İlişkisel ve İlişkisel Olmayan (NoSQL) Veri Tabanı Sistemleri Mimari Performansının Yönetim Bilişim Sistemleri Kapsamında İncelenmesi

Serdar ÖZTÜRK[1], Hatice Ediz ATMACA[2]

[1]Yönetim Bilişim Sistemleri, Bilişim Enstitüsü, Gazi Üniversitesi, Ankara, Türkiye
[2] Endüstri Mühendisliği, Mühendislik Fakültesi, Gazi Üniversitesi, Ankara, Türkiye
serdarozturktr@hotmail.com, hediz@gazi.edu.tr
(Geliş/Received:14.10.2016; Kabul/Accepted:25.05.2017)
DOI: 10.17671/gazibtd.309303

Özet— Veri tabanları, veri fazlalığını kontrol eden ve veri tutarlılığını koruyan en önemli sistemlerden biridir. Veri entegrasyonu ile verilere basit şekilde ulaşılması, düzenlenmesi ve paylaşılması gibi avantajlar sağlamaktadır. Veri depoları oluşturulması sırasında yaygın olan yaklaşım, verilerin uygulama modeline yakın bir model ile saklanması üzerine kurgulanmaktadır. Bu çalışmada öncelikle ilişkisel ve ilişkisel olmayan veri tabanlarının yönetim bilişim sistemleri kapsamında öğrenilmesi ve niteliklerinin belirlenmesi ile ilgili bilgiler sunulmuştur. Sonrasında veri tabanı modelinin seçilmesi ve ölçütlerinin ortaya koyulmasıyla en uygun performans ölçümleri, veri tabanlarının avantaj ve dezavantajları ile oluşturulacak yapının uygun hale getirilmesi hakkında bilgiler verilmiştir. Daha sonra ilişkisel veri tabanı ve dağıtık veri tabanlarının performans karşılaştırması yer almaktadır. Çalışmanın son bölümünde ise veri tabanları yapıları karşılaştırmalı olarak incelenerek sonuç ve öneriler sunulmuştur.

Anahtar Kelimeler— İlişkisel veri tabanı, dağıtık veri tabanı sistemleri, ilişkisel olmayan veri tabanı, NoSQL

The Examination of Relational and Non-relational (NoSQL) Database System's Architectural Performances in Terms of Management of Information Systems

Abstract— Databases are one of the most important systems that control the exceed data and protect the data consistency. They provide advantages such as accessing data easily by data integration, arranging data and sharing them. The prevalent approach while creating databases is based on preserving data with a model that is similar to that of applying data. In this work, information about the learning of relational and non-relational database in the perspective of management information systems and identifying the attributes of them is presented primarily. Then, information about the selection of the database model and identifying the criteria, the most suitable performance measures, advantages and disadvantages of the databases and compiling the created structure is expressed. Subsequently, comparison of the performance of the relational and non- relational database systems is explained. Lastly, structures of databases are examined by comparing each other and results and suggestions are presented.

Keywords— Relational database, distributed database systems, non-relational database system, NoSQL

1. GİRİŞ (INTRODUCTION)

Bilgisayar ve iletişim teknolojilerinde yaşanan hızlı gelişim her geçen gün daha fazla organizasyonu etkileyerek farklı çözümler üretmeye zorlamaktadır. Belli başlı bir amaca ulaşmak için veri veya ham bilginin işlenerek ilgililere yarar sağlayacak biçime dönüştürülmüş hali olan bilgi, organizasyonlar tarafından sürekli daha kısa sürede erişilmek istenen en etkili faktör haline gelmiştir. Bilgisayarlar karar alma sürecinde etkin olarak kullanılarak, "bilgi sistemleri" günümüz trend konuları arasında yerini almıştır [1].

Günümüzde yaşanan bu değişim ve gelişim, verilerin modellenerek saklanmasını ve dolayısıyla veri tabanı kullanımını zorunlu kılmaktadır. Temel bir kurum rehberinden, orta ve büyük ölçekli işletmelerin kurumsal ve ticari bilgilerinin organize edilerek saklanmasına kadar farklı alanlarda veri modelleme ve depolama gerekliliği

ortaya çıkmaktadır. Verinin büyüklüğü, miktarı ve karmaşıklığı gibi etkenlere bağlı olarak farklı veri modelleme, veri depolama ve sorgulama yöntemleri geliştirilmiştir.

Bu kapsamda, okuma ve yazma gibi işlemlerin yoğun olarak kullanıldığı veri tabanlarında ilişkisel veri tabanlarının yanı sıra ilişkisel olmayan veri tabanı yönetim sistemleri de kullanılmaktadır. Performans ve esneklik özellikleri ile ilişkisel olmayan veri tabanı yönetim sistemleri (NoSQL) eBay ve Amazon gibi dünyaca ünlü şirketler tarafından tercih edilebilir hale gelmiştir [2].

Bu çalışmada "bilişim sistemleri" ve "veri tabanı" kavramları incelenerek ilişkisel ve ilişkisel olmayan veri tabanı yönetim sistemleri mimari performansının detaylı karşılaştırılması yapılmıştır.

2. BİLİŞİM SİSTEMLERİ VE YÖNETİMİ (INFORMATION SYSTEMS AND MANAGEMENT)

Bilişim sistemi, organizasyonlarda karar verme aşamasına kadar bilgiyi toplamak, düzenlemek, işlemek ve saklamak olarak tanımlanabilir. Bilişim sistemlerinde üç aktivite bilgiyi üretmek için gereklidir. Bu aktiviteler: girdi, işlem ve çıktıdır. Girdi, organizasyonun içinden veya dış çevresinden, ham bilgileri (veriyi) toplamaktır. İşlem, bu ham veriyi daha anlamlı biçime çevirir. Çıktı, işlenmiş bilgiyi (enformasyon), insanlara veya kullanılacak olan aktivitelere aktarır.

İşletmeler açısından bilişim sistemleri, herhangi bir girdiyi işlemlere tabi tutup çıktı sağlayan mekanik yapılardan daha fazla anlam ifade etmektedir. Bilişim sistemleri, bilişim teknolojileri altyapısından yararlanan yönetsel çözümlerdir. Bilişim sistemlerini etkin bir şekilde kullanmak için organizasyon, yönetim ve teknolojiye hâkim olmak gerekmektedir [3].

Şekil 2.1 Bilişim Sistemleri Bileşenleri
(Information Systems Components)

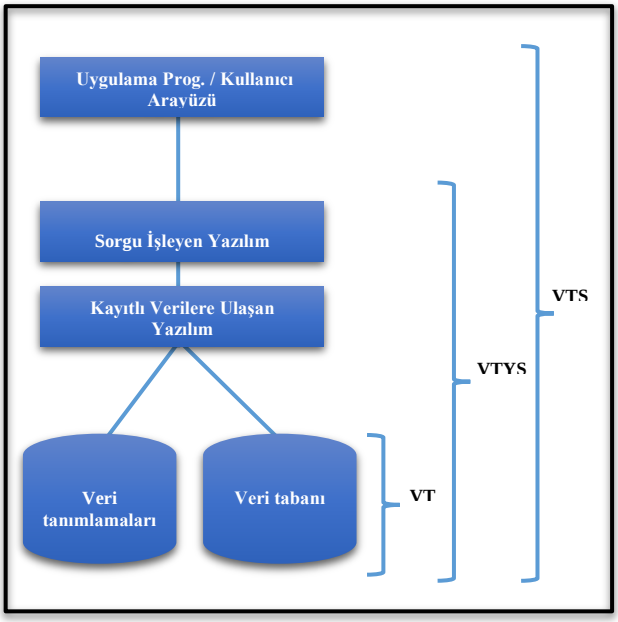
3. VERİ TABANI VE VERİ TABANI YÖNETİM SİSTEMLERİ (DATABASE AND DATABASE MANAGEMENT SYSTEM)

Veri tabanı en genel tanımıyla, kullanım amacına uygun olarak düzenlenmiş veriler topluluğudur. Birbirleriyle ilişkileri olan verilerin tutulduğu, mantıksal ve fiziksel olarak tanımlarının olduğu bilgi depolarıdır. Veri tabanları gerçekte var olan ve birbirleriyle ilişkisi olan nesneleri ve ilişkileri modeller [4].

