Task 1 – Installation

Set up a 3-node (at least) cluster of the latest Hortonworks Data Platform via Ambari installation. You'll need to evaluate which components are actually required based on the rest of the exercise. Utilize whatever technologies you have at your disposal to do so. It is highly recommended that this is done using Amazon AWS services, but local laptop (Mac, PC, etc) are fine also if you feel comfortable using technologies such as VirtualBox.

|  |
| --- |
| - Done in my local machine |

Task 2 – Hadoop Confidence Test

The following will demonstrate that you are able to maneuver around different parts of the cluster and operate both within Ambari and without it.

HDFS

Create a folder called consumercomplaints.

|  |
| --- |
| sudo -u hdfs hdfs dfs -mkdir consumercomplaints  sudo -u hdfs hdfs dfs -chown hdpuser:hdpadmin /consumercomplaints |

Use the following link to download a CSV file from S3 and place it into your new Hadoop cluster in the folder you created above.

|  |
| --- |
| Downloaded the file to windows and copied it to first Hadoop VM using winscp. Found data is not consistent with the schema that we have created. Took some efforts to get the data loaded to Hive schema that we have created  hdfs dfs -put Consumer\_Complaints.csv /consumercomplaints |

Change permissions on the file so that it is readable by the world, but only can be updated by the owner.

|  |
| --- |
| hdfs dfs -chmod 644 /consumercomplaints/Consumer\_Complaints.csv |

HIVE

Utilizing the field names provided in the CSV file above, create an external table over the file in the consumercomplaints folder called consumercomplaints. Create another table called consumercomplaints\_partitioned and partition the table by state.

|  |
| --- |
| CREATE EXTERNAL TABLE consumercomplaints (Date\_received string,Product string,Sub\_product string,Issue string,Sub\_issue string,Consumer\_complaint\_narrative string,Company\_public\_response string,Company string,State string,ZIP\_code string,Tags string,Consumer\_consent\_provided string,Submitted\_via string,Date\_sent\_to\_company string,Company\_response\_to\_consumer string,Timely\_response string,Consumer\_disputed string,Complaint\_ID double)  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'  with serdeproperties (  "separatorChar" = ",",  "quoteChar" = "\"",  "escapeChar" = "\\"  )  STORED AS TEXTFILE  LOCATION '/consumercomplaints'; |

Query your new tables to find the following:

1. Total number of complaints for each company

|  |
| --- |
| *select count(\*) from consumercomplaints;* |

1. Total number of Disputed vs. Undisputed consumer complaints

|  |
| --- |
| *select Consumer\_disputed, count(Consumer\_disputed) from consumercomplaints group by Consumer\_disputed;* |

1. All consumer complaints centered around Mortgages that were being foreclosed upon from Bank of America

|  |
| --- |
| *select \* from consumercomplaints where product='Mortgage' and company='Bank of America' and Issue like '%foreclosure%';* |

Create a new transactional table called consumercomplaints\_new and insert the data from the consumercomplaints table.

|  |
| --- |
| *create table consumercomplaints\_new as select \* from consumercomplaints;* |

Perform the following:

1. Copy all rows that have state = ‘NC’ or state = ‘TX’ out of the consumercomplaints\_new table into a separate table

|  |
| --- |
| *create table consumercomplaints\_nc\_tx as select \* from consumercomplaints\_new where state ='NC' or state = 'TX';* |

1. Delete all rows out of the consumercomplaints\_new table that have state = ‘NC’ or state = ‘TX’

|  |
| --- |
| *insert overwrite table consumercomplaints\_new select \* from consumercomplaints\_new where state <>'NC' and state <> 'TX';* |

1. Insert only the rows where state = ‘NC’ into the consumercomplaints\_new table from the temporary table above

|  |
| --- |
| *insert into table consumercomplaints\_new select \* from consumercomplaints\_nc\_tx where state ='NC';* |

General Hadoop Administration

One of your data nodes has crashed and is not recoverable. Demonstrate how you would decommission the node so that no jobs are run against it.

|  |
| --- |
|  |

Once fully decommissioned, show how you would bring that node back into the cluster and rebalance the cluster.

|  |
| --- |
|  |

Demonstrate changing the namenode java heap size in both Ambari as well as outside of Ambari via the command line.

