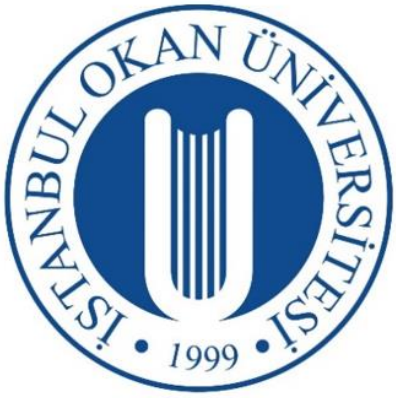
FEN BİLİMLERİ ENSTİTÜSÜ

BİLİŞİM SİSTEMLERİ PROGRAMI



BÜYÜK VERİ ANALİZİNDE

YATAY ÖLÇEKLENDİRİLEN VERİTABANI SİSTEMLERİ

YÜKSEK LİSANS TEZİ

MURAT MENTEŞE

Nisan 2019

Program: Bilişim Sistemleri

BÜYÜK VERİ ANALİZİNDE

YATAY ÖLÇEKLENDİRİLEN VERİTABANI SİSTEMLERİ

MURAT MENTEŞE

tarafından

İSTANBUL OKAN ÜNİVERSİTESİ

Bilişim Sistemleri Programı

YÜKSEK LİSANS

derecesi şartını sağlamak için sunulmuştur.

Onaylayan:

Dr.Öğr.Üyesi
Feridun C. ÖZÇAKIR
Danışman

Dr.Öğr.Üyesi
Nurşen TOPÇUBAŞI
Üye

Dr.Öğr.Üyesi
Erkan KIYAK
Üye

Nisan 2019

Program: Bilişim Sistemleri Programı

ÖZET

Teknolojik gelişmeler ışığında, artan veri hacmi ile yapısal olarak kontrol edilemeyecek boyutlara ulaşan veriyi geleneksel sistemler diye tabir ettiğimiz veritabanları üzerinde işlemek maliyet, zaman ve ek iş yükü ile birlikte artık yetersiz kalmış durumdadır.

Artan bu veri hacmini günümüz yeni teknolojik çözümlerle birlikte saklama, işleme ve analiz etme gibi aşamaları geleneksel sistemlere göre daha az maliyet, daha az zaman ve daha az iş yükü ile yapmak mümkün hale gelmiştir.

Bu tez çalışması kapsamında ise, Geleneksel veritabanı sistemleri ile Yeni nesil dağıtık dosya sistemleri arasındaki ilişkiyle birlikte iki sistem arasındaki performans kıyaslaması ve yeni nesil sistemlerin yapısal özellikleri ele alınmıştır.

Anahtar Kelimeler : Büyük Veri, Hadoop, PostgreSQL, Performans Analizi, Geleneksel Veritabanı Sistemleri, Yeni Nesil Veritabanı Sistemleri

ABSTRACT

Technological developments, increasing the data volume and increasing the size of the data that cannot be controlled by the structural data on the databases we call traditional systems are now insufficient with the cost, time and additional workload.

Increasing this volume of data with the new technological solutions today, such as storing, processing and analysis of the traditional systems less cost, less time and less workload has become possible to do.

In the scope of this thesis, the relationship between traditional database systems and new generation distributed file systems as well as the performance comparison between the two systems and the structural features of the new generation systems are discussed.

Keywords : Big Data, Hadoop, PostgreSQL, Performance Analysis, Traditional Database Systems, New Generation Database Systems

TEŞEKKÜR

Tez çalışmam boyunca gösterdiği her türlü destek ve yardımlarından dolayı çok değerli hocam Dr. Ögr. Üyesi Feridun Özçakır'a en içten dileklerimle teşekkür ederim.

Murat Menteşe

Nisan 2019

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

KISALTMALAR

RDBMS	A Relational Database Management System İlişkisel Veritabanı Yönetim Sistemi
SQL	Structure Query Language Yapılandırılmış Sorgu Dili
VTYS	Veritabanı Yönetim Sistemleri
CODASYL	Veri Dilleri Konferansı
HDFS	Hadoop Distributed File System Hadoop Dağıtılmış Dosya Sistemi
YARN	Yet Another Resource Negotiator Kaynak Yönetimi
HUE	Hadoop User Experience Hadoop Kullanıcı Deneyimi
MapReduce	Distributed Processing Of Large Data Sets Haritalama ve İndirgeme
ACID	An Acronym for Atomicity, Consistency Isolation, Durability İlişkisel Veritabanlarındaki İşlemler İçin Tanımlanmış Özellik Seti
GFS	Google File System Google Dosya Sistemi
OLAP	Online Analytical Processing Çevrimiçi Analitik İşleme
OLTP	Online Transaction Processing Çevrimiçi İşlem İşleme
ORDBMS	Object Relational Database Management System Nesne-İlişkisel Veritabanı Yönetim Sistemi

I. GİRİŞ

Günümüzde kullanılan internet teknolojileri aracılığıyla internette gezinirken bir taraftan da kullanıcılar kendi içeriklerini gezindikleri ortamlara bırakmaktadırlar. İnternet üzerindeki gezintilerinde ilgili yapılara video, resim ve metin yüklemeleri, e-posta ve anlık mesaj göndermeleri ve diğer aktiviteleri doğrultusunda internet ortamında üretilen verinin boyutu ve verinin büyüme hızı artmaktadır.

Günümüzün popüler sosyal medya platformu Facebook 2018 3.çeyrek diliminde 2,27 milyar aktif kullanıcı sayısına ve mobil ortamda 1,15 milyar günlük erişime sahip olduğu ve bu değerlerin her yıl %10 oranında arttığı öngörülmektedir. Her bir saniyede beş yeni Facebook profilinin oluşturulduğunu ve her 60 saniyede 510.000 yorum, 293.000 güncelleme ve 136.000 fotoğraf eklendiği düşünüldüğünde, Facebook ortamında kullanıcıların bir gün içinde üretmiş olduğu verinin büyüklüğü 500 terabyte'ın üzerindedir. Hergün 2,7 milyar beğeninin oluştuğu, her yarım saatte 105 terabyte büyüklüğünde verinin okunduğu bu ortamda veri işleme süreci Hadoop Hive ile yapılmaktadır. (Noyes, 2019)

Amazon'un "Metin İstatistikleri" adlı özelliği kullanılarak oluşturulmuş rapora göre romanlar ortalama 64.000 bin kelime, yaklaşık olarak 320.000 bin harfden oluşmaktadır. Bir romanın bilgisayar diskinde kapladığı alan yaklaşık 2 MB olduğu düşünüldüğünde Facebook'un bir gün içinde üretmiş olduğu 500 terabytelık verinin karşılığı 260 milyon romana denk gelmektedir. (Polat, 2012)

Bu veriler hızla artarken içerisinden anlamlı olanları ayıklamak günümüzde firmalar ve kurumlar için büyük önem taşımaktadır. Bu veriler aracılığıyla kullanıcıların eğilimlerini takip etmek ve kullanıcılar ile ilgili analizler yapmak ancak büyük verinin dağıtık bir yapıda hızlı bir şekilde işlenmesi ile mümkün olmuştur. Örneğin herhangi bir alışveriş sitesinde bir kullanıcının beğendiği bir ürüne baktığı andan itibaren ilgili sitede kullanıcı arka tarafta veri bırakmaya başlar. Kullanıcının arkada bıraktığı bu veriler aracılığıyla, kullanıcının daha sonra Google üzerinde bir arama yaptığında çıkan sonuçlarda ya da kullanıcının karşısına çıkan bir reklam içeriğinde kullanıcının alışveriş sitesinden bakmış olduğu ürün ya da ürünlerler ile bağlantılı veri ve reklamlar karşısına çıkmaktadır.

Firmalar ve kurumlar sosyal medya üzerinde oluşan veriler üzerinde de analiz yapıp aksiyonlar almaktadır. Örneğin, bir firmanın müşterisi firma ile ilgili olarak twitter mikroblog uygulaması üzerinden hoşuna gitmeyen olumsuz bir durumu paylaştığında, arka planda firma tarafından twitter daki firma ile ilgili mesajlarının derlenmesi ve yorumlanması ile birlikte kısa bir süre içinde firmanın müşterisine geri dönüş yaptığı görülmektedir, bu süreç dağıtık veri işleme sayesinde gerçekleşmektedir.

Tüm dijital veriler, kişilerin sosyal medya verileri bir ortamda depolanması gerekir. Depolanan bu veriler bilişimde veritabanı adını verdiğimiz sistemler içinde tutulur. Bu sistemlerde verinin içeriğine ve mimarisine göre birbirinden ayrılmaktadır.

Bu tez çalışmasında farklı nitelikte veritabanı teknolojilerinin birbirleri ile performans parametresi doğrultusunda karşılaştırması hedeflenmiştir. Yapılan bu tez çalışmasının amacı, geleneksel veritabanı sistemleri ile yeni nesil veritabanı sistemleri arasındaki performans farklılıklarını performans testleri aracılığıyla tespit etmek, elde edilen

sonuçlar aracılığı ile özellikle yeni nesil veritabanlarının en önemli özelliği olan yatay olarak büyümenin avantajını ele almaktır.

II. GENEL BİLGİLER

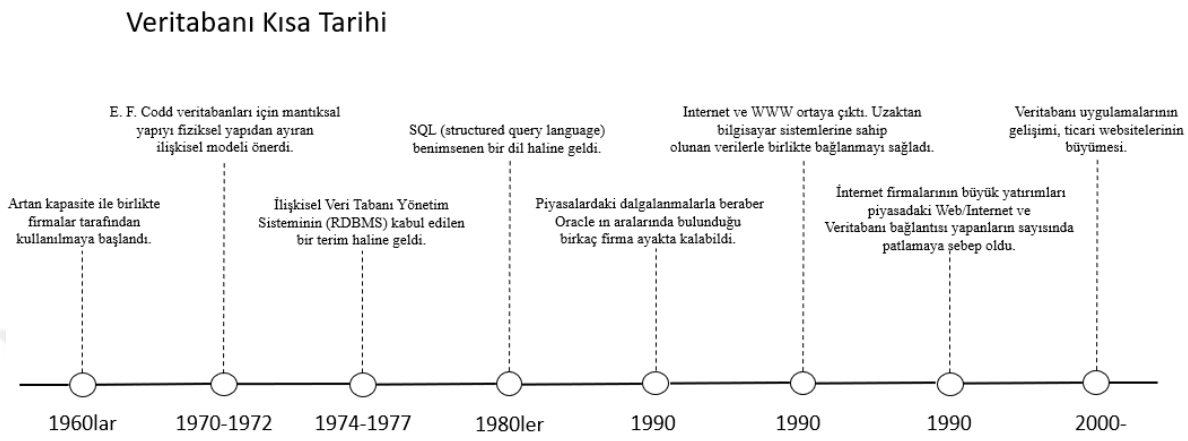
2.1. Veritabanı Nedir

Basit olarak veritabanı, birbirleriyle ilişkili olan verilerin birlikte tutulduğu yönetilebilen, güncellenebilen, taşınabilen ve kullanım amacına göre bir araya getirilmiş bilgiler topluluğudur. (Ullman, Widom, & Molina, 2009)

2.2. Veritabanı Tarihçesi

1960'lı yıllarda, bilgisayar kullanımı özel kuruluşlar tarafından maliyetlerin az olması sebebiyle kullanılmaya başlandı. 1970 ve 1972 yıllarına gelindiğinde E.F. Codd yayınladığı makalesiyle insanların veritabanları hakkındaki düşünme biçimini değiştirdi. 1974 ve 1977 yılları arasında iki büyük ilişkisel veritabanı sistemi prototipi oluşturuldu ve İlişkisel Veritabanı Yönetim Sisteminin (RDBMS) kabul edilen bir terim haline geldi. 1980'lerde, Yapılandırılmış Sorgu Dili (Structured Query Language) veya SQL, standart sorgu dili haline geldi. Bu yıllarda ilişkisel veritabanı sistemleri ve bilgisayar satışlarındaki hızlı artış ile birlikte hiyerarşik veritabanı modellerinin popülaritesinde önemli bir düşüşe neden olmasıyla ticari bir başarı haline geldi. 1990'lı yıllarda internetin gelişmesi veritabanı endüstrisinin üst düzey büyümesine yol açtı. Masaüstü kullanıcıları, eski verileri içeren bilgisayar sistemlerine erişmek için istemci-sunucu veritabanı sistemlerini kullanmaya başladı. Yine bu yıllarda yatırımların artmasıyla birlikte web/internet ve veritabanı bağlantısı yapanların sayısında artışa sebep oldu. 2000'li yıllara gelindiğinde internet endüstrisi bir gerileme yaşada da,

veritabanı uygulamaları yeni teknolojiler ile birlikte büyüme eğilimini sürdürmüştür. (Ullman, Widom, & Molina, 2009)



Şekil 2.1 Veritabanlarının Tarihsel Gelişimi

2.3. Veritabanı Mimarileri

Kullanım amaçlarına veya depolanma tiplerine göre değişik veritabanı mimarileri mevcuttur. Veritabanlarının kullanımına başlanmasından itibaren teknolojinin getirdiği faydalarla birlikte veritabanı mimarilerine yenilikler gelmiştir. Dosya veritabanı olarak adlandırılan veritabanı ilk kullanılan mimaridir. Günümüzde kullanılan en sık veritabanı mimarisi ise ilişkisel veritabanı dediğimiz mimaridir.

2.3.1. Düz Dosya Veritabanı Mimarisi (Flat File Database Model)

Verileri elektronik ortamda düzenlemenin ilk yöntemi düz dosya formatlarıydı. “Düz Dosya” bir dosya dolabına benzer şekilde organize edilmektedir. Bir dosya dolabı sistemi; alfabetik, tarih, konu gibi sıralarda yerleştirilen döküman dosyaları içerir. Dosyalardaki bilgiler, istenilen dökümanı anında bulunabilmesini sağlayacak şekilde indekslenmiş değildir. Aranan bulunuluncaya kadar sırayla taranması gerekir. Düz

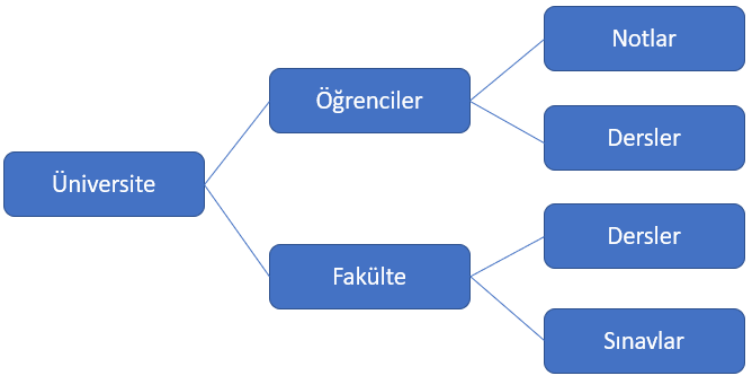
dosyada veri depolamak, küçük miktarda uygulama verisini basit bir şekilde depolamaktır. Dosya, karakter katarlarından oluşur ve basit bir tablo gibi düzenlenebilir. Tipik bir düz dosya, karakter ayraçlarla bölünür ve ASCII metin olarak depolanır. Eğer veriler karmaşık değilse, noktalı virgül ya da iki nokta karakterleri ayraç olabilir. Uygulamada özel bir veriye ihtiyaç varsa, bu veri bulununcaya dek düz dosyada arama yapılır. Dosyada veri aramak verimsiz bir iştir. Eğer uygulama kullanıldıkça verilerde artış bekleniyorsa, o zaman bu veriler düz dosyada depolanmamalıdır. Düz dosya, biçimi ya da boyutu değişmeyecek küçük miktarlardaki veriler için uygun bir veri depolama birimidir. Düz dosyalarda veri depolama sırasında bazı sorunlar ortaya çıkar. Bu sorunlardan biri verilerin bozulmasıdır. Dosya açık iken uygulama çökerse, dosyaya zarar gelmediğinden emin olunamaz. Eğer çökme sırasında uygulama dosyaya yazıyorsa, büyük olasılıkla veriye zarar gelir. Diğer bir sorun eşzamanlı erişimdir. Birden çok uygulama aynı anda aynı dosyayı açamaz. Veriye erişmek için sorgu dili yoksa arama yapılamaz. Günlük olarak satış kayıtlarının bir dosyaya yazıldığını varsayalım. Bu dosyada ad, ödeme şekli ve satılan ürün bilgisi olsun. Birden çok ürün satın alan müşterilerin adı ve ilgili bilgilerin her defasında dosyaya yazılacağından, müşterinin satın aldığı ürün kadar dosyada kaydı bulunur. Dosyadan tarih sırasına göre gözle arama yapılabilir, ancak bu arama sıralı olur. Her siparişte dosyaya aynı kayıtların gereksiz yere yazılmasından dolayı dosyalar büyür ve arama süresi artar. Düz dosyalar birbirleriyle bağlantı kuramadığından, farklı tarihlerde girilen aynı bilgilere sahip müşteri kayıtları, diğer düz dosyalarda da yer alır. (Elmasri & Navathe, 2011)

	Ad – Soyad	Kullanıcı Adı	Parola
Kayıt 1	Murat Çelik	mcelik	kLv62@1!
Kayıt 2	Hasan İspir	Hispir	jhGfsTU$
Kayıt 3	İbrahim Yazıcı	İyazici	uY$1d@mO

Şekil 2.2 Düz Dosya Veritabanı Mimarisi Örneği

2.3.2. Hiyerarşik (Klasörleme) Veritabanı Mimarisi (Hierarchical Data Model)

Bu veritabanı mimarisinde ise veriler klasör ve bunun altındaki diğer alt seviyedeki klasörler olarak depolanır. Yani bir ürün ağacını düşünecek olursak o ürünün bağlı olduğu kategori ve alt kategorileri şeklinde düşünülebilir. Bu veritabanı sisteminin en büyük götürüsü, disk üzerinde büyük yer kaplaması ve performans olarak diğer yöntemlere göre daha yavaş ve kullanım alanlarının sınırlı olmasıdır. (Önder, 2005)



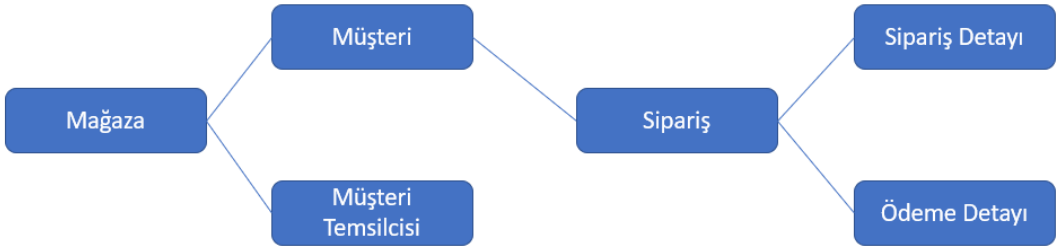
Şekil 2.3 Hiyerarşik Veritabanı Mimarisi Örneği

2.3.3. Ağ Modeli Veritabanı Mimarisi (Network Data Model)

Charles Bachman tarafından geliştirilmiş bir veritabanı modelidir. 1969 yılında Veri Dilleri Konferansı (CODASYL) Konsorsiyomu bu ağ modelini bir şartname haline getirmiş olup ikinci yayınını 1971 yılında tanıtmıştır. Sonrasında neredeyse tüm

uygulamaların temeli haline gelmiştir. Bu ağ modelinin kabul görmesindeki en büyük etken verinin diğer veriler ile ilişkili olmasından kaynaklanmaktadır. Bu modelin faydaları; (Speelpenning, Daux, & Gallus, 2001)

- Basit konsept ile birlikte hiyerarşik modele benzemesi, uygulanabilirliğinin basit ve zahmetsiz olmasıdır.
- Verilere erişim, hiyerarşik modelle karşılaştırıldığında daha kolaydır.
- Veri bütünlüğü, Bir ağ modelinde, ebebeyn ile çocuk bölümleri arasında her zaman bir bağlantı vardır. Çünkü ebebeyn-çocuk ilişkisine bağlıdır.
- Veri bağımsızlığı, Ağ modellerinde hiyerarşik modellerin aksine daha iyidir.

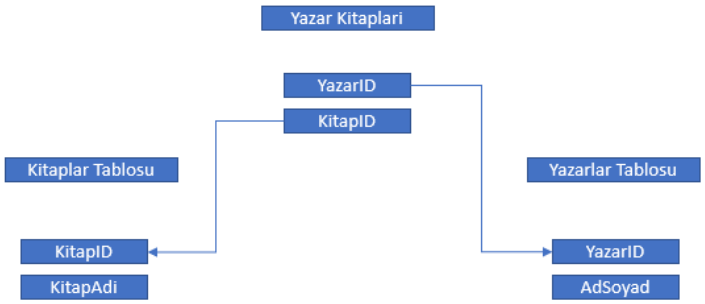


Şekil 2.4 Ağ Modeli Veritabanı Mimarisi Örneği

2.3.4. İlişkisel Veritabanı Mimarisi (Relational Data Model)

Klasörleme ve ağ modelinin, günümüzde verinin büyümesiyle birlikte talepleri karşılama konusunda yetersiz kalması, daha güçlü bir veritabanı mimarisi araması sürecini başlatmış ve bu sayede ilişkisel veri modeli geliştirilmiştir. E.F.Codd'un 1970'de kaleme aldığı "A Relational Model Of Data for Large Shared Data Banks" makalesi ile birlikte ilişkisel veritabanı mimarisinin gelişmesine öncülük etmiştir. İlişkisel veritabanı mimarisinin diğer modellerden ayıran en temel özelliği ilişkidir. Kurulan ilişkiler sayesinde, verinin içindeki ilişkiler birbirleriyle bağlantılı şekilde

modellenir. Bu sayede, ilişkisel bir veritabanı çeşitli ilişkilerin birbirleriyle bağlanmasıyla oluşmaktadır. Bu tip yapılarda ilişkiler satır ve sütunlardan oluşmuş iki boyutlu tablolarla karakterize edilir. Sütun, alan olarakta bilinir ve bir alana ait verileri içerir. Kayıt olarak bilinen satır, tablo içerisinde bulunan her bağımsız girişi içerir. Genellikle veritabanında her tablo için bir dosya bulunur. Dünya üzerinde en çok kullanılan veritabanı türüdür. İlişkisel veritabanı mimarisini, diğer mimarilerden ayıran en önemli özelliklerde biri, çok büyük ve yaygın kullanım alanı vardır ve bağımsız veritabanı tasarımında kullanılır. İkincisi ise, veritabanı yönetim sisteminin temelini oluşturmasıdır. (Kroenke, 1998)



Şekil 2.5 İlişkisel Veritabanı Mimarisi Örneği

2.4. Veritabanı Yönetim Sistemleri (VTYS)

Veritabanları bugün her işletme için gereklidir. Büyük bir Web sitesini (Google, Yahoo !, Amazon.com veya bilgi sağlayan binlerce küçük site) her ziyaret ettiğinizde, talep ettiğiniz bilgilere hizmet eden sahnenin arkasında bir veritabanı bulunur. Şirketler, tüm önemli kayıtlarını veritabanlarında saklamaktadır. Veritabanlarının gücü, on yıllar

boyunca gelişen ve bir veritabanı yönetim sistemi ya da daha yaygın olarak "veritabanı sistemi" olarak adlandırılan özel yazılımlardan oluşmaktadır. Bir VTYS, büyük miktarda veriyi etkin bir şekilde yönetmek ve uzun süreler boyunca güvenli bir şekilde kalmasını sağlamaktadır. Diğer bir ifadeyle; Veritabanlarının kurulması, oluşturulması, yönetilebilmesi ve kullanılabilmesini sağlayan yazılımlara veritabanı sistemi ya da veritabanı yönetim sistemi (VTYS ya da DBMS – Database Management System) denir. Bu sistemler mevcut en karmaşık yazılım türleri arasında yer almaktadır.

