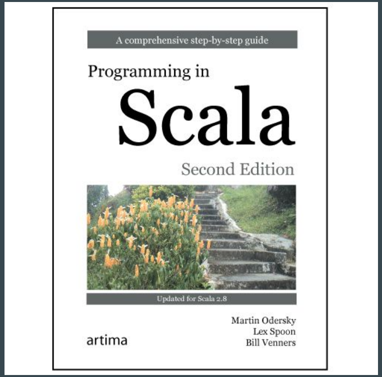
Stage Big Data

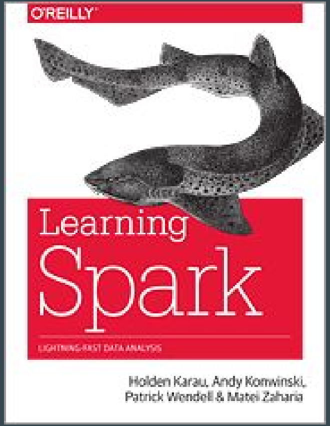
Frederic Everaert & Lorenz Verschingel

Inhoud

Zelfstudie

- Gebruikte bronnen voor Scala en Spark:




O'REILLY
Learning
Spark
LIGHTNING-FAST DATA ANALYSIS
Holden Karau, Andy Konwinski,
Patrick Wendell & Matei Zaharia

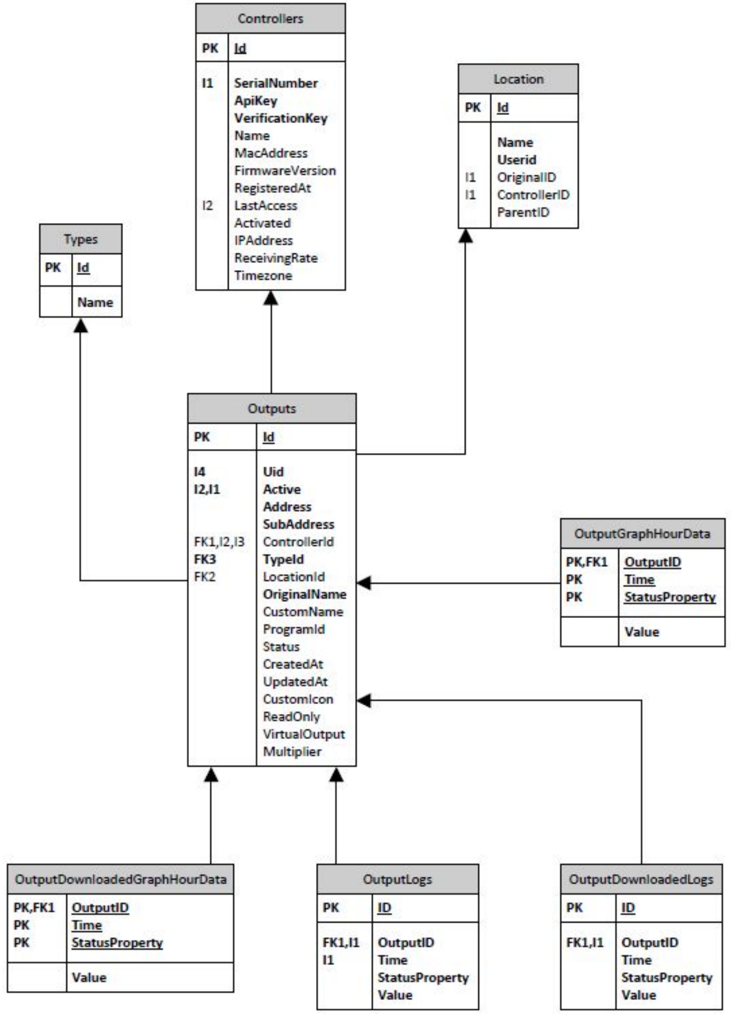
BIG DATA UNIVERSITY

Inhoud

Qbus databank analyse

- ERD opgesteld met Visio
  - Controllers
  - Types met naam ‘THERMO’
  - Outputs
  - Location(s)
  - OutputGraphHourData: measured data
  - OutputLogs: set data

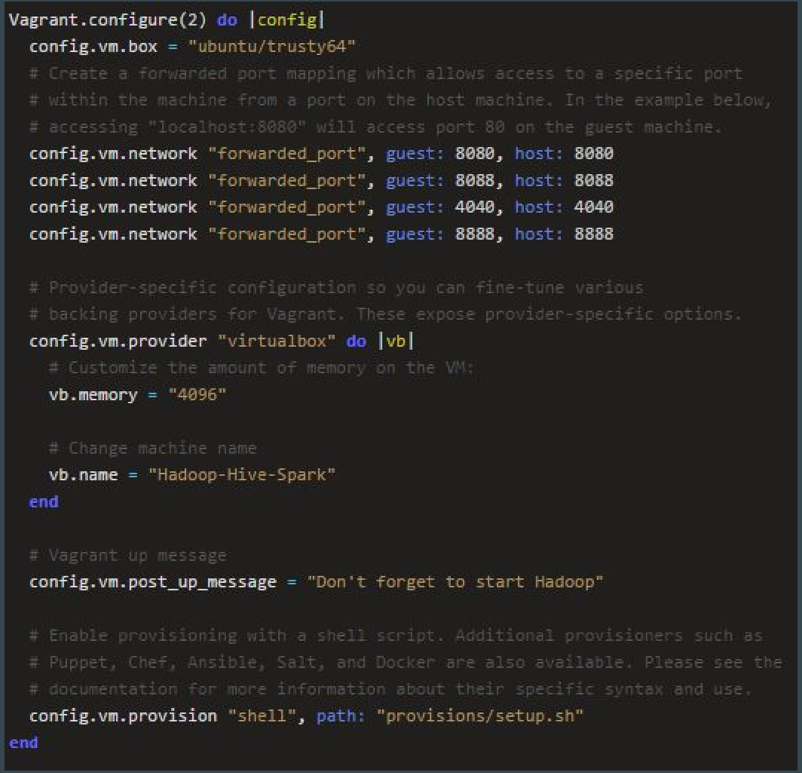
- SQL Server 2012 database geëxporteerd naar csv bestanden



