NoSQL VERİ TABANI

SUEDA ASEL BİLDİK

02210224048

1-Giriş
2-Bilişim Sistemleri ve Yönetimi
3-Veri Tabanı Yönetimi
4-NoSQL Sistemi
5-Veri Tabanları Performans Karşılaştırması

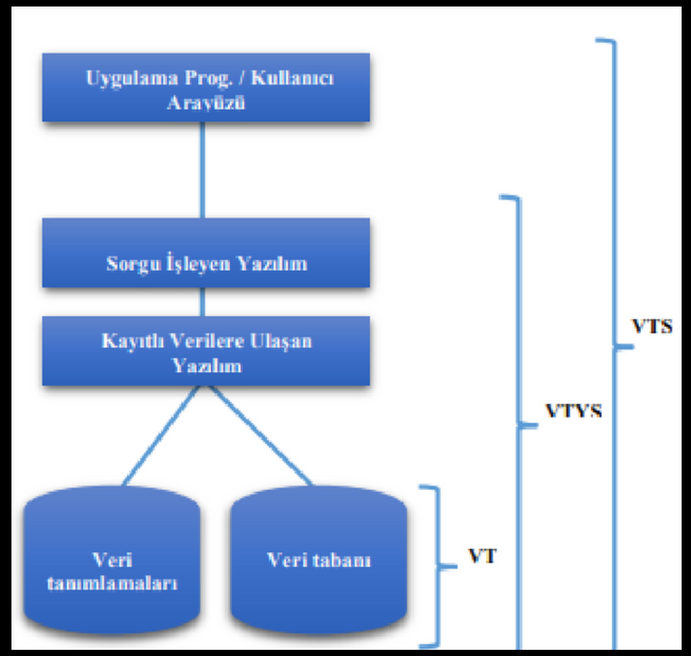
GİRİŞ

Bu metin, bilgi sistemlerinin ve veri tabanlarının öneminin arttığını, ilişkisel ve ilişkisel olmayan (NoSQL) veri tabanı sistemlerinin performans ve esneklik avantajları nedeniyle büyük şirketlerce tercih edildiğini ve bu sistemlerin mimari performanslarının karşılaştırıldığını özetliyor.

BİLİŞİM SİSTEMLERİ VE YÖNETİMİ

Bilişim sistemleri, organizasyonlarda bilginin toplanması, düzenlenmesi, işlenmesi ve saklanmasını içerir ve girdi, işlem, çıktı olmak üzere üç temel aktiviteye dayanır. Bu sistemler, ham verileri anlamlı bilgilere dönüştürerek karar verme süreçlerini destekler. İşletmeler için bilişim sistemleri, teknoloji altyapısını kullanarak yönetimsel çözümler sunar ve etkili kullanımı için organizasyon, yönetim ve teknoloji bilgisine ihtiyaç duyar.

VERİ TABANI VE VERİ TABANI YÖNETİM SİSTEMLERİ



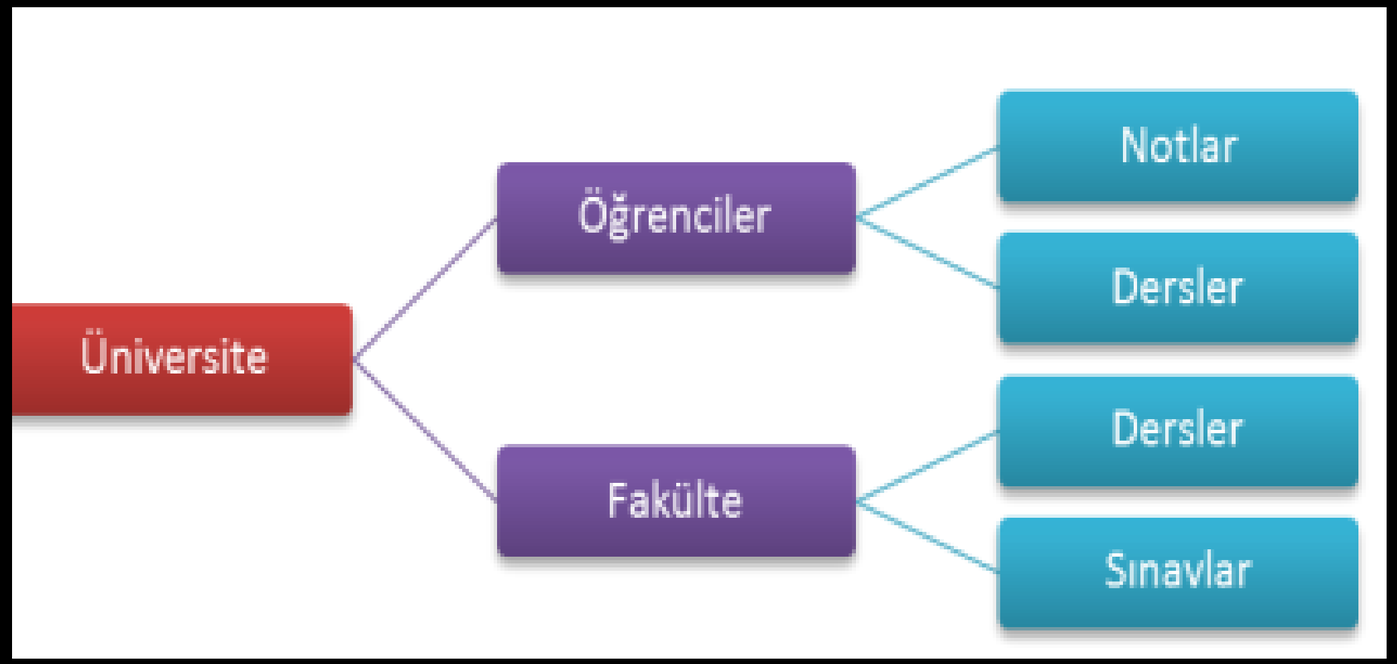
Veri tabanı, düzenlenmiş ve ilişkili verilerin saklandığı yerdir. Veri tabanı yönetim sistemleri, bu verilere düzenli erişim sağlar. Veri tabanı, yönetim sistemi ve kullanıcı arayüzlerinin birleşimi "veri tabanı sistemi" olarak adlandırılır ve gerçek dünya nesnelerini modellemek için kullanılır.

1-DÜZ VERİ MODELİ

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

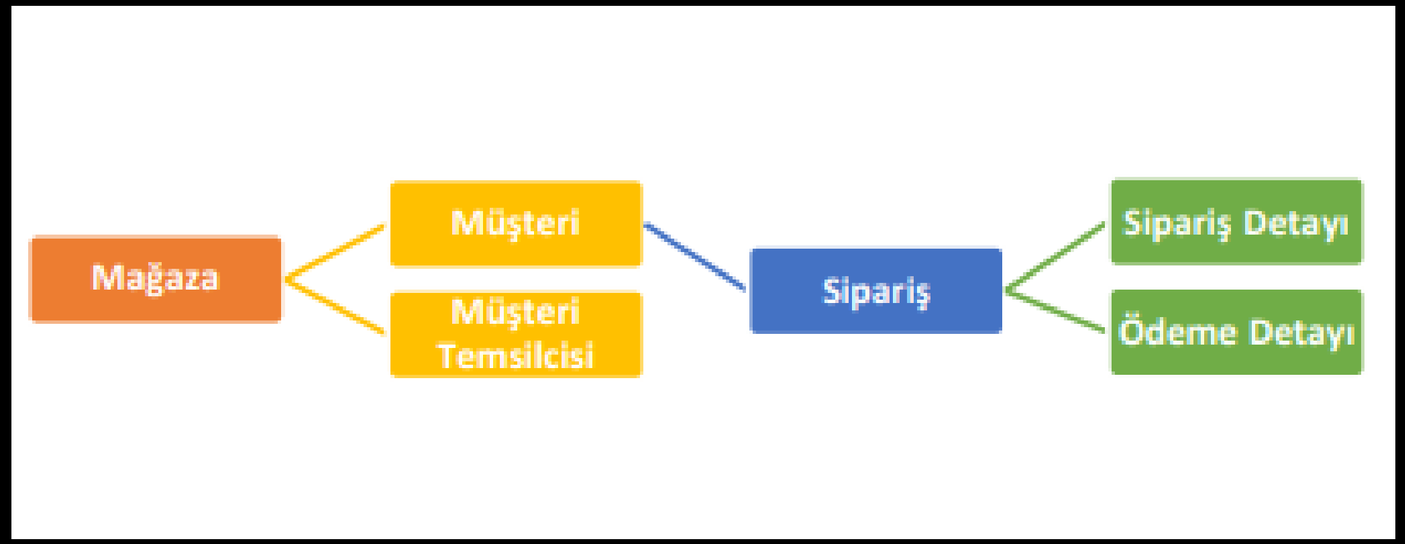
Düz model veya tablo modeli, iki boyutlu, tek tablodan oluşan bir yapıdır; sütunlar benzer verileri, satırlar ise veri gruplarını içerir.