(Widom, Ullman, & Molina, 2008)

Genel olarak VTYS'lerin özellikleri;

- Kullanıcıların yeni veritabanları oluşturmasına ve özel bir veri tanımlama dili kullanarak şemalarını (verilerin mantıksal yapısı) belirtmesine izin verir.
- Kullanıcılara veriyi sorgulama ("sorgu", verilerle ilgili bir soru için veritabanı sözlüğü) sunma ve genellikle bir sorgu dili veya veri işleme dili olarak adlandırılan uygun bir dili kullanarak verileri değiştirme olanağı sağlar.
- Sorgular ve veritabanı değişiklikleri için veriye etkili bir şekilde erişebilmenize izin vererek, çok miktarda verinin depolanmasını sağlar.
- Başarısızlıklar karşısında veritabanının kurtarılması, birçok türde hata veya kasıtlı yanlışların önüne geçerek veritabanının çalışmasına olanak sağlar.
- Birçok kullanıcının veri erişimini kontrol ederek istikrarı sağlar. (Widom, Ullman, & Molina, 2008)

2.4.1. Bilinen Bazı Veritabanı Yönetim Sistemleri

Genel olarak en çok kullanılan ve bilinen VTYS'lere örnek verecek olursak; (Most Popular Database Management, 2018)

Tablo 2.1 Bilinen Veritabanı Yönetim Sistemleri

VTYS	**Firma**
Access	Microsoft
DB2	IBM
MySQL	Açık Kaynak
Oracle	Oracle
PostgreSQL	Açık Kaynak
SQL Server	Microsoft
Teradata	Teradata
MongoDB	Açık Kaynak

2.4.2. IBM DB2

Microsoft Access ve MySQL'e kıyasla performanslı olmakla birlikte daha küçük ölçekli işletmelerin kullanımı için maliyetlidir. IBM tarafından geliştirilmiş ilişkisel veritabanı yönetim sistemidir. Unix başta olmak üzere Linux, IBM i, Z/OS ve Windows sunucularında çalışabilmektedir. (IBM, 2017)

2.4.3. MySQL

Dünyada en çok kullanıcısı olan, açık kaynaklı bir VTYS'dir. Birden fazla iş, çok kullanıcılı, sağlam ve hızlı bir altyapıya sahip olmakla birlikte Linux altında daha yüksek performansla çalışmaktadır. UNIX, OS/2 ve Windows platformları için ücretsiz olan sunulmaktadır. MySQL, veritabanı gerektiren birçok yerde rahatlıkla kullanılabilir. Özellikle web sitelerini barındıran sunucularda en çok kullanılan veritabanıdır ve php,asp gibi programlama dilleri ile birlikte kullanılabilir. (Wikipedia, 2016)

2.4.4. PostgreSQL

PostgreSQL, SQL dilini en karmaşık veri iş yüklerini güvenli bir şekilde saklayan ve ölçekleyen birçok özellik ile birlikte kullanılan güçlü, açık kaynaklı bir ilişkisel veritabanı sistemidir. PostgreSQL'in kökenleri, Berkeley'deki California Üniversitesi'ndeki POSTGRES projesinin bir parçası olarak 1986 yılına dayanmaktadır ve 30 yıldan fazla aktif gelişime sahiptir. İşlem yapısı çok güçlüdür. Postgre sisteminde verinin güvenliği ön planda tutulmaktadır. 32 TB gibi bir veriyi tablo başına tutabilmektedir. "Transaction", "Inheritance" "trigger" ve "stored procedure" özelliklerine sahiptir. (PostgreSQL, 2018)

2.4.5. Oracle

Oracle veritabanı, Oracle Şirketi tarafından geliştirilmiş bir ilişkisel veritabanı yönetim sistemidir. İlk olarak 1977'de Lawrence Ellison ve diğer geliştiriciler tarafından geliştirilen Oracle DB, en güvenilir ve yaygın olarak kullanılan ilişkisel veritabanı sistemidir. Araştırmalara göre dünyanın en güçlü ve en güvenilir veritabanı olarak gösterilmektedir. Maliyet açısından yüksek olmakla birlikte büyük işletmeler tarafından tercih edilmektedir. Windows, Unix, Linux, vb. sistemlerde sınırsız sayıda tablo oluşturulabilmektedir. (Techopedia, 2018)

2.4.6. MS SQL

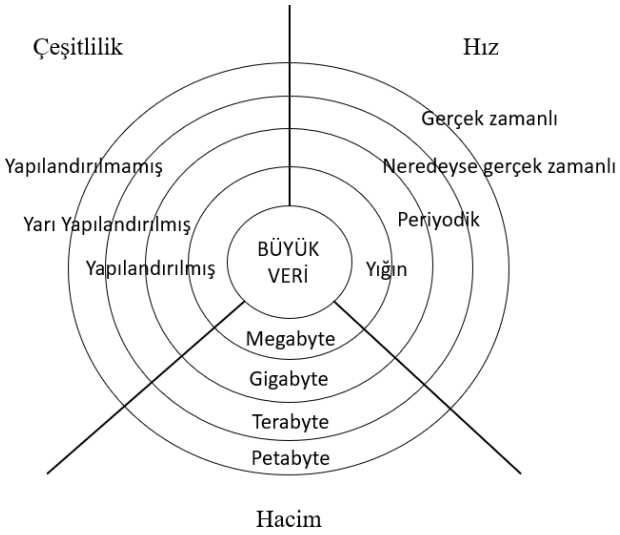
Microsoft tarafından geliştirilen ilişkisel veritabanı yönetim sistemidir. İlk olarak SQL Server 1.0 versiyonu ile birlikte 1989 yılında 16bit sunuculardа çalışmıştır. Tüm büyük veritabanı yönetim sistemleri gibi standart SQL dilini desteklemektedir. Ancak, SQL Server ayrıca kendi SQL uygulaması olan T-SQL'i de içerir. SQL Server, çeşitli

kullanıcı gereksinimlerini karşılamak için farklı özelliklerde ve fiyatlarda Enterprise, Standart, Workgroup ve Express adı altında çeşitli sürümleri bulunmaktadır. (Wikipedia, Microsoft SQL Server, 2018)

2.5. Büyük Veri Kavramı

Bugünkü anlamıyla büyük verinin ortaya çıkışı, verilerin bilgisayar ortamına aktarılması ve kaydedilmesi ile başlamıştır. 1970'li yıllarda verilerin özellikle analiz edilmesi ve saklanması için "veritabanı makineleri" nin yapılmasının büyük veri kavramının doğuşuna zemin hazırladığı kabul edilebilir. 1980'li yıllarda artan veri miktarını işlemek ve kaydetmek için tek bir bilgisayar yetersiz olmaya başlamış ve paralel veritabanı teknolojisi geliştirilmiştir. 1990'lı yıllara gelindiğinde internet servislerinin ortaya çıkışı ve hızla büyümesi gözlemlenmiş, sorgulanabilen ve insanoğlunun o güne kadar görmediği hızda büyüyen bir veri havuzu oluşmasına sebep olmuştur. (Chen, Mao, Zhang, & Leung, 2014)

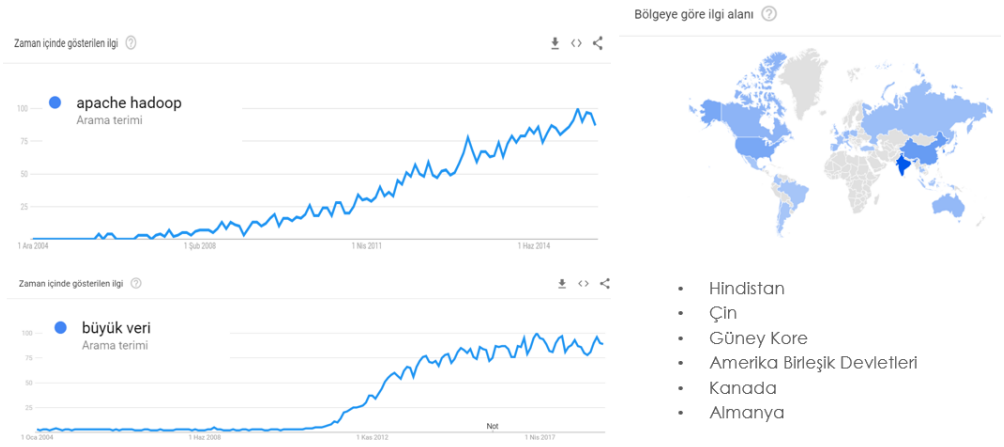
Büyük veri 3 temel birleşenden oluşmaktadır. Bunlar Hacim (Volume), Çeşitlilik (Variety) ve Hız (Velocity)'dir. Hacim verinin büyüklüğünü ifade ederken, çeşitlilik büyük verinin kaynaklarını ve daha çok yapılandırılmamış veri olmasını ifade etmektedir. Hız ise verinin ortaya çıkışı ve gerçek zamanlı işlenmesi/analizi ile ilgilidir. (Ivens, Alencar, & Cowan, 2014)



Şekil 2.6 Büyük Verinin Birleşenleri

Bu alandaki çalışmaların çoğalması ve farkındalığın artması ile birlikte yeni karakteristlik özelliklerin de bu bileşenler içerisine eklendiği belirtilmektedir. Bunlar, Doğruluk (Veracity) verinin analiz edilmek üzere düzenlendikten sonra işe yarar bir veri seti elde edilmesi, Değer (Value) verinin ortaya çıkardığı maddi değer, Değişkenlik (Variability) ise veriden elde edilebilecek özetlerin miktarı veri seti içerisindeki dağılımı, Karmaşıklık (Complexity) ise veriler arasındaki ilişkinin organize edilmesindeki ve analiz edilmesindeki zorluktur. (Benjelloun, 2015)

Büyük verinin hayatımıza girmesiyle bu büyük veriyi anlama ve işleme yöntemlerinin araştırılması da hızla artmıştır. Bu doğru aşağıdaki şekilde Google Trend Analizi ile görülmektedir.



Şekil 2.7 Büyük Veri Arama Geçmişi (google, 2018)

2.6. Dünyada Büyük Verinin Kullanım Çevreleri ile İlgili Örnekler

Dünyada büyük verinin değişik alanlardaki kullanımı ile ilgili başarılı uygulamaları mevcuttur. Bunlar;

2.6.1. Dünya Kupası 2014 Brezilya (World Cup 2014 Brazil)

2014 yılında yapılan FIFA Dünya Kupası şampiyonasında SAP ve Almanya Futbol Federasyonunun, turnuva içerisindeki oyuncuların gelişimini arttırmak için toplanan veriyi sonuç alınacak akıllı kararlara çevirecek bir sistem için işbirliği yapmışlardır. HANA platformu üzerinde çalışan bu akıllı çözüm sayesinde, oyuncuların antreman ve takım içi performansının arttırılması için tasarlanmış bir çözümdür. Oyuncuların üzerlerinde olan bir sensör ile 10 dakikada 10 oyuncu üzerinden 7 milyon gibi bir veri büyüklüğü elde edilmiştir. Bu çözüm ile birlikte takımın bir sonraki karşılaşmanın hazırlığının yapılmasında önemli rol oynamıştır. Bu analizler sonucunda Almanya'nın Dünya Kupasını kazanmasında büyük rol oynamıştır. Büyük verinin bu tip bir organizasyonda kullanılması, büyük veri için kullanımının çeşitliliği ile birlikte futbol

dünyasının gelişimine katkı sağlamıştır. Toplanan bu verinin karar aşamasında kullanılması büyük veri ile birlikte elde edilen sonuçları daha da önemli hale getirmektedir. (SAP, 2014)

2.6.2. Wallmart

Walmart'ın 2004 senesinde müşteri kitlesinin ürün birliktelikleri ile birlikte, ürünün maliyeti, alışveriş sepetinde başka hangi ürünün olduğu, hava durumunun günün saat dilimlerine etkisi gibi durumları veri setleri üzerinde analiz yapmışlardır. Bir fırtına esnasında sadece gerekli ihtiyaçların değil aynı zamanda bir Amerikan yiyeceği olan Pop-Tarts satışlarının da yükseldiğini görmüşlerdir. Bu sayede mağaza ön raflarındaki kasırga ürünlerinin yanına Amerikan Pop-Tarts ları da ekleyerek satış rakamlarını önemli bir şekilde arttırmıştır. Geçmişte sadece mağaza personelinin fark edebileceği bir durumu büyük verinin analizi sayesinde, oluşturulan korelasyonda hızlı bir biçimde gerçekleştirmiş ve şirket için fayda olarak geri dönüş sağlamıştır. (Mayer & Cukier, 2013)

2.7. Yapılan Akademik Çalışmalar

Dünya genelinde Hadoop ve büyük veri ile ilgili olarak birçok tez, proje ve araştırma çalışmaları yapılmıştır. Bunlardan bazılarına değinecek olursak;

Madrid Politeknik Üniversitesinde doktora tezi olarak yazılmış olan MapReduce tabanlı sistemlerde kaynakların yönetilmesi ve bu sistemlerin optimize edilerek daha performanslı çalışması için gerekli algoritmaların yazılarak sisteme entegre edilmesini amaçlayan bir çalışma yapılmıştır. (Memishi, 2016)

Yine Danimarka Aalborg Üniversiteside doktora tezi olarak yazılmış, büyük verinin depolanma sorununun çözümü ve depolanan veri üzerinde parallel sorguların çalıştırılması, mevcut veri depolama sorunlarının ve zorluklarının çeşitli yönlerini ele alan bir çalışma mevcuttur. (Liu, 2012)

Türkiye'de ise Hadoop ile ilgili Kocaeli Üniversitesinde yüksek lisans tezi olarak, Görüntü dosyalarının MapReduce tekniği ile işlenebilmesi için bir Hadoop eklentisi geliştirilmiş ve bu eklenti ile birlikte görüntü dosyalarında yüz saptama uygulaması yapılmıştır. (Demir, 2012)

Son olarak Sakarya Üniversitesinde yüksek lisans tezi olarak, kullanılan MapReduce algoritmaların analizleri yapılarak performansa etki eden parametreleri tespit edip, bu parametreleri düzenlemek suretiyle Hadoop yapısında performans artışı sağlamak hedeflenmiştir. (Şarkışla, 2015)

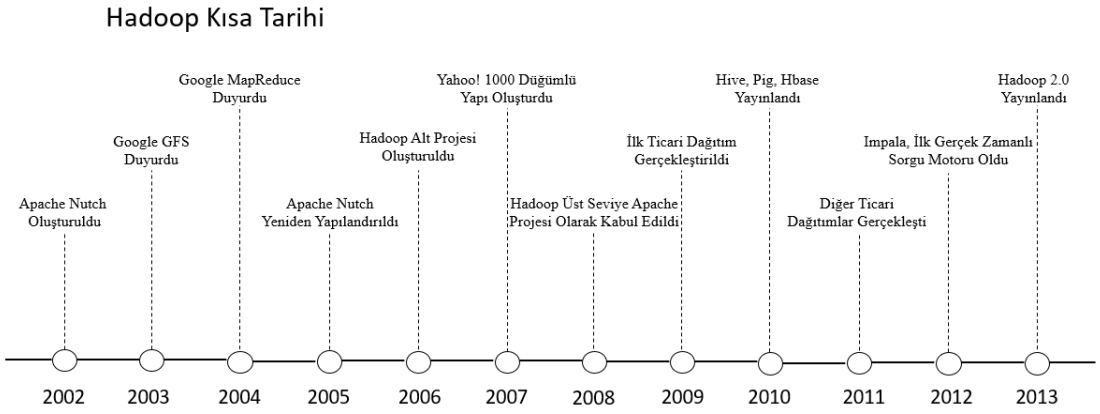
2.8. Yeni Nesil Veritabanı Sistemlerinden Hadoop

Devasa boyuttaki veri kümeleri ile birlikte, oluşturulmuş olan yapı içerisindeki çok sayıda ki sunucu üzerinde paralel olarak işlem yapmamızı sağlayan ve Hadoop Distributed File System (HDFS) olarak bilinen bir dağıtık dosya sistemi ile birlikte Hadoop MapReduce özelliklerini bir araya getiren, geliştirilmesi Java ile yapılmış açık kaynaklı bir yazılımdır. (Patodi, 2011)

Doug Cutting tarafından geliştirilen Hadoop un temelleri, açık kaynak kodlu bir arama motoru geliştirilmesi fikriyle ortaya çıkmıştır. Bu fikir Apache Nutch projesi ile 2002 yılında başlamış ve proje kapsamında, içerikleri çekme ve arama sistemi geliştirme hedeflenmiştir. Ancak projenin başlangıç haliyle milyarlarca sayfaya ölçeklenmesi web

üzerinden mümkün görülmemiştir. Bu doğrultuda, 2003 yılında Google arama motorunun kullandığı dağıtık dosya sistemi mimarisi (Google File System, GFS) ile birlikte 2004 yılında Google'ın geliştirdiği dağıtık veri işleme modeli ve MapReduce programlama yapısından esinlenerek, Nutch için dağıtık bir dosya sistemi ve MapReduce modeli gerçekleştirilmiştir.

İlerleyen dönemlerde, Nutch projesi için oluşturulan altyapının arama dışında da farklı alanlarda da kullanılabileceği düşüncesiyle, Hadoop adı altında bir alt proje oluşturulmuştur. Doug Cutting'in Yahoo'ya katılmasıyla, Hadoop'un geniş ölçekte bir sisteme dönüştürülmesi için gerekli kaynaklar sağlanmıştır. 2008 yılında, Yahoo, arama indeksinin 10.000 çekirdekli bir Hadoop kümesi tarafından oluşturulduğunu duyurmuştur. Hadoop, 2008 yılından itibaren üst seviye bir Apache projesi olarak yer almaktadır. (White, A Brief History of Hadoop, 2011)



Şekil 2.8 Hadoop Tarihsel Gelişimi

2.9. NoSQL Kavramı

NoSQL, ilişkisel veritabanı sistemlerine alternatif bir çözüm olarak ortaya çıkan, yatay olarak ölçeklendirilen bir veri depolama sistemidir. İnternetin gelişmesiyle beraber, sistemlerde çok fazla kullanıcının aktif rol alması, birçok ihtiyacın yanında veritabanı şemasının artan bir sıklıkla değiştirilmesi zorunluluğunu ortaya çıkmasına yol açmıştır. Hepimizin kullandığı ilişkisel veritabanlarındaki hiyerarşi, önce tabloları ve her tabloya ait sütunları oluşturmak sonrasında ise bilgileri satır satır eklemektir. Fakat burada karşımıza çıkan bir sıkıntı, tanımı olmayan alana bilgi kaydetmektir. Bu sıkıntıdan kurtulmak için yapmamız gereken öncelikle tabloyu tekrardan ele alıp yeni sütunlar eklemektir. Bununla birlikte veritabanındaki tüm tablo ve ilişkileri etkileyecek durumlar da ortaya çıkabilmektedir. Bu işlemi gerçekleştirmek sistemi tekrardan ele almayı gerektirdiği için çok maliyetli olabilmektedir. İlişkisel Veritabanı Sistemlerinde maliyetin yükselmemesi adına yapıyı önceden çok iyi planlamak gerekmektedir. Fakat NoSQL veritabanları sistemlerinde, bilgilerin kayıt altına alınması adına bu yapıyı sonradan kurmak ek bir maliyet getirmemekte ya da az bir maliyet getirmektedir. Nedeni ise NoSQL sistemlerde tablo ve sütun kavramının olmamasıdır. NoSQL sistemlerde yeni bir alana ihtiyacınız olduğu durumlarda kaydı direk eklemeniz yeterli olmaktadır. NoSQL sizin yerinize alanı oluşturur ve değeri kaydeder. Kayıtların maliyetsizce gerçekleşmesinin nedeni ise verilerin tablo ve sütunlarda saklanması yerine JSON ve XML formatına benzer yapıda saklanmasıdır. Tüm bu anlatımlardan "NoSQL veritabanları, ilişkisel veritabanlarından çokdaha iyidir" gibi bir sonucu çıkarmamamız gerekmektedir. Çünkü ikisini de yerine göre kullanmakta fayda bulunmaktadır. Örneğin, ilişkisel olayların çok yoğun gerçekleştiği bir bankacılık uygulamasında NoSQL bir veritabanı kullanmamız uygun değildir. NoSQL, Fire and

Forget prensibi ile çalıştığı için bankacılık, alışveriş gibi para üzerinden işlem yapılan kritik uygulamalarda kullanılmamalıdır. Aksine verinin %100 önem arz etmediği durumlarda kullanımı daha uygundur. (Robinson, Eifrem, & Webber, 2015)

2.10. Büyük Veride NoSQL

Yüksek kapasiteli bir web sitesi üzerinden gelen veriler, başarılı bir pazarlama kampanyasına hızlı bir şekilde yanıt vermesi gerektiğinde yavaşlamaktadır. Facebook, Twitter ve diğer bir çok kuruluş artan kullanıcı sayısı ile birlikte bu sorun ile mücadele etmektedirler. Günümüzde web şirketleri arasında giderek daha popüler hale gelen bir veritabanı türü olmuştur. Yapılandırılmamış verilerin NoSQL ile birlikte geleneksel ilişkisel veritabanlarına göre daha kolay ölçeklenebilirlik ve gelişmiş performans sağlamaktadır. (Robinson, Eifrem, & Webber, 2015)

III. GELENEKSEL VE YENİ NESİL VERİTABANI MİMARİLERİ

Veri kaynaklarından elde edilen verinin büyümesiyle birlikte geleneksel veritabanı sistemlerinin yetersiz kalması sonucunda, oluşan bu verinin toplanması, düzenlenmesi ve yorumlanması için yeni nesil veritabanı sistemlerine ihtiyaç duyulmaya başlanmıştır. Bu çalışmada geleneksel veritabanı mimarisine sahip PostgreSQL ile yeni nesil veritabanı mimarisine sahip Apache Hadoop arasında bir karşılaştırma yapılmıştır.

3.1. PostgreSQL Mimarisi

Sql yapı standardına göre tasarlanmış, ücretsiz ve açık kaynak kodlu bir ilişkisel veritabanı yönetim sistemidir. Tüm Unix sistemlerinde ve NT tabanlı Windows sistemlerinde çalışmaktadır. PostgreSQL performanslı kılan nedenlerden birisi güvenlik ve özellik bakımından sürekli olarak geliştirilmekte olmasıdır. Zengin veri tiplerini destekler (Array, JSON, integer, boolean, vb.) (Postgre, 2017)

3.1.1. PostgreSQL Tarihçesi

PostgreSQL'in geliştirilmeye başlanması; 1977'de Ingres adına sahip bir proje ile birlikte Berkeley üniversitesinde geliştirilmeye başlanmıştır. 1986-1994 yıllarında Ingres projesi adını değiştirerek Postgres olarak değiştirmiştir. Illustra tarafından satın alınıp Informix olarak geliştirilmeye devam edilmiştir. 1994 yılına gelindiğinde ise SQL modüllerinin eklenmesiyle birlikte adı Postgres95 olmuştur. 1996 yılında bir çok gönüllü e-mail listeleri ile birlikte Postgres95'i geliştirmek için çalışma yapmışlardır.

1996 yılında ise SQL standartlarının desteklemesiyle PostgreSQL adını almıştır. (Postgre, 2017)

3.1.2. PostgreSQL Genel Özellikleri

PostgreSQL yani kısaca Postgres, SQL dilini geliştirerek kullanan, verileri hızlı ve güvenli bir şekilde tutan açık kaynaklı, obje tabanlı ilişkisel (ORDBMS – Object relational database management system) bir veritabanı yönetim sistemidir. Object-relational olması Postgres'i esnek ve güçlü kılarken standart ilişkisel veritabanlarının desteklemediği bir takım önemli özellikleri destekleyerek rakipleri olan MySQL, Maria DB, Firebird gibi veritabanlarına karşı avantaj sağlamaktadır.

