EKONOMICKÁ UNIVERZITA V BRATISLAVE
FAKULTA HOSPODÁRSKEJ INFORMATIKY

SPRACOVANIE A ANALÝZA LOGOV V NADSTAVBE HIVE V SYSTÉME LINUX

Zadanie z predmetu Big Data

Bc. Ondrej Šima

Bc. Alena Stracenská

2023

Obsah

Úvod **4**

1 Nadstavba Hive na Linuxe **5**

1.1 Inštalácia 5

1.2 Inštalácia databázy Apache Derby 6

1.3 Konfigurácia Apache Hive 6

2 Práca s logmi v Hive **9**

2.1 Spracovanie a analýza logov 10

2.1.1 Spracovanie a analýza 3.2 GB súboru 13

Záver **15**

Zoznam použitej literatúry **16**

Zoznam obrázkov a tabuliek

Obrázok 1 Ukážka súboru s logmi, zdroj: [vlastné spracovanie] 9

Obrázok 2 Ukážka vytvorenia tabuľky a načítania dát, zdroj: [vlastné spracovanie] 11

Obrázok 3 Proces vykonávania jobs pri agregácii, zdroj: [vlastné spracovanie] 12

Obrázok 4 Výsledok SQL dopytu agregácie IP adries, zdroj: [vlastné spracovanie] 12

Obrázok 5 Vytvorenie tabuľky a načítanie dát z celého súboru, zdroj: [vlastné spracovanie] 13

Obrázok 6 Proces vykonávania jobs pri agregácii, zdroj: [vlastné spracovanie] 13

Obrázok 7 Výsledok SQL dopytu agregácie IP adries, zdroj: [vlastné spracovanie] 14

Úvod

V tomto zadaní si ukážeme ako správne nainštalovať nadstavbu Hive na operačný systém Linux. Celá inštalácia pozostáva z niekoľkých krokov, ktoré sú síce zdĺhavejšie, ale vo výsledku nám prinesú funkčnú nadstavbu Hadoopu.

V ďalšom kroku si predstavíme použité logy, ich zdroj a štruktúru. Následne si ukážeme ako je možné logy spracovať cez Hive, od niekoľko megabajtového súboru až po niekoľko gigabajtový súbor. Potom si ukážeme agregácie, pomocou ktorých vieme zistiť z logov zaujímavé informácie.

Veríme, že tento dokument bude prínosný pre čitateľa nie len po teoretickej stránke, ale aj po praktickej, kde sa naučí čo všetko inštalácia Hive na Linux obnáša a ako je možné spracovať a analyzovať pomocou tejto popolárnej nadstavby logy.

1 Nadstavba Hive na Linuxe

Keďže z predchádzajúcich zadaní sme disponovali funkčnou inštaláciou Hadoopu, kroky v podobe overenia verzie Javy, jej inštaláciu, konfiguráciu a overenie funkčnosti samotného Hadoopu sme preskočili. Pri práci s Apache Hive pokračujeme na rovnakom systéme z minulého zadania. Pred prácou s Hive si zapneme potrebné Hadoop služby (HDFS a YARN).

1.1 Inštalácia

Verziu Hive sme vybrali podľa nainštalovanej verzie Hadoopu, v našom prípade pre Hadoop 3.3.0 stiahneme Hive verziu 3.4.3. Stiahnutý balík následne rozbalíme pomocou:

```
$ tar zxvf apache-hive-0.14.0-bin.tar.gz
```

Úspešné rozbalenie môžeme overiť existenciou rozbaleného priečinka so súbormi Hive:

```
$ ls | grep hive
apache-hive-3.1.3-bin.tar.gz
archive.zip
```

Priečinok obsahujúci binárne a konfiguračné súbory Hive skopírujeme do /usr/local/hive:

```
# mv apache-hive-3.1.3-bin /usr/local/hive
```

Podobne ako pri inštalácii Hadoopu si vytvoríme systémové premenné pre cesty ku Hive súborom, tie vložíme na koniec súboru bashrc:

```
$ echo "export HIVE_HOME=/usr/local/hive" >> ~/.bashrc
$ echo "export PATH=$PATH:$HIVE_HOME/bin >> ~/.bashrc
$ echo "export CLASSPATH=$CLASSPATH:/usr/local/Hadoop/lib/*:. >> ~/.bashrc
$ echo "export CLASSPATH=$CLASSPATH:/usr/local/hive/lib/*:. >> ~/.bashrc
$ source ~/.bashrc
```

Do priečinka /usr/local/hive sa teraz môžeme zjednodušene dostať pomocou premennej `$HIVE_HOME` a pokračovať konfiguráciou.