2-HİYERARŞİK VERİ MODELİ



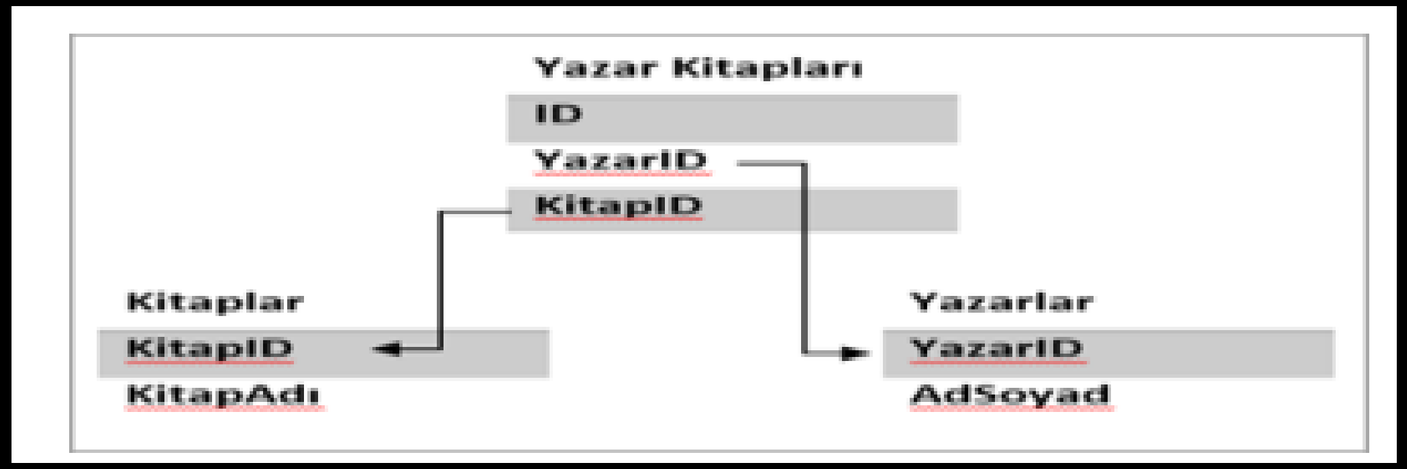
Hiyerarşik veri modeli, 1960'larda geliştirilen ve verileri ağaç yapısı şeklinde düzenleyen bir yapıdır. Bu modelde, kök kayıttan başlayarak her kaydın bir veya daha fazla çocuk kaydı olabilir, ancak kök dışındaki her kaydın sadece bir ebeveyni vardır.

3-AĞ VERİ MODELİ



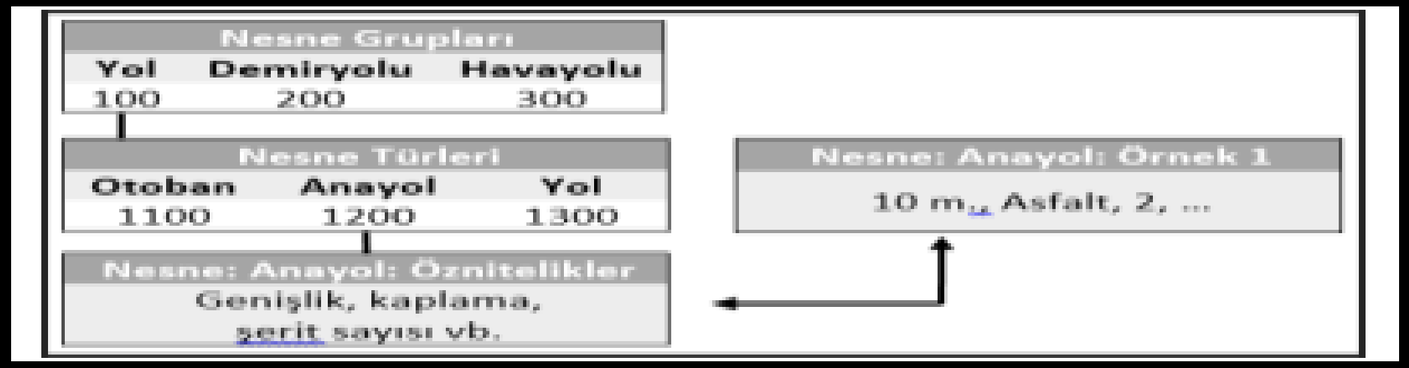
Ağ veri modeli, 1970'lerde geliştirilen ve veriler arasındaki çoka-çok ilişkileri modelleyerek hiyerarşik modeli geliştiren bir yapıdır.

4-İLİŞKİSEL VERİ MODELİ



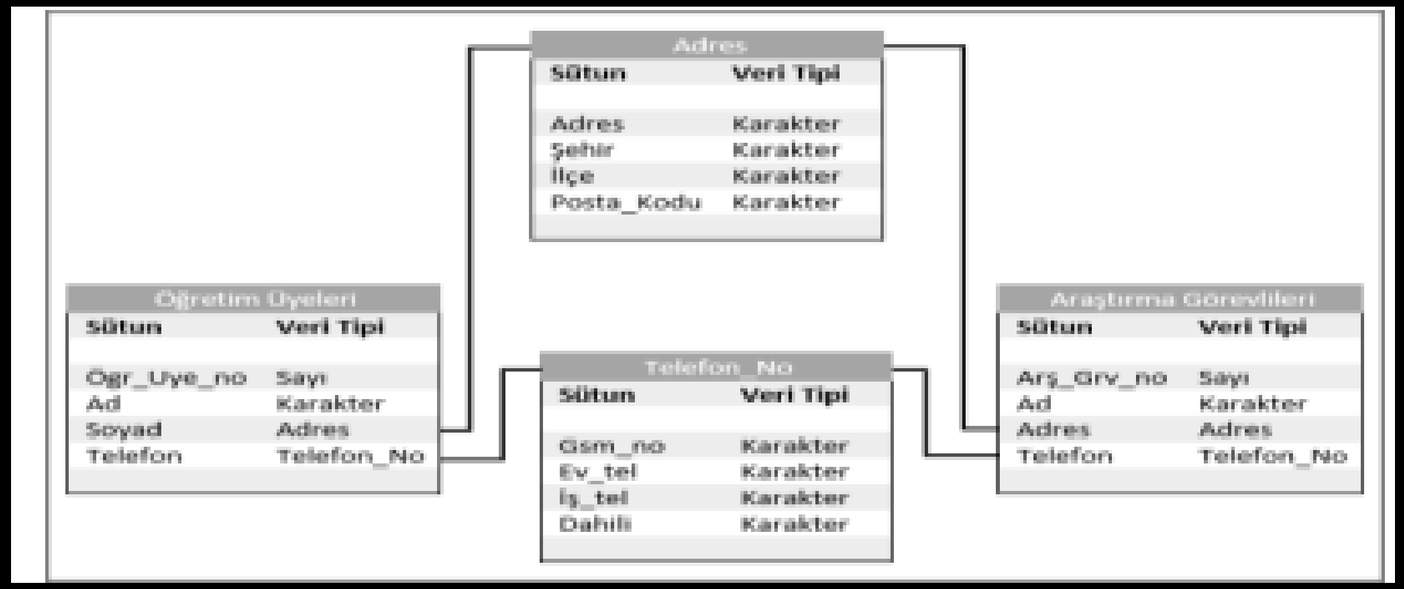
İlişkisel veri modeli, E. F. Codd tarafından 1970'de geliştirilmiş, verilerin ilişkilerle modellendiği bir yapıdır. Bu model, verileri satır ve sütunlardan oluşan iki boyutlu tablolar şeklinde düzenler, her tablo bir dosya ile ilişkilendirilir ve tablonun her satırı ilişkili veri gruplarını, sütunlar ise nitelikleri temsil eder.

