BIG DATA NEDİR?

Depolama alanının büyüklüğü ne olursa olsun günümüzde bu büyüklüğü doldurmaya yetecek miktarda veri sürekli artan bir hızla oluşmaktadır. Veri miktarı oluşumundaki hızlı artışın temel nedeni ise **sosyal ağ** etkileşimlerinin büyüyen hacmi, **lokasyon duyarlı cihazların** artması ve fiziksel dünya hakkında bilgi yakalayan ve ileten "**akıllı sensörler**"in sayısındaki artış olarak özetlenebilir.

Bilişim teknolojisindeki hızlı gelişme ile **işlemciler ve depolama alanları için giderek düşen birim maliyeti**depolanabilir veri miktarının hızla artmasına ve işletmelerin bu verileri kullanmasında bir takım fonksiyonel aksamalarla sonuçlandı.

bilinen veritabanı yönetim sistemleri ve yazılım araçlarının, verileri toplama, saklama, yönetme ve çözümleme yeteneklerini aşan büyüklükteki verilere büyük veri-big data- denir.” basit bir ifadeyle “**bir düzene uyan veya uymayan, artan hızla büyüyen ve depolanan büyük hacimli veriler**” şeklinde ifade edebiliriz. Zamanla “**verimlilik**” ve **“müşteri memnuniyeti**” kavramlarının önem kazanmasıyla bu tür analiz ihtiyaçları da artmıştır. Büyük veri **bu ihtiyaçlardan** ortaya çıkmış bir kavramdır.

Teknolojinin ilerlemesiyle üretilen verilerin hacminde **ciddi artışlar** meydana gelmiştir. Buna paralel olarak **depolama** kapasiteleri de artış gösterince artık çok büyük verileri depolamak mümkün olduğundan “büyük veri (big data)” kavramı daha da işlerlik kazandı. Fakat verilerin boyutu arttığından **klasik** veri işleme yöntemleri yetersiz kalmaya başladı. En bilinen örnekle “**ilişkisel veritabanları**” bu kadar yoğun verileri kaydetmekte yavaş kaldılar.

Yüksek hacimli bir arama motoru hizmeti veren bir firma olduğumuzu varsayalım. Arama motorumuzda hergün milyonlarca kelime aranıyor, milyonlarca tıklama yapılıyor. Eğer klasik yöntemler kullanıyor olsaydık bu verileri saklayabilmek için büyük ihtimalle milyon dolarlık donanım üzerinde çalışan yüzbinlerce dolarlık veritabanı sistemine sahip olmamız gerekirdi. Diyelim ki bu sisteme sahibiz, birbiri ile ilişkili olan kelimelerin analizini yapmak istediğimiz zaman bunu SQL sorguları ile yapabilmemiz pek mümkün değildir. Sorgu işlemleri o kadar uzun sürebilir ki çıkartacağımız sonuç güncelliğini bile yitirebilir. Başka bir engel de, bu verilerin yapılandırılmamış (unstructured) olarak tutuluyor olması durumu. Yani klasik yöntemlerle dev verinin işlenebilmesi pek söz konusu değil. Dev veri, ancak dev veri çözümleri ile işlenebilir.

Google klasik yöntemleri kullanmayarak, ihtiyacı olan teknolojiyi kendisi geliştirerek başarıya ulaştı. Google milyarlarca internet sayfasının verisini [Google File System](http://en.wikipedia.org/wiki/Google_File_System) üzerinde tutuyor, veritabanı olarak [Big Table](http://en.wikipedia.org/wiki/BigTable) kullanıyor, dev veriyi işlemek için [MapReduce](http://devveri.com/hadoop/mapreduce-nedir) kullanıyor. Bu teknolojilerin hepsi düşük maliyetli binlerce bilgisayarın bir araya gelerek oluşturduğu kümeler üzerinde çalışıyor.  Benzer bir şekilde Amazon da verilerini [DynamoDB](http://devveri.com/haberler/amazon-dynamodb) üzerinde tutuyor. Apache projeleri olarak ortaya çıkan Lucene, Solr, Hadoop, HBase gibi projeler  dev veriyi kullanabilen başarılı projelerdir.

