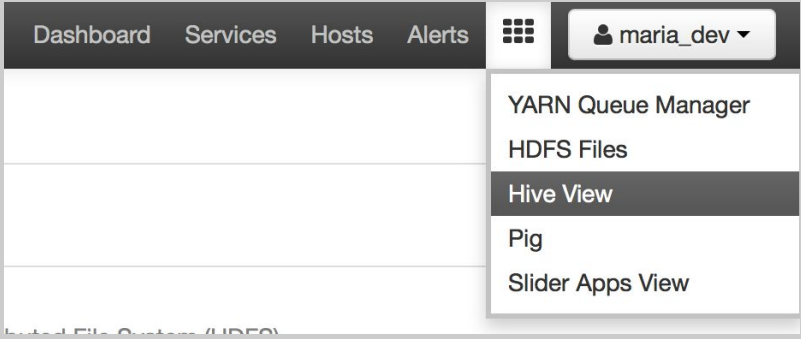


1.	Download and Start Hortonworks Virtual Environment
1.1	Install virtualbox: https://www.virtualbox.org/wiki/Downloads
1.3	Go to http://hortonworks.com/products/hortonworks-sandbox/#install and download HDP 2.2.3
1.4	Start up the virtual environment
1.5	<p>Go to localhost:8080 and log in to Ambari - a hadoop monitoring tool. login: admin password: admin</p> <p>This will give you an overview of your cluster.</p> <p>If the login is different you can look it up at: localhost:8888</p> <p>For 2.4 the following credentials should be used: login: maria_dev password: maria_dev</p>
2.	Process Data with Apache Hive
	<p>Open a command prompt and SSH to it</p> <ul style="list-style-type: none"> - ssh root@127.0.0.1 -p 2222; <p>The default password is hadoop (note that if you've already logged onto the machine then you were forced to pick a new password)</p>
	<p>You are now logged in as root. Switch to the user hdfs with the command: su hdfs</p> <p>With the command hdfs dfsadmin -report you can get an overview of the current health of your cluster</p>
2.1	<p>Create a text file named movies.txt on the virtual machine</p> <ul style="list-style-type: none"> - go to the home folder of your user: cd /home/hdfs - enter vi by typing: vi movies.txt - copy and paste the following content <p>1,The Nightmare Before Christmas,1993,3.9,4568 2,The Mummy,1932,3.5,4388 3,Orphans of the Storm,1921,3.2,9062 4,The Object of Beauty,1991,2.8,6150</p>

	<p>5,Night Tide,1963,2.8,5126 6,One Magic Christmas,1985,3.8,5333 7,Muriel's Wedding,1994,3.5,6323 8,Mother's Boys,1994,3.4,5733 9,Nosferatu: Original Version,1929,3.5,5651 10,Nick of Time,1995,3.4,5333</p> <ul style="list-style-type: none"> - press the escape key and then type :wq to save the file (wq stands for write quit, to exit without saving type :q!)
2.2	<p>First make an extra directory under /user on the Hadoop file system hadoop fs -mkdir /user/hadoop</p> <p>Copy the file into hadoop with the command : hadoop fs -put movies.txt /user/hadoop/movies.txt</p> <p>The put command will only insert, updating an existing file will require passing an extra parameter -f: hadoop fs -put -f movies.txt /user/hadoop/movies.txt</p> <p>Some other commands:</p> <ul style="list-style-type: none"> - make directory : hadoop fs -mkdir /dir/dir2 - remove: hadoop fs -rm /dir1/dirOrFilename
2.3	<p>Type pig to go to the grunt command shell</p> <p>You should see something like >grunt You can exit the grunt shell by typing: quit</p> <p>The shell has auto-completion for commands and variable names, use the tab key.</p> <p>Load the content of movies.txt into a variable named movies</p> <ul style="list-style-type: none"> - movies = LOAD '/user/hadoop/movies.txt' USING PigStorage(',') as (id,name,year,rating,duration); - Or if you want to assign types: movies = LOAD '/user/hadoop/movies.txt' USING PigStorage(',') as (id:int,name:chararray,year:int,rating:float, duration:int); <p>To see the contents of the variable movies type: DUMP movies; <i>It is only now that the variable movies will be processed and a map/reduce job will be generated.</i></p> <p><i>If you made a typo - only now you will discover it :-)</i></p> <p>If you see an error, quit the grunt shell and go to the /home/hdfs folder, there you will see the log files.</p>

2.4	To check the format of the variable movies, type: Describe movies;
2.6	<p>Time to filter some data.</p> <p>Let us filter movies with a rating > 3.5</p> <p>With the following command: movies_greater_than_three_point_five = FILTER movies by rating > 3.5;</p> <p>Let's extract the values for year, rating and name and save them in another variable 'foreachexample'</p> <p>Enter: foreachexample = foreach movies_greater_than_three_point_five generate year,rating,name;</p> <p>Now let us evaluate the variables: DUMP foreachexample;</p> <p>Two records should be returned</p>
2.7	<p>We can also store the values of a variable.</p> <p>STORE movies_greater_than_three_point_five INTO '/user/hadoop/movies_greater_than_three_point_five' USING PigStorage(',');</p>
2.8	<p>Now that the data is stored in HDFS you can use the cat command to access this.</p> <p>cat /user/hadoop/movies_greater_than_three_point_five/part-m-00000</p>
2.9	<p>PIG grunt has commands which can run on HDFS and on the local file system.</p> <pre> grunt> cat /user/hadoop/movies.txt grunt> ls /user/hadoop/ grunt> cd /user/ grunt> ls grunt> cd /user/hadoop grunt> ls grunt> copyToLocal /user/hadoop/movies.txt /home/ grunt> pwd </pre>