Inhoud

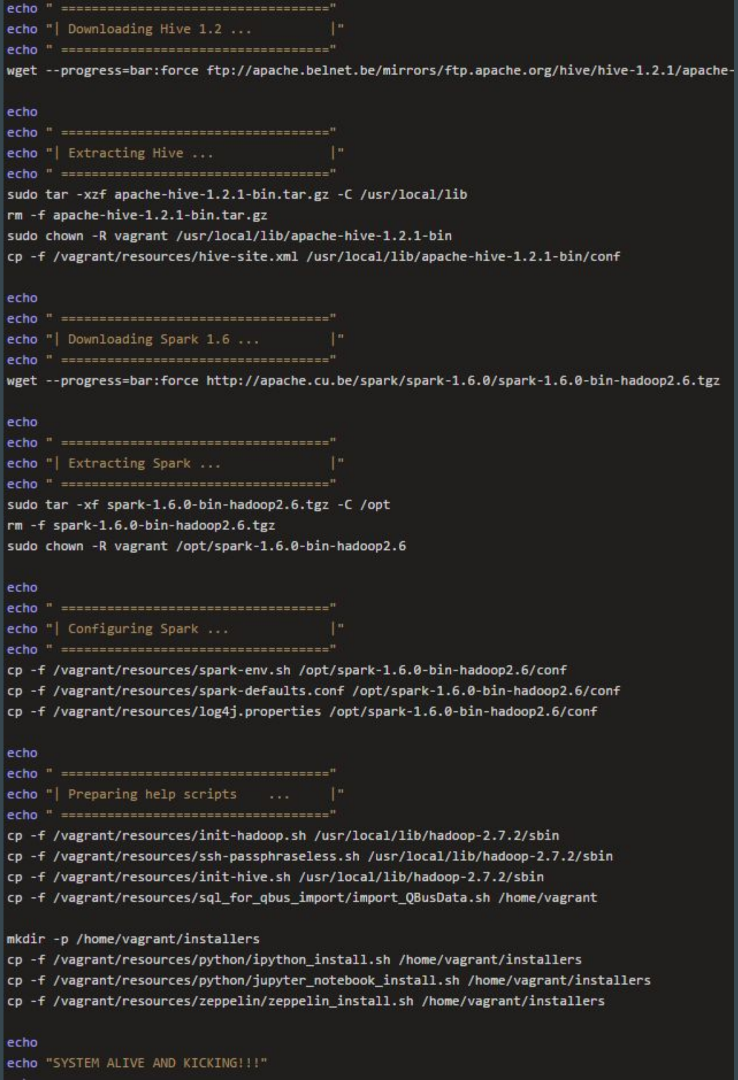
VM Hadoop-Hive-Spark

- Gemaakt met hulp van Vagrant
- VM = Ubuntu 64-bit
- 4 GB RAM
- Port forwarding:
  - 8080: Zeppelin notebook
  - 8088: YARN Cluster UI
  - 4040: Spark UI
  - 8888: Jupyter/IPython notebook



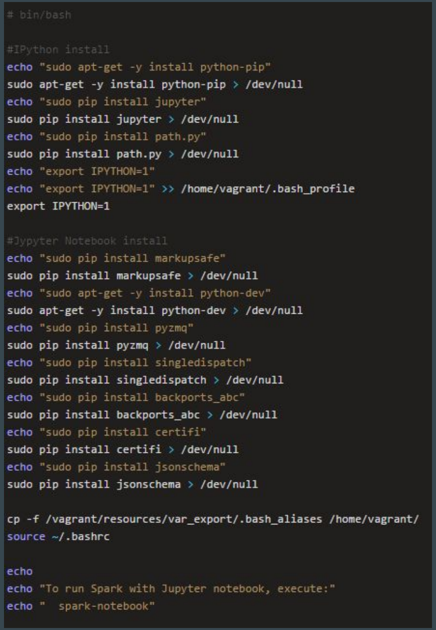
```
Vagrant.configure(2) do |config|
  config.vm.box = "ubuntu/trusty64"
  # Create a forwarded port mapping which allows access to a specific port
  # within the machine from a port on the host machine. In the example below,
  # accessing "localhost:8080" will access port 80 on the guest machine.
  config.vm.network "forwarded_port", guest: 8080, host: 8080
  config.vm.network "forwarded_port", guest: 8088, host: 8088
  config.vm.network "forwarded_port", guest: 4040, host: 4040
  config.vm.network "forwarded_port", guest: 8888, host: 8888

  # Provider-specific configuration so you can fine-tune various
  # backing providers for Vagrant. These expose provider-specific options.
  config.vm.provider "virtualbox" do |vb|
    # Customize the amount of memory on the VM:
    vb.memory = "4096"

    # Change machine name
    vb.name = "Hadoop-Hive-Spark"
  end

  # Vagrant up message
  config.vm.post_up_message = "Don't forget to start Hadoop"

  # Enable provisioning with a shell script. Additional provisioners such as
  # Puppet, Chef, Ansible, Salt, and Docker are also available. Please see the
  # documentation for more information about their specific syntax and use.
  config.vm.provision "shell", path: "provisions/setup.sh"
end
```

Vagrant setup.sh



```
echo " ==================================="
echo "| Downloading Hive 1.2 ...          |"
echo " ==================================="
wget --progress=bar:force ftp://apache.belnet.be/mirrors/ftp.apache.org/hive/hive-1.2.1/apache-

echo
echo " ==================================="
echo "| Extracting Hive ...               |"
echo " ==================================="
sudo tar -xzf apache-hive-1.2.1-bin.tar.gz -C /usr/local/lib
rm -f apache-hive-1.2.1-bin.tar.gz
sudo chown -R vagrant /usr/local/lib/apache-hive-1.2.1-bin
cp -f /vagrant/resources/hive-site.xml /usr/local/lib/apache-hive-1.2.1-bin/conf

echo
echo " ==================================="
echo "| Downloading Spark 1.6 ...         |"
echo " ==================================="
wget --progress=bar:force http://apache.cu.be/spark/spark-1.6.0/spark-1.6.0-bin-hadoop2.6.tgz

echo
echo " ==================================="
echo "| Extracting Spark ...              |"
echo " ==================================="
sudo tar -xf spark-1.6.0-bin-hadoop2.6.tgz -C /opt
rm -f spark-1.6.0-bin-hadoop2.6.tgz
sudo chown -R vagrant /opt/spark-1.6.0-bin-hadoop2.6

echo
echo " ==================================="
echo "| Configuring Spark ...             |"
echo " ==================================="
cp -f /vagrant/resources/spark-env.sh /opt/spark-1.6.0-bin-hadoop2.6/conf
cp -f /vagrant/resources/spark-defaults.conf /opt/spark-1.6.0-bin-hadoop2.6/conf
cp -f /vagrant/resources/log4j.properties /opt/spark-1.6.0-bin-hadoop2.6/conf

echo
echo " ==================================="
echo "| Preparing help scripts    ...     |"
echo " ==================================="
cp -f /vagrant/resources/init-hadoop.sh /usr/local/lib/hadoop-2.7.2/sbin
cp -f /vagrant/resources/ssh-passphraseless.sh /usr/local/lib/hadoop-2.7.2/sbin
cp -f /vagrant/resources/init-hive.sh /usr/local/lib/hadoop-2.7.2/sbin
cp -f /vagrant/resources/sql_for_qbus_import/import_QBusData.sh /home/vagrant

mkdir -p /home/vagrant/installers
cp -f /vagrant/resources/python/ipython_install.sh /home/vagrant/installers
cp -f /vagrant/resources/python/jupyter_notebook_install.sh /home/vagrant/installers
cp -f /vagrant/resources/zeppelin/zeppelin_install.sh /home/vagrant/installers

echo
echo "SYSTEM ALIVE AND KICKING!!!"
```