**BIG DATA OLMASI İÇİN…**

1. **Hacim** (Volume)  
   Bir verinin büyük veri (big data) olmasının en önemli şartı “ciddi boyutlarda” olmasıdır. Verinin boyutu onun değerini belirler. Hızla artan bir veri, kaynağı hakkında daha sağlam ve keskin bilgiler verebilir.
2. **Çeşitlilik** (Variety)  
   Verinin çeşitliliği analiz edilmesindeki anlamı artırdığından önemli bir özelliktir. Çeşitlilik elde edilen verilerin farklı kaynaklardan sağlanmasını ifade eder. Örneğin e-posta ile gelen, düzenli üretilen pdf raporlarından gelen, kullanıcı girişlerinden elde edilen veriler. Özetle “bir düzende olan” veya “bir düzende” olmayan veri çeşitliliğini ifade eder.
3. **Hız** (Velocity)  
   Yavaş bir hızla büyük veri elde edilemez. Bu yüzden verinin çok hızlı ve sürekli elde edilmesi gerekir. Aynı mantıkla; üretilen veri eğer aynı hızla işlenemezse yine bir anlam ifade etmez. Bu yüzden büyük veri hızlı büyüyen ve hızlı işlenen veri olmalıdır.
4. **Doğruluk** (Veracity)  
   Üretilen verinin kirli ve bozulmamış olması gerekir. Gereksiz ve değişime uğramış büyük verileri analiz etmek ciddi vakit kaybına ve hatalı sonuçlara yol açabilir. Bu sebeple verinin doğruluğu kriterler içinde en hassas olanların biridir.
5. **Geçerlilik** (Validity)  
   Geçerlilik, doğruluğa benzer bir madde. Elde edilen verinin geçerliliğini yitirmemiş olması, anlamlı ve doğru olması gerekliliğini ifade eder.

BIG DATA NE İŞE YARAR?

Teknolojide lider bir çok ülke büyük veri (big data) analizi yapmaktadır. Savunmadan seçim analizlerine, uyuşturucuyla mücadeleye kadar önemli alanlarda kullanılmaktadırlar.

**BIG DATA PROJELERİ**

Tamamı açık kaynak kodlu projeler:

• Hadoop

• HDFS

• MapReduce

• Pig

• Hive

• HBase

• Mahout

## **Big Data Tools Overview**

#### **Apache Hadoop**

Hadoop is an open source software framework originally developed by Doug Cutting and Mike Cafarella in 2006. It was specifically built to handle very large data sets. Hadoop is made up of two main parts: the Hadoop Distributed File System (HDFS) and MapReduce. HDFS is the storage component of Hadoop. Hadoop stores data by splitting files into large blocks and distributing it across nodes. MapReduce is the processing engine of Hadoop. Hadoop processes data by delivering code to nodes to process in parallel.

#### **Apache Spark**

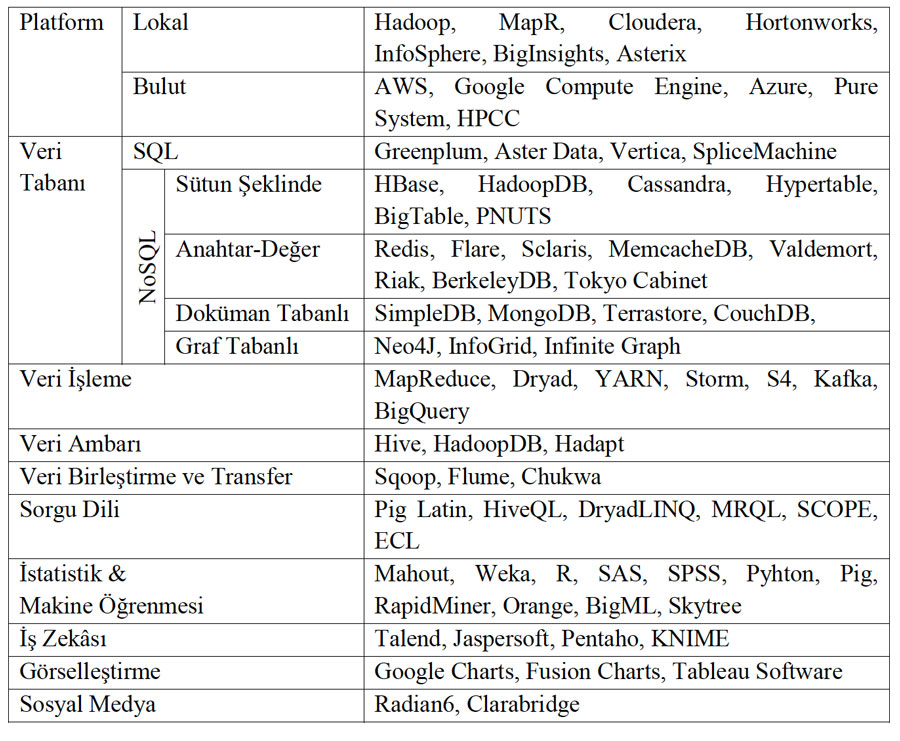
Apache Spark is quickly growing as a data analytics tool. It is an open source framework for cluster computing. Spark is frequently used as an alternate to Hadoop’s MapReduce because it is able to analyze data up to 100 times faster for certain applications. Common use cases for Apache Spark include streaming data, machine learning and interactive analysis.

