How to transfer files from the local computer to VagrantVM, then to hdfs?

1.” You need to install the plugin, like so

```
vagrant plugin install vagrant-scp “
```

2. cd D:\classroom\sem6\bda\vagrant-hadoop-hive-spark

3. vagrant scp solutions.txt node2:/home/vagrant

Then,

4. vagrant ssh

5. hadoop fs -mkdir /user/test

6. hadoop fs -copyFromLocal /home/vagrant/solutions.txt /user/test

Refences for HDFS/Hadoop commands:

https://www.javatpoint.com/hdfs

How to run own pyspark program ?

1.vagrant up, then vagrant ssh.

2.Type

touch yo.txt

then

echo python is very easy > yo.txt

3. Create a file testt.py in “/home/vagrant” and using vim editor write following text in it.

```
import pyspark

sc = pyspark.SparkContext('local[*]')

txt = sc.textFile('file:////home/vagrant/yo.txt')

print(txt.count())
```

```
python_lines = txt.filter(lambda line: 'python' in line.lower())

print(python_lines.count())
```

4.To run the python file, type

/usr/local/spark/bin/spark-submit testt.py

Refences: https://realpython.com/pyspark-intro/

https://opensource.com/article/19/3/getting-started-vim

How to read agency.txt from hdfs using pyspark?

just write below code in a python file and run it using /usr/local/spark/bin/spark-submit testt.py

```
Import pyspark

from pyspark.sql import SparkSession

from pyspark.sql.types import StructType

spark = SparkSession.builder\

      .master("local").appName("hdfs_test").getOrCreate()

booksdata=spark.read.option(“header”,”true”).csv("hdfs:///user/test/bus_stop_data/agency.txt")

booksdata.read(5)
```

Reference:https://www.projectpro.io/recipes/read-data-from-hdfs-pyspark,

https://stackoverflow.com/questions/18142960/whats-the-difference-between-hadoop-fs-shell-commands-and-hdfs-dfs-shell-co

