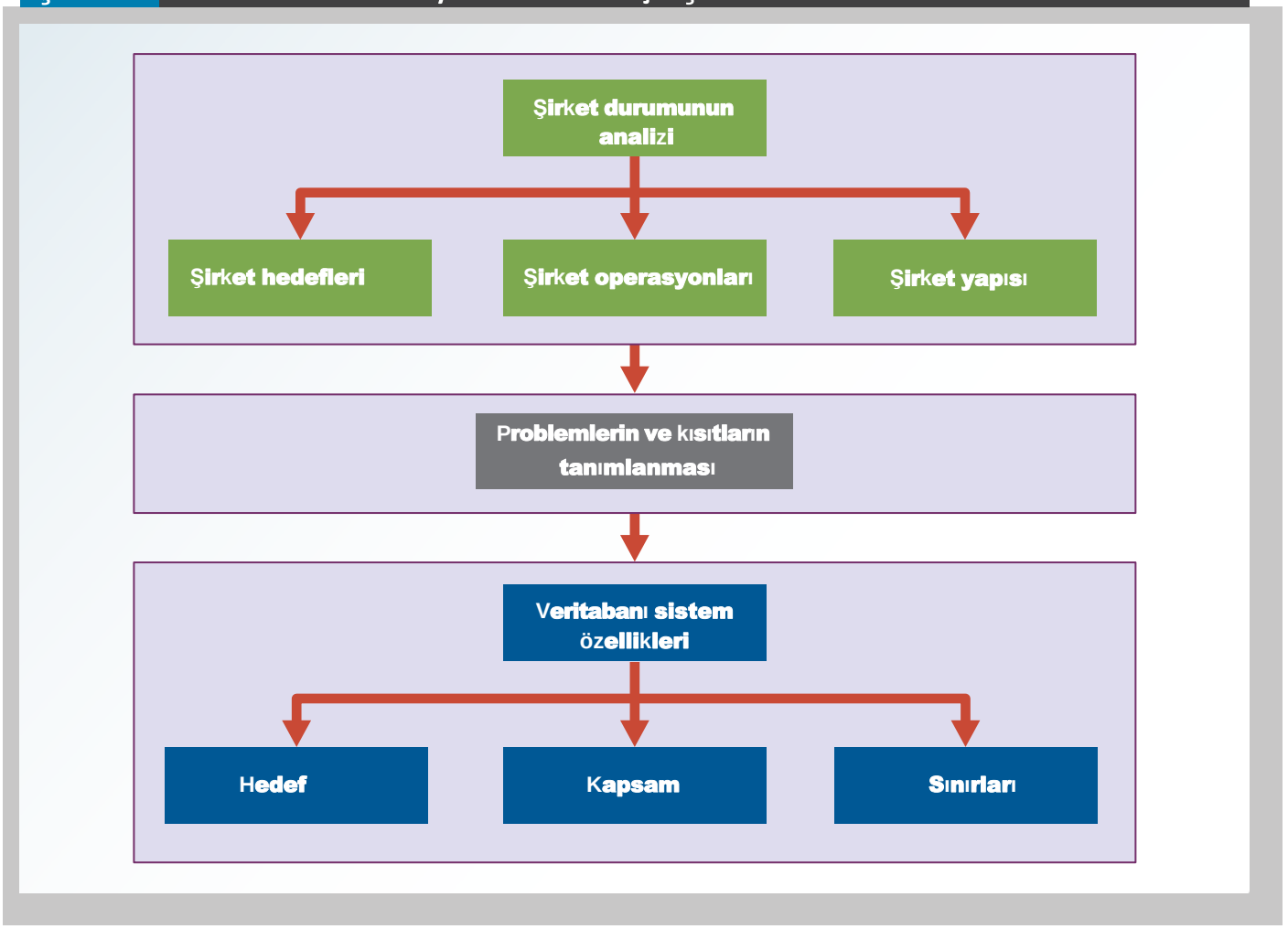
Veritabanı ortamının karmaşıklığına ve kapsamına bağlı olarak, veritabanı tasarımcısı yalnız bir operatör veya bir proje lideri, bir veya daha fazla kıdemli sistem analisti ve bir veya daha fazla alt düzey sistem analistinden oluşan bir sistem geliştirme ekibinin parçası olabilir. Tasarımcı kelimesi burada çok çeşitli tasarım ekibi kompozisyonlarını kapsayacak şekilde genel olarak kullanılmaktadır.

Veritabanı ilk çalışmasının genel amacı:

- Şirket durumunu analiz edin
- Sorunları ve kısıtlamaları tanımlayın
- Hedefleri tanımlayın
- Kapsamı ve sınırları tanımlama

Şekil 9.4, DBLC'nin ilk aşamasını başarıyla tamamlamak için gereken etkileşimli ve yinelemeli süreçleri göstermektedir. Veritabanının ilk çalışma aşamasının aşağıdakilere yol açtığını unutmayın:

Şekil 9.4 Veri Tabanındaki Faaliyetlerin Özeti İlk Çalışma



Veritabanı sistem hedeflerinin geliştirilmesi. Şekil 9.4'ü bir tartışma şablonu olarak kullanarak, bileşenlerinin her birini daha ayrıntılı olarak inceleyin.

Şirket Durumunu Analiz Edin

Şirket durumu, bir şirketin faaliyet gösterdiği genel koşulları, organizasyon yapısını ve misyonunu tanımlar. Şirket durumunu analiz etmek için, veritabanı tasarımcısı şirketin operasyonel bileşenlerini, nasıl çalıştıklarını ve nasıl etkileşime girdiklerini öğrenmelidir.

Aşağıdaki sorunların çözülmesi gerekir:

- *Kuruluşun genel çalışma ortamı nedir ve bu ortam içindeki misyonu nedir? Tasarım, organizasyonun misyonu tarafından yaratılan operasyonel talepleri karşılamalıdır. Örneğin, bir posta siparişi işletmesi, büyük olasılıkla veritabanı için bir üretim işletmesininkinden oldukça farklı operasyonel gereksinimlere sahiptir.*
- *Organizasyonun yapısı nedir? Kimin neyi kontrol ettiğini ve kimin kime rapor verdiğini bilmek, gerekli bilgi akışlarını, belirli rapor ve sorgu biçimlerini vb. tanımlamanız gerektiğinde oldukça yararlıdır.*

Sorunları ve Kısıtlamaları Tanımlayın

Tasarımcının hem resmi hem de gayri resmi bilgi kaynakları vardır. Şirket herhangi bir süredir varlığını sürdürüyorsa, zaten yerinde bir sistemi vardır (manuel veya bilgisayar tabanlı). Mevcut sistem nasıl işliyor? Sistem hangi girdiye ihtiyaç duyar? Belgeler ne işe yarar?

sistem oluşturur mu? Sistem çıktısı kim tarafından ve nasıl kullanılıyor? Kağıt izini incelemek çok bilgilendirici olabilir. Sistemin işleyişinin resmi versiyonuna ek olarak, daha gayri resmi, belki de daha gerçek versiyon da var; Tasarımcı, bunların nasıl farklı olduğunu görecek kadar kurnaz olmalıdır.

