IMPORTANT: THIS GUIDE IS FOCUSED ON LINUX UBUNTU. YOU MIGHT USE OTHER OS AND CORRESPONDING COMMANDS. BUT UBUNTU IS RECOMMENDED.

# PREREQUISITES

First you need to install some tools to save time during practice session.

* Open terminal and update your OS. Execute following
  + sudo apt-get update.
* Get Java
  + detailed guide is [here](https://computingforgeeks.com/install-oracle-java-openjdk-14-on-ubuntu-debian-linux/).
* Get docker ce
  + detailed guide is [here](https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04).
* Get docker compose
  + detailed guide is [here](https://www.digitalocean.com/community/tutorials/how-to-install-docker-compose-on-ubuntu-20-04-quickstart).

# ENVIRONMENT SETUP

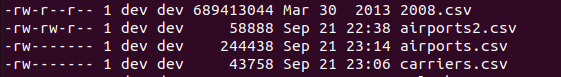
* Open terminal and create directory msdp-hive
  + *mkdir msdp-hive* (I am creating it in ~/dev directory)
* In terminal go to the new directory you created:
  + *cd ~/dev/msdp-hive/*
* In terminal create directory bde2020
  + *mkdir bde2020*
* open directory you’ve just created
  + *cd bde2020*
* Now we will need to clone repository. We are going to use [BDE](https://hub.docker.com/r/bde2020/hive/) (big data Europe distro).
* In terminal type:
  + *git clone* [*https://github.com/big-data-europe/docker-hive.git*](https://github.com/big-data-europe/docker-hive.git)
* Now type following in terminal
  + *ls*
* You need to see that you have new directory created

**

* Now go to that directory
  + *cd docker-hive*
* Now let’s download data we are going to use in our lab
  + *wget -q -O - "* *https://drive.google.com/file/d/18YTS2FNaetDgz5NNwMqmJMScSeiOaqkd/view?usp=sharing" | tar -xzf - -C ~/home/dev/msdp-hive/bde2020/docker-hive*
* Check if you have data directory
  + *ls*

**

* Check if you have all files there
  + *Ls -la data*

**

* Now please open docker-compose.yml in your favorite text editor
* Find hive-server section line 24 and add mapping for docker

**

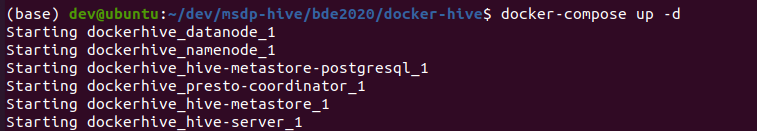
*volumes:*

*- /home/dev/dev/msdp-hive/bde2020/docker-hive/data:/opt/hive/examples/msdpfiles/*

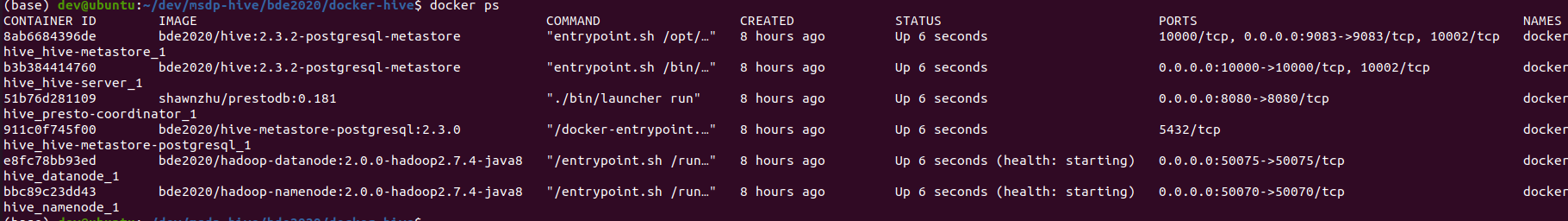
* We are almost done with environment. We need presto installed. Please go to msdp-hive folder and create presto directory. I have it here *~/dev/msdp-hive/presto*
* Install presto into presto directory. Guide is [here](https://prestosql.io/docs/current/installation/cli.html).
* Okay we are done with environment setup. Now let’s start working with hive.

