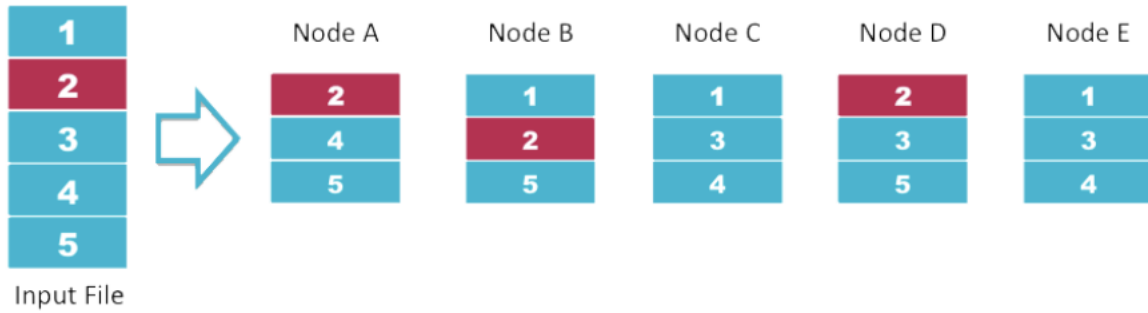


## Hadoop Nedir ?

Hadoop, büyük veri kümeleri ile birden fazla makinede paralel işlem yapma imkanı sağlayan Java tabanlı açık kaynak kodlu bir kütüphanedir.

## Hadoop Büyük Verileri Nasıl Saklar ? HDFS Nedir ?

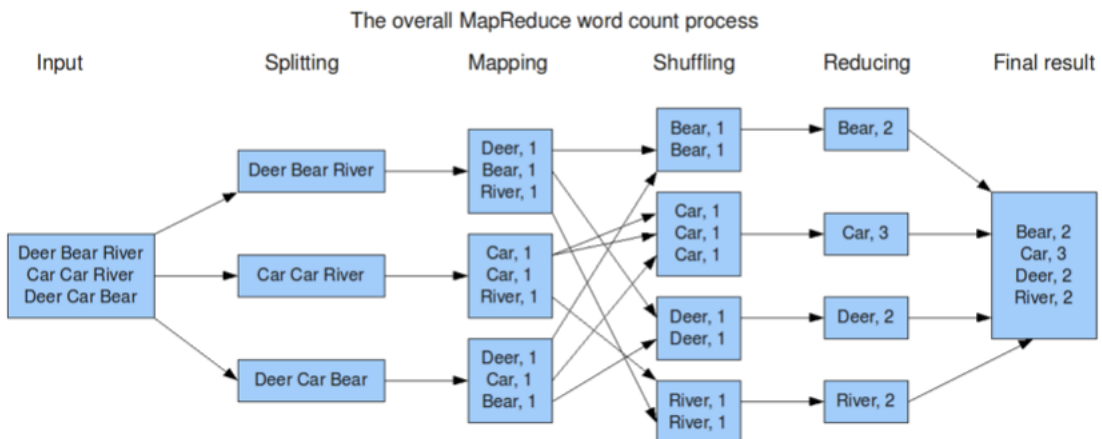
Hadoop kendi içerisinde kullandığı bir dosya sistemi ile beraber gelmektedir. Bu dosya sistemi HDFS(Hadoop Distributed File System)'dir. HDFS üzerine yüklenen veri bloklara ayrılır ve farklı bloklar Hadoop Cluster üzerinde farklı nodelara dağıtılır.



Yukarıda görüldüğü gibi input dosyamız bloklara ayrılmış ve her bir blok birden fazla node üzerinde olacak şekilde dağıtılmıştır. Burada bir bloğun birden fazla node'a gönderilmesinin nedeni nodelardan bir tanesi zarar gördüğünde veya sistemden çıktığında veri kaybının yaşanmasını engellemektir.