Sorunları tanımlama süreci başlangıçta yapılandırılmamış gibi görünebilir. Şirket son kullanıcıları genellikle şirket operasyonlarının daha geniş kapsamını tam olarak tanımlayamaz veya şirket operasyonları sırasında karşılaşılan gerçek sorunları tanımlayamaz. Genellikle, bir şirketin işleyişine ve sorunlarına ilişkin yönetimsel görüş, asıl rutin işi gerçekleştiren son kullanıcılarınkinden farklıdır.

İlk problem tanımlama sürecinde, tasarımcının çok geniş problem tanımları toplaması muhtemeldir. Örneğin, hızla büyüyen, ulusötesi bir üretim şirketinin başkanı tarafından ifade edilen aşağıdaki endişelere dikkat edin:

> **Her ne kadar hızlı büyüme sevindirici olsa da, yönetim ekibinin üyeleri, bu tür bir büyümenin yüksek bir müşteri hizmetleri standardını sürdürme yeteneğini baltalamaya başladığını ve belki de daha kötüsü, üretim standartları kontrolünü azalttığını düşünüyor.**

Problem tanımlama süreci hızlı bir şekilde bir dizi genel problem tanımına yol açar. Örneğin, pazarlama müdürü şu yorumu yapar:

> **Eski bir dosyalama sistemi ile çalışıyorum. 1.700'den fazla özel makine parçası üretiyoruz. Düzenli bir müşteri aradığında, çok hızlı bir envanter taraması alamıyoruz. Yeni bir müşteri ararsa, basit bir açıklama kullanarak mevcut parça araması yapamayız, bu nedenle genellikle satın aldığımız bir parça için bir makine kurulumu yaparız. Bu israf. Ve elbette, hızlı bir yanıt veremediğimizde bazı yeni müşteriler rahatsız oluyor.**

Prodüksiyon müdürü şu yorumu yapıyor:

> **En iyi ihtimalle, zamanlama amacıyla ihtiyaç duyduğum raporları oluşturmak saatler alıyor. Hızlı geri dönüşler için saatlerim yok. Hakkında bilgi sahibi olmadığım bir şeyi yönetmek zor.**
>
> **Hızlı ürün talebi yönlendirmesi almıyorum. Makine kurulumunu ele alalım. Şu anda, ya doğru stoğu bekleyen ya da yeni bir parçanın üretim için planlandığı zaman kendileri alan operatörlerim var. Çok daha düşük ücretli bir işçinin yapması gereken işleri bir operatörün yapmasını karşılayamam. Mevcut zamanlama ile etrafta çok fazla bekleme var. Çok fazla zaman kaybım ve programlarım düzeliyor. Fazla mesai faturamız çok saçma.**
>
> **Bazen zaten envanterde olan parçaları üretiyorum çünkü envanterde bulunanları planladığımızla eşleştiremiyoruz. Nakliye bana bağırıyor çünkü parçaları çıkaramıyorum ve çoğu zaman onları bir bölme aşağıda envanterde bulunduruyorlar. Bu bazen bize büyük paralara mal oluyor.**
>
> **Yeni raporların ofisime ulaşması günler hatta haftalar alabilir. Ve personeli, kesinti süresini, eğitimi vb. planlamak için bir ton rapora ihtiyacım var. ŞİMDİ ihtiyacım olan yeni raporları alamıyorum. İhtiyacım olan şey, kusur yüzdesi, yeniden çalışma yüzdesi, eğitimin etkinliği hakkında hızlı güncellemeler alma yeteneği, adını siz koyun. Vardiyaya, tarihe göre, planlamayı, eğitimi yönetmeme yardımcı olması için düşünebildiğim herhangi bir özelliğe göre bu tür raporlara ihtiyacım var, adını siz koyun.**

Bir makine operatörü şu yorumu yapıyor:

> **Eşyalarımı kurmak uzun zaman alıyor. John evrak işlerini zamanında alamadığı için programımı bozarsam, kurulum özelliklerini, başlangıç materyallerini, çöp kutusu atamalarını ve diğer şeyleri aramaya başlarım. Bazen sadece kurulum için iki veya üç saat harcıyorum. Artık neden programlara yetişemediğimi biliyorsunuz. Üretken olmaya çalışıyorum ama işimi yapmaya hazırlanmak için çok fazla zaman harcıyorum.**

İlk bildirimlerden sonra, veritabanı tasarımcısı, şirket operasyonlarının daha geniş çerçevesi içindeki sorunları tanımlamaya yardımcı olacak ek bilgileri oluşturmak için dikkatli bir şekilde araştırmaya devam etmelidir. Pazarlama yöneticisinin müşterisi sorunu, daha geniş pazarlama departmanı faaliyetleri kümesine nasıl uyuyor? Müşterinin sorununa yönelik çözüm, pazarlama departmanının ve şirketin geri kalanının hedeflerine ulaşılmasına nasıl yardımcı olur? Pazarlama departmanının faaliyetleri ile diğer departmanların faaliyetleri arasında nasıl bir ilişki var? Bu son soru özellikle önemlidir. Pazarlama ve üretim departmanı yöneticileri tarafından açıklanan sorunlarda ortak konular olduğunu unutmayın. Envanter sorgulama süreci iyileştirilebilirse, her iki departmanın da sorunlarının en azından bir kısmına basit çözümler bulması muhtemeldir.

Kesin cevaplar bulmak, özellikle iş birimleri arasındaki operasyonel ilişkilerle ilgili olarak önemlidir. Önerilen bir sistem pazarlama departmanının sorunlarını çözecek, ancak üretim departmanının sorunlarını daha da kötüleştirecekse, çok fazla ilerleme kaydedilmeyecektir. Bir benzetme kullanarak, evinizin su faturasının çok yüksek olduğunu varsayalım. Sorunu tespit ettiniz, musluklar sızdırıyor. Çözüm? Dışarı çıkıyorsunuz ve evin su kaynağını kesiyorsunuz. Bununla birlikte, bu yeterli bir çözüm mü, yoksa musluk yıkayıcılarının değiştirilmesi sorunu çözmek için daha iyi bir iş çıkarır mı? Bu senaryoyu basit bulabilirsiniz, ancak hemen hemen her deneyimli veritabanı tasarımcısı, kuşkusuz daha karmaşık olmalarına rağmen, benzer veritabanı problem çözme örneklerini bulabilir.

