



**T.C.  
DÜZCE ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**SQL VERİTABANLARINDAN NOSQL VERİTABANLARINA VERİ  
GÖÇÜ UYGULAMASI**

**HATİCE BAHAR  
HANİFE BEYZA NUR HAS**

**LİSANS BİTİRME TEZİ**

**DANIŞMAN  
DR. ÖĞR. ÜYESİ ABDULLAH TALHA KABAKUŞ**

**DÜZCE, 2019**

## BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

17 Mayıs 2019

Adı Soyadı

## **TEŞEKKÜR**

Lisans öğrenimimizde ve bu tezin hazırlanmasında gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocamız Dr. Öğr. Üyesi Abdullah Talha KABAKUŞ'a en içten dileklerimizle teşekkür ederiz.

Bu çalışma boyunca yardımlarını ve desteklerini esirgemeyen sevgili aileme ve çalışma arkadaşşıma sonsuz teşekkürlerimizi sunarız.

**17 Mayıs 2019**

**Adı Soyadı**

## İÇİNDEKİLER

ŞEKİL LİSTESİ.....	VII
ÇİZELGE LİSTESİ.....	VIII
HARİTA LİSTESİ.....	IX
KISALTMALAR.....	X
SİMGELER.....	XI
ÖZET.....	XII
ABSTRACT.....	XIII
1. GİRİŞ.....	1
2. MATERYAL VE YÖNTEM(BU BAŞLIK ZORUNLU DEĞİL)	
ERROR! BOOKMARK NOT DEFINED.	
2.1 ALT BAŞLIK 1.....	ERROR! BOOKMARK NOT DEFINED.
2.1.1 Alt Başlık 2.....	Error! Bookmark not defined.
2.1.2 Alt Başlık 2.....	Error! Bookmark not defined.
2.1.2.1 Alt Başlık 3.....	Error! Bookmark not defined.
2.1.2.2 Alt Başlık 3.....	Error! Bookmark not defined.
2.1.3 Dipnotlar.....	Error! Bookmark not defined.
3. YAPILAN ÇALIŞMALARERROR!	BOOKMARK NOT
DEFINED.	
3.1 ALT BAŞLIK 1.....	ERROR! BOOKMARK NOT DEFINED.
3.1.1 Alt Başlık 2.....	Error! Bookmark not defined.
3.1.1.1 Alt Başlık 3.....	Error! Bookmark not defined.
3.1.1.2 Alt Başlık 3.....	Error! Bookmark not defined.
4. BÖLÜM 4.....	ERROR! BOOKMARK NOT DEFINED.
4.1 ALT BAŞLIK 1.....	ERROR! BOOKMARK NOT DEFINED.
4.1.1 Alt Başlık 2.....	11
4.1.1.1 Alt Başlık 3.....	11
4.1.1.2 Alt Başlık 3.....	Error! Bookmark not defined.
5. BULGULAR VE TARTIŞMA.....	19
5.1 ALT BAŞLIK 1.....	19

5.1.1 Alt Başlık 2.....	Error! Bookmark not defined.
5.2 ALT BAŞLIK 1.....	19
5.2.1 Alt Başlık 2.....	Error! Bookmark not defined.
5.2.1.1 Alt Başlık 3.....	Error! Bookmark not defined.
6. KAYNAKLARIN YAZIMI.....	22
6.1 KAYNAK EKLEME YÖNTEMİ...ERROR! BOOKMARK NOT DEFINED.	
6.2 GENEL KURALLAR.....ERROR! BOOKMARK NOT DEFINED.	
6.2.1 Referans İçerisinde Referans Gösterme.....Error! Bookmark not defined.	
6.3 KAYNAKLARIN LİSTELENMESİNE ÖRNEKLER.ERROR!	
BOOKMARK NOT DEFINED.	
7. SONUÇLAR VE ÖNERİLER.....	24
KAYNAKLAR.....	ERROR! BOOKMARK NOT DEFINED.
ÖZGEÇMİŞ.....	27

## ŞEKİL LİSTESİ

### Sayfa No

Şekil 2.1. Manzara.....	Error! Bookmark not defined.
Şekil 2.2. Manzara 2.....	Error! Bookmark not defined.
Şekil 2.3. Manzara 3.....	Error! Bookmark not defined.
Şekil 2.4. Manzara 4.....	Error! Bookmark not defined.
Şekil 3.1. Manzara 5.....	Error! Bookmark not defined.
Şekil 4.1. Manzara 6.....	Error! Bookmark not defined.

## ÇİZELGE LİSTESİ

### Sayfa No

Çizelge 2.1. Birimler 1.....	Error! Bookmark not defined.
Çizelge 2.2. Birimler 2.....	Error! Bookmark not defined.
Çizelge 3.1. Birimler 3.....	Error! Bookmark not defined.
Çizelge 4.1. Birimler 4.....	Error! Bookmark not defined.
Çizelge 5.1. Birimler 5.....	Error! Bookmark not defined.
Çizelge 5.2. Birimler 6.....	Error! Bookmark not defined.

## HARİTA LİSTESİ

### Sayfa No

Harita 2.1 Türkiye solar haritası 1.....	Error! Bookmark not defined.
Harita 3.1 Harita 2.....	Error! Bookmark not defined.
Harita 4.1 Harita 3.....	Error! Bookmark not defined.



## KISALTMALAR

AcO  
ABS  
BBP  
DEGA  
DOP

Asetat  
Akrilonitril/bütadien/stiren  
Benzil bütil ftalat  
Dietilenglikol adipat  
Dioktil ftalat

## SİMGELER

G	İletkenlik
P	Güç (elektrik akımı için)
S	Siemens
$\sigma$	Öz İletkenlik
$\Omega$	Ohm

## ÖZET

# SQL VERİTABANLARINDAN NOSQL VERİTABANLARINA VERİ GÖÇÜ UYGULAMASI

Öğrenci Adı SOYADI

Hatice BAHAR

Hanife Beyza Nur HAS

Düzce Üniversitesi

Mühendislik Fakültesi

Bilgisayar Mühendisliği Bölümü

Lisans Bitirme Tezi

Danışman: Dr. Öğr. Üyesi Abdullah Talha KABAKUŞ

Mayıs 2019, 58 sayfa

Bu proje, bir SQL veritabanından, NoSQL veritabanına veri göçü sağlayabilmektir ve bunu, JAVA programlama dilini kullanarak yapabilmektedir. Göçü sağlamak istenen herhangi bir SQL veritabanı için, kurulu olan MySQL nerede olursa olsun bulunduğu ortamın host adresini karşıdan alarak, NoSQL tarafına ulaşması istenen verilerin, MongoDB'nin kurulu olduğu ortamın host adresini de karşıdan alarak bu iki host arasında veri akışını sonuca ulaştırabilmektir. Bunun sebebi, herhangi tablo, sütun benzeri bir sınırlandırmaya tabi olmadan string ,integer, date, image vb. istenilen tipteki verileri tip ayrımı olmaksızın toplu halde, bir sunucuda bulunan SQL veritabanından farklı bir sunucu üzerinde bulunan NoSQL veritabanına göçünü sağlayarak ilişkisel veritabanındaki kısıtlamalardan sıyrılmak, NoSQL veritabanlarının sunduğu daha hızlı, esnek ve akışkan bir yapı da zaman ve performans kazancı sağlamak. Bu projede yukarıda da belirtildiği gibi SQL veritabanlarından MySQL, NoSQL veritabanlarından MongoDB veritabanı kullanılmıştır. Sonuç olarak MySQL veritabanında mevcut olan veritabanlarının, MongoDB tarafında veritabanı oluşturarak verilerin bu veritabanına göçü sağlanmıştır.

**Anahtar sözcükler:** NoSQL Veritabanları, SQL Veritabanları, Veri Analizi, Büyük Veri, İlişkisel Veritabanları, Göç, Veri Analitiği, Veri Adaptörü, MongoDB, MySQL, Veri Taşıma, Veri Yönetimi, Veri Modeli, İlişkisel Veri Tabanı Taşıma, Büyük Veri İşleme

## ABSTRACT

### DATA MIGRATION APPLICATION FROM SQL DATABASES TO NOSQL DATABASES

Student Name SURNAME

Hatice BAHAR

Hanife Beyza Nur HAS

Düzce University

Engineering Faculty, Department of Computer Engineering

Undergraduate Graduation Thesis

Supervisor: PhD Abdullah Talha KABAKUŞ

May 2019, 58 pages

By providing migration from the SQL database on a server to a NoSQL database on a different server, a faster, more flexible, and more fluid structure that NoSQL databases offer, such as time and performance Gain. In this project, the MongoDB database from MySQL, NoSQL databases from SQL databases is used. As a result, databases that exist in the MySQL database are migrated to this database by creating a database on the MongoDB side.

