T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ TABANI SİSTEMLERİNDE YÜKSEK PERFORMANS VE ERİŞİLEBİLİRLİK İLE GÜVENLİĞİN ORACLE VERİ TABANI SİSTEMİ ÖZELİNDE İNCELENMESİ

YÜKSEK LİSANS TEZİ

Ahmet YORULMAZ

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Prof. Dr. Nejat YUMUŞAK**

Haziran 2018

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ TABANI SİSTEMLERİNDE YÜKSEK PERFORMANS VE ERİŞİLEBİLİRLİK İLE GÜVENLİĞİN ORACLE VERİ TABANI SİSTEMİ ÖZELİNDE İNCELENMESİ

YÜKSEK LİSANS TEZİ

Ahmet YORULMAZ

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 01.06.2018 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

Prof. Dr.	**Prof. Dr.**	**Dr. Öğr. Üyesi**
Nejat YUMUŞAK	**Celal ÇEKEN**	**Amani YAHYAOUI**
Jüri Başkanı	**Üye**	**Üye**

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Ahmet YORULMAZ

22.05.2018

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Prof.Dr. Nejat YUMUŞAK'a şükranlarımı sunarım.

Veri Tabanı Yönetmeni olarak çalıştığım Türkiye Kalkınma Bankası Bilgi İşlem Dairesinde bulunan test veri tabanı sistemlerinde tezim için gerek duyduğum uygulamalar için alt yapılarını kullanma imkânını veren birimimdeki tüm yöneticilere teşekkürlerimi sunarım.

Bu çalışmayı yaparken öncelikle annem ve babama ayrıca sabırlarından dolayı eşim Neslihan YORULMAZ ve çocuklarım Yusuf ve Selim'e sevgilerimi sunuyorum.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

ACID	: Atomictiy, Consistency, Isolation, Durability
AGREE	: Advisory Group on Reliability of Electronic Equipment
ADDM	: Automatic Database Diagnostic Monitor
ARCn	: Archiver Process
ASCII	: American Standart Code for Information Interchange
BCNF	: Boyce Codd Normal Form
CKPT	: Checkpoint
DBA	: Database Administrator
DBWn	: Database Writer Process
DoS	: Denial of Service
FK	: Foreign Key
ITIL	: Information Technologies Infrastructure Library
IPS	: Intrusion Prevention System
LGWr	: Log Writer Process
MTTF	: Mean Time To Failure
MTTF	: Mean Time To Repair
VPD	: Virtual Private Database
PK	: Primary Key
PGA	: Program Global Area
RAM	: Realibility Availability Maintainability
RAID	: Redundant Array of Inexpensive Disk
RMAN	: Recovery Manager
SSO	: Single Sign On
SQL	: Structure Query Language
VTYS	: Veri Tabanı Yönetim Sistemi

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: veri tabanlarında yüksek performans, veri tabanlarında yüksek erişilebilirlik, veri tabanlarında yüksek güvenlik

Endüstri 4.0 ile verinin toplanması ve analiz edilmesi çok daha önem kazanmıştır. Ticari veya kamu kuruluşları artık birbiri ile dijital ortamlarda sürekli konuşur hale gelmiştir. Bu tarz haberleşmek için iki tarafında sistemsel alt yapısı performanslı, erişilebilir ve güvenli olmak zorundadır. Sistemsel alt yapıların omurgasında hangi teknolojik yenilik olursa olsun bu sistemlerin merkezinde veri tabanları durmaya devam edecektir. Bu da veri tabanı yönetim sistemlerin kritik sistemlerin alt yapısı için en temel sistem bileşenlerden biri olduğunu göstermektedir.

Bu çalışma, kurumların en değerli varlıkları arasında olan veri üzerine metodolojik bir araştırmadır. Endüstrideki veri çözümleri, veri tabanı yönetim standartları ışığında karşılaştırmalı olarak incelenmiştir. Veri tabanı endüstrisinin en önemli markalarından olan Oracle'ın özellikle veri tabanı güvenliğine ilişkin yaklaşımları analiz edilmiş; veriler üzerinde yetkilendirilme, gerekli durumlarda karartma, maskeleme ve denetim gibi işlemlerin hangi yöntemlerle yapıldığı uygulamalı olarak gösterilmiştir.

HIGH PERFORMANCE AND AVAILABILITY IN DATABASE SYSTEMS AND SECURITY OF ORACLE DATABASE SYSTEM

SUMMARY

Keywords: high performance in the database, high availability in the database, high security in the database

Collecting and analyzing data by industry 4.0 is more important than ever. The commercial or public organizations are now able to communicate with each other in a digital environment. In that communication, the words constituting data almost wholly represent the words that vocalize information. The consequences of the exchange of information on the other hand directly have an impact on the decision support systems of the organizations. To sustain such a communication, both sides must have infrastructures that are with high performance, availability, and security.

This study is a methodological examination on ‘data’, which is one of the most valuable assets of an organization. Industrial solutions on data are comperatively analyzed under the light of database management standards. One of the most important organizations in the database industry, Oracle's approach to database security has been analyzed and the methods such as authorization, auditing and masking where necessary, have been shown in practice.

BÖLÜM 1. GİRİŞ

1.1. Motivasyon

İnsan emeğinin, insana has aklın değerli çıkarımları sonucunda üretilmesi, bilgiyi önemli kılmıştır. Üretilen bilgiyi yönetmek; pahalı ve değerli olan bilginin korunabilmesi, erişilebilmesi ve paylaşılması için gerekli olduğu anlaşılmıştır. Bilginin katma değer üretebilmesi için ihtiyaç hissedildiği an, en kısa yoldan, en hızlı şekilde doğru yere, doğru kişilere eriştirilmesi gerekir.

Günümüzde geliştirilen her ne uygulama olursa olsun barındıracağı bir veri tabanı olacaktır. Veri tabanı, günümüzde veri ömrünün olabildiğince uzun tutulmaya çalışıldığı, erişimi belli kurallarla sağlandığı, veri yaşam döngüsü içinde geriye dönüş imkânın olduğu, raporlama, izleme gibi teknikleri barındırdığı veri yönetim sistemidir.

Günümüzdeki gelişmiş veya gelişmekte olan tüm ülkeler her türlü süreçleri kullanarak güçlü veri tabanlarına sahip olmak istemektedir. İstenen güçlü veri tabanları ile karar destek sistemlerini daha sağlıklı hale getirmeyi hedeflemektedir. Ancak kontrol edemedikleri devasa sistemlerin devasa çöplüklere dönüşmesi de mümkün olabilmektedir. İşte bu noktada bilimsel olarak her ne amaçla olursa olsun üretilen verilerin hedefe uygun kaydedilmesi ve istendiğinde sunumu yapılabilmesi için alt yapısı her zaman ve her noktada yeterli olması gerekir.

Yapılan araştırmalarda, veri tabanı yönetiminde zorlukların olduğu ve gerekli olan farkındalığın yeterli olmadığı görülmektedir.

Bağımsız Oracle Kullanıcı Grubu (IOUG) 2015 tarihli bir çalışmasında veri tabanı yönetiminde yaşanan zorlukları şu şekilde listelemektedir:
Veri tabanı yöneticilerin;

a. Yüzde elli ikisi (52%) veri tabanı performans problemini hızlı teşhis edemiyor.
b. Yüzde kırk beşi (45%) uygulamaların veri tabanına uyarlamasını tam yapamıyor (SQL).
c. Yüzde otuz yedisi (37%) SQL ayarlamalarının kontrolünü ve uygulanmasını istenen seviyede yapamıyor.

Bir başka çalışmada ise veri tabanlarındaki yapılandırma değişikliklerin yüzde doksanı (90%) yeterli test edilmediği için beklenmedik şekilde sistemlerin durmasına neden olmuştur. Ayrıca veri tabanı yönetmenlerin yüzde ellisinden fazlası (50%+) yapılması gereken yapılandırma değişikliğinin negatif etkisi olma ihtimalinden çekindiği için yapamadıkları tespit edilmiştir [1]. Bu durum veri tabanı yönetiminin yüksek performanslı, 7X24 erişilebilir, yönetebileceği güvenli sistemi kurmasına engel olmaktadır.

Verilerden veri sözlüğüne, güvenliğinden yedeklenmesine, felaket durumundan geri kurtarılmasına kadar türlü fonksiyonları barındıran veri tabanı yönetim sistemini ülkemizin tüm kurumsal yapılarında standartlara uygun hale getirilmesi, ayrıca öngörülmeyen durumlar için proaktif çözümlerle önlenmesi gerekmektedir.

Veri tabanı yönetmenlerin sorumluluğundaki veri tabanı güvenliği, büyüyen verilerle ve gelişen teknolojik yeniliklerle daha da kritik hale gelmiştir.

Gartner Pazar araştırmacılarından Kish ve Lowans [2] veri tabanların bulut teknolojilerine kaydığını ve artık veri tabanların logların izlenmesinin, verilerin redüksiyon yapılmasının bilim dünyasında popüler olduğunu belirtmiştir. Hassas verilerin korunumu için literatürde [3] geçen üçayak üstünde duran veri tabanı güvenliği için olmazsa olmazı şunlardır:

a. Gizlilik (confidentiality)
b. Bütünlük (integrity)
c. Erişilebilirlik (availability)

Raydel Montesino, [4] kontrollü sistemlerin otomatik hale getirdiği yapılarda, güvenlik çok daha kritik hale geldiğini öne sürmüştür. Bu bağlamda veri tabanı yapılarını yöneten otomatik yapılardaki güvenlik açıkların ya da yazılımsal hataların (bug) proaktif çözümlerle giderilmesi elzemdir.

1.2. Tezin Hedefi

Kurumların veri tabanı yönetmenleri, değerli verilerin saklandığı veri tabanı sistemlerinin mimarilerini oluştururlar. Bu mimariler için kullandıkları metodolojiler, bilimsel tecrübelerin ışığında olmalıdır.

Hem proaktif hem de reaktif olarak yüksek seviyede performans, erişilebilirlik ve güvenliği sağlayacak metodolojilerin neler olduğu tezde değerlendirilip farkındalığın arttırılması hedeflenmektedir.

1.3. Tezin Kapsamı

Tezin kapsamı olarak ikinci bölümde veri tabanı dünyasında bilimsel kaynak taramasının sonuçları paylaşılmıştır. Veri tabanı dünyasının kullandığı terimlerin detaylı açıklamaları yapılmıştır.

Üçüncü bölümde, yüksek performanslı, erişilebilir ve güvenli veri tabanı sisteminin olması için gerekenler bilimsel kaynaklardan faydalanarak metodolojik olarak sunulmuştur.

Dördüncü bölümde ise veri tabanı yönetimde güvenlik işlemlerin Oracle veri tabanı yönetim sistemi özelinde incelenmiştir.

Sonuç bölümde ise tezin sonunda varılan değerlendirmeler ve öneriler paylaşılmıştır.

1.4. Yapılan Çalışmalar

Veri tabanında performansı, erişilebilirliği yükseltmek için gereken ilk adım performans metriklerin tespiti, ardından o metriklerin takip edilmesidir. Metriklerin istediğimiz eşik seviyede (treshold) olmadığı teşhis edildiği anda yapılması gerekenlerin ne olduğunu belirlenmesi gerekmektedir. Bu noktada, tezde yönlendirici uygulamalar yapılmıştır.

Oracle veri tabanı yönetim sistemi özelinde, hangi metodolojilerle güvenliğin sağlanabileceği, uygulamalı olarak tezde gösterilmiştir.

Bu sayede veri tabanında yüksek performans, erişilebilirlik ve güvenlik için gereken standart yöntemlerin neler olabileceği uygulamalı olarak sunulmuştur.

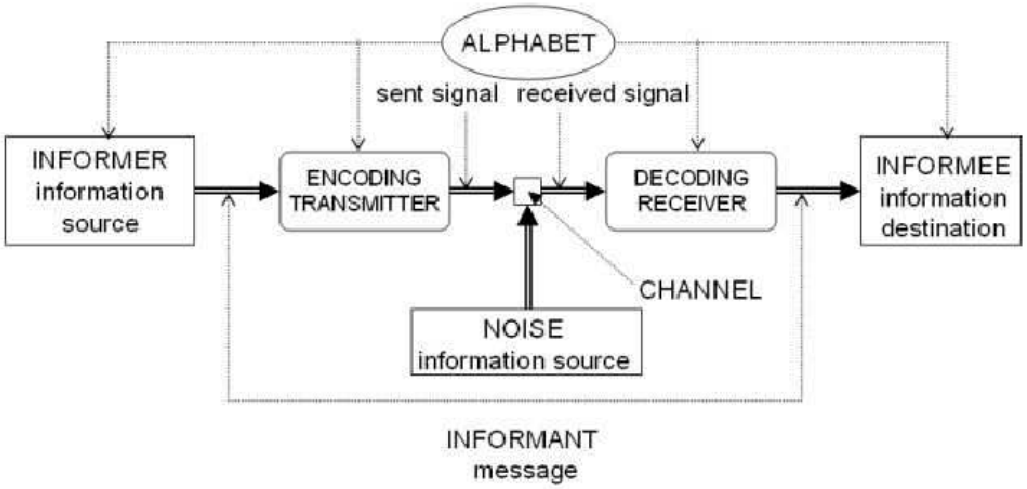
BÖLÜM 2 TEMEL KAVRAMLAR

2.1. Veri Tabanı Alanındaki Bilimsel Kavramlar

2.1.1. Mesajlaşmadan veriye

Mesajdan bilgiye geçişte alfabenin ve bilginin rolünü gösteren Şekil 2.1. incelendiğinde iletişimin temel unsurları, kaynak, hedef ve kanal olduğu görülmektedir. Bu unsurların tümünde alfabe ve bilgi mesajları rol oynadığı anlaşılmaktadır.

İnsanoğlu alfabenin buluşuna kadar sembollerle mesajlaşarak iletişim kurmuştur. Alfabenin buluşu ile mesajlaşmalar hızlanmış ve böylelikle mesajların bir kaynakta tutulmasının ihtiyacını hissetmiştir. Tecrübeleri, önem verdikleri değerleri saklamak isteyen insanoğlu, geçen süreçte bilgi içeren mesajların ilerde kullanması düşüncesiyle mesajları tutan sistemler oluşturmuştur.

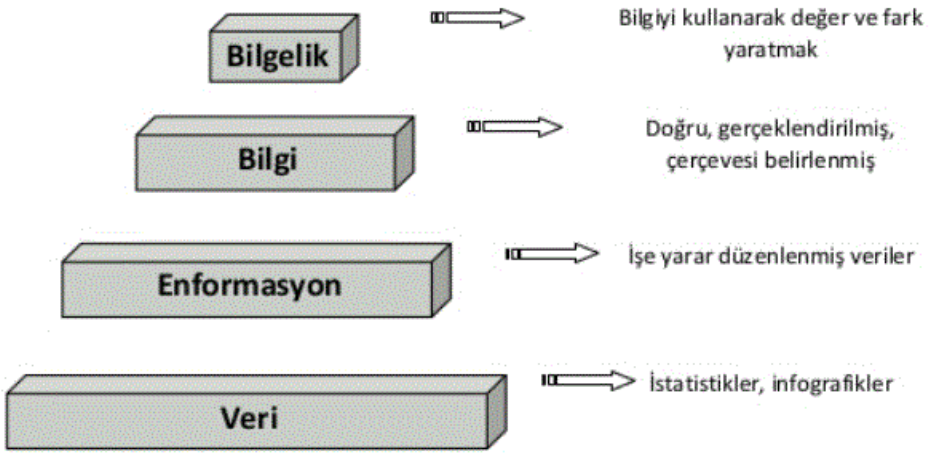


Şekil 2.1. Shannon ve Weaverin iletişim modeli (Stanfort,2015).

2.1.2. Veriden bilgeliğe

Veri, bir durumun birbiriyle bağlantısı henüz kurulmamış bilinenlerle, dijital ortamlarda bulunan ve sinyal bilgileri taşıyan veya bit dizeleri olarak kodlanabilen yapı şeklinde tanımlanabilir[5]. Veriden bilgiye ulaşılan bir hiyerarşik olgunun var olduğu genel kanı olarak bilinmesine karşın Shannon bilgi üzerinde tek bir tanımın olmasının zor olduğunu belirtmiştir [6].

Veri üzerinden bilgiye geçiş verinin değer kazanmasıyla olur.



Şekil 2.2. Veriden bilgeliğe (Kocak,2013).

04712xxx05001 şeklinde elimizde veri olsun. İlk olarak bu rakamlar kümesinin bir anlam ifade etmediği görülür. 0471 - 2xx0 - 5001. 471 şube kodu, 2xx0 hesap numarasını 5001 ise hesap türünü ifade edebileceğini önceki bilgilerden yararlanarak sınıflandırılır. Bu enformasyon sonuncunda 04712xx05001 numarası ile müşteri tablosu ile ilişkilendirilir. İlgili müşterinin kredi istihbarat sonuçlarına göre kredi limitini 15000 TL çıkarılabileceğine karar verilebilmesi sağlanır. Böylelikle 13 haneli bir numaradan kredi limitini arttırma imkânı sağlanmış oldu. Böylelikle veriden bilgiye, bilgiden de karar vermeye bir dönüşüm oldu.

Bilgiye dönüşen verinin değerine karar verebilme yani tanımlanan değerli veri mi yoksa değersiz veri mi olduğunun belirleyebilme en az verinin üzerinde taşıdığı olgu

kadar önemlidir [7]. Günümüzde teknik yeterlilik ve kapasite artık eskiye nazaran çok yüksektir. Bundan dolayı yeni sistemler verinin her şeyini depolayabilir. Ancak son durumda bilgiye dönüşecek veri üzerinde kurulacak doğru modelin bilgi maliyetini en uygun noktaya getirilmesi gerektiği düşünülmelidir.

2.1.3. Varlık ve ilişki

VTYS, bağımsız olarak veriyi çözümlerken varlık – ilişki modelini son yıllarda yoğun kullandığı görülmektedir. Varlık, varlığı aşikâr olan ve benzerlerinden ayrılabilen her nesneye denir. Varlığın nitelikleri vardır. Bu niteliklerin olabileceği değerler kümesine etki alanı (domain) [8] denir. Aralarında ilişki kurulan varlıklardan her birinin ilişkideki işlevine rol denir. Rol; aslında varlıkların etki alanı içinde hareket alanının boyutunu belirler. Sınırları çizilmiş alanda hareketi sağlayan veri tabanı rolü güvenlik anlamında kritik değer taşımaktadır. Rol, ilişkinin ne taraf olduğunu belirler. İlişkiler varlık arasında olduğu gibi varlık kümesi arasında da olabilir. Bu ilişkiler, birden-bire (one to one), birden–çoğa (one to many), çoktan-bire (many to one), çoktan-çoğa (many to many) kurulur. Bu ilişkiler için matematiksel ifade [9] yazılabilir.

Bir varlık kümesi içindeki varlıkları ya da bir ilişki kümesi içindeki ilişkileri birbirinden ayırt etmek için kullanılan nitelik ya da nitelik grubuna bu varlık ya da ilişki kümesinin anahtarı denir. Anahtar kavramının hem varlık kümeleri hem de ilişki kümeleri için kullanabiliriz.

Süper Anahtar: İlişkinin olmazsa olmazı olan süper anahtar, değeri ile bir kümedeki varlıkları ayırt etmeyi sağlayan niteliktir. Buna varlık/ilişki kümesinin süper anahtarı denir. Ayırt etme özelliğine sahip olmak için gereğinden fazla nitelik içerebilir.

Aday Anahtar: Süper anahtar olmaya namzet ancak bir alt kümede ise yani bir varlık / ilişki kümesinin süper anahtarının bir alt kümesi de bu varlık/ilişki kümesini ayırt edebiliyorsa, bu alt kümeye aday anahtar denir.

Her varlık kümesi için bir anahtar bulmak mümkün olmayabilir. Bu durumda varlık kümesinin nitelikleri genişletilir. Genişletilmesine rağmen anahtar bulunamaz ise zayıf varlık kümesi olduğu anlaşılır. Şayet varlık kümesinin en az biriyle anahtar oluşturabilirsek seçilen varlık kümesi güçlü varlık kümesidir. Bu şu anlama gelir: Zayıf varlık kümesi güçlendirmenin yolu güçlü bir varlık kümesi arasında birden-bire ya da güçlüden zayıfa birden çoğa bir ilişki oluşturulabilmelidir. Ayrıca zayıf varlıklar için bu ilişkinin var olma bağımlılığı oluşturulmalıdır. İlişki kurabilme anlamında zayıf varlık kümesinin nitelikleri arasından, aynı güçlü varlığa yani süper anahtara sahip, bağımlı zayıf varlıkları birbirinden ayırt etmeyi sağlayacak bir nitelik grubu (discriminator) oluşturulabilmelidir. Bu şu şekilde elde edilir:

Zayıf bir varlığın anahtarına, bağlı olduğu üstün varlığın anahtarına ayırıcı nitelikler ilave edilerek yapılır.

Bir varlık kümesi üzerinde anahtar kavramını inceleyelim. Dünya bankasından kredi talep eden müşterilerin oluşturduğu varlık kümesini niteliklerini belirleme aşamasında etki alanları tanımladıktan sonra varlık kümesi için anahtarı şu şekilde tespit edilmiştir:

Şekil 2.3. DB_KREDI_TALEP varlığın domaini

Dünya Bankası Kredisi için talep eden firmaları ayırt edebileceğimiz anahtar FIRMA__KOD oluşturuldu.

FIRMA__KOD ‘a sahip firma birden fazla talep de bulunabilir. Öyleyse aynı firmanın taleplerini ayırt edebilmek için TALEP__KOD oluşturulur.

Talepleri ayırt edebilmesine rağmen yine varlık kümesinin zayıf yanı var. Çünkü kaynaklar birden fazla olabilir. Kaynakları da ayırt edebilmek için KAYNAK__KOD oluşturulur.

Bir firmanın birden fazla talebi olabileceği gibi birden fazla kaynağı da kullanabilir. Bunların hepsini aynı firma birden fazla proje ile üretebilir. Öyle ise bir firmanın projelerini ayırt edebilmek için PROJE__NO üretilmelidir.

Her projenin bir hak edişi olacağında hak edişleri de ayırt edilmesi gerekir. HAKEDIS__NO ile firmanın ayırt edilebilen hak edişi bilgisine erişilebilir.

Analiz sonucunda bir firmanın hangi talebi hangi kaynakla, tanımlanan hangi proje ile hak edişini alabileceğin ayırt edebileceğimiz aday anahtarlarla FIRMA_KOD, TALEP_KOD, KAYNAK_KOD, PROJE_KOD, HAKEDIS_KOD birlikte süper anahtar oluşturularak güçlü varlık kümesi oluşturabileceğimiz yapı hazırlanmış oldu. Yukardaki analiz veri tabanın her varlık kümesi için yapılmalıdır.