## Hadoop Verileri Paralel Olarak Nasıl İşler ? MapReduce Nedir ?

MapReduce, büyük veriyi büyük kümelerde (clusters) işlemek üzere geliştirilmiş bir dağıtık programlama modelidir. Genel yapısı itibarıyla MapReduce adından da anlaşılacağı üzere iki ana fonksiyondan oluşur. Birincisi map; bir yığının tüm üyelerini sahip olduğu fonksiyon ile işleyip bir sonuç listesi döndürür. İkincisi reduce; paralel bir şekilde çalışan iki veya daha fazla map fonksiyonundan dönen sonuçları harmanlar ve çözer. Hem map() hem de reduce() paralel bir şekilde çalışırlar, aynı anda aynı sistemde olmaları gerekmez.



Örnek olarak yukarıdaki şemayı inceleyelim. Bu bir Word-Count örneğidir. 6 aşamaya ayrılan örneğin aşamaları şu şekildedir:

- Birinci Aşama: Input dosyasının oluşturulmasıdır.
- İkinci Aşama(Splitting): Oluşturulan Input dosyasının parçalara ayrılması.
- Üçüncü Aşama(Mapping): Burada her bir parçanın hangi kelimeyi ne kadar içerdiği hesaplanır.
- Dördüncü Aşama(Shuffling): Map işleminden çıkan sonuçları Reducer'a yönlendirir. Amacımız word-count uygulaması olduğu için aynı kelime grubu aynı Reducer'a yönlendirilir.
- Beşinci Aşama(Reducing): : Gelen sonuçlar üzerinden toplama işlemi yapılır ve sonuçlar istediğiniz kaynaklara yazılır (HDFS , SQL , NoSQL)

Bu uygulama üstünden devam edecek olursak Mapping paralelliği başlatan ve çalıştıran yapıdır. Reducing aşaması ise bu paralel işlemi devam ettiren ve sonuçları oluşturan aşamadır. Tanımda da geçtiği gibi bu paralellik aynı sistemde de sağlanabilir yada birden fazla sistemde de bu paralel işlemler yapılabilir.

---

## Kurulum

(Kurulum Ubuntu üzerinden anlatılmaktadır.)

Kurulum işlemi 3 aşamada tamamlanmaktadır. Bunlar jdk kurulumu, hadoop için root yetkisi olmayan bir kullanıcının oluşturulması ve hadoop'un kurulması şeklindedir.

### 1-) JDK kurulumu

- Öncelikle sistem güncellenir

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

```
~$ sudo apt install openjdk-8-jdk openjdk-8-jre
```

- Javanın doğru bir şekilde kurulduğunu görmek için

```
~$ java -version
```

- Sistemimizde Javanın doğru çalışması için yapmamız gereken son şey JAVA\_HOME path'ini export etmemiz gerekir. Bunun için

```
~$ sudo nano /etc/profile
```

Açılan dosyanın en altına aşağıdaki satırı ekleyin.

```
~$ export JAVA_HOME=/usr
```

Sonuç olarak dosyamız yandaki gibi olacaktır. CTRL + X ile çıkış yapalım ve eklediğimiz yeni export'a göre dosyamızı güncelleyelim

```
~$ source /etc/profile
```

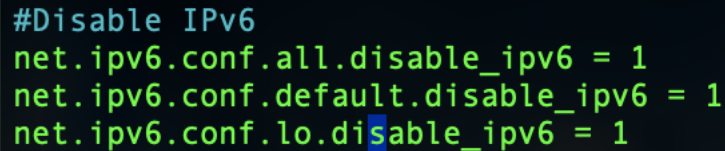
```
if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
unset i
fi
export JAVA_HOME=/usr
```

- Hadoop IPv4 ile çalıştığından sistemimizde IPv6 disable edilmesi gerekiyor.

~\$ **sudo nano /etc/sysctl.conf**

Açılan dosyanın sonuna aşağıdaki satırları ekleyelim.