**Keywords:** NoSQL Databases, SQL Databases, Data Analysis, Big Data, Relational Databases, Migration, Data Analytics, Data Adapter, MongoDB, MySQL, Data Migration, Data Management, Data Model, Relational Database Migration, Big Data Handlin


# 1. GİRİŞ

Geleneksel veritabanları, artan büyük veri işleme ve işleme taleplerine yetişmek için yetersizdir. Geleneksel ilişkisel veritabanlarında veri modeli sınırlamaları, mimari sınırlamalar, ölçeklenebilirlik ve performans sınırlamaları vardır. Yeni nesil NoSQL veritabanı, şema içermeyen tasarım, yüksek kullanılabilirlik, konum bağımsızlığı, ölçeklenebilirlik gibi özelliklere sahiptir ve bu da büyüyen veri yönetimi taleplerini karşılamayı verimli hale getirir. Günümüzde birçok kuruluş MongoDB'yi daha hızlı uygulama geliştirme ve yüksek oranda büyüyen çok çeşitli verilerin verimli bir şekilde kullanılması için benimsiyor. Bu yazıda, taşınan veriler üzerinde MySQL'den MongoDB'ye verilerin taşınması için etkili bir yaklaşım öneriliyor. []

Şu anda, SQL veritabanlarını NoSQL veritabanlarına dönüştürmek için varolan bir yardımcı program yoktur. NoSQL yeni bir eğilim; İstenen dönüşüm için tasarlanmış yazılım veya yardımcı programlar yoktur. Geliştirilen yardımcı programların çoğu, bir SQL veritabanının bir formunu başka bir SQL veritabanına dönüştürmek için geliştirilmiştir. []

Bu nokta da devreye giren bu projede amaç, bir SQL veritabanından NoSQL veritabanına veri göçü sağlayabilmektir ve bunu, JAVA programlama dilini kullanarak başarabilmektir. JAVA'da hazırlanan bu programın çalıştırılması sonucunda, göçü sağlanmak istenen herhangi bir SQL veritabanı için kurulu olan MySQL nerede olursa olsun bulunduğu ortamın host adresini, NoSQL tarafına ulaşması istenen verilerin MongoDB'nin kurulu olduğu ortamın host adresini de karşıdan istediği görülür. Bu host adreslerinin bilgisi alınıp, bu iki host arasında veri akışını sonuca ulaştırabilmek ve veri göçü işlemini alınan bu hostlar arasında gerçekleştirebilmek hedeflenmiştir. Bunun sebebi, bir sunucu üzerinde bulunan SQL veritabanından farklı bir sunucu üzerinde bulunan NoSQL veritabanına göçü sağlanarak ilişkisel veritabanındaki kısıtlamalardan sıyrılma, NoSQL veritabanlarının sunduğu daha hızlı, esnek ve akışkan bir yapı da zaman ve performans kazancı sağlamak. Bu projede yukarıda da belirtildiği gibi SQL veritabanlarından MySQL, NoSQL veritabanlarından MongoDB veritabanı kullanılmıştır. Sonuç olarak MySQL veritabanında mevcut olan veritabanlarının,

MongoDB tarafında veritabanı oluşturarak verilerin bu veritabanına göçü sağlanmıştır.

Hazırlanan bu proje sonunda ilişkisel veritabanlarının veri modeli sınırlamaları, mimari sınırlamalar, ölçeklenebilirlik ve performans sınırlamaları dışına çıkıp, veri tabanı teknolojisinde yeni bir konsept olan NoSQL'in sağladığı büyük miktarda veri desteği, iyi bir veri depolama ve erişim mekanizma olma özelliği dışında hızlı performansını ve verimliliğini kullanabilir bir duruma gelinecektir. 



## 2. ~~YAPILAN PROJELER~~

- I. **Proje Adı:** SQL sisteminin NoSQL sistemine dönüştürülmesi ve SVM kullanarak veri analitiği gerçekleştirme (Transformation of SQL system to NoSQL system and performing data analytics using SVM) [<https://ieeexplore.ieee.org/document/8300833>]

**Yazar(lar):** Sanket Ghule, Ramkrishna Vadali

**Tarih:** 11-12 Mayıs 2017

**Proje Özeti:** Son on yıl internet üzerinden veri geçişi büyük miktarda kaydeder. Bugünün dünya veritabanları büyük boyutta olan sosyal ağ siteleri ile bağlanır. Şu anda, SQL veritabanlarını NoSQL veritabanlarına dönüştürmek için varolan bir yardımcı program yoktur. NoSQL yeni bir eğilim; İstenen dönüşüm için tasarlanmış yazılım veya yardımcı programlar yoktur. Geliştirilen yardımcı programların çoğu, bir SQL veritabanının bir formunu başka bir SQL veritabanına dönüştürmek için geliştirilmiştir. İlişkisel veritabanı ve NoSQL veritabanı dahil olmak üzere hibrit veritabanı mimarisini tanıtmak için bir veri bağdaştırıcısı sistemi önerilir. Veri eşitleme mekanizması kullanarak veritabanı dönüşümleri ile başa çıkmak, ölçeklenebilir nesne depolama sistemlerinde etkin veri analizlerini etkinleştirerek odaklanır. Varolan veriler üzerinde yerinde harita azaltma hesaplaması gerçekleştirmek için bir nesne tabanlı depolama kümesinde bir veri analizi katmanı uygulaması vardır. Bu ~~ASIT~~ özellikleri olarak RDBMS güçlü tutar ve aynı zamanda tüm çalışan işlemlerin izini tutar ve eşzamanlı işlemler ile diğer katmanları ile yönetir bir orta katman aracılığıyla NoSQL yararları sağlayarak, veri bağdaştırıcısı, özgün uygulama değişmeden kalması için uygulama hizmetlerinden sorguları kabul eden bir SQL katmanı kullanır. Veri bağdaştırıcısı, veritabanı dönüşümleri sırasında sorgu akışını da denetler. Herhangi bir yere veri taşımadan ve bu yaklaşımın sonuçları çözümlendiğinde, nesne depolama sistemindeki

varolan büyük ölçekli veriler üzerinde hesaplama gerçekleştiren bir yaklaşım. Bu anahat, ölçeklenebilirlik ve NoSQL veritabanları kullanılabilirliğini etkilemeden sıkı tutarlılık sağlayabilir.

- II. **Proje Adı:** SQL Veritabanının NoSQL'e Şema Dönüştürme Modeli (Schema Conversion Model of SQL Database to NoSQL)

[<https://ieeexplore.ieee.org/document/7024609>]

**Yazar(lar):** Gansen Zhao, Qiaoying Lin, Libo Li, Zijing Li

**Tarih:** 8-10 Kasım 2014

**Proje Özeti:** NoSQL veritabanlarının artan olgunluğunun yanı sıra, büyük miktarda veri üzerine yazmaktan daha fazlasını okumak durumunda, birçok uygulama NoSQL'e başvuruyor ve veri depolama sistemi olarak seçiyor. SQL veritabanından NoSQL'e geçiş yapmak ve verimli sorgu sağlamak zorunlu hale geliyor. Bununla birlikte, birleştirme işlemi NoSQL veritabanında desteklenmez ve birden fazla kez ayrı olarak okunur; bu da gösteriyorki performansın düşük olması kaçınılmazdır. Bu makale, SQL veritabanını NoSQL'e dönüştürmek için ilgili tabloları yerleştirme işleminde yüksek performans sağlayabilen bir şema dönüşüm modeli ve doğru yerleştirilmiş bir sıra sunarak bir tablodaki birleştirme sorgusunun tüm gerekli içeriğini içeren grafik dönüştürme algoritması önermektedir.

- III. **Proje Adı:** SQL ve NoSQL veritabanı arasında sorgulama ve dönüşüm için veri adaptörü (Data adapter for querying and transformation between SQL and NoSQL database) [<https://www.sciencedirect.com/science/article/pii/S0167739X16300085>]

**Yazar(lar):** Ying-Ti Liaoa, Jiazheng Zhoua, Chia-HungLu, Shih-ChangChen, Ching-Hsien Hsu, Wenguang Chen, Mon-Fong Jiange, Yeh-Ching Chunga

**Tarih:** Aralık 2016

**Proje Özeti:** Bu makale, İlişkisel Veri Tabanı (RDB) ve NoSQL sistemlerinde çok yapılandırılmış verilerin otomatik dönüşümünü mümkün kılmak için veri adaptörü sunar. Önerilen veri bağdaştırıcısı ile karma veritabanı sistemlerinin oluşturulması için kesintisiz bir mekanizma sağlanmıştır. Önerilen veri adaptörüyle, hibrit veri tabanı sistemleri elastik bir şekilde gerçekleştirilebilir, yani erişim, veri boyutuna bağlı olarak RDB veya NoSQL olabilir. Bu makale, dönüştürme modunu engelleme (BT modu), boşaltma modunu engelleme (BD modu) ve doğrudan erişim modu (DA modu) ile hızı ve RDB, NoSQL veritabanında ve geçici dosyalarda depolanan verilerde kısmen

çeşitliliğe odaklanmaktadır. nerilen veri adaptör sistemi ile RDB ve NoSQL veritabanlarını aynı anda kullanmak için kesintisiz bir mekanizma sağlayabiliriz.

- IV. **Proje Adı:** SQL ve NoSQL veritabanı sistemleri için tek tip veri erişim platformu (Uniform data access platform for SQL and NoSQL database systems) [<https://www.sciencedirect.com/science/article/abs/pii/S0306437916303398>]

**Yazar(lar):** Ágnes Vathy-Fogarassy, Tamás Húgyák

**Tarih:** Eylül 2017

**Proje Özeti:**Heterojen (ilişkisel ve NoSQL) veritabanı sistemlerinden gelen verilerin aynı anda sorgulanmasına izin veren yeni bir metamodel tabanlı veri entegrasyon yaklaşımı önerilmiştir. Sunulan yöntem, kaynak sistemlerin yapısal, anlamsal ve sözdizimsel heterojenliklerini kapsar. Sunulan yöntem, kullanıcıların ilişkisel veritabanı yönetim sistemlerinden (RDBMS'ler) ve MongoDB'den veri sorgulamalarını sağlayan web tabanlı bir uygulama geliştirilerek deneysel olarak değerlendirildi.