#### **Apache Hive**

Apache Hive is a SQL-on-Hadoop data processing engine. Apache Hive excels at batch processing of ETL jobs and SQL queries. Hive utilizes a query language called HiveQL. HiveQL is based on SQL, but does not strictly follow the SQL-92 standard.

#### **NoSQL Databases**

NoSQL databases have grown in popularity. These Not Only SQL databases are not bound by traditional schema models allowing them to collect unstructured datasets. The flexibility of NoSQL databases like MongoDB, Cassandra, and HBase make them a popular option for big data analytics.



Örneğin, istatiksel analiz için **SPSS**, **ALTERYX** gibi tool 'lar var .SPSS, sosyal bilimcilerin kullanımı amacıyla hazırlanmış bir istatistik programıdır. SPSS, görünüm bakımından Excel’e çok benzemektedir. Ancak, Excel ile yapılamayan bazı analizlere sahip olduğundan dolayı bilimsel çalışmalarda yoğun bir biçimde kullanılmaktadır ve oldukça popülerdir. Akademik çalışmalarda SPSS çoğunlukla anket içeren çalışmalarda katılımcıların demografik özelliklerinin tanımlanması, frekans ve yüzde hesaplamaları, güvenirlik, korelasyon, regresyon ve farklılık analizlerinde kullanılmaktadır.

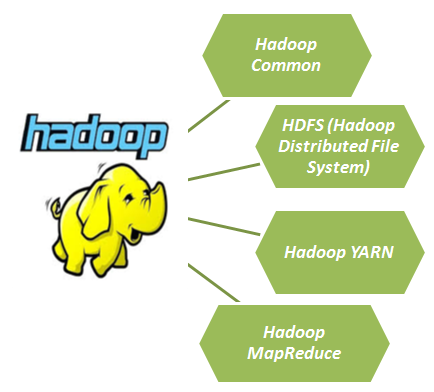
**Hadoop Nedir ?**

Hadoop , büyük veri kümeleri ile birden fazla makinada paralel olarak işlem yapmamızı sağlayan Java ile yazılmış açık kaynak kodlu kütüphanedir . Hadoop büyük verileri birden fazla makinada saklar ve yönetir .  Hadoop, **HDFS** dağıtık dosya sistemi ve **MapReduce** tekniği ile, sıradan sunuculardan oluşan sunucu tarlalarından büyük, tek bir sanal disk oluşturmayı ve bu disk üzerinde paralel operasyonlar gerçekleştirebilmeyi sağlar. Bu sayede Hadoop ile çok büyük boyutta bir çok dosya verimli olarak depolanabilir ve ekstra maliyetli süper güçlü sunuculara gereksinim olmadan paralel işlenebilir.

**Hadoop’un Avantajları**

* Yüksek ölçeklenebilirlik(scalabilitiy)
* Az kaynak israfı ve düşük maliyetli donanım
* Hata toleransı (Fault-tolerance)
* Veriyi taşımak yerine işlemi taşır.

**Hadoop Temel Bileşenleri**



**Hadoop Common**

Bazı modüllerin Hadoop’a erişebilmesi için gerekli olan kütüphaneleri sağlar.Mesela Hive yada Hbase HDFS’e erişmek için bu kütüphaneleri kullanır

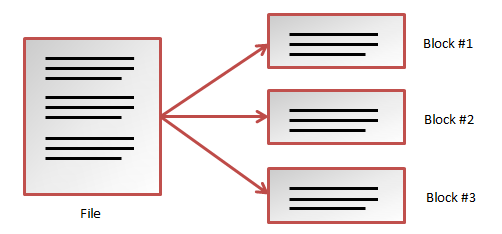
**Hadoop YARN**

Hadoop YARN genel olarak çalışan uygulamalara ne kadar ram,cpu ayrılacağını hesaplar ve yönetir.