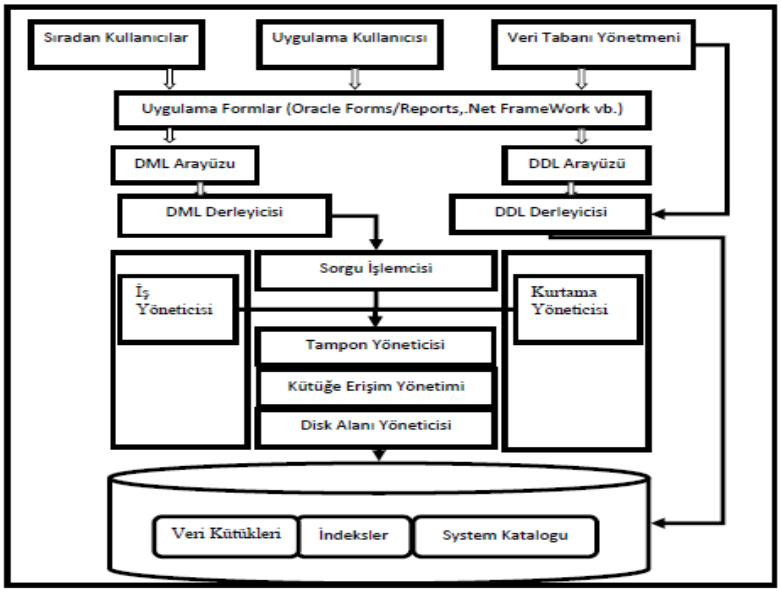
Modelleme yapılırken kullanılan varlık kümesi, ilişki ve nitelik arasındaki ayırım net çizgilerle yapılamaz. Varlık kümesi olarak tanımladığımız olguyu ilişki kümesi olarak da tanımlandığı görebiliriz. Bu ayrımı çizmek tamamen tasarımcının bakış açısı ile ilgilidir. Bakış açısı sübjektiftir. Yukarıda da değindiğimiz gibi bu sübjektiflik kendi içinde tutarlılığı çok önemlidir. Pratik olarak şöyle örnek verilebilir. Kişi varlık kümesinde telefon numarasını barındıran bir yapımız olsun. Telefon numarasına yaklaşım çok değişik tasarımlarda farklı tanımlanabilir. Kişi varlık kümesinden telefonun bağımsız olarak var olamayacağını, bir kişinin sadece bir telefon numarası bulunabileceğini, birden çok kişinin telefon numarasının aynı olabileceği varsayımlarını tasarımcı dikkate alır. Gereken değerlendirmeden sonra bir ER diyagramı tasarlar. ER diyagramından tasarımcı tersine çözümleyerek şu sonuca varır: Kişi varlık kümesi ile telefon numarası varlık kümesi arasında sahiplik ilişki vardır.

Tasarımcı telefon numarasına ilişki kümesi değil varlık kümesi olarak bakmaktadır. Bankacılık alanından bir örnek ile konuyu daha da somutlaştırılmalıdır. Tasarlarken tasarımcı banka şubesi varlık kümesini, müşteri varlık kümesinin tanımlamakta zorlanmaz. Lakin banka hesabı için aynı net durum olmayabilir. Tasarımcının farklı bakış açısı olabilir. Banka hesabını varlık kümesi olarak düşünebileceği gibi ilişki nedeni de düşünebilir. Bu tamamen tasarımcının kararıdır.

2.1.4. Veri tabanı

Veri tabanı belirli tarzda organize edilmiş veri koleksiyonudur. Verileri bir arada tutan yapıdır. Bu yapı, yüklenen veri miktarı ne kadar büyük olsa da veya işlem yapan kullanıcı ne kadar fazla olsa da, veri operasyonlarına güvenilir ve uygun sürede cevap vermesi öncelikli görevidir. Aynı veriyi aynı anda farklı kullanıcıların kullanabilmesi, farklı uygulama programların istemesi, veri üzerinde operasyonların tutarlı, veriye etkisinin bütünlük içinde olması VTYS için gerekli şartlardır.

Veri tabanı birbiriyle ilişkileri olan verilerin mantıksal ve fiziksel olarak yöneten bir depolama sistemidir. Tablo, kolon, kayıt, indeks, tetikleyici gibi bileşenlere sahip veri tabanı yönetim sistemi Şekil 2.4.'de özet olarak ifade edilmiştir.



Şekil 2.4. Veri tabanı yönetim bileşenleri (Connolly Begg,2015)

2.1.5. Veri tanımlama dili

Veri modelini tanımlamak için kullandığımız dile Veri Tanımlama Dili ya da Veri Tanımlama Olanağı denir. Bu dili çözen derleyicide veri tanımlama dili derleyicisidir. Bu tür tanımların bulunduğu yere veri sözlüğü [10] denir. Veri hakkında veri üretirken Veri Tanımla Dilinin derleyicisi kullanılır ve çözümlenir. Çözümleme aşamasında diğer yüksek seviyeli diller gibi söz dizimi denetlenir (syntax). Veri tabanı yöneticisinin tanımladığı veri tanımlama nesnesi üretilir. Bu nesne üzerinde sadece veri tabanı yöneticisi veya onun atadığı veri tabanı kullanıcısının operasyon yetkisi vardır.

Veri tanımını oluşturulan özellikler şunlardır:

a. Verinin Türü
b. Verinin Uzunluğu
c. Verinin varsa varsayılan değeri
d. İndeks
e. Log verisi
f. Veri üzerinde yetkiler
g. Veri üzerindeki kısıtlamalar
h. Verinin fiziksel yeri
i. Parametre tanımları

Yukardaki liste VTYS ‘nin kabiliyetine göre genişletilebilir. Literatürde veri sözlüğü olarak (data dictionary) tanımlanan veri tabanı tanımları, fiziksel ortamda saklanırken veri erişim istatistik bilgilerin tutulduğu arşiv bilgilerini de barındırır.

2.1.6. Veriyi sorgulama dili

Veri tabanı uygulamalarının kullandığı sorgu dilidir. Veri tabanıyla iletişime geçme protokolü denilebilir. Bu dilde uç kullanıcı veri tabanından ne istediğini belirtir. Bu isteği işletebilmek için sorgu İşleyici bileşeni kullanılır.

2.1.7. Veri tabanında atomik bilgi hareketi

Veri tabanına erişebilen her oturum, yapmak istediği her türlü manipülasyon işlemleri sırasında (ekleme, silme, güncelleme) kesintiye uğramamalı veya zorunluluk nedeniyle uğrasa dahi hareket başlangıcına dönebilme olanağına sahip olmalıdır. Bu hareketin minimum komut kümesine atomik bilgi hareketi (transaction) denir.

2.1.8. Veri tabanında tutarlılık

Veri tabanında tutulan her varlığın verisi kendi içinde çelişkisiz olma zorunluluğu vardır. Buna veri tabanında tutarlılık (consistency) denir.

2.1.9. Veri tabanında işlem yalıtımı

Veri tabanında birden fazla oturum açılabilme yani erişimleri çok yerden sağlayabilme özelliği bulunmalıdır. Bu bağlamda açılan her oturum birbirinden ayrık olmalıdır. Aralarında işlemsel yalıtımı olmalıdır. Bu özelliğine veri tabanında işlem yalıtımı (isolation) denir.

2.1.10. Veri tabanında işlem dayanıklılığı

Veri tabanında işlemler sürerken dış etkenlere karşı kendini koruyabilecek olmalıdır. Bu özelliğine veri tabanında işlem dayanıklılığı (durability) denir.

Bir veri tabanı yönetim sisteminin hareket yönetici bileşeni; işlemlerde tutarlılığı, yalıtımı ve dayanıklılığı sağlamalıdır [11] .

2.2. Veri Tabanının Modellenmesi

1960'li yıllarda veri, sıradüzensel (hiyerarşik) model ile modellenirdi. Veri yapısı hiyerarşik tutulurdu. Ağaç modelini andıran bu yöntem ile bir kaydın yalnız bir ebeveyni çocukları olurdu. 1970'lerde ise bir kaydın yalnız bir ebeveyn kaydının

olması şart olmadığı daha fazla ebeveyn kaydı olabileceği düşünüldü. Buna ağ modeli denildi. 70'li yılların sonuna doğru ilişkisel modelin çok daha uygun olduğu anlaşıldı. Veri tablolar üzerinde tutulabileceği ve tablolar arasında istenirse ilişki kurulabileceği görüldü. Daha sonraki yıllarda ise veriyi nesne olarak düşünülebileceği, nesnenin programatik özelliği olan sınıf ile modellenebileceği ortaya konuldu. Veri tabanı ile yazılımlar arsında gelişen tekniklerle çok hızlı veri tabanı uygulamaları (yazılımları) geliştirilmeye başlandı. Günümüzde geçmişe nazaran daha hızlı yazılımlar geliştirilmesinin bir sebebi de veri modellemenin yazılıma yakın olmasıdır. Nesneyle modelleme, verinin bakımının kolaylaştırmıştır. Veri tabanında güvenlik ilkelerini uygularken kullanıcı oluşturma, yetkilendirme ve kullanıcıların işlemlerinin takibini aslında nesneyle modellenmiş veri tabanı yönetim yazılımının kullanıcı sınıfı seviyesinde yaptığı OOP teknikleriyle gerçekleştiği görülmektedir. Bu teknikle kullanıcıyı yönetim işlemini, gerçek hayatın tanımlarını VTYS'ne indirgemiş olduğunu görebiliriz.

İlişkisel veri tabanlarının teorisi 1970 yıllarına dayanmaktadır. IBM'de matematikçi olarak çalışan, Edward Frank Codd'un araştırma yazısında tespit ettiği bilimsel bulgularla ilişkisel veri tabanı modelini ortaya çıkardı. Hiyerarşik ve ağ yapılı veri tabanı modellerinde geliştiriciler, veri tabanı yapısını önceden bilmesi gerekiyordu ve bu da bazı zorlukları getiriyordu. İlişkisel veri tabanı veri tekrarına çözüm getirmek ve fiziksel katmandaki yapının uygulama üzerindeki bağımlılığı ortadan kaldırmanın mümkün olabileceğini ortaya koydu. Matematikçi Codd, küme ve önerme teorisini veri tabanı objelerine uygulanabilirliğini ilgili yazısında göstermiştir [12] .

İlişkisel veri tabanı içinde depolanan veriye çeşitli kısıtlamalar getirilebilir. Bu kısıtlamalar, etki alanı, anahtar, varlık bütünlüğü ve referans bütünlüğüdür.

2.2.1. İlişkisel modellen veri tabanı yönetim sistemi

İlişkisel veri tabanı, bilimsel olarak matematik kuramlarına dayanır. Veri yapısı her ne kadar farklı olsa da ilişki doğru kurulursa, tutarlı sınırlamalar belirlenirse her türlü veri tabanı uygulamalarına ilişkili veri tabanı yönetim sistemi uyum sağlayabilmektedir.

Tablolar, anahtarlar, ilişkiler, indeksler İVTYS'nin temel öğeleridir. İlişkisel Veri tabanı içinde verilerin depolandığı birimler tablolardır. Tabloları birbirine bağlayıp, veri tekrarını azaltmak ve böylece, iyi bir depolama ortamı oluşturmak için birçok tabloda anahtar kullanılır. Veri aramak için indekslerden yararlanılır [13].

2.2.2. İlişkisel olmadan modellen veri tabanları (NoSQL)

1998 yılında Carlo Strozzi NoSQL kavramını ortaya attı. SQL olmayan demek gibi görünse de, aslında o anlamda kullanmamıştır. Bu kavramı "not only sql" olarak ve ilişkisel olmayan veri tabanı olarak veri tabanı dünyasına kazandırılmıştır [14]. Kavram iddialıdır ve ilişkisel veri tabanı sistemlerine farklı alanlarda rekabet edebilecek çözüm olarak kabul görülmüştür. Ancak ilişkisel veri tabanı yönetimin bazı avantajlarını henüz içermemektedir.

NoSQL veri tabanlarının genel özelliklerine bakıldığında [15] dört tip veri tabanı bulunmaktadır:

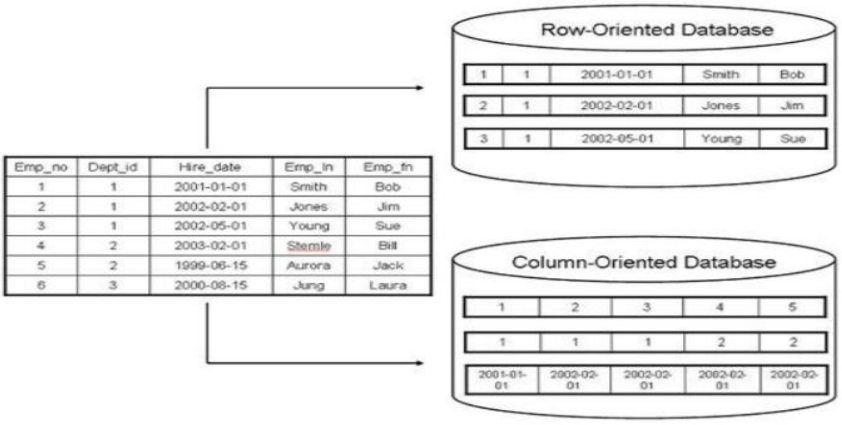
a. Kolon Tipli Veri tabanları
b. Doküman Tipli Veri tabanları
c. Anahtar Değerli Veri Tabanları
d. Graf Tipli Veri Tabanları

Kolon tipli veri tabanları Şekil 2.5.'de görüldüğü gibi veriyi sütun şeklinde tutar. Veriler kolonlar üzerinde dağılır. Bu da birçok sunucuda büyük verilerin sorgulanabilme imkânı tanır. Arama motorları için ideal modeldir.

Doküman Tipli veri tabanları ise veriyi anahtar-değer çiftlerinin toplanmasından oluşan versiyonlanarak tutulduğu dokümanların veri tabanlarıdır.

Anahtar Değerli veri tabanları, sağlama (hash) tablolarına sahiptir. Bu tablolarla (hash) ilgili değerlere ulaşılabilmektedir. Log kayıtları için ideal modeldir.

Graf Tipli veri tabanları graf teorisini temel alan modeldir. Düğümler arası ilişkiler veri tabanında saklanır. Forum uygulamaları graf tipli veri tabanlarda saklanabilir [16].



Şekil 2.5. Satır ve kolon bazlı olarak verinin veri tabanda saklanması (Zemzans,2016)

NoSQL veri tabanları anlık tutarlılığını garanti edemez. Ancak sonunda tutarlı bir raporlama imkânı vereceğini teyit eder. Zaten CAP teorisi olarak bilinen Tutarlılık, Kullanılabilirlik, Parça Toleransı (Consistency, Availability, Partition Tolerans) kavramı; herhangi bir dağıtık sistem aynı anda hem tutarlı hem müsait hem de parçalanma payını aktifleştiremez anlamına gelmektedir. Yani dağıtık sistemin bir parçasına erişen sistemin tümüne aynı anda cevap veremez. Parçalanan verilerden haberdar olması belli bir zaman alacağı aşikârdır.

NoSQL "BASE" (Basically Available- Soft state- Eventually consistent) kısaltması ilişkisel veri tabanların özellikle çevrimiçi hareket atomik bilgi hareket işlemlerine (OLTP) sahip veri tabanlarında kullanılan ACID (Atomocity, Consistency, Isolation, Durability)'in karşılığıdır [17].

2.2.3. İlişkisel ve ilişkisel olmayan veri tabanların karşılaştırılması

İlişkisel veri tabanı ile ilişkisel olmayan veri tabanı sistemini karşılaştırarak iki kavramının daha net anlaşılması sağlanmış olacaktır. Tablo 2.1.'de karşılaştırma yapılmaktadır.

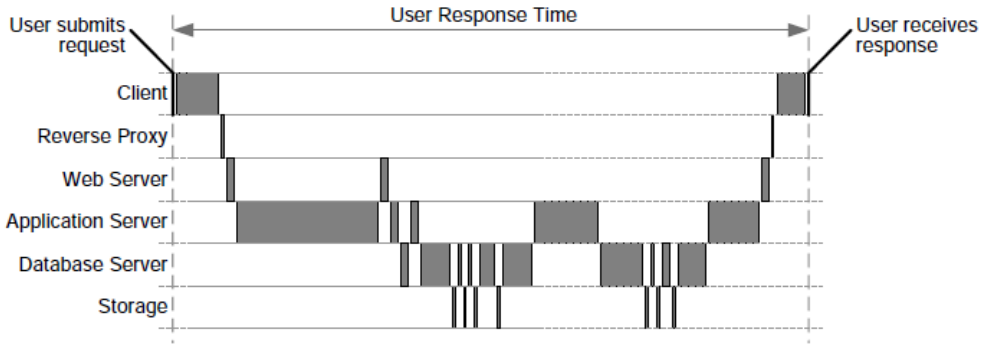
Tablo 2.1. İVYT ile ilişkisel olmayan (NoSQL) veri tabanın karşılaştırılması

İlişkisel Veri Tabanı Sistemi	İlişkisel Olmayan Veri Tabanı Sistemi
Karmaşık verileri yüksek yoğunluklu ve kritik verileri yönetir.	Düşük değerli ve düşük yoğunluklu ve basit verileri yönetir.
Kompleks veri ilişkileri kurabilir.	Çok basit ilişkiler kurar.
Birleşme (join) vardir.	Birleşme (join) yoktur.
Şema ile obje sahipliği vardır.	Şemasız yapısal olmayan veri yönetimi vardır.
İyi tanımlanmış standartlara sahiptir.	Standartları tam belirgin değildir.
Veri Tabanı merkezli yapı kurar.	Uygulama merkezli yapı kurar.

2.3. Veri Tabanı Performans Metrikleri

2.3.1. Performansın temeli

Performans çalışması, endüstride gerçekleştirilen her uygulamanın her aşamasında yapılmalıdır. Şekil 2.6.'de gösterilen bir istemcinin isteği (request) ve ona sunucu tarafından verilen cevap (response) aşamaları incelendiğinde veri tabanı sunucusu alt katmanda görülmektedir. Tezde, bu katmanda yapılması gerekenlere odaklanacaktır.



Şekil 2.6. Uçtan uca kullanıcı isteğine cevap verme süreci

Veri tabanında performansı, sistemin yapılandırma metrikleri ile sorguların metrikleri olarak ikiye ayrılmıştır.

2.3.2. Veri tabanı sisteminin yapılandırma metrikleri (instance tunning)

Performans metriklerin bazı önemli bileşenleri incelendiğinde [18]:

Veri tabanı zamanı (DB Time(s)) : Veri tabanı performansını ölçmek için kullanılan temel parametredir. Veri tabanına erişen toplam kullanıcı işlem parçacığı (user process) kullanıcı isteklerinin işlenmesinde harcadığı birikmiş zamandır. Tüm boş olmayan kullanıcı oturumların bekleme süresini ve CPU zamanını içerir. Bu metrik V$SESS_TIME_MODEL ve V$SYS_TIME_MODEL görünümlerinde takip edilebilir.

Veri tabanı CPU (DB CPU) : Veri tabanı bileşenlerinin işlemcinin(CPU) tükettiği toplam zamanı modeller. Bu değeri işletim sistemi veya veri tabanı çekirdek kodların tükettiği işlemci zamanından bulunur.

Veri tabanı zamanı ile veri tabanının işlemci (CPU) zamanı arasında şu ilişki bulunmaktadır:

DB Time = DB CPU + Boşta Bekleme Süresi (non-idle wait time)

Arka Plan CPU (Background CPU) : İşletim sistemi üzerinden arka plana atılan işlem parçacıkların harcadığı zamandır.

Tekrarlı İşlem Kapasitesi (Redo Size) : Saniyede üretilen tekrarlı işlemlerin oluşturduğu verilerin günlük miktarını ölçer.

Mantıksal okuma, fiziksel okuma ve fiziksel yazma (Logical read block, physical read, physical write) : Bu üç metrik birbiriyle ilişkilidir. Mantıksal veya fizikselden kastedilen aslında okuma veya yazma işlemi bellekten mi yoksa diskten midir bunun tespit edilmesidir.

Blok Değişim (Block Change): Diskte veri bloğun saniyedeki yâda atomik işlem sırasındaki değişim miktarını vermektedir.

Sorguların Parslanması ve Ham Parslanması (Parses ve Hard Parses) : Sorgulama komutların sentakslarının kontrolü ardından mantıksal kontrolün yapıldığı ve o şekilde bellekte parslandığı (soft parses) aşamaya sorguların parslanması denir. Sorgunun ilk halini alan parslanacak şeklinde bellekte duran (hard parses) haline ise ham parslanması denir.

Veri tabanı performans ile ilgili olarak bakılacak diğer metrikler de şunlardır:

a. Veri tabanı bellek bölgesi ikincil bellek(cache) büyüklüğü ve fiziksel disk giriş/çıkış oranı
b. Tablolara erişim oranları
c. İndeks kullanım oranı
d. Disk üzerindeki sıralamalar
e. Verilerin ne oranda zincirlendiği
f. Bellekteki verilerin kitlenme oranı

Veri tabanı bellek bölgesi ikincil bellek (cache) büyüklüğü ve fiziksel disk giriş/çıkış oranı: Bu oran verinin bellekte bulunma ihtimali üzerinde bize yorumlama gücü verir.

Tablolara erişim oranı: Tabloların sahip oldukları kayıtların ne oranda okunduğu hususunda bize bilgi döndürür.

İndeks kullanım oranı: Tablolardaki verilere erişirken ne oranda indeksler kullandığı gösteren metriktir.

Disk üzerindeki sıralamalar: Disk üzerinde hangi oranda sıralamalar yapıldığını gösteren metriktir.

Verilerin ne oranda zincirlendiği: Verilerin büyümesi, disk üzerinde verilerin dağınık olarak tutulmasına neden olabilmektedir. Büyüyen verilerin ne oranda zincirlendiğini gösteren metriktir.

Bellekteki verilerin kitleme oranı: Bellekteki veriye erişim yapılırken bir birini kitlenmesinden dolayı bekleme oranı gösteren metriktir.

2.3.3. Veri tabanı sisteminin sorguların yapılandırma metrikleri (sql tunning)

Sorguların metrikleri şu şekilde belirlenebilir [19] :

a. Sorgunun çalıştırma zamanı (elapsed time)
b. Sorgunun işlemciyi meşgul etme zamanı (CPU time)
c. Sorgunun bellekte ne kadarlık veri bloğunu meşgul ettiği (logical read/get)
d. Sorgunun diskte ne kadarlık veri bloğunu meşgul ettiği (phsyical read)
e. Sorgunun hangi sıklıkla çalıştırıldığı (execution)
f. Sorgunun çalıştırma şekli (hard or soft parse)

2.4. Veri Tabanında Erişilebilirlik Metrikleri

Birçok erişilebilirlik üzerine yapılan çalışmalarda veya hesaplamalarda kullanılan kavramlar şunlardır:

a. Arızaya kadar geçen ortalama süre - MTTF
b. Onarıma kadar geçen ortalama süre - MTTR
c. Arızalar arası ortalama süre – MTBF
d. Belirli bir zaman dilimindeki arıza sayısı
e. Arızanın bedeli

MTBF (mean time between failure) sistem güvenilirliğin ölçülmesinde temel parametredir. Bu parametrelerde temel alınan ölçü saattir.

