GALATASARAY ÜNİVERSİTESİ
MÜHENDİSLİK VE TEKNOLOJİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

İLERİ VERİ TABANLARI

PROJE RAPORU

HADDOOP, HDFS, HIVE ve HBASE ile COVID Verisi Yükleme ve HiveQl ile Sorgulama

MUHAMMED NURULLAH GÜMÜŞ
BERKAY ÜLKE

İÇİNDEKİLER

I.Proje Nedir? **2**

II.Projenin Amacı Nedir? **2**

III. Docker Ortamı **2**

IV. Git **2**

V. Proje Adımları 2-6

VI. Sonuçlar **7-9**

I.Projenin Konusu Nedir?

Bu proje bir hadoop ortamı uygulamasıdır. Hive ve hbase kullanarak veri yükleme ve sorgulama yapılmıştır.

Hadoop, yalın tabiriyle, sıradan sunucularda büyük verileri işlemek amacıyla kullanılan açık kaynak kodlu bir kütüphanedir. Her türlü veri için devasa depolama, çok yüksek işlem gücü ve neredeyse sınırsız sayıda eş zamanlı görevleri yönetme yeteneği sağlar. Dağınık bir bilgi işlem ortamında büyük verileri verimli bir şekilde yönetmenizi ve işlemenizi mümkün kılar. Hadoop dört ana modülden oluşur.

I.I Hadoop'un Büyük Veri Araçları

Hadoop'un büyük verileri işleme becerisini geliştiren açık kaynaklı birçok araç vardır. Bunlar:
Hive: Hadoop'un dosya sisteminde depolanan büyük veri kümelerini işlemek için veri ambarıdır.
Zookeeper: Monitoring aracıdır.
Hbase: Hadoop için açık kaynaklı bir ilişkisiz veritabanıdır.
Sqoop: Hadoop ve ilişkisel veritabanları arasında veri taşımak için kullanılan bir komut satırı aracıdır.
HDFS: Hadoopun özel dağıtık dosya sistemidir.

II.Projenin Amacı Nedir?

Bu projenin amacı https://github.com/owid/covid-19-data/tree/master/public/data adresindeki covid datasını batch data olarak hive yardımıyla hbase'e yüklemek ve datayı sorgulamaktır. Hive hadoop ortamı gerektirdiği için hadoop ekosistemi kurmak gerekir. Tüm geliştirmeler docker ortamında yapılmıştır. Windows üzerinden wsl ile çalıştırılabilir. ARM mimarisi hariç Docker destekleyen tüm Unix sistemlerinde çalışır. Temel amaç proje dosylarında bulunan covid-data.csv dosyasındaki bilgileri HBASE veri tabanına Hive ile yüklemek ve sorgulamaktır.

III.Docker Ortamı

Docker, uygulamalarınızı hızla derlemenize, test etmenize ve dağıtmanıza imkan tanıyan bir yazılım platformudur. Docker, yazılımları kitaplıklar, sistem araçları, kod ve çalışma zamanı dahil olmak üzere yazılımların çalışması için gerekli her şeyi içeren container adlı standartlaştırılmış birimler halinde paketler. Docker'ı kullanarak her ortama hızla uygulama dağıtıp uygulamaları ölçeklendirilebilir ve kodunuzun çalıştığından emin olabilirsiniz. Docker kurulumu için buradaki adımları takip edebilirsiniz.

IV. Git

Git Teknolojisi: Git bir versiyon kontrol aracıdır. Github ise uygulama versiyonlarının bulunduğu bir repo sitesidir. GitHub projelerimizi özel (private) ve herkese açık (public) olarak ekleyebildiğimiz depolama alanıdır. Gir kurulumu için bu linki takip edebilirsiniz.

V. Proje Adımları:

1. Adım: Öncelikle hadoop,hdfs,hbase ve hive uygulamalrını docker üzerinden kurmak için için terminalden:

$ git clone https://github.com/silverdevelopper/hadoop_hive_hbase_on_docker.git

komutu ile projeyi indirin.