Veri tabanı yönetim sistemleri (VTYS), verilere aynı anda birden çok bağlantı sağlayabilme özelliği sağlar. Bu sistemler, veri tabanı yönetiminin bir parçası olarak, verinin nasıl depolanacağı, kullanılacağı ve erişileceğini mantıksal olarak yönlendiren bir kurallar sistemidir.

Veri tabanı, VTYS ve uygulama programlarını ile kullanıcı ara yüzlerini içeren yapıya "veri tabanı sistemi (VTS)" denir. Veri tabanı, veri tabanı yönetim sistemi ve veri tabanı sistemi arasındaki ilişki ve işlevler şekil 3.1'de gösterilmiştir.

Şekil 3.1 VT-VTYS-VTS Arasındaki İlişki ve İşlevler
(Relation and Functions between DB-DBMS-DBS)



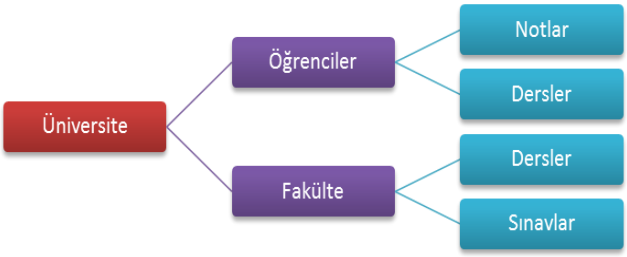
Veri tabanı modellerini sekiz kategoriye ayırabiliriz:

- Düz model veya tablo modeli: İki boyutlu veri grubundan oluşur. Sütunlarda verilerin benzer özellikleri, satırlarda ise veri grupları yer alır. Kullanıcı adlarının ve şifrelerinin tutulduğu veri tabanı buna örnek olarak verilebilir. Böyle bir veri tabanında her satırda bir kullanıcıya ait şifre bilgileri, sütunlarda ise tipleri aynı olan veriler yer alır. Düz veri modeli tek tablodan oluşan bir model olarak düşünülebilir [5].

	Ad Soyad	Kullanıcı Adı	Parola
Kayıt 1	Murat ERGİN	Mergin	kjVdb125
Kayıt 2	Ayşe YILMAZ	Ayılmaz	Bks46db7
Kayıt 3	Can TÜRK	Cturk	fhG8dbt9

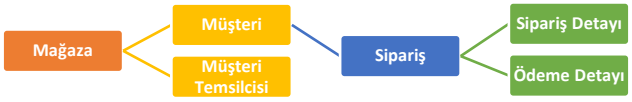
Şekil 3.2 Düz veri modeli örneği
(Instance of flat data model)

- Hiyerarşik Veri Modeli: İlk olarak 1960'lı yıllarda ortaya çıkmış ve adını veriyi depolama yönteminden almıştır. Bu veri tabanının depoladığı yapısal verilere "kayıt" adı verildi. Kayıtlar ağaç mimarisi şeklinde yukarıdan aşağı sıralanmaktadır. Kök adı verilen ilk kaydın bir veya daha çok çocuk kayıtları vardır. Çocuk kayıtlarında kendi çocuk kayıtları olabilir. Kök haricinde bütün kayıtların bir ebeveyni vardır [6].



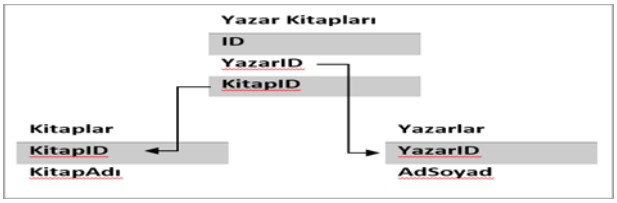
Şekil 3.3 Hiyerarşik Veri Tabanı Modeli
(Hierarchical Database Model)

- Ağ veri modeli: Bu model 1970'li yıların başında geliştirilmiştir. Hiyerarşik veri modelinin geliştirilmiş halidir. Hızlıca kabul görmesinin nedeni bir verinin doğal olarak başka veriler ile ilişkili olmasıdır. Ağ modelinin hiyerarşik modelden en önemli farkı, uç-düğüm pozisyonundaki verinin iç-düğüme işaret edebilmesidir. Böylelikle ağ modelinde bire-çok ilişkiler yanında, çoka-çok ilişkiler de modellenebilir. Bu veri tekrarını önemli ölçüde azaltır [7].



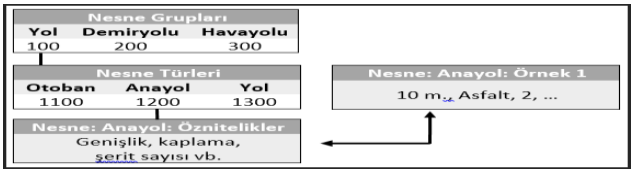
Şekil 3.4 Ağ Veri Modeli
(Network Data Model)

- İlişkisel Veri Modeli: Hiyerarşik ve ağ veri modellerinin, çeşitlenen beklentileri karşılamakta yetersiz kalması, yeni bir model arayışını başlatmış ve ilişkisel veri modeli geliştirilmiştir. E. F. Codd'un 1970'de yazmış olduğu "A Relational Model of Data for Large Shared Data Banks" makalesi ile ilişkisel veri yapılarında büyük bir ilerleme kaydedilmiştir [8-9]. İlişkisel veri modelinin temel kavramı, ilişkidir. İlişkiler yardımıyla, veri içerisindeki ilişkiler modellenir. Dolayısıyla, ilişkisel bir veri tabanı, çeşitli ilişki örneklerinden oluşur. Kavramsal olarak ilişkiler, satır ve sütunlardan oluşan iki boyutlu tablolarla karakterize edilir. Genellikle veri tabanında her tablo için bir dosya bulunur. Tablonun her satırı birbiriyle ilişkili verilerin bir topluluğudur. Sütunlarda ise nitelikler bulunur [10].



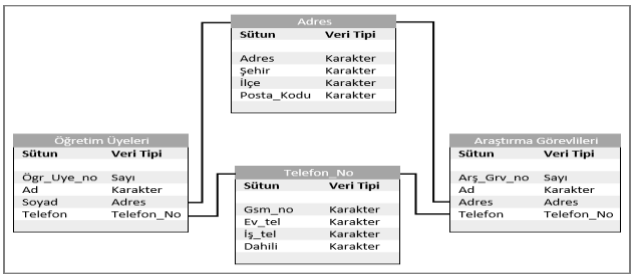
Şekil 3.5 İlişkisel Veri Modeli
(Relational Data Model)

- Nesne Yönelimli Veri Modeli: Daha sonraları ortaya çıkmış ve başarısını kanıtlamıştır. Nesne yönelimli programlamaya dayanan veri modelidir [9].



Şekil 3.6 Nesne Yönelimli Veri Modeli
(Object-Oriented Data Model)

- Nesne İlişkisel Veri Modeli: Nesne ilişkisel veri tabanı, ilişkisel işlevselliğin üzerine nesne yönelimli özellikler içerir. İlişkisel veri tabanları içinde nesne yönelimli karakteristikler içeren ilk veri tabanı 1997 yılında piyasaya sunulan Oracle8'dir.



Şekil 3.7 Nesne İlişkisel Veri Modeli [11]
(Object-Relational Data Model)

- Çoklu Ortam Veri Modeli: Çoklu ortam veri tabanları nesne ilişkisel veri tabanları ile büyük benzerlikler gösterir. Bununla birlikte, film, müzik, metin ve video gibi büyük nesneleri işlemek ve aynı zamanda işleme sırasındaki adımları kullanıcıya göstermemek için farklı özellikler taşır. Çoklu ortam veri tabanlarının desteklemesi gereken üç temel özellik; Veri miktarı, Süreklilik ve Senkronizasyondur. Çoklu ortam veri tabanı uygulaması, imge görüntüleme, uzaktan

görüntülü eğitim, üç boyutlu tıbbi görüntü kayıtları depolanması konularında özellikle tıp bilgi sistemlerinde kullanılmaktadır [6].