3.	<h2>Process Data with Apache Hive</h2>
	<p>For Hive you can use the command line: just quit the pig console and type hive.</p> <p>But you can also easily use the Hive worksheet in the Ambari console. Just open your browser on localhost:8080 => use admin / admin to log in (or maria_dev / maria_dev).</p> <p>Next you can open the Hive view via the icon right to "Admin" and left to "Admin user"</p>  <p>(That icon shows you the different views - also hive, tez, ...)</p> <p>For Hive we will be using the browser interface.</p>
3.1	<p>We will first create a user/admin folder.</p> <p>Open a ssh connection via <code>ssh root@127.0.0.1 -p 2222;</code></p> <p>Verify via HDFS Files whether a directory for your user exists under /user and create it if necessary (admin or maria_dev)</p> <p>Then switch to hdfs user with: <code>sudo su - hdfs</code></p> <p>Create a /user/admin or user/maria_dev folder via: <code>hdfs dfs -mkdir /user/admin</code></p> <p>Set the correct rights on the folder: <code>hdfs dfs -chown root:hdfs /user/admin</code></p> <p>Then create a local file on your machine named movies.txt with content:</p> <ul style="list-style-type: none"> - 1,The Nightmare Before Christmas,1993,3.9,4568 2,The Mummy,1932,3.5,4388 3,Orphans of the Storm,1921,3.2,9062 4,The Object of Beauty,1991,2.8,6150

	<p>5,Night Tide,1963,2.8,5126 6,One Magic Christmas,1985,3.8,5333 7,Muriel's Wedding,1994,3.5,6323 8,Mother's Boys,1994,3.4,5733 9,Nosferatu: Original Version,1929,3.5,5651 10,Nick of Time,1995,3.4,5333</p>
3.2	<p>Go to localhost:8080, login as admin / admin (or maria_dev)</p> <p>Navigate to HDFS files view: this view can be used to upload files into HDFS. Go to the user/admin folder you just created. (It might be possible that you need to change the write permissions on the folder, right click on the admin line and select permissions, make sure that write is enabled)</p> <p>Upload the movies.txt file into the folder</p>
3.3	<p>Now open the Hive view, by clicking on the hive button just like the HDFS view</p> <p>On the right you can see the query editor which allows you to enter queries. At the bottom you have buttons to execute, explain and save the query - you can also open a new worksheet.</p> <p>The first task we will do is create a table to hold the data.</p> <p>Type in the query: create table temp_movies (col_value STRING); No data will be returned as it is just an empty table.</p> <p>Once the query has been executed you can refresh the "database explorer" on the left. This will allow you to see the newly created table under the default database.</p> <p>By clicking on the icon next to the table name a new worksheet will open and a query will be launched to show you the data of the table.</p>
3.4	<p>Time to load in some data.</p> <p>Execute the following statement: LOAD DATA INPATH "/user/admin/movies.txt" OVERWRITE INTO TABLE default.temp_movies;</p> <p>By re-executing the select query you can see that the lines have been loaded in.</p>

	<pre>select * from default.temp_movies;</pre> <p>You can also switch to the default database so you no longer have to type in the default. in front of the table name via use default;</p> <p>Please note that LOAD will move the data to the hive file storage => it will no longer be present in the /user/admin.</p> <p>You can prevent this behaviour by creating an EXTERNAL table:</p> <pre>create EXTERNAL table temp_movies (col_value STRING);</pre> <p>External tables are a better way for loading in data if you do not want to move the files into the hive storage and just store the (intermediate) results for example.</p>
3.5	<p>Now we will transform the data in a more correct format - as all text in 1 column is hardly convenient to query.</p> <pre>create table default.movies (id INT, name STRING, year INT, rating FLOAT, duration INT);</pre> <p>Then we will insert the data into the new table.</p> <p>We can use a regular expression for this:</p> <pre>insert overwrite table movies SELECT regexp_extract(col_value, '^(:([,]*)\,){1}', 1) id, regexp_extract(col_value, '^(:([,]*)\,){2}', 1) name, regexp_extract(col_value, '^(:([,]*)\,){3}', 1) year, regexp_extract(col_value, '^(:([,]*)\,){4}', 1) rating, regexp_extract(col_value, '^(:([,]*)\,){5}', 1) duration from temp_movies;</pre> <p>You can also create a table with a predefined delimiter:</p> <pre>create table default.movies3 (id INT, name STRING, year INT, rating FLOAT, duration INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY ",";</pre> <p>Re-upload the file to the /user/admin folder via the HDFS view.</p> <p>Then LOAD DATA INPATH "/user/admin/movies.txt" OVERWRITE INTO TABLE default.movies3;</p> <p>You will now see that the columns have been filled in.</p>

	<p>important: if your delimiter also appears within columns you will have to preload the data which you want to escape with a correct escape character.</p> <p>ROW FORMAT DELIMITED FIELDS TERMINATED BY "," ESCAPED BY '\\';</p> <p>will accept the following line as 4 fields: 1,some text\\, with comma in it,123,more text</p> <p>(but then your data has to be prepared with these escape characters)</p>
3.6	<p>Time to filter some data.</p> <p>Let us filter movies with a rating > 3.5 select * from movies where rating > 3.5;</p> <p>You can also use group by statements: select year, max(rating) from movies group by year;</p> <p>and joins SELECT a.year, a.name, a.rating from movies a JOIN (SELECT year, max(rating) rating FROM movies GROUP BY year) b ON (a.year = b.year AND a.rating = b.rating) ;</p>
3.7	<p>To get an idea how hive jobs are being generated (like the group by)</p> <p>Go to a new view: open the Tez view Next click on one of the DAG names of a successful job.</p> <p>There you can open a graphical representation of the jobs which have been executed on the data.</p>