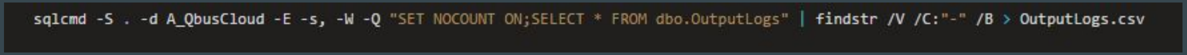
IPython/Jupyter notebook install.sh



```
# bin/bash

#IPython install
echo "sudo apt-get -y install python-pip"
sudo apt-get -y install python-pip > /dev/null
echo "sudo pip install jupyter"
sudo pip install jupyter > /dev/null
echo "sudo pip install path.py"
sudo pip install path.py > /dev/null
echo "export IPYTHON=1"
echo "export IPYTHON=1" >> /home/vagrant/.bash_profile
export IPYTHON=1

#Jypyter Notebook install
echo "sudo pip install markupsafe"
sudo pip install markupsafe > /dev/null
echo "sudo apt-get -y install python-dev"
sudo apt-get -y install python-dev > /dev/null
echo "sudo pip install pyzmq"
sudo pip install pyzmq > /dev/null
echo "sudo pip install singledispatch"
sudo pip install singledispatch > /dev/null
echo "sudo pip install backports_abc"
sudo pip install backports_abc > /dev/null
echo "sudo pip install certifi"
sudo pip install certifi > /dev/null
echo "sudo pip install jsonschema"
sudo pip install jsonschema > /dev/null

cp -f /vagrant/resources/var_export/.bash_aliases /home/vagrant/
source ~/.bashrc

echo
echo "To run Spark with Jupyter notebook, execute:"
echo "  spark-notebook"
```

Import Qbus data naar VM

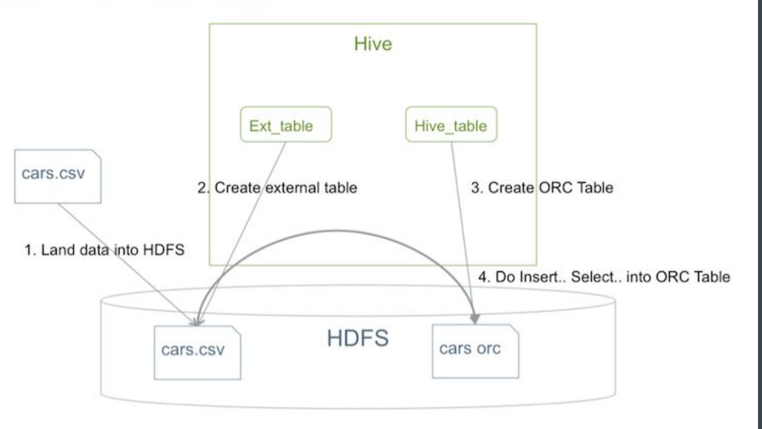
- SQL Server export naar CSV files



```
sqlcmd -S . -d A_QbusCloud -E -s, -W -Q "SET NOCOUNT ON;SELECT * FROM dbo.OutputLogs" | findstr /V /C:"-" /B > OutputLogs.csv
```

- CSV data naar Hive tabellen:
    a. CSV ingeladen in HDFS
    b. Externe tabellen aangemaakt
    c. Interne ORC tabellen aangemaakt
    d. Data van externe -> interne tabellen