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```



```
#Disable IPv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

Dosyanın son hali yukarıdaki gibi olmalıdır.

- Bu aşamada sistemimizi reboot etmemiz gerekmektedir.

~\$ **sudo reboot**

2-) Root Yetkisi olmayan bir kullanıcının oluşturulması (Bu aşama opsiyoneldir tamamen güvenlik amacıyla yaptığımız bir aşama)

- Yeni bir grup oluşturuyoruz

~\$ **sudo addgroup hadoopgroup**

- Yeni oluşturduğumuz grup içerisine hduser adında bir kullanıcı ekliyoruz

~\$ **sudo adduser -ingroup hadoopgroup hduser**

- Yeni oluşturduğumuz kullanıcıya geçiş yapıyoruz

~\$ **su - hduser**

- Bu kullanıcı için hadoop'un ssh ile bağlantı yapacağı sırada kullanacağı key çiftini oluşturuyoruz

~\$ **ssh-keygen -t rsa -P ""**

~\$ **cat /home/hduser/.ssh/id\_rsa.pub >> /home/hduser/.ssh/authorized\_keys**

~\$ **cd .ssh/**

~/ssh\$ **chmod 600 ./authorized\_keys**

~/ssh\$ **ssh-copy-id -i /home/hduser/.ssh/id\_rsa.pub localhost**

- Şimdi localhost'a ssh ile bağlantı testi yapıyoruz

~/ssh\$ **ssh localhost**

Bağlantının doğru olduğunu gördükten sonra çıkış yapıyoruz

~\$ **exit**

### 3-) Hadoop Kurulumu

- Öncelikle hadoop .tar dosyasını indirmemiz gerekiyor. Bu aşamayı mümkünse home dizininde gerçekleştirin ben home dizininde yaptığım için komutlarda ona göre olacak kendiniz başka yerde yaparsanız indirmeyi ona göre komutlarda belirtilen pathleri değiştirmeniz gerekmektedir. Ayrıca ben hadoop 2.7.7 sürümünü indirdim eğer başka sürüm indirirseniz ona göre komutlarınızı değiştirmeniz gerekecektir.

~\$ **wget http://ftp.itu.edu.tr/Mirror/Apache/hadoop/common/hadoop-2.7.7/hadoop-2.7.7.tar.gz**

- İndirdiğimiz tar dosyasını extract ediyoruz

~\$ **tar -xvf hadoop-2.7.7.tar.gz**

- Bu işlem biraz sürebilir bittikten sonra tar dosyasını silebilirsiniz zaten extract edilmiş hali elimizde var

~\$ **sudo rm -r hadoop-2.7.7.tar.gz**

— Kurulumun buradaki aşamasında hduser kullanıcısında olan varsa exit komutu ile hduser kullanıcısından tekrar root yetkisi olan kullanıcıya geçiş yapalım.

- Öncelikle hduser kullanıcısının home dizinine gidiyorum

~\$ **cd /home/hduser/**

- Burada extract ettiğimiz dosyayı /usr/local dizinine taşıyorum

/home/hduser\$ **sudo mv ./hadoop-2.7.7 /usr/local/**

/home/hduser\$ **sudo ln -sf /usr/local/hadoop-2.7.7/ /usr/local/hadoop**

/home/hduser\$ **sudo chown -R hduser:hadoopgroup /usr/local/hadoop-2.7.7/**

- hduser kullanıcına geçiş yapıyorum

/home/hduser\$ **su - hduser**

**Bu aşamadan itibaren konfigürasyonları yapıyor olacağız dikkat etmekte fayda var.**

~\$ **nano ~/.bashrc**

- Komutu çalıştırdıktan sonra açılan pencerenin en altına aşağıdaki satırları ekleyelim