- V. **Proje Adı:** Sağlık sektörü analitik ve yönetimi için sağlık ilişkisel veritabanının NoSQL bulut veritabanına taşınması (Migration of healthcare relational database to NoSQL cloud database for healthcare analytics and management) [<https://www.sciencedirect.com/science/article/pii/B9780128153680000026>]

**Yazar(lar):** Dimpal Tomar, Jai Prakash Bhati, Pradeep Tomar, Gurjit Kaur

**Tarih:** 2019

**Proje Özeti:** EKG'ler, X-ışınları, BT taramaları, ultrason raporları, laboratuvar raporları, radyoloji testleri ve Elektronik Sağlık Kayıtları ve Elektronik şeklinde klinik araştırmalar gibi çeşitli kaynaklardan üretilen sağlık sektöründe katlanarak artan veri hacmi nedeniyle Tıbbi Kayıtlar, ilişkisel veritabanları, Web servisinin kullanılabilirliği ve ölçeklendirme gibi gereklilikleri makul şekilde karşılayamaz. Daha fazla kullanıcı ve veri kullanımı, büyük sunucuların eklenmesi anlamına gelir ve büyük sunucular çok karmaşık ve son derece pahalıdır. Dolayısıyla, kuruluşlar mevcut veya yeni uygulamalar için şu anda kullandıkları ilişkisel veritabanları ile performans problemleriyle karşı karşıya kaldıklarından, özellikle kullanıcı sayısı arttıkça, daha hızlı ve daha esnek bir veritabanı katmanı gerekliliğinin farkındalar. Bu, NoSQL (SADECE DEĞİLDİR) bulut veritabanı için ilişkisel veritabanı sistemlerinden bulut tabanlı bir mimariye geçmenin tam zamanıdır. Daha fazla uygulamayı genişletmek ve daha fazla kullanıcıyı desteklemek için NoSQL bulut veritabanı benimsendi. NoSQL bulut veritabanları,



hizmetlerin performansında ve kullanılabilirliğinde bir artışa olanak tanır; bu nedenle, Structured Query Language (SQL) veritabanlarından NoSQL bulut veritabanlarına veri geçişi hakkında zamanında bilgi iletmek uygun görünmektedir. İlk olarak, veri göçünün zorluklarını ve analizini ve ayrıca NoSQL teknolojisini kullanarak ilişkisel veritabanı yönetim sisteminden (RDBMS) NoSQL'e veri taşıma tekniklerini sunuluyor. İkinci olarak, NoSQL bulut tabanlı teknolojiyle ilgili teknolojik altyapıyı, uygulamaları ve zorlukları tartışılıp, veritabanları için bulut tabanlı mimariler sunuluyor.

- VI. **Proje Adı:** İlişkisel Veritabanının Belge Yönelimli Veritabanına Göçü: Yapı Denormalizasyonu ve Veri Dönüşümü (Migration of Relational Database to Document-Oriented Database: Structure Denormalization and Data Transformation) [<https://ieeexplore.ieee.org/document/7311143/authors#authors>]

**Yazar(lar):** Girts Karnitis, Guntis Arnicans

**Tarih:** 3-5 Haziran 2015

**Proje Özeti:** İlişkisel veritabanları lider veri depolama teknolojisi olmaya devam etmektedir. Bununla birlikte, birçok şirket bulut bilişim teknolojilerini kullanan ölçeklenebilir uygulamalar yapmak için işletme giderlerini azaltmak istiyor. NoSQL veritabanının kullanılması olası çözümlerden biridir ve NoSQL pazarının önümüzdeki beş yıl içinde yaklaşık yüzde 50 CAGR'da büyüyeceği tahmin edilmektedir. Rapor, ilişkisel bir veritabanından doküman odaklı bir veritabanına hızlı veri geçişi için bir çözüm sunar. Fiziksel veriler üzerinde yarı otomatik olarak iki mantıksal seviye yarattık. Kullanıcılar oluşturulan mantıksal veri modelini geliştirebilir ve gerekli her belge için veri taşıma şablonunu yapılandırabilir. Veri taşıma özellikleri ilişkisel veritabanı tarayıcısı Dig Browser içine uygulanır. Gerçek hastaların veri tabanı Clasterpoint veri tabanına taşındı.

- VII. **Proje Adı:** Mapreduce Yaklaşımına Dayalı Naif Bayes Sınıflandırıcısı Kullanarak İlişkisel Veritabanının MongoDB ve Data Analytics'e Göç Edilmesi (Migration of Relational Database to MongoDB and Data Analytics using Naive Bayes Classifier based on Mapreduce Approach) [<https://ieeexplore.ieee.org/document/8463830>]

**Yazar(lar):** Ganesh B. Solanke, K. Rajeswari

**Tarih:** 17-18 Ağustos 2017

**Proje Özeti:** Geleneksel veritabanları, artan büyük veri işleme ve işleme taleplerine yetişmek için yetersizdir. Geleneksel ilişkisel veritabanlarında veri modeli sınırlamaları,

mimari sınırlamalar, ölçeklenebilirlik ve performans sınırlamaları vardır. Yeni nesil NoSQL veritabanı, şema içermeyen tasarım, yüksek kullanılabilirlik, konum bağımsızlığı, ölçeklenebilirlik gibi özelliklere sahiptir ve bu da büyüyen veri yönetimi taleplerini karşılamayı verimli hale getirir. Günümüzde birçok kuruluş MongoDB'yi daha hızlı uygulama geliştirme ve yüksek oranda büyüyen çok çeşitli verilerin verimli bir şekilde kullanılması için benimsiyor. Bu yazıda, taşınan veriler üzerinde MySQL'den MongoDB ve Naive-Bayes sınıflandırma Mapreduce modeline verilerin taşınması için etkili bir yaklaşım öneriliyor. Bu yazıda ayrıca göç ve sınıflandırıcı iki farklı veri setinde değerlendirilmiştir. Deney sonuçları, sorgu yürütme performansının taşıma işleminden sonra geliştiğini göstermektedir. Naive Bayes sınıflandırıcısı Mapreduce modeli, önemli miktarda zaman tasarrufu sağlar ve taşınan veriler üzerinde daha iyi sonuçlar verir.

**VIII. Proje Adı:** En çok kullanılan İlişkisel ve NoSQL veritabanı yönetim sistemlerinin performans karşılaştırması. (Performance comparison of most used Relational and NoSQL database management systems)  
[\[https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp\]](https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp)

**Tez NO:** 537808

**Yazar(lar):** Berna DUMANLI

**Tarih:** 2019

**Proje Özeti:** Veritabanı yönetim sistemleri, sağladıkları yararlar sebebiyle uzun yıllardır kullanılmaktadırlar. Büyük miktarda verinin, hızlı bir şekilde depolanmasında son yıllarda NoSQL veritabanları tercih edilmektedir. Bu sistemler ölçeklenebilme kolaylığı ve hızlı depolama yönünden avantajlı olmasına rağmen, ACID modelini tam olarak desteklememeleri ve bazı sorgularda yavaş kalmaları sebebiyle kullanım alanı kısıtlı olmaktadır. Bu dezavantajlara rağmen yeni sürümleriyle her geçen gün yeni özellikler kazanmakta ve NoSQL veritabanlarının kullanımı artmaktadır. Bu tezde, en çok kullanılan NoSQL veritabanı sistemlerinden üç tane ve en çok kullanılan ilişkisel veritabanlarından üç tane seçilerek sistemlerin yeteneklerini ve farklı işlemlerde nasıl tepkiler verdiklerini ortaya çıkarmak için testler yapılmıştır. Bu amaçla, birçok iş yükü yüklenmiş ve iki adet test ortamı hazırlanmıştır. Bu çalışmanın sonuçları kullanılan her bir sistemin zayıf ve güçlü yanlarını ortaya koymaktadır. Test edilen her bir veritabanı farklı mimari ve özelliklere sahip olduğu için farklı sonuçlar ortaya çıkarmıştır. Karşılaştırmada, ilk olarak Yahoo tarafından geliştirilen Yahoo Cloud Serving

Benchmark (YCSB) kullanılmıştır. Elde edilen sonuçlara göre, NoSQL sistemler içinde MongoDB en iyi performansı gösterirken, Couchbase de ona yakın bir performans göstermiştir. Çok düğüme sahip kümelerde performansı yüksek olan Cassandra ise tek düğüm üzerinde beklenen performansı gösterememiştir. Test edilen ilişkisel veritabanları MySQL, SQL Server ve Oracle ise okuma performansı olarak NoSQL sistemler ile yakın olsa da, ACID özelliğine sahip olmaları nedeniyle yazma ve güncellemede düşük performans göstermişlerdir. İkinci test olarak, veritabanlarına farklı büyüklüklerde veriler yüklenip, farklı türden sorguları ne kadar sürede tamamladıkları ölçülmüştür. En hızlı yanıtları ilişkisel veritabanlarından SQL Server ve Oracle vermiştir. MongoDB testimizdeki diğer ilişkisel veritabanı olan MySQL'e yakın sonuçlar verebilmiştir. Cassandra ve Couchbase ise bu testte iyi bir performans gösterememiştir.