En eksiksiz ve doğru problem tanımı bile her zaman mükemmel çözüme yol açmaz. Gerçek dünya genellikle zaman, bütçe ve personel gibi kısıtlamalar getirerek en zarif veritabanının tasarımını bile sınırlamaya çalışır. Bir ay içinde ve 12.000 ABD Doları tutarında bir bütçe dahilinde bir çözüme sahip olmanız gerekiyorsa, 100.000 ABD Doları tutarında bir veritabanını geliştirmek iki yılınızı alamaz. Tasarımcı, neyin mükemmel olduğunu ve neyin mümkün olduğunu ayırt etmeyi öğrenmelidir.

Hedefleri Tanımlayın

Önerilen bir veritabanı sistemi, en azından problem keşif sürecinde tanımlanan büyük sorunların çözülmesine yardımcı olacak şekilde tasarlanmalıdır. Sorunların listesi ortaya çıktıkça, birkaç ortak kaynağın keşfedilmesi muhtemeldir. Önceki örnekte, hem pazarlama müdürü hem de üretim müdürü envanter verimsizliklerinden muzdarip görünüyor. Tasarımcı, daha verimli parça yönetimi için zemin hazırlayan bir veri tabanı oluşturabilirse, her iki departman da kazançlı çıkar. Bu nedenle ilk amaç, verimli bir envanter sorgulama ve yönetim sistemi oluşturmak olabilir.

İlk çalışma aşamasının aynı zamanda önerilen problem çözümlerini de verdiğini unutmayın. Tasarımcının görevi, veritabanı sistem hedeflerinin son kullanıcı(lar) tarafından öngörülenlere karşılık geldiğinden emin olmaktır. Her durumda, veritabanı tasarımcısı aşağıdaki soruları ele almaya başlamalıdır:

- Önerilen sistemin ilk hedefi nedir?
- Sistem, şirketteki diğer mevcut veya gelecekteki sistemlerle arayüz oluşturacak mı?
- Sistem, verileri diğer sistemlerle veya kullanıcılarla paylaşacak mı?

Kapsam ve Sınırları Tanımlama

Tasarımcı, kapsam ve sınırlar olmak üzere iki sınır kümesini tanımalıdır. Sistemin **kapsamı,** operasyonel gereksinimlere göre tasarımın kapsamını tanımlar. Veritabanı tasarımı tüm organizasyonu mu, organizasyon içindeki bir veya daha fazla departmanı mı yoksa tek bir departmanın bir veya daha fazla işlevini mi kapsayacak? Tasarımcı "basketbol sahasının boyutunu" bilmelidir. Kapsamın bilinmesi, gerekli veri yapılarını, varlıkların türünü ve sayısını, veritabanının fiziksel boyutunu vb. tanımlamaya yardımcı olur.

Kapsam
Operasyonel gereksinimlere göre tasarımın kapsamını tanımlayan bir sistemin parçası.

Önerilen sistem aynı zamanda sistemin dışında olan ve sınırlar olarak bilinen **sınırlara** da tabidir. Herhangi bir tasarımcıya "Dünyada her zaman var" veya "Tasarımı bir araya getirmek için sınırsız bir bütçe ve gerektiği kadar insan kullanın" denildi mi? Sınırlar ayrıca mevcut donanım ve yazılım tarafından da belirlenir. İdeal olarak, tasarımcı sistem hedeflerini en iyi şekilde gerçekleştirecek donanım ve yazılımı seçebilir. Aslında, yazılım

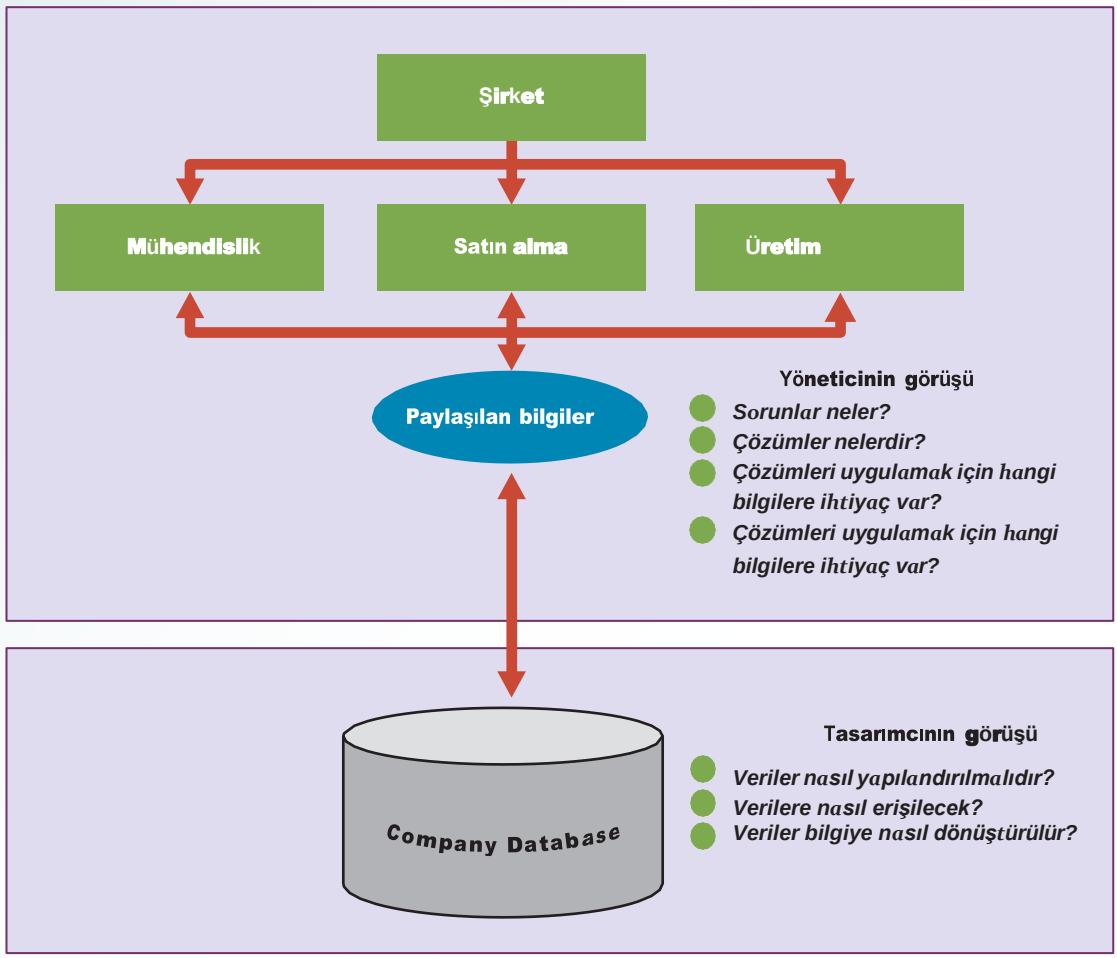
Sınırları
Önerilen herhangi bir sistemin tabi olduğu dış sınırlar. Bu sınırlar bütçeleri, personeli ve mevcut donanım ve yazılımı içerir.

seçimi, Sistem Geliştirme Yaşam Döngüsünün önemli bir yönüdür. Ne yazık ki, gerçek dünyada, bir sistem genellikle mevcut donanım etrafında tasarlanmalıdır. Böylece, kapsam ve sınırlar, tasarımı belirli bir kalıba zorlayan faktörler haline gelir ve tasarımcının görevi, bu kısıtlamalar dahilinde mümkün olan en iyi sistemi tasarlamaktır. (Sorun tanımlarının ve hedeflerin bazen sistem kapsamı ve sınırlarını karşılayacak şekilde yeniden şekillendirilmesi gerektiğini unutmayın.)