1.2 Inštalácia databázy Apache Derby

Pre prácu s Hive potrebujeme tiež nainštalovať databázový server, pomocou ktorého si Hive vytvorí spravuje a číta Metastore databázu. Použiť je možné akúkoľvek SQL databázu, napríklad MySQL alebo MariaDB. V našom prípade sme sa rozhodli použiť databázu Apache Derby.

Po stiahnutí Derby databázy verzie 10.4.2.0 sme rozbalili archív pomocou:

```
$ tar zxvf db-derby-10.4.2.0-bin.tar.gz
```

Priečinok s binárnymi súbormi sme skopírovali do /usr/local:

```
# mv db-derby-10.4.2.0-bin /usr/local/derby
```

Podobne ako v prípade Hive, nastavili sme si potrebné systémové premenné:

```
$ echo "export export DERBY_HOME=/usr/local/derby >> ~/.bashrc
$ echo "export PATH=$PATH:$DERBY_HOME/bin >> ~/.bashrc
$ echo "export CLASSPATH=$CLASSPATH:$DERBY_HOME/lib/derby.jar:$DERBY_HOME/
lib/derbytools.jar >> ~/.bashrc
$ source ~/.bashrc
```

1.3 Konfigurácia Apache Hive

Hive používa podobnú schému podpriečinkov ako Hadoop v predošlom zadaní, konfiguračné súbory nájdeme v priečinku `$HIVE_HOME/conf`. Tu nájdeme predpripravený vzor konfigurácie a shellscript, ktoré sú neaktívne (majú nevhodnú príponu), môžeme ich premenovať alebo skopírovať so správnym názvom:

```
$ cd $HIVE_HOME/conf
$ cp hive-env.sh.template hive-env.sh
$ cp hive-default.xml.template hive-site.xml
```

V súbore hive-env.sh doplníme cestu ku Hadoopu:

```
export HADOOP_HOME=/usr/local/hadoop
```

V konfiguračnom súbore hive-site.xml sme potom vykonali niekoľko zmien. Tieto zmeny sme vykonávali postupne, tak ako sme narážali na dané problémy, kvôli zjednodušeniu procesu opíšeme všetky zmeny naraz.

Vo verzii Hive, ktorý sme stiahli je chyba v konfiguračnom súbore hive-env.sh. Na riadku 3220 je použitý neplatný znak, z dôvodu čoho pravdepodobne dochádza ku zlému čítaniu tohto súboru. Chyba sa nachádza na riadku 3220, textové editory ako napríklad Nano ale Vim tento znak zvýraznia. Ďalej sme do tohto konfiguračného súboru doplnili nasledujúce:

```
   <property>
    <name>system:java.io.tmpdir</name>
    <value>/tmp/hive/java</value>
  </property>
  <property>
    <name>system:user.name</name>
    <value>${user.name}</value>
  </property>
  <property>
   <name>javax.jdo.option.ConnectionURL</name>
   <value>jdbc:derby://localhost:1527/metastore_db;create=true </value>
   <description>JDBC connect string for a JDBC metastore </description>
  </property>
  <property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.apache.derby.jdbc.ClientDriver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>
```

Potom sme v priečinku conf vytvorili súbor s názvom jpox.properties a vložili do neho nasledujúci obsah:

```
javax.jdo.PersistenceManagerFactoryClass =

org.jpox.PersistenceManagerFactoryImpl
org.jpox.autoCreateSchema = false
org.jpox.validateTables = false
org.jpox.validateColumns = false
org.jpox.validateConstraints = false
org.jpox.storeManagerType = rdbms
org.jpox.autoCreateSchema = true
org.jpox.autoStartMechanismMode = checked
org.jpox.transactionIsolation = read_committed
```

```
javax.jdo.option.DetachAllOnCommit = true
javax.jdo.option.NontransactionalRead = true
javax.jdo.option.ConnectionDriverName = org.apache.derby.jdbc.ClientDriver
javax.jdo.option.ConnectionURL = jdbc:derby://hadoop1:1527/metastore_db;create = true
javax.jdo.option.ConnectionUserName = APP
javax.jdo.option.ConnectionPassword = mine
```