Güvenilirlik (Realibility) = $e^{-\left(\frac{Time}{MTBF}\right)}$ (2.1)

A=1-F (2.2)

$F = 1 - f(1-a)^{s+1}$ (2.3)

Formüldeki;

A: Sistemin Ayakta Olma İhtimali (Availability)

F:Sistemin Durma İhtimali (Failure)

a: Sistemin bir nodunun Ayakta Olma İhtimali

s:Ayrılmış olarak Sistem bileşen sayısı

f:Ayrılmış olarak Sistem bileşenlerinden duranların sayısı

n:Sistem içindeki nodül sayısı

Veri tabanı sisteminin erişilebilirliğinden bahsedildiğinde üç temel bileşenin olması gerekmektedir. Bunlar [20] :

a. Veri tabanı servis bileşenlerin tümünün erişilebilir olması
b. Ağ sisteminin erişilebilir olması
c. Tüm disk depolama alanların erişilebilir ve bağlanabilir olması

2.5. Veri Tabanında Güvenlik Metrikleri

Güvenlik kavramı, bilgi sistemi mimarisinin en temel bileşenlerindendir. Bilgi mimarisinin her aşamasında ayrı güvenlik mekanizması uygulanmalıdır. Üç katmanlı mimari yapıda ara katman olan uygulama katmanı kullanıcının direk veri tabanı katmanına erişmesine engel olabilecek yapıdadır. Güvenli olmayan bir katman bir üst veya bir alt katmana negatif etkisi olacaktır. Her katmandaki güvenliğin bileşenleri şunları ihtiva etmeli:

a. Kimlik Doğrulanması (Authentication)
b. Kaynak Kullanım Yetkilendirmesi (Authorization)
c. Erişim Kontrolü(Access Control)
d. Denetleme (Auditing)

e. Şifreleme (Encryption)

Veri tabanı güvenlik özelinde bu parametreleri incelediğimizde ise:

Kimlik Doğrulaması

Veri tabanı kullanıcısının sisteme girebilmesini sağlayan süreçtir. Veri tabanında kullanıcı tanımlama yetkisine sahip kullanıcı tarafından ancak kataloglara tanımlanmış kullanıcı, veri tabanına erişebilme imkânına sahip olur.

Kimlik doğrulama ile veri tabanına erişmeden önce kontrol mekanizması ile güvenliği sağlanır. Kimlik doğrulama metodunun çeşitli algoritmaları vardır.

Bu algoritmalar; gömülü kimlik doğrulama, veri erişim nesneleri oluşturma, veri tabanın tablolarında kullanıcı ve şifre tutma, veri tabanı kullanıcısı oluşturma, tümleşik yapı (SSO, Single Sign On) kurmadır. Bu yöntemlerden en uygunu veri tabanı kullanıcısı oluşturmak veya tümleşik yapı kurmaktır [21] .

Kaynak kullanım yetkilendirilmesi

Erişime izin verilmiş kullanıcının hangi objeye hangi önceliklerle erişilebilirliğini kontrol eden süreçtir.

Erişim kontrolü

Kullanıcıların erişimindeki kısıtlamalardır. Dağıtık sistemlerde erişim kontrolü ayrı bir öneme sahiptir.

Denetleme ve Şifreleme

Erişimi yapılmış, kaynak kullanan kullanıcının sistem üzerindeki hareketlerinin denetlenmesine denir. Bu işlem, yapılanların inkâr edilemezliğini sağlayan süreçtir.

Verinin özel tekniklerle anlaşılmayan formata dönüştürülmesini sağlayan süreçtir. Temel olarak verinin güvende olduğunu anlamamız için gerekli şart, veriye yetkiyle erişim(confidentialty), verinin bütünlüğün korunması(integrity), veriye 7X24 erişilebilir (availability) olmasıdır.

BÖLÜM 3. METODOLOJİ

3.1. Veri Tabanında Yüksek Performansa Metodolojik Yaklaşım

Metriklerin yorumlanabilmesi için ilgili metrik değerlerinin istatistiklerinin toplanmış olması gerekmektedir. Ayrıca metrikler için üst sınır eşik değerleri (treshold) önceden belirlenmesi gerekir.

İstatistik toplama işini, yoğun olmayan zaman diliminde bir veri tabanı işi (system job) olarak yaptırılabilir. Oracle veri tabanı yönetim sisteminde bu işlem, iki şekilde yapılabilir; başlangıç parametre dosyasında yapılandırma tanımlanır ya da daha sonra yetkili kullanıcı ile (SYS) ilgili paket aktif edilir. Paketi aktif etmek için çalıştırılacak yapılandırma komut şu şekildedir [22] :

```
DBMS_SCHEDULER.ENABLE('GATHER_STATS_JOB');
```

Bu paketin bulunduğu işin çalışma penceresi ise varsayılan olarak gece 10 ile sabah 6 arasındadır. Artık veri tabanın yüksek yetkili şeması global istatistik değerlerini toplayacaktır. Bu değerler sağlıklı okunabilmesi için belli süre beklenmesi gerekir. Ayrıca veri tabanın her açılış anından itibaren toplanan veriler sunucunun tekrar açılması ile istatistikler sıfırlanır. Bu nedenle veri tabanı sunucuların reset edilmemesi daha güçlü istatistik için gerekir. V$SYSSTAT, V$SESSTAT ve V$STATNAME tablolarını sorgulayarak performans hakkında bilgi öğrenilebilir [23].

İstatistik toplanmasından sonra veri tabanı yönetmeni izleyeceği performans değerleri olacaktır. Bu değerlerin yorumlayabilmesi için Oracle veri tabanı yönetim sistemi tarafından otomatik rapor üretilebilen yazılımlar mevcuttur. Ayrıca Oracle VTYS de performans takibini yapan prosesler bulunmaktadır (MMON, MMNL). Bu

proseslerden MMON, dakikada bir istatistik verileri toplayıp bu verileri dinamik peformans görünüm olarak (V$) veri tabanına yönetmenine sunar. Bu görünümler anlık olarak fotoğraflanarak (snapshot) diskte saklanır. MMNL ise veri tabanı oturumların geçmişe yönelik ne yaptığının bilgisini tutar. Her on dakikada bir rapor oluşacak veriler ilgili görünümlere aktarılır. Ayrıca her otuz dakikada bir diske kaydedilir. Varsayılan ayarlara göre de bu bilgiler yedi gün boyunca diskte tutulur.

Proaktif yüksek performans ayarlamalarında amaç, sistem konfigürasyonunda donanım değişikliğin ne gibi etkileri olacağını önceden tespitini sağlayabilmektedir. Ayrıca proaktif yüksek performans ayarlamaları ile sistemin plansız olarak durmasına neden olmayan ancak performansın düşmesine ya da önlem alınmaz ise sistemin ileride plansız durmasına neden olabilecek problemlerin önceden tespit edilip çözülmesi hedeflenmektedir.

Otomatik Performans Teşhis İzleme (Automatic Database Diagnostic Monitor) yazılımı ile Oracle proaktif çözümler sunabilmektedir. Tespit ettiği sorunlardan bazıları şunlardır:

a. Yüksek maliyetli sorgular
b. Disk giriş/çıkış (I/O) maliyetleri
c. Kilitleme ve eşzamanlılık (concurrency) sorunları
d. Aşırı derece parse işlemi
e. CPU ve bellek gibi kaynakların darboğazları
f. Doğru yapılandırılmamış bellek alanları

ADDM yazılımın ürettiği raporda problemin bulguları, problemin ana sebebi ve öneriler bulunmaktadır. ADDM önerdiği çözümler şunlar olabilmektedir [24]:

a. Donanım değiştir.
b. Veri tabanı başlangıç parametresindeki konfigürasyonları güncelle.
c. Tabloları parçala veya indeks ekle veya tabloların dağınık verilerini birleştir.
d. Veri tabanına muhatap olunan uygulamaların veri yapılarını değiştir.

ADDM önerdiği yöntemlerden bazılarını detaylandırılırsa;

Donanım değiştirme

Donanım değişikliği özellikle büyüyen verilerden sorgulama yapılmasının zaman alması, erişimlerin yüksek olması nedeniyle kaynak yetersizliği ile gündeme gelebilmektedir. Teknolojik yeniliklerin uygulamalarda çeşitli imkânların doğmasına sebep olabilmektedir. Örneğin karakter tabanlı uygulamalarda tutulacak veriler sınırlı iken, grafik tabanlı uygulamalara geçiş ile uygulamalarla kullanıcı memnuniyetini arttıracak ek çözümler uygulamanın büyümesine neden olabilmektedir. Ayrıca internetin, mobiletinin artması iletişimin genişlemesi de uygulamaların daha çok veri barındırmasına ve erişim imkânın artması erişim sayısını hızla arttırmaktadır. Bunlar, ilk tasarlanan veri yapısını değiştirmekte, VTYS işlemlerin yürütüldüğü donanımın, yetersiz kalmasına neden olabilmektedir.

Veri tabanı başlangıç parametresinin konfigürasyonu değiştirme

Oracle veri tabanı örneğinin ilk açılış anında başlangıç parametrelerine göre bellekte tahsis işlemi yapar (init.ora). Buradaki ayarlamada Oracle örneğinin bellekteki mimarisi ile ilgilidir. Burada olması gereken minimum değerler ile ilgili ADDM tavsiyeler verebilir.

Tabloları bölümle, indeks ekle veya tabloların dağınık verilerini birleştirme

Tabloları bölümleme (partitioning)

Tablolar mantıksal yapısını koruyarak fiziksel olarak parçalanabilir (partition). Yani büyüyen tabloların verileri daha başlangıçta birden fazla segmentte olacak şekilde ayarlanabilir. Bu özellikle arama işlemlerinde performansa direk olumlu etti etmektedir. Tarihsel, coğrafik verilerde bu işlem tavsiye edilmektedir. Örneğin personelin kapı giriş çıkış verilerini tuttuğunuz tabloyu bölümleme yapmaz iseniz belirli ay veya gün için arama yaptığınızda tabloyu ilk satırdan son satıra kadar tarama

ihtimali olacağından performans açısından kayıp olacaktır. Oysa tabloyu aylık olarak bölümleme yaparsanız Oracle VTYS bunu segmentlere böldüğünden o ayın segmentinde veriyi arayacaktır ve arama sonucu çok hızlı dönecektir.

Farklı tablo bölümleme teknikleri vardır. Bunlar aralıklı bölümleme(range partitioning), liste bölümleme (list partitioning), anahtarlı bölümleme (hash partitioning), birleştirme bölümleme (composite partitioning) ve liste-aralıklı(range-list partitioning) bölümleme yöntemleridir.

Tablo bölümlemesine örnek Şekil 3.1.'de gösterilmiştir.

```
CREATE TABLE FATURA
(
  MUSTERI_ID NUMBER,
  AD VARCHAR2(50),
  SOYAD VARCHAR2(50),
  FATURA_TARIH DATE
)

TABLESPACE MUHASEBE
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
MONITORING
PARTITION BY RANGE (FATURA_TARIH)(
PARTITION FATURA201301 VALUES LESS THAN(TO_DATE('01/01/2013','MM/DD/YYYY')),
PARTITION FATURA201302 VALUES LESS THAN(TO_DATE('01/02/2013','MM/DD/YYYY')));
```

Şekil 3.1. Aralıklı tablo bölümleme tekniğine örnek

Şekil 3.1.'de fatura tablosu muhasebe tablo uzayında aralıklı bölümleme (partition by range) tekniği ile oluşturulmaktadır. Fatura tablosunu aylık segmente bölümleme ile ayırmaktadır.

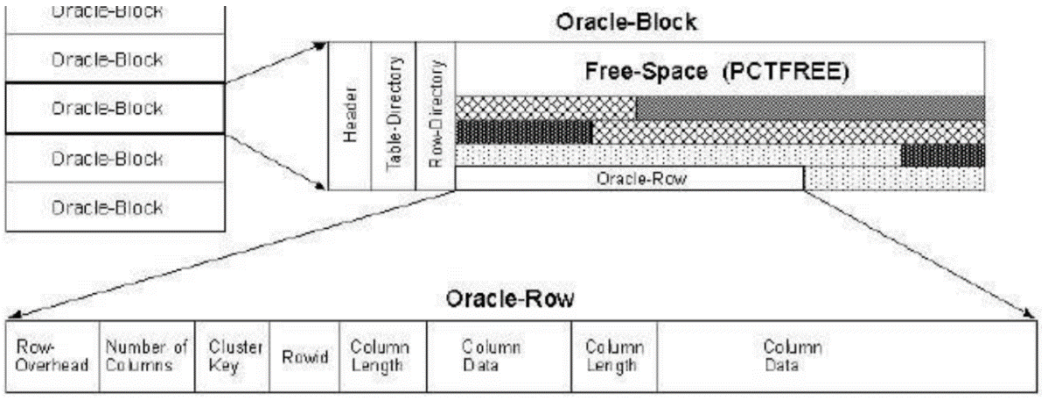
Oracle VTYS yazlımı olan ADDM, veri tabanı yönetmenine bazı tablolar için indeks üretmesini önerebilir. Verilere etkin erişim metodu olan indeksleme tekniği ilişkisel veri tabanı yönetiminde yoğun kullanılmaktadır. Şekil 3.2.'de örnek kodlama ile nasıl oluşturabileceğimizi gösterilmektedir.

```
CREATE INDEX FATURA_IDX$$_F53A0002 ON FATURA
(FATURA_TARIH)
LOGGING
TABLESPACE MUHASEBE
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
           );
```

Şekil 3.2. Örnek indeks oluşturma

Tabloların fiziksel verilerin dağınıklığını birleştirme

Oracle VTYS bir diskin bloğunda veriyi şu şekilde saklamaktadır:



Şekil 3.3. Oracle VTYS bir kayıt bilgisinin diskteki adreslemesi (Zahn,2007)

Oracle bloğuna bakıldığında başlık kısmı, boş alan verinin olduğu bölüm görülmektedir. Önceden ayarlanan boş alan yüzdesi kayıt zincirlemesi (row chained)

veya kayıt göçüne (row migration) etki etmektedir. Bu boş alanın yüzdesini şu komutla değiştirilebilir:

ALTER TABLE <tablo_adı> MODIFY DEFAULT ATTRIBUTES PCTFREE <yüzde oranı>;

Büyüyen tabloların verileri, çok fazla güncellemeye, eklemeye ve silmeye muhatap olabilmektedir. Bu durum tabloların diskteki verilerin fiziksel konumlarını çok fazla dağıtabilmektedir. Bu noktada tablonun verilerinin birbiriyle uzakta kalması performans problemine neden olabilmektedir. Bunun için tablonun verileri mümkün olabildiğince bir arada tutulması sağlanmalıdır. Tablo bölümünün(segment) yüzde olarak boş bırakılma oranı, tabloların zincirleme oranları takip edilerek uygun değer atanmalıdır. Bu değer yüksek performans için önemlidir (pctfree).

```
1  ANALYZE TABLE ITFA COMPUTE STATISTICS;
2
3  SELECT chain_cnt,
4         round(chain_cnt/num_rows*100,2) pct_chained,
5         avg_row_len, pct_free , pct_used
6    FROM user_tables
7  WHERE table_name = 'ITFA';
8
9
```

Data Grid

Messages | Data Grid | Trace | DBMS Output (disabled) | Query Viewer | Explain Plan

CHAIN_CNT	PCT_CHAINED	AVG_ROW_L...	PCT_FR...	PCT_USED
11818	10.18	272	10	

Şekil 3.4. Tablo verilerin fiziksel olarak (disk) dağınıklığı

Tabloların verilerinin dağınık olup olmadığını anlamak için ilk önce analiz ettirmemiz gerekmektedir. Bunu Şekil 3.4.'deki ilk komutla yapılır.

ANALYZE TABLE <tablo adı> COMPUTE STATISTICS;

Daha sonra analiz edilen tablonun, yüksek yetkili kullanıcının (SYS) ilgili tablosundan analiz ettirilen tablonun istatistiki verileri sorgulanır.

Şekil 3.4.'deki sorgudan anlaşılan tablo için ilgili bölümün(segment) yüzde 10'luk kısmı (PCT_FREE) güncellemeler için bırakılmıştır. ITFA tablosunun tam 11818 defa kayıt hareketinin zincirlendiği ya da göç ettiğini CHAIN_CNT kolonundan görülebilmektedir. Bu durum tablonun verilerinin dağınık olduğunu tek başına göstermez. Ayrıca bakılması gereken, tablonun bulunduğu veri tabanındaki tabloların genel durum istatistiğidir. Şekil 3.5.'de bu durum gösterilmektedir.

```
1  SELECT name,value FROM v$sysstat WHERE name like '%table%'
```

Data Grid

Messages | Data Grid | Trace | DBMS Output (disabled) | Query V

NAME	VALUE
physical reads direct temporary tablespace	5293
physical writes direct temporary tablespace	29449
DBWR tablespace checkpoint buffers written	20934
DBWR transaction table writes	256772
transaction tables consistent reads - undo records applied	2517451
transaction tables consistent read rollbacks	3170
auto extends on undo tablespace	0
CLI bytes fls to table	0
table scans (short tables)	16034384
table scans (long tables)	230019
table scans (rowid ranges)	229270
table scans (IM)	0
table scans (cache partitions)	0
table scans (direct read)	226332
table scan rows gotten	65537840121
table scan disk non-IMC rows gotten	65242096913
table scan disk IMC fallback	0
table scan blocks gotten	2397223829
table fetch by rowid	4601797177
table fetch continued row	10588135
LOB table id lookup cache misses	14

Şekil 3.5. Veri tabanı sisteminde tablolar için genel istatistik değerler

Bu şekildeki "table fetch continued row" satırı veri tabanındaki zincirli olarak kayıtların getirilme sayısını vermektedir.

Zincirlenmiş kayıtların getirilmesi yüksek performansa olumsuz etki ettiğine kanaat getirildiği anda uygulayacağımız metodolojiler şunlar olabilir [25] :

a. Tabloyu taşımak
b. İndeksleri tekrar derlemek

Tabloyu taşımak

Problem görülen tablo şu komut ile taşınabilir.

“ALTER TABLE < tablo adi> MOVE “

İndeksler tekrar derlemek

Tablolar taşındığında üzerindeki indeksler kullanılmaz hale gelmektedir. Bu nedenle üzerindeki tüm indeksler tekrar derlenmedir.

ALTER INDEX <indeks adi> REBUILD

Veri tabanına muhatap olunan uygulamaların veri yapılarını değiştirme

Veri tabanı sisteminde ilişkilerin kurulduğu tabloların anahtarları tekil olması (unique) mantıksal tasarımı güçlendirdiği kadar performansa da olumlu etki yapar. Anahtarları oluşturabilmek için veri modelini düzgün kurulması gerekir. Bu normalizasyon veya denormalizasyonun doğru kullanılması ile sağlanabilir.

Normalizsyon ve denormalizasyon

Veri tabanında normalleştirme bir nevi ayrıştırma, veri modellemenin üst seviye işlemleridir. Verileri oluştururken verinin tekrarlanmasına, kaybına veya yetersizliğine neden olan yapının varlığı hem veri katmanında hem de uygulama yazılımın yazıldığı katmanda performans sorunudur. Performans sorunundan kasıt bakımı zor yapı,

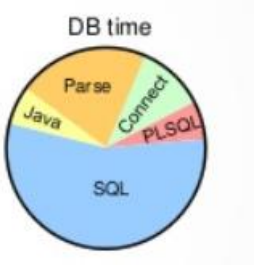
gereksiz disk alanı işgali ve meşgul edilen işlemcidir. Gereksiz verinin bulunmadığı, normalize edilmiş varlık kümelerinin indeksli yapısı arama sürelerine de etki yapacağı muhakkaktır. Kopyalama miktarı azalan veri kütlesinin, depolama maliyetini düşüreceği gibi veri tabanın arama motorlarının daha performanslı olmasını sağlayacaktır.

Her şeyden önce normalizasyon işlemi mantıksal tasarım yani mantıksal şemada gözlemlenir. Mantıksal tasarımda başlıca hedef, verilerin, onların arasındaki ilişkilerin ve sınırlamalarını matematiksel mantıkla kesin, tam ifade edilmesidir. Varlık kümeleri arasında ilişki kümelerini kurarken ilgili matematiksel model dikkate alındığında normalizasyon formları ortaya koyulur. Odaklanacağımız nokta işlevsel bağımlılık ve kısmi bağımlılıktır. Varlık kümesinin anahtarlarının bir biriyle ilişki kümesinde bağımlılık türü normalizasyonda form atlamasına neden olur. Formlar aslında kurduğunuz veri modelini gözden geçirme sırasıdır. Sırasıyla eksikleri giderip bir üst forma geçiş yapılır. Hangi formda bırakılacağı yine tasarımcının inisiyatifindedir. Yönetebileceği yapı ile performans kriterlerin arasında tasarımcı en uygun seçeneği uzmanlık beceresi ile karar verecektir.

Finkseltein ve arkadaşları tarafından temelleri atılan denormalizasyon işleminde ise ilk kural normalizasyon yapılmasıdır. Yani normalizasyonu yapılmış tasarımın denormalizasyon yapılabileceğini belirtilmiştir. Denormalizasyon genelde verinin çok fazla büyümesi sonucu ihtiyaç hissedilmektedir [26].

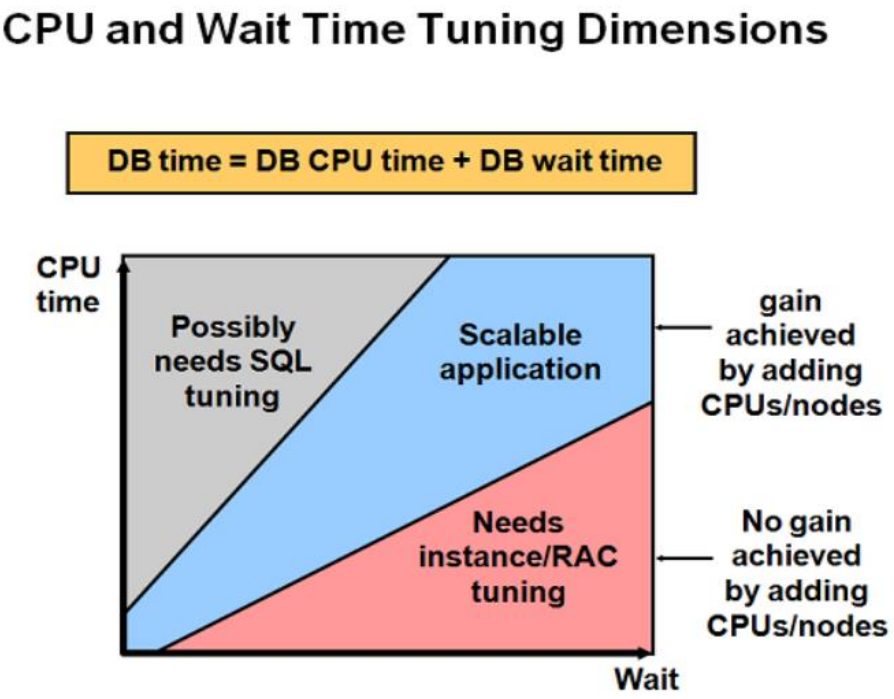
Denormalizasyon yapılırken Coleman, veri miktarı, işletim sisteminin ve hatta donanımın dahi dikkate alınması gerektiğini belirtmiştir. Denormalizasyon, normalizasyon işleyişinin tersi olarak tabloların birleştirilmesi üzerine kuruludur.
Sorgu ayarlamaları

Veri tabanı performans metriklerinde bahsettiğimiz gibi veri tabanı zamanı performans bakımından teşhis konulabilecek verileri içermektedir.