```
# Hadoop config
export HADOOP_PREFIX=/usr/local/hadoop
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_MAPRED_HOME=${HADOOP_HOME}
export HADOOP_COMMON_HOME=${HADOOP_HOME}
export HADOOP_HDFS_HOME=${HADOOP_HOME}
export YARN_HOME=${HADOOP_HOME}
export HADOOP_CONF_DIR=${HADOOP_HOME}/etc/hadoop
# Native path
export HADOOP_COMMON_LIB_NATIVE_DIR=${HADOOP_PREFIX}/lib/native
export HADOOP_OPTS="-Djava.library.path=${HADOOP_PREFIX}/lib/native"
# Java path
export JAVA_HOME="/usr"
# OS path
export PATH=$PATH:$HADOOP_HOME/bin:$JAVA_PATH/bin:$HADOOP_HOME/sbin
```

- .bashrc dosyasını güncelliyoruz

~\$ **source ~/.bashrc**

~\$ **nano /usr/local/hadoop/etc/hadoop/hadoop-env.sh**

Açılan pencerenin en sonuna aşağıdaki satırı ekleyin

```
export JAVA_HOME="/usr"
```

- Konfigürasyon dosyalarımızın bulunduğu klasöre gidiyoruz.

~\$ **cd /usr/local/hadoop/etc/hadoop**

- Burada ekleme yapacağımız 4 tane dosya var bu dosyaları tek tek nano ile açıp içlerini burada gösterilen şekilde dolduruyoruz. Bizim için önemli olan ve eklemeleri yapacağımız alanlar configuration tagleri arasındaki alanlardır

---

### core-site.xml

```
<configuration>
<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:9000</value>
</property>
</configuration>
```

---

### hdfs-site.xml

```
<configuration>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>

<property>
  <name>dfs.name.dir</name>
  <value>file:/usr/local/hadoop/hadoopdata/hdfs/namenode</value>
</property>

<property>
  <name>dfs.data.dir</name>
  <value>file:/usr/local/hadoop/hadoopdata/hdfs/datanode</value>
</property>
</configuration>
```

---

### mapred-site.xml

**normalde mapred-site.xml dosyası yoktur mapre-site.xml.template dosyasını açın eklemeleri yapın ve CTRL + X ile çıkış yaparken isminin sonundaki .template eklentisini silin o şekilde kaydedin**

```
<configuration>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
</configuration>
```

---

## yarn-site.xml

```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
</configuration>
```

### Konfigürasyon işlemlerimizi tamamladık :))

- Sırada hdfs nodelarımızı formatlamak var bunun için aşağıdaki komutu çalıştırıyoruz

```
/usr/local/hadoop/etc/hadoop$ hdfs namenode -format
```

- Formatlama işlemi sonrası nodelarımızı çalıştırmak için aşağıdaki komutları çalıştırıyoruz

```
/usr/local/hadoop/etc/hadoop$ start-dfs.sh
```

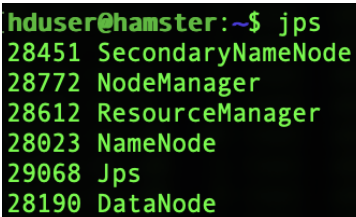
-namenode ve datanode çalıştırılır

```
/usr/local/hadoop/etc/hadoop$ start-yarn.sh
```

-yarn deamon ve resourcemanager çalıştırılır

- Çalışan nodelarımızın neler olduğunu görüntülemek için aşağıdaki komutu çalıştıralım

```
/usr/local/hadoop/etc/hadoop$ jps
```



```
hduser@hamster:~$ jps
28451 SecondaryNameNode
28772 NodeManager
28612 ResourceManager
28023 NameNode
29068 Jps
28190 DataNode
```

**Çıktımız yukarıdaki şekildeki gibi ise tüm kurulumları doğru bir şekilde yaptığımızı görebiliriz.**

---

## WordCount Example

- Home dizininde WordCountTutorial adında bir klasör oluşturalım.