5-NESNE YÖNELİMLİ VERİ MODELİ



Daha sonraları ortaya çıkmış ve başarısını kanıtlamıştır. Nesne yönelimli programlamaya dayanan veri modelidir .

6-NESNE İLİŞKİSEL VERİ MODELİ



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

7-ÇOKLU ORTAM VERİ MODELİ

Çoklu ortam veri modeli, film, müzik, metin ve video gibi büyük verileri işleyen ve özellikle tıp bilgi sistemlerinde kullanılan bir yapıdır.

8-DAĞITIK VERİ MODELİ

Dağıtık veri modeli ise, verileri birden fazla bilgisayara dağıtarak işlem hızını artıran ve kullanıcıya tek bir veri tabanıymış gibi bir deneyim sunan bir yapıdır.

VERİ TABANI TASARIMI

Veri tabanı tasarımı, gereksinimlerin belirlenmesiyle başlar ve gerçek dünyanın veri tabanına modellenmesini içerir. Tasarım, kavramsal şema ile başlayıp, mantıksal ve fiziksel tasarım aşamalarıyla devam eder. Kavramsal tasarım, kullanıcıların anlayabileceği yüksek düzeyli bir modeli ifade ederken, mantıksal tasarım seçilen veri tabanı yönetim sisteminin modeline uygun şekilde yapılır. Fiziksel tasarım ise, verilerin veri tabanında nasıl organize edileceğini ve saklanacağını belirler.

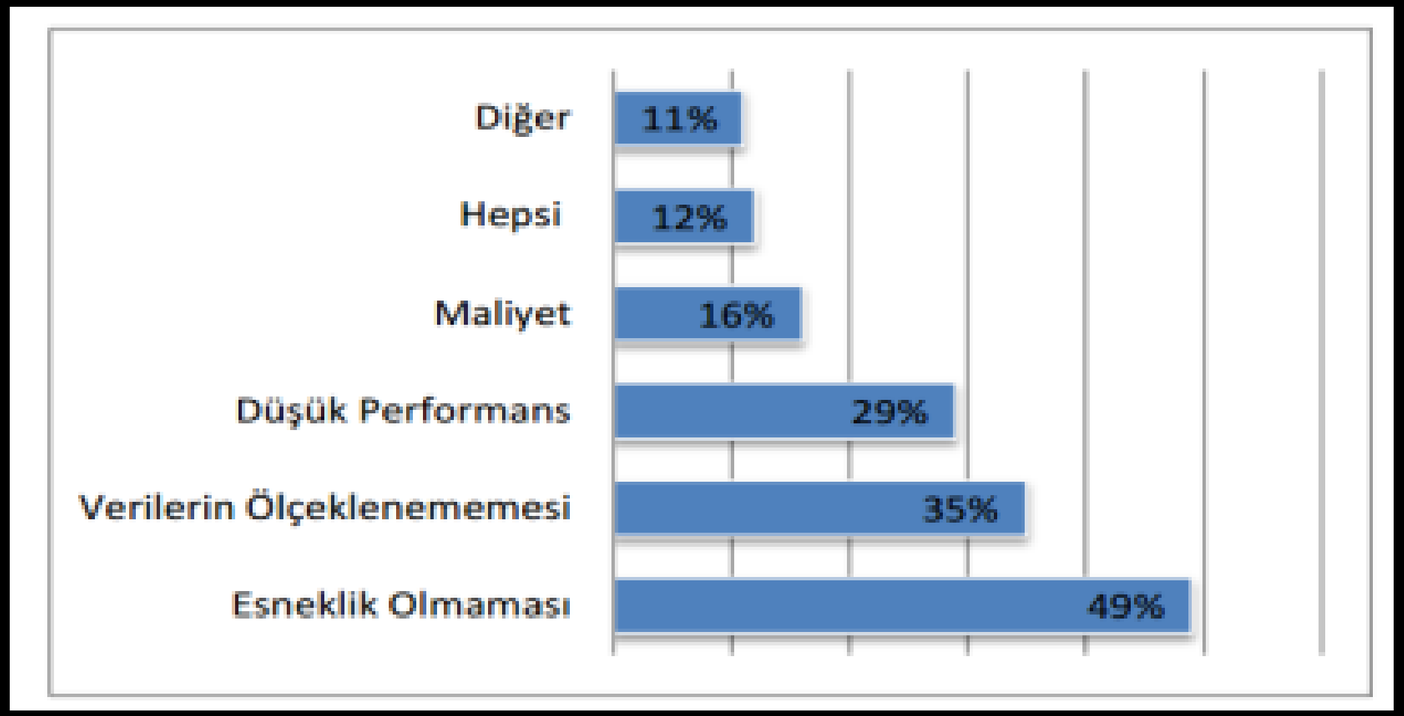
İLİŞKİSEL VE İLİŞKİSEL OLMAYAN (NOSQL) VERİ TABANI SİSTEMLERİ

İLİŞKİSEL VERİ TABANI

İlişkisel veri tabanları, satır ve sütunlardan oluşan, birbiriyle ilişkili tablolardan meydana gelir ve en az iki tablonun birbiriyle ilişkili olması gerekir. ACID prensipleri (Bölünmezlik, Tutarlılık, İzolasyon, Dayanıklılık) bu sistemlerin temel özellikleridir. NoSQL veri tabanları ise, 1998'de Carlo Strozzi tarafından ilişkisel sistemlere alternatif olarak önerilmiş, yatay olarak ölçeklenebilen veri depolama sistemleridir. Özellikle büyük veri depolama ve işleme ihtiyaçlarında, ilişkisel veri tabanlarının sınırlılıklarını aşan NoSQL çözümleri tercih edilir.

İLİŞKİSEL OLMAYAN VERİ TABANI

NoSQL veri tabanları , 1998'de Carlo Strozzi tarafından ilişkisel sistemlere alternatif olarak önerilmiş, yatay olarak ölçeklenebilen veri depolama sistemleridir. Özellikle büyük veri depolama ve işleme ihtiyaçlarında, ilişkisel veri tabanlarının sınırlılıklarını aşan NoSQL çözümleri tercih edilir.