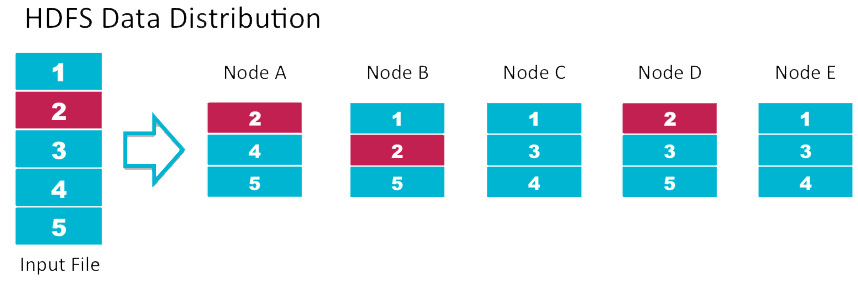
**HDFS Nedir ?**

Bir ağ üzerinden farklı makinalardaki depolama birimlerinin yönetildiği dosya sistemlerine dağıtık dosya sistemleri denir. Apache Hadoop, açılımı **Hadoop Dağıtılmış Dosya Sistemi** olan HDFS olarak kısaltılan bir dağıtılmış dosya sistemi ile birlikte gelmektedir. Hadoop içerisinde büyük verileri sakladığımız bileşene **HDFS** (Hadoop Distributed File System) denir .

Dağıtık dosya sistemleri ağ tabanlı oldukları için, yönetimlerinde karşılaşılan en büyük zorluklardan biri, veri kaybına uğramadan düğüm hatasını tolere etmektir. Büyük verileri HDFS sistemine yüklediğimiz zaman  , Hadoop bu verileri bloklara ayırır .



Farklı bloklara ayrılan veriler Hadoop Cluster üzerinde farklı node lara dağılır . en önemli husulardan bir tanesi her bir blok **çoklanarak** kaydedilir. Bunun asıl  nedeni ise node lardan bir tanesi zarar gördüğünde veya sistemden çıktıgında veri kaybının yaşanmasını engellemektir.



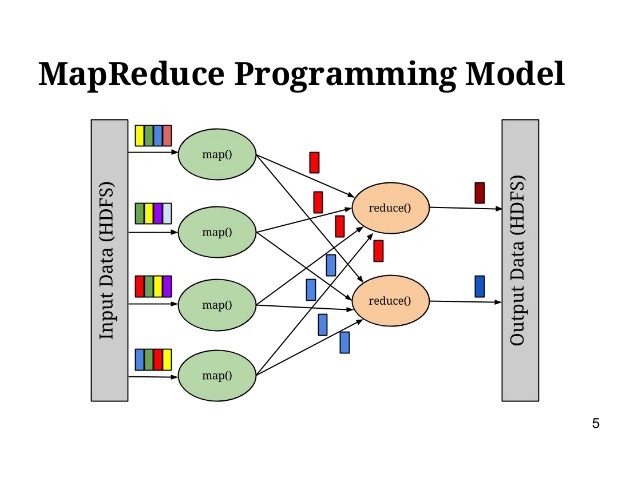
**Hadoop** teknik anlamda bu bileşenlerle işlemektedir.

* Hadoop Common, dev bir kütüphane ve yardımcı Hadoop modüllerinden oluşan büyük veri işlemesine yardımcı kaynaklar
* HDFS, büyük veri kapsamına giren veri yığınlarını işlerken veri kümeleri oluşturmak için kullanılan dağıtılmış veri sistemi
* Hadoop Yarn, işlenecek verilerin üzerinde kaynak yönetimi, uygulama kullanımı ve kişisel ayarlamalar için veri depolama adına yaratılmış olan kaynak yönetim platformu
* Hadoop MapReduce, büyük veri işlemeleri için kullanılan programlama sistemi

## **Hadoop MapReduce**

**Hadoop Mapraduce** büyük veri işlemek için ihtiyacımız olan demirbaşlardan birisidir. Temel anlamda Map ve Reduce bu sistemi oluşturmaktadır. Kullanacağımız depolanmış veriyi filtreleyen map fonksiyonu ve işlediğimiz verilerin analizini sunan Reduce fonksiyonu birleştiğinde **Hadoop MapReduce** ortaya çıkmıştır.

İlişkisel veritabanı sistemleri ile (RDBMS) karşılaştırmak gerekirse, Map aşaması select cümlesi ile ilgili alanları seçmemize ve where ile ilgili sınırlamaları yapmaya, Reduce aşaması ise count, sum, having gibi ilgili veri üzerinde hesaplama yapmamıza benzetilebilir.



