

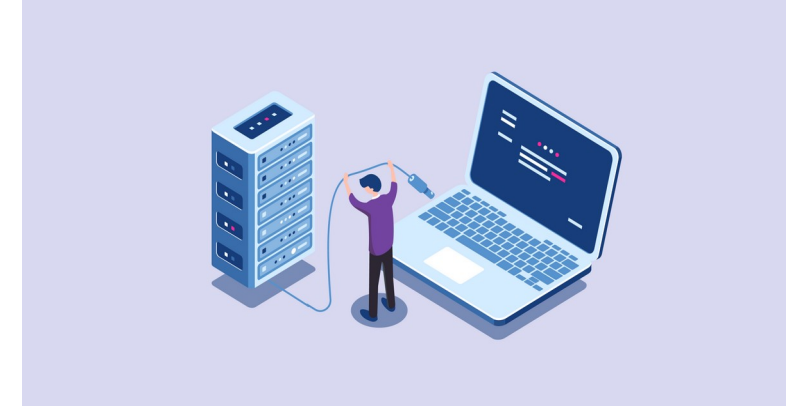
İlişkisel ve İlişkisel Olmayan (NoSQL) Veri Tabanı Sistemleri Mimari Performansının Yönetim Bilişim Sistemleri Kapsamında İncelenmesi

AD-SOYAD : İPEK NUR YURTTAŞ

NUMARA : 02220224025

1. GİRİŞ (INTRODUCTION)

- Teknolojideki hızlı gelişim birçok organizasyonu etkileyerek yeni çözümler üretmeye zorlamaktadır. Bu da belli bir amaca ulaşmak için veri veya bilginin organizasyonlar tarafından kısa sürede erişilmek istenen faktör olmasını sağladı.
- Yaşanan bu gelişim , veri tabanı kullanımını zorunlu hale getirdi. Verinin boyutu ve karmaşıklığı gibi etkenlerden dolayı farklı veri modelleme yöntemleri geliştirilmiştir.
- Bu kapsamda , ilişkisel veri tabanlarının yanı sıra ilişkisel olmayan veri tabanı sistemleri de kullanılmaktadır.



2.BİLİŞİM SİSTEMLERİ VE YÖNETİMİ (SYSTEM AND MANAGEMENT)

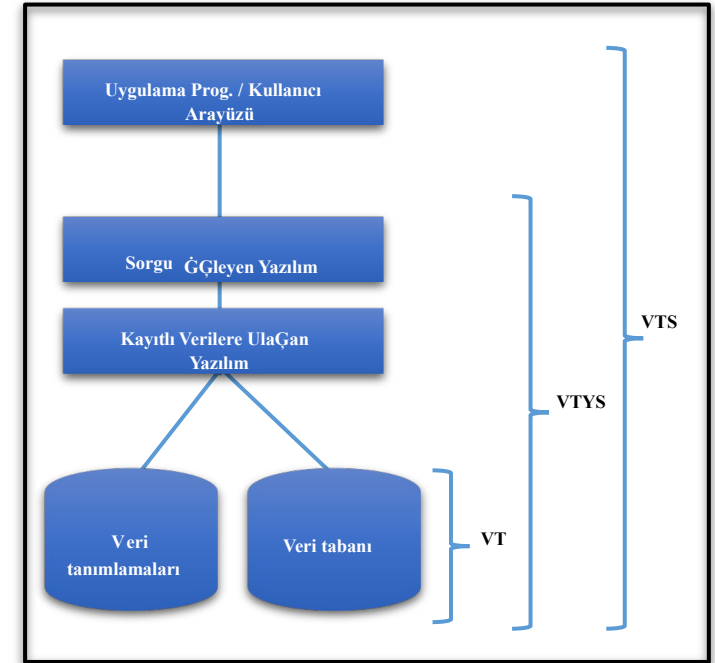
- Bilişim sistemi , organizasyonlar da karar verme aşamasına kadar bilgiyi toplamak , düzenlemek , işlemek ve saklamak olarak tanımlanır. Bilişim sistemlerinde bilgiyi üretmek için üç aktivite gereklidir. Bu aktiviteler girdi , işlem ve çıktıdır.
- Girdi , organizasyonlar da ham bilgileri (veriyi) toplamaktır. İşlem , bu ham veriyi anlamlı hale getirir. Çıktı , işlenmiş veriyi gerekli yere aktarır.
- Bilişim sistemleri, bilişim teknolojileri altyapısından yararlanan yönetsel çözümlerdir. Bilişim sistemlerini etkin bir şekilde kullanmak için organizasyon, yönetim ve teknolojiyi bilmek gerekir.