- Açık kaynak kodlu olması
- Ücretsiz olarak sunulması
- Platformlara bağımsız olması
- Gönüllülük esasına dayalı dünyanın birçok yerinde geliştiricisi olan
- ANSI SQL uyumlu
- Hemen bütün dillerde veritabanı programlama imkânı (C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, vb.) (Postgre, 2017)

Tablo 3.1 PostgreSQL'in Veritabanı Sınırları

PostgreSQL'in Sınırları	
Max veritabanı büyüklüğü	Sınırsız
Max tablo büyüklüğü	32 TB
Max satır büyüklüğü	1.6 TB
Max kolon büyüklüğü	1 GB
Max satır sayısı	Sınırsız
Bir tabloda max kolon sayısı	250-1600 (kolon veri tipine göre)
Tablo başına max index sayısı	Sınırsız

3.1.3. PostgreSQL Veritabanı Yönetim Sisteminin Avantajları

Ücretsiz, açık kaynak kodlu oluşu ve diğer veritabanlarında bulunan özellikleri ve daha fazlasını kapsamaktadır. Tüm Unix ya da Unix türevi tüm işletim sistemlerinde çalışmaktadır. Açık kaynak oluşu nedeniyle geliştirilmeye açıktır ve işletmelere özgü yazılımlar sayesinde hemen hemen tüm işletmelere entegre edilebilmektedir. (Wikipedia, PostgreSQL, 2018)

3.2. Apache Hadoop ile Yeni Nesil Veritabanı Sistemleri

Teknolojinin gelişmesiyle, verinin büyüme hızı ile birlikte boyutu da artmıştır. Devasa boyutlara gelen veri ile birlikte ihtiyaçlar da oluşmaya başlamıştır. Bunlardan ilki, boyutu büyük olan verinin hızlı bir şekilde kayıt altına alınması, diğeri ise kayıt altına alınan verinin hızlı bir şekilde erişilebilir ve analiz edilebilir olmasıdır. Sosyal medya, internet arama motorları, müşteriyle daha yakından iletişim kuran firmalar, ellerindeki büyük veriyi analiz ederek, operasyon verimliliklerinin yükselmesiyle gelirlerini arttırmaktadırlar. Boyutu büyük olan verinin işlenmesi söz konusu olduğunda, oluşan sorunların başında maliyet gelmektedir. Büyük çaptaki verinin hem zaman, hem donanım maliyetlerini en düşük seviyede tutmakla birlikte, ileriye dönük olarak oluşturulacak yapının esnek olması, ileride oluşadan ek maliyetlerin önüne geçmelidir. İhtiyaca çözüm olarak Apache Hadoop, standart sunuculardan oluşan geniş kümeler üzerinde, veriyi işlemek amaçlı, Java kodlaması ile geliştirilmiş açık kaynak kodlu bir kütüphanedir. Verinin büyük olması, verinin işleme sürecinde dağıtık bir sistem üzerinde kullanmakla birlikte veriyi paralel işlemeye de yönlendirmektedir. Hadoop dosya sistemi olan HDFS, dağıtık dosya sistemi yapısıyla ve bu veriyi okumada kullanmış olduğu MapReduce teknolojisiyle, atıl durumdaki sunuculardan oluşan

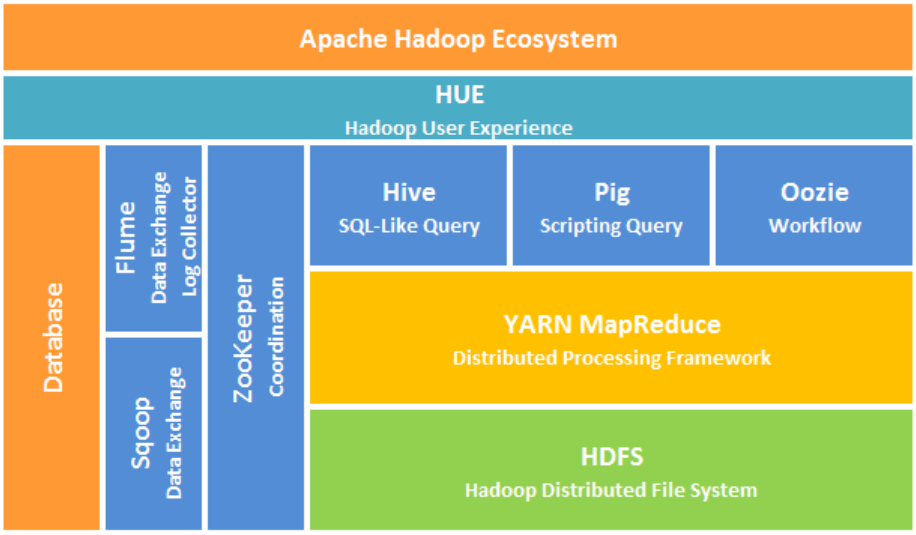
yapının tüm disklerinin tek bir disk olacak şekilde dizayn etmekle birlikte bu disk üzerine yazılan verinin paralel okunabilmesini sağlamaktadır. Hadoop ile birlikte büyük boyuttaki bir çok dosya hızlı bir şekilde depolanabilir ve yüksek maliyet gerektirmeden işlenebilir olmaktadır. (Akdoğan, 2016)

3.2.1. Apache Hadoop Özellikleri

- Büyük veri yığınlarının işlenmesi; Hadoop ile sosyal medya ve IoT (Internet of Things) uygulamaları gibi büyük veri yığınlarından gelen çok miktarda veri işlenebilir ve depolanabilir.
- Hesaplama gücü; Hadoop verilerin işlenmesi için dağıtılmış modeller kullanarak büyük verileri oldukça hızlı işleyebilir.
- Hata toleransı; Hadoop herhangi bir yazılım ve donanım hatasından korunma sağlar. Dağıtılmış modeldeki bir düğüm düşerse, diğer düğümler çalışmaya devam eder. Verilerin kopyaları da saklanır.
- Esneklik; Hadoop kullanarak çok fazla veri saklanabilir. Verilerin ön işlemeye tabi tutulmasına gerek yoktur.
- Düşük Maliyet; Hadoop açık kaynaklı bir platformdur ve kullanımı ücretsizdir. Büyük miktarda veriyi depolamak için yüksek kapasiteli tek bir donanım kullanmak yerine düşük maliyetli birden fazla donanım kullanarak maliyeti düşürmektedir.
- Ölçeklenebilirlik; ihtiyaçlar doğrultusunda ana yapıya düğümler (ek donanımlar) eklenerek kolayca büyütülebilir. En düşük seviyede yönetici yetkisi ile süreç planlanabilir. (Techsparks, 2016)

3.3. Temel Hadoop Bileşenleri

En temel Hadoop bileşenleri HDFS, Map-Reduce ve YARN'dır. Diğer bileşenler ise Hive, Pig, Sqoop, Flume, Oozie, Hue, Spark vb.



Şekil 3.1 Hadoop Ekosistemine Genel Bakış

3.3.1. HDFS (Hadoop Dağıtılmış Dosya Sistemi)

Hadoop dağıtılmış dosya sistemi, çok büyük dosyaları makineler arasında büyük bir kümede güvenilir bir şekilde saklamak için tasarlanmıştır. HDFS ortamına yazılan dosyalar, varsayılan 64MB'lık bloklar ile birden fazla sunucu üzerinde 3 kopya olarak RAID benzeri bir yapıyla yedeklenir. Bu sayede veri kaybının önüne geçilmiş olur. (White, The Hadoop Distributed Filesystem, 2011)

3.3.2. MapReduce (Haritalama ve İndirgeme)

Büyük miktarda veri kümesini güvenilir, hataya dayanıklı bir şekilde paralel olarak işleyebilmek amacıyla kullanılan bir yöntemdir. Map ve Reduce fonksiyonlarından oluşmaktadır. Map fonksiyonu verinin filtrelenmesi için kullanılırken, Reduce

fonksiyonu ise filtrelenmiş veriden sonuç elde etmek için kullanılır. Map ve Reduce programı yazıldıktan sonra Hadoop sistem mimarisi üzerinde çalıştırılır ve bu işlem hadoop kümeleri üzerinde paralel olarak verinin işlenmesini ve sonucunun da bir araya getirilmesi ile devam etmektedir. Hadoop'u farklı kılan, işlenmekte olan verinin daima ilgili iş düğümünün yerel diskinden okuması ile birden fazla iş akışını aynı anda işlemesiyle birlikte doğrusal olarak ölçeklenmesinden gelmektedir. MapReduce, JobTracker ve TaskTracker olmak üzere 2 ana kısımdan oluşmaktadır. JobTracker çalıştırılmış olan MapReduce işinin Hadoop kümesi üzerinde dağıtılarak çalıştırılmasından sorumlu olmakla birlikte bu iş parçalarının çalışma esnasında hata alması durumunda işi sonlandırma ya da yeniden başlatılmasından da sorumludur. TaskTracker ise, verinin tutulduğu sunucu yapılarında çalışır ve JobTracker'dan atanacak işleri bitirmek üzere iş parçacığı talep eder. JobTracker, Ana Düğüm'ün yardımıyla Veri Düğümü'nün yerel diskindeki veri setine göre en uygun Map işini TaskTracker'a verir. Atanan iş parçacıkları tamamlanır ve sonuçlar yine HDFS üzerinde bir dosya parçası olarak yazılırak program sonlanır. (Apache, 2008)

3.3.3. YARN (Kaynak Yönetimi)

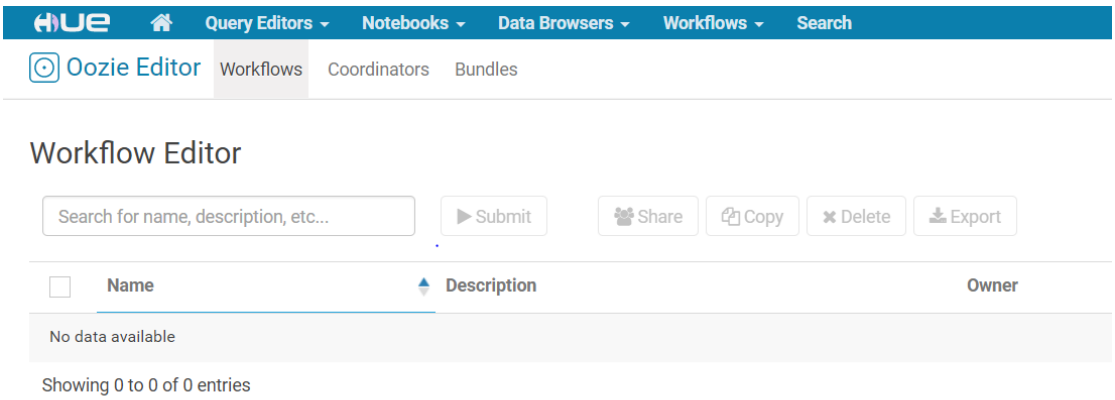
YARN (Yet Another Resource Negotiator), dağıtık olarak çalışan Spark gibi uygulamalar için sistem içerisinden kaynak yönetimi sağlamaktadır (CPU, RAM). Çalışacak olan Spark ya da MapReduce kodları geliştirilirken kaynak yönetimine müdahele etmemize gerek yoktur. Otomatik olarak arka planda iş analiz edilerek ihtiyaca göre ilgili kaynak YARN tarafından çalıştırılan işe aktarılmaktadır. (Apache, 2018)

YARN genel anlamda üç ana sistemi kontrol etmektedir. Bunlar;

- ResourceManager, Hadoop ekosistemi içerisindeki kaynakların takip edilmesi ve uygulamaların yönetilmesinden sorumludur.
- NodeManager, Hadoop ekosistemindeki Veri Düğümü sunucuları üzerinde çalışır ve kendisini belirli aralıklarla ResourceManager'a tanıtmakla birlikte çalıştığı sunucu hakkında bilgileri iletmekle sorumludur.
- Container, belirli kaynakların bir araya gelerek oluşturduğu bileşendir. Sistem içerisinde çalışan işlere atanmaktadır ve bir iş birden fazla kaynağa sahip olabilmektedir. (Apache, 2018)

3.3.4. HUE (Hadoop Kullanıcı Deneyimi)

HUE (Hadoop User Experience), açık kaynaklı bir web arabirimidir. Hive, Impala, Pig, MapReduce ve Spark gibi uygulamalar için editör görevi görür. Ayrıca verileri tarama, sorgulama ve görselleştirme için kullanılır. HUE üzerinde iş akışları oluşturulabilir ve bu akışlar zamanlayıcı ile çalıştırılabilir. (Wikipedia, Hue (Hadoop), 2017)



Şekil 3.2 HUE Kullanıcı Deneyimi Arayüzü

3.3.5. Apache Hive (SQL Sorgulama Dili)

Apache Hive (SQL-Like Query), SQL benzeri bir arayüz yardımıyla Hadoop üzerinde analiz ve sorgulama işlemlerini yapmak için kullanılmaktadır. Hive, verileri özetleme, geçici sorgulama ve büyük hacimli verilerin analizi için tasarlanmıştır. Hive'ın SQL benzeri yapısı sayesinde kullanıcılar kendi tanımlı fonksiyonlarını kolayca entegre edebilmektedirler. (Hive, 2017)

3.3.6. Apache Pig (Komut Dosyası Sorgulama)

Apache Pig (Scripting Query), büyük veri setlerini analiz etmek için kullanılan bir platformdur. Veri kullanıcılarına yüksek seviye java kodu yazmadan hadoop dağıtık mimarisini kullanarak analizler için yardımcı olmaktadır. En önemli özelliği Pig Latin dilini kullanarak kullanıcılar Java bilmeseler bile gelişmiş prosedurel bir script dili ile veriye ulaşma ve işleme imkanı sunar. Bu sayede MapReduce programları yazmaya gerek kalmamaktadır. (Apache Pig, 2018)

3.3.7. Apache Oozie (İş Akışı)

Apache Oozie (Workflow), hadoop işlerini yöneten bir iş akışı zamanlayıcı sistemidir. Hive, Pig, Spark gibi yapılan işleri belirli periyotlarda ya da belirli bir iş akışına göre çalıştırmak için kullanılmaktadır. Oozie ölçeklenebilir, güvenilir ve genişletilebilir bir sistemdir. (Oozie, 2017)

Genel özelliklerinden bahsedecek olursak;

- Açık kaynak kodlu ve ücretsizdir.
- Priyodik işlemler için uygundur. (dakika, saat, hafta..)
- Birçok büyük veri kütüphanesini destekler. (Hadoop, Hive, Pig, Sqoop..)

- İşlemler sırasında mail atabilir.
- Fork işlemlerini destekler, Bir MapReduce işlemi bittikten sonra aynı anda paralel bir şekilde devam edecek Pig ve Spark işlemi başlatabiliriz.
- Hadoop dosya sistemi komutları kullanılabilir.
- Linux komutları çalıştırılabilir.
- Java projesi çalıştırılabilir. (Oozie, 2017)

3.3.8. Apache Sqoop (Veri Değişimi)

Apache Sqoop (Data Exchange), kullanmakta olduğumuz veritabanlarından veri aktarmak için oluşturulmuş açık kaynaklı bir Hadoop bileşenidir. JDBC bağlantısını kullanan tüm veritabanlarıyla da çalışabilmektedir. Sqoop ile hem içe aktarma hemde dışa aktarma yapabilmektedir. En büyük yeteneği ise veri aktarım sırasında belirtilen parametre ile birlikte işin paralel olarak çalışabilmesidir. Bu sayeden aktarım çok daha hızlı gerçekleşmektedir. (Apache Sqoop, 2012)

3.3.9. Apache Flume (Günlük Toplayıcı)

Apache Flume (Log Collector), büyük miktardaki günlük verilerin toplanması ve birleştirilmesi için kullanılır. Akış, veri akışına dayanan basit ve esnek bir mimariye sahiptir. Çevrimiçi analitik uygulamaya olanak tanıyan basit genişletilebilir veri modeli kullanır. Genellikle websitelerinin accesslog'larının alınması için kullanılmaktadır. (Flume, 2017)

3.3.10. Apache Zookeeper (Kordinasyon)

Apache Zookeeper (Coordination), açık kaynaklı bir Apache projesidir. Hadoop ekosistemi içerisindeki servislerin merkezi olarak senkronize bir şekilde çalışmasından sorumludur. (Zookeeper, 2017)

3.4. Hadoop Platformları

Günümüzde en fazla kullanılan platformların başında Hortonworks Ambari, Cloudera Manager ve MapR'dir

3.4.1. Hortonworks Ambari

Tamamen açık kaynaklı yönetim platformudur. Hadoop mimarisinin kurulumu, yönetimi ve güvenliği açısından işletmelerin tercihleri arasında olmakla beraber en büyük avantajı ücretsiz olarak platformu kurabilmektir. (Hortonworks, Apache Ambari, 2018)

Ambari'nin avantajları;

- Basitleştirilmiş kurulum, yapılandırma ve yönetim.
- Merkezi güvenlik.
- Mevcut düzen üzerinde kolay kontrol.
- Genişletilebilir ve özelleştirilebilir.
- Ücretsiz olması.

3.4.2. Cloudera Manager

Claudera firması tarafından çıkarılmış olan Hadoop yönetim sistemidir. Yönetim açısından diğer yönetim araçlarına göre benzerlik göstermektedir. (Cloudera, 2018)

Claudera Manager avantajları;

- Otomatik kurulum ve yapılandırma.
- Özelleştirilebilir, izlenebilir ve raporlanabilir.
- Sorunların kolayca çözülebilmesi.
- Yüksek sürdürülebilirlik.

3.4.3. MapR

Diğer yönetim araçlarından farkı olarak birçok veri platformu üzerinde çalışmaktadır. Bunlardan biride Hadoop'dur. Kurulumu lisansa bağlıdır ve ücretlidir. (Why MapR, 2018)

MapR'nin avantajları;

- Yüksek kullanılabilirlik.
- Gerçek zamanlı akış.
- Veri entegrasyon kolaylığı.
- Veri güvenliği.

3.5. Geleneksel ile Yeni Nesil Veritabanı Arasındaki Yapısal Farklılıklar

Yeni nesil veritabanı sistemi olarak Hadoop'u ele aldığımızda, RDBMS (Relational Database Management System – İlişkisel Veritabanı Yönetim Sistemi) arasında yapısal

olarak birçok fark bulunmaktadır. Hadoop bir veritabanı değildir, temel olarak bilgisayar kümesinde büyük veri kümelerini işlemek ve saklamak için kullanılan dağıtılmış bir dosya sistemidir. (Bista, 2018)

Hadoop HDFS ve MapReduce olmak üzere iki ana çekirdeğe sahiptir. HDFS, bilgisayar kümeleri arasında büyük miktarda veri depolamak için kullanılan depolama katmanıdır. MapReduce, büyük veri kümelerini çeşitli veri bloklarına bölerek işleyen bir programlama modelidir. Bu bloklar daha sonra bilgisayar kümelenmesinin içinde bulunan farklı makineler üzerindeki düğümlere dağıtılır. (Ineni, 2015)

Öte yandan, RDBMS verileri birkaç satır ve sütundan oluşan tablo halinde depolamak için kullanılan bir veritabanıdır. Bu tablolarda bulunan verileri güncellemek ve bunlara erişmek için SQL dilini kullanmaktadır. (Özcan, 2013)

Geleneksel veritabanı sistemleri ile yeni nesil veritabanı sistemleri arasındaki bazı yapısal farklılıklara aşağıdaki alt maddelerde değinilmektedir.

3.5.1. Veri Hacmi

Saklanmakta ve işlenmekte olan veri miktarı anlamına gelir. RDBMS, veri hacmi düşük olduğunda daha iyi çalışır. Ancak veri boyutu büyük olduğunda RDBMS istenen sonuçları zamansal ve performans açısından başarılı bir şekilde ortaya çıkaramaz. Öte yandan, Hadoop veri boyutu büyük olduğunda daha iyi çalışmaktadır. Geleneksel RDBMS'ye kıyasla çok miktarda veriyi kolayca etkili bir şekilde işleyebilir ve depolayabilir. (Waleed, 2018)

3.5.2. Mimari

Son güncellemeler sonrası Hadoop'un şu temel bileşenleri vardır;

- HDFS (Hadoop dağıtılmış dosya sistemi),
- Hadoop MapReduce (Büyük veri kümelerini işlemek için bir programlama modeli),
- Hadoop YARN (Bilgisayar kümelerinde bilgi işlem kaynaklarını yönetme),
- ACID (İlişkisel veritabanlarındaki işlemler için tanımlanmış özellik setidir).

3.5.3. Çıktı

Verim, belirli bir süre içinde işlenen toplam veri hacmini ifade eder. RDBMS, Apache Hadoop ile karşılaştırıldığında daha düşük bir verimle çalışmaktadır. Bu neden ile Hadoop geleneksel ilişkisel veritabanı yönetim sistemlerine göre daha çok tercih edilmektedir. (Munvo, 2013)

Geleneksel RDBMS mimarisinde ise; Atomiklik, tutarlılık, izolasyon ve dayanıklılık gibi ACID özelliklerine sahiptir. Bu özellikler, bir işlem veritabanında gerçekleştirildiğinde veri bütünlüğü ve doğruluğunu korumak ve sağlamaktan sorumludur. Bu işlemler, bankacılık sistemleri, imalat sanayi, telekomünikasyon, çevrimiçi (online) alışveriş, eğitim vb. sektörlerde önem arz etmektedir. (Bista, 2018)

3.5.4. Veri Çeşitliliği

Veri çeşitliliği genellikle işlenecek veri türünü ifade eder. Veri çeşitleri yapılandırılmış, yarı yapılandırılmış ve yapılandırılmamış şeklinde sınıflandırılabilir. Hadoop, tüm veri çeşitliliğini işleme yeteneğine sahiptir, bununla birlikte çoğunlukla büyük miktarda yapılandırılmamış veriyi işlemek için kullanılır. Geleneksel RDBMS sadece

yapılandırılmış ve yarı yapılandırılmış verileri yönetmek için kullanılır. Dolayısıyla yapılandırılmamış verileri yönetmek için Hadoop tercih edilmektedir. (Hansen, 2015)

3.5.5. Gecikme / Yanıt Süresi

Hadoop, geleneksel RDBMS'e göre daha büyük veri kümelerine hızlıca erişebilmektedir. Fakat veri kümesinden belirli bir kayda erişmek istendiğinde Hadoop RDBMS sistemlerine göre daha yavaştır. Bunun da nedeni RDBMS veritabanlarının satır bazlı (kayıt bazlı) çalışma prensibinden kaynaklanmaktadır. Hadoop ise satır bazlı değil kütlesel veri (çoklu satır) prensibi doğrultusunda kurgulanmıştır, yeni geliştirmeler ile birlikte satır bazlı analizlerde de performansı artmaya başlamıştır. (Roy, 2017)

3.5.6. Ölçeklenebilirlik

RDBMS, bir makineyi "ölçekleme" olarak da bilinen dikey ölçeklenebilirlik sağlar. Bu, bilgisayar kümesindeki bir makineye bellek, CPU gibi daha fazla kaynak ve donanım ekleyebileceğiniz anlamına gelir. Oysa Hadoop, bir makinenin "ölçeklenmesi" olarak da bilinen yatay ölçeklenebilirlik sağlar. Bu durum, Hadoop'un hataya toleranslı hale gelmesi nedeniyle mevcut bilgisayar kümelerine daha fazla makine eklemek anlamına gelir. Tek bir başarısızlıktan dolayı sistemin çökme gibi bir durumu oluşmaz. Çünkü kümedeki daha fazla sunucunun varlığı nedeniyle, sunuculardan birinin arızalanmasına bakmaksızın veriler kolayca kurtarılabilmektedir. (Sanjay, 2014)

3.5.7. Veri İşleme

Hadoop, veri madenciliği tekniklerinde kullanılan OLAP (Çevrimiçi Analitik İşleme) yapısını desteklemektedir. OLAP çok karmaşık sorguları ve kümelenmeleri içerir. RDBMS' OLAP uygulandığında veri işleme hızı, veri miktarına ve veritabanı tasarımına bağlıdır, daha az tabloya sahip olunan veritabanlarında OLAP'ın çok karmaşık sorguları ve kümelenmeleri daha hızlı gerçekleşmektedir, veri miktarı ve tablo sayısı arttığında RDBMS'de OLAP sorguları ile veri işleme hızı yavaşlamaktadır. Esasen, RDBMS nispeten hızlı sorgulama işlemini içeren OLTP'yi (Çevrimiçi İşlem İşleme) desteklemektedir. Hadoop ise yapısı gereği büyük miktardaki veri ve çok sayıdaki tablo üzerinde OLAP sorgularını RDBMS'e göre daha hızlı sonuçlandırmaktadır. OLTP genellikle 3NF (bir varlık modeli) şemasını kullanır. OLAP'da tablo sayısı ve veri miktarı arttıkça sorgular ile veri işleme hızı düşmektedir. (Bista, 2018)

3.5.8. Maliyet

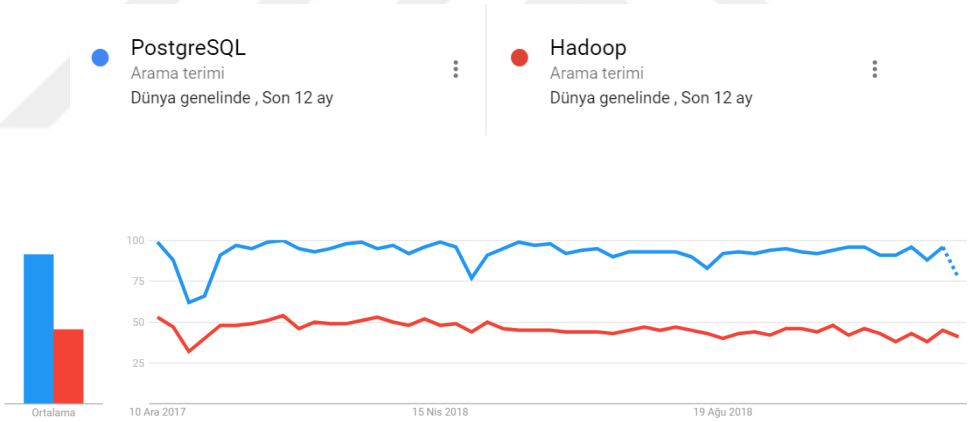
Hadoop ücretsiz ve açık kaynaklı bir yazılıp olup yazılımın lisansını almak için ödeme yapılmasına gerek yoktur. RDBMS'ler lisanslı bir yazılımdır ve ödeme yapılması gereken belli bedelleri vardır. (Waleed, 2018)

3.6. Hadoop ve PostgreSQL Karşılaştırma

Günümüzde Hadoop aşağıdaki üç problemi aynı anda çözmesi için çıkarılmış ve bu konuda da başarısını arttırarak devam etmektedir.