9-3b Veritabanı Tasarımı

DBLC'nin ikinci aşaması, şirket operasyonlarını ve hedeflerini destekleyecek veritabanı modelinin tasarımına odaklanmaktadır. Bu, tartışmasız en kritik DBLC aşamasıdır ve nihai ürünün kullanıcı ve sistem gereksinimlerini karşıladığından emin olur. Veritabanı tasarımı sürecinde, veritabanı modelini oluşturmak için gereken veri özelliklerine odaklanmanız gerekir. Bu noktada, sistem içindeki verinin iki görünümü vardır, bir bilgi kaynağı olarak verinin iş görünümü ve tasarımcının veri yapısına bakışı, erişimi ve veriyi bilgiye dönüştürmek için gereken faaliyetler. Şekil 9.5 bu görünümlerle karşılaştırır. Ne ve nasıl terimlerine bakarak farklı görünümleri özetleyebileceğinizi unutmayın. Verilerin tanımlanması, DBLC'nin ikinci aşamasının ayrılmaz bir parçasıdır.

Şekil 9.5 İki Veri Görünümü: İşletme Yöneticisi ve Tasarımcı



DBLC'de tasarım aşamasını tamamlamak için gereken prosedürleri incelerken şu noktaları unutmayın:

• Veritabanı tasarımı süreci, daha büyük bir sistemin analizi ve tasarımı ile gevşek bir şekilde ilgilidir. Veri bileşeni, daha büyük bir bilgi sisteminin yalnızca bir öğesidir.

• Sistem analistleri veya sistem programcıları, diğer sistem bileşenlerini tasarlamaktan sorumludur. Faaliyetleri, veri tabanındaki verilerin faydalı bilgilere dönüştürülmesine yardımcı olacak prosedürleri oluşturur.

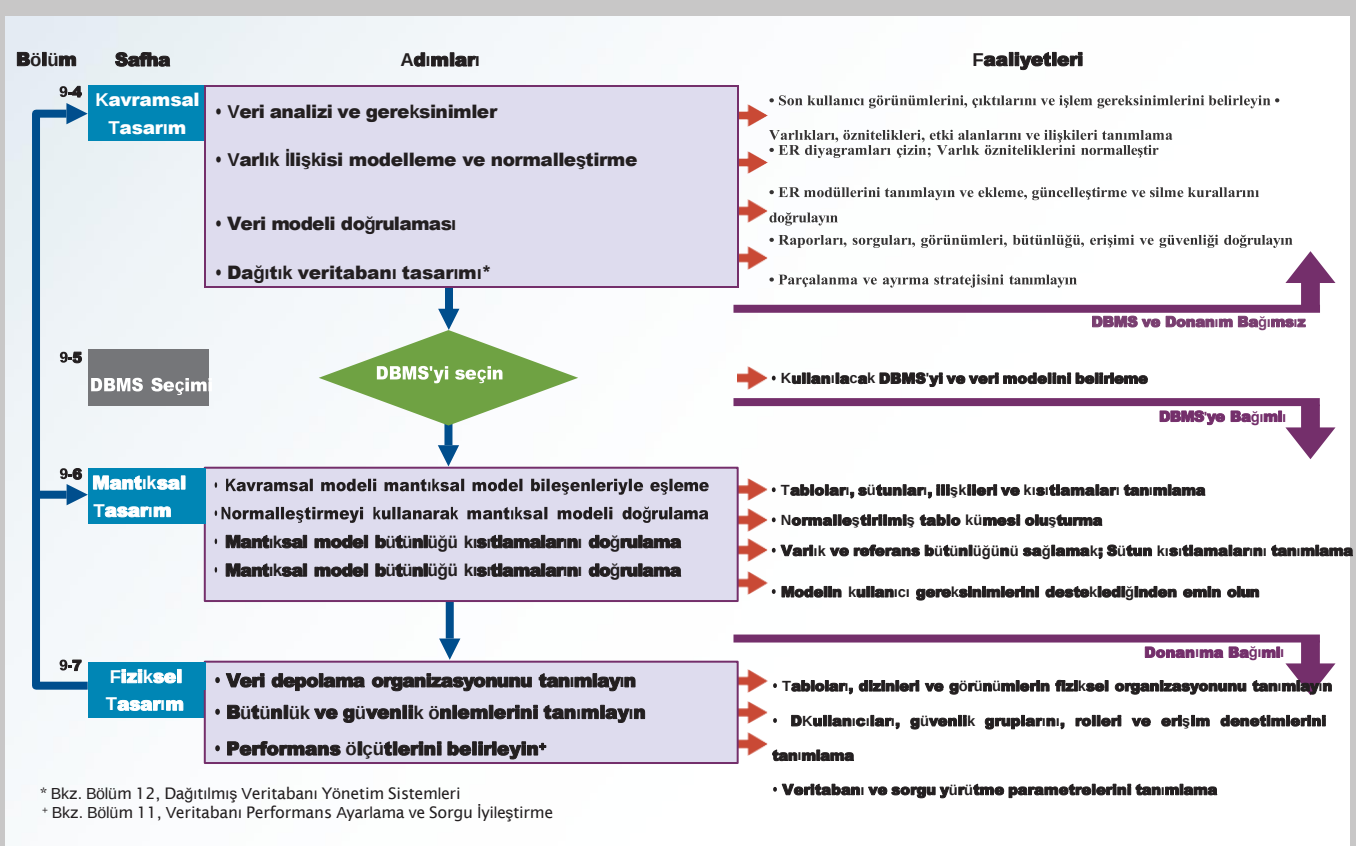
• Veritabanı tasarımı sıralı bir süreç oluşturmaz. Bunun yerine, önceki adımları izlemek için tasarlanmış sürekli geri bildirim sağlayan yinelemeli bir işlemdir.