Şekil 3.6. Veri tabanı zamanın bileşenleri (Oracle,2016)

Şekil 3.6.'daki gösterimle görünüyor ki normal bir veri tabanında, sorgular veri tabanı zamanın yarısınından fazlasını meşgul etmektedir. Veri tabanında yüksek performansla hedeflenen veri tabanı zamanının azaltmaktır. Veri tabanı zamanın içerdiği tüm bileşenleri daha hızlı sonlandırılabilirse veri tabanı zamanın toplam süresi düşer[24] .



Şekil 3.7. Performans ayarlama ihtiyacının tespiti (Oracle,2016)

Şekil 3.7. aslında genel konjonktürü göstermektedir. Veri tabanı zamanının çoğunu CPU alıyorsa, bu kötü kodlanmış sorguların veri tabanı yönetim sisteminde yoğun kullanıldığını gösterir. Eğer ki veri tabanı zamanının çoğu bekleme ile geçiyorsa bu da

kaynakların yeterli olmadığını gösterir. Bunun için ilave bellek veya işlemci eklenmelidir.

Geçici veri saklama bölgeleri, en son kullanılan disk bloklarını depolayarak belleğe göre daha yavaş olan diske erişimi azaltıp performansı arttırmaktadır. Ortak veri havuz bölgesi ise belirli sorguları, işletim planlarını ve çalıştırılmaya hazır pars edilmiş kodları saklayarak tekrar aynı sorguların talep edilmesi durumunda hazır edilmiş sorguyu sunarak performansın arttırılmasına katkı sağlamaktadır.

Performans getirisi olabilmesi için kullanıcıların önceden istediği sorgunun kütüphane belleğinde çalıştırılabilir halde olması beklenir. Bunun olasılığının yüksek olması verimli olduğunu gösterir.

Aynı yazılım uygulamalarının bazı kısımların farklı kullanıcılar tarafından sadece parametresi farklı olarak gönderilmesi, ayrıca aynı uygulamayı değişik zamanda aynı ya da farklı kullanıcının çalıştırması, aynı sorguların sürekli tekrar çalıştırıldığını göstermiştir. Bu farkındalık ile bu tarz sorguların istatistik değerlerine yoğunlaşmasının gerektirdiğini gösterir. Bu da bir sorgu ifadesi için ortalama istekte bulunan kullanıcı sayısıdır.

Sorgu iyileştirici(Sql optimizer), veri tabanı optimizasyonu sağlayan bileşenlerinden biridir. Bu bileşen iki teknikle çalışır: Kural Tabanlı Optimizasyon (Rule-Based Optimizer) ve Maliyet Tabanlı Optimizasyon (Cost-Based Optimizer) . Kural tabanlı optimizasyon tahmin yöntemlerini kullanırken, diğeri teknik akıl yürütme yöntemini kullanır ve sorguları çalıştırır.

Kural Tabanlı Optimizasyon (Rule-Based Optimizer)

Bu araçların hedefi veri tabanının disk yönetimine minimum iş yaptıracak sorgularının oluşturmasıdır. Bu hedef için Rule-Based Optimizer (RBO) önceden tanımlanmış kuralları barındıran seti kullanıp sorguyu çalıştırırken hangi yolu tercih edeceğini belirler. Bu kuralların formatı:

SELECT /* + RULE + */ …

şeklin de belirlenir. Bu komut için veri tabanı sisteminde yapılandırma seviyesinde ayarlamalar yapılabilir.

Maliyet Tabanlı Optimizasyon (Cost-Based Optimizer)

Geniş kapsamlı ve oldukça kompleks çalışma prensibini barındırmaktadır. Yöntemi için çeşitli tablo boyutları, kayıt sayıları, verilerin dağılımı gibi bilgileri kullanmaktadır. Bu tekniği kullanabilmek için özel prosedürü ile objelerin analiz edilmeleri ve istatistiklerinin toplatılması gerekmektedir. Eğer bir tablonun analizi yapılmamışsa RBO tekniği ile yolu belirlenir. Lakin aynı sorgunun bazı objeleri analiz edilmiş bazıları ise analiz edilmemiş ise veri tabanı yönetim sistemi önceliği analiz tekniğini kullanan Cost-Based Optimizeri kullanır. Bu tekniği kullanabilmek içinde konfigürasyon ayarları gerekmektedir.

Veri tabanındaki geçersiz objeler

Veri tabanı sisteminde zamanla bazı objeler geçersiz (invalid) duruma düşebilmektedir. Bu durumlarda objeleri tekrar geçerli (valid) hale getirmek performansa olumlu katkı yapmaktadır. Bu işlem için Oracle VTYS özel paketi bulunmaktadır. Bu paketi çalıştırmak için yüksek yetkili kullanıcı ile sisteme giriş yapıp “utlrp” paketini çalıştırmak gerekmektedir.

Sqlplus / as sysdba
SQL > @?/rdbms/admin/utlrp

Performans izleme

Veri tabanı sisteminizi metodolojiler uygulayarak yüksek performasa getirebilirsiniz. Ancak bunun sürekliliğini sağlayabilmeniz için yani yüksek erişilebilirliğe ulaşabilmeniz için sistemin performans metriklerinin takip edilmesi gerekir. Bunun

için TOAD markasının SPOTLIGHT ürünü gibi görselliği kuvvetli, belirlediğiniz eşik (treshold) değerlerine göre uyarı mekanizmasına (alert) sahip veri tabanı yönetmenleri için kullanılabilirliği yüksek, yazılımlar gereklidir.

Şekil 3.8. Performans izleme uygulaması (Toad,Spotlight)

3.2. Veri Tabanında Yüksek Erişilebilirliğe Metodolojik Yaklaşım

Erişilebilirlik arttıkça sisteme olan güven artmaktadır. Güvenin mühendislik disiplinindeki yeri ise sosyal alandan farklı olmuştur. Mühendislik çalışmaları hesaplamaya dayanır. Güvenilirlik aslında bir karşılaştırma kavramıdır. Mantıksal olarak güvenilir veya güvenilmez nitelemeleri büyük anlam taşımaz. Önemli olan "ne derece güvenilir?" sorusudur.

Von Braoun ve ekibi II. Dünya savaşı sırasında füze çalışmaları yaparken şunu fark etmiştir: Füzenin tüm bileşenlerinden birinin diğerlerinin başarısını engelleyebilmektedir. Bu da bir sistemin güvenilirliğin kazanması için her alt bileşenin tam istenen randımanı sürekli vermesi gerektiğini göstermektedir.

Maliyeti 1$ olan elektronik ürünün güvenilir şekilde sürekli erişilebilir olarak hizmet verebilmesi için 2$ harcanması gerektiğini AGREE (Advisory Group on Reliability of Electronic Equipment) ortaya atılmıştır. Bu durum, teknoloji dünyasına yeni bir anlayışın girmesine neden olmuştur. O anlayış; daha pahalı ama daha az bakım gerektiren ürün tasarımının yapılması gerektiğidir. Bu da erişilebilirlik açısından daha az bakım ile planlı veya plansız hizmet dışı kalma süresinin değişmesi anlamına gelmektedir.

Erişilebilirlik ile güvenirlilik arasında ince ayrım var. Bu nüans, erişilebilirliğin bir oran, güvenirliliğin ise olasılık olduğudur. Güvenirlik, birim zamanda herhangi bir bileşenin, cihazın veya servisin kesintiye uğramama olasılığıdır.

Bir sistemin güvenirliliği [27] belirli bir zaman periyodunda ve verilen koşullar altında sistemin arıza olmaksızın çalışma olasılığıdır. Dolayısıyla güvenirlilik herhangi bir sistemin o an için başarı performansının olasılığı ile ilgilidir.

Sürekliliği sağlanan bir sistem mimarisi beklenen faydayı sağlayacaktır. Bu bağlamda sürekliliğin alt yapısı güçlü olasılığa sahip güvenirliliktir.

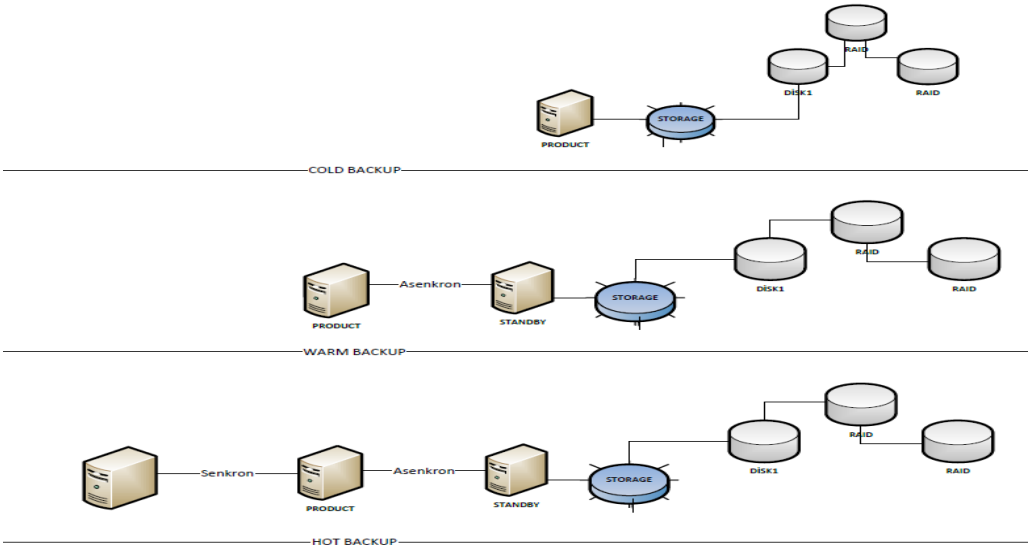
Yüksek devamlılığı ölçmek için çeşitli algoritmalar mevcuttur. Veri tabanı servisinin devamlılığını algoritmik olarak şöyle belirtebiliriz [28].

```
ALTER FUNCTION ufrVeriTabaniErisilebilirligi
   (
     @MTBF real,  --saat olarak
     @MTTRealise real,  -- saat olarak
     @MTTR real     -- saat olarak
   )
RETURNS REAL
AS
BEGIN
   RETURN
```

```
( @MTBF - ( @MTTRealise
                    + @MTTR )
) * 100.000/ @MTBF * 1.0000
END

SELECT  dbo.ufrVeriTabaniErisilebilirligi                (2 * 30 * 24, 4, 8)
--99.1666
```

Veri tabanlarında sistemleri yedekli yaparak devamlılığı arttırılabilmektedir. Günümüzdeki endüstrilerde bilinen iki veri tabanı yönetim sistemlerinde (ORACLE ve MSSQL) üç ana tür yedekleme mekanizmaları kurulmaktadır. Soğuk (Cold),Ilık (Warm) ve Sıcak (Hot) yedekleme (backup). İsimlendirmedeki ana espiri ise yedeğin alış ve saklanış biçimi ile ilgilidir. Soğuk (Cold Backup) yedeklemede, ana ürün veri tabanından ayrılmış eş zamansız (asenkron) şekilde yedeklenen ve ayakta olmayan bir yapıdır. Konfigürasyonu ise ana ürün veri tabanı ile birebir aynıdır. Ilık (Warm Backup) yedeklemede ise, birincil veri tabanından ayrılmış eş zamansız (asenkron) şekilde yedeklenen ama ayakta olan bir yapıdır. Konfigürasyonları ana ürün veri tabanı ile aynıdır. Sıcak yedekleme (Hot Backup) ise ana ürün veri tabanı ile eş zamanlı (senkron) şekilde yedeklenen ve ayakta olan yapıdır [29].



Şekil 3.9. Yüksek erişilebilirlikte veri tabanı mimarisi

Şekil 3.9.'daki bu üç modelin erişilebilirliğini matematik dili ile ifade edersek [30] sistemin ayakta durma ihtimali (Denklem 4.2) ve sistemin durma ihtimali ise Denklem 4.3 ile hesaplanabilir:

A=1-F (4.2)

$F = 1 - f(1 - a)^{s+1}$ (4.3)

Formüldeki;

A: Sistemin Ayakta Olma İhtimali (Availability)

F:Sistemin Durma İhtimali (Failure)

a: Sistemin bir nodunun Ayakta Olma İhtimali

s:Ayrılmış olarak Sistem bileşen sayısı

f:Ayrılmış olarak Sistem bileşenlerinden duranların sayısı

n:Sistem içindeki nodül sayısı

Eğer sistemimiz iki nodülü olarak aktif/aktif tasarlarsak formülü şu şekilde hesaplanmaktadır (Denklem 4.4) :

$A = 1 - F = 1 - (1 - a)^2$ (4.4)

Soğuk yedekleme (Cold Backup) sistemi ile kurulan veri tabanı yönetiminin devamlılığında, ayrılmış sistem bileşen sayısını (s=1) alırsak ve bileşenlerin kendi devamlılık oranlarını 0.99 kabul edersek:

A=$1-(1-0.99)^2$ => 1-0,0001 => 0,9999 olur.

Tablo 3.1. Süreklilik Oranı ve Kesintilerin Etkisi

No	Süreklilik	Kesinti/ Yılda
1	%90.0	36 gün 12 Saat
2	%99.0	87 saat 36 dakika
3	%99.9	8 saat 46 dakika
4	%99.99	52 dakika 33 saniye
5	%99.999	5 dakika 15 saniye
6	%99.9999	31.5 saniye

Tablo 3.1' den dört tane 9s olan satırda yaklaşık 52 dakika 33 saniyelik yıllık kesinti beklenmektedir. Yalnız dikkat edilmesi gereken nokta nodül sayısını arttırmak devamlılığı azaltmaktadır. Çünkü ne kadar sistemi oluşturan bileşenler olursa buda o kadar yolla sistemin durma ihtimalini arttıracaktır. Kombinasyon yaklaşımıyla hesaplarsak şu şekilde olmaktadır (Denklem 4.5):

$$f = \frac{\mathrm{n!}}{(\mathrm{s+1})!(\mathrm{n-s-1})} \tag{4.5}$$

6 nodüllü ve 2 ayrılmış sistem için f kombinasyonu 20 değerini döndürmektedir. Böylelikle 20 yol ve bu yollardan her hangi birinin nedeniyle sistem öngörülmemiş şekilde göçebilir (failover). Bunun devamlılığını hesaplarsak (Denklem 4.6) :

$$F = f(1-a)^{s+1} \tag{4.6}$$

$F = 20(1-0.99)^3$

F=.00002 sistemin durma ihtimali üzerinden hesaplarsak (Denklem 4.7):

$$A = 1 - F \tag{4.7}$$

Bunun sonucunda devamlılık oranımız yaklaşık 5 (beş) 9 sonucunu verir. Buda yaklaşık 5 dakika 15 saniyelik bir aksamayı yıllık öngörebilen bir devamlılığa sahip olduğunu göstermiş olmaktadır. Bu sayede soğuk yedekleme sisteminde 52 dakika 33 saniye olan aksama süresi 2 ayrı sistem ve 6 nodüllü sistemde bu süre 5 dakika 15 saniye dönüşebilmektedir.

Regülasyonlara tabii tutulan kamu kurumların veri merkezleri artık yüksek erişilebilirlik için felakat kurtarma merkezlerinde, iş sürekliliği merkezlerinde eşlenik sunucuları bulundurma zorunluluğu vardır. Bu sunucular yüksek erişilebilirlik için gerekli yedekli yapılardır.

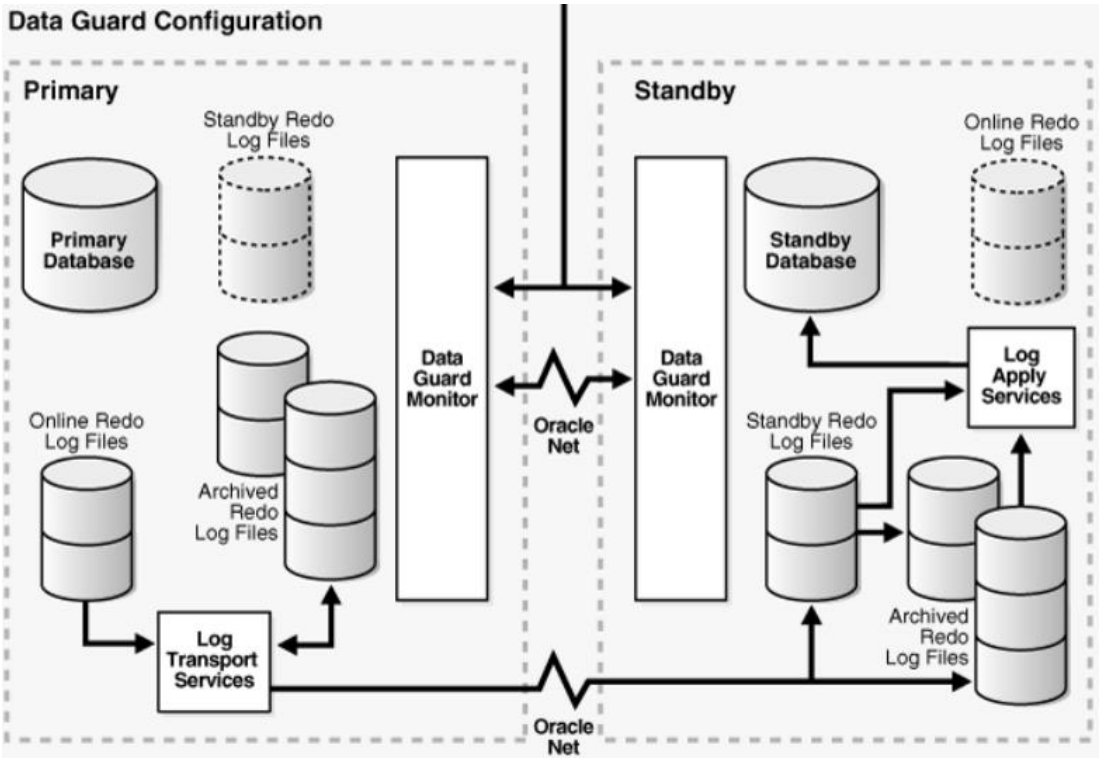
İş sürekliliği merkezleri için kurumlar birincil veri taban sistemlerine birebir aynı (cluster) yapıda sunucular ilave etmektedir. Mimari olarak birincil-ikincil (primary-standby) tekniği ile veri tabanların verileri yedeklenmektedir. Şekil 3.9.'daki mimari ile yedeklenme yönteminde üç tip modu bulunmaktadır [31]:

a. Maksimum Erişilebilirlik
b. Maksimum Koruma
c. Maksimum Performans

Maksimum Erişilebilirlik: Bu modda çalışan mimaride, hareket logların (transaction) çıkardığı arşiv logların yedek sunucuya aktarılmasını beklenir. Önceden belirlenen "zaman aşımı (timeout) süresine" kadar cevap alamazsa sistem kendisini kapatır.

Maksimum Koruma: Bu modda çalışan mimaride, hareket logları (transaction) çıkardığı arşiv logların yedek sunucuya aktarılmasını bekler. Ancak ani kesintide birincil sistem kendisini korumaya alır.

Maksimum Performans: Bu modda çalışan mimaride ise hareket logların (transaction) çıkardığı arşiv logların yedek sunucuya aktarılması beklenmeden birincil sistem işlemlerine devam eder. Diğer taraftaki ikincil sistem (standby) eş zamansız (asenkron) olarak gelecek arşiv logları kendi veri sistemine entegre eder.



Şekil 3.10. Veri tabanında yüksek erişilebilirlik için yedekleme -primary ve standby (Oracle, 2011).

Anlık veri kaybına tahammülü olmayan sistemler için maksimum erişilebilirlik ve koruma tercih edilmesi gerekirken, az miktarda veri kaybı karşılığında yüksek performans talep eden yapılar için maksimum performans modu tercih edilmelidir.

Yüksek erişilebilirliğe ulaşmak için hedeflenen yapı üç kısma ayrılmaktadır. Bunlar:

a. Birincil veri tabanı sisteminin aktif-aktif olarak aynı lokalde eşlenik yedeğinin alınması
b. Birincil veri tabanı sisteminin aktif-aktif olarak farklı lokalde eşlenik yedeğinin alınması
c. Birincil veri tabanın sisteminin aktif-pasif olarak farklı lokalde eşlenik yedeğinin alınması

Aynı lokalde aktif-aktif eşlenik yedek alınması

Veri tabanı sunucuların donanımsal olarak kümeleme ile (cluster) bire bir aynı sisteme sahip eşlenik olarak yeni bir sunucu üretilmektedir. Bu sistemlerde veri tabanı yönetim sistemi tek bir yazılımla (binary) birden fazla sunucuyu yönetebilmektedir. Eşlenik sunucularından birinin kapatılması sistemi kesintiye uğratmamaktadır, bu da erişilebilirliğe katkı sağlamaktadır.

Farklı lokalde aktif-aktif eşlenik yedek alınması

İş sürekliliğinin ana iskeletini oluşturan yapı aktif-aktif farklı lokalde yedek alınmasıdır. Aktif – aktif kasıt her iki nodda ayakta ve hizmet verir durumdadır. Yedek sistem ana sistemden aldıkları arşiv loglarını sistem ayakta iken kendi veri dosyalarına uygulayabilmektedir (hot backup). Ayrıca bu sunuculara sadece okuma modu olarak (read-only) erişildiği için bu sunucuları, raporlama amaçlı kullanılabilmektedir.

Aktif – Aktif Yedekleme Tekniği (Active Data Guard)

Aktif – aktif yedeklenme metodolojisini örnek bir uygulama ile gösterilirse

Öncelikle veri tabanımız arşiv log modunda olması yani çevrimiçi iken yedek alınabilmesi gerekir.

1. Adım: Ana veri tabanı sisteminin log modu “force” çekilir.

   Alter database force logging;

2. Adım: Ana veri tabanı sistemine (primary) diğer sistemin (standby) konfigürasyon bilgileri tanımlanır.
   ALTER SYSTEM SET LOG_ARCHIVE_CONFIG='DG_CONFIG=(<ana veri tabanin adı (unique_name),<anlık yedeklenecek veri tabanı sunucun adı)';

3. Adım: Ana veri tabanında çalıştırılacak oluşturulan arşiv logların diğer sisteme gidileceğini (standby) tanımlayan konfigürasyon kodu tanımlanır.

ALTER SYSTEM SET LOG_ARCHIVE_DEST_2='SERVICE= <"diğer yedek veri tabanı sunucun adı"> NOAFFIRM ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME='<"diğer yedek veri tabanı sunucun adı">';
ALTER SYSTEM SET LOG_ARCHIVE_DEST_STATE_2=ENABLE;

4. Adım: Ana veri tabanındaki arşiv logların parametrik ayarlanması ve şifre dosyalarının sadece bir veri tabanından kullanılabilmesi sağlanır.