- IX. **Proje Adı:** SQL ve NoSQL veritabanlarının karşılaştırılması (A comparison of SQL and NoSQL databases)[<https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp>]

**Tez NO:** 490334

**Yazar(lar):** Souad Rashd RASHD

**Tarih:** 2018

**Proje Özeti:** Bu çalışma SQL ve NoSQL veritabanlarının bir karşılaştırmasını ve avantajları ile dezavantajlarının belirlenmesine yönelik bir araştırmayı içerir. Karşılaştırmalar her iki veritabanı türünü ölçekleme, (ACID ve CAP) kuramı, esneklik, başarımlar, şema, sorgulama dili, maliyet, hız ve veri açısından değerlendirmektedir. Çalışmada, MySQL, MS-SQL Server Express Edition, Oracle 11g Express Edition gibi SQL veritabanlarına örnekler verilmiştir. Ayrıca, çalışmada NoSQL veritabanlarına örnek olarak, Anahtar-Değer Depo veritabanları, Kolon-Yönelimli Veritabanları, Doküman Depo Veritabanları ve Dizge Veritabanları kısaca örneklerle anlatılmıştır. Çalışmanın sonuçları ve karşılaştırmalar SQL veritabanlarının tablo tabanlı sistemler olduğunu, esnekliği, dikey ölçeklemeyi ve yapısal sorgulama dili SQL'i desteklediğini ortaya koymuştur. Diğer taraftan, bu çalışma, SQL sıra düzensel veri yapıları için uygun olmadığını, ve NoSQL veritabanlarının daha az esnek olmasına rağmen, yatay ölçeklenebilir, yapısal olmayan sorgulama dili özellikleri ile, sıra düzensel veritabanı yapıları için daha uygun olduğu ortaya çıkartmıştır.

- X. **Proje Adı:** İlişkisel ve ilişkisel olmayan (NOSQL) veri tabanı yönetim sistemleri

mimari performansının yönetim bilişim sistemleri kapsamında incelenmesi

(The examination of relational and NOSQL database manegement system's architectural performances in terms of management information systems)

[<https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp>]

**Tez NO:** 503001

**Yazar(lar):** Serdar ÖZTÜRK

**Tarih:** 2017

**Proje Özeti:** Günümüzdeki bilgi ve iletişim teknolojilerinde yaşanan değişim ve gelişime bağlı olarak, verilerin modellenmesi ve belli bir modelle ifade edilen verilerin saklanması 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 depolanması bir zorunluluk haline gelmiştir. Verinin büyüklüğü, miktarı ve karmaşıklığı gibi etkenlere bağlı olarak farklı veri modelleme, veri depolama ve sorgulama yöntemleri geliştirilmiştir. Veri miktarındaki ciddi artış, uzun süredir ihtiyaçlara cevap veren ilişkisel veri tabanı sistemleri ile birlikte ilişkisel olmayan (NoSQL) sistemlerin de gelişmesine neden olmuştur. Geliştirilen birçok ilişkisel veri tabanı yanı sıra yakın geçmiş itibarıyla ortaya çıkan çok sayıdaki NoSQL veri tabanları, farklı özellikleri nedeni ile, organizasyonların yönetim bilgi sistemlerinde çalışan veri tabanı yöneticilerinin seçim yapmasını zorlaştırmaktadır. İlişkisel ve NoSQL veri tabanı özelliklerini iyi bilerek, hazırlanan uygulamaların niteliklerine en uygun veri tabanını seçmek gerek maliyet gerekse hız açısından oldukça önemlidir. Yapılan çalışmada işletmeler tarafından kullanılan veri tabanı yönetim sistemlerinin seçimi problemi üzerinde durulmuştur. Firmaların yönetim bilgi sistemlerinde çok sık kullanılan uygulamalar baz alınarak seçilen bir uygulama ile, ilişkisel ve NoSQL veri tabanlarının farklı veri tabanı sorgu türleri karşısında gösterdikleri performans karşılaştırılmıştır. İşletmelere hangi durumda nasıl bir veri tabanı yönetim sistemi kullanmaları konusunda önerilerde bulunulmuştur.

- XI. **Proje Adı:** Hadoop Map Reduce Framework kullanarak büyük veri analizi ve veri geçiş sürecini azaltma (Big Data Analytics Using Hadoop Map Reduce Framework and Data Migration Process) [<https://ieeexplore.ieee.org/document/8463824>]

**Yazar(lar):** Payal M. Bante, K. Rajeswari

**Tarih:** 17-18 Ağustos 2017

**Proje Özeti:** Veritabanları muazzam hız, hacim (terabayt - petabayt arası) ve türleri

(Veri çeşitliliği) açısından giderek daha karmaşık hale geliyor. Bu kadar büyük verileri yönetmek, kapsamlı ve bir o kadar zor. Bu sorunu aşmak için, Verilerin MySQL'den NoSQL'e ve bigdata işlemlerine geçirilmesi, Hadoop MapReduce olarak tanımlanan bir programlama konsepti ile gerçekleştirilir.(Hadoop: Büyük veri kümeleri ile birden fazla makinede paralel olarak işlem yapmayı sağlayan Java ile yazılmış açık kaynak kodlu kütüphanedir. [<http://www.buyukveri.co/hadoop-nedir/>]) Bu makale, verilerin NoSQL (MongoDB) veri tabanına ve Hadoop Haritasını kullanarak bigdata analitiklerine ilişkisinden geçişi için Hadoop Dağıtılmış Dosya Sistemi (HDFS) üzerindeki çerçeveyi azaltan bir metodoloji sunmaktadır. Ayrıca Sqoop, verileri İlişkisel Veritabanından analitik işlem için Hadoop'a geçirmek için kullanılır. Kovan, analiz edilen verileri Hadoop'tan MongoDB'ye taşıdı.

- XII. **Proje Adı:** Model geçişi içinde kullanılan orta model(Mid-Model) tasarım ile İlişkisel veritabanları ve NoSQL veritabanları arasında veri geçişi (Mid-model Design Used in Model Transition and Data Migration between Relational Databases and NoSQL Databases) [<https://ieeexplore.ieee.org/document/7463832>]

**Yazar(lar):** Dongzhao Liang, Yunzhen Lin, Guiguang Ding

**Tarih:** 19-21 Aralık 2015

**Proje Özeti:** Uygun orta modeli seçmek, model geçişinde yüksek verim ve ilişkisel veritabanları ile NoSQL veritabanları arasında veri geçişi için kritik öneme sahiptir. İlişkisel veritabanları ve NoSQL veritabanları arasında model geçişi ve veri geçişi için kullanılan tek tip orta-model tasarım problemini araştırıyoruz. Önerilen yaklaşımımız tam bir süreç önermektedir. Orta modeli oluşturmak için veri özelliklerini ve sorgu özelliklerini kullanırız. Ve önceden tanımlanmış bir strateji ile, orta modeli hedef veri tabanının fiziksel modeline aktarıyoruz. Bu işlemeyi oluşturduktan sonra, verileri kaynak veritabanlarından hedef veritabanlarına taşıyabiliriz.

- XIII. **Proje Adı:** İlişkisel veritabanlarının NoSQL'e geçişi: İleriye giden yol (Migration of a relational databases to NoSQL : The way forward) [<https://ieeexplore.ieee.org/document/7905665>]

**Yazar(lar):** Alae El Alami, Mohamed Bahaj

**Tarih:** 29 Eylül-1 Ekim 2016

**Proje Özeti:** NoSQL veritabanları, birimdeki veri işleme sorunlarını çözmek için tasarlanmıştır. Günümüzde büyüme verileri, verileri bir kaynak veritabanından NoSQL

veritabanına dönüştürmek için yeni kavramlar ve yaklaşımlar hakkında düşünmemize neden oluyor. Bu yazıda bazı yaklaşımlar, göç yöntemleri, veritabanı tersine mühendislik anlatılmaktadır ve ilişkisel bir veritabanından NoSQL (MongoDB) Belge yönelimli bir veritabanına geçiş yaklaşımı sunulmaktadır. Koleksiyonlar arasındaki bütünlük kısıtlamalarını sağlarken, şema ve veri haritalamasının geçişine devam ettiğimiz bir veri modelinin başarılmasına dayanan yaklaşım.