Veritabanı tasarım süreci Şekil 9.6'da gösterilmiştir. Şekil, kavramsal, mantıksal ve fiziksel tasarım olmak üzere üç temel aşamanın yanı sıra, oluşturulacak mantıksal ve fiziksel tasarımların türünü belirlemek için kritik olan DBMS seçim kararı olduğunu göstermektedir. Tasarım süreci kavramsal tasarım ile başlar ve mantıksal ve fiziksel tasarım aşamalarına geçer. Her aşamada, veri modeli tasarımı hakkında daha fazla ayrıntı belirlenir ve belgelenir. Kavramsal tasarımı, son kullanıcı tarafından görülen genel veriler, mantıksal tasarımı DBMS tarafından görülen veriler ve fiziksel tasarımı işletim sisteminin depolama yönetim cihazları tarafından görülen veriler olarak düşünebilirsiniz.

Veritabanı tasarımlarının ve uygulamalarının ezici çoğunluğu ilişkisel modele dayanır ve bu nedenle ilişkisel model yapılarını ve tekniklerini kullanır. Tasarım faaliyetlerini tamamladığınızda, uygulanmaya hazır eksiksiz bir veritabanı tasarımına sahip olacaksınız.

Veritabanı tasarım faaliyetleri Bölüm 9-4'te (Kavramsal Tasarım) ayrıntılı olarak ele alınmıştır, 9-5 (DBMS Yazılım Seçimi), 9-6 (Mantıksal Tasarım) ve 9-7 (Fiziksel Tasarım).

Şekil 9.6 Veritabanı Tasarım Süreci



Sanallaştırma
Temel alınan fiziksel bilgi işlem kaynaklarından bağımsız bilgi işlem kaynaklarının mantıksal temsillerini oluşturan bir teknik.

Online İçerik
İki ek www.cengage.com, basit, gerçek dünya veritabanı geliştirmenin kısa bir örneğini sağlar: Ek B, Üniversite Laboratuvarı: Kavramsal Tasarım ve Ek C, Üniversite Laboratuvarı: Kavramsal Tasarım Doğrulaması, Mantıksal Tasarım ve Uygulama.

9-3c Uygulama ve Yükleme

Veritabanı tasarım aşamasının çıktısı, tabloların, özniteliklerin, etki alanlarının, görünümlerin, dizinlerin, güvenlik kısıtlamalarının ve depolama ve performans yönergelerinin oluşturulmasını ayrıntılandıran bir dizi talimattır. Bu aşamada, aslında tüm bu tasarım özelliklerini uygularsınız.

DBMS'yi yükleyin

Bu adım, yalnızca sistem için DBMS'nin yeni bir özel örneği gerektiğinde gereklidir. Çoğu durumda, kuruluş, teknolojiye yapılan yatırımlardan ve çalışanların halihazırda geliştirdiği becerilerden yararlanmak için belirli bir DBMS'yi standart haline getirmiş olacaktır. DBMS, yeni bir sunucuya veya mevcut sunuculara kurulabilir. Mevcut trendlerden biri sanallaştırma olarak adlandırılıyor. Sanallaştırma, temel alınan fiziksel bilgi işlem kaynaklarından bağımsız bilgi işlem kaynaklarının mantıksal temsillerini oluşturan bir tekniktir. Bu teknik, sanal sunucuların, sanal depolamanın ve sanal özel ağların oluşturulması gibi birçok bilgi işlem alanında kullanılır. Bir veritabanı ortamında, veritabanı sanallaştırma, paylaşılan donanım üzerinde çalışan bir sanal sunucuda DBMS'nin yeni bir örneğinin kurulmasını ifade eder. Bu normalde sistem ve ağ yöneticilerinin sunucu yapılandırmasında ve ağ yönlendirmesinde uygun kullanıcı grupları ve hizmetler oluşturmasını içeren bir görevdir. Diğer bir yaygın eğilim, Microsoft Azure SQL Veritabanı Hizmeti veya Amazon İlişkisel Veritabanı Hizmetleri (RDS) gibi bulut veritabanı hizmetlerinin kullanılmasıdır. Bu yeni nesil hizmetler, kullanıcıların gerektiğinde kolayca yönetilebilen, test edilebilen ve ölçeklendirilebilen veritabanları oluşturmasına olanak tanır.

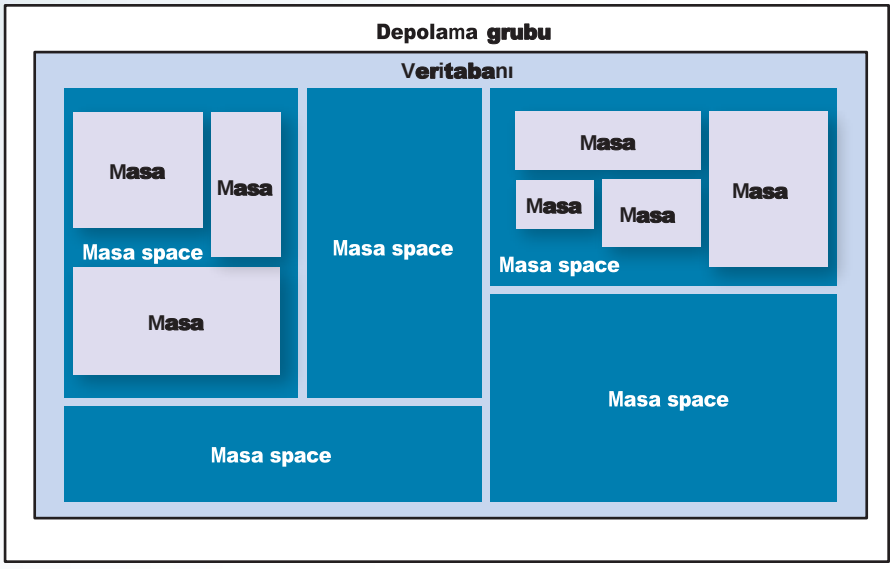
Veritabanlarını Oluşturun

Modern ilişkisel DBMS'lerin çoğunda, yeni bir veritabanı uygulaması, son kullanıcı tablolarını barındırmak için depolamayla ilgili özel yapıların oluşturulmasını gerektirir. Yapılar genellikle depolama grubunu (veya dosya gruplarını), tablo alanlarını ve tabloları içerir. Şekil 9.7'de, bir depolama grubunun birden fazla tablo alanı içerebileceği ve bir tablo alanının birden fazla tablo içerebileceği gösterilmektedir.

Mantıksal tasarımın uygulanması, veritabanı yöneticisinin depolama gruplarını, tablo alanlarını, tabloları oluşturmasını ve son olarak belirli kullanıcılara veya gruplara tablolara erişim hakları atamasını gerektirir. Örneğin, aşağıdaki komut kullanılarak, PRO FESSOR adlı bir tabloya erişim hakları, kimlik kodu mhendley olan Madison Hendley kullanıcısına verilebilir

MHENDLEY'E PROFESÖR SEÇIMI YAPIN;

Şekil 9.7 Bir Veritabanı Ortamının Fiziksel Organizasyonu



Erişim hakları, tüm tablolar yerine görünümlerle sınırlı olabilir. İlişkisel ortamda veritabanı erişimi için görünümlerin oluşturulması gerekli değildir, ancak görünümler güvenlik açısından tercih edilir.