- Dağıtık Veri Modeli: Dağıtık veri tabanları, iki ya da daha fazla bilgisayarda depolanan ve bir ağ üzerinde dağıtılan bilgiler için kullanılan veri tabanı grubudur. Veri tabanını ağ üzerinden paralel kullanmak için parçalara ayırmak, sorguların daha hızlı işlenmesini sağlar. Böyle bir sistemde, birden fazla veri tabanına erişilmesine rağmen, kullanıcı bir tek veri tabanıyla çalışıyormuş gibi işlem yapar [6].

4. VERİ TABANI TASARIMI (DATABASE DESIGN)

Veri tabanı tasarımında; gerçeğin, gereksinim ve beklentiler çerçevesinde modellenerek veri tabanına aktarılması gerekir [10].

Veri tabanı tasarımında ilk olarak, olası veri tabanı kullanıcı gereksinimlerinin belirlenmesi gerekir. Söz konusu gereksinimler, veri tabanında yer alacak veri gruplarını, verilerin tiplerini ve verinin fiziksel olarak depolanması için kullanılacak olan veri yapılarını belirler. Gerçeğin veri tabanındaki sayısal temsili, onun belli bir perspektiften bir modeli olup, bir veri tabanı sisteminde gerek kullanıcılar ve gerekse bilgisayar tarafından anlaşılabilecek bir tarzda tanımlanması gerekir. Böyle bir tanımlama, veri tabanı literatüründe "şema" olarak adlandırılır. Kullanıcı ve bilgisayar düzeyleri sırasıyla "kavramsal" ve "fiziksel" düzeyler, bu düzeylerdeki şemalar da "kavramsal şema" ve "iç şema" olarak anılırlar. Kavramsal ve fiziksel düzeylerdeki şemalar, farklı anlayış mekanizmalarına hitap ettiklerinden, kullanılacak veri modelleri de farklı olacaktır. Her iki düzeyde kullanılmak üzere, çeşitli veri modelleri geliştirilmiştir [10].

Şekil 4.1 Veri Tabanı Tasarım Aşamaları
(Database Design Stages)

Geleneksel veri tabanı tasarımı, kullanıcı düzeyinden fiziksel düzeye doğrudur. Kavramsal tasarımda, gereksinimlere göre kavramsal şema belirlenir. Kavramsal şema tanımlamada, kavramsal ya da mantıksal veri modelleri kullanılabilir. Kavramsal şema, ortalama veri tabanı kullanıcısı için, veri tabanının yapısını genel olarak tanımlar. Kullanıcıların veri tabanının yapısını anlamalarına ve böylece uygulamalarını modellemelerini sağlar. Kavramsal şema, fiziksel depolama yapılarının ayrıntılarına girmeden, varlıklar, veri tipleri, varlıklar arasındaki ilişki tipleri ve kısıtlayıcılar üzerinde yoğunlaşır. Bu bakımdan kavramsal şema, yüksek düzeyli bir tanımlamadır. Diğer bir ifadeyle, kavramsal şema, yazılım ve donanımdan bağımsızdır ve son kullanıcı tarafından anlaşılması da daha kolaydır [10].

Kavramsal veri modelleri oldukça yüksek düzeyli olduklarından, kavramsal bir veri modelinde tanımlı bir şema genellikle doğrudan gerçekleştirilemez. Bu nedenle, geleneksel veri tabanı tasarımında, kavramsal tasarımdan sonraki adım, çoğunlukla, gerçekleştirim için kullanılacak bir veri tabanı yönetim sisteminin seçimidir. Öte yandan, bugün piyasadaki veri tabanı yönetim sistemlerinin çoğu mantıksal bir veri modeli kullanmaktadırlar. O nedenle mantıksal veri modelleri, gerçekleştirim veri modelleri olarak da bilinirler. O halde, kavramsal veri modelindeki kavramsal şema, veri tabanı yönetim sisteminin veri modelinde yeniden tanımlanmalıdır. Buradaki işlem, iki veri modeli arasında bir dönüşüm olup, bazen "mantıksal veri tabanı tasarımı" olarak anılır [10].

Fiziksel tasarım aşamasında, verinin en yüksek verim için, veri tabanında fiziksel olarak nasıl organize edilmesi gerektiği belirlenir. Sonuç, iç şemadır [10].

İç şema depolama yapılarını, kayıt formatlarını, kayıt alanlarını, veri tabanına giriş yol ve yöntemleri ile veri tabanının fiziksel gerçekleştirimini ilgilendiren diğer bütün detayları tanımlar. İç şema tanımlamada, genellikle veri yapıları olarak bilinen, fiziksel veri modelleri kullanılır. İç şema, yazılım ve donanıma bağımlıdır [10].

5. İLİŞKİSEL VE İLİŞKİSEL OLMAYAN (NoSQL) VERİ TABANI SİSTEMLERİ (RELATIONAL AND NON-RELATIONAL DATABASE (NoSQL) SYSTEMS)

5.1 İlişkisel Veri Tabanı (Relational Database System)

Günümüzde en yaygın kullanılan veri tabanı sistemlerinden biridir. Satır ve sütunların meydana getirdiği tablolardan oluşur. Bu tablolar birbiri ile ilişkileri olan tablolardır. Dolayısıyla bir veri tabanında ilişkiden söz edebilmek için en az iki tablonun yer alması ve bu iki tablodaki verilerin birbiri ili bir şekilde ilişkilendiriliyor olması gerekir. Bu şekilde ilişkisel veri tabanları, veri

tabanı denilen büyük dosyalardan oluşur. Her bir tablo, belli yapıya uygun verileri saklamak üzere tasarlanır [12].

ACID; klasik ilişkisel veri tabanı sistemlerinde sağlanan temel özellikler aşağıda sunulmuştur [13]:

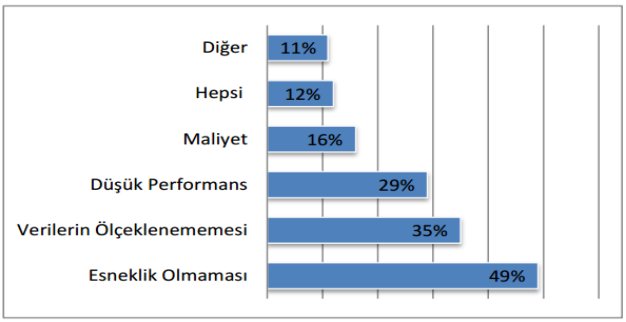
- Bölünmezlik (Atomicity)
- Tutarlılık (Consistency)
- İzolasyon (Isolation)
- Dayanıklılık (Durability)

5.2 İlişkisel Olmayan (NoSQL) Veri tabanı (Non-Relational Database System)

İlişkisel olmayan (NoSQL) veri tabanı; 1998 yılında ilk olarak Carlo Strozzi tarafından öne sürülen bir kavramdır. NoSQL, ilişkisel veri tabanı sistemlerine alternatif bir çözüm olarak ortaya çıkmıştır. İlişkisel olamayan veri tabanları yatay olarak ölçeklendirilen bir veri depolama sistemidir [14].

Dünya'da NoSQL örneklerini incelediğimizde; sosyal ağlarda Digg'in 3 TB'lık çözümü, Facebook'un gelen postaları arama için 50 TB ve eBay'ın bütün verileri için 2 PB'lık çözümleri vardır. Veri tabanlarına ilişkin problemlerden biri olan ölçek sorununa, diğer çözümlerin içinde en iyi cevap vereni NoSQL'dir [15]. Günlük 7 TB'lık işlem hacmine sahip Twitter [16] ve 10 TB'lık Facebook örneğindeki gibi, çok büyük verilerin depolanması ve yazılmasında ilişkisel veri tabanlarının eksik kaldığı hususlarda, yatay ölçekleme yapan dağıtık NoSQL çözümleri geliştirilmiştir.

İlişkisel veri tabanı kullanıcılarının, araştırmalar neticesinde NoSQL veri tabanına geçmek istemelerinin nedenleri şekil 5.1'de yüzde olarak gösterilmiştir.



Şekil 5.1 Neden NoSQL Gerekli
(Why NoSQL Necessary)

Amazon bu gereksinimi "DynamoDB", Google ise "Big Table" ismini verdiği NoSQL veri tabanı sistemi ile çözmektedir. İlişkisel veri tabanını yerine NoSQL veri tabanını tercihi, özellikle hız ve yatay büyüme ile gereksiz ek maliyetten kurtulmaya dayanmaktadır.