Figure 1.1. Example: Moving .CSV Data into Hive

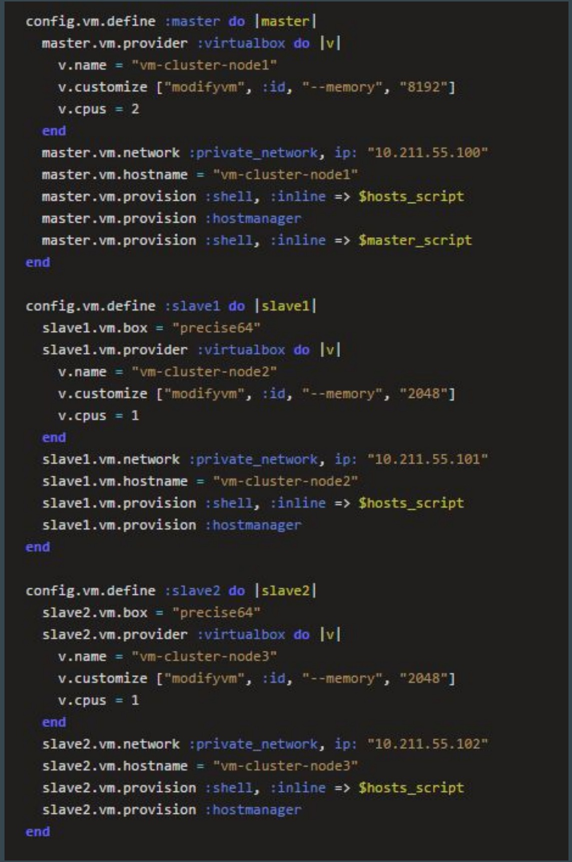


Jupyter notebook - Demo

Cloudera cluster VM

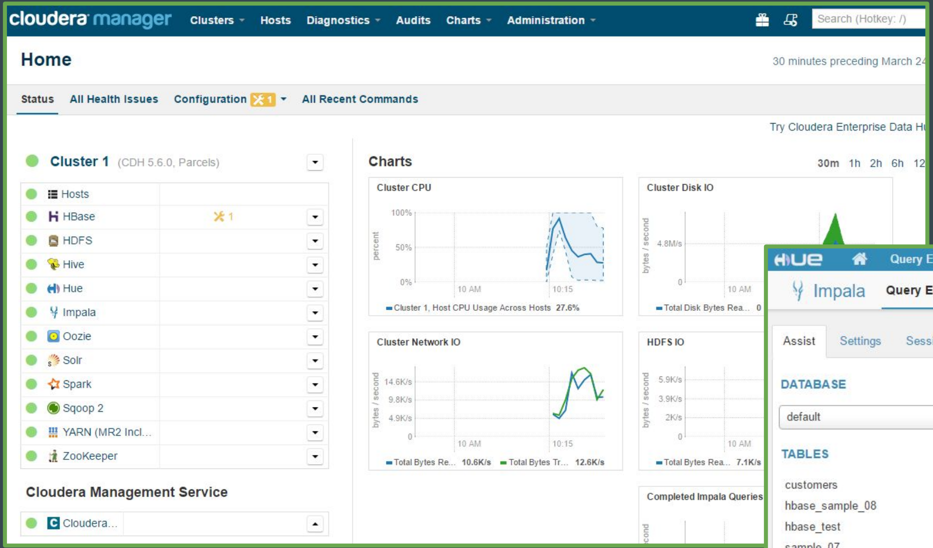
Overgenomen van: http://blog.cloudera.com/blog/2014/06/how-to-install-a-virtual-apache-hadoop-cluster-with-vagrant-and-cloudera-manager/

- Simuleren van cloudera.ugent.be omgeving
- 3 VM's
  - Master: 8GB RAM
  - 2 slaves elk 2GB RAM
- Hive met Impala vergeleken
- Met CDH 5 kan alles via YARN geregeld worden: dit moest nog geconfigureerd worden voor Impala
- HBase succesvol geconfigureerd MAAR Impala kan niet lezen uit HBase tabellen

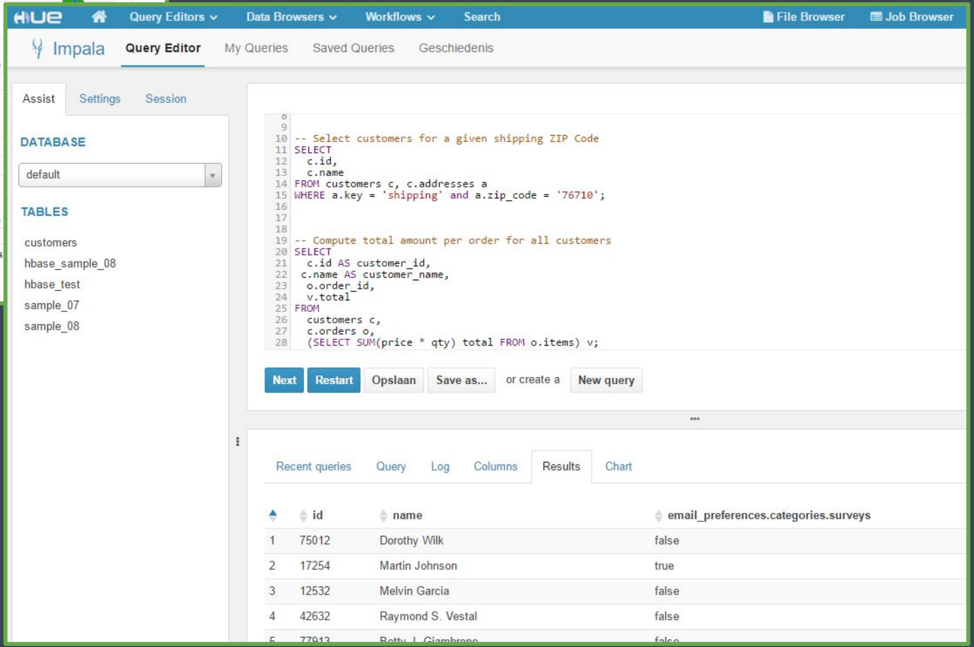


```
config.vm.define :master do |master|
  master.vm.provider :virtualbox do |v|
    v.name = "vm-cluster-node1"
    v.customize ["modifyvm", :id, "--memory", "8192"]
    v.cpus = 2
  end
  master.vm.network :private_network, ip: "10.211.55.100"
  master.vm.hostname = "vm-cluster-node1"
  master.vm.provision :shell, :inline => $hosts_script
  master.vm.provision :hostmanager
  master.vm.provision :shell, :inline => $master_script
end

config.vm.define :slave1 do |slave1|
  slave1.vm.box = "precise64"
  slave1.vm.provider :virtualbox do |v|
    v.name = "vm-cluster-node2"
    v.customize ["modifyvm", :id, "--memory", "2048"]
    v.cpus = 1
  end
  slave1.vm.network :private_network, ip: "10.211.55.101"
  slave1.vm.hostname = "vm-cluster-node2"
  slave1.vm.provision :shell, :inline => $hosts_script
  slave1.vm.provision :hostmanager
end

config.vm.define :slave2 do |slave2|
  slave2.vm.box = "precise64"
  slave2.vm.provider :virtualbox do |v|
    v.name = "vm-cluster-node3"
    v.customize ["modifyvm", :id, "--memory", "2048"]
    v.cpus = 1
  end
  slave2.vm.network :private_network, ip: "10.211.55.102"
  slave2.vm.hostname = "vm-cluster-node3"
  slave2.vm.provision :shell, :inline => $hosts_script
  slave2.vm.provision :hostmanager
end
```

Cloudera manager UI



HUE Impala query



```
-- Select customers for a given shipping ZIP Code
SELECT
  c.id,
  c.name
FROM customers c, c.addresses a
WHERE a.key = 'shipping' and a.zip_code = '76710';



-- Compute total amount per order for all customers
SELECT
  c.id AS customer_id,
 c.name AS customer_name,
  o.order_id,
  v.total
FROM
  customers c,
  c.orders o,
  (SELECT SUM(price * qty) total FROM o.items) v;
```

	id	name	email_preferences.categories.surveys
1	75012	Dorothy Wilk	false
2	17254	Martin Johnson	true
3	12532	Melvin Garcia	false
4	42632	Raymond S. Vestal	false
5	77913	Betty L. Giambrone	false