- Büyük veri problemleri.
- Giderek artan veri hacmini yönetmek.
- Artan veri hızını yönetmek ve daha fazla veri yapısıyla uğraşmak.

Dünya genelinde Google Trends kullanılarak yapılan Google Web Aramalarında iki sistemin son 12 ay içerisindeki arama dağılımı aşağıdaki gibi olup, zaman içerisinde Hadoop hızla yükseliş göstermektedir.



Şekil 3.3 Hadoop ve PostgreSQL Google Arama Geçmişi (Google Trend, 2018)

Genel olarak iki sistem birbirleri ile karşılaştırıldıklarında Hadoop her yeni güncelleme ile birlikte veritabanı yapısını biraz daha geliştirmektedir. (Apache Hadoop vs. PostgreSQL, 2017)

Tablo 3.2 Hadoop ve PostgreSQL'in Yapısal Olarak Karşılaştırılması

Özellik	Hadoop	PostgreSQL
Veritabanı Modeli	Max veritabanı büyüklüğü	Sınırsız
Lisans	Açık Kaynak	Açık Kaynak
Geliştirici	Apache Yazılım Vakfı	PostgreSQL Küresel Geliştirme Grubu
Uygulama Dili	Java	C
Sunucu İşletim Sistemi	Java VM Kullanan Tüm İşletim Sistemleri	Linux, Windows, FreeBSD, Solaris, Unix
SQL Dili	Var	Var
Veri Şeması	Var	Var
Erişim Motodları	JDBC, ODBC, Thrift	JDBC, ODBC, ADO.NET, native C library
Desteklenen Programlama Dilleri	C++, Java, PHP, Python	.Net, C, C++, Delphi, Java, Perl, PHP, Python,Tcl
Bellek İçi Yetenekler	Var	Yok
Sıralı İşlem	Var	Var
Kolon büyüklüğü	10 PB	1 GB
Map-Reduce	Var	Yok
Tam Metin Araması	Var	Var
Gerçek Zamanlı Analiz	Var	Var
Veri Tipi	Byte	BLOB

3.6.2. Hadoop Sisteminin Güçlü Yanları

Hadoop'un çıkışıyla birlikte, geçmişten gelen verilerin toplanması, saklanması ve analiz edilmesi karmaşıklığını da ortadan kaldırmaktadır. Hadoop sisteminin güçlü yanlarını sorgulandığında karşımıza çıkan sonuçlar, büyük verinin artık geçmişte olduğu gibi karmaşık bir konu olmaktan çıkarmaktadır. Hadoop, temelde dağıtılmış bir dosya sistemine dayanır. Veri işleme araçları genellikle verinin bulunduğu sunucularda bulunur ve bu da daha hızlı veri işlemeye olanak sağlamaktadır. Büyük hacimli yapılandırılmamış verilerde, Hadoop çok kısa sürelerde bu verileri işleyebilmektedir. Veriler farklı sunucularda yedekli bir şekilde tutulduğundan, herhangi bir şekilde meydana gelen sorunda hata toleransı sağlamaktadır. Yapının işletmelere düşük maliyetle birlikte, yapının esnek ve ölçeklenebilirliği sayesinde kolayca genişletilebilmektedir. (Ritika Prasad, 2017)

Şekil 3.4 Hadoop Sisteminin Güçlü Yanları

3.6.3. PostgreSQL Sisteminin Güçlü Yanları

PostgreSQL, şuanda Dünyanın en gelişmiş açık kaynaklı veritabanıdır. 1996 yılından itibaren, gönüllülerin çabalarıyla topluluk tabanlı olarak hiçbir kurum, kuruluş veya şikete bağlı olmadan geliştirilmektedir. Birçok veritabanı sisteminin aksine tamamen uyarlanabilir bir yapıdadır. Özel fonksiyonlar sayesinde farklı veri tiplerini veritabanını derlemeye gerek kalmadan kullanılmasına olanak sağlamaktadır. Ölçeklenebilirliği ve esnekliği sayesinde hemen hemen tüm sistemlerde kullanılılır ve aynı zamanda birçok programlama dili tarafından desteklenmektedir. Güvenlik anlamında da birçok başarılı sonuçlar elde etmektedir. Açık kaynaklı olsa bile yeni sürüm çıkarmadan önce ilk etapta beta sürüm testleri yapılmakta ve hatasız olarak PostgreSQL kullanan sistemlere sunulmaktadır.

Şekil 3.5 PostgreSQL Sisteminin Güçlü Yanları

IV. YÖNTEM

4.1. Tez Çalışmasının Bileşenleri

Bu tez çalışması kapsamında farklı sistemlerin birbirleriyle karşılaştırılması amacıyla aşağıdaki bileşenler kullanılmaktadır.

4.2. PostgreSQL

Bu çalışmada PostgreSQL kullanılmasının sebebi olarak, Postgre resmi sitesinde kullandıkları "The Wold's most advanced open source relational database" ifadesiyle dünyanın en gelişmiş açık kaynaklı veritabanı olduklarını iddia etmektedirler. Ayrıca çok geniş veri tiplerini desteklemektedir. Bu veri tiplerinin bazılarını MySQL, MariaDB ve Firebird veritabanları da desteklese bile hepsinin aynı anda desteklemesi Postgres'i öne çıkarmaktadır.

4.3. Hadoop Yapısı

Hadoop NameNode ve DataNode olmak üzere iki kısımdan oluşmaktadır.

4.3.1. NameNode (Ana Düğüm)

Master olarak da bilinmektedir. Ana Düğüm yalnızca dosya sistemindeki tüm dosyaların dizin ağacı olan HDFS'in meta verilerini depolar ve kümede izler. Gerçek veriyi veya veri setini kaydetmez. Verilerin kendisi aslında DataNode (Veri Düğümü) içinde saklanır. Ana Düğüm HDFS üzerindeki herhangi bir dosya için bloklarının listesini ve konumunu bilir. Bu bilgiler sayesinde Ana Düğüm dosyayı bloklardan nasıl

oluşturacağını bilir. Ana Düğüm HDFS için çok önemlidir. Ana Düğüm kapalıyken, HDFS ve Hadoop kümesine erişilemez ve kapalı sayılır. (Apache Wiki, 2011)

4.3.2. DataNode (Veri Düğümü)

Slave olarak da bilinmektedir. Gerçek verilerin HDFS'e kaydedilmesinden sorumludur. Ana Düğüm ile Veri Düğümü sürekli olarak iletişim halindedir. Bir Veri Düğümü başladığında, kendisinin sorumlu olduğu blokların listesini Ana Düğüme bildirir. Bir Veri Düğümü kapalı olduğunda verilerin veya kümenin kullanılabilirliğini etkilemez. Ana Düğüm, kullanılmayan İşçi Düğümü tarafından yönetilen bloklar için diğer Veri Düğümler üzerinde blokların çoğaltılması işlemini yapacaktır. Veri Düğümler genellikle çok fazla sabit disk alanı ile yapılandırılmıştır. Çünkü veriler burada saklanmaktadır. (Apache Wiki, 2011)

4.4. Sunucular Hakkında Genel Bilgiler

Karşılaştırılacak sunucuların özellik dağılımı donanımsal açıdan birbirine üstün gelmeyecek şekilde tasarlanmıştır. Tek düğümlü Hadoop yapısında 24 GB hafıza miktarı verilirken, çok düğümlü yapıda da kullanılacak toplam hafıza miktarı da 24 GB olarak ayarlanmıştır. Disk yapısında ise yine tekli sunucularda 100 GB olarak tasarlanmış çok düğümlü yapıda ise blok kopyalama değeri baz alındığından yine disk kapasitesi 100 GB olacaktır. EK-1'de Hadoop sisteminin kurulum aşamaları anlatılmıştır.

Tablo 4.1 Kullanılan Sunucuların Özellikleri

Sunucu Özellikleri				
Server İp	**CPU**	**RAM**	**HDD**	**Mod**
Tek Sunuculu Hadoop Sunucu Özellikleri				
192.168.5.100	2x Intel Xeon E5520 2.27 GHz	24GB	100 GB	Namenode + Datanode
3 Sunuculu Hadoop Sunucu Özellikleri				
192.168.5.110	2x Intel Xeon E5520 2.27 GHz	8GB	100 GB	Namenode
192.168.5.111	2x Intel Xeon E5520 2.27 GHz	8GB	100 GB	Datanode
192.168.5.112	2x Intel Xeon E5520 2.27 GHz	8GB	100 GB	Datanode
192.168.5.113	2x Intel Xeon E5520 2.27 GHz	8GB	100 GB	Datanode
PostgreSQL Sunucu Özellikleri				
192.168.5.200	2x Intel Xeon E5520 2.27 GHz	24GB	100 GB	Klasik Veritabanı

Çok düğümlü yapıda blok kopyalama değeri minimum 3 olduğundan kullanılan disk kapasiteside (100+100+100) / 3 şeklinde olacaktır.

4.5. TPC-H

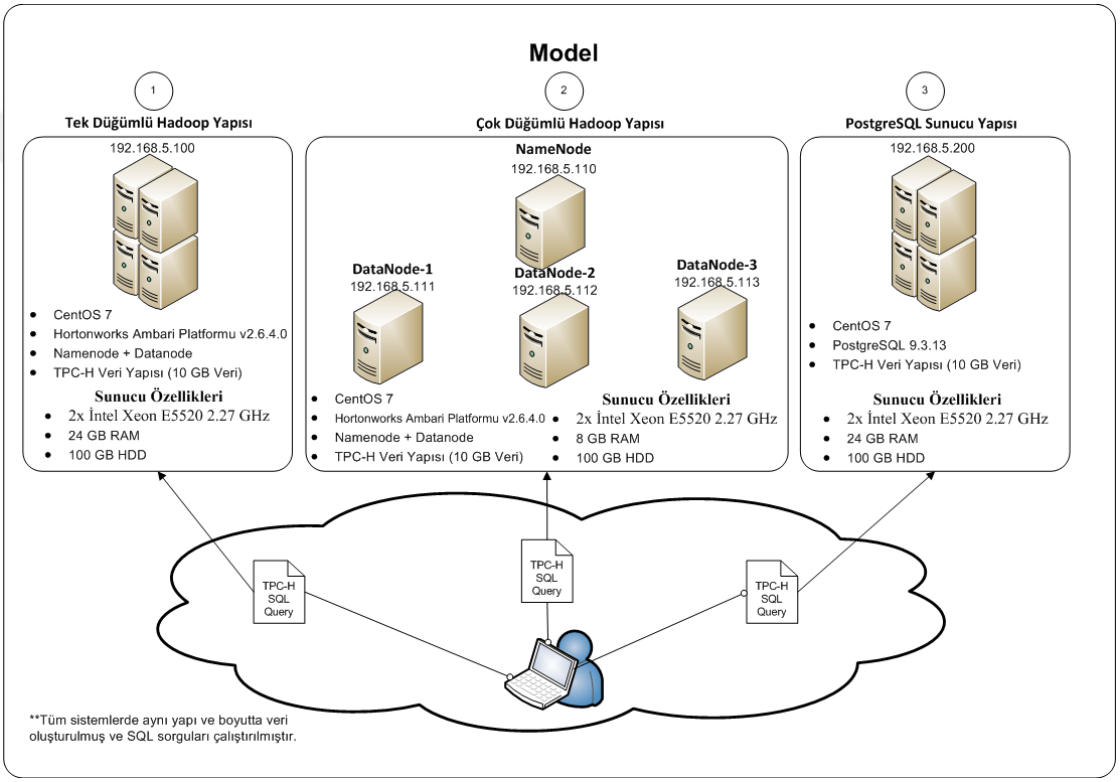
TPC-H bir karar destek testidir. Hemen hemen tüm veritabanı sistemlerinde çalışabilen, kendi datasını ve bu dataya özel sql sorguları üretip ilgili sistemlerin performansını ölçen dünya çapında kabul görmüş bir uygulamadır. TPC-H birçok firma tarafından (Oracle, Teradata, MySQL, MS-SQL vb.) veritabanı performans test aracı olarak kullanılmaktadır. EK-1'de TPC-H yazılımının sisteme kurulması anlatışmıştır. (TPC, 2018)

4.5.1. TPC-H Avantajları

- Ücretsizdir.
- Tüm veritabanı sistemleri ile uyumlu çalışmaktadır.
- Kurulduğu her sistemde aynı içerikli datayı üretmektedir.
- Üretmiş olduğu veri'ye uygun olarak SQL sorguları hazırlamaktadır.

4.6. Model

Yukarıdaki bileşenler doğrultusunda geleneksel veritabanları ile birlikte yatay ölçeklendirilen veritabanları arasında bir karşılaştırma yapısı oluşturmak amaçlı bir model oluşturulmuştur. Oluşturulan modelin katmanları aşağıdaki şekilde yaklaşık olarak görülmektedir.



Şekil 4.1 Oluşturulan Modelin Yapısı

1 ve 3 nolu yapıda aynı özelliklere sahip tek bir sunucu kullanıldı. Her iki sunucuda da TPC-H yapısı kullanılarak 10 GB büyüklüğünde veri seti oluşturuldu. Oluşturulan bu veri seti, her iki sunucuda da aynı tablo ve aynı içeriğe sahiptir. TPC-H, veriyi oluştururken aynı zamanda bu veriler üzerinde test amaçlı 20 adet SQL sorgusunu da sunucu üzerine yazmaktadır. Bu sorgular iki sistemde de çalıştırıldı ve sonuçları aşağıda belirtildiği şekildedir.

2 nolu yapıda ise Hadoop yapısı kuruldu. Toplam kaynak bakımından 1 ve 3 nolu sistemlere eşit olacak şekilde dizayn edilip TPC-H ile 10 GB büyüklüğündeki aynı veri seti bu sistem üzerine de entegre edildi.

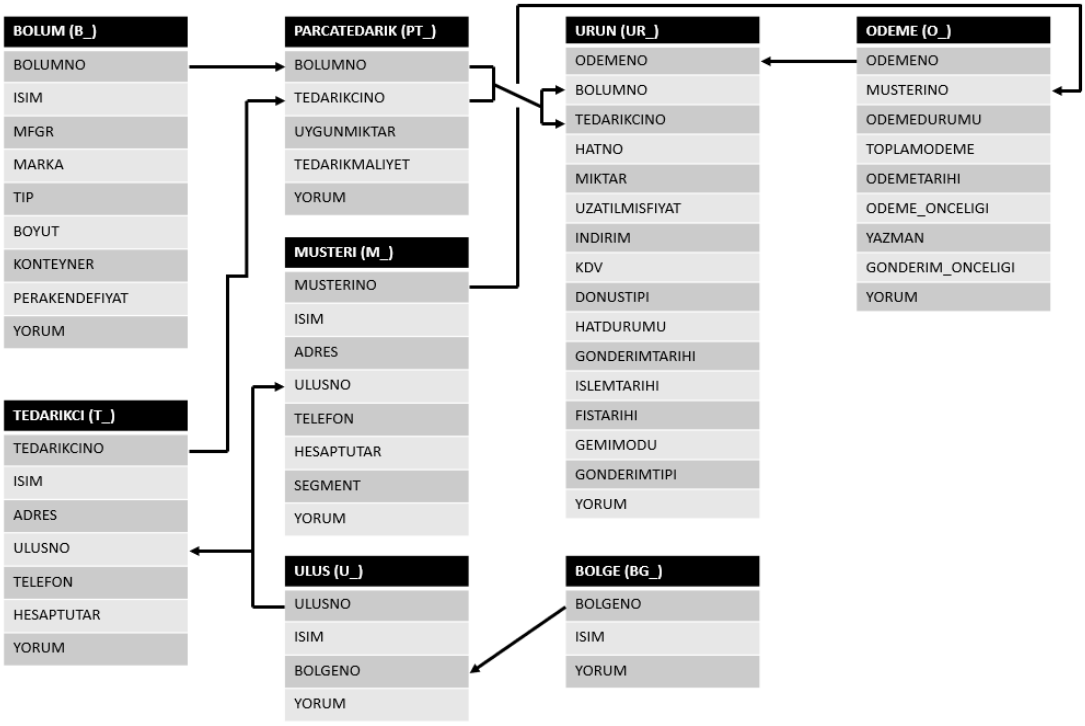
Tüm sistemlerde sırasıyla aynı SQL sorguları 5'er defa çalıştırıldı ve sonuçların ortalaması kabul edildi. Sistemlerde sorguların 5'er defa çalışmasında ki amaç SQL sorgusunun çalışma anında sistem tarafından çalıştırılan herhangi bir işin sonuca etki etmesini azaltmaktır.

V. BULGULAR VE SONUÇLAR

Oluşturulan modeller üzerinde TPC-H yazılımı ile oluşturulan SQL sorguları çalıştırılmak üzere sistemlerin sorgu performans süreleri kayıt altına alınmış ve yapılar birbirleriyle karşılaştırılmıştır.

5.1. Sorgu Yapısı ve İlişkisel Dağılımı

TPC-H yazılımı tarafından veriye uygun olarak oluşturulan sorgular, kurulan sistemler üzerinde beş defa çalıştırılıp ortalaması alındı. 1, 2, 3, 4 ve 5 ifadeleri aynı sorgunun birden fazla çalıştırıldığında elde edilen sorgu sürelerini ifade etmektedir. Veri büyüklüğü olarak 10 GB olarak şekilde tanımlandı ve veri seti detayı aşağıdaki şekilde oluşturuldu.



Şekil 5.1 Kullanılan Verinin Yapısal İlişkisi (TPC, TPC BENCHMARK, 2017)

5.1.1. Sorgu 1 – Fiyatlandırma Özet Raporu Sorgusu

- Sorgu Açıklaması;

Bu sorgu belirli bir tarihten itibaren, işletmelerin ürünleri için özet fiyatlandırma raporunun oluşmasını sağlamaktadır.

```
select
      u_donustipi,
      u_hatdurumu,
      sum(u_miktar) as toplam_miktar,
      sum(u_uzatilmisfiyat) as toplam_baz_fiyat,
      sum(u_uzatilmisfiyat * (1 - u_indirim)) as toplam_indirim_tutari,
      sum(u_uzatilmisfiyat * (1 - u_indirim) * (1 + u_kdv)) as
toplam_ucret,
      avg(u_miktar) as ortalama_miktar,
      avg(u_uzatilmisfiyat) as ortalama_fiyat,
      avg(u_indirim) as ortalama_indirim,
      count(*) as odeme_sayisi
from
      urun
where
      u_gonderimtarihi <= '1998-09-16'
group by
      u_donustipi,
      u_hatdurumu
order by
      u_donustipi DESC,
      u_hatdurumu;
```

Sorgu sonuç ortalama değerleri;

1.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.1 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 1	10,07 sn	7,29 sn	201,932 sn

5.1.2. Sorgu 2 – Minimum Maliyetli Tedarikçi Sorgusu

- Sorgu Açıklaması;

Bu sorgu belirli bir bölgedeki, belirli bir bölüm için hangi tedarikçinin sipariş vermesi gerektiğini bulmaktadır.

```
select
      t_hesaptutar,
      t_isim,
      u_isim,
      b_bolumno,
      b_mfgr,
      t_adres,
      t_telefon,
      t_yorum
from  bolum,
      tedarikci,
      parcatedarik,
      ulus,
      bolge,
      q2_miu_pt_arz_maliyet
where
      b_bolumno = pt_bolumno
      and t_tedarikcino = pt_tedarikcino
      and b_boyut = 37
      and b_tip like '%BAKIR'
      and t_ulusno = u_ulusno
      and u_bolgeno = bg_bolgeno
      and bg_isim = 'AVRUPA'
      and pt_tedarikmaliyet = miu_pt_tedarikmaliyet
      and b_bolumno = miu_b_bolumno
order by
      t_hesaptutar desc,
      u_isim,
      t_isim,
      b_bolumno;
```

Sorgu sonuç ortalama değerleri;

2.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.2 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 2	12,88 sn	11,19 sn	7,856 sn

5.1.3. Sorgu 3 – Nakliye Öncelik Kontrolü Sorgusu

- Sorgu Açıklaması;

Bu sorgu, en yüksek değere sahip gönderilemeyen siparişleri bulmaktadır.

```
select
      u_odemeno,
      sum(ur_uzatilmisfiyat * (1 - ur_indirim)) as gelir,
      o_odemetarihi,
      o_gonderim_onceligi
from
      musteri,
      odeme,
      urun
where
      m_segment = 'YAPI'
      and m_musterino = o_musterino
      and ur_odemeno = o_odemeno
      and o_odemetarihi < '1995-03-22'
      and ur_gonderimtarihi > '1995-03-22'
group by
      ur_odemeno,
      o_odemetarihi,
      o_gonder_onceligi
order by
      gelir desc,
      o_odemetarihi;
```

Sorgu sonuç ortalama değerleri;

3.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.3 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 3	14,94 sn	10,79 sn	28,934 sn

5.1.4. Sorgu 4 – Sipariş Öncelik Kontrolü Sorgusu

Sorgu Açıklaması;

Bu sorgu, sipariş öncelik sisteminin verimliliğini kontrol etmek için yapılmaktadır.

```
select
      o_odeme_onceligi,
      count(*) as odeme_sayisi
from
      odeme as o
where
      o_odemetarihi >= '1996-05-01'
      and o_odemetarihi < '1996-08-01'
      and exists (
            select * from urun
            where ur_odemeno = o.o_odemeno
                  and ur_islemtarihi < ur_fistarihi)
group by
      o_odeme_onceligi
order by
      o_odeme_onceligi DESC LIMIT 10;
```

Sorgu sonuç ortalama değerleri;

4.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.4 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 4	49,99 sn	92,09 sn	31,52 sn

5.1.5. Sorgu 5 – Yerel Tedarikçi Hacim Sorgusu

Sorgu Açıklaması;

Bu sorgu, yerel tedarikçiler aracılığıyla yapılan gelir hacmini listelemektedir.

```
select
      u_isim,
      sum(ur_uzatilmisfiyat * (1 - ur_indirim)) as gelir
from
      musteri,
      odeme,
      urun,
      tedarikci,
      ulus,
      bolge
where
      m_musterino = o_musterino
      and ur_odemeno = o_odemeno
      and ur_tedarikcino = t_tedarikcino
      and m_ulusno = t_ulusno
      and t_ulusno = u_ulusno
      and u_bolgeno = bg_bolgeno
      and bg_isim = 'AFRICA'
      and o_odemetarihi >= '1993-01-01'
      and o_odemetarihi < '1994-01-01'
group by
      u_isim
order by
      gelir desc
limit 10;
```

Sorgu sonuç ortalama değerleri;

5.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.5 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 5	24,84 sn	13,67 sn	48,21 sn