Verileri Yükleyin veya Dönüştürün

Veritabanı oluşturulduktan sonra, verilerin veritabanı tablolarına yüklenmesi gerekir. Genellikle, verilerin sistemin önceki sürümünden geçirilmesi gerekir. Genellikle, sisteme dahil edilecek verilerin birden çok kaynaktan toplanması gerekir. En iyi senaryoda, tüm veriler ilişkisel bir veritabanında olacaktır, böylece yeni veritabanına kolayca aktarılabilir. Bununla birlikte, bazı durumlarda verilerin diğer ilişkisel veritabanlarından, ilişkisel olmayan veritabanlarından, düz dosyalardan, eski sistemlerden ve hatta manuel kağıt ve kalem sistemlerinden içe aktarılması gerekebilir. Veri biçimi yeni veritabanına doğrudan içe aktarmayı desteklemiyorsa, verileri içe aktarmak üzere yeniden biçimlendirmek için dönüştürme programlarının oluşturulması gerekebilir. En kötü senaryoda, verilerin çoğunun veritabanına manuel olarak girilmesi gerekebilir. Veriler yüklendikten sonra DBA, veritabanını test etmek ve değerlendirmek için uygulama geliştiricileriyle birlikte çalışır.

Mevcut verileri bulut tabanlı bir veritabanı hizmetine yüklemek bazen pahalı olabilir. Bunun nedeni, çoğu bulut hizmetinin yalnızca depolanacak veri hacmine göre değil, aynı zamanda ağ üzerinden dolaşan veri miktarına göre de fiyatlandırılmasıdır. Bu gibi durumlarda, 1 TB'lık bir veritabanını yüklemek çok pahalı bir teklif olabilir. Bu nedenle, sistem yöneticileri, "gizli" maliyetlerin olmayacağından emin olmak için bulut hizmeti sözleşmelerinin şartlarını okurken ve müzakere ederken çok dikkatli olmalıdır.

9-3d Test ve Değerlendirme

Tasarım aşamasında, veritabanının bütünlüğünü, güvenliğini, performansını ve tanınabilirliğini sağlamak için kararlar alındı. Uygulama ve yükleme sırasında bu planlar uygulamaya konuldu. Test ve değerlendirmede DBA, beklendiği gibi çalıştığından emin olmak için veritabanını test eder ve ince ayar yapar. Bu aşama, uygulama programlama ile birlikte gerçekleşir. Programcılar, programların kodlanması sırasında uygulamaların prototipini oluşturmak için veritabanı araçlarını kullanır. Rapor oluşturucular, ekran boyacıları ve menü oluşturucular gibi araçlar özellikle uygulama programcıları için yararlıdır.

Veritabanını Test Edin

Bu adım sırasında DBA, verilerin bütünlüğünü ve gizliliğini koruduğundan emin olmak için veritabanını test eder. Veri bütünlüğü, birincil ve yabancı anahtar kurallarının uygun kullanımı yoluyla DBMS tarafından zorlanır. Birçok DBMS, etki alanı kısıtlamalarının ve veritabanı tetikleyicilerinin oluşturulmasını da destekler. Test, bu kısıtlamaların uygun şekilde tasarlanmasını ve uygulanmasını sağlayacaktır. Veri bütünlüğü, kapsamlı bir veri yönetimi çerçevesinin parçası olan, uygun şekilde uygulanan veri yönetimi politikalarının da bir sonucudur. Bu konuyla ilgili daha ayrıntılı bir çalışma için, Bölüm 16, Veritabanı Yönetimi ve Güvenliği'ndeki "DBA'nın Yönetimsel Rolü" bölümüne bakın.

Daha önce, kullanıcılara verilere erişim izni vermek için kullanıcılar ve roller oluşturuluyordu. Bu aşamada, yalnızca bu ayrıcalıklar test edilmekle kalmamalı, aynı zamanda veri gizliliği ve güvenliğine ilişkin daha geniş bir bakış açısı da ele alınmalıdır. Şirket veri tabanında depolanan veriler, yabancı kullanıcıların erişimine karşı korunmalıdır. (Öğrencilerin bir öğrenci veritabanına erişimi varsa veya çalışanların bordro verilerine erişimi varsa, olası sonuçları tahmin etmek çok fazla hayal gücü gerektirmez!) Sonuç olarak, en azından aşağıdakileri test etmelisiniz:

- *Fiziksel güvenlik, yalnızca yetkili personelin belirli alanlara fiziksel olarak erişmesine izin verir. Bununla birlikte, veritabanı uygulamasının türüne bağlı olarak, fiziksel güvenlik oluşturmak her zaman pratik olmayabilir. Örneğin, bir üniversite öğrencisi araştırma veritabanı, fiziksel güvenlik için olası bir aday değildir.*
- Parola güvenliği, erişim haklarının belirli yetkili kullanıcılara atanmasına izin verir. Parola güvenliği genellikle işletim sistemi düzeyinde oturum açma sırasında uygulanır.

- *Erişim hakları, veritabanı yazılımı kullanılarak oluşturulabilir. Erişim haklarının atanması, veritabanları, Masalar, görünümler, sorgular ve raporlar gibi önceden belirlenmiş nesneler üzerindeki işlemleri (OLUŞTURMA, GÜNCELLEŞTIRME, SILME vb.) kısıtlayabilir.*
- *Denetim izleri genellikle erişim ihlallerini kontrol etmek için DBMS tarafından sağlanır. Denetim izi, sonradan ortaya çıkan bir cihaz olmasına rağmen, yalnızca varlığı yetkisiz kullanımı caydırabilir.*
- *Uygulama kodu ve veritabanı güvenliği. Uygulamayı SQL eklemeleri ve gereksiz ayrıcalıklar gibi kod güvenlik açıkları için test edin. Veritabanı grubu, uygulamayı korumak için gerekli güvenlik önlemlerini oluşturmak için ağ ve güvenlik gruplarıyla yakın bir şekilde çalışmalıdır.*
- *Veri şifreleme, bazı veritabanı güvenlik katmanlarını ihlal etmiş olabilecek yetkisiz kullanıcılar için verileri işe yaramaz hale getirebilir.*
- *Disksiz iş istasyonları, son kullanıcıların iş istasyonlarından bilgileri aşağı yükleyemeden veritabanına erişmesine olanak tanır.*

Güvenlik sorunları hakkında daha ayrıntılı bir tartışma için, Bölüm 16, Veritabanı Yönetimi ve Güvenlik'e bakın.