XIV. **Proje Adı:** Büyük veri tasarım zorlukları üzerine bir sosyal alışveriş uygulaması kullanılarak ilişkisel veritabanları ve NoSQL akımları arasında bir karşılaştırma (A comparison between relational database models and NoSQL trends on big data design challenges using a social shopping application) [\[https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp\]](https://tez.yok.gov.tr/UlusalTezMerkezi/giris.jsp)

**Tez NO:** 395452

**Yazar(lar):** Serhat UZUNBAYIR

**Tarih:** 2015

**Proje Özeti:** Veri yaratımı günden güne fazlaca artmaktadır. Sonuç olarak bugünlerde karşı karşıya kaldığımız verinin anlamını genişletmek için Büyük Veri terimi ortaya çıktı. Geleneksel veritabanı teknolojileri büyük miktarlarda veriler içeren uygulamaları çalıştırırken mücadeleler yaşamaya başladı. Bu tür sorunlar araştırmacıları yepyeni veri işleme metotları geliştirmeye yöneltti. Tüm sistemler gerekli olduğu zaman yeni gereksinimlerden kaynaklanan değişimlere ayak uydurmak zorundadırlar. Piyasada çeşitli veritabanı yönetim sistemleri ve ürünleri bulunmaktadır. İlişkisel veritabanları 1970'lerden beri veri saklama ve işleme konusunda etkiliydiler. Fakat bugünkü verinin miktarı geçtiğimiz birkaç yıla göre karşılaştırıldığında bile çok fazladır. Bu durum kaçınılmaz olarak bazı sistemlerin tasarımlarını ilişkisel modellerden NoSQL akımına çevirmeye zorlamıştır. Farklı şirketler tarafından geliştirilen bir çok farklı NoSQL ürünü vardır. Bu durumda geliştiriciler sistemlerindeki Büyük Veri ve onun problemleri ile uğraşmak için hangi tür veritabanı seçeceklerine karar vermekte zorlanmaktadırlar. Bu tezde, farklı veritabanı yönetim sistemleri ve zorlukları özetlenmiştir. İlişkisel ve grafik tabanlı olmak üzere iki farklı veritabanı teknolojisi analiz edilmiş ve karşılaştırılmıştır. Bu iki teknoloji için sosyal ağ ile çevrimiçi alışveriş uygulaması olan TrendPin üzerinde veri modelleri tasarlanmış ve geliştirilmiştir. Tasarım modelleri ve farklı sorgu performansları gösterilmiştir. Ek olarak bilgi çıkarımı konusu açıklanmış ve grafik modeli tasarımında karşılaşılan sorunların üstesinden gelmek için bir bilgi bankası

oluşturulmuştur.

- XV. **Proje Adı:** NoSQL'e İlişkisel Veri Tabanı Geçişi İçin Belge Odaklı Veri Şeması (Document-Oriented Data Schema for Relational Database Migration to NoSQL) [<https://ieeexplore.ieee.org/document/8316298>]

**Yazar(lar):** Shady Hamouda, Zurinahni Zainol

**Tarih:** 21-23 Ağustos 2017

**Proje Özeti:** Büyük veri çok önemli bir konu haline geldi ve modern dünyadaki en önemli teknolojilerden biri. Büyük veriler için yarı yapılandırılmış bir veri formatının yönetimi ele alınması gereken önemli bir husustur. İlişkisel veritabanı yönetim sistemlerinin çoğu, büyük verilerin ölçeklenebilirliğini ve esnekliğini ele almayı başaramaz. Veri tabanı teknolojisinde yeni bir konsept olan NoSQL, büyük miktarda veri desteği sağlayarak iyi bir veri depolama ve erişim mekanizması sunar. Ancak, NoSQL'in standart veri modelleme yöntemi yoktur. Veri tabanı ilişkilerinin ele alınmasında önemli özelliklerin NoSQL veri modellemesinde dikkate alınması gerekir. Bu çalışma, büyük veriler için belgeye dayalı bir veri modeli önermekte ve daha sonra bu modeli, ER modelinin özelliklerine dayanarak ilişkisel veritabanı uygulamalarını NoSQL'e geçirmek için uygulamaktadır.

Bölüm 1, Tezin Giriş kısmıydı. Bölüm 2 ve sonrası tezin içeriğindeki bölüm başlıklarını ve alt başlıkları içerir. İstenildiği kadar yapılabilir.

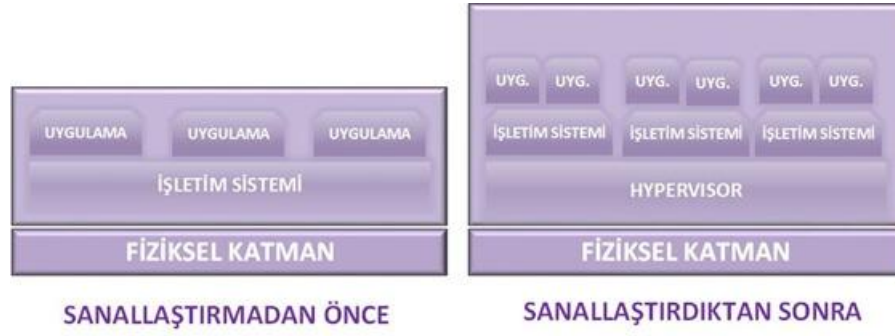
### 3. ~~YAPILAN İŞ~~

#### 3.1. SUNUCU KURULUMU

##### 3.1.1. Sanallaştırma Nedir?

Sanallaştırma aynı bilgisayar üzerinde konuk olarak isimlendirebilebilen birden fazla işletim sisteminin çalışmasını ifade eder. Sanallaştırma bir çeşit sistem olarak ele alındığında bu sistem, kullanıcının doğrudan fiziksel olan sistem kaynaklarına erişimini engellemiş olur. Bu sayede kullanıcı talepleri uygun bir biçimde alınarak donanıma iletimi sağlanır ve sanallaştırma çekirdeği denilen **VMM** veya

**Hypervisor** denilen mantıksal bir kısımdır.



**Şekil 3.1.1.1. Sanallaştırmanın Farkı**

Sanallaştırma işleminin yapılabilmesi için kullanılan bazı programlar vardır.

Bunlardan en yaygın olarak kullanılanlar ;

- VirtualBox
- Vmware
- Hyper-v
- Citrix XenServer

vs dir.

Projede kullandığımız sanallaştırma aracı olduğu için VirtualBox , bir defada birden fazla işletim sistemi çalıştırmaya olanak sağlar. Bu yol ile , bir işletim sistemi için yazılmış olan bu yazılımı Mac ,Linux ya da Windows yazılımı gibi yazılımları çalıştırabilirsiniz.

Kurulduktan sonra sanal makine ve sanal sabit diskleri , ana bilgisayarlar arasında keyfi olarak dondurabilen , uyandırabilen , kopyalayabilen ,yedekleyebilen ve taşıyabilen bir kutu olarak kabul edilir.

### **3.1.2. VirtualBox Kurulumu**

Oracle VM VirtualBox isminden de anlaşılacağı üzere Oracle Corporation tarafından geliştirilen x86 sanallaştırma için ücretsiz ve açık kaynak barındırılan hipervizördür.

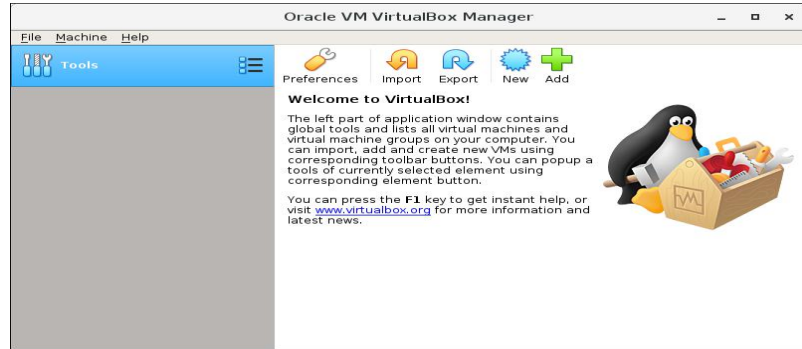
VirtualBox kurulum dosyasının öncelikle bilgisayar indirilmesi gerekmektedir. Bu dosya Oracle’a bağlı VirtualBox’ın kendi web sitesinden rahatlıkla temin edilebilir.

Yüklemeden sonra Oracle VM VirtualBox’i aşağıdaki şekilde başlatabilirsiniz :

- Windows işletim sistemi olan bir bilgisayarda , Programlar menüsünde, VirtualBox grubundaki öğeye tıklayın.



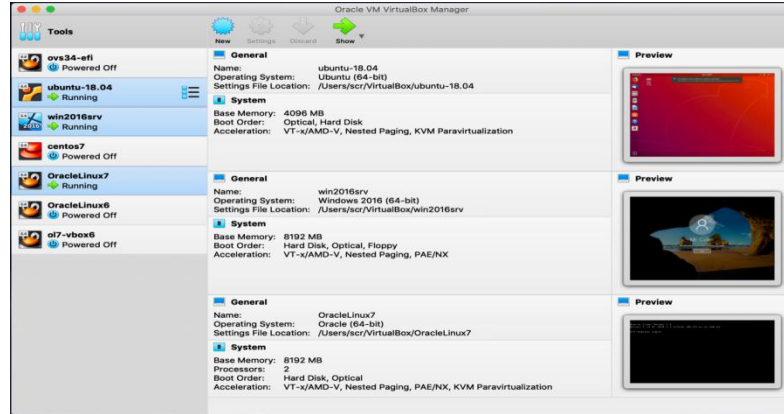
- Oracle VM VirtualBox’u ilk kez başlattığınızda , aşağıdaki gibi bir pencere görüntülenir. Bu pencereye VirtualBox Manager denir. Sol bölme daha sonra tüm sanal makinelerinizi listeler.



Şekil 3.1.1.2. Sanal Makine Kurulum 1. Adımı

Henüz herhangi bir sanal makine oluşturmadığınız için bu liste boştur. Araçlar düğmesi, Sanal Medya Yöneticisi gibi kullanıcı araçlarına erişim sağlar.

- Aşağıda sanal makinelerin oluşturulmasının ardından VirtualBox da meydana gelen değişimler görülmektedir.



Şekil 3.1.1.3. Sanal Makine Kurulum 2. Adımı

- VirtualBox Manager penceresinde Yeni’ye tıklayın.Yeni bir sanal makina siz kurarken size yardımcı olacak bir sihirbaz gelir.