5.1.6. Sorgu 6 – Gelir Değişim Öngörü Sorgusu

Sorgu Açıklaması;

Bu sorgu, belirli bir tarih aralığındaki indirimlerin gelir artışına etkisini ölçmek amacıyla kullanılmaktadır.

```
select
      sum(ur_uzatilmisfiyat * ur_indirim) as gelir
from
      urun
where
      ur_gonderimtarihi >= '1993-01-01'
      and ur_gonderimtarihi < '1994-01-01'
      and ur_indirim between 0.06 - 0.01 and 0.06 + 0.01
      and ur_miktar < 25;
```

Sorgu sonuç ortalama değerleri;

6.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.6 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 6	3,06 sn	2,82 sn	14,776 sn

5.1.7. Sorgu 7 – Toplu Gönderi Gönderme Sorgusu

Sorgu Açıklaması;

Bu sorgu, bazı ülkeler arasında sevk edilen malların, sevkiyatın yeniden müzakere edilmesine yardımcı olmak amacıyla kullanılmaktadır.

```
select tedarikci_ulus,
      musteri_ulus,
      ur_yil,
      sum(hacim) as gelir
from  (select
                  n1.u_isim as tedarikci_ulus,
                  n2.u_isim as musteri_ulus,
                  year(ur_gonderimtarihi) as ur_yil,
                  ur_extendedprice * (1 - ur_indirim) as hacim
            from
                  tedarikci,
                  urun,
                  odeme,
                  musteri,
                  ulus n1,
                  ulus n2
            where t_tedarikcino = ur_tedarikcino
                  and o_orderkey = ur_orderkey
                  and m_custkey = o_custkey
                  and t_nationkey = n1.u_nationkey
                  and m_nationkey = n2.u_nationkey
                  and ((n1.u_isim = 'KENYA' and n2.u_isim = 'PERU')
                  or (n1.u_isim = 'PERU' and n2.u_isim = 'KENYA'))
                  and  ur_gonderimtarihi  between  '1995-01-01'  and
'1996-12-31') as gonderim
group by
      tedarikci_ulus,
      musteri_ulus,
      ur_yil
order by tedarikci_ulus DESC, musteri_ulus, ur_yil;
```

Sorgu sonuç ortalama değerleri;

7.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.7 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 7	25,80 sn	41,74 sn	27,376 sn

5.1.8. Sorgu 8 – Ulusal Pazar Payı Sorgusu

Sorgu Açıklaması;

Bu sorgu, belirli bir bölgedeki belli bir ulusun pazar payının iki yılda nasıl değiştiğini göstermektedir.

```
select
      o_yil,
      sum(case when ulus = 'PERU' then hacim else 0 end) / sum(hacim)
as mkt_paylasim
from  (     select
            year(o_odemetarihi) as o_yil,
            ur_uzatilmisfiyat * (1 - ur_indirim) as hacim,
            n2.n_isim as ulus from
            bolum,
            tedarikci,
            urun,
            odeme,
            musteri,
            ulus n1,
            ulus n2,
            bolge
            where b_bolumno = ur_bolumno
            and t_tedarikcino = ur_tedarikcino
            and ur_odemeno = o_odemeno
            and o_musterino = m_musterino
            and m_ulusno = n1.u_ulusno
            and n1.u_bolgeno = bg_bolgeno
            and bg_isim = 'AMERICA'
            and t_ulusno = n2.u_ulusno
            and o_odemetarihi between '1995-01-01' and '1996-12-31'
            and b_tip = 'EKONOMI YANIKLI NIKEL' ) as tum_uluslar
group by o_yil
order by o_yil DESC;
```

Sorgu sonuç ortalama değerleri;

8.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.8 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 8	14,75 sn	11,46 sn	25,264 sn

5.1.9. Sorgu 9 – Ürün Türü Kâr Ölçüm Sorgusu

Sorgu Açıklaması;

Bu sorgu, belirli bir parça hattında tedarikçinin ülkesine ve yılına göre ne kadar kâr elde ettiğini göstermektedir.

```
select
      ulus,
      o_yil,
      sum(tutar) as toplam_kar
from
      (           select
                  u_isim as ulus,
                  year(o_odemetarihi) as o_yil,
                  ur_uzatilmisfiyat    *    (1    -    ur_indirim)    -
pt_tedarikmaliyet * ur_miktar as tutar
            from
                  bolum,
                  tedarikci,
                  urun,
                  parcatedarik,
                  odeme,
                  ulus
            where
                  t_tedarikcino = ur_tedarikcino
                  and pt_tedarikcino = ur_tedarikcino
                  and pt_bolumno = ur_bolumno
                  and b_bolumno = ur_bolumno
                  and o_odemeno = ur_odemeno
                  and t_ulusno = u_ulusno
                  and b_isim like '%plum%' ) as kar
group by ulus, o_yil
order by ulus, o_yil desc;
```

Sorgu sonuç ortalama değerleri;

9.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.9 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 9	36,25 sn	33,36 sn	169,6 sn

5.1.10. Sorgu 10 – İade edilen Ürün Rapor Sorgusu

Sorgu Açıklaması;

Bu sorgu, kendisine gönderilen ürünler ile ilgili sorun yaşayan müşterileri belirlemek için kullanılmaktadır.

```
select
      m_musterino,
      m_isim,
      sum(ur_uzatilmisfiyat * (1 - ur_indirim)) as gelir,
      m_hesaptutar,
      u_isim,
      m_adres,
      m_telefon,
      m_yorum
from
      musteri,
      odeme,
      urun,
      ulus
where
      m_musterino = o_musterino
      and ur_odemeno = o_odemeno
      and o_odemetarihi >= '1993-07-01'
      and o_odemetarihi < '1993-10-01'
      and ur_donustipi = 'R'
      and m_ulusno = u_ulusno
group by m_musterino, m_isim, m_hesaptutar, m_telefon, u_isim, m_adres,
m_yorum
order by gelir desc;
```

Sorgu sonuç ortalama değerleri;

10.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.10 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 10	25,60 sn	20,8 sn	41,56 sn

5.1.11. Sorgu 11 – Önemli Stok Tanımlama Sorgusu

Sorgu Açıklaması;

Bu sorgu, belirli bir ülkedeki tedarikçilerin mevcut stoklarını ile kullanılabilir toplam stokları arasındaki değeri hesaplamak için kullanılmaktadır.

```
drop view q11_bolum_tmp_onbellek
drop view q11_toplam_tmp_onbellek

create view q11_bolum_tmp_onbellek as
select
      pt_bolumno,
      sum(pt_tedarikmaliyet * pt_uygunmiktar) as bolum_hacmi
from
      parcatedarik,
      tedarikci,
      ulus
where
      pt_tedarikcino= t_tedarikcino
      and t_ulusno = y_ulusno
      and u_isim = 'ALMANYA'
group by pt_bolumno;

create view q11_toplam_tmp_onbellek as
select
      sum(bolum_hacmi) as toplam_hacim
from q11_part_tmp_cached;

select pt_bolumno, toplam_hacim as hacim
from (select
            pt_bolumno,
            bolum_hacmi,
            toplam_hacim
      from  q11_bolum_tmp_onbellek join q11_toplam_tmp_onbellek) a
where bolum_hacmi > toplam_hacim * 0.0001
order by hacim desc;
```

Sorgu sonuç ortalama değerleri;

11.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.11 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 11	7,84 sn	4,68 sn	5,036 sn

5.1.12. Sorgu 12 – Nakliye Modları ve Öncelikli Sipariş Sorgusu

Sorgu Açıklaması;

Bu sorgu, daha ucuz nakliye yöntemlerinin seçilmesiyle birlikte, taahhüt edilen tarihten sonra müşteri tarafından daha fazla parçanın alınmasına neden olup olmayacağını belirlemektedir.

```
select
      ur_gonderimtipi,
      sum(case
            when o_odeme_onceligi = '1-ACIL'
                  or o_odeme_onceligi = '2-YUKSEK'
                  then 1
            else 0
      end) as yuksek_oncelik_sayisi,
      sum(case
            when o_odeme_onceligi <> '1-ACIL'
                  and o_odeme_onceligi <> '2-YUKSEK'
                  then 1
            else 0
      end) as dusuk_oncelik_sayisi
from
      odeme,
      urun
where
      o_odemeno = ur_odemeno
      and ur_gonderimtipi in ('HAVA YOLU', 'POSTA')
      and ur_islemtarihi < ur_fistarihi
      and ur_gonderimtarihi < ur_islemtarihi
      and ur_fistarihi >= '1995-01-01'
      and ur_fistarihi < '1996-01-01'
group by ur_gonderimtipi
order by ur_gonderimtipi DESC;
```

Sorgu sonuç ortalama değerleri;

12.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.12 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 12	6,50 sn	6,06 sn	20,828 sn

5.1.13. Sorgu 13 – Müşteri Dağıtım Sorgusu

Sorgu Açıklaması;

Bu sorgu, müşteriler ile siparişlerinin büyüklüğü arasındaki ilişkiyi bulmak için kullanılmaktadır.

```
select
      m_sayi,
      count(*) as farklimusteri
from
      (select
            m_musterino,
            count(o_odemeno) as m_sayi
            from
            musteri left outer join odeme on
            m_musterino = o_musterino
            and o_yorum not like '%olagandisi%hesap%'
            group by m_musterino ) m_odeme
group by m_sayi
order by farklimusteri desc, m_sayi desc;
```

Sorgu sonuç ortalama değerleri;

13.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.13 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 13	34,73 sn	25,01 sn	30,534 sn

5.1.14. Sorgu 14 – Promosyon Etki Sorgusu

Sorgu Açıklaması;

Bu sorgu, TV reklamları veya özel bir kampanyanın tanıtılması ile birlikte, bu tanıtımların geri dönüşlerini izlemek için kullanılmaktadır.

```
select
      100.00 * sum(case
      when b_tip like 'KAMPANYA%'
      then ur_uzatilmisfiyat * (1 - ur_indirim)
      else 0
      end)   /   sum(ur_uzatilmisfiyat   *   (1   -   ur_indirim))   as
kampanya_geliri
from
      urun,
      bolum
where
      ur_bolumno = b_bolumno
      and ur_gonderimtarihi >= '1995-08-01'
      and ur_gonderimtarihi < '1995-09-01';
```

Sorgu sonuç ortalama değerleri;

14.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.14 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 14	5,41 sn	9,89 sn	13,716 sn

5.1.15. Sorgu 15 – En İyi Tedarikçi Sorgusu

Sorgu Açıklaması;

Bu sorgu, en iyi tedarikçiyi belirlemek ve ödüllendirebilmek için kullanılmaktadır.

```
drop view gelir_onbellek;
drop view max_gelir_onbellek;

create view gelir_onbellek as
select
      ur_tedarikcino as tedarikci_no,
      sum(ur_uzatilmisfiyat * (1 - ur_indirim)) as toplam_gelir
from
      urun
where
      ur_gonderimtarihi >= '1996-01-01'
      and ur_gonderimtarihi < '1996-04-01'
group by ur_tedarikcino;

create view max_gelir_onbellek as
select
      max(toplam_gelir ) as max_gelir
from
      gelir_onbellek;

select t_tedarikcino,
       t_isim,
       t_adres,
       t_telefon,
       toplam_gelir
from  tedarikci,
      gelir_onbellek,
      max_gelir_onbellek
where t_tedarikcino = tedarikci_no
      and toplam_gelir = max_gelir
order by t_tedarikcino DESC;
```

Sorgu sonuç ortalama değerleri;

15.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.15 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 15	11,91 sn	12,06 sn	29,118 sn

5.1.16. Sorgu 16 – Parça – Tedarikçi İlişkisi Sorgusu

Sorgu Açıklaması;

Bu sorgu, verilen siparişlerin kaç tedarikçi tarafından tedarik edilebileceğini göstermekle birlikte sipariş edilen parçalar için yeterli sayıda tedarikçinin bulunup bulunmadığını belirlemek için kullanılmaktadır.

```
select
      b_marka,
      b_tip,
      b_boyut,
      count(distinct pt_tedarikcino) as tedarikci_sayisi
from  parcatedarik,
      bolum
where b_bolumno = pt_bolumno
      and b_marka <> 'Marka#34'
      and b_tip not like 'EKONOMI FIRCA%'
      and b_boyut in (22, 14, 27, 49, 21, 33, 35, 28)
      and parcatedarik.pt_tedarikcino not in (
            select
                  t_tedarikcino
            from
                  tedarikci
            where
                  t_yorum like '%Musteri%Şikayetler%')
group by
      b_marka,
      b_tip,
      b_boyut
order by
      tedarikci_sayisi desc,
      b_marka,
      b_tip,
      b_boyut;
```

Sorgu sonuç ortalama değerleri;

16.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.16 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 16	8,41 sn	11,63 sn	58,04 sn

5.1.17. Sorgu 17 – Küçük Miktarlı Sipariş Geliri Sorgusu

Sorgu Açıklaması;

Bu sorgu, satışları daha büyük sevkiyatlara yoğunlaştırarak genel kalem giderlerini azaltmak için kullanılmaktadır.

```
with q17_bolum as (
  select b_bolumno from bolum where
  b_marka= 'Marka#23'
  and b_konteyner = 'MEDIKAL KUTU'),
q17_avg as (
  select ur_bolumno as z_bolumno, 0.2 * avg(ur_miktar) as
z_ortalama_miktar
  from urun
  where ur_bolumno IN (select b_bolumno from q17_bolum)
  group by ur_bolumno
),
q17_fiyat as (
  select
  ur_miktar,
  ur_bolumno,
  ur_uzatilmisfiyat
  from
  urun
  where
  ur_bolumno IN (select b_bolumno from q17_bolum)
)
select cast(sum(ur_uzatilmisfiyat) / 7.0 as decimal(32,2)) as
ortalama_yillik
from q17_avg, q17_fiyat
where
z_bolumno, = ur_bolumno and ur_miktar < z_ortalama_miktar;
```

Sorgu sonuç ortalama değerleri;

17.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.17 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 17	13,23 sn	9,52 sn	90,064 sn

5.1.18. Sorgu 18 – Sipariş Emirleri Bekleyen Tedarikçiler Sorgusu

Sorgu Açıklaması;

Bu sorgu, gerekli parçaları zamanında sevk edememiş tedarikçileri tanımlamak için kullanılmaktadır.

```
create temporary table l3 stored as orc as
select ur_odemeno,count(distinct ur_tedarikcino) as tedarikci_sayi
from urun
where ur_fistarihi > ur_islemtarihi and ur_odemeno is not null
group by ur_odemeno
having tedarikci_sayi = 1;

with lokasyon as (
select tedarikci.* from tedarikci, ulus where
t_ulusno = u_ulusno and u_isim = 'SUUDI ARABISTAN')

select t_isim, count(*) as bekleme
from
(
select li.ur_tedarikcino, li.ur_odemeno
from urun li join odeme o on li.ur_odemeno = o.o_odemeno and
                    o.o_odemedurumu = 'F'
     join
     (select    ur_odemeno,    count(distinct    ur_tedarikcino)    as
tedarikci_sayi
     from urun
     group by ur_odemeno
     ) l2 on li.ur_odemeno = l2.ur_odemeno and
             li.ur_fistarihi > li.ur_islemtarihi and
             l2.tedarikci_sayi > 1) l1 join l3
on l1.ur_odemeno = l3.odemeno
 join lokasyon s on l1.ur_tedarikcino = s.t_tedarikcino
group by t_isim
order by bekleme desc, t_isim;
```

Sorgu sonuç ortalama değerleri;

18.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.18 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 18	208,38 sn	115,09 sn	73,536 sn

5.1.19. Sorgu 19 – Küresel Satış Sorgulama Sorgusu

Sorgu Açıklaması;

Bu sorgu, satın alma olasılığı yüksek olan müşterilerin bulunduğu coğrafi bölgeleri tanımlamak için kullanılmaktadır.

```
drop view q22_musteri_tmp_onbellek;
drop view q22_musteri_tmp1_onbellek;
drop view q22_odeme_tmp_onbellek;

create view if not exists q22_musteri_tmp_onbellek as
select
      m_hesaptutar,
      n_musterino,
      substr(m_telefon, 1, 2) as ulkekodu
from
      musteri
where
      substr(m_telefon, 1, 2) = '13' or
      substr(m_telefon, 1, 2) = '31' or
      substr(m_telefon, 1, 2) = '23' or
      substr(m_telefon, 1, 2) = '29' or
      substr(m_telefon, 1, 2) = '30' or
      substr(m_telefon, 1, 2) = '18' or
      substr(m_telefon, 1, 2) = '17';

create view if not exists q22_musteri_tmp1_onbellek as
select
      avg(m_hesaptutar) as ortalama_hesaptutar
from
      q22_musteri_tmp_onbellek
where
      m_hesaptutar > 0.00;

create view if not exists q22_odeme_tmp_onbellek as
select
      o_musterino
from
      odeme
group by
      o_musterino;

select
      ulkekodu,
      count(1) as nummust,
      sum(m_hesaptutar) as toplamhesaptutar
from (
      select
            ulkekodu,,
            m_hesaptutar,
            ortalama_hesaptutar
      from
            q22_musteri_tmp1_onbellek ct1 join (
```

```
                  select
                        ulkekodu,
                        m_hesaptutar
                  from
                        q22_odeme_tmp_onbellek ot
                        right outer join q22_musteri_tmp_onbellek ct
                        on ct.m_musterino = ot.o_musterino
                  where
                        o_musterino is null
            ) ct2
) a
where
      m_hesaptutar > ortalama_hesaptutar
group by
      ulkekodu
order by
      ulkekodu DESC;
```

Sorgu sonuç ortalama değerleri;

19.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

Tablo 5.19 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 19	20,46 sn	23,32 sn	9,478 sn

5.1.20. Sorgu 20 – Potansiyel Parça Tanıtım Sorgusu

Sorgu Açıklaması;

Bu sorgu, belirli bir ülkedeki tedarikçilerin promosyon teklifi için değişiklik gösterebilecek seçilmiş parçaları göstermektedir.

```
with tmp1 as (
    select b_bolumno from bolum where b_isim like 'orman%'),
tmp2 as (select t_isim,t_adres, t_tedarikcino from tedarikci, ulus
    where t_ulusno = u_ulusno and u_isim = 'KANADA'),
tmp3 as (
    select ur_bolumno, 0.5 * sum(ur_miktar) as toplam_miktar,
ur_tedarikcino
    from urun, tmp2
    where ur_gonderimtarihi >= '1994-01-01' and ur_gonderimtarihi <=
'1995-01-01'
    and ur_tedarikcino = t_tedarikcino
    group by ur_bolumno, ur_tedarikcino),
tmp4 as (
    select pt_bolumno, pt_tedarikcino, pt_uygunmiktar
    from parcatedarik
    where pt_bolumno IN (select b_bolumno from tmp1)),
tmp5 as (
select pt_tedarikcino from tmp4, tmp3
where
    pt_bolumno = ur_bolumno
    and pt_tedarikcino = ur_tedarikcino
    and pt_uygunmiktar > toplam_miktar)
select t_isim,
    t_adres
from tedarikci
where t_tedarikcino IN (select pt_tedarikcino from tmp5)
order by t_isim DESC;
```

Sorgu sonuç ortalama değerleri;

20.sorgu tüm sistemlerde 5 kere çalıştırıldı, her çalıştırılma sonrasındaki sorgu sürelerinin ortalaması alındı.

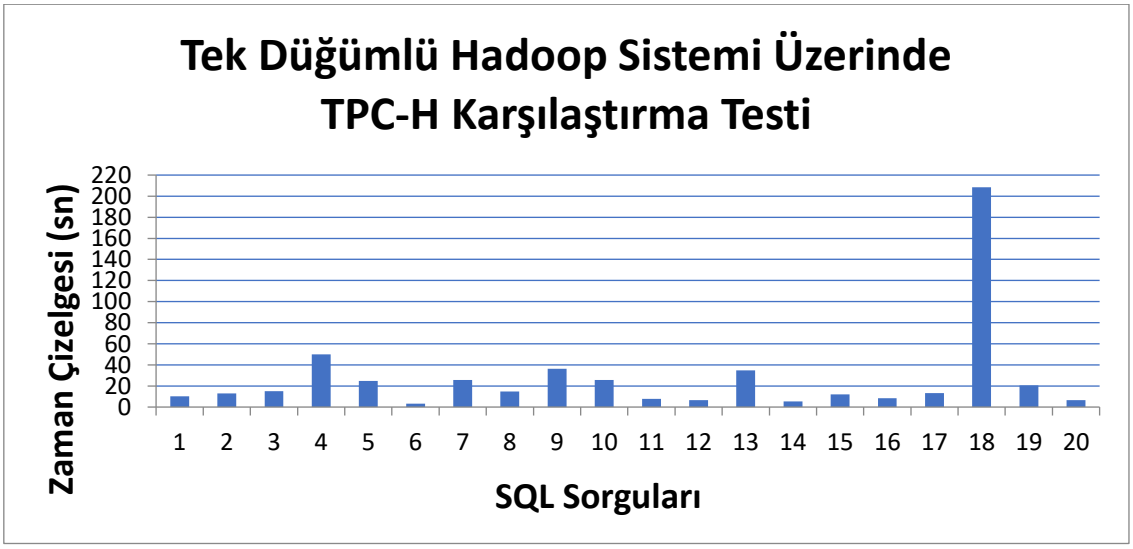
Tablo 5.20 Çalıştırılan Sorgunun Sonucu

Sorgu Adımları	Tek Düğümlü Hadoop	Çok Düğümlü Hadoop	PostgreSQL
Sorgu 20	6,48 sn	7,28 sn	55,196 sn

5.2. Tek Düğümlü Hadoop Sistemi SQL Sorgu Sonuç Dağılımı

Tablo 5.21 Tek Düğümlü Hadoop Sisteminin TPC-H Karşılaştırma Test Sonuçları

Tek Düğümlü Hadoop Sistemi Üzerinde TPC-H Karşılaştırma Testi						
Sorgu Adımları	**Akış 1**	**Akış 2**	**Akış 3**	**Akış 4**	**Akış 5**	**Ortalama**
Sorgu 1	10,63	9,76	10,73	9,42	9,81	10,07
Sorgu 2	11,68	12,03	12,25	13,87	14,56	12,88
Sorgu 3	16,58	14,38	15,96	13,47	14,33	14,94
Sorgu 4	49,51	52,99	48,91	47,59	50,96	49,99
Sorgu 5	26,62	24,02	23,15	24,76	25,64	24,84
Sorgu 6	2,81	2,91	3,52	2,85	3,23	3,06
Sorgu 7	27,77	25,29	25,34	24,64	25,97	25,80
Sorgu 8	17,79	14,1	14	13,79	14,06	14,75
Sorgu 9	35,59	35,92	39,03	36,29	34,44	36,25
Sorgu 10	26,07	32,35	20,59	25,54	23,44	25,60
Sorgu 11	7,94	7,22	8,19	7,76	8,08	7,84
Sorgu 12	8,57	6,29	5,82	5,95	5,86	6,50
Sorgu 13	35,6	34,8	33,78	34,64	34,82	34,73
Sorgu 14	5,86	5,17	5,05	5,45	5,54	5,41
Sorgu 15	12,53	11,49	12,14	11,75	11,64	11,91
Sorgu 16	10,4	7,98	7,49	8,41	7,76	8,41
Sorgu 17	14,12	12,77	13,41	13,11	12,75	13,23
Sorgu 18	213,88	200,13	208,56	205,13	214,22	208,38
Sorgu 19	21,08	20,02	19,56	22,01	19,65	20,46
Sorgu 20	7,38	5,88	6,32	6,27	6,54	6,48
Toplam Geçen Süre :						541,54