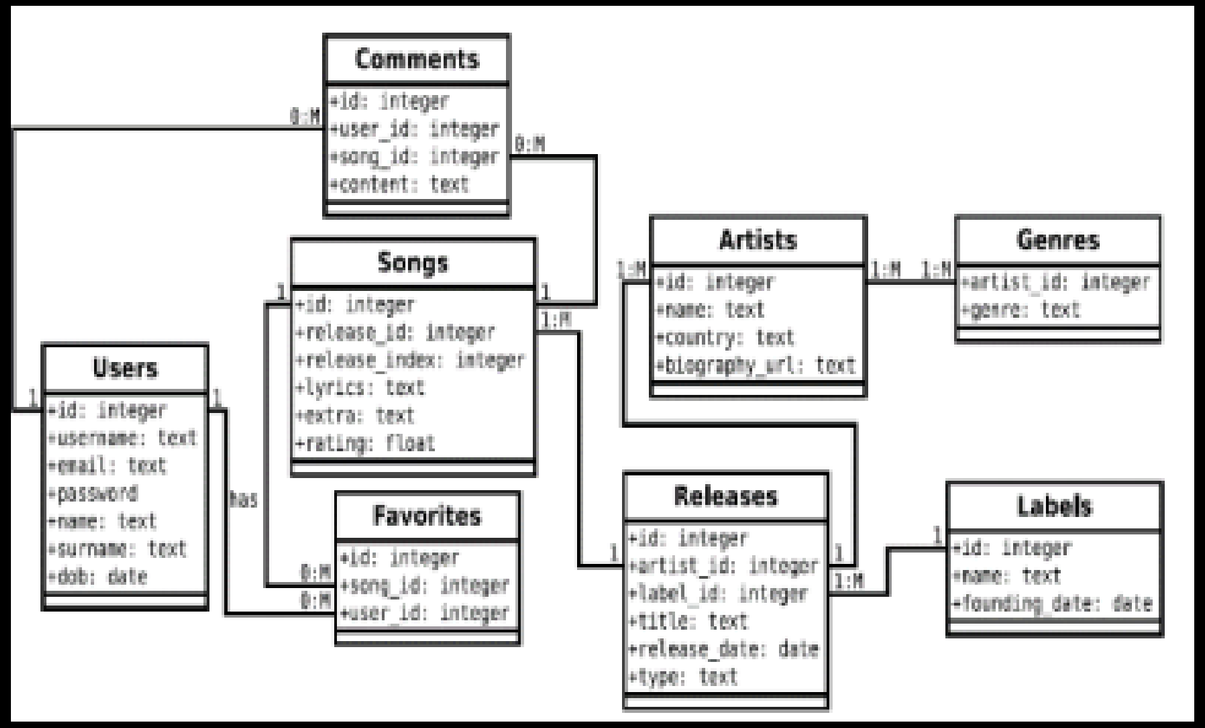
NEDEN NOSQL GEREKLİ SORUSUNA YANIT

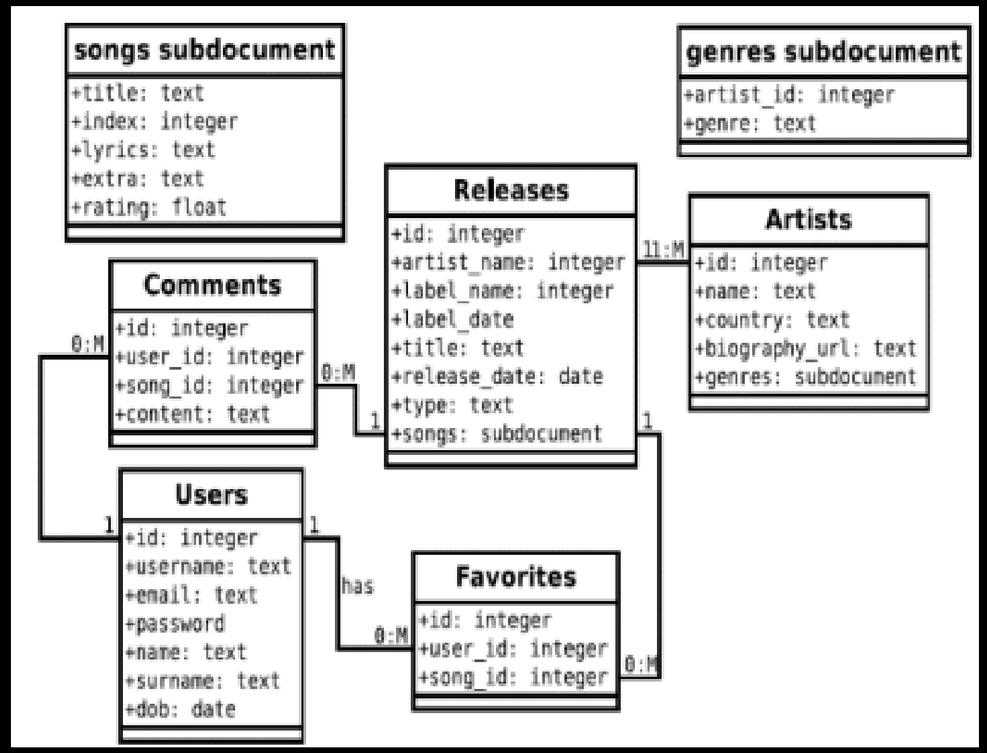
VERİTABANI MİMARİLERİNİN PERFORMANS KARŞILAŞTIRMASI

Bu çalışmada, günümüzde yaygın kullanılan iki veri tabanı sistemi; ilişkisel veri tabanı sistemi olan MySQL ve ilişkisel olmayan (NoSQL) bir alternatif olarak yatay ölçeklendirilebilir MongoDB veri tabanı incelenmiştir. Performans ve yatay ölçeklenebilirlik analizi için, veri tabanı sunucu özelliklerinin belirlenmesi, veri tabanı şemalarının oluşturulması, sorguların belirlenmesi, veri tabanı ayarlarının yapılması, ölçümler ve performans analizi gibi adımlar izlenmiştir. Projede, kullanıcıların müzik tercihleri üzerine önerilerde bulunan bir müzik uygulaması için MySQL ve MongoDB kullanılarak iki farklı veri tabanı şeması tasarlanmıştır.

VERİ TABANI ŞEMASI

Projede iki adet veri tabanı şeması tasarlanmıştır. Biri MySQL , diğeri ise MongoDB veri tabanıdır. Şemalar, diğer kullanıcılara şarkılar önermek için tasarlanmış farklı algoritmalar kullanan bir müzik uygulaması etrafında modellenmiştir.





VERİ TABANI SORGULARI

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

```
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
```

```
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 '
```

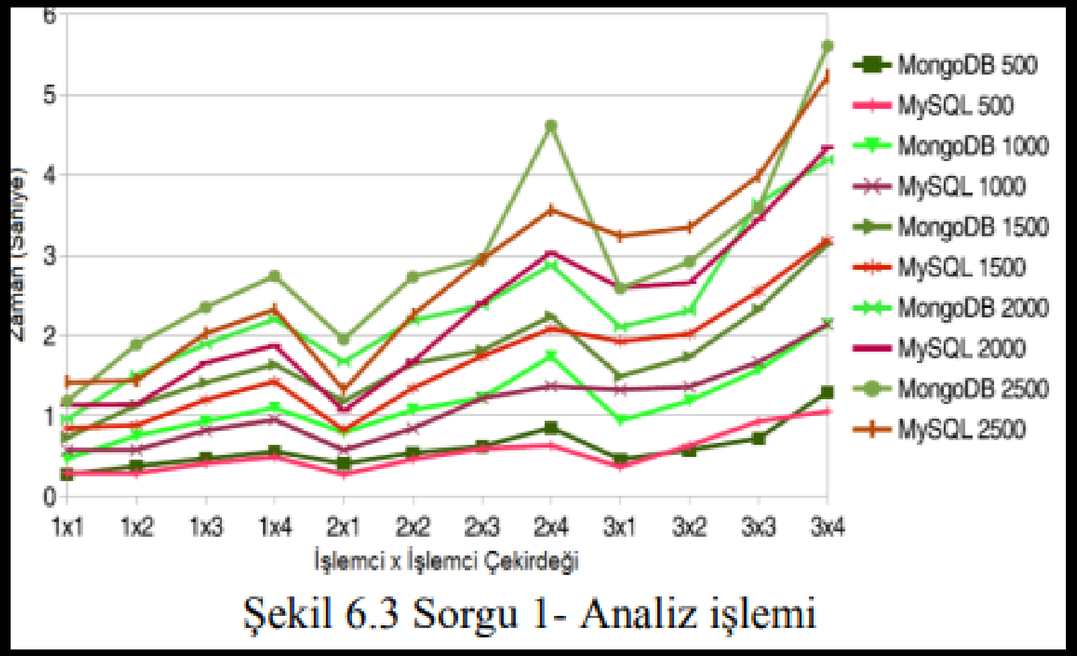
3 sorgu yapılmıştır. Sırasıyla basit,karmaşık,detaylı ve karmaşık.

ÖLÇÜM SONUÇLARI İÇİN 3 YÖNTEM KULLANILMIŞTIR.