Refences For pyspark sql programming

https://spark.apache.org/docs/3.1.1/api/python/reference/pyspark.sql.html

https://sparkbyexamples.com/pyspark/

Important tables and their columns

routes:route_short_name,route_long_name,route_type,route_id,agency_id

stop_times:trip_id,arrival_time,departure_time,stop_id,stop_sequence

stops: stop_id, stop_code, stop_name, stop_lat, stop_lon, zone_id

trips :route_id,service_id,trip_id,shape_id

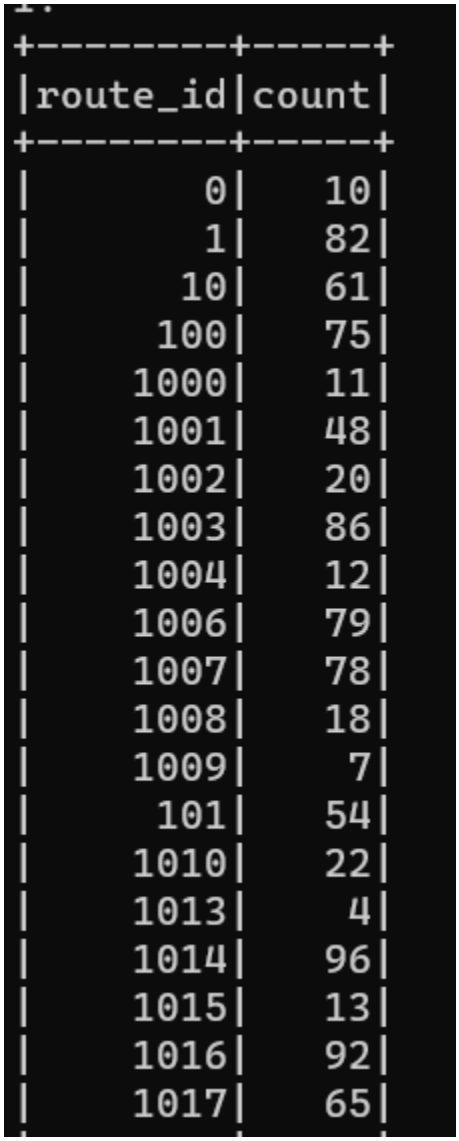
Preprocessing

stop_sequence column of stop_times was changed to IntegerType

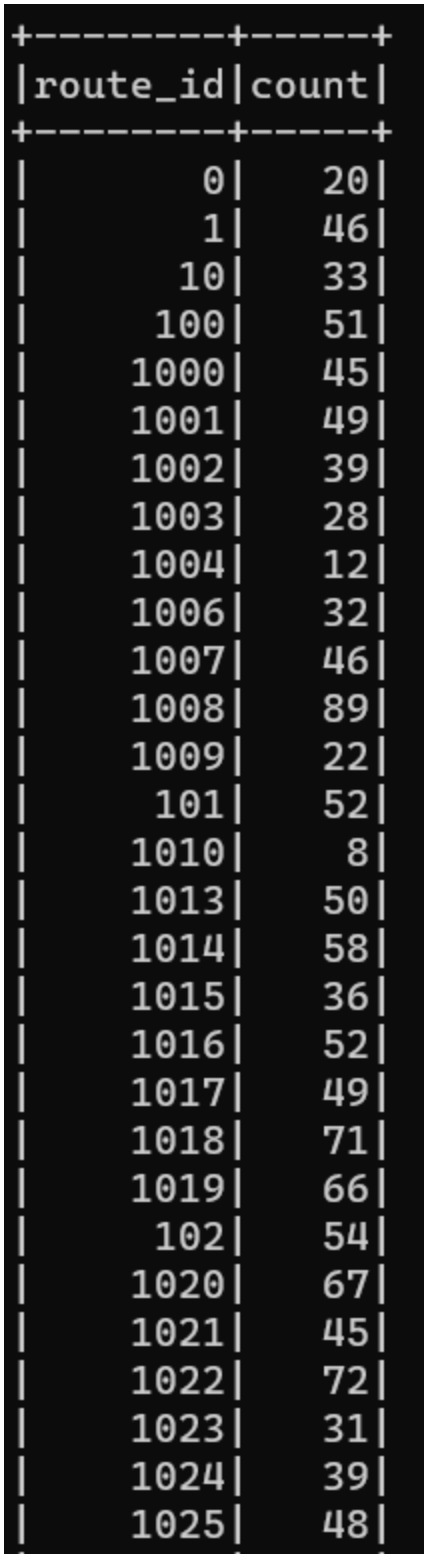
arrival_time column of stop_times was changed to to_timestamp

departure_time column of stop_times was changed to to_timestamp

Q1 trip count



route_id	count
0	10
1	82
10	61
100	75
1000	11
1001	48
1002	20
1003	86
1004	12
1006	79
1007	78
1008	18
1009	7
101	54
1010	22
1013	4
1014	96
1015	13
1016	92
1017	65



route_id	count
0	20
1	46
10	33
100	51
1000	45
1001	49
1002	39
1003	28
1004	12
1006	32
1007	46
1008	89
1009	22
101	52
1010	8
1013	50
1014	58
1015	36
1016	52
1017	49
1018	71
1019	66
102	54
1020	67
1021	45
1022	72
1023	31
1024	39
1025	48

stop count

Q2

a)

route_id	stop_name	stop_lat	stop_lon
605	Dwarka More Metro Station (Terminal)	28.618575	77.031707
387	Azadpur Terminal	28.7072	77.1783
597	Rani Khera Depot 2	28.699440999999997	77.033514
961	Dilshad Garden Depot Cluster	28.685631	77.331179
117	Bakkargarh Village	28.665996999999997	77.017004
820	Ghoga Village	28.83063	77.04937
1003	Haider Pur Village	28.722855	77.151477
505	Kair Depot	28.618288	76.930672
1252	Nilothi Crossing	28.66444000000001	77.052583
623	Najafgarh Terminal	28.614614000000003	76.978024
653	Kalyan Vihar Terminal	28.693527000000003	77.199123
182	Mori Gate Terminal	28.666217	77.221928
847	West Enclave Terminal	28.691667	77.101167

b)

route_id	stop_name	stop_lat	stop_lon
1055	Old Delhi Railway Station	28.660087	77.227876
91	Satyapuram Jharoda	28.663282000000002	76.944684
687	Kendriya Terminal (Church Road)	28.617308	77.203983
743	GTK Depot	28.731116999999998	77.158967
354	Uttam Nagar Terminal	28.625329	77.066948
233	Rewla Khanpur Depot	28.568610999999997	76.97664
291	Dichau Kalan Depot	28.629315000000002	76.997763
436	Hari Nagar Shaheed Pawan Sahni Chowk	28.6252	77.1108