Şekil 3.1.1.4. Sanal Makine Kurulum 3. Adımı

- Yukarıda görüldüğü üzere eklenecek sanal makine için bazı özellikler eklenmeli. Ad, sanal makine'nin dosyası, makinenin tipi ve makinenin versiyonu gibi istekleri olur sihirbazın. Gerekli bilgilerin girişinin yapılmasının ardından devam edilebilir.
- Gelecek tuşunun ardından ekrana bir sabit disk ayırma penceresi açılır.



Şekil 3.1.1.5. Sanal Makine Kurulum 4. Adımı

- Bu pencerede de yeni bir virtual hard disk' i oluştur seçeneğini seçiyoruz.
- Ardından açılan pencere ise eklenen dosya ve dosyanın konumu ile ilgili olacaktır.



**Şekil 3.1.1.6. Sanal Makine Kurulum 5. Adımı**

- Açılan bu pencerede gerekli RAM kapasitesi belirlenir. Ardından sona gelinmiş olur. Yapılan bu işlemlerin ardından bir sanal makine VirtualBox sanallaştırma aracının üzerine kurulmuş olur.

Projemizde sanalda kurduğumuz iki adet Linux işletim sistemi var. Aynı VirtualBox üzerine kurduğumuz bu işletim sistemlerinin birine MongoDB diğerine ise MySQL yükledik. Bu sayede iki farklı sunucu üzerinde çalışmış olduk.



### **3.1.3. MongoDB'nin Sanal Makina Üzerine Kurulumu**

VirtualBox üzerine kurulan sanal makinelerden birine MongoDB nin kurulabilmesi için gerekli kodları kurulu olduğu sanal makina üzerindeki terminale yazılması gereken kodlar aşağıda verilmiştir :

```
$ adduser yenikullanici
```

```
$ usermod -aG sudo yenikullanici
```

```
$ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv  
0C49F3730359A14518585931BC711F9BA15703C6
```

```
$ echo "deb [ arch=amd64,arm64 ]
```

```
http://repo.mongodb.org/apt/ubuntu/xenial/mongodb-org/3.4multiverse" | sudo  
tee/etc/apt/sources.list.d/mongodb-org-3.4.list
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install mongodb-org
```

```
$ sudo systemctl start mongod
```

```
$ sudo systemctl status mongod
```

```
$ sudo systemctl enable mongod
```

Bu aşamada mongod servisini yeniden başlatacağız.

```
$ sudo systemctl restart mongod
```

```
$ sudo systemctl status mongod
```

Bu aşamada kurulum tamamlanmış olur. Eğer üzerinde ekstra bazı eklemeler yapılmak istenirse mongoDB buna imkan verir. Terminal komutları aracılığı ile işlemlerinizi gerçekleştirebilirsiniz.

#### **3.1.4. MongoDB'ye Terminal Üzerinden Bağlanma**

MongoDB ye bağlanma işlemini terminal üzerinden rahatlıkla gerçekleştirebiliriz.

```
$ mongo
```

Komutunun yazılması ile terminal üzerinden mongoDB'ye bağlanmış oluruz.

#### **3.1.5. MySQL' in Sanal Makina Üzerine Kurulumu**

VirtualBox üzerine kurulan sanal makinelerden birine MySQL'in kurulabilmesi için gerekli kodları kurulu olduğu sanal makina üzerindeki terminale yazılması gereken kodlar aşağıda verilmiştir :

```
$ sudo apt-update
```

```
$ sudo apt install mysql-server
```

```
$ sudo systemctl status mysql
```

```
$ sudo mysql_secure_installation
```

```
$ sudo mysql
```

komutları ile kurulum gerçekleştirilir.

#### **3.1.6. MySQL'e Terminal Üzerinden Bağlanma**

MySQL'e bağlanma işlemini terminal üzerinden rahatlıkla gerçekleştirebiliriz.

```
$ sudo mysql
```

Komutunun yazılması ile terminal üzerinden MySQL'e bağlanmış oluruz.

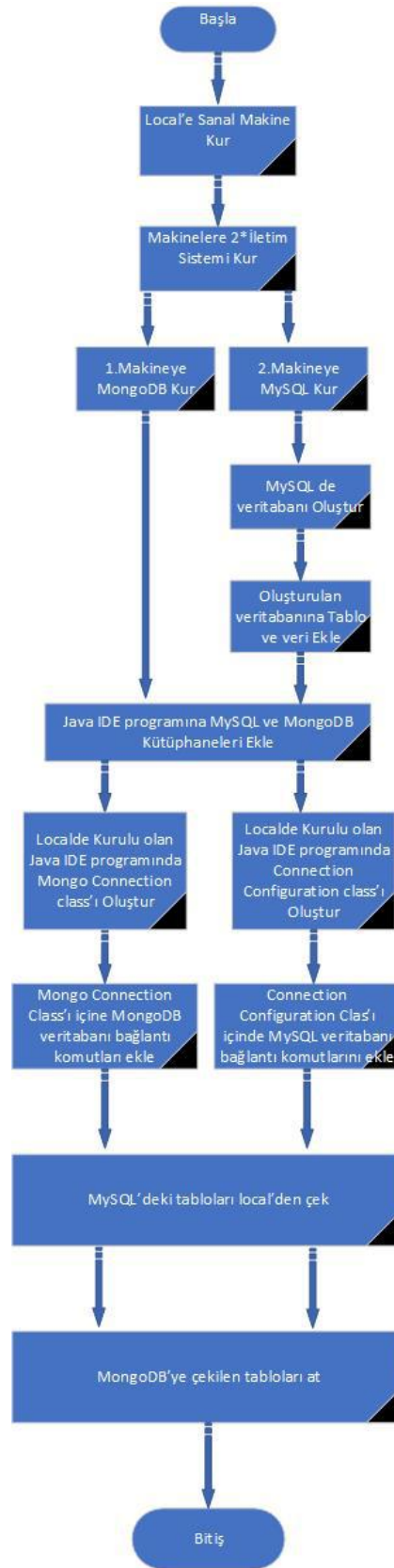
### **3.2. VERİ GÖÇÜ ALGORİTMALARI**

Projeyi gerçekleştirmek için kullandığımız veri göçü algoritması aşağıdaki gibidir:

1. Başla

2. Locale sanal makine kur.
3. Makinelere 2\*İşletim Sistemi kur.
4. 1. makineye MongoDB kur.
5. 2. makineye MySQL kur.
6. JAVA'ya MySQL ve MongoDB kütüphanelerini ekle.
7. MySQL'de veritabanı oluştur.
8. Oluşturulan veritabanına tablo ve veri ekle.
9. Localde kurulu olan JAVA IDE programında MongoConnection ve ConnectionConfiguration classlarını(Sınıf) oluştur.
10. MongoConnection classı içinde MongoDB veritabanına bağlantı komutlarını ekle.
11. ConnectionConfiguration classı içinde MySQL veritabanı bağlantı komutlarını ekle.
12. MySQL'deki tabloları localden çek.
13. MongoDB'ye çekilen tabloları at.
14. Bitiş

### **3.2.1. Akış Diyagramı**



## 4. BULGULAR VE TARTIŞMA

### 4.1. MongoDB vs. MySQL

İlişkisel olmayan veritabanı MongoDB kullanmanın , ilişkisel veritabanı MySQL ile bası izlemlerden geçirilerek kıyaslandı . Bu işlemler, herhangi bir veritabanında gerçekleştirilebilecek dört temel işlemidir, yani:

1. Insert (Ekleme)
2. Select (Seçme/Sorgu)
3. Update (Güncelle)
4. Delete (Silme)

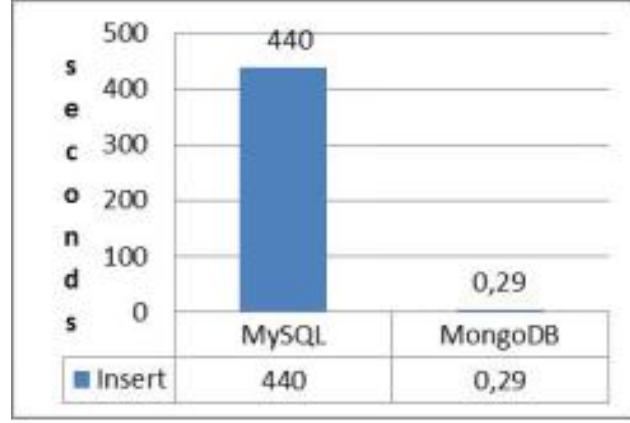
Yapılan bu işlemler sonucunda aşağıda sunulan tüm sonuçların, aşağıdaki özelliklere sahip bir bilgisayardan elde edildiğine dikkat edilmelidir.

Windows 7 Ultimate 64-bit, işlemci Intel Core i3 (2,4 GHz), 4 GB RAM belleği.

#### 4.1.1 Ekleme İşleme

Veri ekleme, kullanıcıların eklenmesiyle her iki veritabanında da başladı. Her veritabanı için 10.000 kullanıcı eklenmiştir. Kimlikleri her iki veritabanı tarafından otomatik olarak oluşturulmuştur. Bir tane MySQL ve bir de Mongo için olmak üzere iki komut dosyasını çalıştırdıktan sonra, Şekil 4.1, 440 saniyede 10.000 kullanıcının MySQL'e eklendiğini, MongoDB'de ise zamanın sadece 0.29 saniye olduğunu gösteriyor.