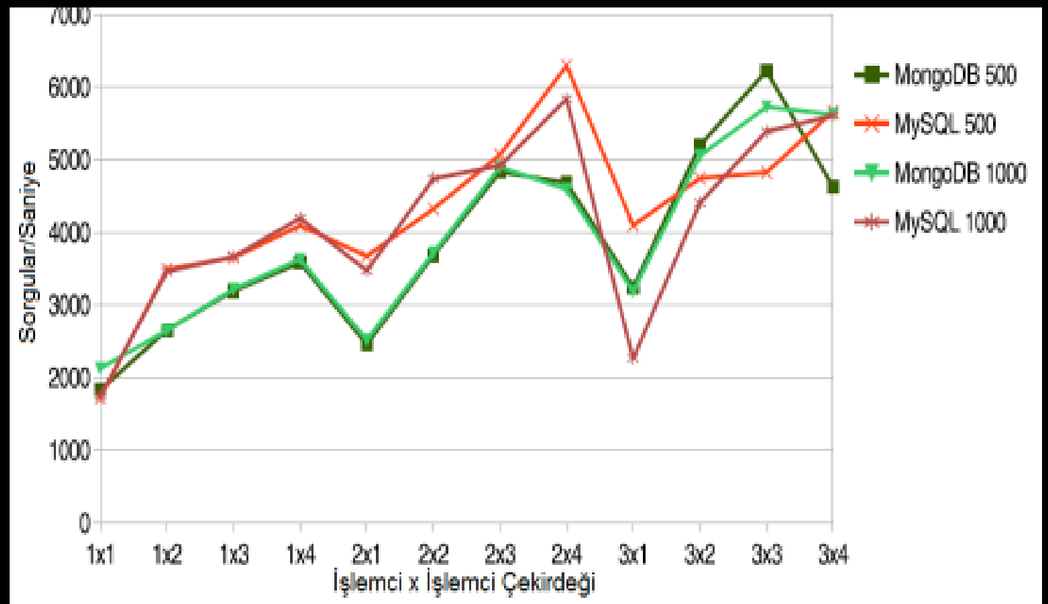
1: Clock() fonksiyonu kullanımı ile belirli bir süre CPU üzerinde harcanan zaman sonuçlarının elde edilmesini sağlamaktır.

2: milisaniye hassasiyetiyle zamanlamaları sağlayan Gettimeofday() fonksiyonu kullanılarak sonuçların elde edilmesini sağlamaktır

3: Slow Query Log (Yavaş sorgu kaydı) olarak adlandırılmaktadır



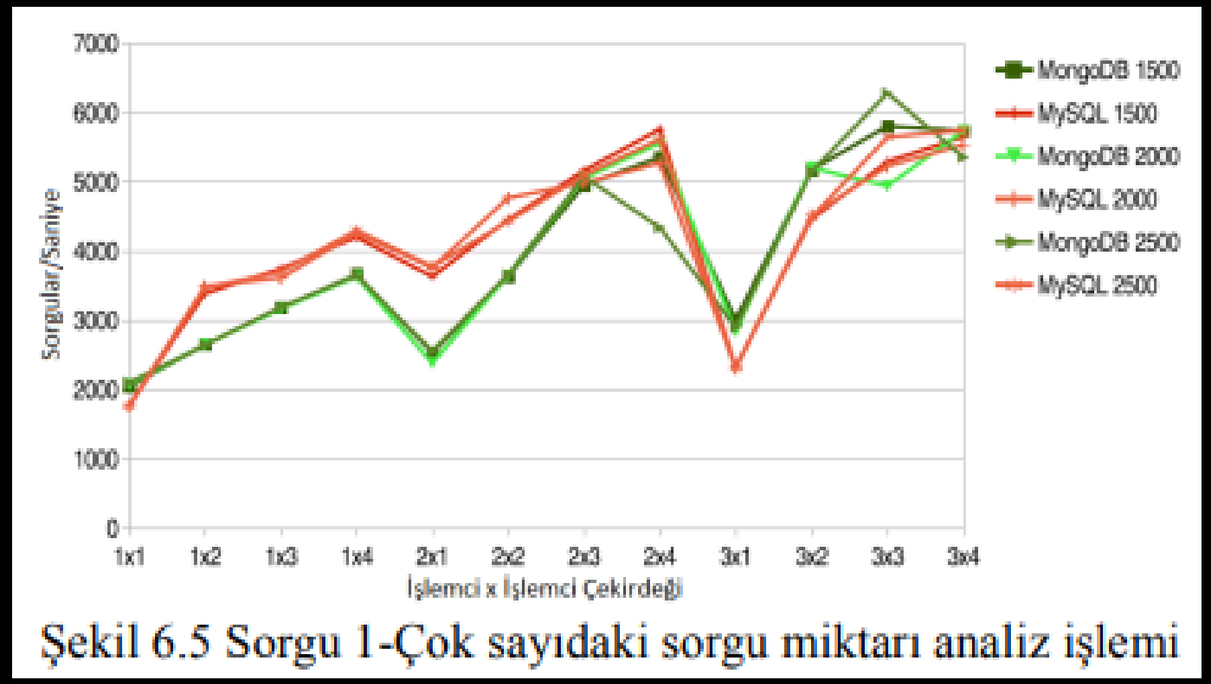
Şekil 6.3 Sorgu 1- Analiz işlemi



Şekil 6.4 Sorgu 1 - Sorgu/saniye analiz işlemi

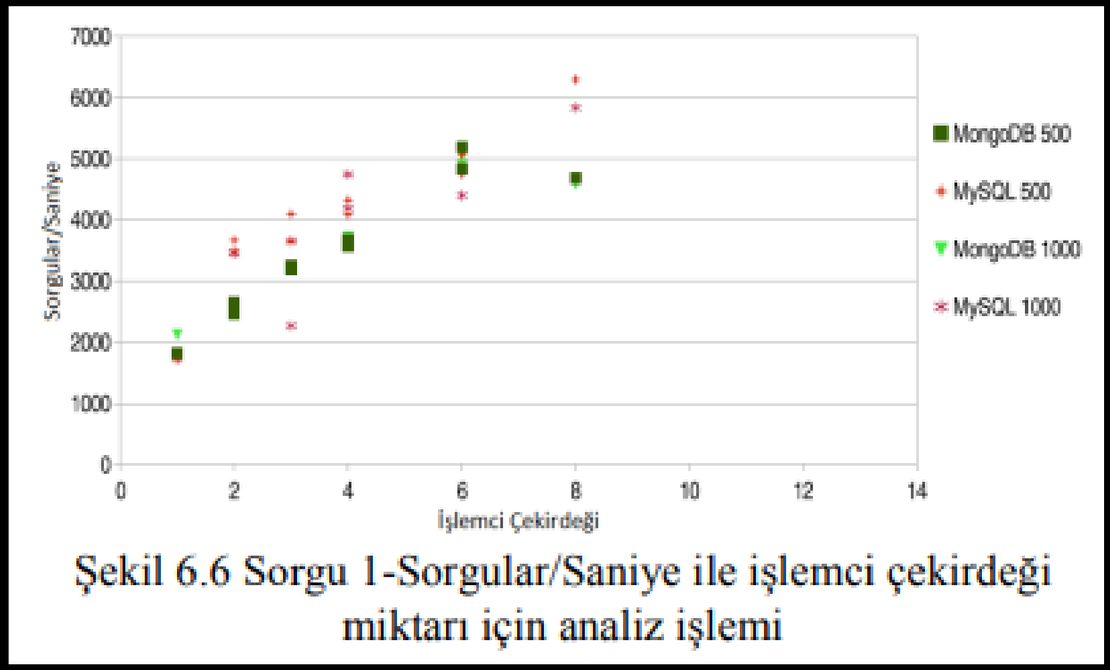
MySQL ve MongoDB veri tabanlarına sorgu 1 (basit sorgu) ile karşılaştırma testi uygulanmıştır.