Jeden z ďalších problémov, na ktorý sme narazili je inkonzistencia verzií knižnice guava.jar medzi inštaláciou Hive a Hadoop. Zatiaľ čo Hive používal verziu 19.0, Hadoop mal k dispozícii novšiu verziu, 27.0, z dôvodu čoho Hive nefungoval. Tento problém sme vyriešili nahradením staršej verzie v priečinku `$HIVE_HOME/lib` novšou verziou z Hive:

```
$ sudo mv /usr/local/hadoop/share/hadoop/hdfs/lib/guava-27.0-jre.jar
/usr/local/hive/lib/guava-27.0-jre.jar
```

Pred samotnou prácou v Hive si vytvoríme Metastore databázu v ktorej si Hive ukladá informácie, ako schémy, ku samotným dátam v Hadoope. Môžeme tak urobiť pomocou:

```
$ cd $HIVE_HOME
$ hive --service metastore
```

Nakoniec si pre Hive vytvoríme v HDFS priečinky a priradíme im práva:

```
$ $HADOOP_HOME/bin/hadoop fs -mkdir /tmp
$ $HADOOP_HOME/bin/hadoop fs -mkdir -p /user/hive/warehouse
$ $HADOOP_HOME/bin/hadoop fs -chmod g+w /tmp
$ $HADOOP_HOME/bin/hadoop fs -chmod g+w /user/hive/warehouse
```

Funkčnosť inštalácie môžme overiť správnym spustením Hive a úspešným zadaním príkazu, napríklad výpisom všetkých tabuliek:

```
$ cd $HIVE_HOME
$ hive
>show tables;
```

Ak je Hive nainštalovaný správne mal by zobraziť:

```
OK
Time taken: 2.798 seconds
```

Alternatívne, ak plánujeme Hive využívať ako server, môžeme ho spustiť pomocou príkazu:

```
$ hive --service hiveserver2 --hiveconf hive.server2.thrift.port=10000
--hiveconf hive.root.logger=INFO,console
```

Potom sa na Hive môžeme pripojiť napríklad pomocou JDBC konektora z Java alebo Python aplikácie (aktuálna verzia balíčka implementujúceho JDBC konektor pre Python nejde nainštalovať z dôvodu závislosti na už nedostupnej systémovej knižnici libsasl2-dev). Alternatívne môžeme funkčnosť Hive servera overiť pomocou nástroja beeline:

```
$ beeline -u jdbc:hive2://localhost:10000/
```

V prípade funkčného pripojenia sa nám zobrazí hive prompt a bude možné vykonávať príkazy. [1]

2 Práca s logmi v Hive

Dáta, s ktorými sme pracovali boli vo forme apačovských webových logov. Môžeme si ich stiahnuť na kaggle.com. Pre naše potreby sme po stiahnutí dát vytiahli 100 000 riadkov, celý súbor bude spracovaný v kapitole 2.1.1. Sú to pološtruktúrované dáta znázornené na obrázku 1.

Obrázok 1: Ukážka súboru s logmi, zdroj: [vlastné spracovanie]