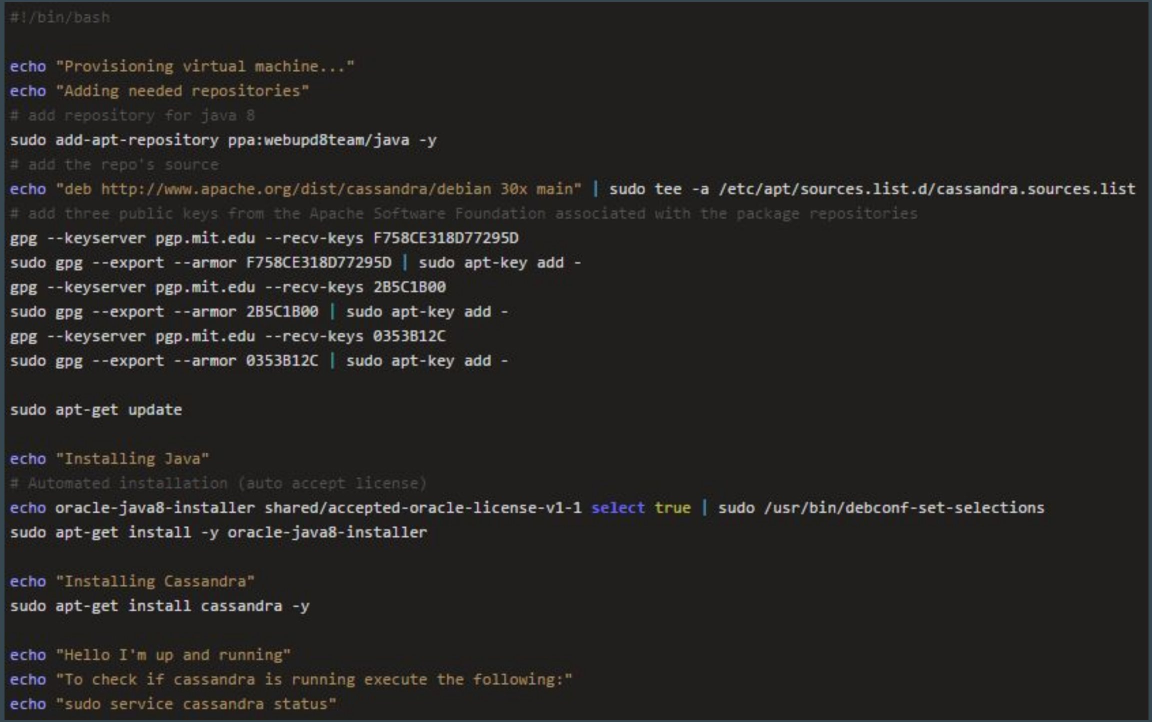
Vergelijking Hive, Impala en Spark op Cloudera

- Uitgevoerd op 3 verschillende voorbeeld query's en tabellen (klein met ~700 entries)
- Resultaten:

	Hive	**Impala**	**Spark Scala**	**PySpark**
Job loss	38.303s	0.63s	28s	32.3s
Salary growth	36.769s	0.61s	23s	25.2s
Top salary	27.548s	0.42s	23s	20.3s

Cassandra single-node VM

- Gemaakt met hulp van Vagrant
- VM = Ubuntu 64-bit
- 4 GB RAM
- Cassandra installeren via setup file



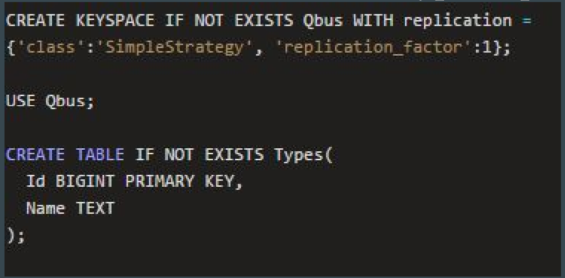
```
#!/bin/bash

echo "Provisioning virtual machine..."
echo "Adding needed repositories"
# add repository for java 8
sudo add-apt-repository ppa:webupd8team/java -y
# add the repo's source
echo "deb http://www.apache.org/dist/cassandra/debian 30x main" | sudo tee -a /etc/apt/sources.list.d/cassandra.sources.list
# add three public keys from the Apache Software Foundation associated with the package repositories
gpg --keyserver pgp.mit.edu --recv-keys F758CE318D77295D
sudo gpg --export --armor F758CE318D77295D | sudo apt-key add -
gpg --keyserver pgp.mit.edu --recv-keys 2B5C1B00
sudo gpg --export --armor 2B5C1B00 | sudo apt-key add -
gpg --keyserver pgp.mit.edu --recv-keys 0353B12C
sudo gpg --export --armor 0353B12C | sudo apt-key add -

sudo apt-get update

echo "Installing Java"
# Automated installation (auto accept license)
echo oracle-java8-installer shared/accepted-oracle-license-v1-1 select true | sudo /usr/bin/debconf-set-selections
sudo apt-get install -y oracle-java8-installer

echo "Installing Cassandra"
sudo apt-get install cassandra -y

echo "Hello I'm up and running"
echo "To check if cassandra is running execute the following:"
echo "sudo service cassandra status"
```

Importeren QBus data

- Keyspace en tabellen aanmaken
- Data uit CSV importeren
- Uitvoeren cql files

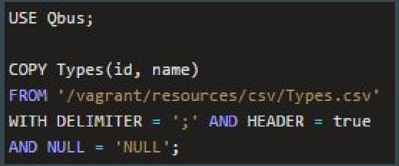
- Problemen:
  - Cassandra is strenger voor import data, weigert rijen te importeren
  - Meer kolommen op een rij dan verwacht
    - Slechte delimiter ,
    - Opnieuw export met delimiter ;
  - ASCII controle karakters (zoals NUL bytes)
  - Float waarden niet juist
  - Te grote csv files, Cassandra kan geen csv importeren groter dan 1 miljoen rijen
    - Gesplitst in kleinere files

Types.cql



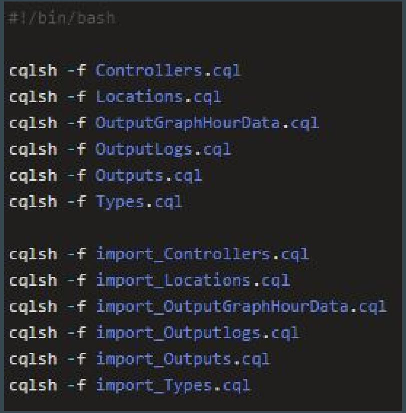
```
CREATE KEYSPACE IF NOT EXISTS Qbus WITH replication =
{'class':'SimpleStrategy', 'replication_factor':1};

USE Qbus;

CREATE TABLE IF NOT EXISTS Types(
  Id BIGINT PRIMARY KEY,
  Name TEXT
);
```

import_Types.cql



```
USE Qbus;

COPY Types(id, name)
FROM '/vagrant/resources/csv/Types.csv'
WITH DELIMITER = ';' AND HEADER = true
AND NULL = 'NULL';
```

create_and_import.sh



```
#!/bin/bash

cqlsh -f Controllers.cql
cqlsh -f Locations.cql
cqlsh -f OutputGraphHourData.cql
cqlsh -f OutputLogs.cql
cqlsh -f Outputs.cql
cqlsh -f Types.cql

cqlsh -f import_Controllers.cql
cqlsh -f import_Locations.cql
cqlsh -f import_OutputGraphHourData.cql
cqlsh -f import_Outputlogs.cql
cqlsh -f import_Outputs.cql
cqlsh -f import_Types.cql
```