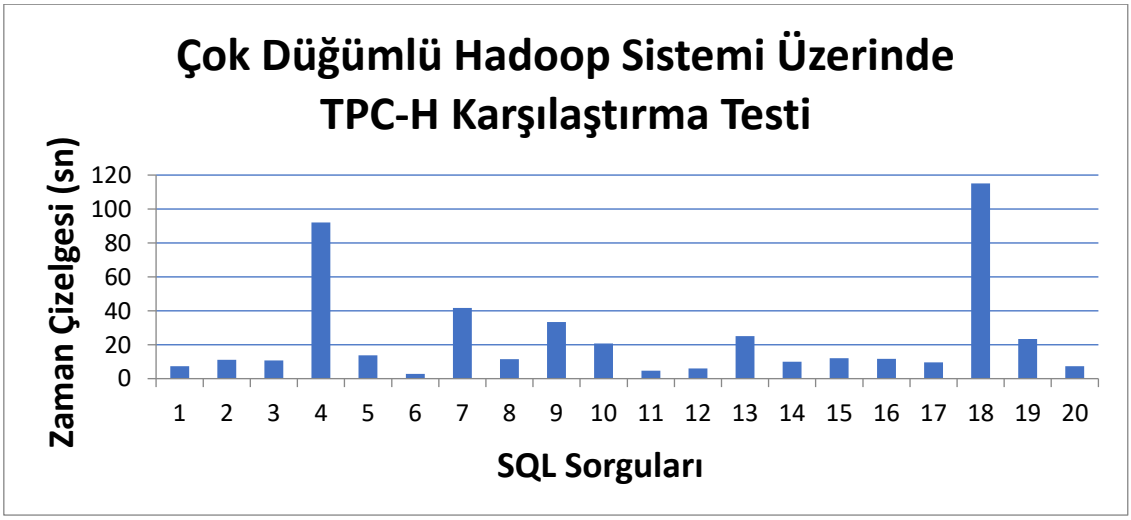


Şekil 5.2 Tek Düğümlü Hadoop Sisteminin Grafiksel Olarak Karşılaştırma Sonuçları

5.3. Çok Düğümlü Hadoop Sistemi SQL Sorgu Sonuç Dağılımı

Tablo 5.22 Çok Düğümlü Hadoop Sisteminin TPC-H Karşılaştırma Test Sonuçları

Çok Düğümlü Hadoop Sistemi Üzerinde TPC-H Karşılaştırma Testi						
Sorgu Adımları	**Akış 1**	**Akış 2**	**Akış 3**	**Akış 4**	**Akış 5**	**Ortalama**
Sorgu 1	7,56	7,15	7,49	6,8	7,43	7,29
Sorgu 2	10,98	11,03	11,43	10,97	11,54	11,19
Sorgu 3	11,05	11,37	10,43	10,75	10,36	10,79
Sorgu 4	94,31	90,45	92,13	92,68	90,89	92,09
Sorgu 5	15,16	12,12	13,56	13,82	13,68	13,67
Sorgu 6	2,96	2,92	2,76	2,81	2,67	2,82
Sorgu 7	43,63	40,67	41,58	41,95	40,87	41,74
Sorgu 8	11,44	11,53	11,68	12,25	10,39	11,46
Sorgu 9	34,19	33,64	32,56	33,43	32,97	33,36
Sorgu 10	19,73	20,43	21,69	22,15	20,01	20,80
Sorgu 11	4,59	5,11	4,28	5,43	4	4,68
Sorgu 12	5,84	5,67	6,64	6,17	5,99	6,06
Sorgu 13	24,44	26,04	25,74	24,16	24,68	25,01
Sorgu 14	10,35	9,72	9,77	9,89	9,72	9,89
Sorgu 15	11,28	11,57	12,82	13,48	11,14	12,06
Sorgu 16	11,89	11,33	11,1	12,18	11,65	11,63
Sorgu 17	11,24	12,24	7,42	9,58	7,13	9,52
Sorgu 18	113,84	116,76	120,5	114,14	110,21	115,09
Sorgu 19	37,57	18,72	19,12	21,56	19,64	23,32
Sorgu 20	8,3	6,12	7,67	7,45	6,87	7,28
Toplam Geçen Süre :						469,76

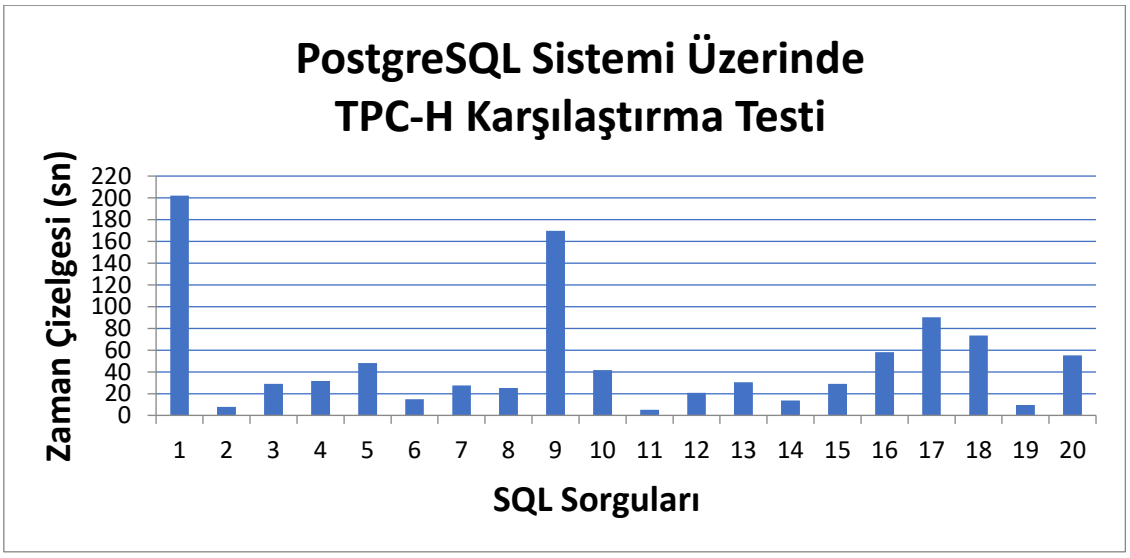


Şekil 5.3 Çok Düğümlü Hadoop Sisteminin Grafiksel Olarak Karşılaştırma Sonuçları

5.4. PostgreSQL Sistemi SQL Sorgu Sonuç Dağılımı

Tablo 5.23 PostgreSQL Sisteminin TPC-H Karşılaştırma Test Sonuçları

PostgreSQL Sistemi Üzerinde TPC-H Karşılaştırma Testi						
Sorgu Adımları	**Akış 1**	**Akış 2**	**Akış 3**	**Akış 4**	**Akış 5**	**Ortalama**
Sorgu 1	202,69	198,81	204,13	202,84	201,19	201,93
Sorgu 2	7,9	8,01	7,84	7,62	7,91	7,86
Sorgu 3	29,51	29,06	28,67	28,45	28,98	28,93
Sorgu 4	30,03	32,56	31,98	31,75	31,28	31,52
Sorgu 5	48,32	48,18	47,84	48,42	48,29	48,21
Sorgu 6	14,12	15,54	14,47	15,08	14,67	14,78
Sorgu 7	27,41	27,11	27,92	27,53	26,91	27,38
Sorgu 8	25,2	25,54	25,72	24,95	24,91	25,26
Sorgu 9	169,97	170,53	168,56	169,65	169,29	169,60
Sorgu 10	41,72	41,8	40,96	41,57	41,79	41,57
Sorgu 11	4,67	4,72	5,18	5,72	4,89	5,04
Sorgu 12	20,55	21,16	20,67	20,94	20,82	20,83
Sorgu 13	30,46	30,36	31,02	30,45	30,38	30,53
Sorgu 14	13,58	13,67	14,02	13,86	13,45	13,72
Sorgu 15	29,56	28,2	28,9	29,68	29,25	29,12
Sorgu 16	58,16	57,19	58,25	58,71	57,89	58,04
Sorgu 17	90,46	89,56	90,72	89,67	89,91	90,06
Sorgu 18	74,93	72,16	72,79	73,62	74,18	73,54
Sorgu 19	9,14	9,54	9,67	9,23	9,81	9,48
Sorgu 20	55,27	55,23	54,9	55,34	55,24	55,20
Toplam Geçen Süre :						982,58



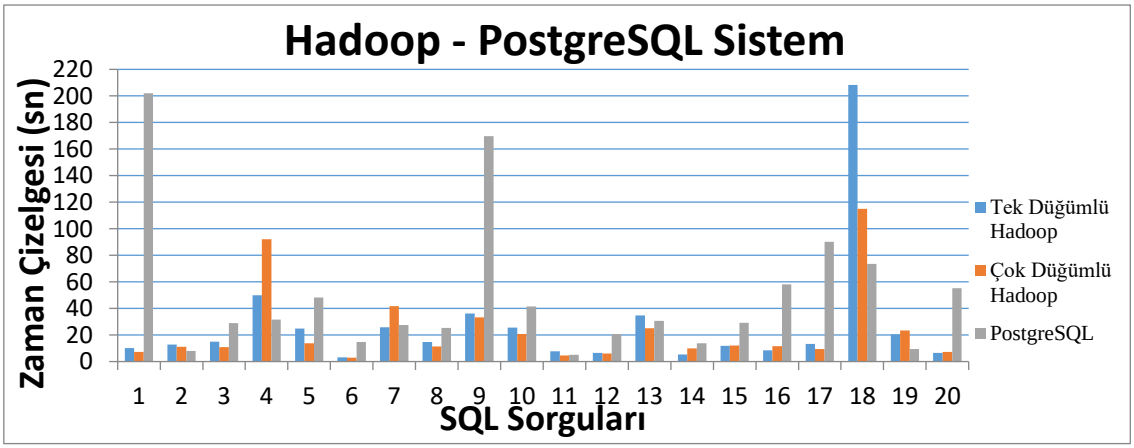
Şekil 5.4 PostgreSQL Sisteminin Grafiksel Olarak Karşılaştırma Sonuçları

5.5. Hadoop (Tek Düğümlü – Çok Düğümlü) ve PostgreSQL Sistemlerinin Karşılaştırmalı (Benchmark) TPC-H Sonuçları

Oluşturulan sistemlerin ortalama performans süreleri aşağıdaki tabloda gözükmektedir.

Tablo 5.24 Hadoop ve PostgreSQL Sistemlerin TPC-H Karşılaştırmalı Test Sonuçları

Hadoop ve PostgreSQL Sistemlerinin TPC-H Karşılaştırma Test Sonuçları			
Sorgu Adımları	**Tek Düğümlü Hadoop**	**Çok Düğümlü Hadoop**	**PostgreSQL**
Sorgu 1	10,07	7,29	201,932
Sorgu 2	12,88	11,19	7,856
Sorgu 3	14,94	10,79	28,934
Sorgu 4	49,99	92,09	31,52
Sorgu 5	24,84	13,67	48,21
Sorgu 6	3,06	2,82	14,776
Sorgu 7	25,80	41,74	27,376
Sorgu 8	14,75	11,46	25,264
Sorgu 9	36,25	33,36	169,6
Sorgu 10	25,60	20,8	41,568
Sorgu 11	7,84	4,68	5,036
Sorgu 12	6,50	6,06	20,828
Sorgu 13	34,73	25,01	30,534
Sorgu 14	5,41	9,89	13,716
Sorgu 15	11,91	12,06	29,118
Sorgu 16	8,41	11,63	58,04
Sorgu 17	13,23	9,52	90,064
Sorgu 18	208,38	115,09	73,536
Sorgu 19	20,46	23,32	9,478
Sorgu 20	6,48	7,28	55,196
Toplam Geçen Süre :	541,53	469,75	982,582



Şekil 5.5 Hadoop ve PostgreSQL Sistemlerin Grafiksel Karşılaştırma Sonuçları

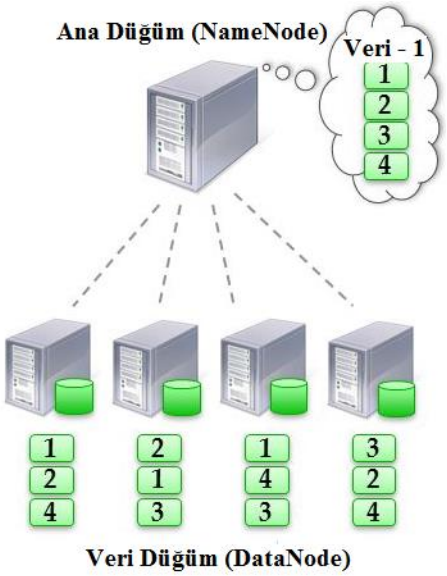
VI. DEĞERLENDİRME

Geleneksel veritabanı olarak nitelendirilen veritabanları mimari olarak verinin bir noktada toplanması üzerine oluşturulmuştur, bazı durumlarda yedekleme amaçlı kümeleme sistemleri ile yapısal olarak desteklenmektedirler. PostgreSQL açık kaynak veritabanı sistemi geleneksel veritabanı olarak nitelendirebileceğimiz kategori içinde yer alan bir üründür. Yeni nesil veritabanı olarak ta ifade edilen yatay ölçeklendirilebilen veritabanı mimarisi ise verinin tek bir merkez noktada toplanması ve işlenmesi yerine birden fazla konumda işlevsel olarak konumlandırılması ve işlenmesi üzerine oluşturulmuştur. Lisanslı bir ürün olmayan Hadoop esasen genişletilmiş ve işlevsel dosya sistemi mimarisi üzerine oluşturulmuştur. Bu yapının avantajları doğrultusunda Hadoop bir veritabanı yapısı olarak kullanıldığında yeni nesil veritabanı mimarisine örnek olarak gösterilmektedir.

Tek düğümlü Hadoop sistemi genel olarak PostgreSQL veritabanı sistemine benzemektedir. Sunucu kaynakları ile birlikte veritabanı sistemi kaynaklarının ortak kullanılması tek bir sunucu üzerinde büyümenin zor olduğunu göstermektedir. Bu tip sistemler kurulmaya başlandığında, zamanın gerektirdiği kaynak ihtiyacına göre tasarlandığından ileride oluşabilecek iş yüküne karşı zaman içerisinde cevap veremeyecek pozisyona gelmektedirler. Bu nedenle şirketler birkaç yıl içinde bu tip sistemlerini yenileme ihtiyacı duymaktadır. Bu durum şirketler için hem maliyet, hem fazladan operasyonel iş yükü hemde zaman kaybı olarak karşılarına çıkmaktadır.

Diğer açıdan yatay olarak büyüme (çok düğümlü Hadoop sistemi) bakım sözleşmeleri bitmiş ya da atıl durumdaki sunucuları devre dışı bırakmak yerine sisteme entegre

edilmeleri, herhangi bir maliyete katlanılmadan yapının büyütülmesine imkan tanımaktadır.



Şekil 6.1 Hadoop Sisteminde Verinin Dağılımı

Yatay olarak büyüme ile birlikte Hadoop sistem mimarisi gereği varsayılan olarak her bir verinin sistem içerisinde farklı sunucular üzerinde dağılmış 3 adet kopyası bulunmaktadır. Ana Düğüm üzerinde bu veri bloklarının hangi Veri Düğüm sunucularında bulunduğu yer alır. Veriye erişim sağlanmak istendiğinde ise Ana Düğüm noktasından ilgili veriye ait lokasyon bilgileri alınarak paralel işleme süreci başlar. Herhangi bir sebeple bir Veri Düğüm noktasına erişim olmadığı (sunucunun arızalanması, ağ bağlantısının kopması, vb.) durumlarda süreç farklı bir Veri Düğüm noktasına yönlendirilir. Verinin farklı sunucular üzerinde kopyaları bulunduğundan şirketlere ve kurumlara fazladan bir yedekleme maliyeti getirmemektedir.

Geleneksel sistemlerde sistemin bir ürünü ya da katmanı olan yazılım araçları ile (yönetim, raporlama ve analiz servisi, entegrasyon servisi, vb.) analiz ve kontroller

yapılırken, yeni nesil sistemlerde ise farklı açık kaynak yazılımlarının oluşturduğu çeşitlilik ile (Örneğin, Kylo – Veri Entegrasyonu, Nifi – Anlık Dış Kaynak Verisi Transferi, Presto – Sistem belleği üzerinde hızlı analitik sorgu aracı, vb.) geleneksel sistemlerdeki kısıtlı ve çok aşamalı iş süreçlerinin rahat ve entegre yapılabilmesine olanaklar oluşmaktadır. Örneğin geleneksel yapı üzerine bir R (İstatistlik ve Raporlama Aracı) altyapısı kurulamazken, yeni nesil veritabanı mimarilerinde sunucular üzerine R sisteminin entegre edilmesi ve analizlerin paralel yürütülmesi sağlanabilmektedir.

Geleneksel sistemlerde verinin toplanması ve işlenmesi aşamaları tek bir sunucu üzerindeki kaynakların (ram, işlemci vb.) kullanımı ile gerçekleşmektedir. Fakat yeni nesil sistemlerde ise birden fazla (en az üç) donanımsal yeterliliği daha düşük sunucunun kullanımına bağlı olarak kaynakların dağıtılması verimli ve etkin bir kullanım ortamı oluşturmaktadır. Bu doğrultuda yeni nesil sistemler, veri üzerinde yapılan analizlerin daha kısa sürelerde sonuçlanmasına imkan tanımakta, gereksiz zaman kayıplarının önüne geçmektedirler.

Bu çalışmada geleneksel veritabanı sistemleri (PostgreSQL) ile yeni nesil veritabanı sistemleri (Hadoop) arasında performans tabanlı bir karşılaştırma gerçekleştirmek hedeflendi. Bu doğrultuda benzer donanıma sahip ortamlar oluşturuldu. Bu ortamlar üzerinde aynı boyuta ve içeriğe sahip veriler üzerinde aynı sorgu cümleleri ile sorgulamalar gerçekleştirildi. Bu sorgulamalara sistemlerin verdiği tepkiler (zaman boyutunda) ölçüldü. Elde edilen sonuçlar doğrultusunda iki farklı mimarinin birbirlerine karşı farklılıkları tespit edildi.

Tek Düğümlü Hadoop mimarisi ile PostgreSQL üzerinde yapılan sorgulamalarda (20 farklı sorguda) çalışma kapsamında elde edilen bulgulara göre; yeni nesil veritabanı

mimarisi olan Hadoop yapısının geleneksel mimariyi temsil eden PostgreSQL'e göre yaklaşık %82 daha hızlı sonuç ürettiği görülmüştür. Çalışma kapsamında oluşturulan Çok Düğümlü Hadoop mimarisi ile PostgreSQL yapısı üzerinde yapılan sorgulamalarda ise (20 farklı sorguda) Hadoop yapısının PostgreSQL'e göre yaklaşık %110 (iki katının üstünde) daha hızlı sonuç ürettiği ve ayrıca Çok Düğümlü Hadoop mimarisinin Tek Düğümlü Hadoop mimarisine oranla %13 daha hızlı sonuç ürettiği görülmüştür.

Yukarıda belirtilen bulgulardan da anlaşıldığı gibi benzer donanıma, aynı boyutta veriye sahip iki sistem arasında performans açısından oldukça büyük farklar olduğu görülmektedir. Maliyet ve işlem süresi (verimlilik) açısından yeni nesil veritabanlarının geleneksel veritabanlarına göre büyük veri ile çalışan şirket ve kurumlar tarafından tercih edilmesi gereken bir mimari olduğu açık ve net olarak bu çalışmada ispatlanmış ve ortaya konulmuştur.

Günümüzde işletmeler artık ihtiyaçları doğrultusunda sistem mimarileri tasarlarken, ölçeklenebilirlik, süreklilik, paralel işlem yetenekleri, fiyat/performans, esnek tasarım gibi parametreleri göz önünde bulundurmaktadırlar. Hadoop gibi yatay olarak ölçeklendirilebilen veritabanı mimarileri bu parametrelerin hepsini karşılamaktadır. Ayrıca açık kaynak dünyasının da hızla gelişimine bağlı olarak Hadoop ve benzeri yeni nesil veritabanı mimarileri kendilerine yeni nitelikler ve özellikler kazandırarak, gelecekte şirketlerin bünyesinde tercih edilen sistemler olacaklardır.

KAYNAKÇA

Akdoğan, H. (2016, 7). *Apache Hadoop*. kodcu: https://kodcu.com/2016/07/apache-hadoop/ adresinden alındı

Apache. (2008). *Apache Hadoop*. MapReduce Tutorial: https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html#Overview adresinden alındı

Apache. (2018). *Apache Hadoop YARN*. Apache: https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html adresinden alındı

Apache Hadoop vs. PostgreSQL. (2017). vschart: http://vschart.com/compare/apache-hadoop/vs/postgresql adresinden alındı

Apache Pig. (2018). *Welcome to Apache Pig!* Apache: https://pig.apache.org/ adresinden alındı

Apache Sqoop. (2012, 04 02). *Apache Sqoop*. Apache Blog: https://blogs.apache.org/sqoop/entry/apache_sqoop_graduates_from_incubator adresinden alındı

Apache Wiki. (2011). *DataNode*. Hadoop Wiki: https://wiki.apache.org/hadoop/DataNode adresinden alındı

Apache Wiki. (2011). *NameNode*. Hadoop Wiki: https://wiki.apache.org/hadoop/NameNode adresinden alındı

Benjelloun. (2015). An overview of big data opportunities, applications and tools. *Intelligent Systems and Computer Vision (ISCV)*, 1-6.

Bista, N. (2018). *HADOOP vs RDBMS|Know The 12 Useful Differences*. Educba: https://www.educba.com/hadoop-vs-rdbms/ adresinden alındı

Chen, M., Mao, S., Zhang, Y., & Leung, V. (2014). *Big Data Related Technologies, Challenges and Future Prospect.* New York: Springer.

Cloudera. (2018). *Cloudera Manager*. Cloudera: https://www.cloudera.com/products/product-components/cloudera-manager.html adresinden alındı

Demir, İ. (2012). *Hadoop Tabalı Büyük Ölçekli Görüntü İşleme Altyapısı.* Kocaeli: Yüksek Lisans Tezi.

Elmasri, R., & Navathe, S. (2011). *Fundamentals Of Database Systems 6th Edition.* USA: Addison-Wesley.

Flume. (2017). *Apache Flube*. Apache: https://flume.apache.org/ adresinden alındı

google. (2018). *google trends*. https://trends.google.com/trends/explore?date=all&q=apache%20hadoop adresinden alındı

Google Trend. (2018, 93). Google: https://trends.google.com/trends/explore?date=today%2012-m,today%2012-m&geo=,&q=PostgreSQL,Hadoop adresinden alındı

Hansen, A. (2015). *What is the difference between a Hadoop database and a traditional relational database*. quora: https://www.quora.com/What-is-the-difference-between-a-Hadoop-database-and-a-traditional-relational-database-e-g-Oracle-Why-when-would-you-choose-one-over-the-other adresinden alındı

Hive, A. (2017). *What Is Hive*. Apache: https://cwiki.apache.org/confluence/display/Hive/Tutorial#Tutorial-WhatIsHive adresinden alındı

Hortonworks. (2017). https://github.com/hortonworks/hive-testbench adresinden alındı

Hortonworks. (2018). *Apache Ambari*. Hortonworks: https://hortonworks.com/apache/ambari/ adresinden alındı

IBM. (2017, 11 19). *IBM DB2*. wikipedia: https://tr.wikipedia.org/wiki/IBM_DB2 adresinden alındı

Ineni, N. (2015). *What is the difference between a Hadoop database and a traditional relational database*. quora: https://www.quora.com/What-is-the-difference-between-a-Hadoop-database-and-a-traditional-relational-database-e-g-Oracle-Why-when-would-you-choose-one-over-the-other adresinden alındı

Ivens, P., Alencar, P., & Cowan, D. (2014). A Survey on Domain-Specific Languages for Machine Learning in Big Data. s. 2. https://omurbenek.com/2014/06/01/buyuk-verinin-4-adimi-4-vs-of-big-data/ adresinden alındı

Kroenke, D. (1998). *Database Processing: Fundamentals, Design and Implementation.* Prentice Hall, USA: Pearson.

Liu, X. (2012). *DataWarehousingTechnologiesfor Large-scaleandRight-timeData.* Denmark: Doctoral Thesis.

Mayer, V., & Cukier, K. (2013). *Büyük Veri - Yaşama, Çalışma ve Düşünme Şeklimizi Dönüştürecek Bir Devrim.* İstanbul: Paloma.

Memishi, B. (2016). *Optimizing the reliability and resource efficiency of MapReduce-based systems.* Madrid: Doctoral Thesis.