Projenin Dosya yapısı

```
├── Makefile
├── README.md
├── base
│   ├── Dockerfile
│   └── entrypoint.sh
├── covid-data.csv
├── createdata.sql
├── distributed
│   ├── Makefile
│   ├── README.md
│   ├── docker-compose-hadoop.yml
│   ├── docker-compose-hbase.yml
│   ├── docker-compose-zookeeper.yml
│   ├── hadoop.env
│   └── hbase.env
├── docker-compose-standalone.yml
├── docker-compose.yml
├── hadoop-cluster-docker
│   ├── Dockerfile
│   ├── LICENSE
│   ├── README.md
│   ├── build-image.sh
│   ├── config
│   │   ├── core-site.xml
│   │   ├── hadoop-env.sh
│   │   ├── hdfs-site.xml
│   │   ├── mapred-site.xml
│   │   ├── run-wordcount.sh
│   │   ├── slaves
│   │   ├── ssh_config
│   │   ├── start-hadoop.sh
│   │   └── yarn-site.xml
│   ├── hadoop-cluster-docker.png
│   ├── resize-cluster.sh
│   └── start-container.sh
├── hadoop-hive.env
├── hadoop.env
├── hbase-distributed-local.env
├── hbase-standalone.env
├── hive-commands.txt
├── hmaster
│   ├── Dockerfile
│   └── run.sh
├── hregionserver
│   ├── Dockerfile
│   └── run.sh
├── init-network.sh
├── jars
│   ├── hbase-client-1.7.1.jar
│   ├── hbase-hadoop-compat-1.7.1.jar
│   ├── hbase-hadoop2-compat-1.7.1.jar
│   ├── hbase-it-1.7.1.jar
│   └── hbase-server-1.7.1.jar
├── namenode-with-sqoop
│   ├── Dockerfile
│   └── build.sh
└── standalone
    ├── Dockerfile
    └── run.sh
```

2. Adım: Makefile dosyasını çalıştırarak docker konteynırlarını indirip ve Dockerfile dosyalarını build edilmeli. Bu adım projenin root klasöründe yapılmalı.
Terminal ekranında aşağıdaki komutu çalıştırınız.

$ sudo make build

3.Adım: Dockerda network oluşturmak için:

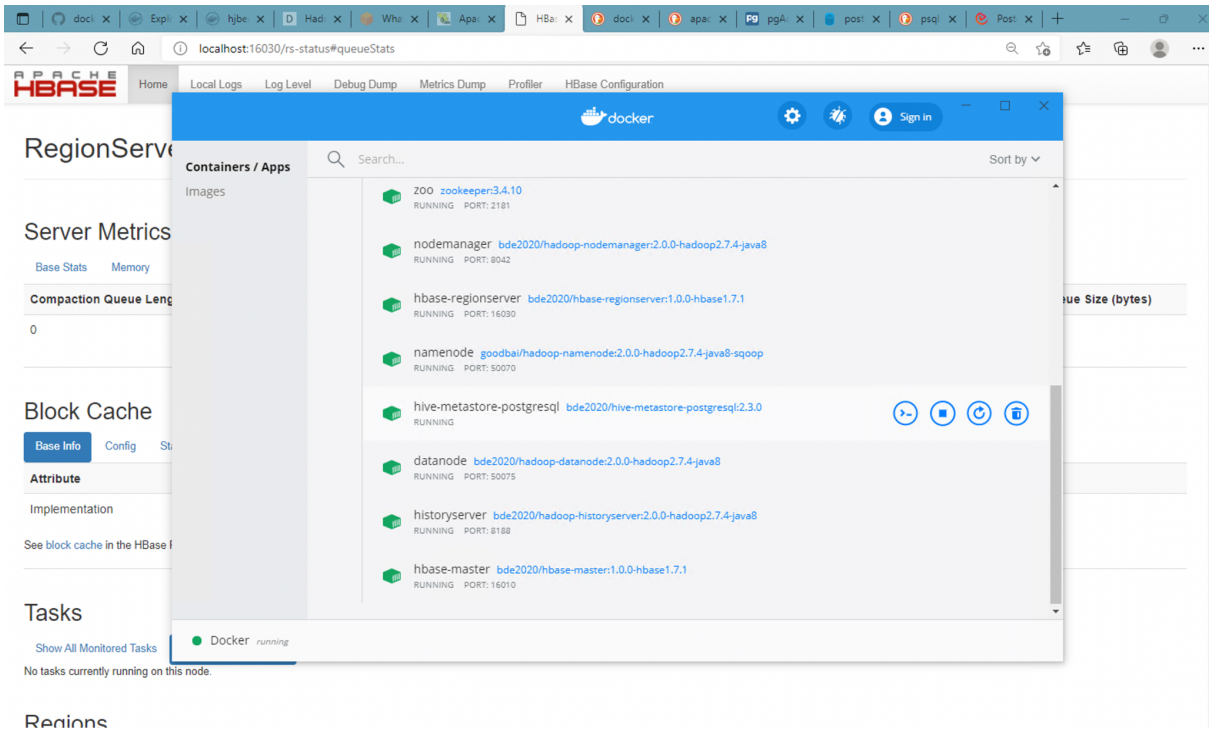
$ docker network create big-data

4.Adım: Makefile dosyası ile oluşturulan docker konteynırları ayağı kaldırılmalı. Bunun için basitçe terminalden:

$ sudo make start

Makefile dosyası proje için gerekli tüm konteynırları indirip, isteğimiz hadoop ekosistemini yaratacaktır.