Yukarıda bahsedilmiş olan hız farkının anlaşılabilmesi için bir çalışma yapılmıştır. Bu çalışmaya göre bir tartışma forumu oluşturulmuştur. Bu forumda ki tartışmacıların her birinin bıraktığı yorumlar bir yorum tablosuna ve her bir tartışma konusu alt bir forum tablosuna eklenmiştir. Eklenen bu verilerin eklenme hızı karşılaştırılması sonucunda MySQL'in MongoDB'ye oranla daha yavaş olduğu sonucuna varılmıştır. Şekil 4.1.1.1 'deki görselde de yapılan işlemin grafiğe dökülmüş hali gözlemlenebilmektedir.

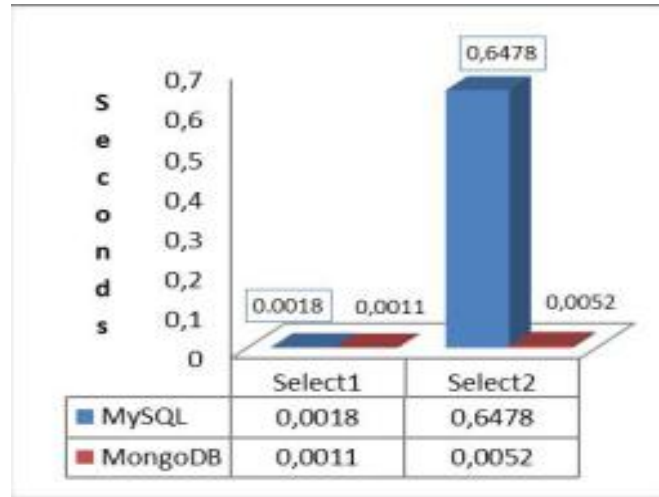


Şekil 4.1.1.1. Veritabanlarının Hız karşılaştırması

#### 4.1.2 Seçme İşlemi

Sorgu işlemlerinin performansını test etmek için iki seçme sorgusu yaptık:

- Kullanıcıların katılmış olduğu ve belirli bir tarihten farklı bir tarihte yaptığı tüm tartışmaların seçimi;
- Veritabanından tüm kullanıcıların seçimi ve her kullanıcı tarafından başlatılan tartışmaların sayısı



Şekil 4.1.2.1. Sorgulama İşlemleri

İlk seçim MySQL’de 0.0018 saniyede , MongoDB’de 0.0011 saniyede ,ikinci seçim ise MySQL’de 0.6478 saniyede ve MongoDB ‘de 0.0052 saniyede gerçekleştirildi. Grafikten , MongoDB'nin performans için MySQL'den daha az zaman harcadığını fark ediyoruz. Şekil 4.1.2.1’de aradaki farkı gözlemleyebiliriz.



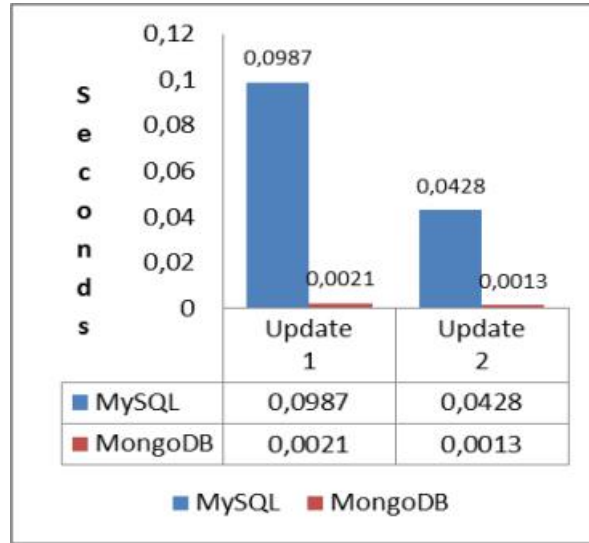
### 4.1.3 Güncelleme İşlemi

Güncelleme işlemini veritabanlarında test etmek için aşağıdaki iki sorgu gerçekleştirildi:

- Özel bir kullanıcı tarafından yazılmış bir yorumu güncelleme
- Bir kullanıcının e-posta adresini güncelleme

Yukarıdaki sorgular yapılarak aşağıdaki sonuçlar elde edilmiştir.

İlk güncelleme MySQL'de 0.0987 saniyede, MongoDB'de 0.0021 saniyede, ikinci güncelleme ise MySQL'de 0.0428 saniyede ve MongoDB'de 0.0013 saniyede gerçekleştirildi. Şekil 4.'ten itibaren, MongoDB'nin, Şekil 4.1.3.1.'de gösterildiği gibi güncelleme işlemini gerçekleştirmek için MySQL'den daha az zaman harcadığını görüyoruz.



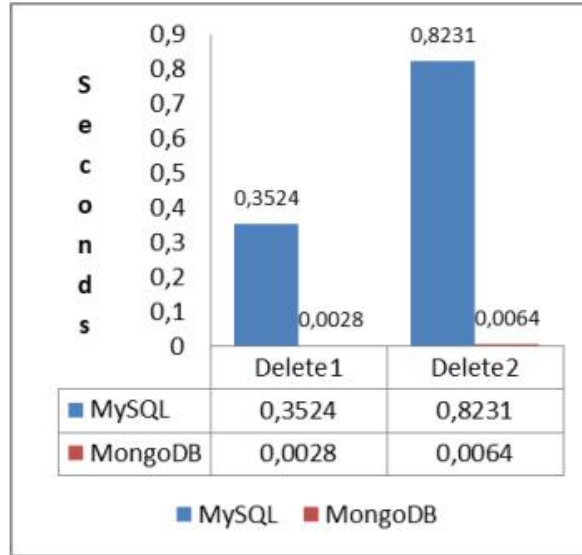
Şekil 4.1.3.1. Güncelleme işlemi sonuçları

### 4.1.4 Silme İşlemi

Yukarıda açıklanan diğer işlemler gibi, her veritabanı için iki silme sorgusu gerçekleştirdik.

- İlk sorgu, bir kullanıcı tarafından gönderilen tüm yorumları sildi
- İkinci sorgu, belirli bir kullanıcı tarafından oluşturulan tüm forumları siler

Bir forumun silinmesiyle, belirtilen forumda yer alan tüm alt forumların, yorumların ve tartışmaların hem MySQL hem de MongoDB'de otomatik olarak silindiğini belirtmek önemlidir.



**Şekil 4.1.4.1.** Silme işleminin sonuçları

MongoDB, dört temel işlemin hepsinde MySQL'den daha düşük yürütme süreleri sağladı; bu, bir uygulamanın aynı anda binlerce kullanıcıya destek vermesi gerektiğinde zorunludur. Bu nedenle, yukarıdaki karşılaştırma, büyük miktarda veri için MongoDB'nin iyi bir performansa sahip olduğunu ve MySQL'e tercih edildiğini kanıtlamaktadır.

## 5. KAYNAKLAR

### Makale ve Tezler

- Sanket Ghule and Ramkrishna Vadali, “Transformation of SQL system to NoSQL system and performing data analytics using SVM”, 2017.
- Ying-Ti Liaoa, Jiazheng Zhoua, Chia-HungLu, Shih-ChangChen, Ching-Hsien Hsu, Wenguang Chen, Mon-Fong Jiange and Yeh-Ching Chunga, “Data adapter for querying and transformation between SQL and NoSQL database”, *Future Generation Computer Systems*, vol. 65, pp. 111–121, 2016.
- Gansen Zhao, Qiaoying Lin, Libo Li and Zijing Li, “Schema Conversion Model of SQL Database to NoSQL”, 2014.
- Ágnes Vathy-Fogarassy and TamásHugyák, “Uniform data access platform for SQL and NoSQL database systems”, *Information Systems*, vol. 69, pp. 93-105, 2017.
- Dimpal Tomar, Jai Prakash Bhati, Pradeep Tomar and Gurjit Kaur, “Migration of healthcare relational database to NoSQL cloud database for healthcare analytics and management”, *Healthcare Data Analytics and Management*, pp. 59-87, 2019.
- Girts Karnitis and Guntis Arnicans, “Migration of Relational Database to Document-Oriented Database: Structure Denormalization and Data Transformation”, 2015.

- Ganesh B. Solanke and K. Rajeswari, “Migration of Relational Database to MongoDB and Data Analytics using Naive Bayes Classifier based on Mapreduce Approach”, 2017.
- Berna Dumanlı, “En çok kullanılan İlişkisel ve NoSQL veritabanı yönetim sistemlerinin performans karşılaştırması”, 2019
- Souad Rashd Rashd, “SQL ve NoSQL veritabanlarının karşılaştırılması”, 2018.
- Serdar Öztürk, “İlişkisel ve ilişkisel olmayan (NOSQL) veri tabanı yönetim sistemleri mimari performansının yönetim bilişim sistemleri kapsamında incelenmesi”, 2017.
- Serhat Uzunbayır, “Büyük veri tasarım zorlukları üzerine bir sosyal alışveriş uygulaması kullanılarak ilişkisel veritabanları ve NoSQL akımları arasında bir karşılaştırma”, 2015.
- Payal M. Bante and K. Rajeswari, “Big Data Analytics Using Hadoop Map Reduce Framework and Data Migration Process”, 2017.
- Dongzhao Liang, Yunzhen Lin and Guiguang Ding, “Mid-model Design Used in Model Transition and Data Migration between Relational Databases and NoSQL Databases”, 2015.
- Alae El Alami and Mohamed Bahaj, “Migration of a relational databases to NoSQL: The way forward”, 2016.
- “Document-Oriented Data Schema for Relational Database Migration to NoSQL”

### **Kitaplar**

- K. Sanobar, M. Vanita, “SQL Support over MongoDB using Metadata”, International Journal of Scientific and Research Publications, Volume 3, Issue 10, October 2013.
- R. P Padhy, M. R. Patra, S. C. Satapathy, “RDBMS to NoSQL: Reviewing Some Next-Generation Non-Relational Database’s”, International Journal of Advance Engineering Sciences and Technologies, Vol. 11, Issue No. 1, 015-030, 2011.
- J. Clarence, M. Tauro, S. Aravindh, A. B. Shreeharsha, “Comparative Study of the New Generation, Agile, Scalable, High Performance NOSQL Database”, International Journal of Computerapplications, ISSN 0975 – 888, Volume 48– No.20, June 2012.
- S. Hoberman, “Data Modeling for MongoDB”, Publisher by Technics Publications, LLC 2 Lindsley Road Basking Ridge, NJ 07920, USA, ISBN 978-1-935504-70-2, 2014.