Most Popular Database Management. (2018, 08 31). Top 30 Most Popular Database Management Software: Complete List: https://www.softwaretestinghelp.com/database-management-software/ adresinden alındı

Munvo. (2013). *Hadoop Big Data vs. Relational Databases*. munvo: http://munvo.com/2017/07/18/hadoop-big-data-vs-relational-databases/ adresinden alındı

Noyes, D. (2019). *The Top 20 Valuable Facebook Statistics – Updated March 2019*. Zephoria Digital Marketing: https://techcrunch.com/2012/08/22/how-big-is-facebooks-data-2-5-billion-pieces-of-content-and-500-terabytes-ingested-every-day/ adresinden alındı

Oozie. (2017). *Apache Oozie Workflow Scheduler for Hadoop*. Apache: http://oozie.apache.org/ adresinden alındı

Önder, E. (2005). *Yönetim BiliGim Sistemleri Kapsamında Web Tabanlı İlişkisel Veri tabanı Yönetim Sistemleri ve Bir Uygulama.* istanbul: İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü.

Özcan, S. (2013). *Difference between Hadoop and RDBMS*. oraclesys: https://oraclesys.com/2013/04/03/difference-between-hadoop-and-rdbms/ adresinden alındı

Patodi, R. (2011). *What is Hadoop*. Java Code Geeks: https://www.javacodegeeks.com/2011/05/hadoop-soft-introduction.html adresinden alındı

Polat, E. (2012). *Kitaplar ortalama kaç sözcükten oluşur?* Edebiyat Haber: http://www.edebiyathaber.net/kitaplar-ortalama-kac-sozcukten-olusur/ adresinden alındı

Postgre. (2017). https://www.postgresql.org adresinden alındı

Postgre. (2017). https://www.postgresql.org/about/history/ adresinden alındı

Postgre. (2017). https://www.postgresql.org/about/advantages/ adresinden alındı

PostgreSQL. (2018). *About*. PostgreSQL: https://www.postgresql.org/about/ adresinden alındı

Ritika Prasad. (2017, 7 31). *What is the advantages of Hadoop and Big data?* Quora: https://www.quora.com/What-is-the-advantages-of-Hadoop-and-Big-data adresinden alındı

Robinson, I., Eifrem, E., & Webber, J. (2015). I. Robinson, E. Eifrem, & J. Webber içinde, *Graph Databases* (s. 6-15). United States of America: O'Reilly Media, Inc.

Roy, S. (2017). *Comparison between hadoop and RDBMS*. data-flair.training: http://data-flair.training/forums/topic/comparison-between-hadoop-and-rdbms adresinden alındı

Sanjay. (2014). *Hadoop Basics - Horizontal Scaling instead of Vertical Scaling*. Tech Universe: http://samjay-complete.blogspot.com.tr/2014/03/hadoop-basics-horizontal-scaling.html adresinden alındı

SAP. (2014). *SAP and the German Football Association Turn Big Data Into Smart Decisions to Improve Player Performance at the World Cup in Brazil*. sap: https://news.sap.com/sap-dfb-turn-big-data-smart-data-world-cup-brazil/ adresinden alındı

Speelpenning, J., Daux, P., & Gallus, J. (2001). Database. O. Corporation içinde, *Data Modeling and Relational Database Design* (s. 2-6). San Francisco: Shores.

Şarkışla, H. (2015). *HADOOP MAPREDUCE ALGORİTMASININ ANALİZİ İLE PERFORMANSA ETKİ EDEN PARAMETRELERİN TESPİTİ VE HADOOP ÜZERİNDE BAŞARIM ARTIMI.* Sakarya: Yüksek Lisans Tezi.

Techopedia. (2018). *Oracle Database*. Techopedia: https://www.techopedia.com/definition/8711/oracle-database adresinden alındı

Techsparks. (2016). *Big Data and Hadoop*. Techsparks: http://www.techsparks.co.in/thesis-topics-in-big-data-and-hadoop/ adresinden alındı

TPC. (2017). *TPC BENCHMARK*. TPC: http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.17.1.pdf adresinden alındı

TPC. (2018). http://www.tpc.org/information/about/abouttpc.asp adresinden alındı

Ullman, J., Widom, J., & Molina, H. (2009). Database Systems. J. D. Ullman, J. Widom, & H. Molina içinde, *DATABASE SYSTEMS The Complete Book* (s. 1-3). New Jersey: Pearson Education Inc.

Waleed, A. (2018). *Difference Between Big Data Hadoop And Traditional RDBMS*. w3trainingschool: https://www.w3trainingschool.com/difference-big-data-hadoop-traditional-rdbms adresinden alındı

White, T. (2011). A Brief History of Hadoop. T. White içinde, *Hadoop: The Definitive Guide* (s. 9). United States of America: O'Reilly Media.

White, T. (2011). The Hadoop Distributed Filesystem. T. White içinde, *Hadoop: The Definitive Guide* (s. 41). United States of America: O'Reilly Media, Inc.

Why MapR. (2018). MapR: https://mapr.com/why-mapr/ adresinden alındı

Widom, J., Ullman, J., & Molina, H. (2008). *DATABASE SYSTEMS The Complete Book - Second Edition.* New Jersey: Department of Computer Science Stanford University.

Wikipedia. (2016). *MySQL*. https://tr.wikipedia.org/wiki/MySQL adresinden alındı

Wikipedia. (2017). *Hue (Hadoop)*. Wikipedia: https://en.wikipedia.org/wiki/Hue_(Hadoop) adresinden alındı

Wikipedia. (2018). *Microsoft SQL Server*. Wikipedia: https://en.wikipedia.org/wiki/Microsoft_SQL_Server adresinden alındı

Wikipedia. (2018). *PostgreSQL*. Wikipedia: https://tr.wikipedia.org/wiki/PostgreSQL adresinden alındı

Wu, J. (2016). http://myfpgablog.blogspot.com.tr/2016/08/tpc-h-queries-on-postgresql.html adresinden alındı

Zookeeper. (2017). *What is ZooKeeper?* Apache: https://zookeeper.apache.org/ adresinden alındı

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı: Murat MENTEŞE

Uyruğu: TC

Doğum Tarihi ve Yeri: 30/06/1986 Kadıköy / İstanbul

Tel: 0555 558 67 25

Email: muratmentese@gmail.com

EĞİTİM

Derece	**Eğitim Birimi**	**Eğitim Tarihi**
Yüksek Lisans	Okan Üniversitesi / Bilişim Sistemleri	2015-2018
Lisans	Anadolu Üniversitesi/ İşletme Fakültesi	2013-2015
Ön Lisans	Marmara Üni./ Elektrik Bölümü	2004-2006
Lise	Haydapaşa Endüstri Meslek Lisesi/ Elektrik	2001-2004

İŞ DENEYİMLERİ

Kurum	**Görev**	**Yıl**
Migros Ticaret A.Ş.	Veriambarı ve Raporlama - Senior	2018-Halen
Migros Ticaret A.Ş.	Veritaban Bölüm Sorumlısı	2018-2018
Migros Ticaret A.Ş.	Veritabanı Uzmanı	2013-2018
Migros Ticaret A.Ş.	Sistem Uzmanı	2011-2013
Superonline	Sistem Destek Uzmanı	2010-2011
Turkcell	Çalışan Yardım Masası	2009-2010

EK-1

1.1. Apache Hadoop Kurulumu

CentOS 7 işletim sistemi kurulumu tamamlandıktan sonra Hadoop kurulumu için gerekli olan kurulumlar aşağıdaki şekildedir.

1.1.1. Sunucu Ayarları

Tüm sunucularımızın hosts dosyasına ana ve veri düğümlü Hadoop sunucularımızın IP ve sunucu isimleri kaydedilir.

- vi /etc/hosts

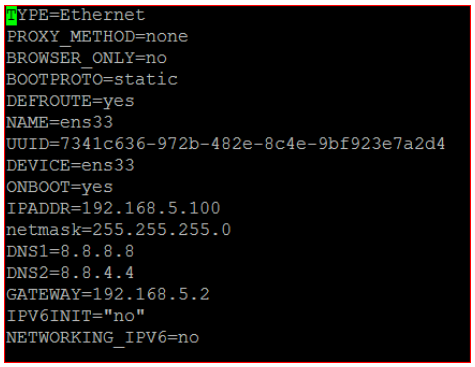
```
#127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
#::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.5.100 master
192.168.5.101 slave1
192.168.5.102 slave2
192.168.5.103 slave3
```

Şekil EK-1.1 Hadoop Kurulum Aşamaları – Host Dosyası Düzenleme

Ağ ayarları için sunucularımızın IP adreslerini belirlemek için ağ aygıtının ismi ile birlikte ilgili ağ dosyasının IP yapılandırması yapılır ve devamında ağ servisi yeniden başlatılır.

- vi /etc/sysconfig/network-scripts/ifcfg-ens33
- /etc/init.d/network restart



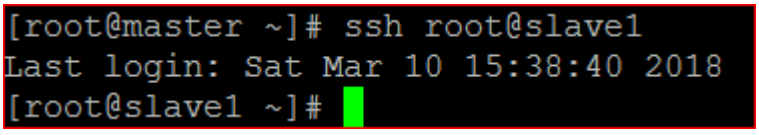
Şekil EK-1.2 Hadoop Kurulum Aşamaları – Ağ Ayarları

Ana düğüm sunucumuzun diğer veri düğümlü sunucularımıza direk olarak erişim sağlaması adına SSH güvenlik anahtarı oluşturulur ve bu anahtarlar veri düğümlü sunuculara gönderilir.

- ssh-keygen
- cat /root/.ssh/id_rsa.pub >> /root/.ssh/authorized_keys
- chmod 700 ~/.ssh
- chmod 600 ~/.ssh/authorized_keys
- ssh-copy-id kullanıcıadı@veridüğümipadresi

ilgili güvenlik anahtarı veri düğümü sunucularına aktarıldıktan sonra ana düğüm sunucusu üzerinden ilgili komut ile şifresiz erişim sağlanır.

- ssh root@slave1



Şekil EK-1.3 Hadoop Kurulum Aşamaları – SSH Yönlendirme

Sunucular arasındaki zaman farkını önlemek adına NTP servisi kurulu ve aktif edilir.

- yum install ntp
- service ntpd start
- systemctl enable ntpd

Sunucular üzerinde varsa güvenlik duvarı kapatılır ve sunucu yeniden başladıktan sonra tekrar aktif hale gelmemesi için kalıcı olarak devre dışı bırakılır.

- systemctl stop firewalld
- systemctl disable firewalld

CentOS işletim sisteminde zorunlu erişim denetimi yapan SELinux servisi kalıcı olarak kapatılır.

- vi /etc/selinux/config

SELINUX=disabled

Ana ve veri düğümleri üzerinde iptables servisi durdurulur ve kalıcı olarak kapatılır.

- systemctl stop iptables
- systemctl stop ip6tables
- systemctl disable iptables
- systemctl disable ip6tables

Kurulum esnasında işletim sistemi üzerinde oluşturulacak kullanıcılar için varsayılan yetkilendirme ayarı yapılır.

- umask 0022

echo umask 0022 >> /etc/profile

1.1.2. Ambari Kurulumları

İşletim sistemi ayarları tamamlandıktan sonra Hortonworks sitesi üzerinden Ambari kurulumu için seçmiş olduğumuz versiyona ait kaynak dosyasını indirerek kurulum başlatılır. Ambari kurulumu ana düğüm sunucusu üzerine kurulur ve veri düğümlü sunucular üzerine Ambari Agent kurulumu yapılır.

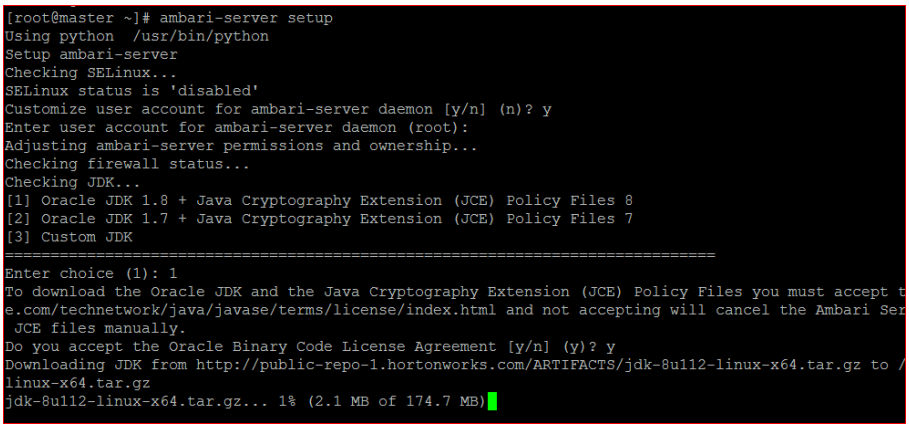
1.1.1.1 Ambari Server Kurulumu

- wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.6.1.5/ambari.repo -O /etc/yum.repos.d/ambari.repo
- yum repolist
- yum install ambari-server

Ambari Server kurulumu tamamlandıktan sonra kurulum ayarları yapılarak servis başlatılır.

- ambari-server setup

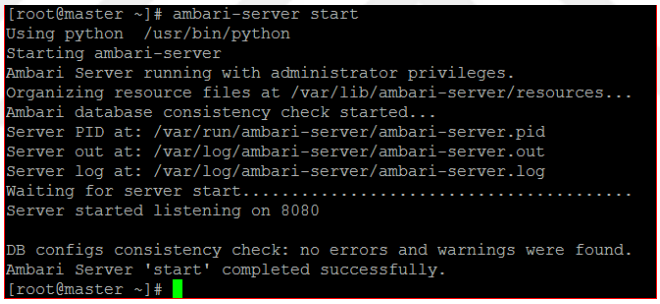
Karşımıza çıkan menüden java kurulumu ile birlikte metadata verilerinin tutulacağı yerel veritabanı oluşturulur ve ardından kurulum tamamlanır.



Şekil EK-1.4 Hadoop Kurulum Aşamaları – Ambari Kurulumu

Tamamlanan kurulum sonrasında servis başlatılır.

- ambari-server start



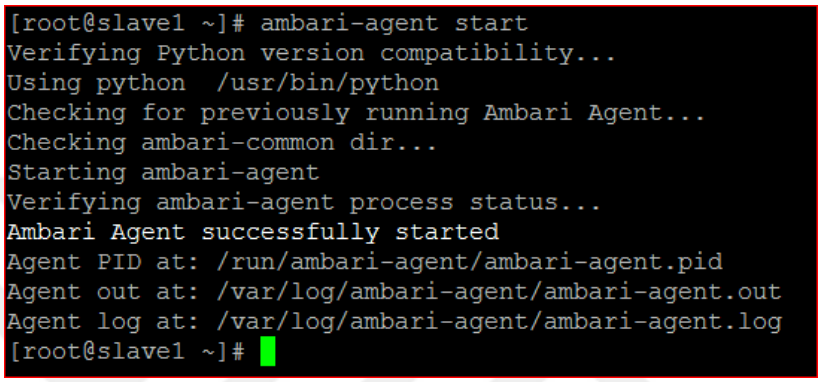
Şekil EK-1.5 Hadoop Kurulum Aşamaları – Ambari Servisinin Başlatılması

1.1.1.2. Ambari Agent Kurulumu

- wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.6.1.5/ambari.repo -O /etc/yum.repos.d/ambari.repo
- yum repolist
- yum install ambari-agent

Ambari Agent kurulumu tamamlandıktan sonra ana düğüm ile iletişime geçebilmesi için ilgili ayar dosyasına ana düğüm IP adresi yazılır ve Ambari Agent başlatılır.

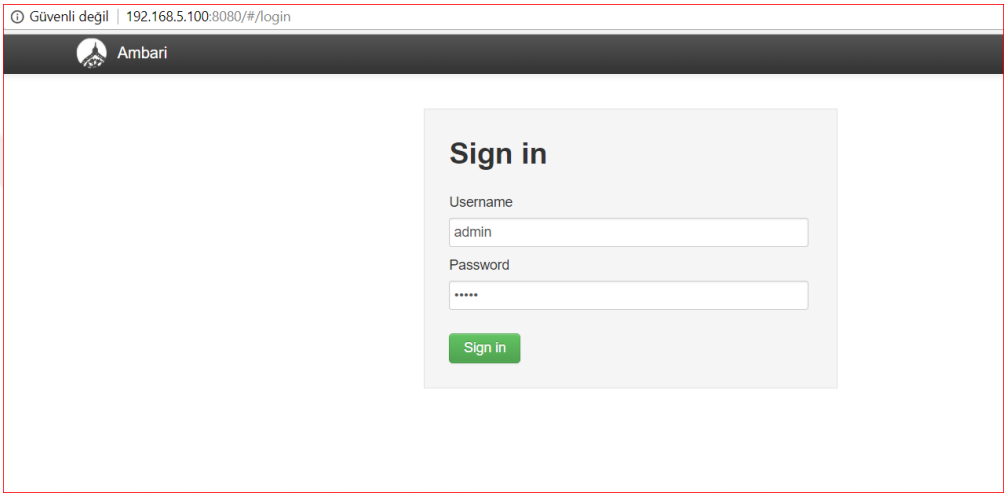
- vi /etc/ambari-agent/conf/ambari-agent.ini
- ambari-agent start



Şekil EK-1.6 Hadoop Kurulum Aşamaları – Ambari Agent Kurulumu

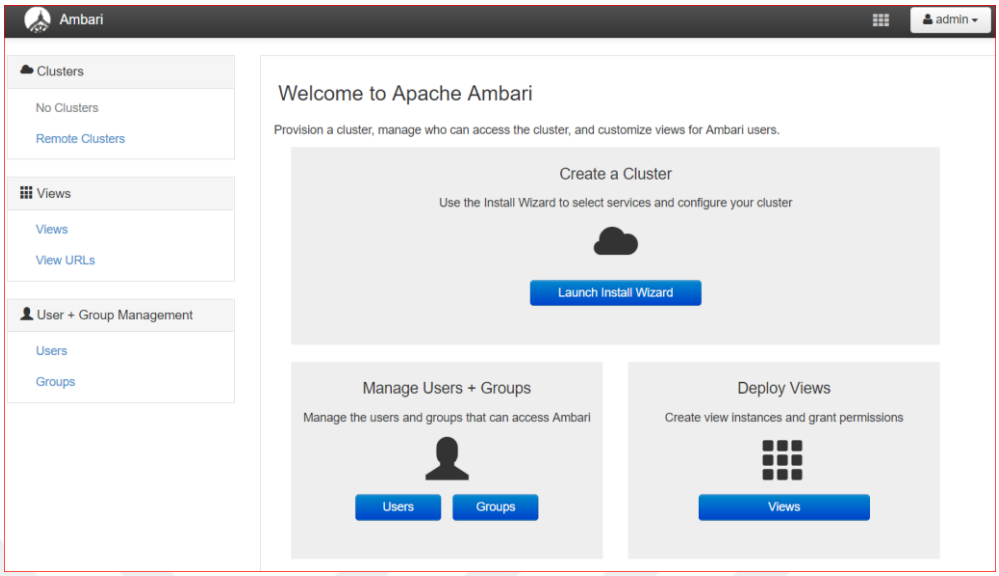
1.1.3. Ambari Sunucu Entegrasyonu ve Hadoop Ekosisteminin Kurulması

Ambari Server ve Agent kurulumları tamamlandıktan ve servisler başlatıldıktan sonra internet tarayıcısı üzerinden ana düğüm sunucumuzun IP ve port adresini yazarak kurulumun web arayüzüne erişim sağlanır. İlk kurulum sonrasında web arayüzü için başlangıçtaki kullanıcı adı ve şifresi admin olarak tanımlanmıştır.



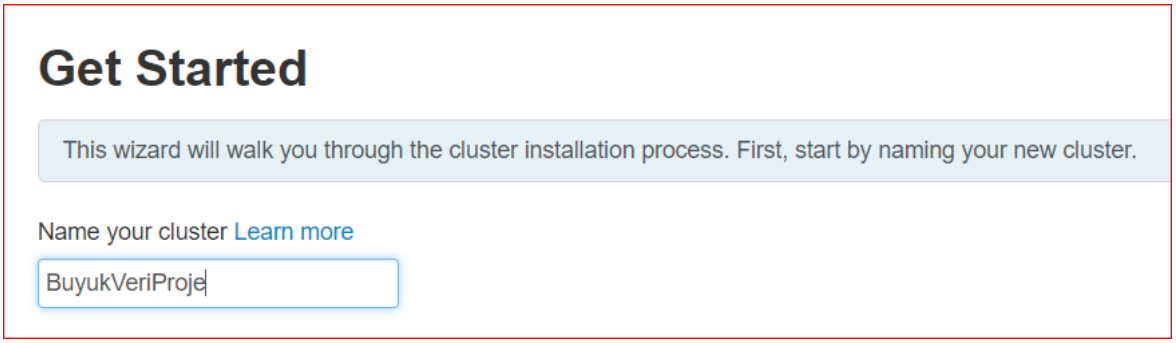
Şekil EK-1.7 Ambari Kurulum Aşamaları – Ambari Yönetim Paneli Giriş Ekranı

Sisteme giriş yapıldıktan sonra ilk adım olarak yeni bir yapı oluşturulması için “Launch Install Wizard” seçeneği seçilir.



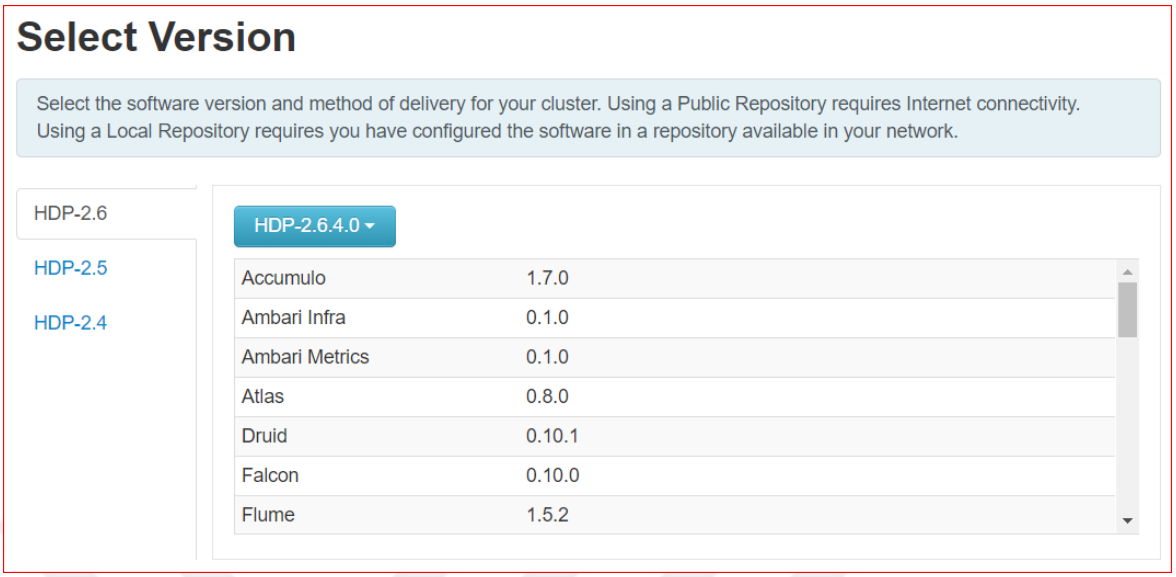
Şekil EK-1.8 Ambari Kurulum Aşamaları – Cluster Kurulumu

Proje ismi yazılır.