2.1 Spracovanie a analýza logov

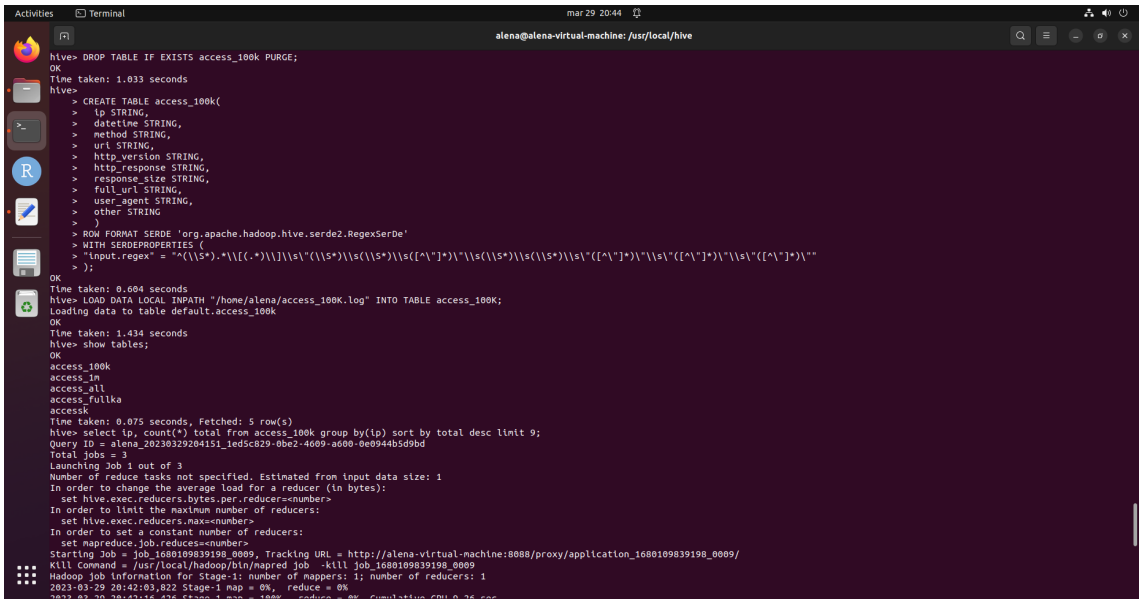
Základom pre spracovanie a analýzu logov bolo vytvoriť si tabuľku, do ktorej sme pomocou regexu parsovali jednotlivé logy. Tabuľku sme rozdelili na 10 polí a pomenovali podľa jednotlivých zložiek. Ak sme tabuľku vopred už mali vytvorenú, tak sme si ju vymazali.

```
DROP TABLE IF EXISTS access_100K PURGE;

CREATE TABLE access_100k (
  ip STRING,
  datetime STRING,
  method STRING,
  uri STRING,
  http_version STRING,
  http_response STRING,
  response_size STRING,
  full_url STRING,
  user_agent STRING,
  other STRING
  )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
"input.regex" = "^(\\S*).*\\[(.*)\\]\\s\"(\\S*)\\s(\\S*)
\\s([^\"]*)\"\\s(\\S*)\\s(\\S*)\\s\"([^\"]*)\"\\s\"([^\"]*)\"\\s\"([^\"]*)\""
);
```

Regex pre členenie logu sme si spravili na https://regex101.com/r/z9JTRm/1. Po otvorení si vieme pozrieť pekne jednotlivé vyparsované položky logu a pochopiť syntax regexu. Následne sme vyššie uvedený kód vložili do shellu. Výsledkom je vytvorená tabuľka, do ktorej sme následne cez load data načítali logy pomocou príkazu:

```
LOAD DATA LOCAL INPATH "/home/alena/access_100K.log" INTO TABLE access_100k;
```

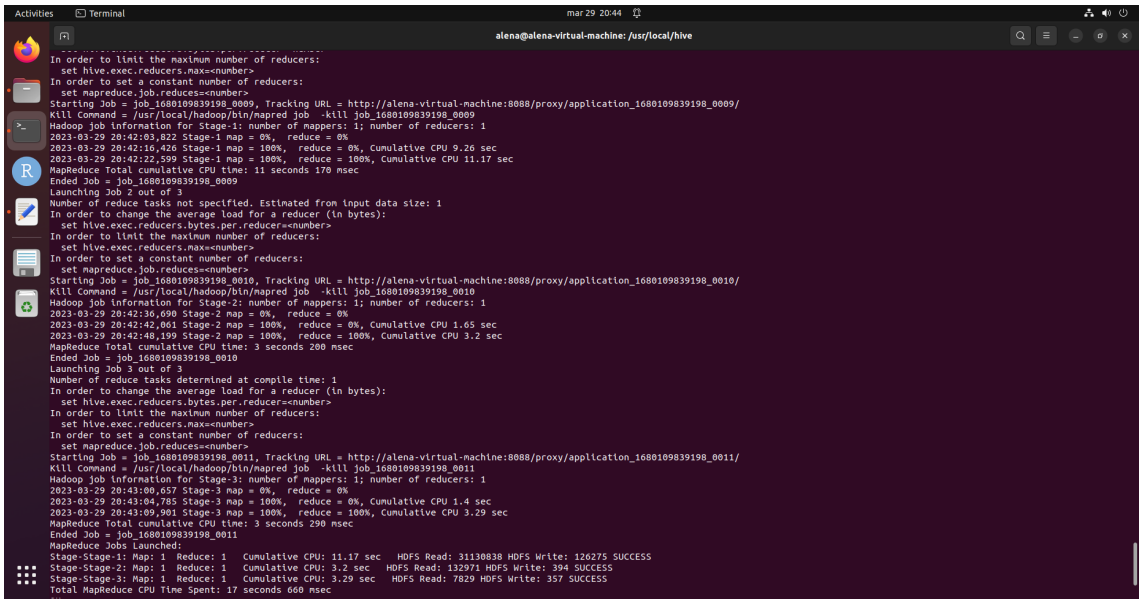


Obrázok 2: Ukážka vytvorenia tabuľky a načítania dát, zdroj: [vlastné spracovanie]