### **Bilgisayar Programları**

- IntelliJ IDEA, *Bilgisayar Programı*, JetBrains, 2001
- VirtualBox, *Bilgisayar Programı*, Oracle Corporation, 2010
- MySQL, *Bilgisayar Programı*, Oracle,

## 5. SONUÇLAR VE ÖNERİLER

Geleneksel -SQL- veritabanları , artan büyük veri işleme ve işleme taleplerine yetişmek için yetersizdir. Geleneksel ilişkisel veritabanlarında veri modeli sınırlamaları, mimari sınırlamalar, ölçeklenebilirlik ve performans sınırlamaları vardır. Yeni nesil NoSQL veritabanı, şema içermeyen tasarım, yüksek kullanılabilirlik, konum bağımsızlığı, ölçeklenebilirlik gibi özelliklere sahiptir ve bu da büyüyen veri yönetimi taleplerini karşılamayı verimli hale getirir.[7. **Tezin Linki**] İlişkisel veritabanı yönetim sistemlerinin çoğu, büyük verilerin ölçeklenebilirliğini ve esnekliğini ele almayı başaramaz. Veri tabanı teknolojisinde yeni bir konsept olan NoSQL, büyük miktarda veri desteği sağlayarak iyi bir veri depolama ve erişim mekanizması sunar. NoSQL veritabanları, birimdeki veri işleme sorunlarını çözmek için tasarlanmıştır. Günümüzde büyüme verileri, verileri bir kaynak veritabanından NoSQL veritabanına dönüştürmek için yeni kavramlar ve yaklaşımlar hakkında düşünmemize neden oluyor.[14. **Tezin Linki**]

Amaç, NoSQL veritabanı tarafına geçmek isteyen SQL kullanıcıları için kolaylığı ortaya koymak ve bu geçişi sağlayabilecek ortamı, programı JAVA dilinin sunduğu kolaylık, dinamiklik, güvenilirlik, sağlamlık, taşınabilir ve yorumlanabilirlik özelliklerinden yararlanarak oluşturabilmektir. 2 veritabanı arasındaki veri geçişini, JAVA üzerinden gerçekleştirerek sonuca ulaştırabilmektir.

Bu projede geleneksel veritabanlarından MySQL, NoSQL veritabanlarından MongoDB kullanılmıştır. JAVA tarafında bu veritabanları üzerinde işlemlerin gerçekleştirilmesini sağlayacak kütüphaneler JAVA'ya eklenerek projeye ilk adım atılmıştır. Sadece local, yani kullanıcıya ait sunucular üzerinden işlem yapılmaksızın, farklı ortamda bulunan her sunucu aynı işlemi gerçekleştirebilsin diye sanal işletim sistemi imkanı sunan VirtualBox programı locale kurulmuştur. VirtualBox, farklı işletim sistemleri kullanabilme imkanı sunan bir yazılımdır ve platformlar arası sanallaştırma uygulamasıdır. VirtualBox üzerine 2 Linux-Ubuntu işletim sistemi kurulmuştur ve terminal üzerinden birine MySQL, diğerine ise MongoDB kurulumu gerçekleştirilmiştir. MySQL tarafında örnek bir veritabanı oluşturularak verilerin eklenmesi sağlanmıştır. Oluşturulan bu veritabanını MongoDB tarafına aktarmak için

JAVA kısmına yazılması gereken kodlar için şunlar bilinmelidir:

SQL	MongoDB
Database(Veritabanı)	Database
Table(Tablo)	Collection(Koleksiyon/Toplama)
Row(Satır)	Document veya BSON Document(Belge)
Column(Sütun)	Field(Alan)
Index(Dizin)	Index(Dizin)
Table joins	embedded documents and linking
Primary Key(Birincil Anahtar) Herhangi bir benzersiz sütun veya sütun kombinasyonunu birincil anahtar olarak belirtin.	Primary Key MongoDB'de, birincil anahtar otomatik olarak <code>_id</code> alanına ayarlanır .
Aggregation(toplama) (örneğin grupla)	Aggregation Framework

**Tablo 4.1.** SQL'deki Terminoloji ve Kavramların MongoDB'deki Karşılığı

[\[https://docs.mongodb.com/manual/introduction/\]](https://docs.mongodb.com/manual/introduction/)

Yukarıdaki tabloda da belirtildiği gibi MongoDB'nin terminolojisi ve kavramları, ilişkisel veritabanı yönetim sistemlerinin terminoloji ve kavramlarına göre farklılık göstermektedir. MySQL'de oluşturulan tabloların MongoDB'de karşılığı collection'lardır. Tablo satırları, MongoDB'de document(Belge) yapısına karşılık gelir. MongoDB'deki bir kayıt, alan ve değer çiftlerinden oluşan bir veri yapısı olan bir belgedir.[] MongoDB belgeleri JSON nesnelere benzer.[] Alanların değerleri diğer belgeleri, dizileri ve belge dizilerini içerebilir.[]

Belge kullanımı birçok açıdan avantaj sağlayacaktır. Mesela,

- Belgeler (yani nesneler) birçok programlama dilinde yerel veri türlerine karşılık gelir.[]
- Gömülü belgeler ve diziler pahalı birleştirme gereksinimini azaltır.[]
- Dinamik şema, akıcı polimorfizmi

destekliyor. [\[https://docs.mongodb.com/manual/introduction/\]](https://docs.mongodb.com/manual/introduction/)

MySQL tablolarında sütunları, MongoDB’de field yapısı temsil etmektedir. MongoDB belgeleri alan-değer çiftlerinden oluşur ve aşağıdaki yapıya sahiptir[<https://docs.mongodb.com/manual/core/document/>]:

```
{  
  Field1 :  değer1 ,  
  field2 :  deger2 ,  
  alan3 :  değer3 ,  
  ...  
  fieldN :  değerN  
}
```

Bir alanın değeri, diğer belgeler, diziler ve belge dizileri dahil olmak üzere BSON veri türlerinden herhangi biri olabilir.[]

JAVA’da bu terminoloji farklılıkları göz önüne alınarak yazılan kodların başarılı bir şekilde çalışıp çalışmadığı kontrol edile edile sonuca ulaşılmaya çalışılmıştır. Bunu önce basit bir veritabanı oluşturup oluşturulan bu veritabanı üzerinden sonuca ulaşmak hedeflenmiştir. Sonuca ulaşıldığında ilgili veritabanının veri göçü işleminin başarılı olduğu gözlemlenirken, büyük verilere sahip veritabanlarında da aynı başarıya ulaşabilmesi hedeflenmiştir ve bu yönde deneme işlemleri yapılmıştır. Deneme işlemi için büyük verilere sahip örnek bir veritabanı, MySQL veritabanına import/export işlemi ile yani içe/dışa aktarma işlemi ile aktarılmıştır. Bu işlemden sonra gerçekleşmesi istenen bir sonraki adım, yani ilgili veritabanının MongoDB’ye aktarımı, göçü için JAVA üzerinde hazırlanan program çalıştırılmıştır. JAVA üzerinden göç işlemi başlatılmıştır ve verilerin sırayla geldiği gözlemlenmiştir. Lakin sonunda birkaç hata gözlemlenirken, verilerin bütünlüğünde bir sıkıntı olmadığı görülmüştür. Hataların giderilmesi ile veri göçü işlemi tekrarlanmıştır. Bu şekilde hatasız ve sıkıntısız bir şekilde MySQL üzerindeki veritabanının, MongoDB’ye geçişi sağlanmış olup, veri göçü işleminin başarılı bir şekilde sonuna gelinmiştir.

Proje üzerinde çalışacaklar için projede eklenmesi, geliştirilmesi gerekenler:

- Programı daha işlevsel hale getirebilmek için arayüz oluşturarak bu arayüz üzerinden daha kolay gerçekleştirilebilmeli,
- Kullanıcıdan alınan veri girişlerinde çıkabilecek yazım hatalarından doğacak sorunları düşünerek arayüzünde değişiklikler yapıp bu durum engellenmeli, Veri göçü başlangıcından sonuna kadar geçen zaman aralığını düşürecek önlemler alınmalı, yani veri göçü işlemini daha hızlı hale getirebilmeli.

## ÖZGEÇMİŞ

### KİŞİSEL BİLGİLER

Adı Soyadı :  
Doğum Tarihi ve Yeri :  
Yabancı Dili :  
E-posta :

### ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Lisans	Elektrik Elektronik Müh.	...Üniversitesi	2010
Lise		... Lisesi	2006