Ayrıntılı ortalama süre sonuçları



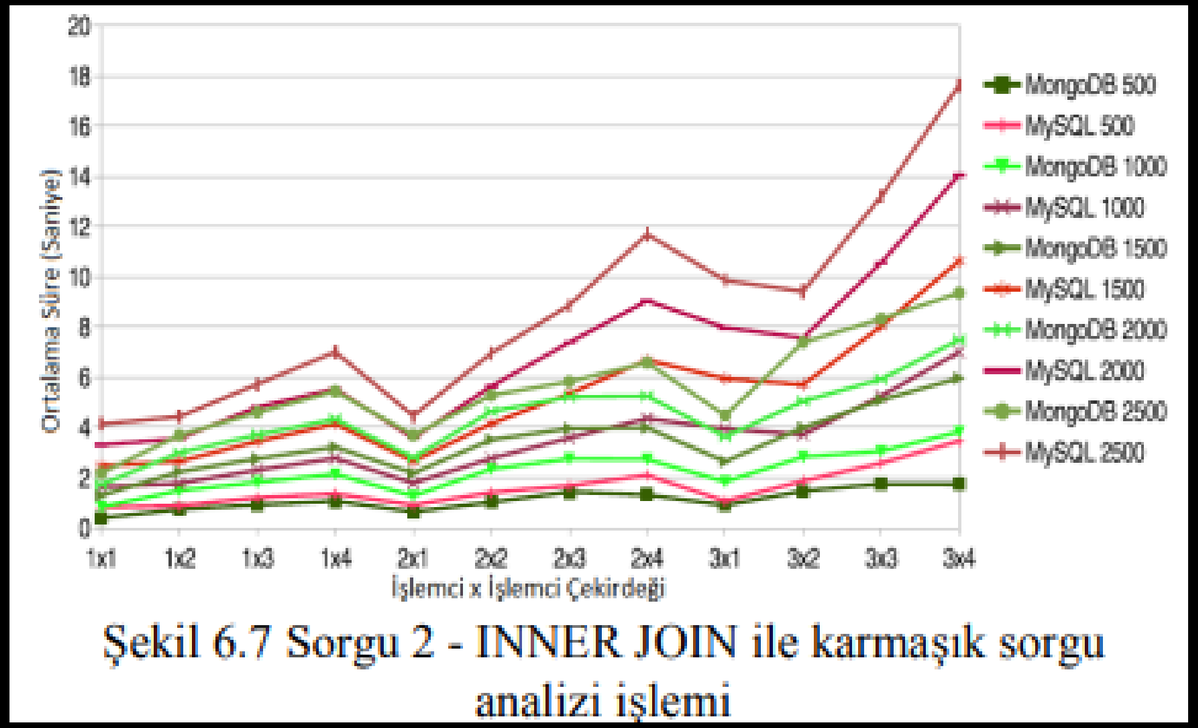
Şekil 6.5 Sorgu 1-Çok sayıdaki sorgu miktarı analiz işlemi

MySQL veri tabanı sisteminin, sorgu sayıları arttığında MongoDB üzerinde avantaj sahibi olduğu görülmektedir.



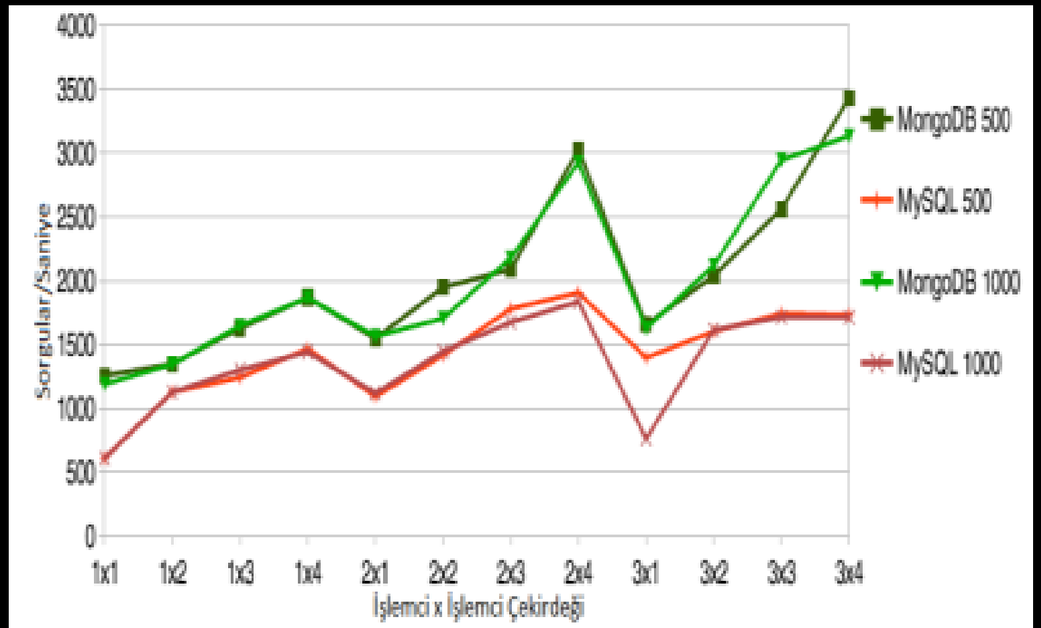
Şekil 6.6 Sorgu 1-Sorgular/Saniye ile işlemci çekirdeği miktarı için analiz işlemi

işlemci çekirdeği miktarı ile saniye başına yapılan sorgu sayıları arasındaki ilişki analizi gösterilmektedir



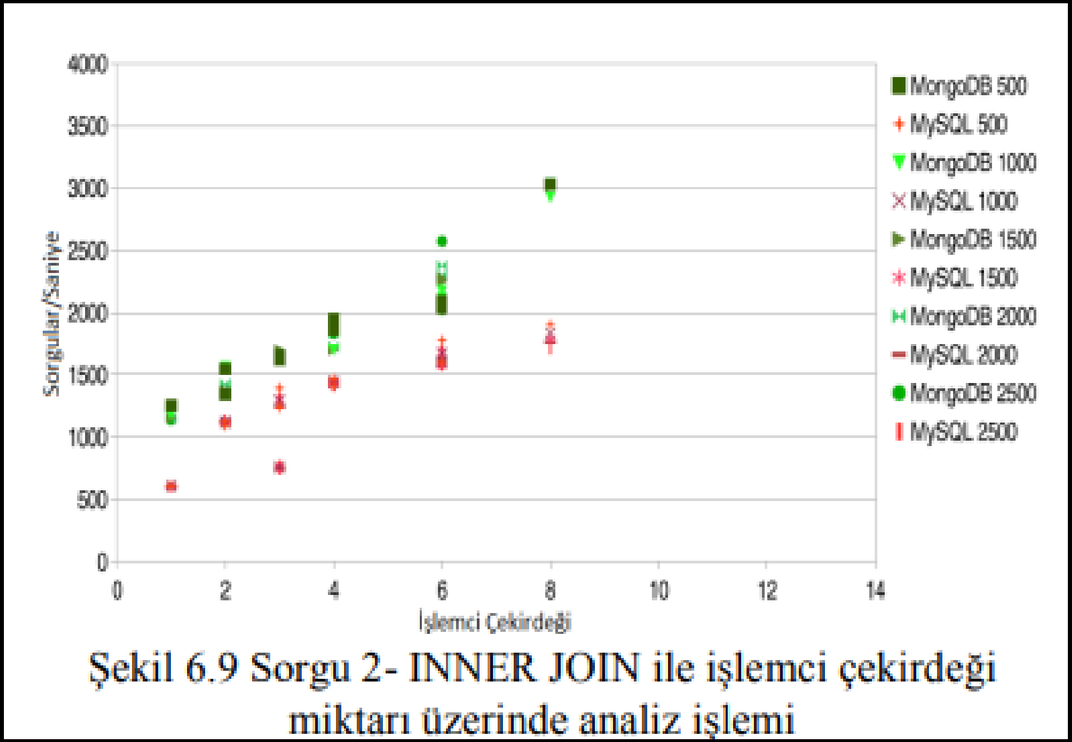
Şekil 6.7 Sorgu 2 - INNER JOIN ile karmaşık sorgu analizi işlemi

MySQL ve MongoDB veri tabanlarına ikinci sorgu kodu ile karşılaştırma testi uygulanmıştır