|  |
| --- |
| *HDFS -> Configs -> NameNode Java heap size -> Edit*  *mapred-site.xml*  *<property>      <name>mapred.child.java.opts</name>      <value>-Xmx4096m</value>  </property>*  *hadoop-env.sh:*  *export HADOOP\_OPTS="<value>"* |

Install the Falcon service on your management node via Ambari and show that it will start successfully.

|  |
| --- |
| *Actions -> Add Service -> Falcon*  *(If oozie is not istalled, it will prompt to install oozie and you need to install oozie as Falcon needs oozie)* |

Sqoop

Utilizing the following mysql instance, sqoop the tables you deem are relevant in order to complete the below task. Sqoop them into HCatalog tables in a new database in Hive called AdventureWorks.

Database Server: hadooptest.cyrjpmsirks6.us-east-1.rds.amazonaws.com

Login: HadoopTest

Password: CanUSq00p1T?!

Schema: AdventureWorks

Write a query to determine the total sales for each product within ProductModel 'Racing Socks'.

Sol: After checking mysql tables in database server, I observed that there are three tables needs to import from mysql to hive to answer the above question.

|  |
| --- |
| **HIVE:**  *CREATE TABLE* ***productmodel*** *(ProductModelID STRING, Name STRING, CatalogDescription STRING, Instructions STRING, rowguid STRING, ModifiedDate STRING) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'*  *with serdeproperties (*  *"separatorChar" = ",",*  *"quoteChar" = "\"",*  *"escapeChar" = "\\"*  *)*  *STORED AS TEXTFILE;*  *CREATE TABLE* ***product*** *(ProductID INT, Name varchar(50), ProductNumber varchar(25), MakeFlag tinyint, FinishedGoodsFlag tinyint, Color varchar(15), SafetyStockLevel smallint, ReorderPoINT smallint, StandardCost double, ListPrice double, Size varchar(5), SizeUnitMeasureCode varchar(3), WeightUnitMeasureCode varchar(3), Weight double, DaysToManufacture INT, ProductLine varchar(2), Class varchar(2), Style varchar(2), ProductSubcategoryID INT, ProductModelID INT, SellStartDate string, SellEndDate string, DiscontinuedDate string, rowguid string, ModifiedDate string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'with serdeproperties ( "separatorChar" = ",", "quoteChar" = "\"", "escapeChar" = "\\" )STORED AS TEXTFILE;*  *CREATE TABLE* ***transactionhistory*** *( TransactionID int, ProductID int, ReferenceOrderID int, ReferenceOrderLineID int, TransactionDate string, TransactionType varchar(1), Quantity int, ActualCost double, ModifiedDate string) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'*  *with serdeproperties (*  *"separatorChar" = ",",*  *"quoteChar" = "\"",*  *"escapeChar" = "\\"*  *)*  *STORED AS TEXTFILE;* |

Import the data using sqoop into hive tables.

|  |
| --- |
| **SQOOP**:  *sqoop import --driver com.mysql.jdbc.Driver --connect jdbc:mysql://54.156.179.10/adventureworks --username HadoopTest --password CanUSq00p1T?! --table productmodel --hcatalog-database inmar --hcatalog-table productmodel*  *sqoop import --driver com.mysql.jdbc.Driver --connect jdbc:mysql://54.156.179.10/adventureworks --username HadoopTest --password CanUSq00p1T?! --table product --hcatalog-database inmar --hcatalog-table product*  *sqoop import --driver com.mysql.jdbc.Driver --connect jdbc:mysql://54.156.179.10/adventureworks --username HadoopTest --password CanUSq00p1T?! --table transactionhistory --hcatalog-database inmar --hcatalog-table transactionhistory* |

|  |
| --- |
| **Query:**  *select sum(actualcost) from transactionhistory where productid in (select a.productid from product a, productmodel b*  *where a.productmodelid=b.productmodelid and b.name ='Racing Socks');* |

Oozie

Create a basic workflow to run one of the sqoop table jobs above and schedule it to run nightly at 1AM.