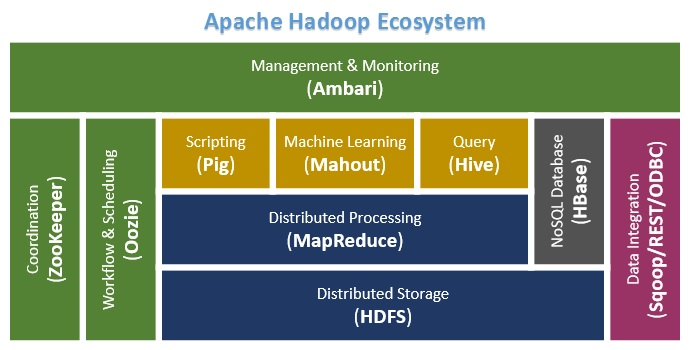
Hadoop Veri İşleme Örneği…

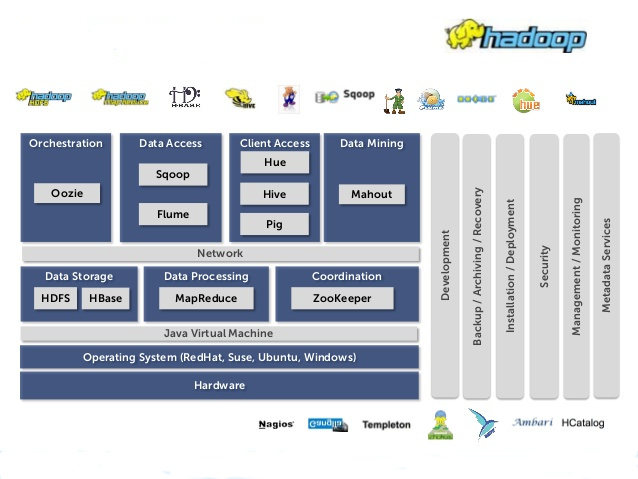
Bir text dosyasının içerisindeki kelime sayısını bulan MapReduce programını inceleyelim.MapReduce şu adımlardan oluşacaktır.



* Spliting : Veriler 64 MB lık bloklara ayrılır . Bu değer değiştirilebilir
* Mapping :  Burada her bir kelime key(word) ve value(1) şeklinde bölümlere ayrılır.
* Shuffling : Map işleminden çıkan sonuçları Reducer a yönlendirir . Amacımız word-count uygulaması oldugu için aynı kelime grubu aynı Reducer a yönlendirilir .
* Reducing : Gelen sonuçlar üzerinden toplama işlemi yapılır ve sonuçlar istediğiniz kaynaklara yazılır (HDFS , SQL , NoSQL)

Apache Hadoop Ecosystem Mimarisi





**Hadoop Ekosistemi**

**Avro:** RPC çağrıları için verilerin kolay taşınmasını sağlayan bir serileştirme sistemidir.

**MapReduce:** Ortak makinelerde, dağıtık bir mimari ile modelleme ve çalıştırma mimarisidir. MapReduce programı Java,[Apache Pig](http://www.buyukveri.co/apache-pig-nedir/) veya Apache Hive ile geliştirilebilir

**HDFS:** Ortak makinelerde, dağıtık bir mimariye sahip dosya sistemidir.

**Pig:** Veri akışını kontrol eden ve büyük boyutlu verilerde işlem yapmayı kolaylaştıran bir araçtır. HDFS ve MapReduce kümelerinde çalışır.

**Hive:** Dağıtık yapılı bir veri ambarıdır. SQL benzeri bir dil sağlar. Hadoop üzerinde veri sorgulama ve dwh  uygulamaları geliştirmeyi sağlar.

**Hbase:** Dağıtık temelli, kolon bazlı bir veritabanıdır. HBase supports random reads and also batch computations using MapReduce.

**Zookeeeper:** İşlerin sürekli çalışır halde kalmasını sağlayan bir takip sistemidir.

**Sqoop:** Klasik veritabanı ile HDFS arasında toplu olarak veri transferi yapılmasını sağlayan bir araçtır.

**Oozie:** Bütün işleri zamanlamak için kullanılan bir araçtır.

**AMBARI**

A Hadoop component, Ambari is a RESTful API which provides easy to use web user interface for Hadoop management. Ambari provides step-by-step wizard for installing Hadoop ecosystem services. It is equipped with central management to start, stop and re-configure Hadoop services and it facilitates the metrics collection, alert framework, which can monitor the health status of the Hadoop cluster.

**MAHOUT**

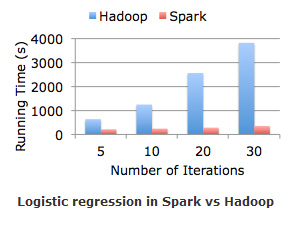
Mahout is an important Hadoop component for machine learning, this provides implementation of various machine learning algorithms. This Hadoop component helps with considering user behavior in providing suggestions, categorizing the items to its respective group, classifying items based on the categorization and supporting in implementation group mining or itemset mining, to determine items which appear in group. İstatistik ilkelerini kullanarak, Machine Learning uygulamaları sistemlere verilerden öğrenmeyi ve gelecekteki davranışı belirlemek için geçmiş çıktıları kullanmayı öğretir.