Bu iki aşamadan sonra Docker Desktop uygulamasında aşağıdaki görseli elde etmelisiniz.



Oluşan konteynırlar:

Zookeeper(Zoo): Dağıtık yapılar için bir koordinatör hizmeti sağlayan uygulamadır. Koordinasyonu sağlamak için birçok görevi yerine getirmesi gerekir ki uyum, işbirliği, ardışık işlemlerin gerçekleşmesini sağlamak bunların en önemlileri.

Nodemenager: HDFS nodlarını yöneten kontetnırdır.

Hbase-regionserver: HBase Mimarisinde bir bölge, o Bölgeye atanan başlangıç anahtarı ile bitiş anahtarı arasındaki tüm satırlardan oluşur. Ve HBase Cluster'daki düğümlere atadığımız bu Bölgeler, “Region Server” olarak adlandırdığımız şeydir.

NameNode: HDFS hizmetiyle ilişkilidir. Namenode, hdfs meta veri bilgilerini tutar.. Namenode yardımı ile hdfs seviyesinde bulunan hdfs dosyasının bilgisini alıyoruz.

Hive-master: Hive uygulamasını master konteynırıdır.

datanode: Hadoop ekosistemindeki hdfs sisteminin bir parçasıdır. Slave node da denir. Bir çok datanode bulunabilir. HDFS işleri bu nodelara dağıtarak yapar.

historyserver: History Server ise bir hadoop bileşenidir. Son çalışan uygulamaların başarılı olup olmadığının bilgisini tutar.

hbase-master: HBase veri tabanının ana uygulamasıdır.

4.Adım:

```
(base) gumus@ngumus-laptop:/mnt/c/Users/gumus/Desktop/docker-compose-hadoop-cluster-with-hbase-hive/docker-compose-hadoop-cluster-with
-hive$ sudo docker ps
CONTAINER ID        IMAGE                                                   COMMAND                  CREATED
       STATUS                     PORTS                                                   NAMES
8557d314c970        bde2020/hadoop-resourcemanager:2.0.0-hadoop2.7.4-java8   "/entrypoint.sh /run…"   11 minutes ago      Up 11 minute
lthy)   0.0.0.0:8088->8088/tcp                          resourcemanager
94dc46c04fc2        bde2020/hive:2.3.2-postgresql-metastore                  "entrypoint.sh /opt/…"   11 minutes ago      Up 11 minute
        10000/tcp, 0.0.0.0:9083->9083/tcp, 10002/tcp   hive-metastore
3adee9b49d80        goodbai/hive:2.3.2-postgresql-metastore-sqoop            "entrypoint.sh /bin/…"   11 minutes ago      Up 11 minute
        0.0.0.0:10000->10000/tcp, 10002/tcp            hive-server
b52c6e839403        zookeeper:3.4.10                                         "/docker-entrypoint.…"   11 minutes ago      Up 11 minute
        2888/tcp, 0.0.0.0:2181->2181/tcp, 3888/tcp     zoo
5117c97ab0f9        bde2020/hadoop-nodemanager:2.0.0-hadoop2.7.4-java8       "/entrypoint.sh /run…"   11 minutes ago      Up 11 minute
lthy)   0.0.0.0:8042->8042/tcp                          nodemanager
ad4a404334dc        bde2020/hbase-regionserver:1.0.0-hbase1.7.1              "/entrypoint.sh /run…"   11 minutes ago      Up 11 minute
        16020/tcp, 0.0.0.0:16030->16030/tcp            hbase-regionserver
c6fe01c01790        goodbai/hadoop-namenode:2.0.0-hadoop2.7.4-java8-sqoop    "/entrypoint.sh /run…"   11 minutes ago      Up 11 minute
lthy)   0.0.0.0:50070->50070/tcp                        namenode
fdbdf252fa2e        bde2020/hive-metastore-postgresql:2.3.0                  "/docker-entrypoint.…"   11 minutes ago      Up 11 minute
        5432/tcp                                       hive-metastore-postgresql
```

5. Adım: Hive konteynırına bağlanmak ve hive çalıştımak için

```
(base) gumus@ngumus-laptop:/mnt/c/Users/gumus/Desktop/docker-compose-hadoop-cluster-with-hbase-hive/docker-compose-hadoop-cluster-with
-hive$ sudo docker exec -it hive-server bash
root@3adee9b49d80:/opt# hive
```