İlişkisel veri tabanlarının kullandığı ACID işlemselliğine karşın NoSQL "BASE" (Basically Available- Soft state-Eventually consistent) kısaltması ile ifade edilir.

- Kolay Ulaşılabilirlik (Basically Available): Veri erişim sorunlarını ortadan kaldırmak için kopyaları kullanır ve paylaşılmış ya da bölümlenmiş veriyi birçok sunucudan alır.
- Esnek Durum (Soft state): ACID mantığında veri tutarlılığının olmazsa olmaz bir gereklilik olduğu savunulurdu fakat NoSQL sistemler tutarsız ve süreksiz verilerin barınmasına da izin verir.
- Eninde sonunda Tutarlı (Eventually consistent): Uygulamalar anlık tutarlılıkla ilgili olmasına rağmen, NoSQL sistemlerin gelecekte bir zamanda tutarlı olacağı farz edilir. ACID'in zorunlu tuttuğu tutarlılığa karşın NoSQL'de tanımlanmayan bir zamanda tutarlılığın oluşacağı garanti edilir [17].

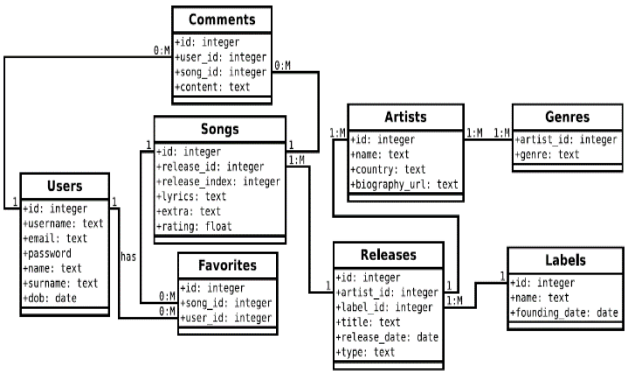
NoSQL veri tabanları göreceli olarak yeni bir gelişmedir. Fakat e-ticaret, internet arama motorları ve sosyal ağlar gibi büyük ölçekli internet uygulamaları için güvenilirliğini kanıtlamıştır. Birçok kayıt saklama teknolojisi kendilerini NoSQL ürünü olarak sınıflandırmaktadır ve her biri kendilerine özgü karakteristiklere sahiptir [18].

En bilinen lider NoSQL ürünlerinin teknik karşılaştırmaları Tablo 1'de verilmiştir [19].

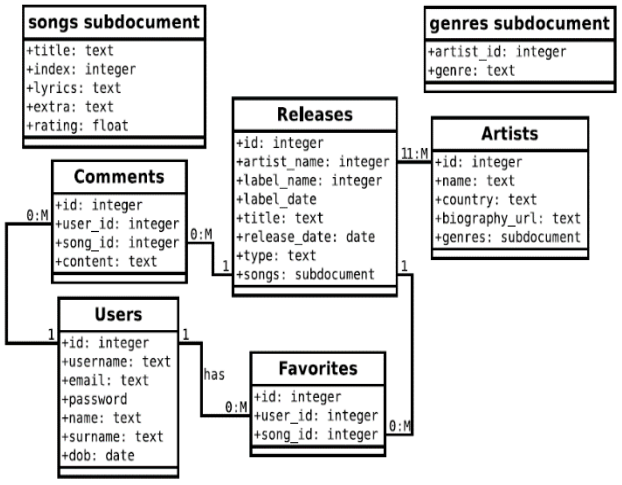
Tablo 1: Lider NoSQL ürünlerinin teknik karşılaştırması
(The Technical comparison about the leader NoSQL's products)

	MongoDB	**CouchDB**	**Riak**	**Redis**	**Voldemort**	**Cassandra**	**HBase**
Dil	C++	Erlang	Erlang	C++	Java	Java	Java
Lisans	AGPL	Apache	Apache	BSD	Apache	Apache	Apache
Model	Document	Document	Key/Value	Key/Value	Key/Value	Wide Column	Wide Column
Protokol	BSON	HTTP/REST	HTTP/REST or TCP/Proto	bufs TCP		TCP/Thrift	HTTP/REST or
Depolama	Memory mapped b-trees	COW-BTree	Pluggable: InnoDB, LevelDB, Bitcask	In memory, snapshot to disk	Pluggable: BSV, MySQL, in-	Memtable/SStable	HDFS

Esinlenilen		Dynamo	Dynamo		Dynamo	BigTable, Dynamo	BigTable
Arama	Evet	Hayır	Evet	Hayır	Hayır	Evet	Evet
Sadeleştirme	Evet	Hayır	Evet	Hayır	Hayır	Evet	Evet



Şekil 6.1 MySQL veri tabanı şeması
(Relational database schema)



Şekil 6.2 MongoDB şeması
(MongoDB schema)

6. VERİTABANI MİMARİLERİNİN PERFORMANS KARŞILAŞTIRMASI (PERFORMANCE COMPARISON OF DATABASE ARCHITECTURE)

Veri tabanı mimarilerinde oldukça bol çeşit ve bir o kadar da seçenek vardır. Bu çalışmada ilişkisel veri tabanı olarak günümüzde en yaygın kullanılan veri tabanı sistemlerinden biri olan MySQL ve ilişkisel olmayan (NoSQL) veri tabanı olarak ilişkisel veri tabanı sistemlerine alternatif bir çözüm olarak ortaya çıkan, yatay olarak ölçeklendirilen bir veri depolama sistemi olan MongoDB veri tabanı sistemi kullanılmıştır.
Yapılan çalışmada; MySQL ve MongoDB veri tabanı sistemlerinin performans ve yatay ölçeklenebilirlik incelemesi için aşağıdaki işlemlerin uygulanması ve sonuçlarının ortaya çıkarılması hedeflenmiştir.
Bunlar;

- Veri tabanı sunucu sistemleri özellikleri belirlenmesi,
- Veri tabanı şemaları oluşturulması,
- Sorguların belirlenmesi,
- Veri tabanı ayarlarının yapılması,
- Ölçümler ve ölçüm metrikleri bilgileri,
- Performans analizi ve sonuçlarıdır.

Veri Tabanı Şeması: Projede iki adet veri tabanı şeması tasarlanmıştır. Biri MySQL (şekil 6.1), diğeri ise MongoDB (şekil 6.2) veri tabanıdır. Şemalar, kendi zevk ve tercihleri doğrultusunda diğer kullanıcılara şarkılar önermek için tasarlanmış farklı algoritmalar kullanan bir müzik uygulaması etrafında modellenmiştir. Tablolar arasında herhangi bir veri tekrarını ortadan kaldırmak için normalizasyon değerlendirmesi sağlanmıştır.

Veri Tabanı Sorguları: Bu çalışmada üç farklı veri tabanı sorgusu kullanılmıştır. Birinci sorgu için sadece "SELECT" deyimi içeren basit bir sorgu hazırlanmıştır. İkinci sorgu için daha karmaşık "INNER JOIN" deyimi içeren bir sorgu hazırlanmıştır. Üçüncü sorgu için ise "SELECT" ile birlikte iç içe "JOIN", "INNER JOIN" ve "WHERE" deyimi içeren detaylı karmaşık bir sorgu hazırlanmıştır.
Sorgu 1: Basit

```
SELECT * FROM Users WHERE username = 'username '
```

Sorgu 2: Karmaşık

```
SELECT ' Favourites. song_id ' AS fSID , ' Favourites. user_id ' AS fUID
FROM Favourites AS b INNER JOIN Favourites AS a
ON b. user_id = a. user_id
WHERE a. song_id = 123456 AND a. user_i d != 987654
```

Sorgu 3: Detaylı ve karmaşık

```
SELECT ' Songs. release_id ' AS sId , ' Releases. id ' ASrId
FROM Songs INNER JOIN Releases
ON Songs. release_id = Releases. id
WHERE artist_id IN
SELECT ' Genres. artist_id ' AS gAID
FROM Genres AS c
INNER JOIN Artists AS d
ON c.artist_id = d.id WHERE d.name = ' artist_name '
```

Ölçümler: Projede ölçümler için öncelikle zaman kavramı ön planda tutulması hedeflenmiştir. Zaman ölçümleri için üç yöntem ile hareket edilmiştir.