Düşük maliyet ile yüksek derecede ölçeklenebilirlik sağlayan Hadoop projesi geleceğin veri işleme platformu olarak adlandırılmakta ve aşağıda bir kısmı belirtilen birçok büyük firma tarafından desteklenmektedir:

1. [**Amazon**: Amazon Elastic MapReduce altyapısının](https://aws.amazon.com/documentation/elastic-mapreduce/) sağladığı esneklik sayesinde Hadoop kullanmaya yeni başlayacaklar ya da ihtiyaçları anlık olarak değişebilecek.
2. [**Cloudera**](http://www.cloudera.com/)**:** Açık kaynaklı olan projeyi kurumsal altyapılara uygun hale getirmeye odaklanmıştır.
3. **EMC**: EMC Greenplum HD ürünü ile tek bir yazılım platformu üzerinde ilişkisel veritabanı sistemini ve Hadoop’un gücünü birleştirmektedir.
4. [**Hortonworks**](https://en.wikipedia.org/wiki/Hortonworks):Yahoo tarafından kurulmuş, Hadoop’un açık kaynaklı gelişmesine odaklanmış, bağımsız bir firmadır.
5. **Hadapt**: Apache Hive projesinde olduğu gibi SQL arayüzünü kullanarak ilişkisel veritabanı ve Hadoop sistemini bulut ortamında birleştirmektedir.
6. [**IBM:** InfoSphere BigInsights](http://www.ibm.com/developerworks/data/library/techarticle/dm-1110biginsightsintro/) ürününde Apache Hadoop sistemini Apache Pig analiz dili ile beraber DB2 veri tabanı desteğini bir araya getiriyor.
7. [**Microsoft:** Azure](https://azure.microsoft.com/tr-tr/)bulut sistemlerine beta olarak Hadoop’u dâhil etmiştir.
8. [**Oracle:** büyük veri işleme konusunda kendisine en iyi alternatif olan Hadoop projesini](http://www.oracle.com/technetwork/articles/bigdata/hadoop-optimal-store-big-data-2188939.html)kendi ürünleri ile çok iyi şekilde entegre ederek bunu bir avantaja dönüştürmektedir.

# Apache Spark Nedir ?

Spark genel kullanıma açık bir veri işleme motorudur. Genel olarak Spark ı , MapReduce alternatifi olarak kullanabiliriz . Hadoop kullanarak verileri yine HDFS de saklayabiliriz fakat Apache Spark ile bu verileri daha **kolay** ve daha **hızlı** bir biçimde işleyebiliriz. Sunmuş olduğu API ve araçlar veri bilimciler ve uygulama geliştiricilerin kendi uygulamalarına Spark’ı entegre etmelerine olanak tanımaktadır. Böylece veriyi hızlı biçimde sorgulamak, analiz etmek ve dönüştürmek ölçümlenebilir şekilde mümkündür. Spark’ın esnekliği onu neredeyse kullanım durumlarında karşılaşılan tüm problemleri çözmede uygun bir araç olmasını ve Spark’ın petabyte’larca bilgiyi fiziksel ve sanal sunuculardan oluşan bir küme üzerinde dağıtık biçimde işleyebilmesine olanak sağlıyor.



* Java  , Scala , Python ile geliştirilebilir .
* Spark Streaming ile verileri anlık olarak işlenebilir.
* Spark SQL  , DataFrame gibi yapılarla büyük veriler üzerinde SQL tabanlı analizler yapılabilir .
* Farklı kaynaklarda tutulan büyük verileri analiz edebilirsiniz .( HDFS, Kafka,Cassandra, HBase, S3 … )

SPARK İLE VERİ SORGULAMA

import com.devveri.spark.model.ExchangeRecord;

import com.devveri.spark.util.SparkHelper;

import org.apache.spark.api.java.JavaRDD;

import org.apache.spark.api.java.JavaSparkContext;

import org.apache.spark.api.java.function.Function;

import org.apache.spark.sql.api.java.JavaSQLContext;

import org.apache.spark.sql.api.java.JavaSchemaRDD;

import org.apache.spark.sql.api.java.Row;

import java.util.List;

public class NYSEExample {

    public static void main(String[] args) throws Exception {

        final String appName = "NYSEExample " + System.currentTimeMillis();

        final String stockFile = "[file:///Users/devveri/dev/test/NYSE-2000-2001.tsv](file:///\\Users\devveri\dev\test\NYSE-2000-2001.tsv)";

**// get contexts**

        JavaSparkContext jsc = new JavaSparkContext(SparkHelper.getDefaultConf());

        JavaSQLContext sqlContext = new JavaSQLContext(jsc);