Şekil 2.1 Bilişim Sistemleri Bileşenleri
(Information Systems Components)

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

- Veri tabanı amaca uygun düzenlenmiş veriler topluluğudur. Veri tabanları gerçekte var olan ve birbirleriyle ilişkisi olan nesneleri ve ilişkileri modeller. Veri tabanı yönetim sistemleri (VTYS), verilerin nasıl depolanacağı, kullanılacağı ve erişileceğini yönlendiren kurallar sistemidir. Veri tabanı, VTYS ve uygulama programları ile kullanıcı arayüzlerini içeren yapıya veri tabanı sistemi (VTS) denir.



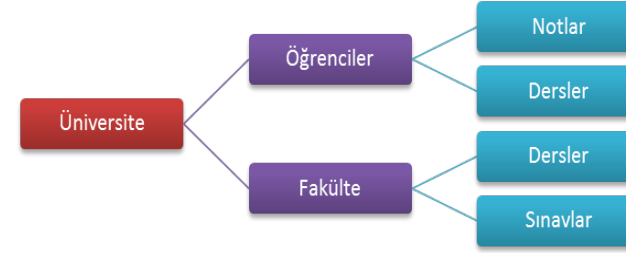
- Veri tabanı modellerini sekiz kategoriye ayırabiliriz :

- **Düz Model veya Tablo Modeli :** İki boyutlu veri grubundan oluşur. Sütunlarda verilerin benzer özellikleri, satırlarda ise veri grupları yer alır.

	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:** Bu veri tabanının depoladığı yapısal verilere “kayıt” adı verilir. Kayıtlar ağaç mimarisi şeklinde yukarıdan aşağı sıralanır. Kök adı verilen ilk kaydın bir veya daha fazla çocuk kayıtları olabilir. Çocuk kayıtlarında kendi çocuk kayıtları olabilir. Kök haricinde bütün kayıtların bir ebeveyni vardır .



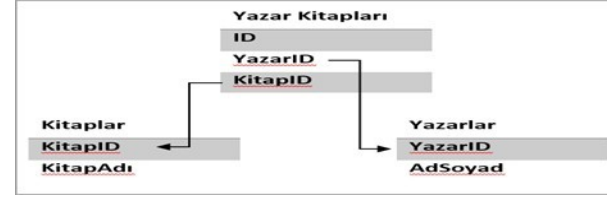
Şekil 3.3 Hiyerarşik Veri Tabanı Modeli
(Hierarchical Database Model)

- **Ağ veri modeli:** Hiyerarşik veri modelinin geliştirilmiş halidir. Ağ modelinin hiyerarşik modelden en önemli farkı, uç-düğüm pozisyonundaki verinin iç-düğümüne işaret edebilmesidir.



Şekil 3.4 Ağ Veri Modeli
(Network Data Model)

- **İlişkisel Veri Modeli:** Bu veri modelinin temel kavramı, ilişkidir. İlişkiler, satır ve sütunlardan oluşan iki boyutlu tablolarla karakterize edilir . Tablonun her satırı birbiriyle ilişkili verilerin bir topluluğudur. Sütunlarda ise nitelikler bulunur.



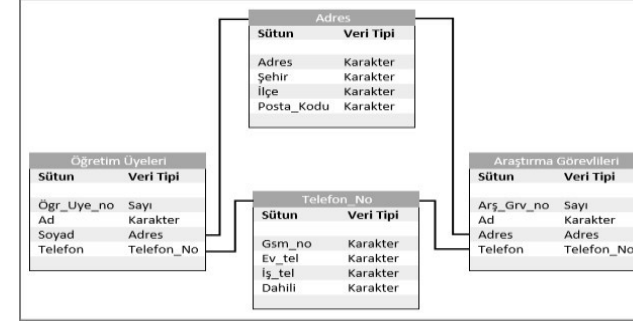
Şekil 3.5 İlişkisel Veri Modeli
(Relational Data Model)

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



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

- **Nesne İlişkisel Veri Modeli:** İlişkisel işlevselliğin üzerine nesne yönelimli özellikler içerir. Bu modeli içeren ilk veri tabanı 1997 yılında piyasaya sunulan Oracle8'dir.