```
~$ mkdir WordCountTutorial
```

- Oluşturduğumuz klasöre girelim

```
~$ cd WordCountTutorial
```

- Java kodumuzu oluşturalım aşağıdaki kodu kopyalayabilirsiniz

```
~$ nano WordCount.java
```

```
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

    public static class TokenizerMapper
        extends Mapper<Object, Text, Text, IntWritable>{

        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(Object key, Text value, Context context
            ) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }

    public static class IntSumReducer
        extends Reducer<Text,IntWritable,Text,IntWritable> {
        private IntWritable result = new IntWritable();

        public void reduce(Text key, Iterable<IntWritable> values,
            Context context
            ) throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }
            result.set(sum);
            context.write(key, result);
        }
    }
}
```



```

}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

- input\_data adında bir klasör oluşturup girelim

```
~/WordCountTutorial$ mkdir input_data
```

```
~/WordCountTutorial$ cd input_data
```

- Inputlarımızı bir text dosyasına yazalım

```
~/WordCountTutorial$ nano input.txt
```

içine rastgele tekrar eden kelimeler yazabilirsiniz.

ÖR:

GSU

Veri

Tabanı

Veri

GSU

- Tekrar WordCountTutorial klasörüne dönelim

```
~/WordCountTutorial/input_data$ cd ..
```

- Oluşturduğumuz WordCount.java dosyasının derlenmesi sonucu oluşacak class dosyaları için tutorial\_classes adında bir klasör oluşturalım

```
~/WordCountTutorial$ mkdir tutorial_classes
```

- HADOOP\_CLASSPATH'i export edelim

```
~/WordCountTutorial$ export HADOOP_CLASSPATH=$(hadoop classpath)
```

~/WordCountTutorial\$ **echo \$HADOOP\_CLASSPATH**

- Yukarıdaki komutu çalıştırdığımızda çıktımız aşağıdaki şekilde olmalıdır.

```
/usr/local/hadoop/etc/hadoop:/usr/local/hadoop/share/hadoop/common/lib/*:/usr/local/hadoop/share/hadoop/common/*:/usr/local/hadoop/share/hadoop/hdfs:/usr/local/hadoop/share/hadoop/hdfs/lib/*:/usr/local/hadoop/share/hadoop/hdfs/*:/usr/local/hadoop/share/hadoop/yarn/lib/*:/usr/local/hadoop/share/hadoop/yarn/*:/usr/local/hadoop/share/hadoop/mapreduce/lib/*:/usr/local/hadoop/share/hadoop/mapreduce/*:/usr/local/hadoop/contrib/capacity-scheduler/*.jar
```

- Şimdi kendi dosya sistemimizde yaptığımız proje yapılandırmasını HDFS üstünde de yapacağız.

~/WordCountTutorial\$ **hadoop fs -mkdir /WordCountTutorial**

~/WordCountTutorial\$ **hadoop fs -mkdir /WordCountTutorial/Input**

~/WordCountTutorial\$ **hadoop fs -put '/home/hduser/WordCountTutorial/input\_data/input.txt' /WordCountTutorial/Input**

- Şimdi yazmış olduğumuz kelime sayan java programını javac komutu ile derleyeceğiz ve çıktıları da tutorial\_classes klasörüne yazdıracağız

~/WordCountTutorial\$ **javac -classpath \${HADOOP\_CLASSPATH} -d '/home/hduser/WordCountTutorial/tutorial\_classes' '/home/hduser/WordCountTutorial/WordCount.java'**

- tutorial\_classes klasörü içerisinde bulunan class dosyalarından bir jar dosyası oluşturuyoruz

~/WordCountTutorial\$ **jar -cvf wordCountExample.jar -C tutorial\_classes/ .**

- Ve bu oluşturulan jar dosyasını hadoop üzerinde çalıştırıyoruz

~/WordCountTutorial\$ **hadoop jar '/home/hduser/WordCountTutorial/wordCountExample.jar' WordCount /WordCountTutorial/Input /WordCountTutorial/Output**

- Sonuçları görmek için aşağıdaki komutu çalıştırabilirsiniz

~/WordCountTutorial\$ **hadoop dfs -cat /WordCountTutorial/Output/\***

Mutlu son :) İyi çalışmalar...

Mustafa DAĞDELEN