**// load data and convert to java object**

        JavaRDD exchanges = jsc.textFile(stockFile).map(new Function<String, ExchangeRecord>() {

            public ExchangeRecord call(String s) throws Exception {

                return new ExchangeRecord(s);

            }

        });

**// apply schema**

        JavaSchemaRDD schemaExchange = sqlContext.applySchema(exchanges, ExchangeRecord.class);

        schemaExchange.registerAsTable("exchanges");

**// execute an sql query**

        JavaSchemaRDD topExchanges = sqlContext.sql(

                "select stockSymbol, sum(stockPriceClose) as total\_value " +

                "from exchanges " +

                "where stockVolume > 1000 " +

                "group by stockSymbol " +

                "order by total\_value desc " +

                "limit 10");

        List result = topExchanges.map(new Function<Row, String>() {

            public String call(Row row) throws Exception {

                return String.format("%s\t%f",

                        row.getString(0),

                        row.getDouble(1));

            }

        }).collect();

        for (String s : result) {

            System.out.println(s);

        }

    }

}

SPARK İLE BIG DATA ANALİZİ (with java)

public class App

{

    public static void main( String[] args ) throws FileNotFoundException

    {

      SparkConf conf = new SparkConf().setAppName("FirstApplication");

conf.setMaster("local[\*]");

JavaSparkContext sContext = new JavaSparkContext(conf);

SQLContext sqlContext = new  SQLContext(sContext);

List dataList = new ArrayList();

Scanner sc = new Scanner(new File("C:\\Users\\serkans\\Desktop\\temp\\kddcup.data\_10\_percent\\testData"));

while(sc.hasNext()){

     String line = sc.nextLine();

     String[] list = line.split(",");

     Row row = RowFactory.create(list[0],list[1]);

     DataList.add(row);

}

List fields = new ArrayList();

fields.add(DataTypes.createStructField("ip", DataTypes.StringType, true));

fields.add(DataTypes.createStructField("src\_bytes", DataTypes.StringType, true));

StructType schema = DataTypes.createStructType(fields);

DataFrame contentDF = sqlContext.createDataFrame(dataList, schema).cache();

contentDF.registerTempTable("contentDF");

contentDF.printSchema();

contentDF.show();

|  |  |
| --- | --- |
|  | DataFrame ipGroupDF =  sqlContext.sql("select ip,count(1) as count from contentDF group by ip");  ipGroupDF.show(); |

    }

}

Data Scientist (Veri Bilimci)

Data Scientist, bağlı bulunduğu şirket yada kuruluşlar için değer yaratmak amacıyla büyük ve çeşitliliğe sahip veriyi uçtan uca analiz eder. Data Scientist bu verileri analiz ederek, sonuçlarını yönetim ile paylaşıp aksiyon alınmasını sağlar ve “Veri Odaklı Uygulamalar” geliştirir.  Data Scientist çok farklı kaynaklardan, daha fazla veri toplar ve bu veriyi işlemek için yeni araçlar geliştirir. Çoğunlukla mevcut araçların işleyemediği büyük miktarda, yapılandırılmamış ve kirli veriyle çalışır.

**Data scientistler ne iş yaparlar ?**

Şirket /Müşteri sorularına cevap bulurlar.

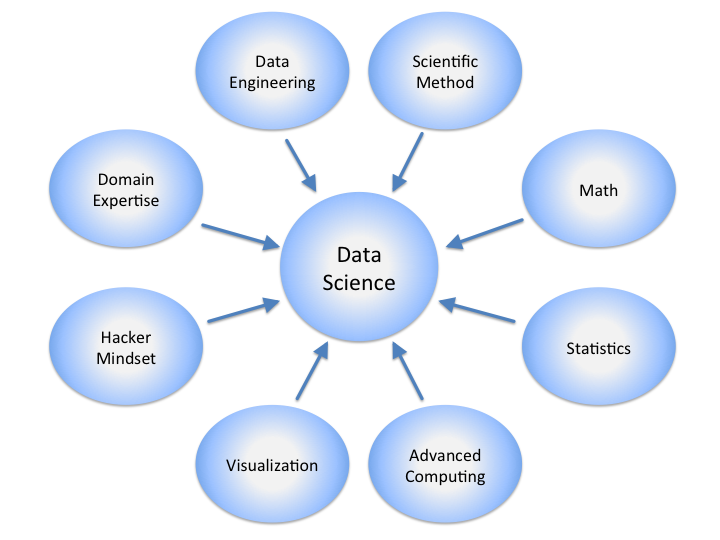
En uygun veriyi bulurlar,ortaya çıkarırlar.

Veriye nasıl ulaşacağımıza karar verirler.