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


- **Çoklu Ortam Veri Modeli:** 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.
- **Dağıtık Veri Modeli:** Dağıtık veri tabanları, iki ya da daha fazla bilgisayarda depolanan ve bir ağ üzerinde dağıtılan bilgiler için kullanılan veri tabanı grubudur. Böyle bir sistemde, birden fazla veri tabanına erişilmesine rağmen, kullanıcı bir tek veri tabanıyla çalışıyormuş gibi işlem yapar.

4. VERİ TABANI TASARIMI (DATABASE DESIGN)

- Veri tabanı tasarımında ; gerçeğin , gereksinim ve beklentiler çerçevesinde veri tabanına aktarılması gerekir. Veri tabanı sisteminde ilk , olası veri tabanı kullanıcı gereksinimlerinin belirlenmesi gerekir. Gerçeğin veri tabanındaki sayısal temsili , veri tabanı sisteminde kullanıcılar ve bilgisayar tarafından anlaşılacak şekilde tanımlanması gerekir. Bu tanımlama veri tabanında '**şema**' olarak tanımlanır. Kullanıcı ve bilgisayar düzeyleri sırasıyla '**kavramsal**' ve '**fiziksel**' düzeyler , bu düzeylerdeki şemalar da '**kavramsal şema**' ve '**iç şema**' olarak adlandırılır.



Şekil 4.1 Veri Tabanı Tasarım Aşamaları
(Database Design Stages)

- 
- ▶ Geleneksel veri tabanı tasarım , kullanıcı düzeyinden fiziksel düzeye doğrudur. Kavramsal tasarımda gereksinimlere göre kavramsal şema belirlenir. Kavramsal şema , veri tabanı kullanıcısı için , veri tabanının genel yapısını tanımlar. Kavramsal şema , fiziksel depolama yapılarının ayrıntılarına girmeden varlıklar , veri tipleri , varlıklar arasındaki ilişkiler ve kısıtlayıcılar üzerine yoğunlaşır. Bundan dolayı kavramsal şema yüksek düzeyli olarak tanımlanır ve doğrudan gerçekleştirilemez. Bu nedenle , geleneksel veri tabanı tasarımında kavramsal tasarımdan sonraki adım , gerçekleştirim için kullanılacak bir veri tabanı seçilir.
 - ▶ Fiziksel tasarım aşamasında , en yüksek verim için , veri tabanında fiziksel olarak nasıl organize edilmesi gerekir belirlenir. Sonuç , iç şemadır.
 - ▶ İç şema depolama yapılarını , kayıt formatlarını , kayıt alanlarını , veri tabanına giriş yol ve yöntemleri ile veri tabanını ilgilendiren detayları tanımlar. Kavramsal şema , yazılım ve donanımdan bağımsız iken iç şema yazılım ve donanıma bağımlıdır.

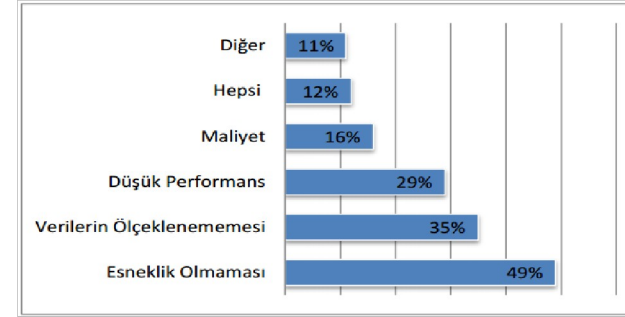
5. İLİŞKİSEL VE İLİŞKİSEL OLMAYAN (NoSQL) VERİ TABANI SİSTEMLERİ (RELATIONAL AND NONRELATIONAL DATABASE (NoSQL) SYSTEMS)

- 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 bu yüzden en az iki tablonun bulunması gerekir. Her bir tablo, belli yapıya uygun verileri saklamak için tasarlanır.
- ▶ ACID; klasik ilişkisel veri tabanı sistemlerinde sağlanan temel özellikler :
 - Bölünmezlik (Atomicity)
 - Tutarlılık (Consistency)
 - İzolasyon (Isolation)
 - Dayanıklılık (Durability)

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

▶ İlişkisel olmayan veri tabanı yatay olarak ölçeklendirilen veri depolama sistemidir. Çok büyük verilerin depolanması ve yazılmasında NoSQL kullanılmıştır. İlişkisel veri tabanı kullanıcılarının NoSQL veri tabanına geçmek istemesinin nedenleri yanda verilmiştir.