Q3

find trip_ids going through stop_id = 469

v1 = select trip_id from stop_times where stop_id = 469

find corresponding route_ids

v2 = select route_id from trips, v1 where trips.trip_id = v1.trip_id

find trip_ids corresponding to route_ids

v3 = select route_id,trip_id from v2, trips where v2.route_id = trips.route_id

find route_ids and stop_sequences

v4 = select route_id, stop_times.stop_sequence from stop_times,v3 where v3.trip_id = stop_times.trip_id

find length of these routes.

v5 = select route_id, max(stop_sequence) from v4 groupby route_id

Find distance of shortest route through Govind Puri Metro Station stop id 469

minseq = select min(stop_sequence) from v5

Find 1 route that has that shortest distance

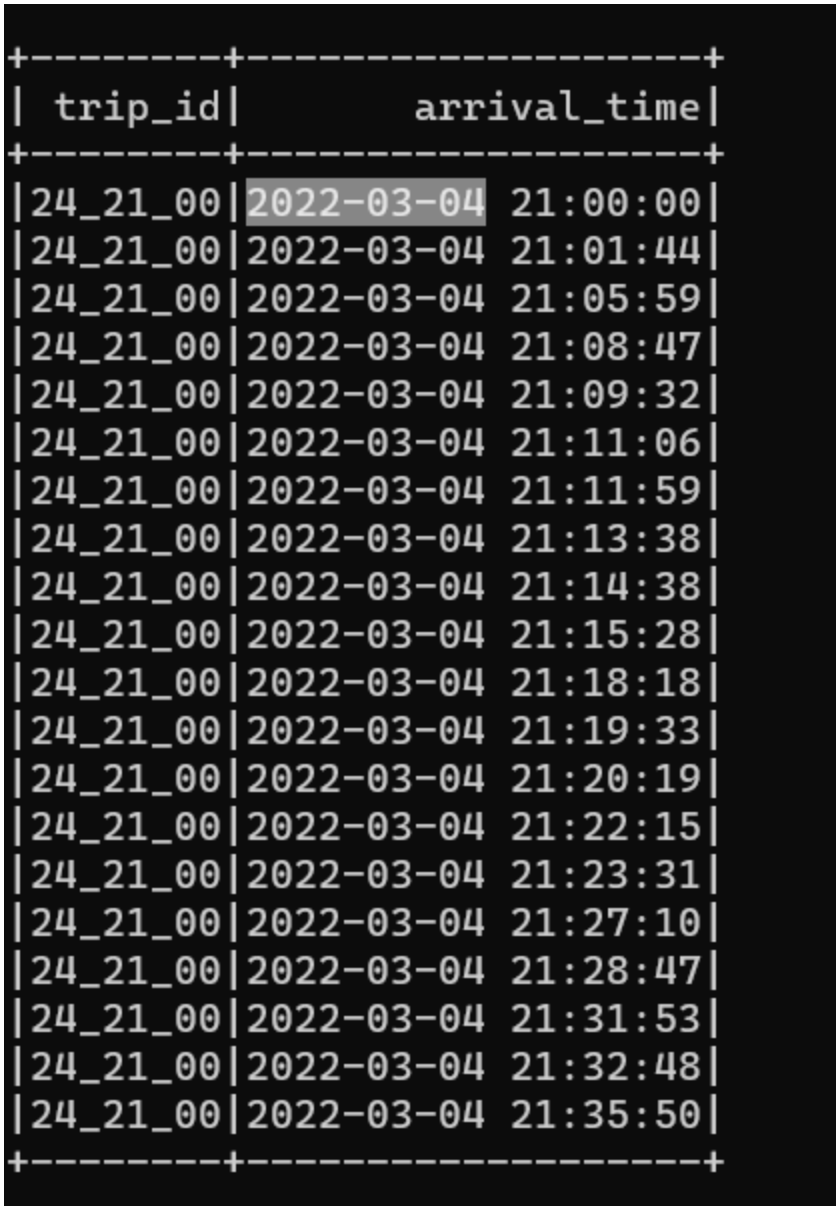
v6 = (select route_id from v5 where minseq = stop_sequence)limit 1