Veriye elde edip temiz bir hale getirirler.

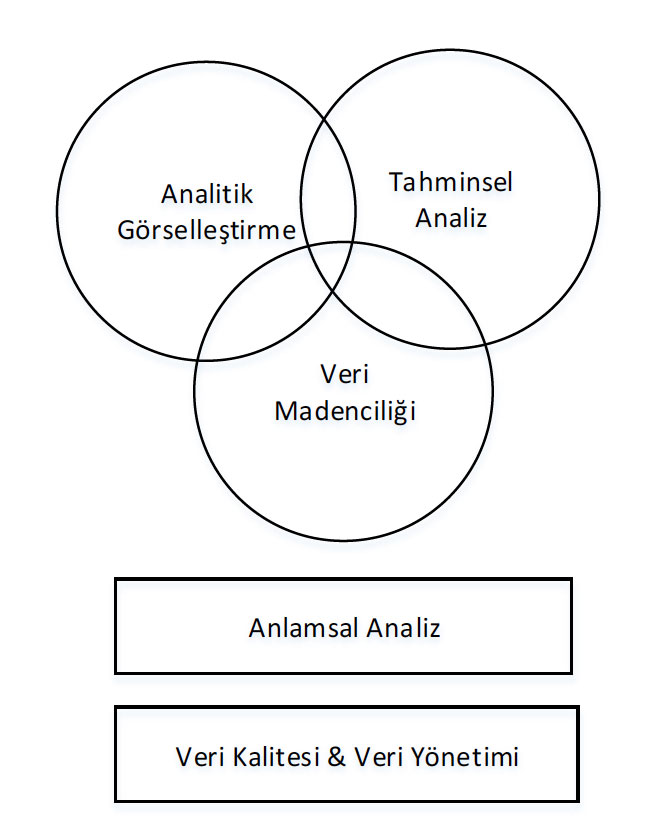
Veriyi detaylı analiz ederler.

Istatistiksel model ve tahmin oluştururlar.



BIG DATA ANALİZİ

Büyük veri analitiğinde kullanılan araçlar aşağıda sınırlandırılmıştır.



Analitik görselleştirme, verinin altında yatan detaylara erişerek hedef kitleye cevapları sunar. Veri madenciliği, gizli örüntüleri ortaya çıkarır. Tahminsel analiz, görselleştirme ve madencilik algoritmaları ve araçları sayesinde öngörülerde bulunmayı sağlar. Anlamsal analiz, yapısal olmayan verinin çözümleme, özünü elde etme ve analiz süreçlerini kapsamaktadır. Veri kalitesi ve veri yönetimi ise, kalite ve yönetim süreçlerinin tanımlandığı denetim uygulamaları ve kurumsal işlemleri içeren teknolojik araçlarla yapılır.

Yukardaki görselde kategorileştirildiği gibi büyük veri araçları genellikle açık kaynaklı olmakla beraber, dağıtık dosya sistemleri, paralel hesaplama algoritmaları veya NoSQL (Not Only SQL) veri tabanları kullanmaktadırlar.

**KODLAMA KISMI**

**HDFS komutları**

cat: dosya içeriğini ekrana basar.  
chgrp: dosya grup ilişkilerini değiştirir.

chmod: dosya yetkilerini değiştirir.

chown: dosya sahipliklerini değiştirir.

copyFromLocal: yerelden hdfs içerisine dosya kopyalar.

copyToLocal: hdfs içerisinden yerel dizine dosya kopyalar.

cp: hdfs içinde dosya kopyalar.  
du: dizin ya da dosya boyutunu listeler.  
dus: dizin ya da dosya boyunutu özet olarak listeler.  
expunge: çöp kutusunu boşaltır.

get: yerel dizine dosya kopyalar.  
getmerge: kaynak dizindeki tüm dosyaları birleştirip tek bir dosyada birleştirir.  
ls: dosya ve dizin listeler.  
lsr: recursive olarak dosya ve dizinleri listeler.  
mkdir: dizin oluşturur.  
movefromLocal: yerelden hdfs içerisine dosya taşır.  
mv: dosya taşır.  
put: yerelden hdfs içerisine dosya taşır.  
rm: dosya siler.  
rmr: silme işlemini recursive olarak yapar.  
setrep: dosya replikasyon değerini değiştirir.  
stat: dosya sistemi yolunun istatistik bilgilerini verir.  
tail: bir dosyanın en son bölümünü ekrana basar.  
test: dosya ve dizinler için kontrol işlemlerini gerçekleştirir.  
text: bir kaynağı okuyup çıktısını text formatında verir.  
touchz: içi boş dosya oluşturur.