▶ İlişkisel veri tabanlarının kullandığı ACID işlemselliğine karşın NoSQL 'BASE' (Basically Available- Soft state - Eventually consistent) kısaltmasıyla ifade edilir.

- **Kolay Ulaşılabilirlik (Basically Available) :** Veri erişim sorunlarını ortadan kaldırmak için kullanılır.
- **Esnek Durum (Soft state):** NoSQL sistemler tutarsız ve süreksiz verilerin kullanılmasına da izin verir.
- **Eninde Sonunda Tutarlı (Eventually consistent):** NoSQL de gelecekte tanımlanmayan bir zamanda tutarlılığın oluşacağı vaat edilir.

► En bilinen lider NoSQL ürünlerinin teknik karşılaştırmaları :

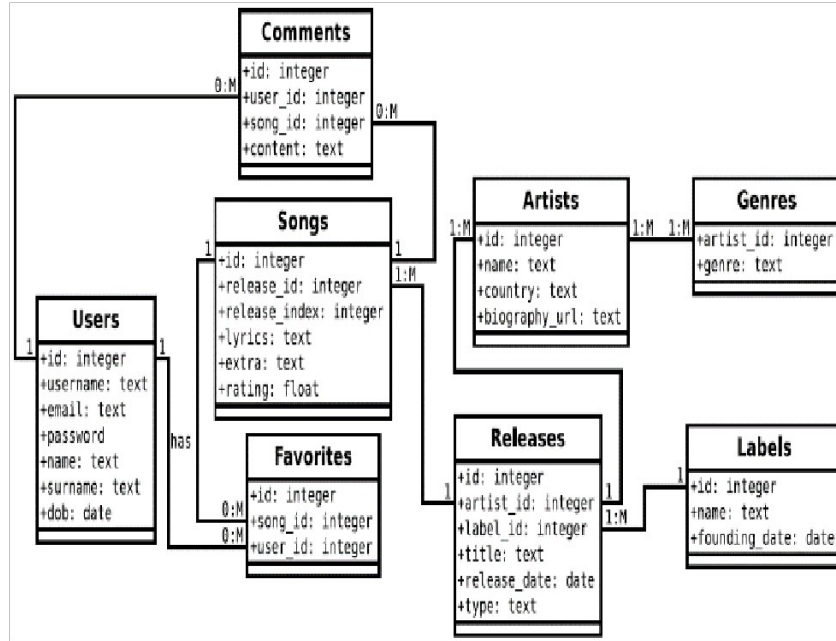
Sadeleştirme	Arama	Esinlenilen	Depolama	Protokol	Model	Lisans	Dil	
Evet	Evet		Memory mapped b-trees	BSON	Document	AGPL	C++	MongoDB
Hayır	Hayır	Dynamo	COW-BTree	HTTP/REST	Document	Apache	Erlang	CouchDB
Evet	Evet	Dynamo	Pluggable: InnoDB, LevelDB, Bitcask	HTTP/REST or TCP/Proto	Key/Value	Apache	Erlang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