# First ropes with Hive

* Let’s start environment we created. Open docker-hive directory in terminal and run
  + docker-compose up -d



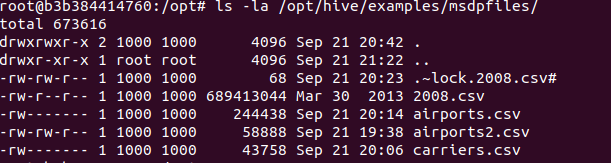
* Let’s verify docker images are actually running
  + docker ps



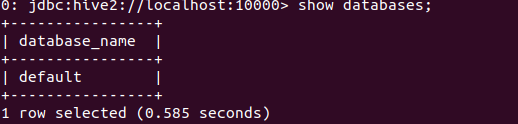
* Now let’s connect to hive server docker
  + *docker-compose exec hive-server bash*



* Let’s check you’ve connected files
  + *ls -la /opt/hive/examples/msdpfiles/*



* Let’s run hive CLI
  + beeline -u jdbc:hive2://localhost:10000
* Type show databases;



* Now let’s create msdphive database



* Let’s open it
  + Use msdphive;



* Now let’s create and load data;

## LOAD DATA

Flight data table

CREATE TABLE flight\_data(

year INT,

month INT,

day INT,

day\_of\_week INT,

dep\_time INT,

crs\_dep\_time INT,

arr\_time INT,

crs\_arr\_time INT,

unique\_carrier STRING,

flight\_num INT,

tail\_num STRING,

actual\_elapsed\_time INT,

crs\_elapsed\_time INT,

air\_time INT,

arr\_delay INT,

dep\_delay INT,

origin STRING,

dest STRING,

distance INT,

taxi\_in INT,

taxi\_out INT,

cancelled INT,

cancellation\_code STRING,

diverted INT,

carrier\_delay STRING,

weather\_delay STRING,

nas\_delay STRING,

security\_delay STRING,

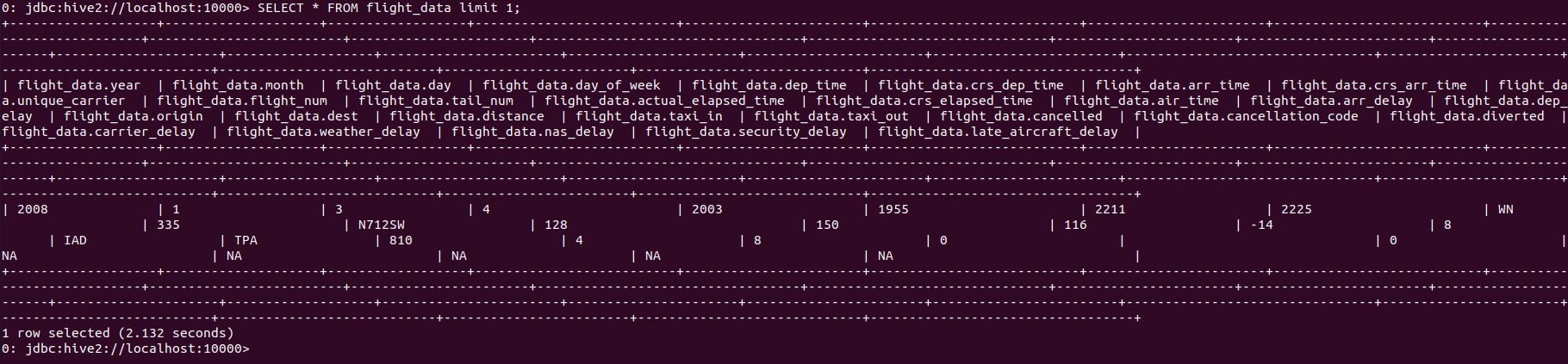
late\_aircraft\_delay STRING

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',';