|  |
| --- |
| Initial Setup:  *sudo -u hdfs hdfs dfs -mkdir -p /project1/data/productmodel*  *sudo -u hdfs hdfs dfs -mkdir -p /project1/oozie*  *sudo -u hdfs hdfs dfs -chown -R hdpuser:hdpadmin /project1*  *sudo -u hdfs hdfs dfs -mkdir /user/hdpuser*  *sudo -u hdfs hdfs dfs -chown hdpuser:hdpadmin /user/hdpuser*  *export OOZIE\_URL="http://hn2.hadoop.com:11000/oozie/"*  *sudo chown -R hdpuser:hdpadmin /home/hdpuser/coord\_sqoop\_productmodel\_data*  *hdfs dfs -rm -r -skipTrash /project1/oozie/coord\_sqoop\_productmodel\_data*  *hdfs dfs -put /home/hdpuser/coord\_sqoop\_productmodel\_data /project1/oozie/*  *cd /home/hdpuser/coord\_sqoop\_productmodel\_data*  *oozie job -config job.coordinator.properties -run*  *oozie job -info <coordinator\_id>*  *hdfs dfs -ls /project1/data/productmodel*  *Found 2 items*  *drwxr-xr-x - hdpuser hdpadmin 0 2018-12-02 18:37 /project1/data/productmodel/2018-12-01*  *drwxr-xr-x - hdpuser hdpadmin 0 2018-12-02 18:40 /project1/data/productmodel/2018-12-02* |

Here is the attachment with code files that are created for setup:



|  |
| --- |
| **Status check:**  [hdpuser@hn2 ~]$ *oozie job -info 0000006-181202182421759-oozie-oozi-C*  **Job ID : 0000006-181202182421759-oozie-oozi-C**  **------------------------------------------------------------------------------------------------------------------------------------**  **Job Name : coord\_sqoop\_productmodel\_data**  **App Path : hdfs://hn1.hadoop.com:8020/project1/oozie/coord\_sqoop\_productmodel\_data**  **Status : RUNNING**  **Start Time : 2018-12-01 01:00 GMT**  **End Time : 2018-12-05 23:59 GMT**  **Pause Time : -**  **Concurrency : 1**  **------------------------------------------------------------------------------------------------------------------------------------**  **ID Status Ext ID Err Code Created Nominal Time**  **0000006-181202182421759-oozie-oozi-C@1 SUCCEEDED 0000007-181202182421759-oozie-oozi-W - 2018-12-02 13:06 GMT 2018-12-01 01:00 GMT**  **------------------------------------------------------------------------------------------------------------------------------------**  **0000006-181202182421759-oozie-oozi-C@2 SUCCEEDED 0000008-181202182421759-oozie-oozi-W - 2018-12-02 13:09 GMT 2018-12-02 01:00 GMT**  **------------------------------------------------------------------------------------------------------------------------------------** |

Task 3 - Streaming Architecture w/ Hive

Hive

1. Store the twitter data called sample\_twitter\_data in HDFS. You can use [this link](https://s3.inmar.com/dbaexercise/sample_twitter_data.txt) to download it. The data is in json format and should not be altered.

|  |
| --- |
| *[hdpuser@hn2 ~]$ hdfs dfs -mkdir /project1/twitter/*  *[hdpuser@hn2 ~]$ hdfs dfs -put sample\_twitter\_data.txt /project1/twitter/* |

1. Once the data is in HDFS, create an hcat/hive schema to be able to answer the following question: What are all the tweets by the twitter user "Aimee\_Cottle"? You will need to provide the query that answers this question.

*Hint: there are multiple ways to do this, the preferred method involves org.apache.hcatalog.data.JsonSerDe - if that doesn't work search for Json serde's in the www - there are some you can compile from source to get it to work*

|  |
| --- |
| *set hive.support.sql11.reserved.keywords=false;*  *CREATE EXTERNAL TABLE twitter (user struct<userlocation:string, id:string, name:string, screenname:string, geoenabled:string>, tweetmessage string, createddate string, geolocation string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' LOCATION '/project1/twitter/';*  hive> *select user.screenname,tweetmessage from twitter where user.screenname='Aimee\_Cottle';*  **OK**  **Aimee\_Cottle Gastroenteritis has pretty much killed me this week :( off work for a few days whilst I recover!**  **Time taken: 0.309 seconds, Fetched: 1 row(s)** |

Streaming

1. Implement a storm topology (<https://hortonworks.com/apache/storm/>) that streams in tweets (<https://dev.twitter.com/streaming/overview>), does some interesting analytics in real-time on the tweets, and then persists into HDFS.

…… Could not complete.