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

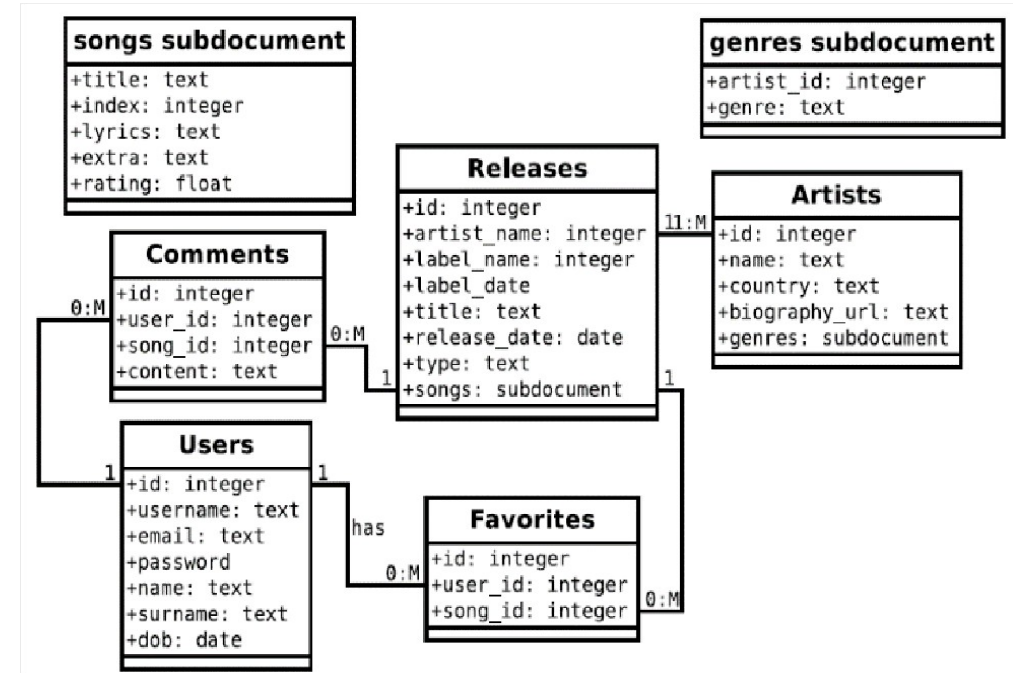
► İlişkisel veri tabanı olan MySQL ve ilişkisel olmayan NoSQL veri tabanına alternatif olarak , yatay olarak ölçeklendirilen veri depolama sistemi olan MongoDB veri tabanı sistemi kullanılmıştır. MySQL ve MongoDB veri tabanı sistemlerinin performans ve yatay olarak incenlemesi için :

- 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 diğeri ise MongoDB veri tabanıdır. Tablolarda veri tekrarını önlemek için normalizasyon değerlendirilmesi yapılmıştır.



MySQL Veri Tabanı Şeması



MongoDB Veri Tabanı Şeması

► **Veri Tabanı Sorguları :** Bu projede üç farklı veri tabanı sorgusu kullanılmıştır.

- Sorgu 1 : Basit (SELECT) :

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

- Sorgu 2 : Karmaşık (INNER JOIN) :

```
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_id != 987654
```