Find corresponding trips

v7 = select trip_id from trips where (route_id in v6)

Find arrival times of these trips

v8 = select trip_id, arrival_time from stop_times where(trip_id in v7)

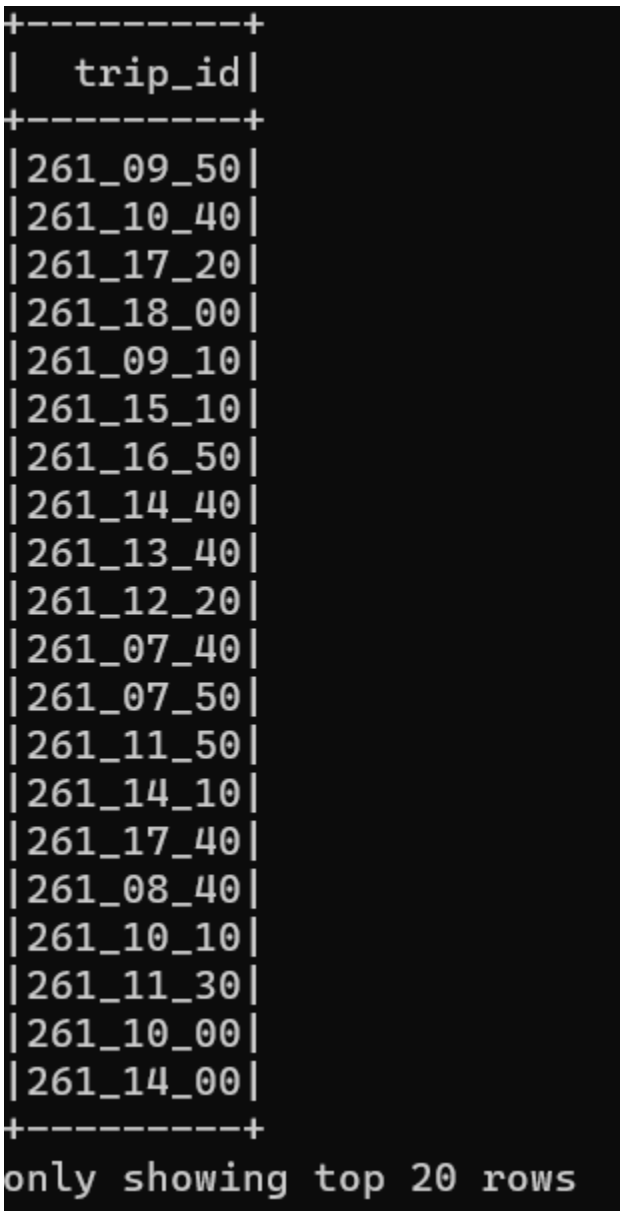


```
+--------+-------------------+
| trip_id|       arrival_time|
+--------+-------------------+
|24_21_00|2022-03-04 21:00:00|
|24_21_00|2022-03-04 21:01:44|
|24_21_00|2022-03-04 21:05:59|
|24_21_00|2022-03-04 21:08:47|
|24_21_00|2022-03-04 21:09:32|
|24_21_00|2022-03-04 21:11:06|
|24_21_00|2022-03-04 21:11:59|
|24_21_00|2022-03-04 21:13:38|
|24_21_00|2022-03-04 21:14:38|
|24_21_00|2022-03-04 21:15:28|
|24_21_00|2022-03-04 21:18:18|
|24_21_00|2022-03-04 21:19:33|
|24_21_00|2022-03-04 21:20:19|
|24_21_00|2022-03-04 21:22:15|
|24_21_00|2022-03-04 21:23:31|
|24_21_00|2022-03-04 21:27:10|
|24_21_00|2022-03-04 21:28:47|
|24_21_00|2022-03-04 21:31:53|
|24_21_00|2022-03-04 21:32:48|
|24_21_00|2022-03-04 21:35:50|
+--------+-------------------+
```