Birinci yöntem; Clock() fonksiyonu kullanımı ile belirli bir süre CPU üzerinde harcanan zaman sonuçlarının elde edilmesini sağlamaktır.

İkinci yöntem; milisaniye hassasiyetiyle zamanlamaları sağlayan Gettimeofday() fonksiyonu kullanılarak sonuçların elde edilmesini sağlamaktır.

Üçüncü yöntem; Slow Query Log (Yavaş sorgu kaydı) olarak adlandırılmaktadır. Her veri tabanı zamanı ölçmek için kendi yöntemini sunmaktadır. Bir veri tabanı için önceden belirlenmiş uzun süren sorguları kaydedebilir ve mikro saniye doğruluğu için yapılandırılabilmektedir [16].

Ölçüm Metrikleri: Veri tabanlarının performansını ölçmek için ortak bir metrik gereklidir. Bir uygulama için en önemli faktör, bir görevi tamamlamak için gereken süre ve veri tabanının bir işlemi tamamlaması durumu için gerekli zamandır. Bu kavramlar iyi anlaşılmalı ve birbirinden ayrı tutulmalıdır. Aşağıdaki formül sorguları hesaplamak için kullanılmaktadır.

$$\text{Saniyedeki Sorgu} = \frac{\text{Toplam Sorgu Sayısı x Toplam İş Parçacığı Sayısı}}{\text{Ortalama Sorgu Süresi}}$$

Her iş parçacığının saniye saniye sorgu başına nasıl tepki verdiğini ölçmek için aşağıdaki formül kullanılır.

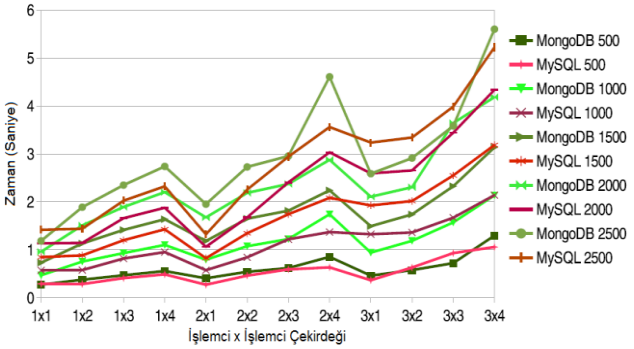
$$\text{İş Parçacığı başına saniyedeki sorgular} = \frac{\text{Toplam Sorgu Sayısı}}{\text{Ortalama Sorgu Süresi}}$$

Analiz ve sonuçlar: Burada öncelikle veri tabanlarının farklı sorgu türlerine göre nasıl yanıt verdiği hem okuma hem yazma ile analiz edilen sorguların toplam sayısı ve sonuçları şekillerle gösterilmiştir. Son olarak veri tabanı boyutunun performansa etkisi konusunda inceleme yapılmıştır.

Yapılan çalışmada daha önce açıklanan koşullar kapsamında veri tabanlarının detaylı olarak karşılaştırılabilmesi için çok çeşitli durumlar yaratılmak istenmiştir. Ölçüm için kullanılan yapılandırmalar 1'den 3'e kadar işlemci sayısı ve 1'den 4'e kadar işlemci çekirdek sayısı olarak değişmektedir. Ölçümlerde yapılan sorgu sayısı 500 ile 2500 arasındadır. Her bir ölçüm beş adet test ile bitirilmiştir. Her test sonucunda sorgulardan her birini gerçekleştirmek için alınan ortalama süreler hesaplanarak raporlanmıştır.

MySQL ve MongoDB veri tabanları sistemlerinin her ikisine eşdeğer miktarda sorgular yapılmaktadır. Bu alandaki yapılan sorgular veri tabanı sorguları bölümünde yer alan sorgu koşullarına göre, ölçümler ise ölçüm metriklerinde tanımlanan formüllere göre hesaplanarak yapılmaktadır.

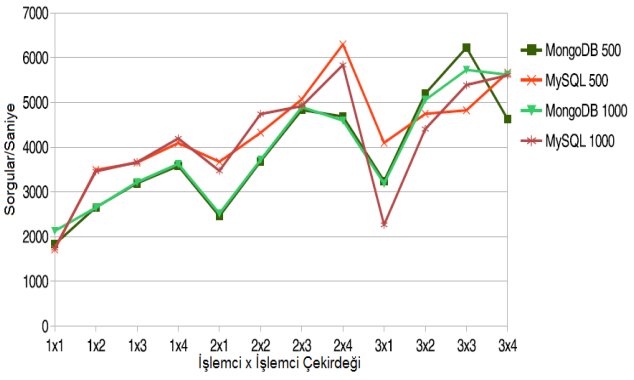
Şekil 6.3'de MySQL ve MongoDB veri tabanlarına sorgu 1 (basit sorgu) ile karşılaştırma testi uygulanmıştır. Yapılan analizde; MongoDB, sorgu sayısı farkı arttıkça daha belirgin bir performans kötülüğü gösterdiği tespit edilmiştir. Bu karşılaştırma, işlemci çekirdeği sayılarının toplam sayısı aynı olduğu zaman, 2 ya da 1 işlemci kullanımının değişmez olduğunu açıkça ortaya koymuştur (1x2 ve 2x1). MySQL veri tabanının, özellikle 3 işlemci sayısı ile 1 işlemci çekirdeği sayısına göre incelendiğinde daha kötü performans gösterdiği görülmektedir.



Şekil 6.3 Sorgu 1- Analiz işlemi

(Query 1- Analysis process)

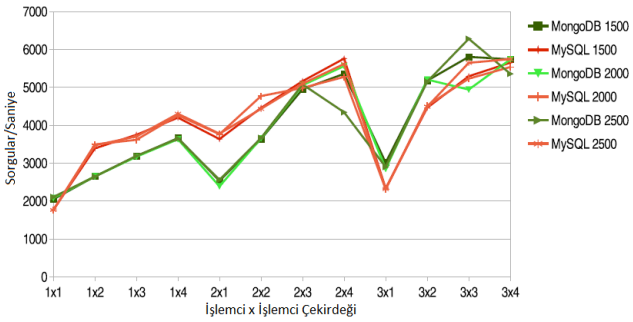
Ayrıca, sorgular/saniye ölçüm metrik grafiği ile de şekil 6.4'de görüldüğü üzere ayrıntılı ortalama süre sonuçları elde edilmiştir.



Şekil 6.4 Sorgu 1 - Sorgu/saniye analiz işlemi

(Query 1 - Queries/second analysis process)

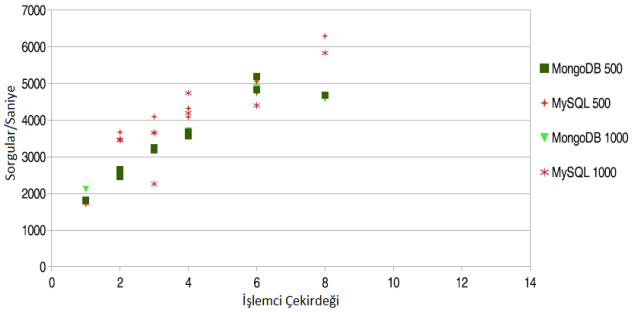
Şekil 6.5'de MySQL veri tabanı sisteminin, sorgu sayıları arttığında MongoDB üzerinde avantaj sahibi olduğu görülmektedir. Fakat 2 işlemci ve 3 işlemci çekirdeği yapılandırmasından sonraki diğer yüksek işlemci-işlemci çekirdeği sayılarında sorgu/saniye grafiğinde keskin bir şekilde azalma görülmektedir. MongoDB bu yapılandırmalarda daha fazla avantaj göstermiştir.