Veritabanında ince ayar yapın

Veritabanı performansını değerlendirmek zor olabilir, ancak bu, tipik olarak veritabanı uygulamasındaki en önemli faktörlerden biridir. Farklı sistemler, veritabanı üzerinde farklı performans gereksinimleri ortaya koyacaktır. Tipik performans göstergeleri sorgu işlem hacmi (desteklenen eş zamanlı sorgu sayısı), sorgu yanıt süresi (bir sorgunun tamamlanması için geçen ortalama süre) ve sorgu bekleme süresidir (bir sorgunun kuyrukta beklediği ortalama süre). Müşteriyle etkileşimde olan sistemler, tipik olarak yüksek hacimli ekleme, güncelleme ve silme işlemleri sırasında üstün performans gerektirir. Karar destek sistemleri gibi diğer sistemler, karmaşık veri çekme (erişim) görevleri için üstün performans gerektirebilir. Veritabanının çeşitli görevlerdeki performansını, veritabanının bulunduğu donanım ve yazılım ortamı dahil olmak üzere birçok faktör etkileyebilir. Doğal olarak, verilerin özellikleri ve hacmi de veritabanı performansını etkiler - 10 demetin (tuple) aranması, 100.000 demetin aranmasından daha hızlıdır. Veritabanı performansındaki diğer önemli faktörler arasında veri yerleşimi, erişim yolu tanımı, indekslerin kullanımı ve arabellek boyutu gibi sistem ve veritabanı yapılandırma parametreleri bulunur. Veritabanı performansı konularının daha derinlemesine tartışılması için, Bölüm 11, Veritabanı Performans Ayarlama ve Sorgu Optimizasyonu'na bakın.

Veritabanını ve uygulama programlarını değerlendirin

Veritabanı ve uygulama programları oluşturulup test edilirken, sistemin de daha bütüncül bir yaklaşım kullanılarak değerlendirilmesi gerekir. Tek tek bileşenlerin test edilmesi ve değerlendirilmesi, tüm bileşenlerin kullanıcıların ihtiyaçlarını karşılamak için düzgün bir şekilde etk-ileşime girmesini sağlamak için çeşitli daha geniş sistem testleriyle sonuçlanmalıdır. Bu aşa-mada, entegrasyon sorunları ve dağıtım planları iyileştirilir, kullanıcı eğitimi gerçekleştirilir ve sistem dokümantasyonu sonlandırılır. Sistem nihai onayı aldıktan sonra, kuruluş için sürdürülebilir bir kaynak olmalıdır. Veritabanında yer alan verilerin kaybolmaya karşı korun-duğundan emin olmak için yedekleme ve kurtarma planları test edilir.

Zamanında veri kullanılabilirliği hemen hemen her veritabanı için çok önemlidir. Ne yazık ki, veritabanı istenmeyen silmeler, elektrik kesintileri ve diğer nedenlerle veri kaybedebilir. Veri yedekleme ve kurtarma prosedürleri, tutarlı verilerin kullanılabilirliğini sağlayan bir emniyet valfi oluşturur. Tipik olarak, veritabanı satıcıları, bir donanım arızası durumunda veritabanının sürekli çalışmasını sağlamak için kesintisiz güç kaynağı (UPS) birimleri, RAID depolama aygıt-ları, kümelenmiş sunucular ve veri çoğaltma teknolojileri gibi hataya dayanıklı bileşenlerin kul-lanımını teşvik eder. Bu bileşenlerle bile, yedekleme ve geri yükleme işlevleri günlük veritabanı işlemlerinin çok önemli bir bölümünü oluşturur. Bazı DBMS'ler, veritabanı yöneticisinin diskler, DVD'ler, teypler ve çevrimiçi depolama gibi kalıcı depolama aygıtlarına otomatik veritabanı yedeklemeleri planlamasına olanak tanıyan işlevler sağlar. Veritabanı yedeklemeleri farklı düzeylerde gerçekleştirilebilir:

- Tüm veritabanının **tam yedeklemesi** veya dökümü. Bu durumda, tüm veritabanı nesneleri bütünüyle yedeklenir.

Tam yedekleme
Veritabanının tamamının ayrı bir konuma kaydedilen ve düzenli aralıklarla güncelleştirilen tam bir kopyası (veritabanı dökümü olarak da adlandırılır). Tam yedekleme, fiziksel bir felaket veya veritabanı bütünlüğü arızasından sonra tüm verilerin tam olarak kurtarılmasını sağlar.

• Yalnızca son tam yedeklemeden bu yana güncellenen veya değiştirilen nesnelerin yedeklendiği veritabanının diferansiyel bir yedeği.

• Yalnızca veritabanının önceki bir yedek kopyasına yansıtılmayan işlem günlüğü işlemlerini yedekleyen bir işlem günlüğü yedeği. Bu durumda, başka hiçbir veritabanı nesnesi yedeklenmez. (İşlem günlüğünün tam bir açıklaması için, Bölüm 10, İşlem Yönetimi ve Eşzamanlılık Denetimi'ne bakın.)

Diferansiyel yedekleme
Yalnızca veritabanında yapılan son değişikliklerin kopyalandığı bir veritabanı yedekleme düzeyi.

İşlem günlüğü yedekleme
Yalnızca veritabanının önceki bir yedek kopyasına yansıtılmayan işlem günlüğü işlemlerinin yedeği.

Veritabanı yedeği, genellikle veritabanının kendisinden farklı bir binada güvenli bir yerde saklanır ve yangın, hırsızlık, sel ve diğer olası felaketler gibi tehlikelere karşı korunur. Yedeklemenin temel amacı, bir donanım veya yazılım arızasının ardından veritabanı geri yüklemesini garanti etmektir.

Veritabanlarını ve sistemleri rahatsız eden arızalar genellikle yazılım, donanım, programlama muafiyetleri, işlemler veya dış faktörlerden kaynaklanır. Masa 9.1, veritabanı hatasının en yaygın kaynaklarını özetler.