LOAD DATA LOCAL INPATH '/opt/hive/examples/msdpfiles/2008.csv' OVERWRITE INTO TABLE flight\_data;



AIRPORTS TABLE

CREATE TABLE airports (

name STRING,

country STRING,

area\_code INT,

code STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ',';

LOAD DATA LOCAL INPATH '/opt/hive/examples/msdpfiles/airports2.csv' OVERWRITE INTO TABLE airports;



Carriers Table

CREATE TABLE carriers(

code STRING,

description STRING

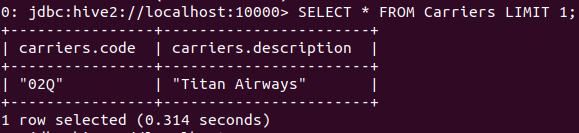
)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

tblproperties ("skip.header.line.count"="1");

LOAD DATA LOCAL INPATH '/opt/hive/examples/msdpfiles/carriers.csv' OVERWRITE INTO TABLE carriers;



## BASIC ANALYSIS

* Find most popular carrier

SELECT unique\_carrier, flight\_count, cc.description FROM

(

SELECT unique\_carrier, Count(\*) flight\_count FROM flight\_data f GROUP BY unique\_carrier

) fd

JOIN carriers cc on cc.code = fd.unique\_carrier

ORDER BY flight\_count DESC

LIMIT 3;

* Find 10 worst airports with highest average delay

SELECT

name,

avgDelay

FROM (

SELECT

name,

AVG(arr\_delay) avgDelay

FROM

flight\_data f

INNER JOIN airports a

ON (f.origin=a.code)

GROUP BY name

) a

ORDER BY avgDelay DESC

LIMIT 10;

* Most busy days

SELECT day\_of\_week, count(\*) cc from flight\_data f

WHERE month > 5 and month < 9

GROUP BY day\_of\_week

ORDER BY cc DESC

SELECT day\_of\_week, count(\*) cc from flight\_data f

GROUP BY day\_of\_week

ORDER BY cc DESC

* Flights inside US

WITH us\_flights AS (

SELECT COUNT(\*) cc1 FROM flight\_data f JOIN airports a on f.dest = a.code WHERE a.area\_code = 67

UNION

SELECT COUNT(\*) cc1 FROM flight\_data f JOIN airports a on f.origin = a.code WHERE a.area\_code = 67

) SELECT SUM(cc1) from us\_flights

* Cancelled us flights carriers

WITH us\_flights AS (

SELECT unique\_carrier, COUNT(\*) cc1 FROM flight\_data f WHERE f.cancelled = 1 GROUP BY unique\_carrier

) SELECT u.cc1, cc.description from us\_flights u JOIN carriers cc on cc.code = u.unique\_carrier ORDER BY u.cc1 DESC LIMIT 10

analyze table flight\_data compute statistics;

analyze table flight\_data compute statistics for columns;

desc formatted table\_name;

desc formatted table\_name.id;

desc formatted table\_name.id partition(time='2016-05-27');

sudo nmap -v -p 12345 127.0.0.1

echo -e '\x1dclose\x0d' | telnet 127.0.0.1 12345

## SAMPLING AND BUCKETING

SELECT \* FROM flight\_data TABLESAMPLE(BUCKET 1 OUT OF 4 ON unique\_carrier) LIMIT 100;

## PARTITIONING

SET hive.exec.dynamic.partition.mode=nonstrict;