Şekil 6.5 Sorgu 1-Çok sayıdaki sorgu miktarı analiz işlemi

(Query 1- Many query amount analysis process)

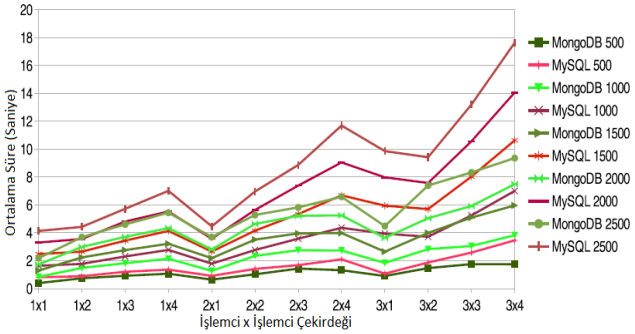
Şekil 6.6'da işlemci çekirdeği miktarı ile saniye başına yapılan sorgu sayıları arasındaki ilişki analizi gösterilmektedir. MySQL için biraz daha iyi olan performans 4 işlemci çekirdeğine kadar hemen hemen aynıdır.



Şekil 6.6 Sorgu 1-Sorgular/Saniye ile işlemci çekirdeği miktarı için analiz işlemi

(Queries/second and amount of processor core analysis process)

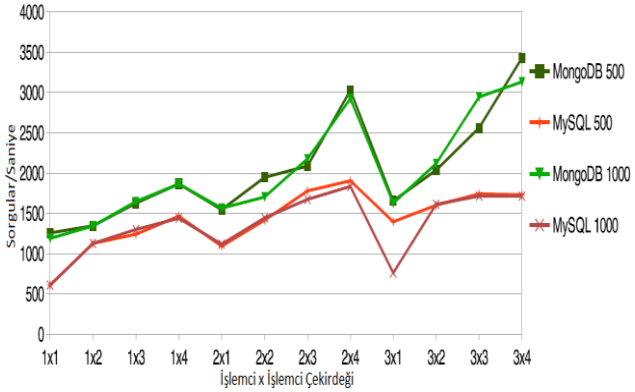
Şekil 6.7'de MySQL ve MongoDB veri tabanlarına ikinci sorgu kodu ile karşılaştırma testi uygulanmıştır. Yapılan analizde; MySQL veri tabanı sisteminin MongoDB'ye göre ortalama sorgu süreleri sonuçları, sorgu sayısı farkı arttıkça daha belirgin bir performans kötülüğü göstermiştir.



Şekil 6.7 Sorgu 2 - INNER JOIN ile karmaşık sorgu analizi işlemi

(complex query analysis process with INNER JOIN)

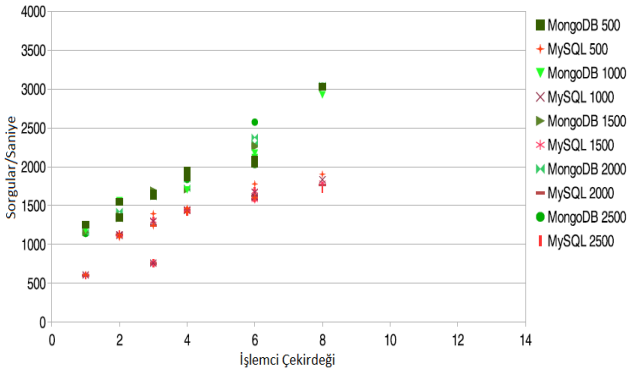
Şekil 6.8'de MySQL ve MongoDB veri tabanlarına ikinci sorgu kodu ile karşılaştırma testi uygulanmıştır. Bu test 500 ve 1000 gibi küçük veri kayıtları üzerinde yapılmıştır. Yapılan analizde; MongoDB veri tabanı sisteminin, daha az bir sürede daha çok sorgu yürütmesinin mümkün olduğu, sorgu sayısı değiştikçe performans ölçümünün daha belirgin hale gelerek sorgu/saniye başına %40 oranında daha iyi performans sergilediği gözlemlenmiştir.



Şekil 6.8 Sorgu 2- INNER JOIN ile 500 ve 1000 veri için sorgu/saniye analizi işlemi

(500 and 1000 data query / second analysis process with INNER JOIN)

İşlemci çekirdeği miktarı ile saniye başına yapılan sorgu sayıları arasındaki ilişki analizi şekil 6.9'da gösterilmektedir. İkinci sorgu kodu ile karşılaştırma testi uygulanmıştır. Bu test 500 ile 2500 veri kayıt setleri üzerinde yapılmıştır. MySQL veri tabanı sistemi, veri kayıt miktarı ve sorgu sayısı artışına göre öncelikle kademeli bir düşüş ve ardından küçük performans artışları gösterdiği tespit edilmiştir. MySQL, artan işlemci çekirdeği ve veri kayıt miktarlarında birbirine çok yakın sorgu/saniye performansı göstermiştir. MongoDB veri tabanı sisteminin, MySQL'e göre oldukça yüksek bir performans sergilediği gözlemlenmiştir. Aynı veri kayıt setlerinde MongoDB'nin MySQL üzerindeki belirgin performans farkı ve avantajı görülmektedir.

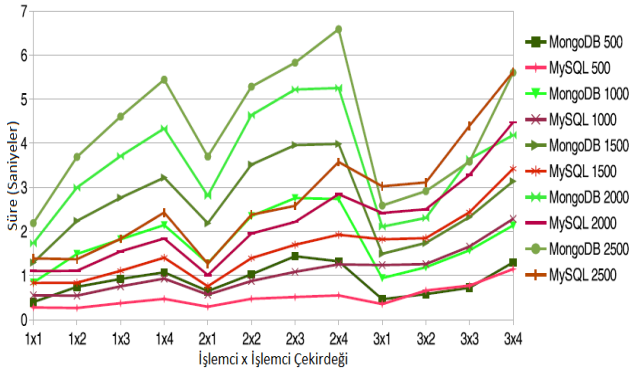


Şekil 6.9 Sorgu 2- INNER JOIN ile işlemci çekirdeği miktarı üzerinde analiz işlemi

(analysis on the amount of processor core operation with INNER JOIN)

Şekil 6.10'da iç içe geçmiş "SELECT" ve "WHERE" işlemlerini içeren üçüncü sorgu neticesinde ortaya çıkan performans değerleri gösterilmektedir.

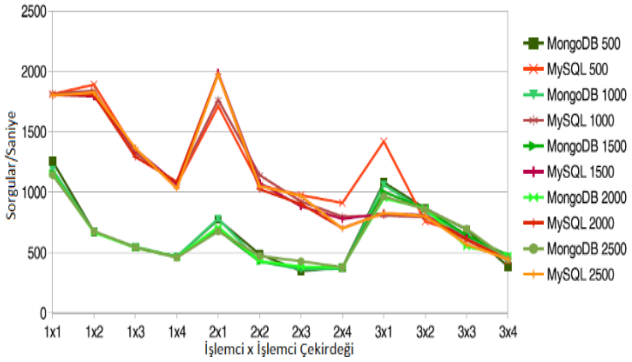
Yapılan analizlerde; MySQL veri tabanı sisteminin MongoDB 'ye göre sorgu süresi sonuçları, veri kayıt sayısı farkı arttıkça iyi bir performans göstermiştir. Fakat işlemci ve işlemci çekirdeği yapılandırmalarının değiştirildiği durumlarda performans farklılıkları daha belirgin hale gelmiştir. Bu karşılaştırma, işlemci ve işlemci çekirdeği sayılarının 3x1, 3x2, 3x3 ve 3x4 şeklinde yapılandırıldığı anlarda iki veri tabanı birbiri üzerine hemen hemen aynı performansı gösterdiği gözlemlenmiştir.