Şekil Veritabanı hatasının yaygın kaynakları

Kaynak	Açıklama	Örnek
Yazılım	Yazılım kaynaklı arızalar işletim sistemine, DBMS yazılımına, uygulama programlarına veya virüslere ve diğer kötü amaçlı yazılımlara kadar izlenebilir.	Oracle E-Business Suite'te bulunan bir güvenlik açığı, kimliği doğrulanmamış bir saldırganın kritik verileri oluşturmasına, değiştirmesine veya silmesine olanak tanıdı.[6]
Donanım	Donanım kaynaklı arızalar arasında bellek yongası hataları, disk çökmeleri, bozuk disk sektörleri ve disk dolu hatalar yer alabilir.	Bir veritabanı sistemindeki bozuk bir bellek modülü veya çoklu sabit disk arızası, onu aniden durdurabilir.
Programlama muafiyetleri	Uygulama programları veya son kullanıcılar, belirli koşullar tanımlandığında işlemleri geri alabilir. Programlama muafiyetleri, bilgisayar korsanları tarafından yararlanılabilecek kötü amaçlı veya yanlış test edilmiş koddan da kaynaklanabilir.	Son zamanlarda, bir grup kimliği belirsiz bilgisayar korsanı, New York Federal Rezerv Bankası'na Bangladeş merkez bankasından Filipinler'deki hesaplara 81 milyon dolar aktarması talimatını verdi. Bilgisayar korsanları, PDF okuyucu kılığına girmiş kötü amaçlı yazılımlar tarafından enjekte edilen sahte mesajlar kullandı.[7]
Hareket	Sistem kilitlenmeleri algılar ve işlemlerden birini iptal eder. (Bölüm 10'a bakın.)	Kilitlenme, aynı anda birden fazla işlem yürütülürken ortaya çıkar.
Dış faktörler	Power Backup jeneratörleri, bir sistem deprem, sel veya diğer doğal afetler gibi dış etkenlerden muzdarip olduğunda özellikle önemlidir.	Son birkaç yılda, Louisiana genel elektrik kesintilerine neden olan birkaç kasırga yaşadı. Kesintiler, BT hizmet sağlayıcılarının yanı sıra genel işletmeleri de etkiledi.

Hatanın türüne ve kapsamına bağlı olarak, kurtarma işlemi kısa vadeli küçük bir rahatsızlıktan uzun vadeli büyük bir yeniden oluşturmaya kadar değişir. Gerekli kurtarma işleminin kapsamı ne olursa olsun, kullanılabilir bir yedekleme olmadan kurtarma mümkün değildir.

Veritabanı kurtarma genellikle öngörülebilir bir senaryoyu izler. İlk olarak, gerekli geri kazanımın türü ve kapsamı belirlenir. Veritabanının tamamının tutarlı bir duruma kurtarılması gerekiyorsa, kurtarma işlemi, veritabanının bilinen tutarlı bir durumdaki en son yedek kopyasını kullanır. Yedek kopya daha sonra, işlem günlüğü bilgileri kullanılarak sonraki tüm işlemleri geri yüklemek için ileri alınır. Veritabanının kurtarılması gerekiyorsa ancak veritabanının işlenen bölümü hala kullanılabilir durumdaysa, kurtarma işlemi, gerçekleştirilmemiş tüm işlemleri "geri almak" için işlem günlüğünü kullanır (bkz. Bölüm 10, İşlem Yönetimi ve Eşzamanlılık Denetimi).

Bu aşamanın sonunda veritabanı, sistemin operasyonel aşamaya girmeye hazır olduğu onaylanana kadar devam eden yinelemeli bir test, değerlendirme ve değiştirme sürecini tamamlar.

[6]For a list of the most recent vulnerabilities, visit https://nvd.nist.gov/ and search for "Oracle database vulnerabilities."
[7]"Report: DOJ Sees Bangladesh Heist Tie to North Korea," Mathew J. Schwartz, https://www.wired.com/2016/05/insane-81m-bangladesh-bank-heist-heres-know/.

9-3e İşlem

Veritabanı değerlendirme ve test aşamasını geçtikten sonra çalışır durumda olarak kabul edilir. Bu noktada veri tabanı, yönetimi, kullanıcıları ve uygulama programları tam bir bilgi sistemi oluşturmaktadır.

Operasyonel aşamanın başlangıcı her zaman sistem evrimi sürecini başlatır. Hedeflenen son kullanıcıların tamamı operasyon aşamasına girer girmez, test aşamasında öngörülemeyen sorunlar su yüzüne çıkmaya başlar. Sorunlardan bazıları acil "yama işi" gerektirecek kadar ciddiyken, diğerleri sadece küçük rahatsızlıklardır. Örneğin, veritabanı tasarımı web ile arayüz oluşturmak için uygulanırsa, işlemlerin hacmi iyi tasarlanmış bir sistemin bile tıkanmasına neden olabilir. Bu durumda tasarımcılar darboğazın kaynağını tespit etmek ve alternatif çözümler üretmek zorundadır. Bu çözümler, işlemleri birden fazla bilgisayar arasında dağıtmak için yük dengeleme yazılımının kullanılmasını, DBMS için kullanılabilir önbelleğin artırılmasını vb. içerebilir. Operasyonel bir veritabanı sisteminin bir diğer kritik yönü de uyumluluktur. Veritabanı ve verileri, veri gizliliği, mahremiyet, fikri mülkiyet vb. ile ilgili sürekli değişen eyalet ve federal düzenlemelere uygun olmalıdır. Değişim talebi, tasarımcının sürekli endişesidir ve bu da 6. aşamaya, bakıma ve evrime yol açar.

9-3f Bakım ve Evrim

Veritabanı yöneticisi, veritabanı içinde rutin bakım faaliyetlerini gerçekleştirmek için hazırlıklı olmalıdır. Gerekli periyodik bakım faaliyetlerinden bazıları şunlardır:

- Önleyici bakım (yedekleme)
- Düzeltici bakım (kurtarma)
- Uyarlamalı bakım (performansı artırma, varlıklar ve öznitelikler ekleme vb.)
- Yeni ve eski kullanıcılar için erişim izinlerinin atanması ve bakımı
- Sistem denetimlerinin verimliliğini ve kullanışlılığını artırmak ve sistem performansını izlemek için veritabanı erişim istatistiklerinin oluşturulması
- Sistem tarafından oluşturulan istatistiklere dayalı periyodik güvenlik denetimleri
- Dahili faturalandırma veya bütçeleme amaçları için aylık, üç aylık veya yıllık sistem kullanım özetleri

Yeni bilgi gereksinimlerinin olasılığı ve ek raporlar ve yeni sorgu biçimleri talebi, uygulama değişiklikleri ve veritabanı bileşenlerinde ve içeriğinde olası küçük değişiklikler gerektirir. Bu değişiklikler yalnızca veritabanı tasarımı esnek olduğunda ve tüm belgeler güncellendiğinde ve çevrimiçi olduğunda kolayca uygulanabilir. Sonunda, en iyi tasarlanmış veritabanı ortamı bile artık bu tür evrimsel değişiklikleri içeremeyecek ve ardından tüm DBLC süreci yeniden başlayacaktır.

Gördüğünüz gibi, DBLC'de açıklanan faaliyetlerin çoğu SDLC'dekilere benzer. Bu şaşırtıcı olmamalıdır çünkü SDLC, DBLC faaliyetlerinin gerçekleştiği çerçevedir. SDLC ve DBLC içinde meydana gelen paralel faaliyetlerin bir özeti Şekil 9.8'de gösterilmektedir.

Kavramsal Tasarım
Gerçek dünyadaki nesneleri olabildiğince gerçekçi bir şekilde temsil eden bir veritabanı yapısı modeli oluşturmak için veri modelleme tekniklerini kullanan bir süreç. Tasarım hem yazılımdan hem de donanımdan bağımsızdır.