   ALTER SYSTEM SET LOG_ARCHIVE_FORMAT='%t_%s_%r.arc' SCOPE=SPFILE;
   ALTER SYSTEM SET LOG_ARCHIVE_MAX_PROCESSES=30;
   ALTER SYSTEM SET REMOTE_LOGIN_PASSWORDFILE= EXCLUSIVE SCOPE=SPFILE;

5. Adım: Ana veri tabanı ile yedek veri tabanın rollerinin birbirinin yerine geçmesini sağlayabilecek yapılandırma kodu tanımlanır.

   ALTER SYSTEM SET FAL_SERVER='<yedek veri tabanı adı>';
   ALTER SYSTEM SET STANDBY_FILE_MANAGEMENT=AUTO;

6. Adım: Her iki veri tabanı sunucusunda tnsnames.ora dosyalarına kendilerini birbirine tanıtacağı konfigürasyon girişlerin yapılması sağlanır.

   Örnek tnsnames.ora her iki sunucuda tanımlı olmalıdır.

   ANA_VYTS =
   (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = anavtys.domain)(PORT = 1521)))
   (CONNECT_DATA = (SERVICE_NAME = ANAVTYS)))

   STDBY_VTYS =

(DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST =stdbyvtys.domain)(PORT = 1521)))
(CONNECT_DATA = (SERVICE_NAME = STDBYVTYS)))

7. Adım: Ana veri tabanında diğer yedek veri tabanına (standby) aktarılması gereken kontrol ve parametre dosyasıdır. Bu iki konfig dosyaları her iki sunucuda aynı olmalıdır. Aşağıdaki komutlar ana veri tabanında çalıştırılır.

   ALTER DATABASE CREATE STANDBY CONTROLFILE AS '/tmp/stdby.ctl';
   CREATE PFILE='/tmp/initstdby.ora' FROM SPFILE;

8. Adım: Yedek veri tabanında (standby) oluşturulması gereken klasörler oluşturulur. Bu dizinler ana veri tabanın aynı dizin adresinde bulunmalıdır.

   mkdir -p /u01/app/oracle/oradata/<ana veri tabanı adı>
   mkdir -p /u01/app/oracle/flash_recovery_area/<ana veri tabanı adı>
   mkdir -p /u01/app/oracle/admin/<ana veri tabanı adı>/adump

9. Adım: Ana veri tabanında yedek veri tabanına aktarılacak dosyalarının güvenli kopyalama aktarılımı yapılır.

   scp/tmp/stby.ctl
   oracle@<yedekveritabanısunucuip>:/u01/app/oracle/oradata/<anaveritabaniadi>/control01.ctl
   scp/tmp/stby.ctl
   oracle@<yedekveritabanısunucuip>:u01/app/oracle/flash_recovery_area/<anaveritabaniadi>//control02.ctl
   scp /tmp/initstby.ora oracle@<yedekveritabanısunucuip>::/tmp/initstby.ora
   scp /u01/app/oracle/product/db_1/dbs/orapw
   oracle@<yedekveritabanısunucuip>::/u01/app/oracle/product/db_1/dbs

10. Adım: Yedek veri tabanında listener.ora dosyası konfigüre edilip hizmete açılır.

    lsnrctl start

11. Adım: Her iki tarafta bir biri ile iletişim kurabilecek Oracle VTYS ait redo log dosyaları örneğin 50M lık oluşturulur.

    ALTER DATABASE ADD STANDBY LOGFILE ('/u01/app/oracle/oradata/<anaveritabaniadi>/standby_redo01.log') SIZE 50M;
    ALTER DATABASE ADD STANDBY LOGFILE ('/u01/app/oracle/oradata/<anaveritabaniadi>/standby_redo02.log') SIZE 50M;
    ALTER DATABASE ADD STANDBY LOGFILE ('/u01/app/oracle/oradata//<anaveritabaniadi>/standby_redo03.log') SIZE 50M;
    ALTER DATABASE ADD STANDBY LOGFILE ('/u01/app/oracle/oradata//<anaveritabaniadi>/standby_redo04.log') SIZE 50M;

12. Adım: Yedek veri tabanı (standby) “nomount” modunda açılır.

    sqlplus / as sysdba
    STARTUP NOMOUNT PFILE='/tmp/initstby.ora';

13. Adım: Ana veri tabanı tarafında RMAN uygulamasına bağlanılır.

    rman TARGET sys/<şifre>@<ana veri tabanı adı> AUXILIARY sys/<şifre>@<yedek veritabanı adı>

14. Adım: Ana veri tabanında RMAN uygulamasında şu kritik komut çalıştırılır.

DUPLICATE TARGET DATABASE FOR STANDBY
FROM ACTIVE DATABASE DORECOVER
SPFILE SET db_unique_name='<yedek veri tabanı adı>' COMMENT ' Is standby'
SET LOG_ARCHIVE_DEST_2='SERVICE=<ana veri tabanı adı> ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=<ana veri tabanı adı>'
SET FAL_SERVER='<ana veri tabanı adı>' COMMENT 'Is primary'
NOFILENAMECHECK;

15. Adım: "Nomount" modundaki yedek veri tabanında gelen arşiv logları uygulaması için şu komut çalıştırılır:

ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION

16. Adım: Ana veri tabanında kontrol amaçlı olarak şu sorgu yazılarak eldeki arşiv logların yedek veri tabanında olup olmadığını anlaşılabilir.

SELECT sequence#, first_time, next_time FROM v$archived_log ORDER BY sequence#;

17. Adım: Aynı şekilde yedek veri tabanında kontrol amaçlı şu sorgu çalıştırılır:

SELECT thread#,low_sequence#,high_sequence# FROM v$archive_gap;
SELECT sequence#, first_time, next_time FROM v$archived_log ORDER BY sequence#;

18. Adım: Eğer işlemler bu adıma kadar beklendiği gibi eşlenik ilerlemiş ise yedek veri tabanını "nomount" moddan "mount" moda çevirilir. Sistem aktif-aktif hale getirilmiş olur.

```
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
ALTER DATABASE OPEN READ ONLY;
```

Farklı lokalde aktif-pasif eşlenik yedek alınması

Felaket kurtarma merkezlerin genel yapıları bu şekilde tasarlanır. Ana sistemin aktif iken yedek sistem yapılan değişiklik loglarını alır ancak direk kendi veri dosyalarına etki ettirmez (warm backup). FKM'de bulunan sunuculardan hizmet alınması için ek komut çalıştırılması gerekmektedir.

Aktif – pasif yedek alma tekniği aktif – aktif yedek alınmasından farkı yedek veri tabanın yalnız okuma modunda açılmaması olarak düşünülebilir.

Yüksek erişilebilirlik için bu üç mimarinin de bir arada bulunması gerekmektedir. Ayrıca bu üç mimarinin yıllık belli dönemlerde testlerini yapılması gerekmektedir. Bu testler ile sistemsel erişilebilirlik ayakta ve istenen hizmeti verebildiğinden emin olmak anlamına gelecektir. Bu testlerden biri veri tabanı yön değiştir (switch-over) işlemidir.

Örnek Uygulama

Veri tabanı yön değiştir işlemi (switch-over) şu şekilde yapılmaktadır.

1. Adım: Birincil veri tabanını önce kapatılıp sonra restrict modunda açıp yedek moduna çevirme işlemidir.

```
srvctl stop database -d <veri tabanı adı>
sqlplus / as sysdba << EOF
 shutdown immediate
 startup restrict
alter system archive log all;
```

```
alter database commit to switchover to physical standby with session shutdown
wait;
exit
EOF
```

2. Adım: Sonra yedek sunucu önce kapatılır sonra restrict modunda ana sistem olmaya hazırlanır.

```
sqlplus / as sysdba << EOF
alter database recover managed standby database disconnect;
exit
EOF
sleep 5
sqlplus / as sysdba <<EOF
alter database recover managed standby database cancel;
alter database commit to switchover to primary with session shutdown wait;
shutdown immediate
exit
EOF
srvctl stop database -d <veri tabani adi>
sqlplus / as sysdba <<EOF
startup
exit
EOF
```

3. Adım: Birincil veri tabanı önce kapatılır sonra ”mount” modda açılır. Tamamen yedekleme moduna sahip olarak gönderilecek arşiv loglarını bekler.

```
sqlplus / as sysdba << EOF
shutdown immediate
EOF
srvctl start database -d <veri tabanı adi> -o mount
sqlplus / as sysdba << EOF
```

startup

alter database recover managed standby database using current logfile disconnect;

exit

EOF

4. Adım: En son adım olarak yedek sistemin tamamen ana sistem olacak yapılandırma komutları çalıştırılır.

sqlplus / as sysdba << EOF

shutdown immediate

exit

EOF

srvctl start database -d <veri tabanı adı>

sqlplus / as sysdba <<EOF

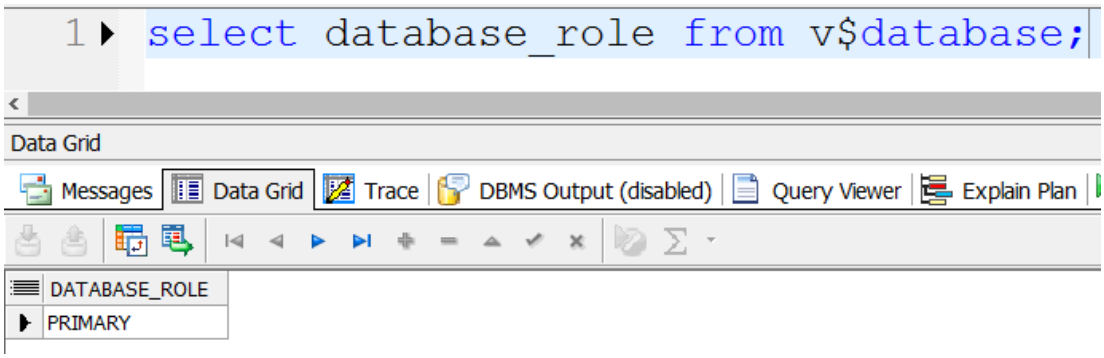
startup

alter system switch logfile;

exit

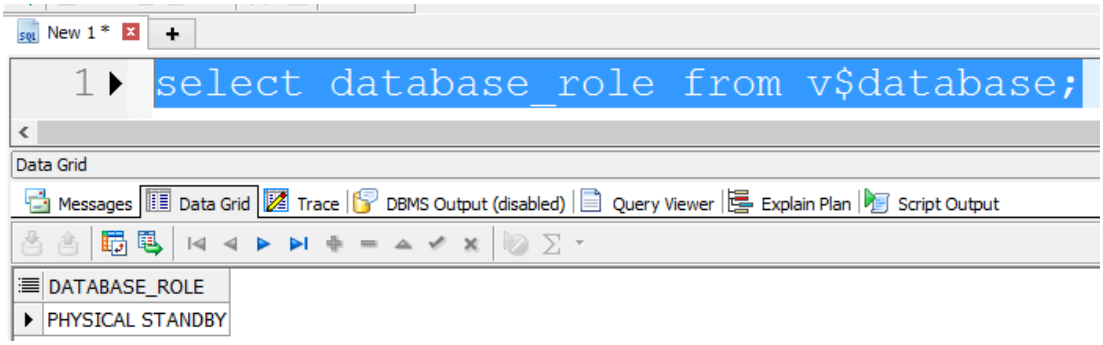
EOF

Yaptığımız yön değiştirme (switch-over) işleminin başarılı olup olmadığını iki ayrı noda da şu komutu yüksek yetkili kullanıcı olarak çalıştırmak gerekiyor.



Şekil 3.11.Veri tabanı sisteminin fonksiyonel rolünü gösteren komut (primary)

Veri tabanı rolünün ana (primary) sunucu olduğunu görülmektedir.



Şekil 3.12.Veri tabanı sisteminin fonksiyonel rolü gösteren komut (standby)

Veri tabanı rolünün yedekleme (physical standby) olduğu görülmektedir.

3.3. Veri Tabanında Güvenliğe Metodolojik Yaklaşım

Katma değer sağlayacak her hangi bir işin geçerli olunduğundan emin olunmak istenir. Sistematik olarak işleyecek bir süreci gerçekleştirmek istenirse o sürecin tüm olasılıkları önceden düşünülmelidir. Aksi takdirde ne o işten fayda sağlanır ne de kurulan sistemin geçerliliği olur.

Güven duyma mühendislik çalışmalarında aranan özelliktir. Güven duyulan sistemlerin en önemli özellikleri standartlaşmış normlar, geçerlik ve güvenirliktir [32]. Çünkü belirli normlar ortak aklın ürünüdür. Geçerlilik kazanması, belirlediğimiz metodolojinin belli aşamaları aşıp fayda sağladığını ve sonuçta güvenebileceğiniz bir yöntem olduğunu gösterir. Güvenlik ise güvenilmiş, güvenirliği sağlanmış bir yapının devamlılığı ile alakalı olduğu belirtilebilir. Gerekliliği belirlenmiş, çözüm sağlanmış ama politika haline getirilmemiş bir süreç güvenli denilemez. Güvenlik, belli standartlar çerçevesinde politika belirleme ve uygulama işlemlerin bütünüdür.

Günümüzde üretilen her tekniğin standartları konurken o tekniğin kullanıldığı sistemde güvenliğin sistemin tümüne olan etkisi hesaba katılmalıdır. Güvenilir olmayan hiçbir tekniğin anlamı yoktur. Veri tabanı yönetmenleri VTYS güvenliğine bakışları bu çerçevede olmalıdır.

Amerika'da 30 Haziran 2002 yılında SOX (Sarbanes–Oxley) olarak bilinen kanun çıkartılmak zorunda kalınmıştır [33]. Bu kanun, finans sektöründeki usulsüzlüklere karşı alınan önlemdi. Halka açık tüm şirketler artık tüm süreçleri için prosedürel yaklaşımlarını, risklerini ve kontrollerini belirtmek zorundaydı ve tüm raporlar bağımsız kuruluş tarafından kamuyla paylaşılmaktaydı. Hatta bu kanun ile gönüllü denetçiler (whistle blower) olarak bilinen tüm usulsüzlükleri anında bağımsız sisteme aktaran kişiler özel olarak korunmaktadır.

Amerika'da artık tüm veri tabanı yönetmenleri (Database Administrator) bu kanun çerçevesinde verilerin güvenliğinden birinci dereceden sorumludur. ISO 17799 standartlarına uyulmadığı gönüllü denetçileriler (whistle blower) tarafından tespit edildiğinde sonu hapse bile varan soruşturmalar açılmaktadır.

Bilinen Veri Tabanı Güvenlik Tehditleri ve Çözümleri

Saldırı atağı olmadan önce tehdidi öngörebilmeli, bunu keşfedebilmeli ve tehditten korunmalıdır. Ayrıca saldırı olduğu an kontrol sistemi, bilgi sistemini korumalıdır. Saldırı püskürtüldüğü an sistemi tekrar tehdide karşı koruma pozisyona çekmelidir. Bunun için gereken altyapı olan keşfetme, geliştirme evrelerini her zaman aktif tutan yapı güvenlik kontrol sisteminde barındırmalıdır [34]. Veri tabanının güvenliğini tehdit edebilecek zafiyetleri listelenirse [35]:

a. Aşırı Yetki Suiistimali
b. Meşru Yetki Suiistimali
c. Yetki Yükseltimi
d. Veri tabanı Platformu Güvenlik Açıkları
e. SQL Enjeksiyonu
f. Zayıf Denetimi Takibi
g. Hizmet Dışı Bırakma
h. Zayıf Kimlik Denetimi
i. Yedek Verinin Açığa Çıkması

Aşırı Yetki Suiistimali

Yapılması istenmeyen işlemleri bir veri tabanı kullanıcısının yapmasıdır. Bunun doğuracağı sorun, yetkisi olmaması gereken işlevi yerine getirerek yetki gaspı veya elde etmemesi gerek enformasyon elde etmesidir. Örneğin bir veri tabanı kullanıcısı bir tabloya erişim hakkı olabilir. Ancak o tablonun tüm sütunları görmesini istenmeyebilir. Bu ayrıştırmayı yapamaz isek veri tabanı objesi sonuçta aşırı yetki suiistimaline maruz bırakmış olunur.

Meşru Yetki Suiistimali

Verilen yetki dâhilinde amacına uygun olmayan işlevleri yerine getirerek yetkiyi kötüye kullanabilmesidir yani enformasyonu kötü amaçlarla kullanmasıdır.

Yetki Yükseltimi

Verilmiş yetkiyle yazılım tekniğini kullanarak veri tabanı yönetim tasarımın açıklarını yakalayarak yetkisini, veri tabanı yöneticisine yükseltilmesidir.

Veri Tabanı Platformu Güvenlik Açıkları

Veri tabanın kullandığı servislerin işletim sistemi seviyesindeki açıklarını kullanılarak etkisiz hale getirilmesidir.

SQL Enjeksiyonu

Bir SQL enjeksiyonu saldırısında, saldırgan genellikle güvenlik açığı bulunan bir SQL veri kanalına yetkilendirilmemiş veri tabanı yordamları ekler. Genellikle hedef veri kanalları saklanmış yordamlar veya web uygulama giriş parametreleridir. Bu eklenen yordamlar veri tabanına aktarılır ve orada çalıştırılır. SQL enjeksiyonu ile saldırganlar veri tabanına sınırsız yetki ile erişebilir.

Zayıf Denetim Takibi

Anında tespit ve kurtarma sisteminin olmaması, hesap verebilirliğin, kanıtlanabilirliğin olmaması ve caydırıcılık sisteminin olmaması ve bunların performans kaybına neden olmasıdır. Bu zafiyeti veri arşivleme, esnek işlem takibi ve sürekli raporlama ile aşılabilir.

Hizmet Dışı Bırakma

DoS (Denial of Service) atakları ile verinin okunamaması, verin bozulması ve yüksek seviye performans kaybı sonucunda iş yapılamamasıdır.

Zayıf Kimlik Denetimi

Saldırının kaba kuvvet, sosyal mühendislik ve doğrudan kimlik hırsızlığı yöntemlerle yapılmasıdır.

Yedek Verinin Açığa Çıkması

Yedeklenen veri tabanı sisteminin yedeğine yapılan saldırı ile veri hırsızlığın yapılmasıdır.

Veri tabanındaki olabilecek güvenlik zafiyetlere karşı çözüm noktasında uygulayabileceğimiz metodolojiler bulunmaktadır. Veri tabanı mimarisinde, veri tabanına bağlanmak isteyen yani oturum açmak isteyen istemci ilk karşılaşma sırasında karşısına veri tabanındaki veri tabanı kullanıcıları veya çeşitli objelere sahip şemalardan ilk talep edilecek olan şifre kontrol mekanizmasıdır. Ancak bu basit güvenlik mekanizması elbette ki günümüz veri tabanı güvenliği için yeterli olmamaktadır. Bu bağlamda ilave güvenlik parametreleri ile güvenliğin sağlanması gerektiği anlaşılmıştır. Bu bakımından erişim talebinde bulunan istemcinin erişirken kullandığı IP, makine adı, LDAP kullanıcı adı, erişirken kullandığı uygulama gibi parametrelerin bağlanmadan önce bağlanılmak istenen veri tabanı güvenlik sisteminde

kayıtlı olursa bu kayıt ile o an erişen kayıt bilgisinin karşılaştırılması sağlanacak ve bu da erişim güvenliğini bir üst seviyeye taşıyacaktır. Erişimin başarılı olması (authentication) için sisteme daha önceden tanımlı olması iç güvenlik anlamında elzemdir. Bu tanımlamadan başarıyla geçse dahi objelere yetkilendirmesi için ayrıca (authorazation) kurallar içinde hareket imkânı verilmesi gerekir. O kurallar dairesi içinde hareket kabiliyetine sahip veri tabanı kullanıcısının ayrıca her an erişim ve hareket logları çeşitli notifikasyonlarla veri tabanı yönetmenlerine iletilmesi, farkındalığı arttırması proaktif bir çözüm olarak sunulması gerekir.

Veri tabanı sistemsel yedeklenme ise reaktif bir çözüm olarak doğru metodoloji ile sağlanmalıdır. Tüm sistemin birebir yedeklenmesi, belli bir dosyaların yedeklenmesi, değişen verilerin yedeklenmesi gibi seçenekler mevcuttur. Burada farkına varmamız gereken husus; yedekleme reaktif bir çözüm olacağı yani güvenliğin bir şekilde aşılıp verilerin hasar görmesi ve sistemin bu plansız kesinti süresini (down-time) minimuma indirip bir an önce veri kaybını sıfıra indirerek sistemi tekrar ayağa kaldıracak yedekleme politikasını oluşturmaktır.

Yedeklerin tutulduğu ortamlar, güvenlik zafiyeti doğurabilir. Bu durumlar için alınan yedeğin bir şekilde şifrelenmesi ve kullanılmak istendiği an tekrar geriye dönük işlemlerle(restore) o ana geri dönülmesi (recover) sağlanması gerekir. Olabilecek her türlü felaket anında, o yedekteki bilgilere erişim işletim sisteminden bağımsız olarak kurulacak herhangi bir veri tabanı örneği ile erişilebilir hale gelmesi gerekmektedir.
Ek bir maliyet karşılığında olsa bile verilerin şifreli yedekleme seçeneği aktif edilmelidir.

BÖLÜM 4. VERİ TABANI GÜVENLİĞİNİ ORACLE VERİ TABANI YÖNETİM SİSTEMİ ÖZELİNDE İNCELENMESİ

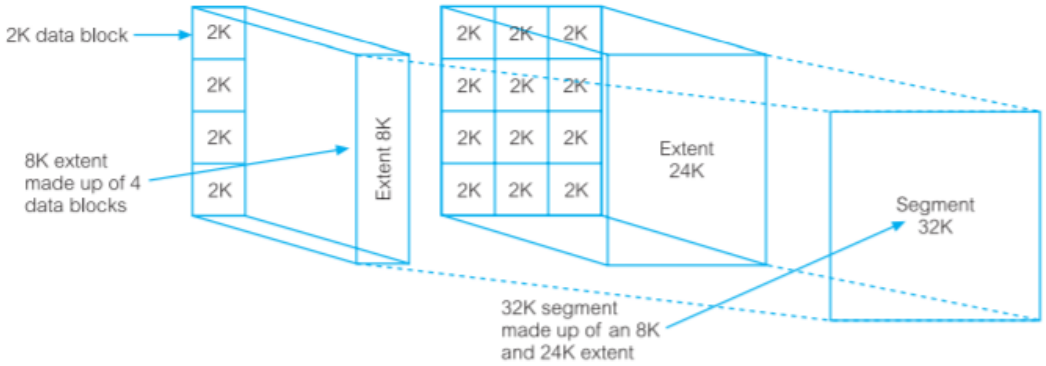
4.1. Oracle Veri Tabanı Yönetim Sistemi

Oracle veri tabanı yazılımı, veriye mantıksal ve fiziksel olarak bakar. Bu bakış fiziksel yapının mantıksal yapıya bir etkisinin olmamasını sağlar. Mantıksal bakış, tamamen kavramsal boyutta yapılan işlemleri kapsar. Fiziksel bakış, dosya yapısı ve işletim sistemine bakan tarafıyla alakalıdır.