Şekil 6.10 Sorgu 3 – Detaylı karmaşık sorgu süre analizi

(Detailed and complex query time analysis)

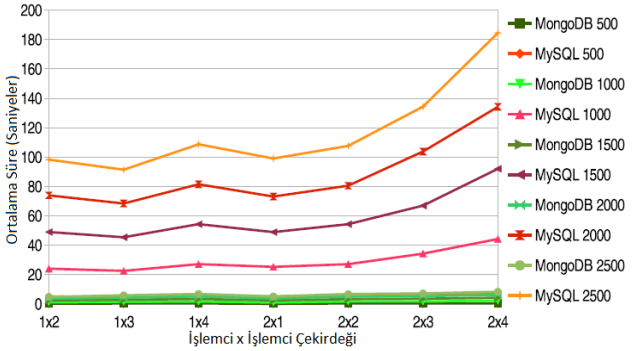
MySQL ve MongoDB veri tabanlarına üçüncü sorgu kullanılarak uygulanan karşılaştırma testi sonuçları Şekil 6.11'de daha iyi anlaşılmaktadır. MySQL veri tabanı sisteminin 2x4 işlemci ve işlemci çekirdeği yapılandırmasında en iyi performansı gösterdiği açık bir şekilde görülmektedir. Fakat 2x1 ve 3x1 işlemci ve işlemci çekirdeği yapılandırmalarında her iki veri tabanı için performans sorunu olduğu gözlemlenmektedir. Bu performans sorunu MySQL'de daha belirgin haldedir.



Şekil 6.11 Sorgu 3- Detaylı ve karmaşık sorgu ile Sorgular/saniye analiz işlemi

(Queries/second analysis process with detailed and complex queries)

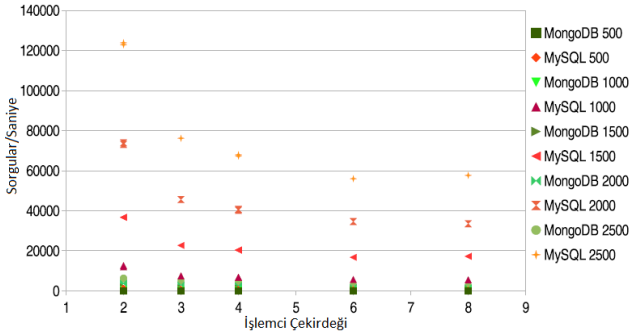
Şekil 6.12'de üçüncü sorgu ile ortalama süre ölçümleri gösterilmiştir. Yapılan analizde; MySQL veri tabanı sisteminin MongoDB'ye göre ortalama sorgu süreleri sonuçları, veri kayıt sayısı farkı arttıkça oldukça belirgin bir performans kötülüğü gösterdiği gözlemlenmiştir. Bu karşılaştırma, eşdeğer veri kayıt setlerinde yapılmasına rağmen MySQL veri tabanının sorguları işleme ve sonuçlandırma süresi ortalamasının oldukça yüksek olduğu görülmektedir. MongoDB daha belirgin ve istikrarlı bir performans göstermektedir.



Şekil 6.12 Sorgu 3 – Detaylı ve karmaşık sorgu kodu ile ortalama süre analiz işlemi

(The average time analysis process with detailed and complex query code)

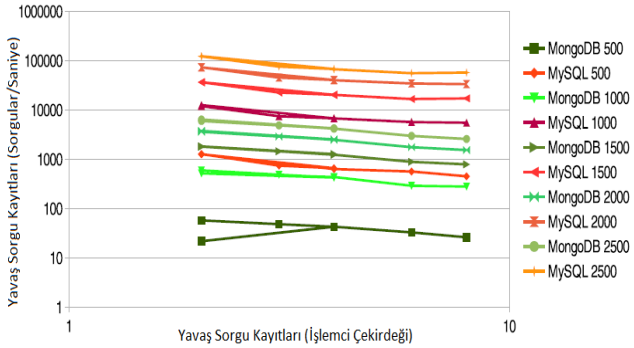
MySQL ve MongoDB veri tabanlarına üçüncü sorgu olarak tanımlanan detaylı ve karmaşık sorgu kodu içeren karşılaştırma testi analizi Şekil 6.13'de gösterilmiştir. Bu test 500 ile 2500 veri kayıt setleri üzerinde yapılmıştır. MySQL veri tabanı sistemi, iki eksen boyunca logaritma kullanılarak çizilen grafikte logaritmik bir eğilim olduğu görüntüsü sergilemektedir. MongoDB'nin ise eğilimi nerede ve nasıl gösterdiği net olarak görülmemektedir.



Şekil 6.13 Sorgu 3 - Detaylı ve karmaşık sorgu ile işlemci çekirdeği üzerinde analiz işlemi

(Process analysis of processor cores with setailed and complex queries)

Zamanlama ölçeği büyütülerek veri tabanları sistemleri arasındaki performans farkının Şekil 6.14'de daha anlaşılabilir hale geldiği görülmektedir. Ölçek büyüdüğünde MySQL'in performansındaki dezavantaj açıkça görülmektedir. MongoDB tüm veri kayıt setlerinde oldukça iyi bir performans gösterdiği ortaya konulmuştur.



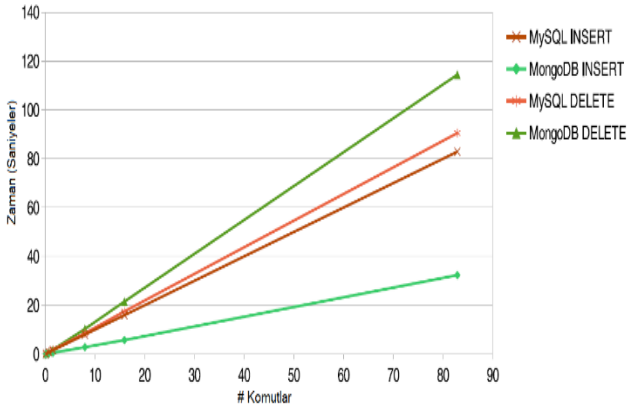
Şekil 6.14 Sorgu 3- Detaylı ve karmaşık sorgu ile ölçeklendirilmiş analiz işlemi

(Scaled analysis process detailed and complex queries)

Son olarak MySQL ve MongoDB veri tabanlarına veri ekleme "INSERT" ve silme "DELETE" işlemleri uygulanmıştır. Veri tabanı komutlarının çoğu veri okuma ve yazma sorguları olmasına rağmen silme sorguları da önemli bir faktör olarak görülmektedir. Şekil 6.15'de her iki veri tabanı sisteminin INSERT ve DELETE işlemlerine ait performans grafiği gösterilmektedir.

Yapılan analizde; her iki veri tabanının komut sayılarına göre işlem süreleri doğrusal bir eğilim göstermektedir. MongoDB'nin veri ekleme işlemi MySQL'e göre çok daha iyi bir performansa sahiptir.

Veri silme işleminde ise MongoDB'nin MySQL ile benzer bir performansa sahip olduğu fakat veri silme komut sayılarının artışı ile MySQL veri tabanı sisteminin silme işleminde iyi bir performans sergilediği gözlemlenmiştir.



Şekil 6.15 INSERT ve DELETE işlemleri

(INSERT and DELETE operations)

7. SONUÇ VE DEĞERLENDİRME (RESULT AND EVALUATION)

Klasik ilişkisel veri tabanlarını altküme olarak gören, aynı zamanda SQL ve Daha Fazlası (Not Only SQL) olarak da adlandırılan dağıtık mimari ile oluşturulmuş veri tabanları ile ilişkisel veri tabanları karşılaştırılmış ve yönetim bilişim sistemleri açısından incelenmiştir.

Bu çalışmada, yönetim bilişim sistemleri kapsamında veri tabanlarının modellemesi ve niteliklerinin belirlenmesi, en uygun performans ölçümleri, sürecin uygun hale getirilmesi ve en uygun veri tabanı seçiminde kullanıcılara ışık tutulması hedeflenmiştir. Son yıllardaki teknolojik ilerleme ile paralel olarak yapılan çalışmalar, veri tabanı sistemlerinde de etkisini göstererek NoSQL kavramını gün yüzüne çıkartmıştır. Hazırlanan bu makalede ilişkisel ve ilişkisel olmayan veri tabanı yönetim sistemlerinin performans karşılaştırmasının yapılması ve farklı faktörlerin her bir veri tabanını nasıl etkilediğini keşfederek hangi teknolojinin hangi durumlarda diğerinden daha uygun olduğunun araştırılması amaçlanmıştır.