- Sorgu 3 : Detaylı ve Karmaşık (WHERE) :

```
SELECT ' Songs. release_id ' AS sId , ' Releases. id ' AS rId  
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 :** Zaman ölçümleri için üç yöntem kullanılmıştır .

- **Birinci Yöntem :** Clock() fonksiyonu ile CPU üzerinden harcanan zaman sonuçları elde edilmiştir.
- **İkinci Yöntem :** Milisaniye hassasiyetini sağlayan Gettimeofday() fonksiyonuyla sonuçlar elde edilmiştir.
- **Üçüncü Yöntem :** Slow Query Log (Yavaş sorgu kaydı) olarak tanımlanır. Her bir veri tabanı zamanı ölçmek için kendi yöntemini kullanır.

- **Ölçüm Metrikleri :** Veri tabanının performansını ölçmek için ortak metrik gereklidir. Aşağıda formüller sorguları hesaplamak için kullanılır.

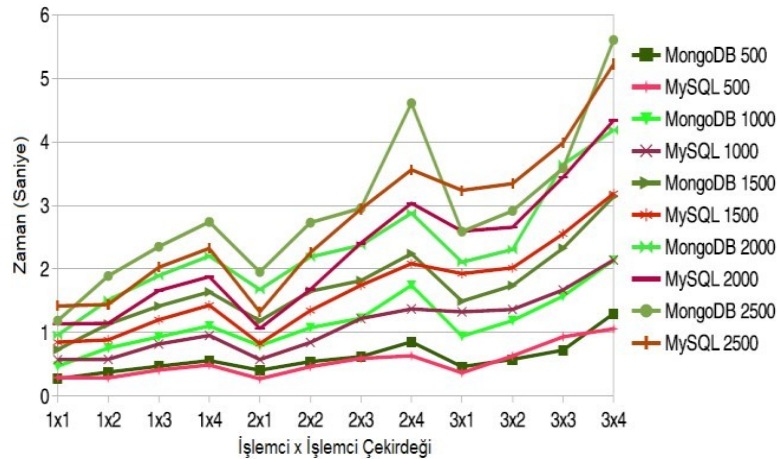
$$\text{Saniyedeki Sorgu} = \frac{\text{Toplam Sorgu Sayısı} \times \text{Toplam İş Parçacığı Sayısı}}{\text{Ortalama Sorgu Süresi}}$$