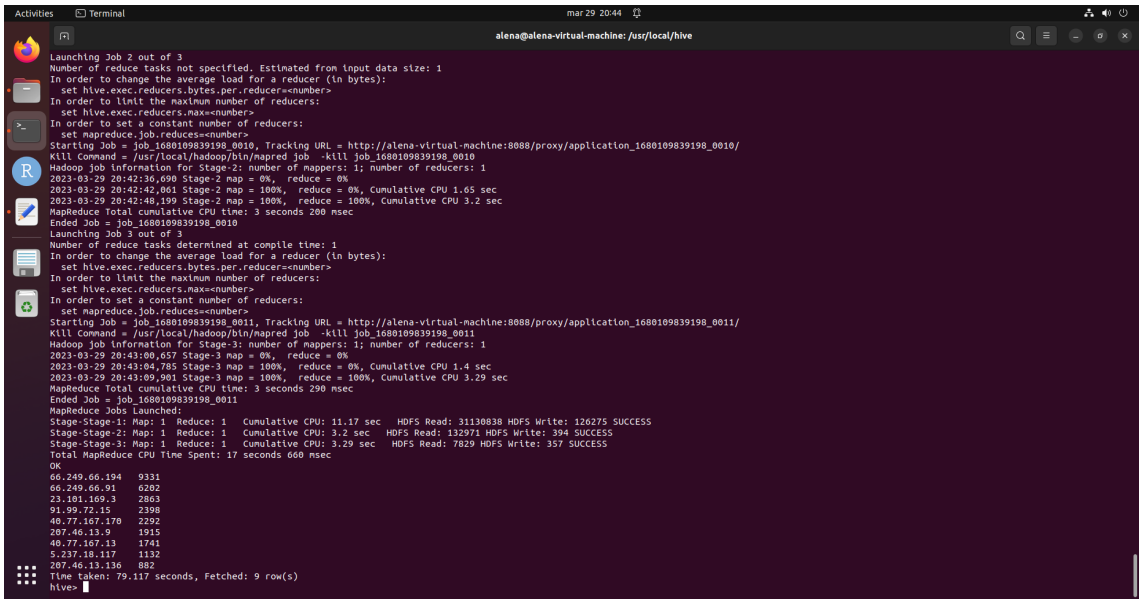
Posledným krokom bolo nad tabuľkou spraviť agregácie. Ako príklad si uveďme koľkokrát sa vyskytujú dané IP adresy v súbore resp. môžeme vidieť najaktívnejšie IP adresy. Môžeme tak spraviť prostredníctvom príkazu uvedeného nižšie.

```
select ip, count(*) total from access_100K group by(ip) sort by total desc limit 9;
```

Následne počkáme zopár minút pre zobrazenie výsledku nášho dopytu. V tejto fáze sú volané potrebné súčasti Hadoopu, a to MapReducer, ktorý vykoná svoju úlohu a následne získame výsledok.



Obrázok 3: Proces vykonávania jobs pri agregácii, zdroj: [vlastné spracovanie]