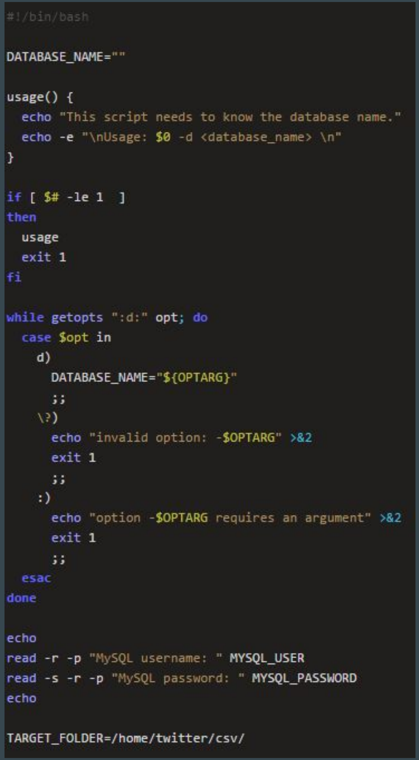
Inhoud

- Zelfstudie
- Qbus databank analyse
- Lokale virtuele machines
- **Twitter data export**
- SQL query’s marktonderzoek voor master student
- Club Brugge DB

Twitter data export

- MySQL databanken van verschillende schijven exporteren naar CSV files
- Automatisatie via bash script

twitter_export.sh (1)

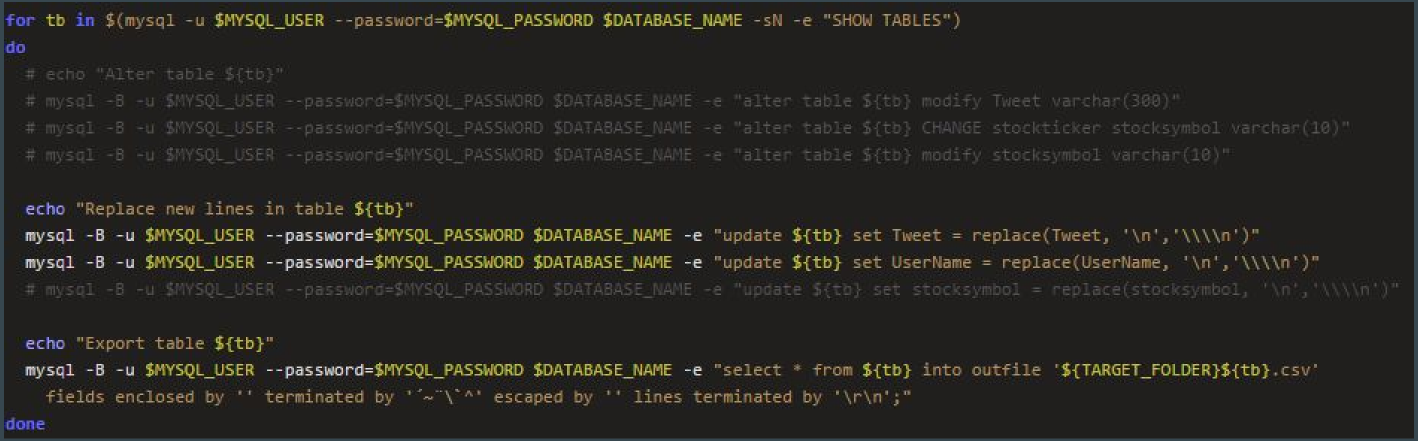


```bash
#!/bin/bash

DATABASE_NAME=""

usage() {
  echo "This script needs to know the database name."
  echo -e "\nUsage: $0 -d <database_name> \n"
}

if [ $# -le 1  ]
then
  usage
  exit 1
fi

while getopts ":d:" opt; do
  case $opt in
    d)
      DATABASE_NAME="${OPTARG}"
      ;;
    \?)
      echo "invalid option: -$OPTARG" >&2
      exit 1
      ;;
    :)
      echo "option -$OPTARG requires an argument" >&2
      exit 1
      ;;
  esac
done

echo
read -r -p "MySQL username: " MYSQL_USER
read -s -r -p "MySQL password: " MYSQL_PASSWORD
echo

TARGET_FOLDER=/home/twitter/csv/
```

Twitter data export

- MySQL databanken van verschillende schijven exporteren naar CSV files
- Automatisatie via bash script

twitter_export.sh (2)



```
for tb in $(mysql -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -sN -e "SHOW TABLES")
do
  # echo "Alter table ${tb}"
  # mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "alter table ${tb} modify Tweet varchar(300)"
  # mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "alter table ${tb} CHANGE stockticker stocksymbol varchar(10)"
  # mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "alter table ${tb} modify stocksymbol varchar(10)"

  echo "Replace new lines in table ${tb}"
  mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "update ${tb} set Tweet = replace(Tweet, '\n','\\\\n')"
  mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "update ${tb} set UserName = replace(UserName, '\n','\\\\n')"
  # mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "update ${tb} set stocksymbol = replace(stocksymbol, '\n','\\\\n')"

  echo "Export table ${tb}"
  mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e "select * from ${tb} into outfile '${TARGET_FOLDER}${tb}.csv'
    fields enclosed by '' terminated by '´~¨\`^' escaped by '' lines terminated by '\r\n';"
done
```

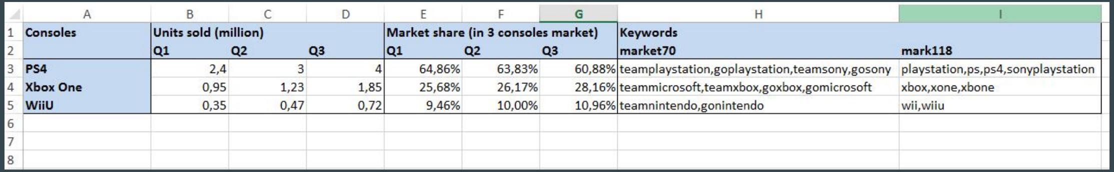
Inhoud

- Zelfstudie
- Qbus databank analyse
- Lokale virtuele machines
- Twitter data export
- **SQL query's marktonderzoek voor master student**
- Club Brugge DB