Mantıksal yapının alt kategorileri şema, veri blokları, genişletmeler(extent), bölümlemeler (segmentler) ve tablo alanıdır(tablespace).

Oracle VTYS'in diski blok seviyesinde yönetir. Veri taşıma, silme ve güncelleme operasyonlarını mantıksal olarak Oracle veri bloğun boyutunu ölçü alarak yapar. DB_BLOCK_SIZE veri bloğun boyutunu bayt (byte) cinsinden belirleyen parametredir. Extentler veri bloğundan oluşur. Segment, belirli bir mantıksal yapı için tahsis edilmiş extentler kümesidir. Bir segmentin içindeki tüm extentler dolduğunda, Oracle dinamik olarak yeni yer tahsis eder. Extentler gerek duyulduğunda tahsis edildiğinden, segmente ait extentlerin ardışık olma zorunlulukları yoktur.

Herhangi bir yeni nesne oluştururken bir bölüm (segment) tahsis edilir. Örneğin CREATE TABLE söz dizimi ile bölüm (segment) oluşturur. Genişlemeler (extent) ise dinamik olarak oluşturulur. Bölümlerin (segment) sınırını ise tablo uzayı (tablespace) belirler. Veri bloğu, extentler ve segmentler arasındaki mantıksal ilişkisi Şekil 4.1.'de gösterilmiştir.



Şekil 4.1. Extend ve segment (Connnoly ve Begg, 2015)

Oracle yapılandırma komutları ile DB_BLOCK_SIZE parametresi ile Oracle blokları set edilir

Oracle Veri Tabanındaki fiziksel dosyalar datafile, redo log, kontrol, parametre, şifre, arşiv, uyarı ve izleme, yedekleme dosyalarıdır.

Oracle VTYS üstün özelliklerinin en iyi analiz edildiği medya bellektir. PGA (Program Global Area) ve SGA (System Global Area) bellek yönetimin anahtar bileşenleridir. Bunlar ilk kez veri tabanı yüklenirken varsayım ayarlarıyla tahsis alanı oluşturulur. Buradaki ayarlara, verim alınabilecek değerler verilirse sistemin performansı artacaktır.

Veri Tabanı Tampon Bölgesi (Database Buffer Cache): Veri bloklarının tutulduğu bölgedir. Verinin bellekte olması en maliyetli işlemdir. Onun için bu bölgenin yönetimi için çok çeşitli algoritmalar üretilmiştir. Verinin bu bölgedeki pozisyonları okunmuş ya da okunmamış olabilir. Okunmuş olma ihtimaline erişim(hit) oranı denir. Ayrıca okunup değişen veriler bu bölgelerden diske henüz yazılmamış olabilir. Sık kullanılan verileri bellekte tutmak diske erişimi azalttığı için performansa pozitif katkı sağlar.

Hareket Kayıt Log Tampon Bölgesi (Redo Log Buffer): Veri tabanında yapılan değişikliklerin tutulduğu bölgedir. Tüm veri tabanı hareket logların (transaction)

tutulduğu geriye kurtarmada çok kritik görevi olan bellek bölgesidir. Bu bellek bölgesi redo kütüklerine yazılır.

Paylaşımlı Havuz (Shared Pool): Tüm sorgu cümlelerin (query) hafızaya alındığı (cache) bölgedir. Sorgular bu alanda pars edilir. Performans açısından bu bölgenin yoğun kullanılması tavsiye edilmektedir. Bu bölgenin verimli kullanmanın yolu ise parametrik sorgulama tekniğini kullanmaktır.

Geniş Havuz (Large Pool): Yedekleme ve kurtarma için Oracle veri tabanın kullandığı bölgedir. RMAN işlem parçacıkları (prosess) bu bölgede görevlidir.

Java Havuzu (Pool): Java program dili için ayrılmış bölgedir. Java prosedürlerinin kullanıldığı bölgedir. Eğer veri tabanı üzerinde Java uygulaması çalışıyorsa bu alanın boyutu set edilebilir. JVM dediğimiz sanal makinenin java kodlarını derlediği ortam burasıdır.

Akan Veri Havuzu (Stream Pool): Oracle VTYS akan veri(stream) bileşenlerin kullanıldığı bölgedir. Bu bileşenler kullanılacaksa Paylaşımlı Havuz'un (Shared Pool) yüzde onu kadar tahsis edilmelidir.

Oracle VTYS açılış anında, işletim sistemden devralacağı bellek miktarını SGA_MAX_SIZE parametresi ile belirler. Oracle SGA'in bileşenlerin otomatik olarak o anki kullanım oranlarına göre set edilmesini istiyorsak SGA_MAX_SIZE değeri ile SGA_TARGET eşitlenmesi gerekir. Böylelikle veri tabanı yöneticisinin müdahale edeceği parametrelerin sayısı azalmış olur. Yoğun kullanılan bir yapıdır. Sonuç olarak veri tabanı yönetmenin sürekli bu parametreleri takip etme gibi bir şansı olmayabilir.

PGA (Program Global Area) : Her kullanıcı isteği için işlemci tarafından heap dediğimiz bölgeden dinamik olarak yer ayrılan alandır.

Eğer veri tabanı örneği doğrudan erişim (dedicated) modda konfigüre edilmiş ise PGA kapsamında UGA'yı da içerir. Sıralama, Bitmap birleştirme ve sağlama (hashing) için

kullanılan alandır. Oracle VTYS tarafından otomatik yönetilen bu alan istenirse elle de yönetilebilir.

Ancak elle ayarlama (tuning) dikkat edilmesi gereken titiz çalışmadır. Gereğinden fazla ayırma olursa beklenmeyen kullanıcı trafiğinde tüm bellek alanı daralıp işlem göremez noktaya gelebilir. Onun için otomatik ayarlanma modunun tercih edilmesi tavsiye edilmektedir.

Ayarlama için sadece maksimum boyut bilgisini PGA_AGGREGATE_TARGET parametresine atayarak her kullanıcı prosesi için otomatik olarak PGA alanı verilmiş olur.

Oracle VTYS, veri tabanı işlemine (proses) iki türlü bakar: Kullanıcı tarafından yapılan işlemler, bir de Oracle'ın kendi yönettiği işlemler.

Kullanıcı İşlemleri: VTYS'e bağlanmayı başarabilmiş her tür kullanıcı işlemlerin veri tabanı sunucusu ile iletişim kurmasını sağlayan işlemlerdir.

Oracle'ın kendi yönetim işlemleri: Kullanıcının isteklerine cevap verirken kullanıcı işlemleri Oracle VTYS işlemlerini çağırır. Oracle işlemleri yine Oracle sunucusunun kodunu çalıştırarak oluşturulabilir. Bu tür işlem ise iki tür işlemi barındırır: Sunucu işlemleri, arka plan işlemleridir.

Sunucu İşlemleri: Oracle sunucu işlemlerini kullanıcının veri tabanı anına bağlanmasını sağlar. Sunucu işlemleri SQL komutunu pars edip çalıştırıp uygulamaya gönderilmesinden, diskteki veri dosyasıyla SGA arasında köprü işlemlerinden ve işlemler hakkında uygulamaların detaylı bilgi sahibi olabilmesinden sorumludur.

Arka Plan İşlemleri (Background Prosess) : Performansı en üst seviyeye getirmek ve aynı anda birçok kullanıcıya hizmet verebilmek için Oracle, sistemin arka planında çalışan işlemleri kullanır.

Bu işlemler şunlardır:

a. VT Yazıcısı (Database Writer- DBWn)
b. Log Yazıcısı (Log Writer LGWR)
c. Değişme Noktası (Checkpoint- CKPT)
d. Sistem Analizi (System Monitor-SMON)
e. İşlem Analizi (Process Monitor-PMON)
f. Yedekleyici (Archiver-ARCn)
g. Geri Kurtarıcı (Recoverer-RECO)
h. Dağıtıcı (Dispatcer-Dnnn)
i. Kilit (Lock-LCKO)
j. İş Kuyruğu (Job Queue-SNPn)

4.2. Oracle Veri Tabanında Güvenlik Uygulamaları

4.2.1. Kimlik doğrulama

Veri tabanından bağlantı talep eden her uygulama ilk olarak kimlik doğrulama işlemine tabi tutulur. Bu bağlamda öncellikle Oracle VTYS kullanıcının şifresi üzerine güvenlik politikalarını belirleyebilecek tekniğe sahiptir. Kullanıcının şifresi yenilenebilir olması, şifreye ömür verilebilmesi, şifrenin karmaşık olmasını zorlaması, şifre tarihini tutulabilmesi, hatalı şifre giriş sayısını denetleyebilmesi gibi özelliklerle kullanıcı şifre yönetimini yapabilmektedir.

Oracle kullanıcıların şifrelerinin karmaşık olmasını zorlayacak bir fonksiyonu varsayılan olarak sunmaktadır. Bu fonksiyon Oracle VTYS kurulum yapıldığı ($ORACLE_HOME/rdbms/admin/utlpwdmg.sql) dizinde bulunmaktadır.

Bu fonksiyonu kurumsal ihtiyaçlarına göre revize edilebilir. Bu fonksiyonun (VERIFY_FUNCTION) içeriğinden birkaç kontrol noktaları aşağıda gösterilmiştir [36].

```
  -- Kullanıcı adı ile şifre aynı mı diye kontrol edildiği kod parçası
IF NLS_LOWER(sifre) like '%'||NLS_LOWER(kullanici)||'%' THEN
    raise_application_error(-20001, ' Şifreniz kullanıcı kodunu içeriyor! ');
END IF;
  -- Şifrenini uzunluğu 10 hane mi diye kontrol ediliyor
IF length(sifre) < 10 THEN
     raise_application_error(-20002, 'Şifre uzunluğu 10 haneden kısa!');
  END IF;
--  Eski şifreden en az 3 hane farklı olması kontrol ediliyor
IF eski_sifre = '' THEN
     raise_application_error(-20004, 'Önceki şifre boş');
END IF;
  differ := length(old_password) - length(sifre);
IF abs(fark) < 3 THEN
     IF length(sifre) < length(eski_sifre) THEN
       m := length(sifre);
     ELSE
       m := length(eski_sifre);
     END IF;
     fark := abs(fark);
     FOR i IN 1..son LOOP
        IF substr(sifre,i,1) != substr(eski_sifre,i,1) THEN
          fark := fark + 1;
      END IF;
     END LOOP;
IF fark < 3 THEN
        raise_application_error(-20004, Şifreniz eski şifrenizden en 3 karakter farklı
olmalı');
END IF;
END IF;
   RETURN(TRUE);
END;
```

Oracle ürünlerinde şifrelemeyi sağlayan paketler vardır. Bu paketler ile tabloları, tabloların sütunlarını şifreleyebilinmektedir. Veri dosyalarını (datafile) da şifreleme imkânı vardır. Bu sayede fiziksel olarak da verilerimizin güvenliğini sağlamış oluyoruz.

Oracle VTYS kullanıcıların şifrelerini 12c sürümü ile sağlama(hashlerken) yaparken PBKDF2 ve SHA-512 kriptorografik algoritma ile şifrelemektedir. İlk olarak şifreyi PBDKF2 algoritmasına giriş(input) olarak verip daha sonra SHA-512 ile son adım olarak çıktısını(output) almaktadır.

Oracle veri tabanında kullanıcı oluştururken kullanıcıdan şifre isteme yöntemini belirleyebilmektedir.

Bu yöntemler şu şekildedir:

a. Şifreyi, Oracle VTYS koruması altında olarak tutma (Password)
b. Şifreyi, Oracle VTYS'nin yüklü olduğu İşletim Sisteminin koruması altında (External) tutma
c. Şifreyi Oracle Internet Dizininin (Oracle LDAP) koruması altında (Globally) tutma

Oracle mantık olarak şifreleri istersen ben tutayım istersen işletim sistemine bırakayım ya da herhangi bir şifre oluşturmadan erişim sağlanabilen kullanıcı oluşturayım demektedir.

Ek bilgi olması bakımından Unix İşletim Sisteminde harici(external) olarak üretilen işletim sisteminin kullanıcısına ön ek olarak OPS$ eklemektedir. Bu ön ek değiştirilebilir.

SQL>CREATE USER <kullanıcı_adi> IDENTIFIED BY EXTERNALLY;

Şeklinde kullanıcı oluşturulabilir.

Bazı yazılım uygulamaları veri tabanına bağlanırken bağlantı cümesini kodlarının içine yazmaktadır. Bu tavsiye edilmeyen kodlamadır. Her ne kadar yazılımın kodları çalıştırılabilir hale getirilince okunamasa da tersine mühendislik operasyonda kullanıcı adı ve şifresi ortaya çıkabilmesi bir güvenlik tehdididir.

Ana veri tabanında herhangi bir amaca hizmet etmeyen kullanıcılar olmamalıdır. Veri tabanı kullanıcıların şifreleri kesinlikle karmaşık olmalıdır. Eğer şema olarak veri tabanında obje oluşturmuş ise muhakkak bu objenin "CREATA SESSION " yetkisi olmamalıdır.

Veri tabanı uygulaması geliştiren yazılımlar veri tabanındaki şemanın objelerini kullanırlar. Bu şema kullanıcı gibi şifresi bulunur, bu şemanın profilinde FAILED_LOGIN_ATTEMPTS parametresi tanımlanmış ise ve bu örneğin 5 gibi değer ise aslında bu durum bir şifre tahmin saldırısına (brute-force) açık demektir.

Önlemek için bu parametre yerine SEC_MAX_FAILED_LOGIN_ATTEMPSTS parametresinin kullanılması gerekir. Bu parametre belli başarısız şifre denemesinden sonra şemayı kitlemek yerine bağlanmak isteyen istemcinin bağlantısını koparark servis dışında kalma tehlikesini önlemiş olur.

Oracle kimlik denetimde özellikle yüksek yetkili veri tabanı kullanıcıların erişimlerini denetimini, araya bir katman ekleyerek, daha arttırmıştır. Bunu Oracle Data Vault olarak tanımladığı ürün ile sağlamaktadır.

Oracle Database Vault

Oracle Database Vault, verinin erişim güvenliği konusunda (kullanıcıların hassas uygulama verilerine erişimi vb.) dinamik ve esnek erişim kontrollerini sağlayan, Oracle Enterprise sürümleriyle birlikte gelen ve veri tabanı kurulumunda isteğe bağlı olarak yer alan bir güvenlik platformudur. Veri tabanı çekirdek (kernel) katmanında çalışan bu özellik, plsql kullanılarak uygulanan güvenlik uygulamalarından çok daha

işlevseldir ve RAC(Real Application Cluster) mimarisindeki yapılarda bir şekilde kullanılmaktadır.

Oracle Database Vault özelliğine sahip bir veri tabanı şu yetenekleri vardır:

Yaptıkları iş gereği veri tabanınıza erişmesine izin vermek durumunda olduğunuz yetkili kullanıcıların, çalışanların, müşterilerin ve tedarikçilerin sadece izin verdiğiniz verilere eriştiğinden emin olmanızı sağlar. Bir veri tabanı yöneticisinin dahi kritik verilerinize (kredi, müşteri özel bilgileri, hesap detayları, personel maaş bilgileri, hesaplamalar, harcamalar, görüşme detayları gibi) erişimini engelleyebilir.

a. Veri tabanı görev ayrılığı kuralı getirir.
b. Veri tabanınızın bilginiz dışında izinsiz olarak değiştirilmesini engeller.
c. Oracle Database Vault mevcut veri tabanı ortamını korur. İstenilmeyen programların bağlanması engellenebilir. Veri tabanını istenilmeyen ataklardan korur.
d. Gerçek zamanlı olarak anlık kontroller eklemenizi, değiştirmenizi ve takibini yapmanızı sağlar.
e. Veri tabanındaki objeler üzerinde (alter, drop, truncate vb.) veya veri içeriğinde yapılacak değişiklikler (insert, delete, update vb.), nasıl, ne zaman, ne ile yapılacakları belirlenip kısıtlanabilir.
f. Oracle Database Vault, uygulamaları ve verileri korumak için çok güçlü güvenli bir ortam sağlar.

Oracle Database Vault görev ayrılığı özelliği, veri tabanı üzerinde veri tabanı yönetimi, hesap ve güvenlik yönetimi olarak üçe ayrılır. Güvenlik yöneticisi (Security Administration, DVOWNER), güvenlikten sorumlu kişi aynı zamanda Oracle Database Vault'un da sahibidir. Veri tabanındaki bütün güvenlik operasyonundan sorumludur. Fakat uygulama verilerine erişme hakkı yoktur. Hesap yöneticisi (Account Management, DVADMIN), kullanıcı hesaplarını oluşturma, silme ve değiştirme yetkilerine sahiptir. Veri tabanı yöneticisi (Database Administration, DBA)

ise, yedekleme/iyileştirme, yama uygulama ve performans yönetimi gibi görevleri yürütür.

4.2.2. Yetkilendirme

Yetkilendirme, giriş yapan veri tabanı kullanıcısı veri kaynağından nasıl faydalanacağını belirleyen kurallardır. Profillerle kaynağı, rollerle nesneleri yetkilendirebiliriz. Kaynakları kullanıcı bazlı yetkilendirmenin en baştan sınırlayabilmenin yolu profillerdir.

Profil tanımlama ile kullanıcın neler yapılabileceği tanımlanır. Kaynakları kısıtlanmış kullanıcının Oracle VTYS ilk girişte tahsis edilen profilindeki sınırlandırmanın kullanıcının performansa azaltıcı yönde etkisi olacağı bellidir.

Kullanıcı profilinde yapılabilecek sınırlandırmalar; aynı anda aynı kullanıcının kaç oturum açabileceği, kaynağından daha fazla tüketim yaptırılamayacağı gibi listelenebilir.

Rol ise nesneler üzerindeki hareketlerini yetkilendirir.

Oracle tanımlanan yetkileri oturum boyunca saklamaktadır. Kullanıcılar için yaptığınız yetki değişikliği oturumun tekrar başlatıldığında etkili olmaktadır. Bir rolü bir başka role ya da kullanıcıya atayabilmek için GRANT ANY ROLE yetkisine yada bu role WITH ADMIN OPTION ile sahip olunması gerekmektedir.

Bir prosedür veya fonksiyon çalıştırılmak istendiğinde Oracle VTYS yetki anlamında iki yöntem kullanır. Bunlar prosedürü tanımlayanın yetkisi ve prosedürü çağıranın yetkisi.

Prosedürü tanımlayanın varsayılan modda hangi yetkilere sahip ise o kadar yetki ile ilgili prosedürü kullanabilir. Ancak çağıranın hakları olarak bir prosedür çağırıldığında ise hem tanımlanan yetkiler kullanılır hem de prosedürün içinde yetki alanını daha da

geniş kullanabilir. Bunun için prosedürün yazılımına AUTHID CURRENT_USER eklenir.

Prosedür veya fonksiyonun çalıştırılması sırasında bu şekilde ayırt edilmesi bize ayrı bir katman güvenliği sağlar.

Örnek bir uygulama yapılırsa iki prosedür tanımlıyoruz. Bunlar tanimlayici_test ve cagiran_test olarak.

```
CREATE OR REPLACE PROCEDURE tanimlayici_test
 AUTHID DEFINER IS
 BEGIN
  FOR kayit IN (SELECT table_name FROM dba_tables)
  LOOP
   dbms_output.put_line(kayit.table_name);
  END LOOP;
 END tanimlayaci_test;
 /
```

DBA_TABLES tablosundan sorgu çekmek DBA rolü gerekir bu prosedürü oluşturan kullanıcıda DBA rolüne sahiptir. Bu prosedürü DBA yetkisine sahip olmayan biri çalıştırabilir.

```
CREATE OR REPLACE PROCEDURE cagiran_test
 AUTHID CURRENT_USER IS
 BEGIN
  FOR kayit IN (SELECT table_name FROM dba_tables)
  LOOP
   dbms_output.put_line(kayit.table_name);
  END LOOP;
 END cagiran_test;
 /
```

Bu prosedür AUTHID CURRENT_USER ile oluşturulmuş ve DBA yetkisine sahip olmayan biri bunu derlerse hata verir.

Tablo Yetkilendirmeleri

Veri tabanı yönetmeni tablo oluşturulduktan sonra bu tablonun üzerinde işlemler yapılabilmesi için yetkilendirmesi gerekecektir. Veri maniplasyon işlemleri (DML) ve veri tanımlama işlemleri (DDL) verilecek yetkilerle belirlenir. Ekleme, silme, güncelleme ve seçme (DML) işlemleri tablo bazında yapılabilirken ayrıca ekleme ve güncelleme işlemleri kolon bazında da yetkilendirme yapılabilir.

Veri tanımlama olarak ise (DDL) değiştirme, indeks ve referans işlemleri (ALTER, INDEX, REFERENCES) tabloya yetki olarak verilebilir.

Oracle 12c sürüm ile seçme (SELECT) yetkisinin yanına okuma (READ) yetkisi de gelmiştir. Bu önceki sürümün eksikliğini gidermiştir. Yapılan işlem seçme yetkisini bölerek yetkiyi hassaslaştırmıştır. Okuma yetkisine sahip kullanıcı tabloyu özel modda kitleyemez ve güncelleme için seçemez. Okuma yetkisine sahip kullanıcı sadece tabloyu sorgular. Aslında bu şu anlama geliyor bir tablo üzerinde sorgulama yaparken okuma modunda iken tabloyu kitlemeyerek diğer kullanıcıların tabloya erişimini kapatmamış oluyoruz. Önceden tabloların kitlenmesini önlemek için bir görünüm yapılır bu görünüme seçme (SELECT) yetkisi verilirdi. Buna artık gerek kalmamıştır.

Veri tabanı yetki analizi

Yetki analizi veri tabanın belirli bir süre boyunca çalıştırılan komutlarının hangi yetkilerle ile çalıştırıldığının kayıt edilmesi ve raporlanması işlemidir. Bu işlem ile hangi kullanıcıların hangi yetkileri kullandığı ve hangi yetkileri gereksiz olduğu görülebilir.

Yetki analizi için takip edilecek metodoloji şu şekildedir:

1. Adım: DBMS_PRIVILEGE_CAPTURE paketinin CREATE_CAPTURE prosedürü tanımlanır.
2. Adım: Aynı paketin ENABLE_CAPTURE prosedürü çalıştırılır. Böylelikle izleme başlamış olur ve yeterli bir süre raporlanabilmesi için beklenir.
3. Adım: Aynı paketin DISABLE_CAPTURE prosedürü çalıştırılarak izleme bitirilir.
4. Adım: Aynı paketin GENERATE_RESULT prosedürü ile ilgili görünümlerden sorgulama imkânı doğar ve kullanıcıların yetkilerinin kullanım oranları incelenebilir hale gelmiş olur.