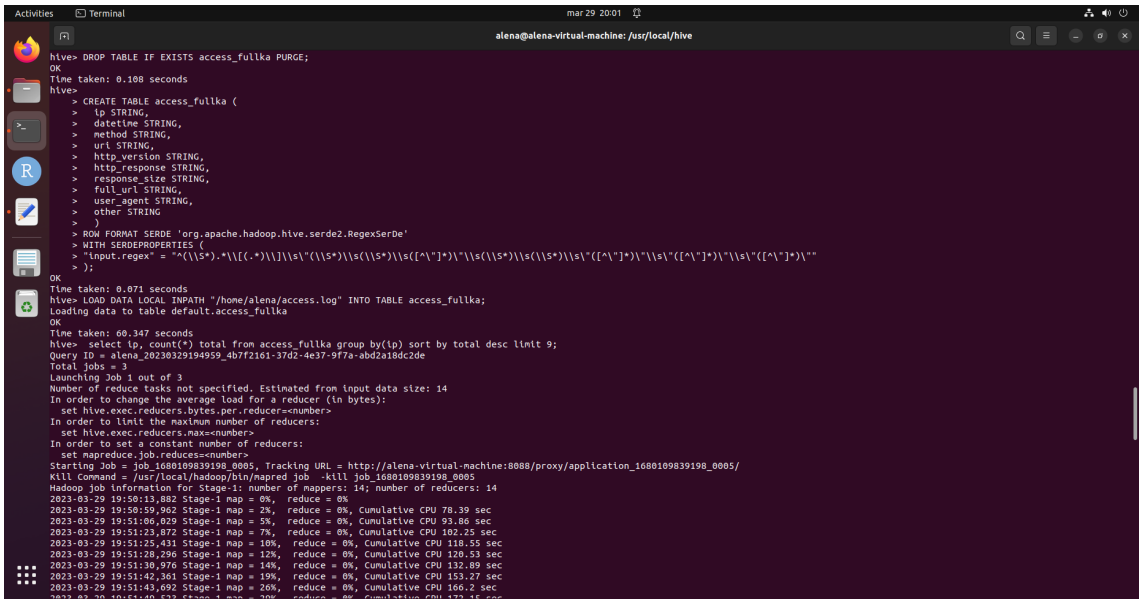


Obrázok 4: Výsledok SQL dopytu agregácie IP adries, zdroj: [vlastné spracovanie]

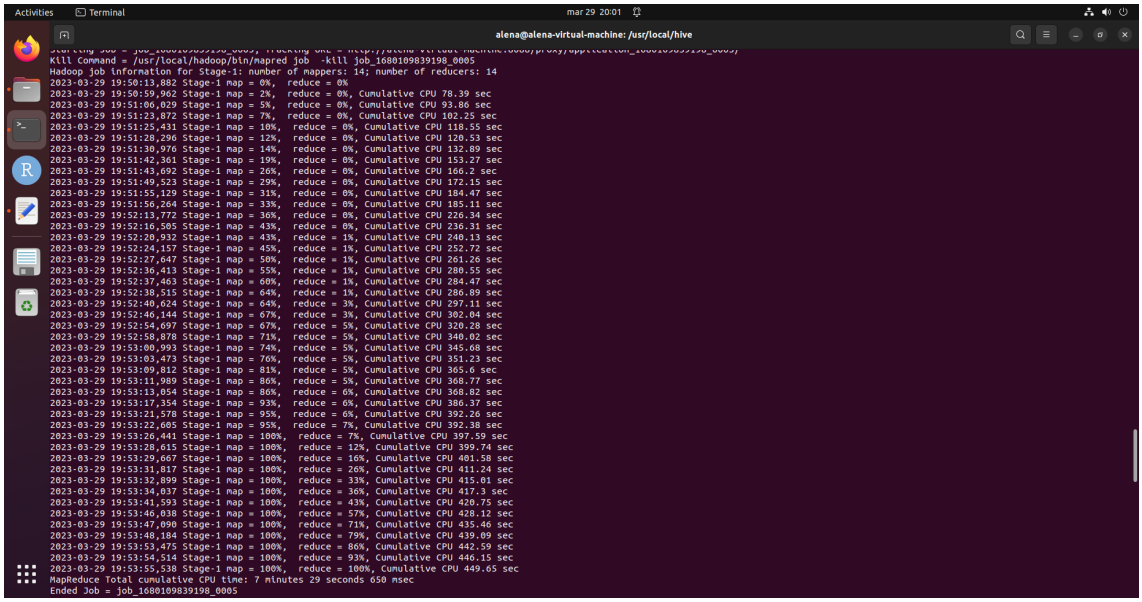
Podobným spôsobom by sme pomocou ďalšieho SQL dopytu mohli napríklad rozdeliť čas na jednotlivé zložky, ak by sme chceli zobraziť grafy prípadne iné metriky.