SQL query’s marktonderzoek master student

- Opdracht gegeven door Mr. Van den Poel
- Excel met marktaandelen per markt en keywords die te onderzoeken zijn

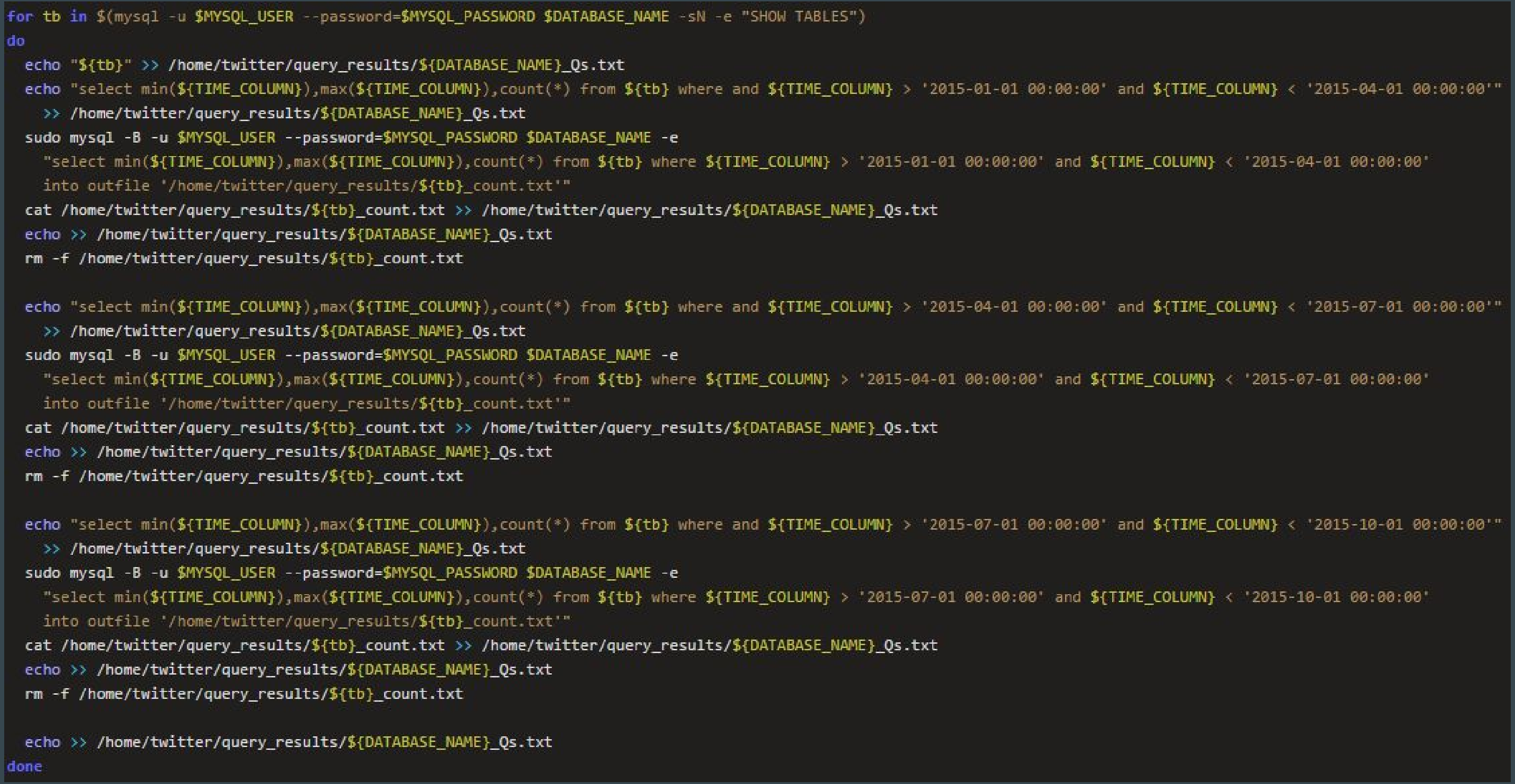
Console markt



	A	B	C	D	E	F	G	H	I
1	**Consoles**	**Units sold (million)**			**Market share (in 3 consoles market)**			**Keywords**	
2		**Q1**	**Q2**	**Q3**	**Q1**	**Q2**	**Q3**	**market70**	**mark118**
3	**PS4**	2,4	3	4	64,86%	63,83%	60,88%	teamplaystation,goplaystation,teamsony,gosony	playstation,ps,ps4,sonyplaystation
4	**Xbox One**	0,95	1,23	1,85	25,68%	26,17%	28,16%	teammicrosoft,teamxbox,goxbox,gomicrosoft	xbox,xone,xbone
5	**WiiU**	0,35	0,47	0,72	9,46%	10,00%	10,96%	teamnintendo,gonintendo	wii,wiiu
6									
7									
8									

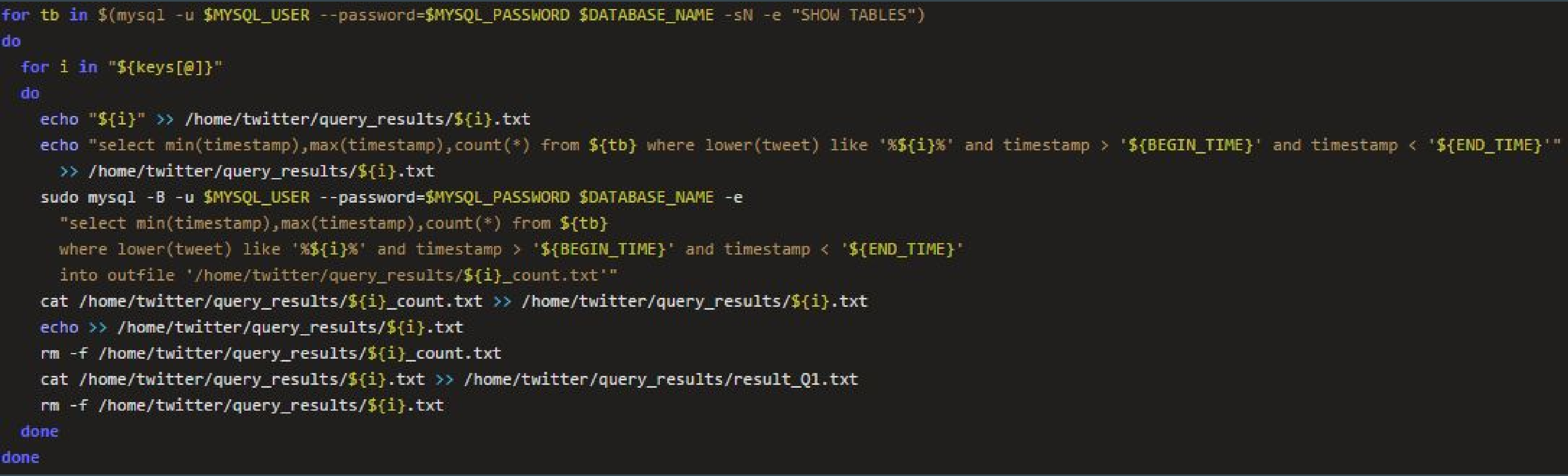
- Overlappende of vaak voorkomende keywords filteren
    - Vb.: ps en ps4