Şekil EK-1.9 Ambari Kurulum Aşamaları – Cluster İsim Verme

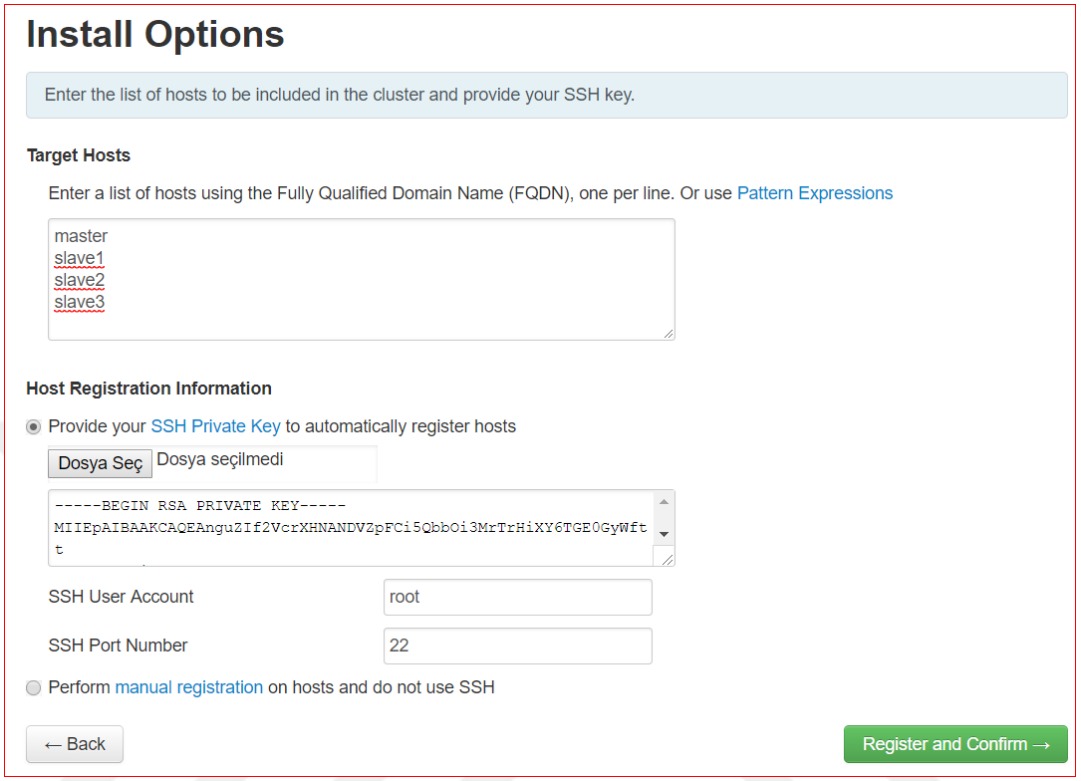
Kurulacak versiyon seçimi yapılır.



Şekil EK-1.10 Ambari Kurulum Aşamaları – Versiyon Seçimi

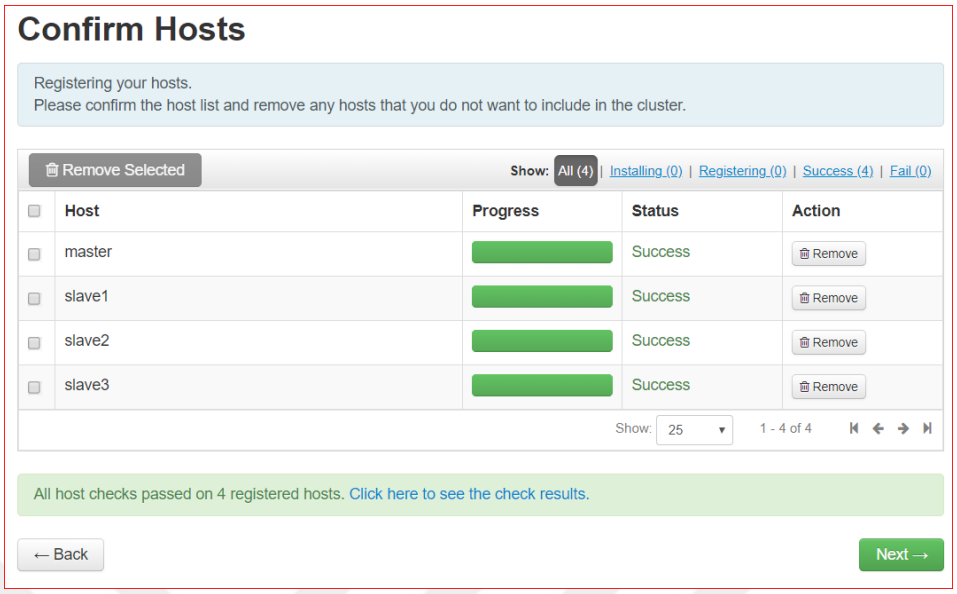
Sunucuların isimleri ve ana düğüm sunucusunun güvenlik anahtarı eklenir. Güvenlik anahtarı ana düğüm sunucusu üzerinde aşağıdaki komut ile elde edilir.

- cat ~/.ssh/id_rsa



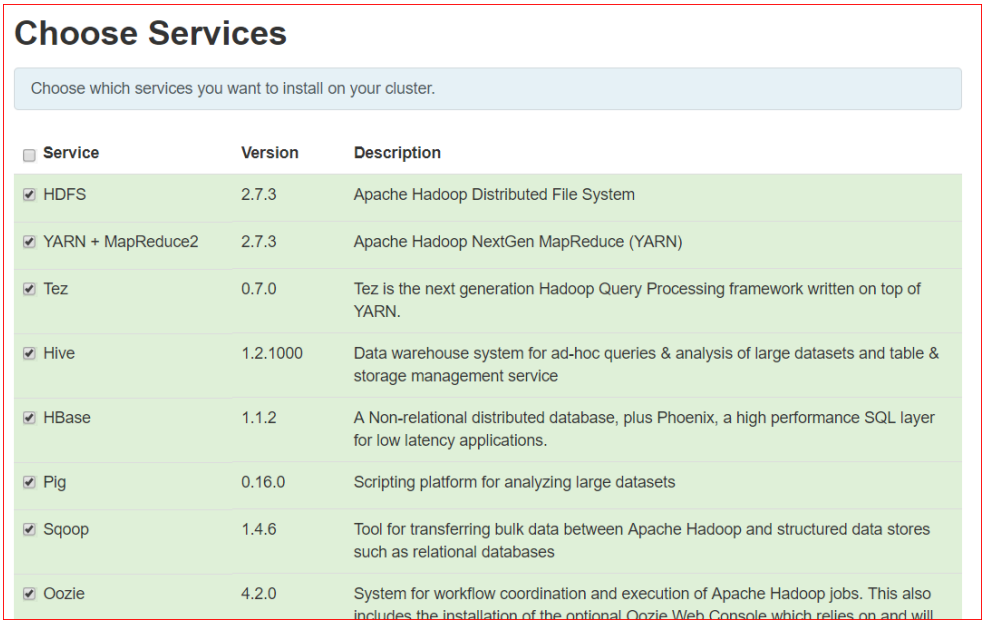
Şekil EK-1.11 Ambari Kurulum Aşamaları – Sunucuları Ekleme

Ana düğüm ile veri düğüm sunucuları üzerinde otomatik kontroller servis tarafından yapılır ve başarılı bir şekilde bitmesi beklenir. Eğer herhangi bir hata alıyor ise ilgili hata kontrol edilerek hangi sunucuda almış ise düzeltilerek işlem tekrarlanır.



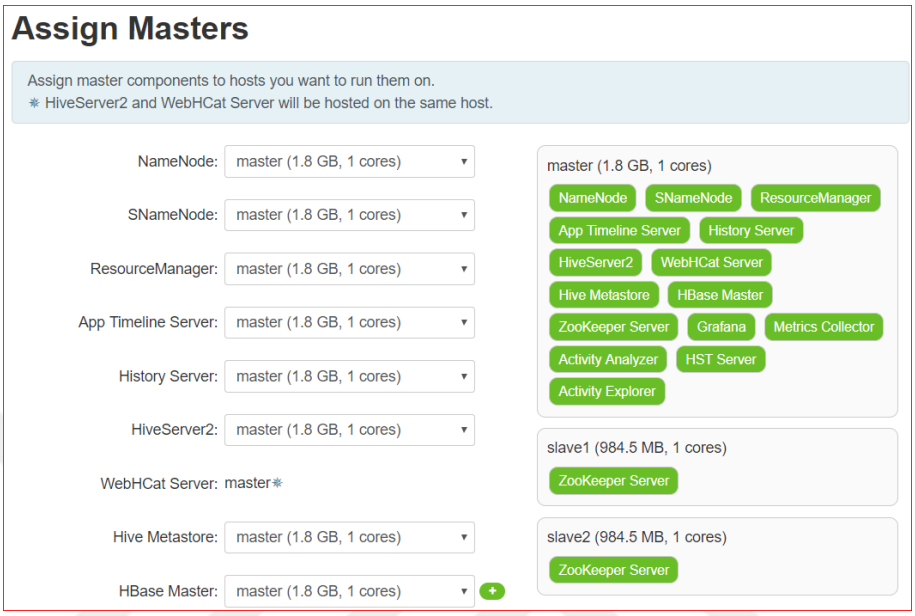
Şekil EK-1.12 Ambari Kurulum Aşamaları – Sunucuları Kontrol Etme

Kontrol işlemi başarılı bir şekilde tamamlandıktan sonra Hadoop ekosistemi için kurulacak servisler seçilir.



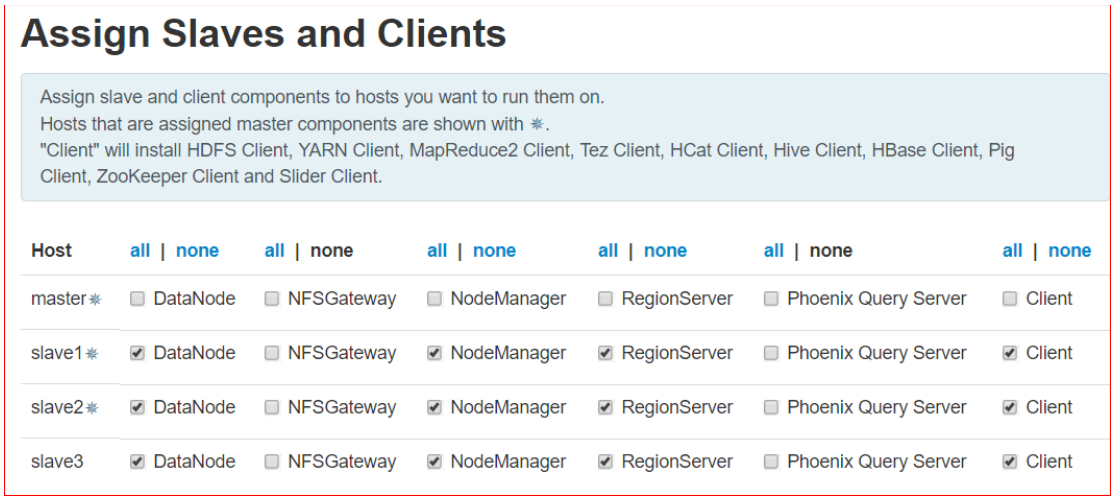
Şekil EK-1.13 Ambari Kurulum Aşamaları – Kurulacak Servislerin Seçimi

Seçilen servislerin hangi sunucular üzerine kurulacağı belirlenir. Genel olarak servisler ana düğüme taşınır. Veri düğüm sunucuları üzerine servis ataması yapılmaz.



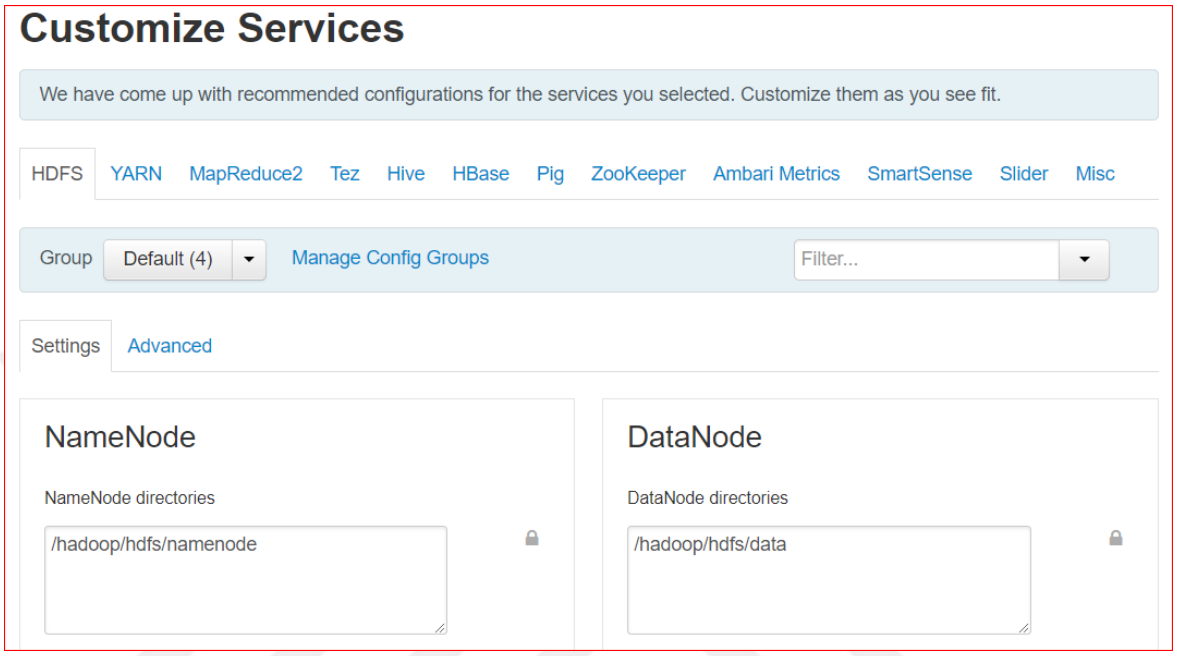
Şekil EK-1.14 Ambari Kurulum Aşamaları – Servisleri Sunuculara Atama

Veri düğüm sunucuları "DataNode" olarak tanımlanır ve seçilen servislerin "Client" özellikleri eklenir.



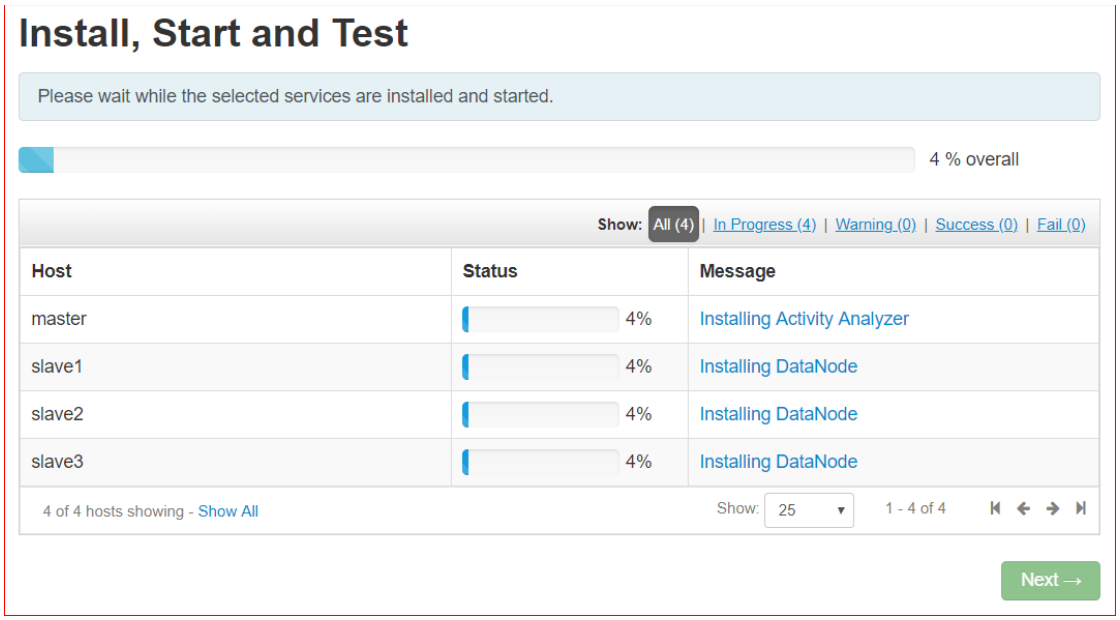
Şekil EK-1.15 Ambari Kurulum Aşamaları – Servislerin Çalıştırılabilir Dosyalarını Sunuculara Kopyalama

Kurulacak olan servislerin ayarları yapılır. (log dosya yolu, başlantı şifreleri vb.) Ve kurulum başlatılır.



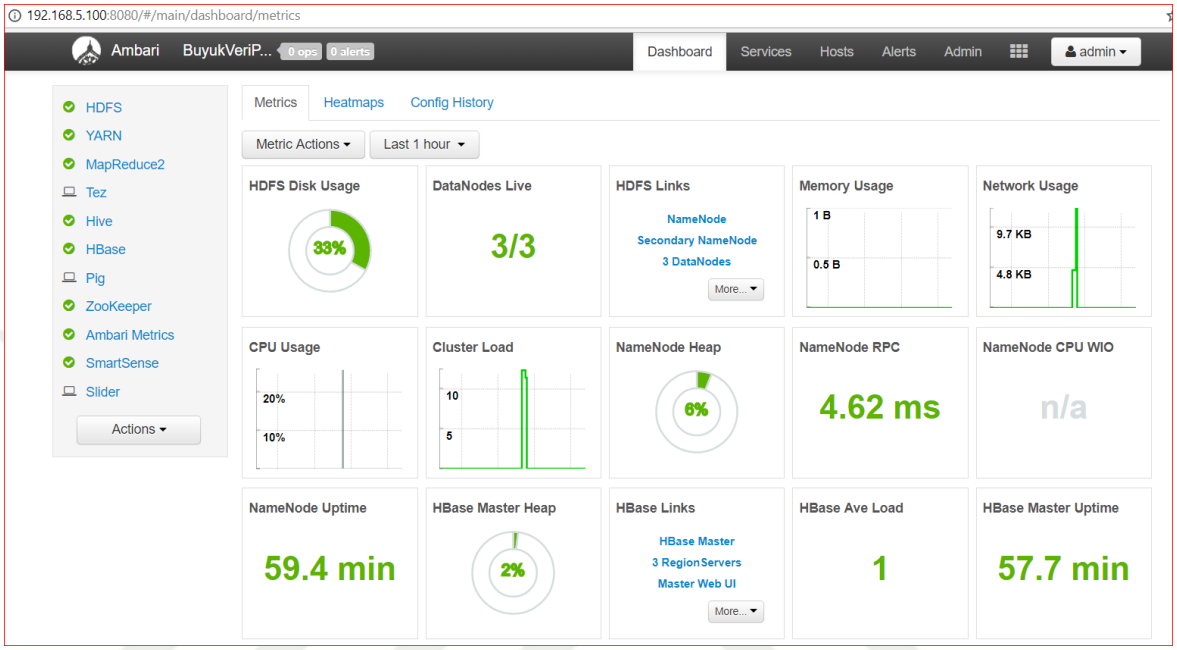
Şekil EK-1.16 Ambari Kurulum Aşamaları – Servislerin Ayarlarının Yapılması

Kurulum başlatıldıktan sonra Ambari'nin sunucular üzerindeki kurulumları izlenir.



Şekil EK-1.17 Ambari Kurulum Aşamaları – Yükleme İşleminin Başlaması

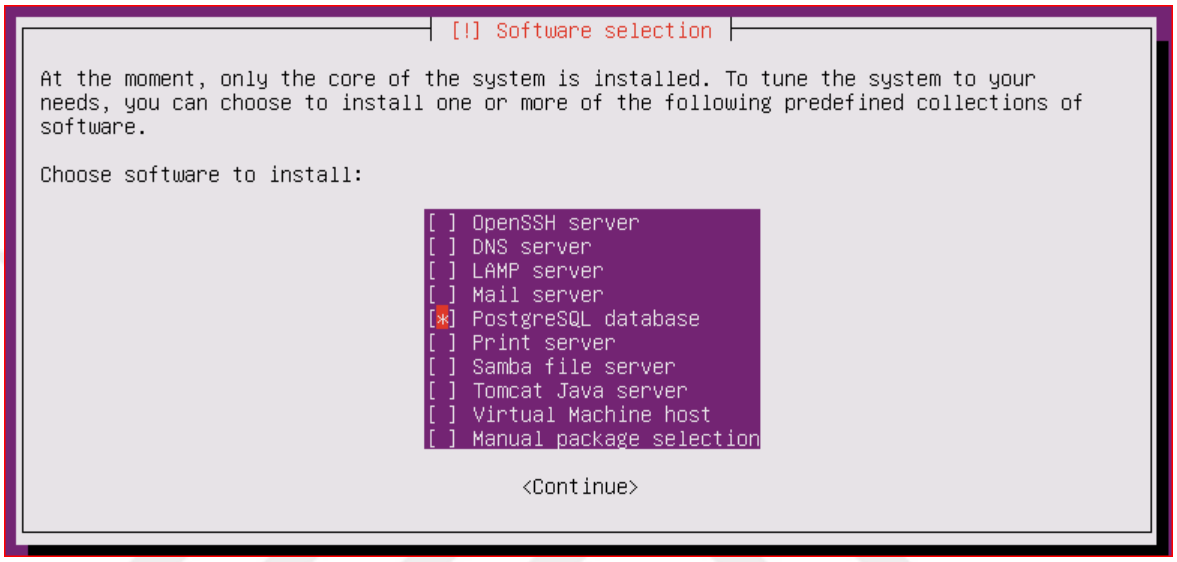
Kurulumların ardından Büyük Veri Platformu kullanıma hazır olacaktır.



Şekil EK-1.18 Ambari Kurulum Aşamaları – Ambari Ana Ekranının Görünümü

1.2 PostgreSQL Kurulumu

CentOS 7 versiyonu ile birlikte işletim sistemi kurulumu yapılırken "Yazılım Seçenekleri" ekranında PostgreSQL database seçimi yapılarak kurulum gerçekleştirilir.



Şekil EK-1.19 PostgreSQL Kurulum Aşamaları – Servisin Seçimi ve Kurulumu

1.3 PostgreSQL TPC-H Kurulumu

PostgreSQL kurulu sisteme TPC-H kurulumunun yapılması. (Wu, 2016)

Gerekli olan paketlerin yüklenmesi;

- `# apt-get install unzip, make, gcc, 389-ds-base, multimon, mgen, mono-devel paketlerinin kurulması.`

İlgili Dizinin oluşturulması ve dosyanın açılması;

- `# mkdir /opt/db/tpch-tool`
- `# cd /opt/db/tpch-tool`
- `# unzip tpc-h-tool_2_17_0.zip`
- `# cd /opt/db/tpch-tool/2.17.2/dbgen`

Dosyaların derlenmesi;

- `# cp makefile.suite Makefile`
- `# vi Makefile`
- `CC=gcc`
- `DATABASE=ORACLE`
- `MACHINE=LINUX`
- `WORKLOAD=TPCH`
- `# make`

Veri boyutunun ayarlanması ve oluşturulması;

- `# ./dbgen -s 10 (10 GB'lık veri oluşturmakta)`
- `# for i in `ls *.tbl`; do sed 's/|$//' $i > ${i/tbl/csv}; echo $i; done;`

PostgreSQL üzerinde veritabanı oluşturulması ve verinin yüklenmesi.

- `# su – postgres`
- `# psql`
- `# create database tpch;`
- `# \c tpch;`
- `# \i /opt/db/tpch-tool/2.17.2/dbgen/queries/tpch-build-db.sql`
- `# \d+`

TPC-H SQL sorgularının oluşturulması.

- `# cd /opt/db/tpch-tool/2.17.2/dbgen/queries`
- `# cp /opt/db/tpch-tool/2.17.2/dbgen/dists.dss .`
- `# for q in `seq 1 22`;do DSS_QUERY=/opt/db/tpch-tool/2.17.2/dbgen/queries qgen $q > $q.sql;done;`

Oluşturulan TPC-H sorgularının çalıştırılması.

- `# su – postgres`
- `# psql`
- `# \c tpch;`
- `# \timing`
- `# \i /opt/db/tpch-tool/2.17.2/dbgen/queries/1.sql`
- `# q`

1.4 Apache Hadoop TPC-H Kurulumu

Hadoop kurulu sisteme TPC-H kurulumunun yapılması. (Hortonworks, 2017)

Gerekli olan paketlerin yüklenmesi;

- `# apt-get install build-essential`

İlgili Dizinin oluşturulması ve dosyanın açılması;

- `# mkdir /opt/db/tpch-tool`
- `# cd /opt/db/tpch-tool`
- `# unzip hive-testbench-hive14.zip`
- `# cd hive-testbench-hive14/dbgen`

Dosyaların derlenmesi;

- `./tpch-build.sh`

Veri boyutunun ayarlanması ve oluşturulması;

- `./tpch-setup.sh 10`

TPC-H sorgularının çalıştırılması.

- `# cd sample-queries-tpch`
- `# hive -i testbench.settings`
- `# hive> use tpch_flat_orc_10;`
- `# hive> source tpch_query1.sql;`