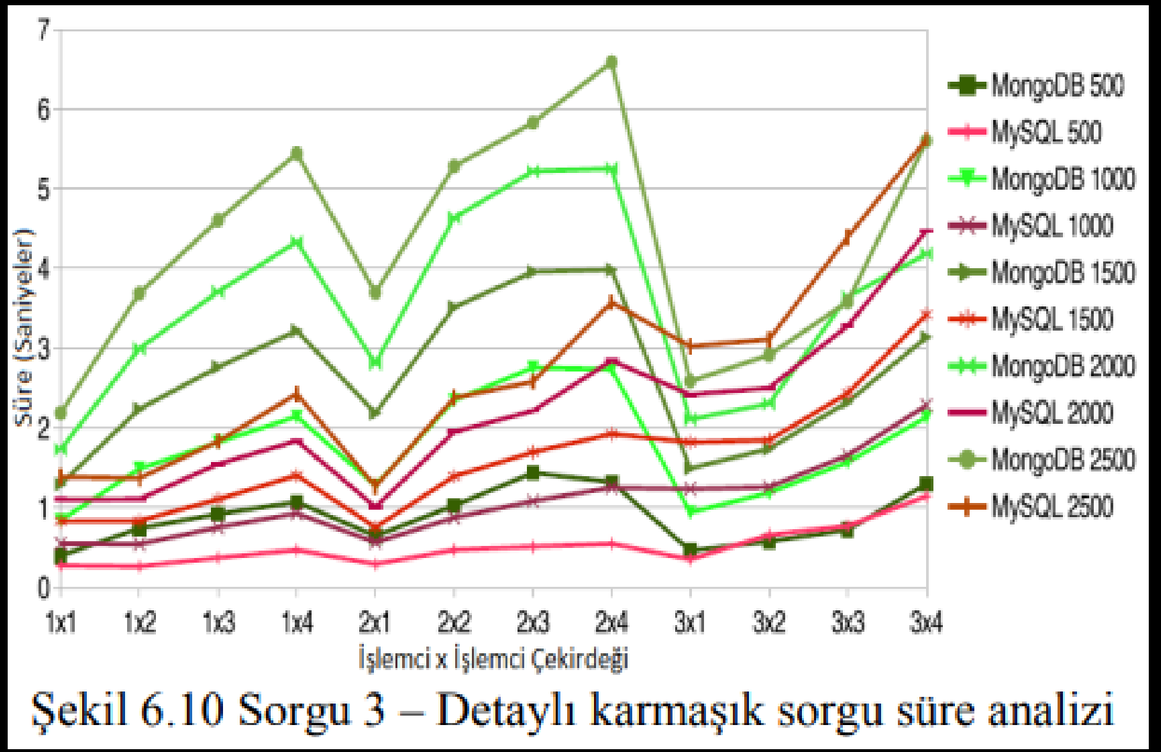
Şekil 6.8 Sorgu 2- INNER JOIN ile 500 ve 1000 veri için sorgu/saniye analizi işlemi

MySQL ve MongoDB veri tabanlarına ikinci sorgu kodu ile karşılaştırma testi uygulanmıştır. Bu test 500 ve 1000 gibi küçük veri kayıtları üzerinde yapılmıştır.



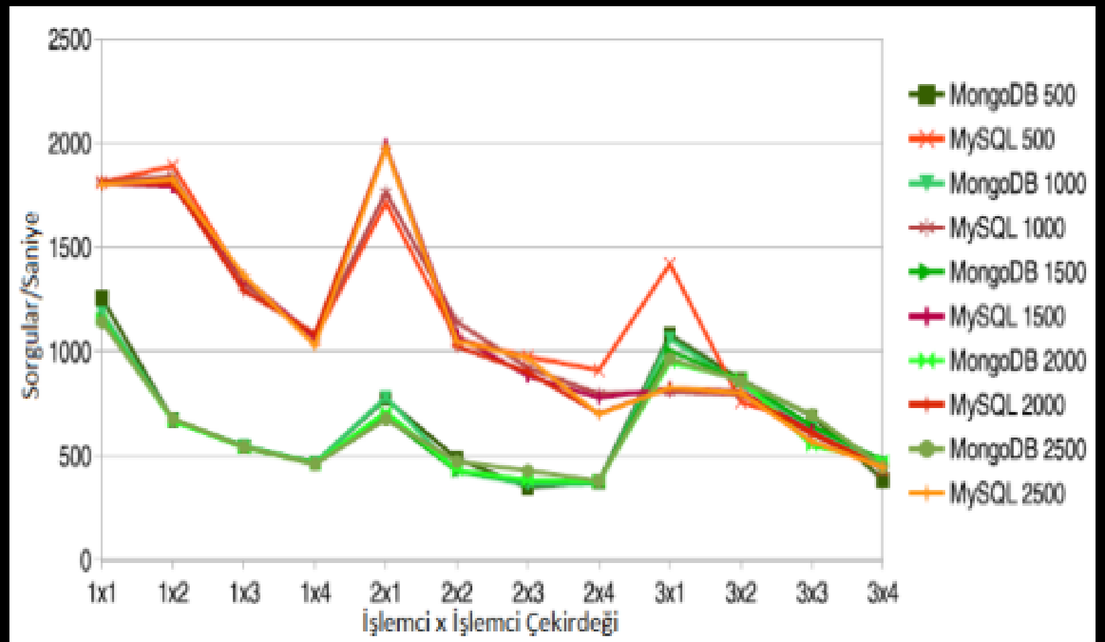
Şekil 6.9 Sorgu 2- INNER JOIN ile işlemci çekirdeği miktarı üzerinde analiz işlemi

İşlemci çekirdeği miktarı ile saniye başına yapılan sorgu sayıları arasındaki ilişki analizi



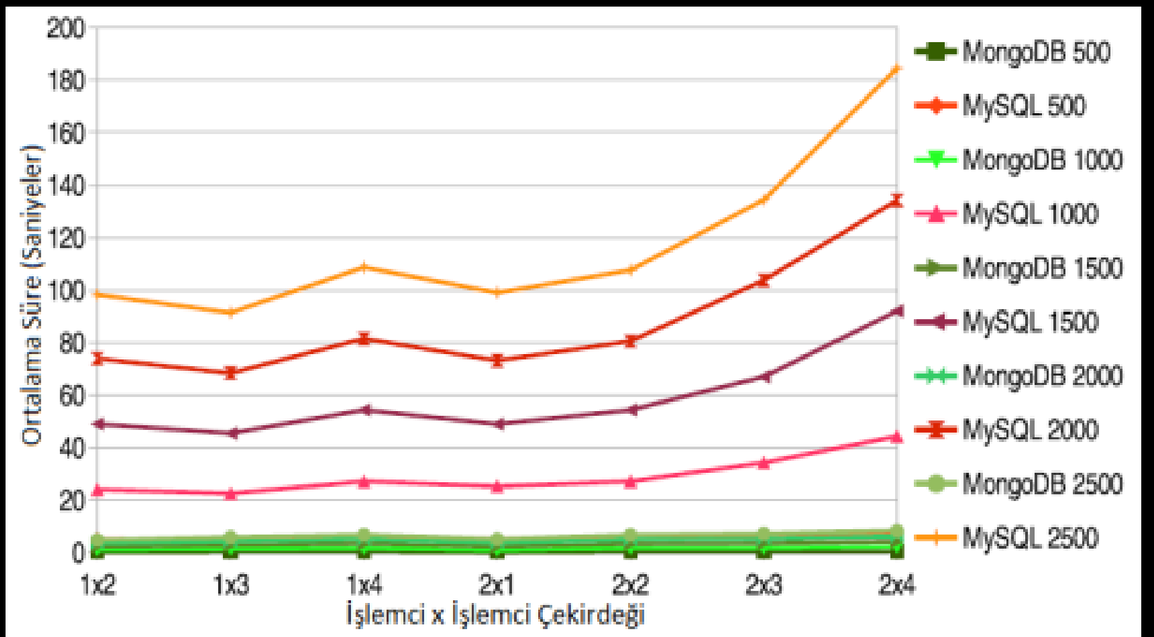
Şekil 6.10 Sorgu 3 – Detaylı karmaşık sorgu süre analizi

İç içe geçmiş "SELECT" ve "WHERE" işlemlerini içeren üçüncü sorgu neticesinde ortaya çıkan performans değerleri gösterilmektedir.