2.1.1 Spracovanie a analýza 3.2 GB súboru

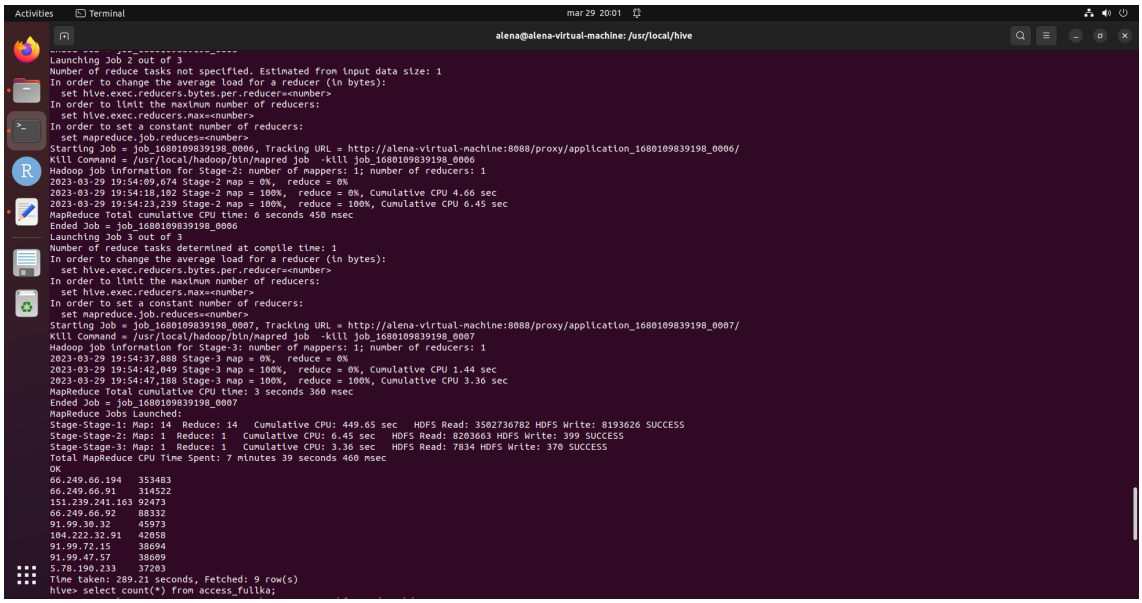
To isté sme spravili taktiež pre celý súbor s logmi, ktorý má veľkosť 3.2 GB. V Hortonworkse sme sa pri načítavaní takto veľkého súboru stretli s dlhým časovým trvaním, z tohto dôvodu je ukážka spracovania celého súboru len v tomto zadaní.



Obrázok 5: Vytvorenie tabuľky a načítanie dát z celého súboru, zdroj: [vlastné spracovanie]



Obrázok 6: Proces vykonávania jobs pri agregácii, zdroj: [vlastné spracovanie]



Obrázok 7: Výsledok SQL dopytu agregácie IP adries, zdroj: [vlastné spracovanie]

Záver

Cieľom tohto zadania bolo opísať inštaláciu Hive na Linuxe a následne spracovať a analyzovať niekoľkomegabajtový súbor až po trojgigabajtový súbor. Počas inštalácie sme sa stretli s problémami, ktoré sme opísali vyššie v dokumente, ale aj napriek tomu sme dokázali náš cieľ spracovania a analýzy logov splniť.

V porovnaní s produktom Hortonworks Sandbox je spracovanie logov v systéme Linux intuitívnejšie a rýchlejšie. Taktiež je veľmi natívne a povedali by sme aj jednoduchšie oproti Hortonworks Sandbox, keďže už spomenutý Hortonworks nám prišiel pomalší a zložitejší na naučenie, taktiež sme v ňom mali problém s administrátorskými právami a načítaním niekoľkogigabajtového súboru. Preto by sme na základe porovnania Hive v Linuxe a v Hortonworkse odporučili použiť Hive natívne pod Linuxom.

Veríme, že toto zadanie prinesie čitateľovi nový pohľad na možnosť spracovávať veľké súbory a získa nové prípadne obohatí už doterajšie vedomosti a schopnosti o danej problematike, ktorá je tiež súčasťou Big Data.

Zoznam použitej literatúry

[1] Tutorials Point India Private Limited. 2023. *Hive - Installation* In tutorialspoint.com [online]. 2023. [citované dňa 29.03.2023]. Dostupné na internete: `https://www.tutorialspoint.com/hive/hive_installation.htm`.