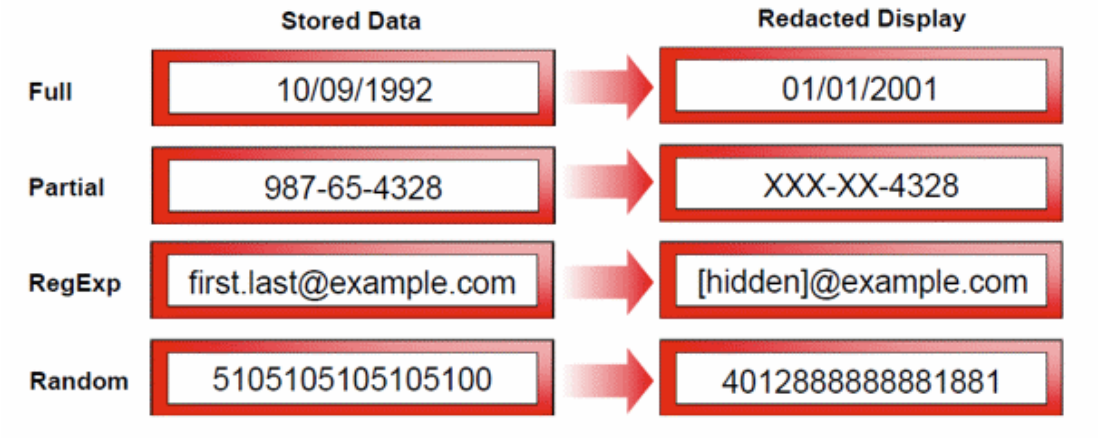
Bu metodoloji ile ileride meşru veya aşrı yetki aşımına neden olacak yetki probleminin önceden tespit edilip proaktif işlemlerle daha güvenli hale getirilmiş olacaktır.

4.2.3. Erişim kontrolü

Veri tabanında güvenlik aşamalarından ilki kimlik kontrolünü idi. Bağlantı kuran kullanıcının yetkileri kontrolü yöntemleri ise daha sonraki aşama idi. Şimdi ise yetkileri olan kullanıcın, veriye erişimin kontrolünü hassaslaştıracak yöntemlerden bahsedilecektir. Kritik uygulamalar için hangi metodoloji ile sağlanması gerektiğini incelenecektir.

Verinin redaksiyonu

Veri tabanındaki hassas verilerin korunması için değişik algoritmalar geliştirilmiştir. Bunlardan biri de verinin redaksiyon edilmesidir. Verinin kendi yerinde yani diskte herhangi bir fiziksel olarak değişiklik yapılmadan bellekte yapılan değişikliktir. Veriyi sunarken formatlamak yazılım uygulamalarında süregelen işlemlerdendir. Ancak buradaki formatlama uygulama sunucusuna gelmeden(on the fly) yapılan işlemlerdir. Bu ürün Oracle Gelişmiş Güvenlik paketinde olan bir opsiyondur.



Şekil 6.1. Oracle VTYS redaksiyon çeşitleri (Zaballa,2017)

Şekil 6.1.'den de anlaşılacağı üzere dört redaksiyon şekli bulunmaktadır. Bunlar:

a. Tam (full)
b. Parça (partial)
c. Kurallı ifade (regular expression)
d. Rastgele (random)

Bu tekniği veri tabanında uygulayabilmek için DBMS_REDACT.ADD_POLICY paketinin var olması gerekmektedir. Ayrıca güvenlik paketinin politikaları ENABLE pozisyonuna getirilmesi gerekmektedir [37]. Bir kullanıcının bu paketi kullanabilmesi için EXECUTE yetkisine sahip olması gerekmektedir. Politikayı uygulayan prosedür tanımı şu şekildedir:

```
DBMS_REDACT.ADD_POLICY (
Nesnenin_semasi          IN VARCHAR2 := NULL,
Nesnenin_adi           IN VARCHAR2,
İlgili_kolon_adi           IN VARCHAR2 := NULL,
Redaksiyon_politika_adi        IN VARCHAR2,
Fonksiyon_tipi          IN BINARY_INTEGER := NULL,
aciklama              IN VARCHAR2,
aktiflik              IN BOOLEAN := TRUE);
```

Örneğin tam redaksiyon yapmak istersek kod şu şekilde olabilir:

```
BEGIN
 DBMS_REDACT.ADD_POLICY(
   object_schema    => 'banka',  -- banka şemasının
   object_name      => 'personel_maas',  -- personel_maas tablosuna
   column_name      => 'maas_yuzde_prim', -- yüzdelik prim oran kolonuna
   policy_name      => 'redact_maas_yuzde',  -- politikanın adı
   function_type    => DBMS_REDACT.FULL, -- politikanın tipi
   expression       => '1=1');
END;
/
```

Böyle bir redaksiyon politikasını test etmek istersek:

```
SELECT maas_yuzde_prim FROM banka.personel_maas;

MAAS_YUZDE_PRIM
---------------
0
```

Tam redaksiyon olduğu görünmektedir.

Sanal özelleşmiş veri tabanı (Virtual private database)

Oracle veri tabanı güvenlik sistemi, verileri kolon bazında da gizlenebilecek hale getirebilmektedir. Bu teknik literatürde Sanal Özelleşmiş Veri Tabanı (Virtual Private Database) olarak geçmektedir. Aslında ayrı bir veri tabanı olarak düşünülmemelidir. Burada amaç kullandığın veri tabanı aynı ancak ihtiyaca özel hale getirilmiş sanallaştırılmış bir veri tabanı olarak düşünülmesinin gerektiğini belirtmek amacıyla bu isim verilmiştir.

Bu teknik veri tabanında kullanılan veri manipülasyon işlemleri için (INSERT, UPDATE, DELETE, SELECT) sonuna WHERE şartını yazılım kodlamasından bağımsız eklemektedir. Yani bir bakıma yazılımların kodlama bakım maliyeti oluşturmadan güvenlik oluşturulabilmektedir. Örneğin;

```
SELECT * FROM PERSONEL;
```

Sorgusunu çalıştıran uygulamada kullanıcının yöneticisinin bilgilerini görme imkânı vermek istemezsek bunu VPD ile sağlayabiliriz.

Bu sorguyu çağıran oturumun komutuna yazılımdan bağımsız WHERE PERSONEL_TIP <> 'YONETICI' eklenebilir. Böylelikle aslında uygulama kullanıcısı şu komutu çağırmış olacaktır.

```
SELECT * FROM PERSONEL WHERE PERSONEL_TIP <> 'YONETICI'
```

Bu tekniği kullanabilmek için DBMS_RLS paketinin kullanabilir durumda olması gerekiyor ve bu komutu EXECUTE edebilme hakkına sahip olması gerekiyor. Bu paketin DBMS_RLS.ADD_POLICY prosedürünü çağırılması gerekiyor. Bu prosedürün tanımı şu şekildedir:

```
DBMS_RLS.ADD _POLICY(
Nesneye_sahip_sema        VARCHAR2,
Nesnenin_adi          VARCHAR2,
Politika_grup         VARCHAR2,
Politika_adi          VARCHAR2,
Semanin_fonksiyonu       VARCHAR2,
politikanin_fonksiyonu       VARCHAR2,
Durum_tipi       VARCHAR2,
Güncellendi_mi          BOOLEAN,
aktiflik            BOOLEAN,
Sabit_politika_mi         IN BOOLEAN  FALSE,
```

```
Politika_tipi          IN BINARY_INTEGER NULL,
long_predicae        IN BOOLEAN  FALSE,
İlgili_sutunlarin_kosul    IN VARCHAR2,
İlgili_sutun   IN BINARY_INTEGER NULL);
```

Bu tanımı uygulamadan önce tanımda geçen fonksiyonu hazırlıyoruz. Örneğin şöyle ki:

```
CREATE OR REPLACE FUNCTION yonetici_personel (
W_sema IN VARCHAR2,
W_yonetici_id,
 w_obje IN VARCHAR2)
RETURN VARCHAR2 AS
kosul VARCHAR2 (200);
BEGIN
W_yonetici_id=’10’;
 gorev_id:=w_yonetici_id';
Koşul:=görev_id;
RETURN (kosul);
END yönetici_personel
/
```

Sonra bu fonksiyonu kullanacak VPD prosedürünü oluşturuyoruz. Şu şekilde:

```
BEGIN
 DBMS_RLS.ADD_POLICY (
 object_schema     => 'banka',  ---- veri tabanı şeması
 object_name       => 'personel', -- şemanın objesi
 policy_name       => ‘yonetici_gizle_politika’--- politikanın adı
 policy_function   => 'yonetici_personel, --- etkileyecek olan fonksiyon
 sec_relevant_cols => 'personel_ad,personel_soyad); --- hangi kolonlar
---etkileyecek
END;
```

Böylelikle BANKA şemasının PERSONEL tablosundaki PERSONEL_AD ve PERSONEL_SOYAD sütunların yapılan sorgularda ayrı bir "WHERE" şartı ile veri saklama olayını aktif hale getirilmiş ve yazılım katmanından bağımsız veri tabanı objelerine ekstra güvenlik uygulanabilir hale gelmiş oldu.

Veri maskeleme

7 Nisan 2016 tarihli resmi gazete de Kişisel Verileri Koruma kanunun yayınlanmasından itibaren tüm kurumların bir kat daha önem vermesi gereken kurumsal veri ile anonimleştirilmemiş verilerin ayrıştırılması ve izinsiz veri kullanımını engellenmesinin sağlanması yasal zorunluluk hale gelmiştir. Bu bakımından kurumların sahip oldukları veri tabanlarındaki verilerin tutulması kadar gerektiği an silinmesi veya kamuoyuna ilan edilmesi belli koşullara bağlanmış oldu [38] .

Veri tabanı sistemleri için bu noktada veri maskeleme çok daha önem kazanmıştır. Oracle VTYS veriyi yine Oracle ürünü olan Enterprise Cloud Control ürünü ile sağlamaktadır. Özellikle kurumların test yapılarındaki kullandıkları veri ana sistemlerindeki yapılardan farklı olması gerekmektedir.

4.2.4. Denetleme

Regülatör kurumların otoriter olarak tanımladığı ilkeler ile artık veri sahipliği ve verinin inkâr edilmezliği kavramları ile denetim kavramı (audit) veri tabanı yönetim sisteminde çok kritik duruma gelmiştir. İz kayıtların takibi veri tabanı güvenlik sürecinde önem kazanmıştır.

İz kayıtların takibi ile

a. Veri tabanında şüpheli hareketler sorgulanabilir ve kimlik bilgileri inkâr edilemez şekilde elde edilebilir
b. Veri tabanı yönetmenler için şeffaflık sağlar.

c. Yetkisiz işlemler tespit edilebilir.
d. Uygulamaların daha performanslı çalışması için log verileri elde edilebilir.

Nesne Denetimi

Piyasadaki güçlü veri tabanı yazılımı sunan Oracle; denetim mekanizması için çözümler sunmaktadır. Denetim(Audit) olarak tanımlayacağımız üç tip denetleme vardır:

a. Veri tabanın denetlenmesi
b. Değer tabanlı denetleme
c. Fine-grained denetleme

Veri tabanın denetlenmesi: Veri tabanında her türlü objenin erişim bilgilerini kayıt altında tutan prosedürüdür. Oracle Veri tabanın bu uygulamasından faydalanmak için AUDIT_TRAIL olarak bilinen yapılandırma dosyasında set edilme gereği vardır.

SQL> show parameters audit

Söz dizimi kullanıldığında audit_trail parametresinin son halini getirecektir. Audit_trail parametresi olabilecek dört değeri vardır: None, Db(extend), Os ve Xml(extend). Bu değerlerden NONE veri tabanı yönetim sisteminizde denetim (audit) mekanizmasının olmadığını gösterir. DB seçmeniz bütün denetimleriniz SYS şemasının DBA_AUDIT_TRAIL görünümünde (view) kayıt altına alınacağını belirtir. OS olarak set etmeniz halinde ise tüm erişim işlemleri işletim sistemi seviyesinde tutulacağını gösterir. Burada gözlemlendiği gibi Oracle Yazılımı denetlemeyi tablo bazında ve işletim sisteminin dosya bazında kayıt altında tutmaktadır. Aşağıdaki komutu çalıştırdığımızda dba_audit_trail görünümünde artık tanımlanan kullanıcı takip edilebilir hale gelecektir.

SQL>AUDIT SELECT ANY TABLE BY <KULLANICI_ADI> BY SESSION;

Aşağıdaki komutlarda artık belirtilen şemanın tablosundaki her türlü güncelleme ve silme operasyonu takip edilebilir.

SQL> AUDIT UPDATE, DELETE ON <ŞEMA_SAHİBİ>.<TABLO_ADI>

Ancak bazen çalıştırılan sorguyu da takip etmek isteyebiliriz. Veri tabanında yönetici konumdaki kullanıcı standart olarak tanımlanan DBMS_FGA.ADD_POLICY prosedürünü güvenlik ekibince belirlenen strateji uygun oluşturulur. Bu prosedürü yetkili kullanıcı olarak derlenmesiyle prosedürün taşıdığı politika uygulamaya geçmiş olur.

Veri tabanı yönetim sistemi denetlemeye uygun hale getirildikten sonra denetlenmek istenen objeler için kural konfigürasyonu çalıştırılarak veri tabanındaki yüksek yetkili SYS kullanıcısının AUD$ tablosuna iz verileri gelmeye başlayacaktır.

Günümüzde artık iz kayıtları için farklı bir veri tabanı sunucusu oluşturulmaktadır. Ana veri tabanında çok hızlı büyüyen verileri tutmaktan ise denetleme işleminde sorumlu ayrı bir katman oluşturulmaktadır.

Oracle VTYS için bu ürün Oracle Audit Vault ve Database Firewall, Oracle veri tabanı, işletim sistemi, dosya sistemi veya diğer veri tabanlarından toplanan tüm denetim verilerinin esnek ve kapsayıcı bir şekilde izlenmesine olanak sağlayan bir güvenlik çözümüdür. Aynı zamanda Oracle Database Firewall ağ üzerinde ilk korunma mekanizmasını sağladığı için ağdan beklenen hataları, SQL enjeksiyonu ve veri tabanındaki diğer zararlı durumları engelleyebilir. Oracle Audit Vault and Database Firewall birçok veri tabanından gelen denetim (audit) verisini bir merkezde toplayabilir ve bunlardaki SQL trafiğini, yetkisiz erişimleri ve SQL politika ihlallerini aynı zamanda gösterebilir. Denetim verilerinin bir merkezde toplanması IT giderlerinin düşmesine de yardımcı olur. Oracle Audit Vault and Database Firewall Oracle veri tabanı, Microsoft SQL Sunucu, IBM DB2 for LUW, SAP Sybase ASE, Oracle MySQL veri tabanları ve Oracle Big Data Appliance gibi ürünleri destekler.

Audit Vault denetim (audit) verileri için merkezi ölçeklenebilir, güvenli bir depolama alanı sunar. Audit Vault raporlama, uyarma (alert) ve veri tabanı politika yönetimi (policy management) için ideal bir platformdur. Ajanlar (agent) kullanarak izleme(audit) verisi hedef sisteme aktarılabilir. Audit Vault bütün veri tabanlarından çektiği denetim verisini konsolide edebilir.

Oracle Database Firewall ağ katmanına kurulumu yapılan ve yönetilen bir üründür. DBFW'a korunması istenen veri tabanlarını gösterebilir. Bu veri tabanları Oracle, Sybase, DB2, SQL Server veya MySQL (5.1 versiyonu ile) olabilir. DBFW, ağ katmanından veri tabanına doğru gitmekte olan SQL sorgularının analiz, izleme, bloklama, değiştirme ya da direkt olarak veri tabanına iletme görevini üstlenmektedir. Geliştirilen özel yapısı sayesinde SQL Injection saldırılarını rahatlıkla anlayabilmekte, bu tipte saldırıları bloklayabilmekte ve istenildiği takdirde belirli alıcılara mail olarak gönderebilmektedir. SQL Injection saldırılarını SQL'in yapısını inceleyerek anlayabilmektedir. Çok yüksek düzeyde veri tabanı güvenliği sağlamaktadır. Oluşturabildiği güvenlik politikaları ve listeleriyle veri tabanına iletilmesi istenen ya da istemeyen herhangi bir sorgu rahatlıkla tanımlanabilmektedir. Aynı zamanda gerçek zamanlı raporlama ve bloklama özelliğine de sahip bir üründür. In-Line ve Out of Band olmak üzere iki tip DBFW koruma yöntemi bulunmaktadır. In-line koruma seçeneğinde bloklama ve izleme, out-of-band koruma seçeneğinde ise yalnızca izleme özelliği kullanılabilmektedir. In-line bağlantı için kurulacak DBFW ile korunacak veritabanının IP-briged bir ağ yapısına sahip olması gerekmektedir. Bu sayede veri tabanına gelmekte olan sorgular DBFW tarafından önce incelenir ve uygun görülürse veri tabanına gönderilir. Bu işlemin zaman kaybını son kullanıcı asla anlamaz (milisaniyelerle ölçülen bir eşleştirme söz konusudur). Out-of-Band koruma seçeneğinde ise switch üzerinde SPAN portu tanımlanır ve DBFW'un bulunacağı Linux makineye IP Mirroring yapılır. Veri tabanına giden bütün SQL sorguları aynı zamanda DBFW'un olduğu sunucuya da gönderilir. Bu durumda bloklama özelliği devreden çıkar; ancak anlık raporlama ve analiz işlemleri devam eder. Bu sebepten dolayı Oracle tam koruma için DBFW - Audit Vault - Database Vault üçlüsünü tavsiye etmektedir. In-line korumanın bir diğer etkili özelliği ise gelen SQL sorgularının yapısını değiştirebilmesidir.

4.2.5. Veri Şifreleme

Veri tabanı dosyalarında veri varsayılan olarak açık(clear-text) yani okunabilir halinde durur. Bu veri tabanı güvenliği açısından riskli bir durumdur. Bu durum için uygulama katmanından izole edilmiş olacak şekilde veri dosyaların şifrelenmesi gerekiyor. Veri dosyaları şifrelendiği için yedekleri de şifreli olarak saklanacaktır.

Oracle veri tabanı yönetim sisteminde bu işlem için Transparan Veri Şifreleme (Transparent Data Encryption (TDE)) kullanılmaktadır.

Şifrelemeyi kolon bazında veya tablo uzayı bazında yapılabilmektedir. Kolon bazlı şifrelemede ilgili kolonun yabancı anahtar(foreign key) olmamasına ayrıca indeks tanımlanamayacağına dikkat edilmelidir. Bu bakımlardan tablo uzayının şifrelenmesi daha performanslıdır.

Şifreleme için 3DES168 algoritması kullanılmaktadır. Tablo uzayın şifrelenmesinde ise AES128 kullanılmaktadır. Kolon bazında ise AES192 kullanılmaktadır.

Bu metodolojide dikkat edilmesi gerekenler şunlardır:

Şifreli tablo uzayı, sıkıştırılmış olabilir. Böyle bir durumda öncelikle sıkıştırılmanın açılması sonra şifreleme yapılması gerekir. TDE performansı ise Oracle bu tekniğini donanım destekli yapmaktadır. Oracle buna tümleşik özelleştirilmiş mühendislik sistemi demektedir (engineer system appliance). Yani hem donanımı hem yazılım kısmını varsayılan olarak mühendislik disiplini ile entegre edip ürünlerini sunmaktadır. Bu bakımından şifrelemenin ek performans maliyetini donanımsal destekle aştığını belirtir.

Ayrıca kolon şifreleme tekniğini kullanırken ilgili kolon çok güncelleme yapılmadığına dikkat edilmelidir. Örneğin anne kızlık soyadı kritik bir veridir. Bir kere girilir yani güncellenmesi nadirdir. Bu kolon şifrelenebilir. Ancak şifrelenecek kolon

yabancı anahtar veya indeksli ve çok sayıda kolon şifrelenmesi gerekiyorsa mecburen tablo uzayı şifrelenmesi gerekiyor.

4.2.6. Yedekleme

Yedekleme, veri tabanı güvenliği konseptinde reaktif bir çözümdür. Yani saldırı sonrası minimum kayıpla en hızlı şekilde yedekten dönülmeyi öngörülmektedir.

Oracle veri tabanı yedeklemeyi farklı bakış altında incelemektedir. Yedeklemeyi işletim sistemi seviyesinde, RMAN adındaki Oracle kendi dosya yönetimi ile (fiziksel yöntemlerle) ve mantıksal yöntemlerle gerçekleştirebileceğini belirtmektedir.

İşletim sistemi seviyesinden yüklü bulunduğu işletim sistemin anladığı komutları yürüterek kendi dosyaları ve nesnelerini yedekleyebilmesidir. Bu yedekleme iki şekilde alınabilir: Veri tabanı servislerinin kapalı olduğunda veya açık olduğunda alınabilir.

Veri tabanı servisleri kapalı iken yedekleme

Klasik kopyalama mantığı ile durdurulan veri tabanın fiziksel olarak bulunduğu dosyaları, veri dosyaları, kontrol dosyaları, parametre dosyaları gibi dosyaların bir başka disk ortamına kopyalanmasıdır(hard copy). Yalnız burada dikkat edilecek nokta veri tabanı açıldığında Oracle dosya servisleri dosyaları fark edebilmesi için OFA uyumlu dizin adlandırılması ve yerleşimi gerekmektedir.

Veri dosyaların yeri: ORACLE_BASE/oradata/<SID>/ altında bulunmalı.

ORACLE_BASE: Oracle VTYS sisteme kurulduğunda varsayım olarak Oracle yazılımın yüklü olduğu (binary) dosyalarının bir önceki klasör adıdır. Oracle yazılımın dosyaları kurulduğu yer ORACLE_HOME dur. SID, Oracle yazılımın kurulduğu sunucunun(host) adıdır.

Veri tabanı açık olarak yedekleme

Veri tabanı servisleri açık iken çevrimdışı kopyalama (hard copy) yapılamaz. Çünkü ilgili dosyalar anlık işlem görmektedir. Bundan dolayı Oracle mühendislerin artı hizmet üretmeleri ile bazı özel komutlarla veri tabanı açık iken (online) yedek alınabilir hale gelebiliyor. Bunun için ilk şart veri tabanın ARCHIVELOG modunda çalışıyor olması gerekiyor. Daha sonra tek tek veri dosyalarının mantıksal düzeydeki karşılığı olan tablo uzaylarının modunu yedeklemeye çekilmesi gerekir.

ARCHIVELOG*:* Oracle veri tabanın açılış aşamalarında set edilebilen modudur. Bu parametre set edilebilirliği opsiyoneldir. Oracle veri tabanın "NOMOUNT" aşamasında

ALTER DATABASE ARCHIVELOG;

Komutunun çalıştırılması ile arşivleme modu açılmış olur.

Kontrol dosyaları ve parametre dosyaların yeri: ORACLE_BASE/admin/<SID>/ dizinde bulunur.

ALTER TABLESPACE <tablo uzayının adı> BEGIN BACKUP;

Komutun çalıştırılması ile yedekleme modundaki ilgili veri dosyaları işletim sistemi seviyesindeki gibi(hard copy) kopyaları alınabilir. Bu işlem bittikten sonra bunun bittiğini Oracle VTYS

ALTER TABLESPACE <tablo uzayının adı> END BACKUP;

komutu ile bildirilir. Bu işlem tüm veri dosyaları için tek tek uygulanır. Sonra kontrol dosyalarının kopyalanması gerekir. O işlemi de

ALTER DATABASE <SID> BACKUP CONTROLFILE TO
<kopyalanacağı_dizin_adı>

ile bu işlem de tamamlanır.