6. Adım: Csv dosyasını copy etmek için

7. Adım: Hbase'de tablo oluşturmak için repodaki creadata.sql kodu hive ile çalıştrılmalı.

```
hive> CREATE TABLE Covid
    >   (iso_code string, continent string, location string, create_date string, total_cases string, new_cases string, new_cases_smoot
ring, total_deaths string, new_deaths string, new_deaths_smoothed string, total_cases_per_million string, new_cases_per_million string
cases_smoothed_per_million string, total_deaths_per_million string, new_deaths_per_million string, new_deaths_smoothed_per_million str
eproduction_rate string, icu_patients string, icu_patients_per_million string, hosp_patients string, hosp_patients_per_million string,
y_icu_admissions string, weekly_icu_admissions_per_million string, weekly_hosp_admissions string, weekly_hosp_admissions_per_million s
 new_tests string, total_tests string, total_tests_per_thousand string, new_tests_per_thousand string, new_tests_smoothed string, new_
smoothed_per_thousand string, positive_rate string, tests_per_case string, tests_units string, total_vaccinations string, people_vacci
string, people_fully_vaccinated string, total_boosters string, new_vaccinations string, new_vaccinations_smoothed string, total_vaccin
_per_hundred string, people_vaccinated_per_hundred string, people_fully_vaccinated_per_hundred string, total_boosters_per_hundred stri
w_vaccinations_smoothed_per_million string, new_people_vaccinated_smoothed string, new_people_vaccinated_smoothed_per_hundred string,
ency_index string, population string, population_density string, median_age string, aged_65_older string, aged_70_older string, gdp_pe
ta string, extreme_poverty string, cardiovasc_death_rate string, diabetes_prevalence string, female_smokers string, male_smokers strin
dwashing_facilities string, hospital_beds_per_thousand string, life_expectancy string, human_development_index string, excess_mortalit
lative_absolute string, excess_mortality_cumulative string, excess_mortality string, excess_mortality_cumulative_per_million string)
    >   row format delimited fields terminated by ',';
```

8.Adım:

$ hadoop fs -put ./covid-data.csv /covid-data.csv

9. Adım: Csv datasını hbasee yüklemek için

```
hive> LOAD DATA LOCAL INPATH '/opt/covid-data.csv' OVERWRITE INTO TABLE Covid;
Loading data to table default.covid
OK
```

10. Adım: HiveQl ile sorgu atmak için

```
hive> Select * from covid where iso_code="TUR";
OK
TUR     Asia    Turkey  2020-03-11      1.0     1.0                                     0.012   0.012
3.15    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-12      1.0     0.0                                     0.012   0.0
3.15    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-13      5.0     4.0                                     0.059   0.047
3.15    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-14      5.0     0.0                                     0.059   0.0
3.15    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-15      6.0     1.0                                     0.071   0.012
3.15    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-16      18.0    12.0    2.571                           0.212   0.141   0.03
2.78    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-17      47.0    29.0    6.714   1.0     1.0     0.143   0.553   0.341   0.079   0.012   0.012   0.002
2.78    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-18      98.0    51.0    13.857  1.0     0.0     0.143   1.152   0.6     0.163   0.012   0.0     0.002
8.33    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-19      192.0   94.0    27.286  3.0     2.0     0.429   2.258   1.105   0.321   0.035   0.024   0.005
8.33    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-20      359.0   167.0   50.571  4.0     1.0     0.571   4.221   1.964   0.595   0.047   0.012   0.007
8.33    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
TUR     Asia    Turkey  2020-03-21      670.0   311.0   95.0    9.0     5.0     1.286   7.878   3.657   1.117   0.106   0.059   0.015
2.04    85042736.0      104.914 31.6    8.153   5.061   25129.341       0.2     171.285 12.13   14.1    41.1            2.81    77.69
```

Resimlerde yazılan komutları çalıştırınız.

VI. Sonuçlar:

Hbase veri tabanı üzerine hive ile atılan sorguların çok kısa sürdüğü görüldü. Normal databselerde çok daha uzun sürmesi beklenen sorgular Hadoop ortamında yüksek performans ile çalışır. İşte bir kaç örnek:

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/hadoop-2.7.4/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.clas
s]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

Logging initialized using configuration in file:/opt/hive/conf/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark,
 tez) or using Hive 1.X releases.
hive> (select avg(cast(people_fully_vaccinated as int)) from Covid)
    > ;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.
e. spark, tez) or using Hive 1.X releases.
Query ID = root_20220128181336_4dfebf88-20a1-49fd-9160-f8ce6acd2580
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2022-01-28 18:13:40,328 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local613415092_0001
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 89409240 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
4.253717484615595E7
Time taken: 3.751 seconds, Fetched: 1 row(s)
hive>
```

```
671
736
444
466
436
658
721
681
679
1
Time taken: 1.393 seconds, Fetched: 239 row(s)
hive> select count(*) from Covid;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.
e. spark, tez) or using Hive 1.X releases.
Query ID = root_20220128181421_43dd0ec2-528f-4ab8-a468-5db0cb3b6f4b
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Job running in-process (local Hadoop)
2022-01-28 18:14:22,815 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1434830489_0003
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 268227720 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
157709
Time taken: 1.337 seconds, Fetched: 1 row(s)
hive>
```