MySQL query’s - Kwartalen ontdekken



```
for tb in $(mysql -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -sN -e "SHOW TABLES")
do
  echo "${tb}" >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  echo "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where and ${TIME_COLUMN} > '2015-01-01 00:00:00' and ${TIME_COLUMN} < '2015-04-01 00:00:00'"
    >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  sudo mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e
    "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where ${TIME_COLUMN} > '2015-01-01 00:00:00' and ${TIME_COLUMN} < '2015-04-01 00:00:00'
    into outfile '/home/twitter/query_results/${tb}_count.txt'"
  cat /home/twitter/query_results/${tb}_count.txt >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  echo >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  rm -f /home/twitter/query_results/${tb}_count.txt

  echo "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where and ${TIME_COLUMN} > '2015-04-01 00:00:00' and ${TIME_COLUMN} < '2015-07-01 00:00:00'"
    >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  sudo mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e
    "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where ${TIME_COLUMN} > '2015-04-01 00:00:00' and ${TIME_COLUMN} < '2015-07-01 00:00:00'
    into outfile '/home/twitter/query_results/${tb}_count.txt'"
  cat /home/twitter/query_results/${tb}_count.txt >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  echo >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  rm -f /home/twitter/query_results/${tb}_count.txt

  echo "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where and ${TIME_COLUMN} > '2015-07-01 00:00:00' and ${TIME_COLUMN} < '2015-10-01 00:00:00'"
    >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  sudo mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e
    "select min(${TIME_COLUMN}),max(${TIME_COLUMN}),count(*) from ${tb} where ${TIME_COLUMN} > '2015-07-01 00:00:00' and ${TIME_COLUMN} < '2015-10-01 00:00:00'
    into outfile '/home/twitter/query_results/${tb}_count.txt'"
  cat /home/twitter/query_results/${tb}_count.txt >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  echo >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
  rm -f /home/twitter/query_results/${tb}_count.txt

  echo >> /home/twitter/query_results/${DATABASE_NAME}_Qs.txt
done
```

MySQL query's - Count voor keywords



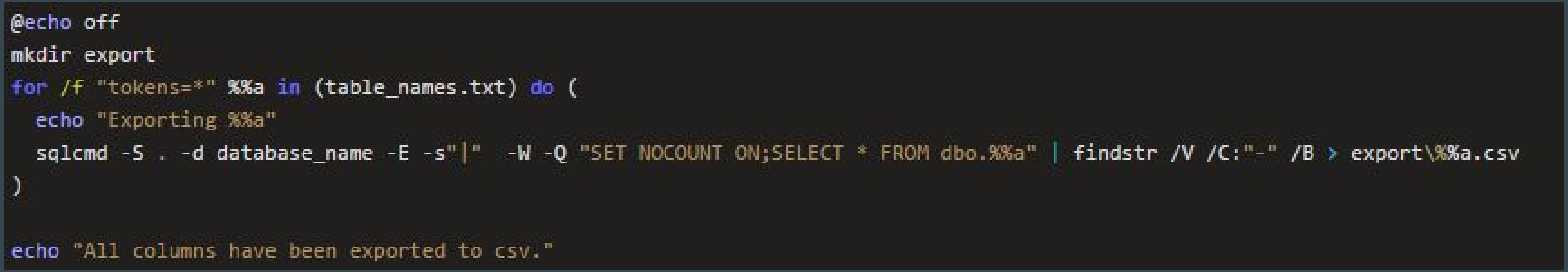
```
for tb in $(mysql -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -sN -e "SHOW TABLES")
do
  for i in "${keys[@]}"
  do
    echo "${i}" >> /home/twitter/query_results/${i}.txt
    echo "select min(timestamp),max(timestamp),count(*) from ${tb} where lower(tweet) like '%${i}%' and timestamp > '${BEGIN_TIME}' and timestamp < '${END_TIME}'"
      >> /home/twitter/query_results/${i}.txt
    sudo mysql -B -u $MYSQL_USER --password=$MYSQL_PASSWORD $DATABASE_NAME -e
      "select min(timestamp),max(timestamp),count(*) from ${tb}
      where lower(tweet) like '%${i}%' and timestamp > '${BEGIN_TIME}' and timestamp < '${END_TIME}'
      into outfile '/home/twitter/query_results/${i}_count.txt'"
    cat /home/twitter/query_results/${i}_count.txt >> /home/twitter/query_results/${i}.txt
    echo >> /home/twitter/query_results/${i}.txt
    rm -f /home/twitter/query_results/${i}_count.txt
    cat /home/twitter/query_results/${i}.txt >> /home/twitter/query_results/result_Q1.txt
    rm -f /home/twitter/query_results/${i}.txt
  done
done
```

Inhoud

- Zelfstudie
- Qbus databank analyse
- Lokale virtuele machines
- Twitter data export
- SQL query’s marktonderzoek voor master student
- **Club Brugge DB**

Club Brugge db

- Opdracht gegeven door mr. Van den Poel
- SQL server database
- Exporteren naar csv



```
@echo off
mkdir export
for /f "tokens=*" %%a in (table_names.txt) do (
  echo "Exporting %%a"
  sqlcmd -S . -d database_name -E -s"|"  -W -Q "SET NOCOUNT ON;SELECT * FROM dbo.%%a" | findstr /V /C:"-" /B > export\%%a.csv
)

echo "All columns have been exported to csv."
```

Zijn er nog vragen?