CREATE EXTERNAL TABLE flight\_data\_by\_day\_of\_week(

year INT,

month INT,

day INT,

dep\_time INT,

crs\_dep\_time INT,

arr\_time INT,

crs\_arr\_time INT,

unique\_carrier STRING,

flight\_num INT,

tail\_num STRING,

actual\_elapsed\_time INT,

crs\_elapsed\_time INT,

air\_time INT,

arr\_delay INT,

dep\_delay INT,

origin STRING,

dest STRING,

distance INT,

taxi\_in INT,

taxi\_out INT,

cancelled INT,

cancellation\_code STRING,

diverted INT,

carrier\_delay STRING,

weather\_delay STRING,

nas\_delay STRING,

security\_delay STRING,

late\_aircraft\_delay STRING

)

PARTITIONED BY (day\_of\_week int)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/hive/warehouse/flight\_data\_partition\_by\_day\_of\_week';

INSERT OVERWRITE TABLE flight\_data\_by\_day\_of\_week PARTITION(day\_of\_week)

SELECT

year,

month,

day,

dep\_time,

crs\_dep\_time,

arr\_time,

crs\_arr\_time,

unique\_carrier,

flight\_num,

tail\_num,

actual\_elapsed\_time,

crs\_elapsed\_time,

air\_time,

arr\_delay,

dep\_delay,

origin,

dest,

distance,

taxi\_in,

taxi\_out,

cancelled,

cancellation\_code,

diverted,

carrier\_delay,

weather\_delay,

nas\_delay,

security\_delay,

late\_aircraft\_delay,

day\_of\_week

FROM

flight\_data;

SELECT day\_of\_week, count(\*) cc from flight\_data\_by\_day\_of\_week f

GROUP BY day\_of\_week

ORDER BY cc DESC

## Bucketing

CREATE EXTERNAL TABLE flight\_data\_bucket\_by\_carrier(

year INT,

month INT,

day INT,

day\_of\_week INT,

dep\_time INT,

crs\_dep\_time INT,

arr\_time INT,

crs\_arr\_time INT,

unique\_carrier STRING,

flight\_num INT,

tail\_num STRING,

actual\_elapsed\_time INT,

crs\_elapsed\_time INT,

air\_time INT,

arr\_delay INT,

dep\_delay INT,

origin STRING,

dest STRING,

distance INT,

taxi\_in INT,

taxi\_out INT,

cancelled INT,

cancellation\_code STRING,

diverted INT,

carrier\_delay STRING,

weather\_delay STRING,

nas\_delay STRING,

security\_delay STRING,

late\_aircraft\_delay STRING

)

CLUSTERED BY (unique\_carrier) INTO 4 BUCKETS

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LOCATION '/user/hive/warehouse/flight\_data\_bucket\_by\_carrier';

SET hive.enforce.bucketing = true;

INSERT OVERWRITE TABLE flight\_data\_bucket\_by\_carrier

SELECT

\*

FROM

flight\_data;

SET hive.enforce.bucketing = false;

SELECT

\*

FROM

flight\_data\_b TABLESAMPLE(BUCKET 1 OUT OF 4 ON unique\_carrier) LIMIT 100;

## OPTIMIZATION

set hive.vectorized.execution = true

set hive.vectorized.execution.enabled = true

set hive.cbo.enable=true;

set hive.compute.query.using.stats=true;

set hive.stats.fetch.column.stats=true;

set hive.stats.fetch.partition.stats=true;

UDF

delete jar /opt/hive/examples/msdpfiles/msdp-hive-udf-1.0.jar;

add jar /opt/hive/examples/msdpfiles/msdp-hive-udf-1.0.jar;

create temporary function DayOfWeekUDF as 'com.epam.GetDayName';

DROP TEMPORARY FUNCTION IF EXISTS DayOfWeekUDF;

SELECT DayOfWeekUDF(day\_of\_week), count(\*) cc from flight\_data\_by\_day\_of\_week f

GROUP BY day\_of\_week

ORDER BY cc DESC

## PRESTO

./presto.jar --server localhost:8080 --catalog hive --schema msdphive