Q4

For every trip,

totaltime = (max departure time)- (min arrival time)



```
+---------+
|  trip_id|
+---------+
|261_09_50|
|261_10_40|
|261_17_20|
|261_18_00|
|261_09_10|
|261_15_10|
|261_16_50|
|261_14_40|
|261_13_40|
|261_12_20|
|261_07_40|
|261_07_50|
|261_11_50|
|261_14_10|
|261_17_40|
|261_08_40|
|261_10_10|
|261_11_30|
|261_10_00|
|261_14_00|
+---------+
only showing top 20 rows
```

Q5

find all trip_ids that go through Govind Puri Metro Station stop id 469

t1 = select trip_id from stop_times where stop_id = 469

find all routes that go through these trips

t2 = select distinct(route_id) from t1, trips where trips.trip_id = t1.trip_id

Find all trip_ids corresponding to these route_ids

t3 = select trip_id from t2, trips where trips.route_id == t2.route_id

Find all stop_ids corresponding to these trip_ids

t4 = select stop_times.stop_id from stop_times, t3 where stop_times.trip_id = trips.trip_id

find stop_ids not in t4

t5 = select distinct(stop_id) from t4 where stop_id not in stops

find stop names corresponding to stop_id s in t5

t6 = select stops.stop_name from t5, stops where stops.stop_id in t5

5.

stop_name
Narela Terminal
Police Station Narela
Safiyabad Crossing
Ramdev Chowk Pithori Jhori
Narela A-6 / CPJ College
State Bank Of Allahbad
Sec A-9 Narela
Sec A-9 and A-6 Narela
A-7 Narela Sec- 10A Pocket- 66
Harish Chandra Hospital
Kasturi Ram School
Munim Ji Ka Bagh
New Anaj Mandi
Kurani More
Prem Nagar Narela
Maharaja Agrassen School
Bharat Mata School
Kaushal Devi Netraheen Ashram
Sannoth Crossing / Ghoga Crossing
Delhi Jal Board Bawana

only showing top 20 rows

Q6

stop_id	stop_name
111	Old Delhi Railway Station
706	Uttam Nagar Terminal
770	Anand Vihar ISBT Terminal

How to install KUDU-pyspark on ubuntu 21 using docker ?

1. type git clone "https://github.com/ThinkBigEg/kudu-pyspark"

2. type cd kudu-pyspark

3.

3.1Install docker by following this video https://www.youtube.com/watch?v=akUl6iSroiI till 7:03

3.2 Verify using sudo docker run hello-world

4. Make changes in Dockerfile

```
        echo "conda activate kudu" >> ~/.bashrc

# Adding Kudu Cloudera Repository and installing Kudu
RUN echo "deb [check-valid-until=no] http://cdn-fastly.deb.debian.org/debian jessie main" > /etc/apt/sources.list.d/jessie.list
RUN echo "deb [check-valid-until=no] http://archive.debian.org/debian jessie-backports main" > /etc/apt/sources.list.d/jessie-backports.list
RUN sed -i '/deb http:\/\/deb.debian.org\/debian jessie-updates main/d' /etc/apt/sources.list
RUN apt-get -o Acquire::Check-Valid-Until=false update
RUN     wget -qO - http://archive.cloudera.com/kudu/ubuntu/trusty/amd64/kudu/archive.key | apt-key add - && \
        wget -P /etc/apt/sources.list.d/ "http://archive.cloudera.com/kudu/ubuntu/trusty/amd64/kudu/cloudera.list" && \
        apt-get -o Acquire::Check-Valid-Until=false update -y && \
        apt-get -y install kudu kudu-master kudu-tserver libkuduclient0 libkuduclient-dev

#Referencing Pyspark to the virtual environment named kudu
RUN     echo "/spark/python" >> /opt/conda/envs/kudu/lib/python3.7/site-packages/pyspark.pth && \
        mkdir -p /opt/spark/notebooks
```

5. Build the image

```
docker build . --tag kudu-python
```

6.Install Compose on Linux systems

6.1 Run this command to download the current stable release of Docker Compose:

```
$ sudo curl -L
"https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(u
name -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

6.2Apply executable permissions to the binary:

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

6.3 Test the installation.

```
$ docker-compose --version
```

Reference:https://docs.docker.com/compose/install/

7. Build the containers

```
docker-compose up -d
```

8. From here, follow instructions in readme of https://github.com/ThinkBigEg/kudu-pyspark