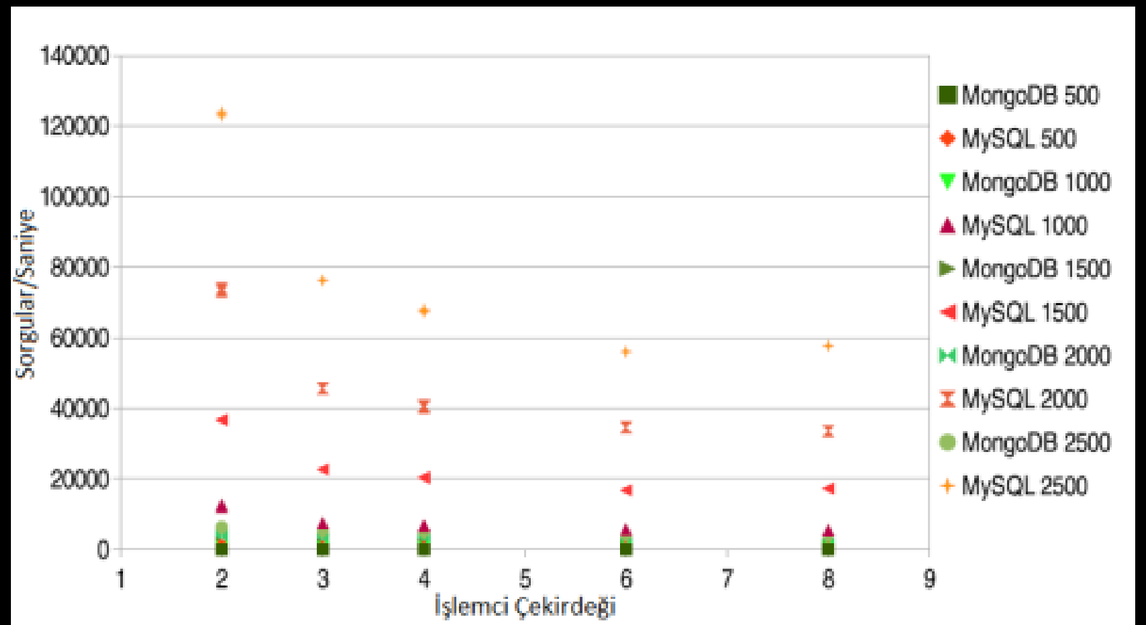
Şekil 6.11 Sorgu 3- Detaylı ve karmaşık sorgu ile Sorgular/saniye analiz işlemi

MySQL ve MongoDB veri tabanlarına üçüncü sorgu kullanılarak uygulanan karşılaştırma testi sonuçları



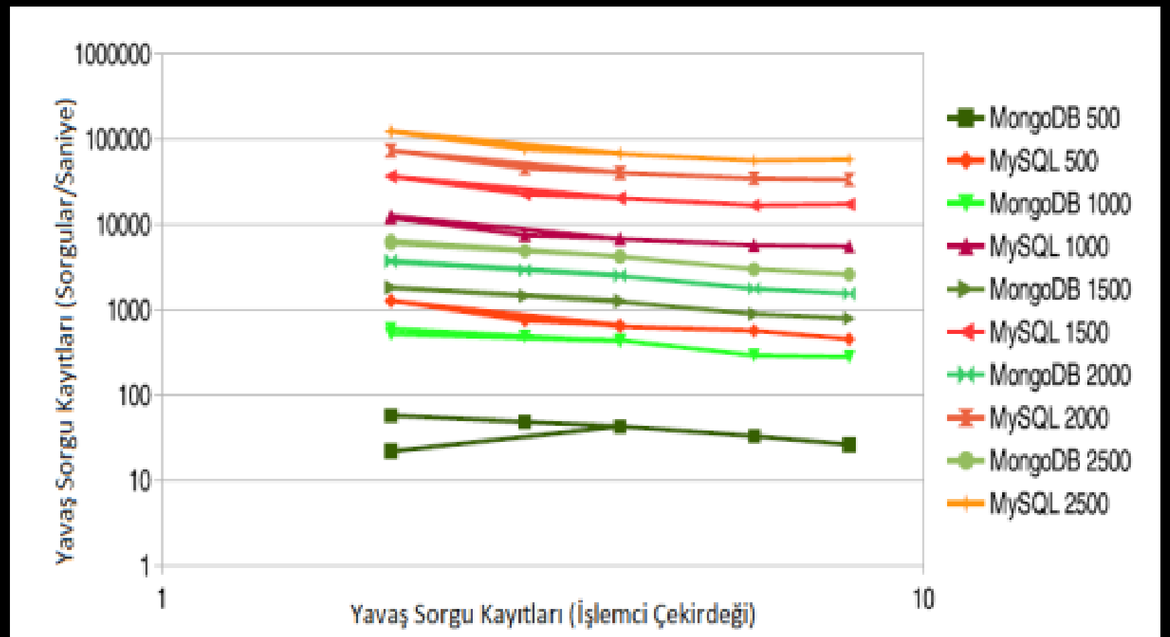
Şekil 6.12 Sorgu 3 – Detaylı ve karmaşık sorgu kodu ile ortalama süre analiz işlemi

MySQL veri tabanı sisteminin MongoDB'ye göre ortalama sorgu süreleri sonuçları, veri kayıt sayısı farkı arttıkça oldukça belirgin bir performans kötülüğü gösterdiği gözlemlenmiştir



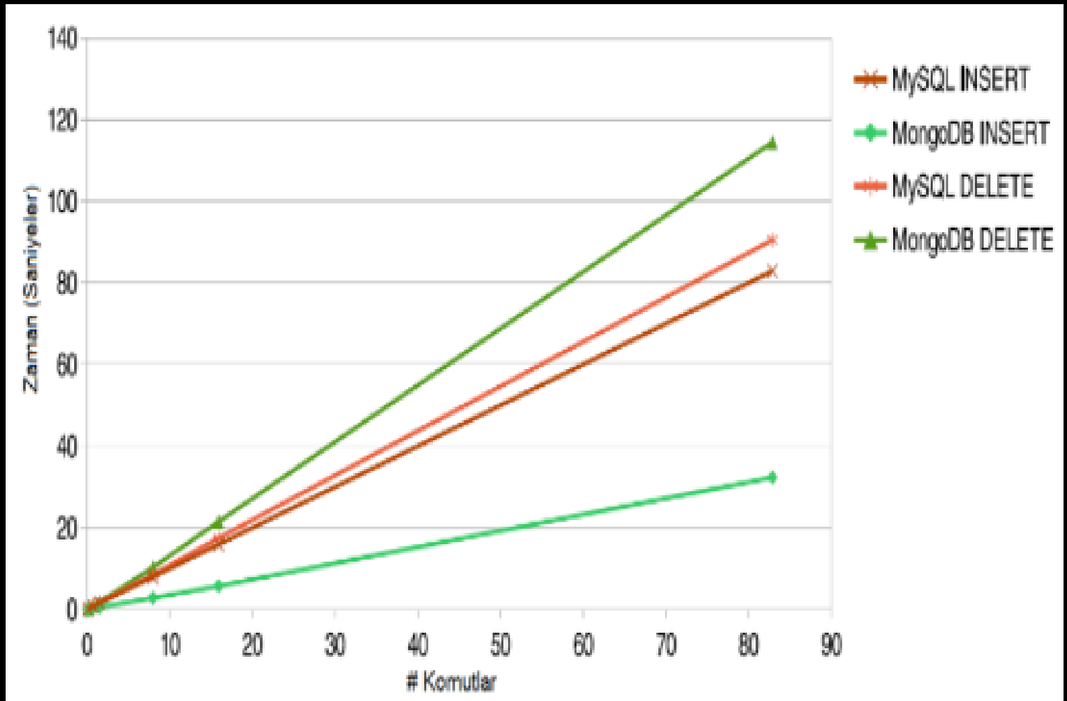
Şekil 6.13 Sorgu 3 - Detaylı ve karmaşık sorgu ile işlemci çekirdeği üzerinde analiz işlemi

MySQL ve MongoDB veri tabanlarına üçüncü sorgu olarak tanımlanan detaylı ve karmaşık sorgu kodu içeren karşılaştırma testi analizi



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

Zamanlama ölçeği büyütülerek veri tabanları sistemleri arasındaki performans farkının Şekil 6.14'de daha anlaşılabilir hale geldiği görülmektedir



Şekil 6.15 INSERT ve DELETE işlemleri

Her iki veri tabanı sisteminin INSERT ve DELETE işlemlerine ait performans grafiği gösterilmektedir.

SONUÇ VE DEĞERLENDİRME

Bu çalışma, yönetim bilişim sistemleri açısından ilişkisel ve NoSQL veri tabanlarının performans karşılaştırmasını yaparak, hangi senaryolarda hangi veri tabanı türünün daha uygun olduğunu analiz etmiştir. Literatürde benzer çalışmalar incelenmiş, MongoDB ve MySQL örneklerinde detaylı testler yapılmıştır. Testler, NoSQL veri tabanlarının büyük veri ve karmaşık sorgularda avantaj sağladığını, ancak işlem hızı gibi bazı durumlarda ilişkisel veri tabanlarının da iyi performans gösterebildiğini ortaya koymuştur. Her iki veri tabanı türünün de avantajları ve dezavantajları olduğu, kullanım durumuna göre en uygun seçeneğin belirlenmesi gerektiği sonucuna varılmıştır.