RMAN yazılım aracı ile yedekleme

Fiziksel yedekleme işlemini Oracle VTYS gerçekleştirmesini Recovery MANagement kısaltması olarak RMAN yazılımı gerçekleştirmektedir. RMAN veri tabanın tümünü veya tek tek tablo uzayını, kontrol dosyalarını, arşiv logları yedekleme için kullanabilir. Blok seviyesinde yedekleme yeteneği olduğu için blokları değişenleri sadece değişen kısımları yedekleme gibi ayrı sunulan opsiyonları vardır(INCREMENTAL). Ayrıca boş alanları yedeklememesi yedeğinde performanslı yapı sunduğunun kanıtıdır. Oysa işletim sistemi seviyesinde boş alanlarda gereksiz yere yedeklenmekteydi. Ayrıca bu işlemler için tablo uzayların modunu değiştirme gibi zorunluluğu yoktur.

RMAN veri tabanı düzgün kapatılmamış ise yedeğini INCONSISTENT biçimde alabilir. Yalnız yine RMAN ile kurtarma gerektirir. Kurtarma kataloğu RMAN programının veri sözlüğüdür. Kataloğun sürekli senkronize edilmesi gerekir. Bunu sağlayan komut:

RMAN> RESYNC CATALOG;

RMAN veri tabanı yöneticisinin el ile tek tek yapmak zorunda olduğu yedekleme işlemlerini otomatik yapmayı sağlar. Örneğin veri tabanın yedeğinin düzenli olarak alma işlemi, herhangi bir arıza anında arşiv logları uygulayıp sistemi kurtarma işlemini RMAN otomatik yapabilir.

RMAN işletim sisteminden bağımsız bir araçtır. RMAN'ın komut yapısı ikiye ayrılır. Tek başına çalışan komutlar (stand-alone) ve grup içinde çalışması gereken komutlardır.

Grup içinde çalışan komutlara örnek aşağıda incelenebilir:

```
Run
{ /* rman blok komut kümesi başlangıç komutu*/
Allocate channel c1 type disk;  /* ilgili diske kanal açılıyor */
Restore database; /o kanaldan belirlenen o noktaya hazırlan */
Recover database; /* belleğe getirilen o noktadaki yedeklemeyi uygulama
}
```

Yedekleme politikası kurtarma senaryosuna göre belirlenir.
Bunun faktörleri:

a. Veri tabanı kurtarmak için gerekli olan ortalama süre
b. Ne kadarlık veri kaybına tolere edilebilirliği
c. Yedekleme süresi

Yedekleme planında esas olan veri tabanı en kısa sürede kurtarabilme üzerine olmalıdır. Eğer veri tabanı sıfır kesinti ile çalışması gerekiyorsa sadece veri tabanı yeterli olmayabilir. Ayrıca replikasyon, paralel sunucu gibi yöntemlerin olması gereklidir. Oracle ARCHIVELOG mod kesintisiz yedek alma imkânı sunan tekniğini kullanılmalıdır.

Herhangi bir anı yüzde yüz kurtarılabilmesi için gerek ve yeter şart o andan önceki arşiv logların ve en yakın yedeğinin sistemde sağlam olması gerekir. Sistemde var ise kayıpsız şekilde en hızlı olabilecek şekilde o anı kurtarılabilir. Eğer yedeğin alma tarihi oldukça eski ise ve arşiv loglar da büyük ise bu kurtarma hızına negatif etkileyecektir. Yedeklemede nelere dikkat edilmesi gerektiğini aşağıda özetlenmiştir:

a. Veri tabanın ne kadar kesintiye tahammülü varsa ona uygun yedekleme süresini belirleyebilirsin.
b. Tablo uzayı eklenip veya silinirse, veri dosyası eklenip silinirse yani yapısal her hangi bir değişiklikte o işlem yapılmadan önce yedeği alınmalıdır.

c. Logların tutulmasına yeniden başlamadan(reset) önce hemen sonra veri tabanın tam bir yedeği veri tabanı kapalı iken alınmalıdır.
d. Arşiv logların kaybolması veya bozulması ihtimaline karşı arşiv loglar birden fazla yere yazılmalıdır.
e. Eski yedekler saklanmalıdır. Bazen yeni alınan bir yedek herhangi bir nedenle bozulabilir veya kaybolabilir. Daha eski yedek gerekebilir. Veri tabanı kapalı iken veya açık iken kurtarma gereği olabilir.

Kurtarma işlemi veri tabanı hayat döngüsünü (incarnation) son güncelleme tarihine getirmektir. Bu işlemi yaparken Oracle SCN (System Change Number) adı verilen bir sayı ile belirler. Bu bilgi kontrol dosyalarında tutulur. Bu sayıyı ise LGWR prosesi her log anahtarlama (log swich) işlemi sırasında redo log dosyalarına yazar. Kontrol dosyası ne kadar güncel ise veri tabanı hayatı o kadar günceldir.

Redo log dosyaları her türlü işlem proseslerin(transaction) nasıl muamelede bulunulduğu, onaylanıp (COMMIT) ya da geriye alındığı (ROLLBACK) türü bilgileri saklar. Rollback segmentte geriye alınan veriler tutulur. Kurtarma sırasında onaylanmamış değişikler “rollback segmentle “eşleşip geriye dönüşü sağlanabilir.

Elektrik kesintisinin ani olması durumda bellekteki verilerin diske aktarımı olamayacaktır. Ancak Oracle veri tabanı otomatik olarak veri tabanı örneğinin açılması sırasında bellektekileri kurtarma tekniğine sahiptir. Buna veri tabanının otomatik kurtarması (instance recovery) denir. Veri tabanı yöneticisi sadece bu durumu ALERT loglardan durumu anlayabilir.

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Yapılan tez çalışmasında, veri tabanı dünyasında kullanılan temel kavramlar ışığında veri tabanın ne olduğu, nasıl hizmet verebildiği, veri tabanın nasıl modellendiği incelenmiştir. Veri tabanı üzerinde çalışan performansın, erişilebilirliğin ve güvenliğin metrikleri tespit edilerek, değerleri incelenmiştir.

Yüksek performans, erişilebilirlik ve güvenlik için uygulanabilir metodolojiler ve yapılması gerekenler, metodoloji başlığı altında incelenmiştir.

Yüksek performans için uygulanabilir metodolojiler için şu sonuca varılmıştır:

Veri tabanını yönetirken, yüksek performans için öncelikle veri tabanı sisteminin kendi metrikleri üzerinden istatistik toplanması gerekmektedir. İstatistik toplanması için veri tabanında bir zamanlanmış iş, sistemin yoğun olmadığı bir dönemde çalıştırılması ile toplanan veriler yüksek yetkili kullanıcın tablolarında biriktirildiği görülmüştür. Biriktirilen veriler Otomatik Performans Teşhis İzleme (ADDM) yazılımı ile yorumlanabilecek rapor şekline dönüştürülmüştür. ADDM, yapay zekâ metotları aracılığıyla tavsiyelerde bulunabildiğinin farkına varılmıştır. Önerdiği tavsiyelerden bazıları şunlardır: donanım değişikliği, başlangıç parametrelerin tekrar ayarlanması, tabloları bölümleme, indeks oluşturma, yüksek sayıda zincilenmiş tablo verileri birleştirme, sorguları iyileştirmedir.

Yüksek erişilebilirlik için gereken uygulanabilir metodoloji için şu sonuca varılmıştır:

Erişilebilirliğin bir oran, güvenirliliğin ise olasılık olmasından güvenirlik içinde sürekliliği yükseltilmesi sistem mimarisine güven vermektedir. Bu bağlamda, sürekliliğin alt yapısı güçlü olasılığa sahip ise bu sistem güvenli denilebilmiştir.

Erişilebilirliği yükseltebilecek mimari yapılar, tezde ispat edilmeye çalışılmıştır. Anlık veri kaybına tahammülü olmayan sistemler için maksimum erişilebilirlik ve koruma tercih edilmesi gerekirken, az miktarda veri kaybı karşılığında yüksek performans talep eden yüksek erişilebilirlik için aktif – aktif hem aynı lokalde hem de farklı lokalde yedeklenen sistemlerin kurulması ayrıca aktif – pasif olarak bir üçüncü yapının var olması ve bu üç mimarinin de bir arada bulunması gerektiği gözlenmiştir. Ayrıca bu üç mimarinin yıllık belli dönemlerle testlerini yapılması gerektiği anlaşılmıştır. Sonuçta bu testler ile sistemsel erişilebilirlik ayakta ve istenen hizmeti verebildiği kontrol edilmiş olacaktır.

Veri tabanı sisteminizi metodolojilere göre yüksek performansa getirebilirsiniz. Ancak bunun sürekliliğini sağlayabilmeniz için yani yüksek erişilebilirliğe ulaşabilmeniz için sistemin performans metriklerini takip edilmesi gerektiği ortaya koyulmuştur. Belirlediğiniz eşik(treshold) değerlerine göre uyarı verebilen notifikasyonlara sahip mekanizmaların veri tabanı yönetmenleri için elzem olduğu görülmüştür.

Oracle veri tabanı yönetim sistemi ve Oracle VTYS özelinde uygulanabilir metodolojiler için şu sonuca varılmıştır:

Veri tabanında güvenlikte oluşabilecek tehditler öngörülebilmesi, keşfedilebilmesi ve tehditlerden korunabilmesi gerektiği anlaşılmıştır. Özellikle veri tabanın iç mimarisinde oluşabilecek tehditlerin farkına varılmış ve bu tehditler için proaktif ve reaktif çözüm yöntemleri için metodolojik çalışma yapılmıştır.

Veri tabanı dünyasında çözüm ortakları arasında endüstriyel marka olan Oracle Veri Tabanı Yönetim Sistemi, tezde özel olarak incelenmiştir. Mimarisinden bahsedilerek bazı kavramlar anlaşılır hale getirilmiştir. Bazı Oracle parametrelerin kavramsal olarak ne anlama geldiğinden bahsedilmiştir. Böylelikle uluslararası bir markanın yaklaşımını görerek bir veri tabanı yönetim yazılımında olması gerekenleri tespit edilmiştir.

Oracle veri tabanı yönetim sisteminde veri tabanı güvenliği olarak ne gibi teknoloik çözüm getirdiği araştırılmıştır. Kimlik doğrulamada, yetkilendirmede, erişim kontrolünde, şifrelemede ve denetlemede proaktif sunduğu metodolojik çözümler ayrıntılı incelenmiştir. Reaktif olarak güvenli yedekleme sistemini üzerinde çalışma yapılmıştır.

Kimlik doğrulamada şifrelemenin daha karmaşık oluşturulması gerektiği sonucuna varılmıştır. Ayrıca Oracle Database Vault ürünü ile kullanıcı oluşturma işlemini bir ara katman aracılığıyla yapılması gerektiği ortaya çıkmıştır. Böylelikle yüksek yetkili kullanıcıların hareket alanı daraltılmış oldu.

Yetkilendirme ile kimliği doğrulanan kullanıcının veri tabanı objeler üzerindeki işlem kabiliyeti kontrol altına alınmış oldu. Veri tabanı objelerinden fonksiyon ve prosedür üzerinden yetkilendirme işlemlerinde AUTHID DEFINER ve AUTHID CURRENT_USER parametrelerle yapılabileceğini gösterilmiş oldu.

Erişim kontrolü yönteminde verinin redaksiyonu incelenmiştir. Bu noktada tam, parça, kurallı ifade ve rastgele veri redaksiyonları ifade edilmiş ve tam redaksiyon detaylandırılmıştır. Sanallaştırılarak özelleştirilen veri tabanı tekniği ile veriye erişimin kontrolü arttırılmıştır. Böylelikle kimliği doğrulansa da veri tabanına erişimin daha kontrollü hale getirilmiş oldu. Özellikle test sistemlerinde kullanılmak üzere veriyi maskelemeden bahsedilmiştir. Gerçek verinin diskte tutuluş şekli tamamen değiştirilerek gerçek veriden farklı hale getirerek kurumların test mekanizmalarında iç tehditlerine karşı koruma sağlanmış olacaktır.

Veri tabanını denetleme mekanizmasında hangi tip denetlemelerin olduğu gösterilmiştir. Sorguları da barındıracak şekilde denetim için iz kaydını tutan mekanizmaların oluşturulmasının gereğinden bahsedilmiştir. Oracle VTYS geliştirdiği Oracle Audit Vault ve Database Firewall ürünü ile tüm denetim mekanizmaları sağlanabildiği ortaya konulmuştur.

Oracle VTYS'de kullanılan Transparan Veri Şifreleme ürünü ile veri şifrelemenin nasıl yapıldığında ortaya çıkarılmıştır. Şifrelemeyi kolon bazından tablo uzayına kadar yapılabildiği gösterilmiştir. Yedeklerin açık (clear-text) olarak durmasının sakıncalarından bahsedilmiş ayrıca performansa da olumlu etkisi olduğu gösterilmiştir.

Oracle yedeklemede RMAN tekniğinin üstün özellikleri ile verinin yedeklenmesinde oldukça etkili yöntemlere sahip olduğu görülmüştür.

Veri tabanlarında yüksek performans, erişilebilirlik ve güvenlik için bazı öneriler şu şekildedir:

a. Yüksek performans ve erişilebilirlik için mümkün olduğu kadar az sistemin yeniden başlatılması,
b. İstatistikler için tüm altyapının hazır olması,
c. Tüm metriklerin takip edilebileceği raporların alınması günlük, haftalık, aylık şeklinde periyodik olarak raporların genişletilmesi,
d. Performans metriklerindeki eşik değerleri aştığında çeşitli yollarla notifakasyon üretilmesi,
e. Otomatik Performans Teşhis İzleme (ADDM) gibi akıllı uygulamaların önerilerinin dikkate alınması
f. Donanımsal veya yazılımsal güncellemelerde sistemin planlı kesintiye uğrama sürelerin mümkün olan en kısa sürede olması hatta sıfır olması için gereken test sistemlerin, yedek sistemlerin oluşturulması
g. Sistemlerin kesinklikle aktif-aktif, aktif-pasif olarak hem aynı hem de farklı merkezlerde birebir aynı yedeklere sahip olması
h. Yüksek erişilebilirlik için oluşturulan sistemlerin periyodik olarak yön değiştir (switch-over) işlemi gibi işlemlerle yılda en az 2 defa test edilmesi
i. Veri tabanında oluşturulan kullanıcıların şifreleri için profil ayarlarını güçlü ve şifre kompleks fonksiyonu ile şifrelerin kompleks olmaya zorlamasını
j. Veri tabanı dünyasında bilinen (SYS, SYSTEM) yüksek yetkili veri tabanı kullanıcıların kitlenmesi

k. Yüksek yetkili veri tabanı kullanıcıların yetkilerini ayrıştırıp ilgili kurumların veri tabanı yönetmenlerine adreslemesinin sağlanması
l. Oracle Database Vault gibi uygulamalarla kullanıcıların veri tabanına erişimini ve yetkileri için ayrı güvenlik katmanın oluşturulması
m. Oracle Veri Redaksiyon gibi uygulamalarla verilerinin gereksiz veya yetkisiz erişimlere karşı önlem alınması ve bellekte karartma tekniği ile erişim kısıtlaması
n. Oracle Özelleştirilmiş Sanal Veri tabanı ile yazılım uygulamalarından bağımsız verilere filtre konulması
o. Oracle Veri Şifreleme ile yedeklerin şifrelenmesi
p. Oracle Veri Maskeleme ile test sistemlerinin oluşturulması
q. Verilerin denetlenmesi için mümkün olan en geniş parametrelerle Oracle Audit Vault ve Database Firewall uygulaması ile ayrı sunucuda log toplanması
r. RMAN gibi özel uygulamalarla efektif yedek alınması ve yedeğinin şifrelenmesi

sağlanarak veri tabanı yönetim sistemlerinin yüksek performans, erişilebilirlik ve güvenli yapıya geçişi için alt yapısı uygun olacağını tezin sonucunda önerilmiştir.

KAYNAKLAR

[1] Fries M, Vrinda S, Bilien C, Maximizing Database Performance, Societte Generale, San Francisco, 2016.

[2] Kish D, Brian L, Market Trends: Database Security, Worldwide, 2017.

[3] Deepika N.S, Database Security: Threats and Security Techniques, International Journal of Advanced Research in Computer Science and Software Engineering, 2015.

[4] Montesino R, Fenz S, Information Security Automation: How far can w ego? ,Sixth International Conference on Availability, Realiability and Security, 2011.

[5] Sağıroğlu Ş, Canbek G, Bilgi ve Bilgisayar Güvenliği: Casus Yazılımlar ve Koruma Yöntemleri 2-5, Grafiker Ltd, 2006.

[6] Floridi L, Semantic Conceptions of Information, Stanford Encyclopedia of Philosophy, 2015.

[7] Stephen R, Plew R, Begining Databases, Pearson Education, 2003.

[8] Sudarsan S, Korth H, Silberschartz A, Database System Concepts, The McGraw-Hill Companies, 2001.

[9] Çağıltay N, Tokdemir G, Veri Tabanı Sistemleri Dersi Teoriden Pratiğe, Ada Matbaacılık,4-11,26-28, 2010.

[10] Yarımağan Ü, Veri Tabanı Sistemleri, Akademi Yayıncılık,24-25, 2010.

[11] Su C, Xie C, Salt: Combining ACID and BASE in D.Databases, OSDI, 2014.

[12] Bocij B, Business Information Systems, Prentice Hall, 2003.

[13] Önder E, Yönetim Bilişim Sistemleri Kapsamında Web Tabanlı İllişkisel Veri Tabanı Yönetim Sistemleri ve Bir Uygulama, İstanbul Üniversitesi Sosyal Bilimler Enstitüsü İşletme Ana Bilim Dalı, 2005.

[14] Öztürk S, ATMACA Hatice E, İlişkisel ve İlişkisel Olmayan (NoSQL) Veri Tabanı Sistemleri Mimari Performansının Yönetim Bilişim Sistemleri Kapsamında İncelenmesi, Bilişim Teknolojileri Dergisi, Cilt:10 Sayı 2, 2017.

[15] Zemzans O, Exploring NoSQL Databases, MAMK, 2016.

[16] Eken S, Kaya F, Doküman Tabanlı NoSQL Veri Tabanları: MongoDB ve CouchDb yatay ölçeklenebilirlik karşılaştırılması, Kocaeli Üniversitesi, 2015.

[17] Öztürk E, Mesut A ve Diri B, NoSQL veri tabanlarında kullanılan veri sıkıştırma yöntemlerinin performans analizi, Trakya Üniversitesi, 2016.

[18] Liu H.H, Oracle Database Performance and Scalability, Wiley, 2012.

[19] Schumacher R, Oracle Performance Troubleshooting with Dictionary Internals SQL&Tunning Scripts, Rampart Tech Press, 2003.

[20] http://www.cmg.org/publications/measureit/2006-2/mit33/measureit-issue-4-07-itil-availability-management-beyond-the-framework-by-james-yaple.,, Erişim Tarihi: 24.04.2018.

[21] White Paper, Best Practices for Gathering Optimizer Statistics with Oracle Database 12c Release 2,Oracle, 2017.

[22] Kumar S, Oracle Database Performance Tuning Guide, 11G R2, Oracle,2009.

[23] Lee Y, Database Performance Evaluation Methodology for Desing Automation Systems: A case Study of MICON, Canegie Mellon Departmant of Electrical and Computer Engineering, 1992.

[24] Bryce J, Oracle Database Performance Tuning Advanced Features and Best Practise for DBAs, Brillix,2017

[25] Zahn M, The Secrets of Oracle Row Chaining and Migration, Information Technology, 2007.

[26] Simpkins C, Database Normalization, Georgia Tech College of Computing, 2014.

[27] http://www.yasinkaplan.com/tr/docs/HA.pdf., Erişim Tarihi: 24.04.2018.

[28] Türkan A, Güvenilirlik Analizinde Kullanılan İstatiksel Dağılım Modelleri, 2007

[29] Avelar W, Mean Time Between Failure: Explantion and Standarts White Paper 78, 2011.

[30] Highleyman W, Calculating Availability – Redundant Systems, 2006.

[31] Kim C, Oracle Data Guard 11g, Oracle Cooperation, 2011.

[32] Ebrinç S, Psikiyatrik Derecelendirme Ölçekleri ve Klinik Çalışmaları, Klinik Psikofarmakoloji Bülteni, Cilt: 10, Sayı 2, 2000.

[33] Jahmani Y, Dowling W, The Impact of Sarbanes-Oxley Act, Journal of Business & Economics Research Volume 6 -10, 2008.

[34] Türköz T, Sezer A, Oracle Veri Tabanı Güvenliği Klavuzu, Ulusal Elektronik Ve Kriptoloji Araştırma Enstitüsü Doküman Kod: BGT-5001, 2008.

[35] Yazıcı M, Sql Veri Tabanı Sistemlerinde Güvenlik ve Örnek Uygulama, Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, 2008.

[36] Akduygu Y, Oracle Veri Tabanı Güvenliği, Abaküs Yayınları, 2016.

[37] Connolly T, Begg C, A Practical Approach to Design, Implementation and Management Sixth Edition, Pearson, 2015.

[38] https://www.kvkk.gov.tr/yayinlar/Kisisel_Verilerin_Silinmesi_Yok_Edilmesi_veya_Ananonim_Hale_Getirilmesi.pdf. Erişim Tarihi : 27.05.2018

[39] Yumuşak N, Yorulmaz A, Performance Tuning in Database Systems In High Availability Architecture and Reducing Query Costs Strategies in Oracle Database Management System, International Conference on Advanced Technologies, Computer Engineering and Science, Karabük, 2018.

[40] Yumuşak N, Yorulmaz A, Veri Tabanı Yönetim Yazılımında Objelere Erişim ve Yetkilendirme Güvenliği: Oracle Veri Tabanı Özelinde İncelenmesi,3.Uluslararası Mühendislik ve Tasarım Kongresi, Kocaeli, 2018.

ÖZGEÇMİŞ

Ahmet Yorulmaz, 1980 yılında Karlıova'da dünyaya geldi. İlkokul ve ortaokulu Türkiye'nin farklı şehirlerinde okudu. Liseyi Balıkesir Ziraat Bankası Fen Lisesinde daha sonra da Bursa Malcılar Lisesinde tamamladı. Selçuk Üniversitesinde Bilgisayar Mühendisliği'nde lisans eğitimimi tamamladı. Sakarya Üniversitesi'nde 2011 yılında Bilgisayar ve Bilişim Mühendisliği Anabilim dalında yüksek lisans eğitimine başladı. 2007 yılında Türkiye Kalkınma Bankası'nın açtığı sınavı kazanarak Uzman Yardımcısı olarak Bilgi İşlem Dairesi Başkanlığında göreve başladı. 2010 yılında "Veri Tabanı Performans Çalışması" tezini başarıyla savunarak Kıdemli Uzman oldu. Yaklaşık 12 yıldır Türkiye Kalkınma Bankası'nda Bilgi İşlem Dairesi Başkanlığı'nda Veri Tabanı Yönetmenliği görevini yapmaktadır.