Yapılan literatür taramasında en yakın olarak görülen çalışmalarda; Ahmet ALADILY tarafından 2015 yılında yazılan "En çok kullanılan NoSQL veri tabanları performans karşılaştırması" konulu tezde 4 ayrı dağıtık veri tabanı (Couchdb, Mongodb, Cassandra ve Hbase) performansı karşılaştırıldığı görülmüştür. Yılmaz GÖKŞEN ve Hakan AŞAN tarafından hazırlanan "Veri büyüklüklerinin veri tabanı yönetim sistemlerinde meydana getirdiği değişim: NoSQL" konulu çalışmada NoSQL veri tabanları özelinde bilgiler verilerek esneklik, tutarlılık, performans, sorgulama dili ve veri bağlantıları kapsamında ilişkisel veri tabanları ile genel bir karşılaştırma yapılmıştır. Bu çalışmada ise yapılan uygulama ile bilinen tutarlılık, esneklik gibi özelliklerden bahsedilmek sureti ile performans özelinde çok farklı durumlar yaratılarak detaylı bir analiz yapılmıştır.

Artan yazılım rekabetinde kullanıcının en önemli tercih kıstaslarından biri olan çalışma hızı dikkate alınarak, oldukça yaygın kullanım alanına sahip veri tabanı yönetim sistemlerinden MongoDB ve MySQL'in mümkün olduğunca eşit koşullarda işlem süreleri hesaplanarak performansları karşılaştırılmıştır. Farklı sorgu tipleri çalıştırılan testlerde, detaylı ve karmaşık yapılandırmalar ile veri tabanları analiz edilmiştir.

Yapılan analizlerde NoSQL ağırlıklı bir veri tabanının büyük miktarda veri çiftleri içerebildiği, veri çoğaltmada da basit şeması nedeniyle MongoDB kullanılarak daha hızlı daha karmaşık sorgu tiplerinin çalıştırılabildiği izlenmiştir. Her iki veri tabanı sisteminde farklı yapılandırma durumlarında ikinci sorgu tipi ile yapılan performans testlerinde, MongoDB veri tabanı sistemi MySQL'e göre en iyi performansı göstermiştir. Sonraki karşılaştırma testlerinde detaylı ve karmaşık sorgular ele alınarak MongoDB, alt belge koleksiyonu kullanımı nedeniyle MySQL üzerinde çok büyük bir avantaja sahip olduğunu göstermiştir. Bu avantaj, veri tekrarı yaşanabilme durumu pahasına dikkate değer bir şekilde görülmüştür. Bu tür sorgularda büyük veri tabanı boyutundan kaynaklanan depolama ve bellek miktarı maliyetini hesaplarken alternatif olarak NoSQL veri tabanlarını dikkate almak çok önemlidir.

Yapılan son performans testleri ise yazma ve silme işlemleridir. Basit arama sorgularında mantıksal olarak veri silme işlemi dikkate alınarak yapılan karşılaştırmalar sonucunda MySQL veri tabanı sistemi iyi bir performans göstermiştir. Öncelikle silinecek verinin bulunması gerektiğinden silme işlemine direk olarak bağlantılanır. Her iki veri tabanı bu karşılaştırma sonucunda doğrusal bir eğilim gösterirken MongoDB eklemeler sırasında MySQL'e göre oldukça belirgin ve çok daha iyi bir performans göstermiştir.

Yapılan testlerin bir diğer önemli yönü, işlemci ve işlemci çekirdeklerinin farklı şekillerde yapılandırılarak kullanılmasıdır. Bu çalışmada, veri tabanlarının 1, 2 ve 3 işlemci ile birden fazla işlemci çekirdeği üzerinde nasıl performans göstereceğinin test edilmesi ve karşılaştırılması için çoklu sorgulama işlemleri kullanılmıştır. Bu test detaylı ve karmaşık sorgular ile birlikte yapılmıştır. Burada sorgu karmaşıklığına bağlı olarak veri tabanları farklı davranmaktadır. Bu farklılık ise daha fazla sorgu sayısı ile sorgular/saniye analiz grafiklerinde görülmüştür.

Çok çeşitli durumlar yaratılarak elde edilen sonuçlar ile işletmelere hangi durumda nasıl bir veri tabanı yönetim sistemi kullanmaları konusunda fikir verilmiştir.

Sonuç olarak, farklı kriterler ile bu veri tabanlarını incelediğimizde iki veri tabanının da avantaj ve dezavantajları olduğu görülmüştür. İlişkisel veri tabanı yönetim sistemlerinin kullanıldığı uygulamaların ilişkisel olmayan (NoSQL) sistemlere taşınmasının ilk etapta zor olması, veri kaybının söz konusu olabilmesi ve NoSQL veri tabanı sistemlerinin veri güvenliği alanında ilişkisel veri tabanı yönetim sistemleri kadar mesafe kat etmemiş olması gibi dezavantajları olsa dahi hız, geliştirme zamanı ve ölçeklenebilirlik gibi özellikleri ile ilişkisel olmayan (NoSQL) veri tabanlarının kullanılması performans açısından daha etkin sonuçlar almamızı sağlayacaktır.

KAYNAKLAR (REFERENCES)

[1] A. Emhan, "**Karar Verme Süreci ve Bu Süreçte Bilişim Sistemlerinin Kullanılması**", Elektronik Sosyal Bilimler Dergisi, 212-224, 2007.

[2] Y. Gökşen, "**Veri Büyüklüklerinin Veri tabanı Yönetim Sistemlerinde Meydana Getirdiği Değişim: NoSQL**", Bilişim Teknolojileri Dergisi, 8-3, 2015

[3] D. Karahoca, "**Yönetim Bilişim Sistemleri**", Beta Basım Yayım Dağıtım A.Ş., İstanbul, 1998.

[4] Internet: E. Baycan, "**Veri Tabanı Yönetim Sistemleri"**, http://www.gencklavyeler.com/forum/attachment.php?aid=1135.

[5] Internet: "**Veri Tabanı Modeli**", http://en.wikipedia.org/wiki/Database_model, 6.02.2017.

[6] E. Önder, "**Yönetim Bilişim Sistemleri Kapsamında Web Tabanlı İlişkisel Veri tabanı Yönetim Sistemleri ve Bir Uygulama**", Yüksek Lisans Tezi, İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, İşletme Anabilim Dalı, Sayısal Yöntemler Bilim Dalı, 2005.

[7] Ö. Özaslan, "**Web Tabanlı Jeodezik Amaçlı Mekânsal Veri Tabanı tasarımı ve Uygulamaları**" Doktora Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Jeodezi ve Fotogrametri Mühendisliği, Geomatik Mühendisliği, 2011.

[8] E.F. Codd, "**A Relational Model of Data for Large Shared Data Banks**", 1970.

[9] D. Gündüz, "**Veri tabanlarına Giriş Seminer Notları**", IV. Akademik Bilişim Konferansı, Konya, 2002.

[10] Ö. Özaslan, "**Jeodezik Veri Tabanı Tasarımı ve WEB Tabanlı Yönetimi**" Yüksek Lisan Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, 2003.

[11] R. Stephens, R. Plew, "**Veri Tabanları**", Çeviren: Nalan Güven Küçükler, Alfa Basım Yayım Dağıtım Ltd. Şti, İstanbul, 2003.

[12] F. Sürmeli, M. Erdoğan, N. Erdoğan, K. Banar, E. Kaya, A. Sevim, "**Muhasebe Bilgi Sistemi**", Anadolu Üniversitesi, Eskişehir, 2006.

[13] Internet: "**Acid**" http://en.wikipedia.org/wiki/Acid; 08.08.2014.

[14] Internet: S. Davaz, "**NoSQL Nedir Avantajları ve Dezavantajları Hakkında Bilgi**" https://blog.kodcu.com/2014/03/nosql-nedir-avantajlari-ve-dezavantajlari-hakkinda-bilgi, 28.03.2014.

[15] G. Vaish, "**Getting Started With NoSQL**", Packt Publishing, United Kingdom, 2013.

[16] M. Otey, "**NoSQL? No Way!**", SQL Server Magazine, pp:5, 2010.

[17] Internet: "**Oracle NoSQL Database**", http://www.oracle.com/technetwork/database/nosqldb/learnmore/nosql-database-498041.pdf, 11. 2011.

[18] N. Rozanski, E. Woods, "**Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives**", Pearson Education Inc., USA, 2012.

[19] G. Burd, "**NoSQL**", Sysadmin Journal, Vol:36 No. 5 ss:5-12 ABD, 2011.