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

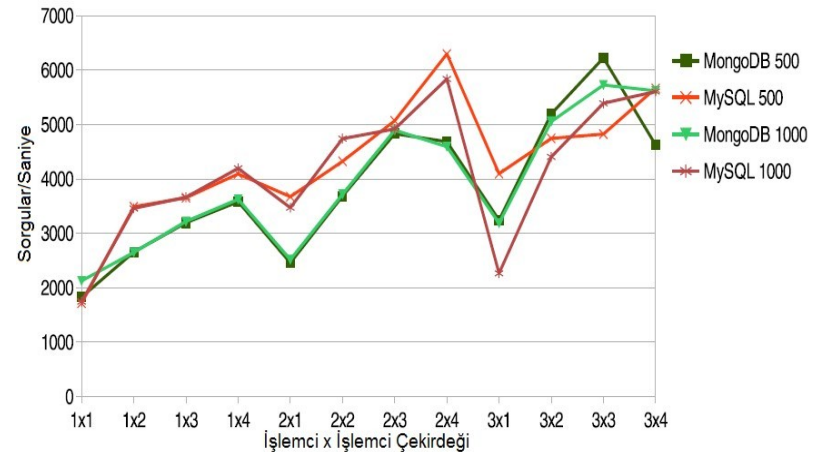
- **Analiz ve sonuçlar :** Veri tabanlarının farklı sorgu türlerine nasıl yanıt verdiği ile analiz edilen sorguların toplam sayısı gösterilmiştir.

- **Veri tabanı sistemlerinin karşılaştırması için kullanılan ölçümler grafiklerle aşağıda verilmiştir.**

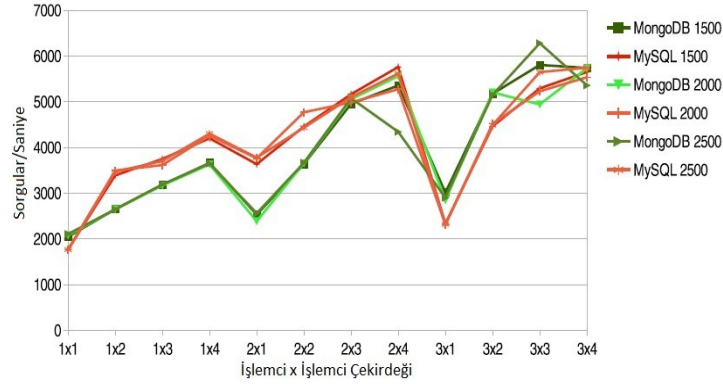
Sorgu 1- Analiz işlemi



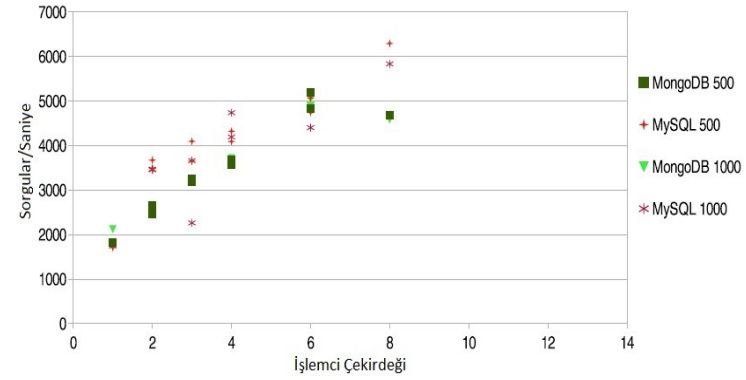
Sorgu 1 - Sorgu/saniye Analiz İşlemi



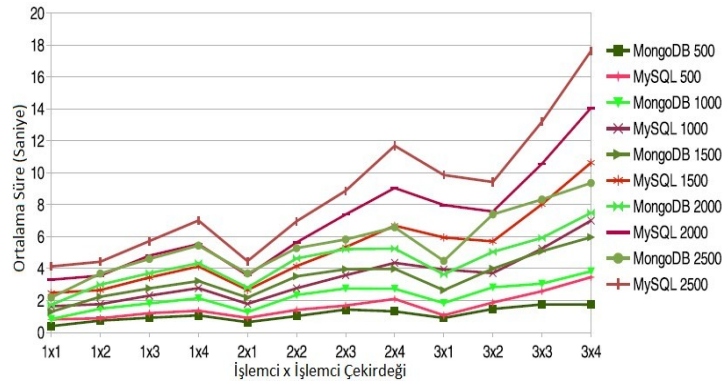
Sorgu 1-Çok sayıdaki sorgu miktarı analiz işlemi



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



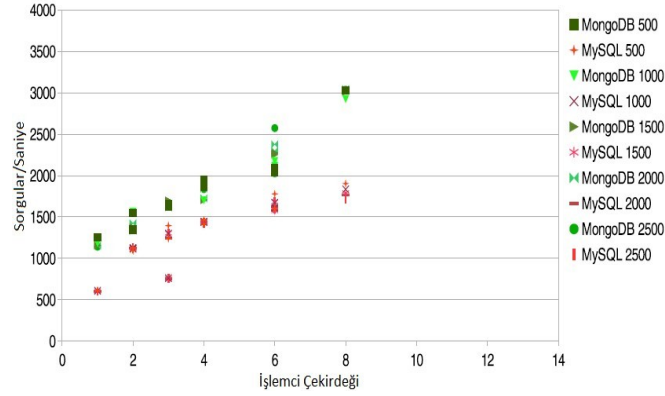
Sorgu 2 - INNER JOIN ile karmaşık sorgu analizi işlem



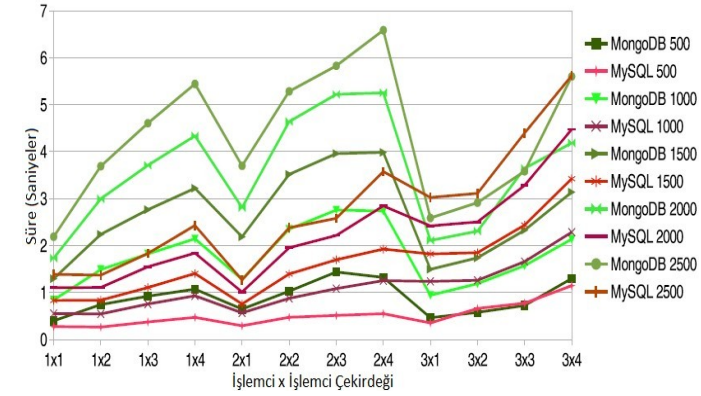
Sorgu 2- INNER JOIN ile 500 ve 1000 veri için sorgu/saniye analizi işlemi



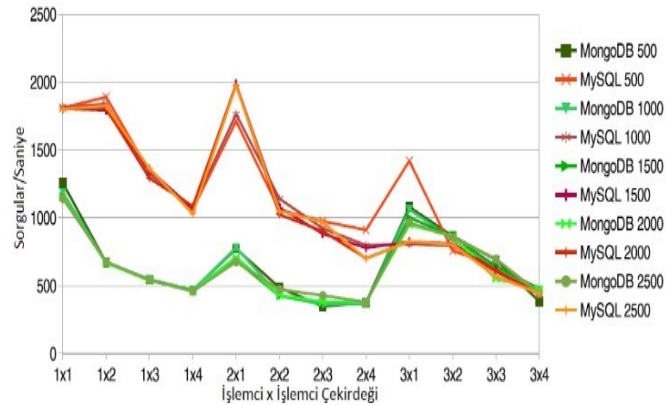
Sorgu 2- INNER JOIN ile işlemci çekirdeği miktarı üzerinde analiz işlemi



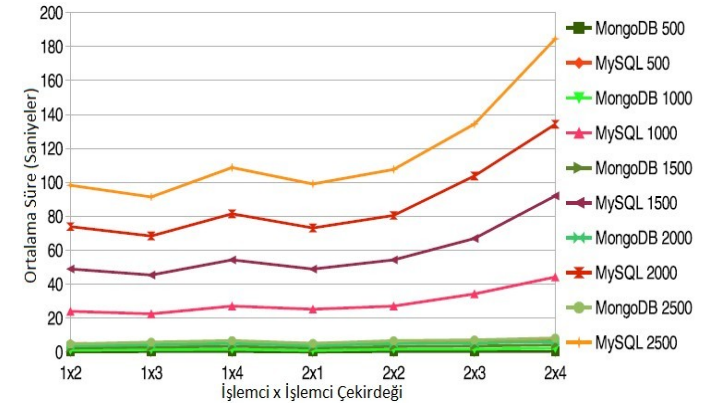
Sorgu 3 – Detaylı karmaşık sorgu süre analizi (Detailed and complex query time analysis)



Sorgu 3- Detaylı ve karmaşık sorgu ile Sorgular/saniye analiz işlem

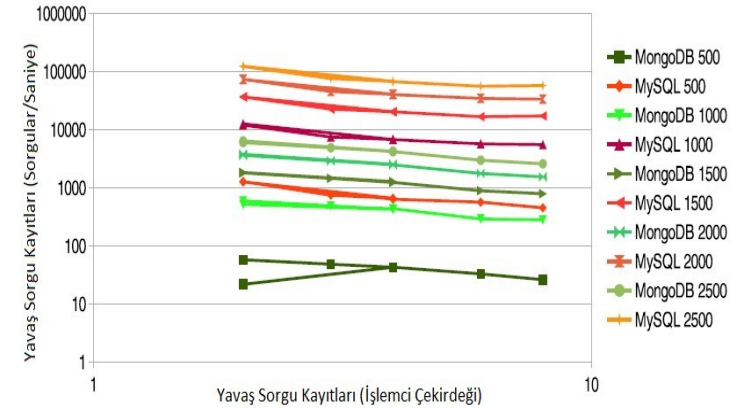
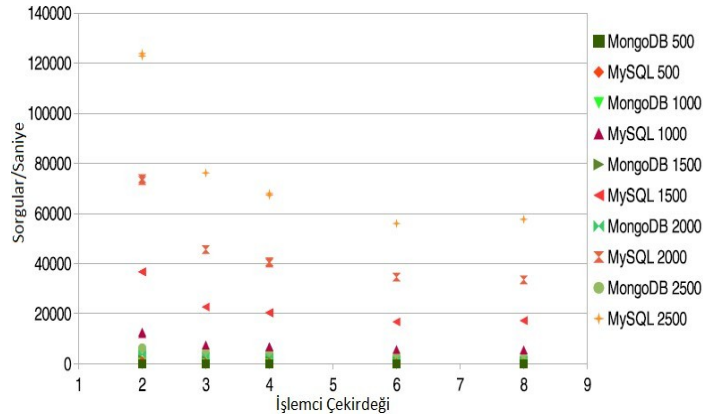


Sorgu 3 – Detaylı ve karmaşık sorgu kodu ile ortalama süre analiz işlemi

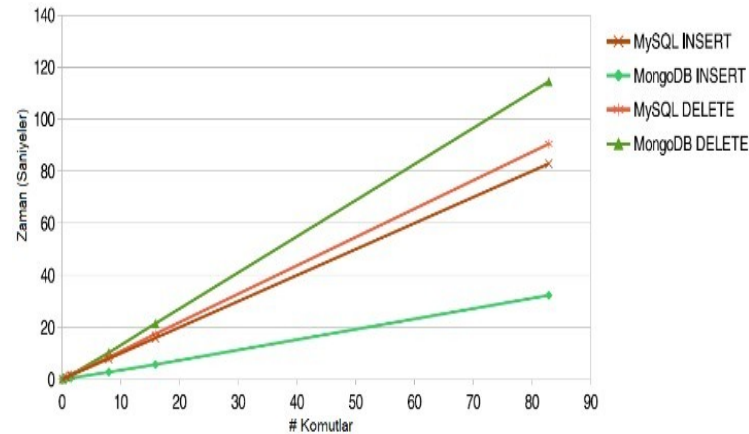


Sorgu 3 - Detaylı ve karmaşık sorgu ile işlemci çekirdeği üzerinde analiz işlemi

Sorgu 3- Detaylı ve karmaşık sorgu ile ölçeklendirilmiş analiz işlemi (Scaled analysis process detailed and complex queries)



INSERT ve DELETE işlemleri (INSERT and DELETE operations)



SONUÇ :

- ▶ Yapılan Analizde : MongoDB ' nin veri ekleme işlemi MySQL ' e göre çok daha iyi performansa sahiptir. Veri silme işleminde ise MySQL iyi bir performansa sahiptir.

