CCA175 Q&As

CCA Spark and Hadoop Developer Exam

Pass Cloudera CCA175 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.passapply.com/cca175.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Cloudera Official Exam Center

QUESTION 1

Problem Scenario 35 : You have been given a file named spark7/EmployeeName.csv (id,name). EmployeeName.csv E01,Lokesh E02,Bhupesh E03,Amit E04,Ratan E05,Dinesh E06,Pavan E07,Tejas E08,Sheela E09,Kumar E10,Venkat

1. Load this file from hdfs and sort it by name and save it back as (id,name) in results directory. However, make sure while saving it should be able to write In a single file.

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution:

Step 1 : Create file in hdfs (We will do using Hue). However, you can first create in local

filesystem and then upload it to hdfs.

Step 2 : Load EmployeeName.csv file from hdfs and create PairRDDs

val name = sc.textFile("spark7/EmployeeName.csv")

val namePairRDD = name.map(x=> (x.split(",")(0),x.split(",")(1)))

Step 3 : Now swap namePairRDD RDD.

val swapped = namePairRDD.map(item => item.swap)

step 4: Now sort the rdd by key.

val sortedOutput = swapped.sortByKey()

Step 5 : Now swap the result back

val swappedBack = sortedOutput.map(item => item.swap}

Step 6 : Save the output as a Text file and output must be written in a single file.

swappedBack. repartition(1).saveAsTextFile("spark7/result.txt")

QUESTION 2

Problem Scenario 39 : You have been given two files spark16/file1.txt 1,9,5 2,7,4 3,8,3 spark16/file2.txt 1,g,h 2,i,j 3,k,l Load these two tiles as Spark RDD and join them to produce the below results (l,((9,5),(g,h))) (2, ((7,4), (i,j))) (3, ((8,3), (k,l))) And write code snippet which will sum the second columns of above joined results (5+4+3).

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create tiles in hdfs using Hue.

Step 2 : Create pairRDD for both the files.

val one = sc.textFile("spark16/file1.txt").map{

_.split(",",-1) match {

case Array(a, b, c) => (a, (b, c))

} }

val two = sc.textFHe(Mspark16/file2.txt").map{

_.split(\\'7\-1) match {

case Array(a, b, c) => (a, (b, c))

} }

Step 3 : Join both the RDD. val joined = one.join(two)

Step 4 : Sum second column values.

val sum = joined.map {

case (_, ((_, num2), (_, _))) => num2.tolnt

}.reduce(_ + _)

QUESTION 3

Problem Scenario 89 : You have been given below patient data in csv format, patientID,name,dateOfBirth,lastVisitDate
1001,Ah Teck,1991-12-31,2012-01-20 1002,Kumar,2011-10-29,2012-09-20 1003,Ali,2011-01-30,2012-10-21
Accomplish following activities.

1.

Find all the patients whose lastVisitDate between current time and \\'2012-09-15\\'

2.

Find all the patients who born in 2011

3.

Find all the patients age

4.

List patients whose last visited more than 60 days ago

5.

Select patients 18 years old or younger

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1:

hdfs dfs -mkdir sparksql3

hdfs dfs -put patients.csv sparksql3/

Step 2 : Now in spark shell

// SQLContext entry point for working with structured data

val sqlContext = neworg.apache.spark.sql.SQLContext(sc)

// this is used to implicitly convert an RDD to a DataFrame.

import sqlContext.implicits._

// Import Spark SQL data types and Row.

import org.apache.spark.sql._

// load the data into a new RDD

val patients = sc.textFilef\\'sparksqIS/patients.csv")

// Return the first element in this RDD

patients.first()

//define the schema using a case class

case class Patient(patientid: Integer, name: String, dateOfBirth:String , lastVisitDate:

String)

// create an RDD of Product objects

val patRDD = patients.map(_.split(M,M)).map(p => Patient(p(0).tolnt,p(1),p(2),p(3)))

patRDD.first()

patRDD.count{}

// change RDD of Product objects to a DataFrame val patDF = patRDD.toDF()

// register the DataFrame as a temp table patDF.registerTempTable("patients"}

// Select data from table

val results = sqlContext.sql(.....SELECT* FROM patients \\'.....)

// display dataframe in a tabular format

results.show()

//Find all the patients whose lastVisitDate between current time and \\'2012-09-15\\'

val results = sqlContext.sql(.....SELECT * FROM patients WHERE

TO_DATE(CAST(UNIX_TIMESTAMP(lastVisitDate, \\'yyyy-MM-dd\\') AS TIMESTAMP))

BETWEEN \\'2012-09-15\\' AND current_timestamp() ORDER BY lastVisitDate.....)

results.showQ

/.Find all the patients who born in 2011

val results = sqlContext.sql(.....SELECT * FROM patients WHERE

YEAR(TO_DATE(CAST(UNIXJTIMESTAMP(dateOfBirth, \\'yyyy-MM-dd\\') AS

TIMESTAMP))) = 2011 .....)

results. show()

//Find all the patients age

val results = sqlContext.sql(.....SELECT name, dateOfBirth, datediff(current_date(),

TO_DATE(CAST(UNIX_TIMESTAMP(dateOfBirth, \\'yyyy-MM-dd\\') AS TIMESTAMP}}}/365

AS age

FROM patients

Mini >

results.show() //List patients whose last visited more than 60 days ago -- List patients whose last visited more than 60 days ago val results = sqlContext.sql(.....SELECT name, lastVisitDate FROM patients WHERE

datediff(current_date(), TO_DATE(CAST(UNIX_TIMESTAMP[lastVisitDate, \\'yyyy-MM-dd\\')

AS T1MESTAMP))) > 60.....);

results. showQ;

-- Select patients 18 years old or younger

SELECT\\' FROM patients WHERE TO_DATE(CAST(UNIXJTIMESTAMP(dateOfBirth,

\\'yyyy-MM-dd\\') AS TIMESTAMP}) > DATE_SUB(current_date(),INTERVAL 18 YEAR);

val results = sqlContext.sql(.....SELECT\\' FROM patients WHERE

TO_DATE(CAST(UNIX_TIMESTAMP(dateOfBirth, \\'yyyy-MM--dd\\') AS TIMESTAMP)) >

DATE_SUB(current_date(), T8*365).....);

results. showQ;

val results = sqlContext.sql(.....SELECT DATE_SUB(current_date(), 18*365) FROM

patients.....);

results.show();

QUESTION 4

Problem Scenario 52 : You have been given below code snippet.

val b = sc.parallelize(List(1,2,3,4,5,6,7,8,2,4,2,1,1,1,1,1))

Operation_xyz

Write a correct code snippet for Operation_xyz which will produce below output.

scalaxollection.Map[lnt,Long] = Map(5 -> 1, 8 -> 1, 3 -> 1, 6 -> 1, 1 -> S, 2 -> 3, 4 -> 2, 7 ->

1)

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution : b.countByValue countByValue Returns a map that contains all unique values of the RDD and their respective occurrence counts. (Warning: This operation will finally aggregate the information in a single reducer.) Listing Variants def countByValue(): Map[T, Long]

QUESTION 5

Problem Scenario 45 : You have been given 2 files , with the content as given Below (spark12/technology.txt) (spark12/salary.txt) (spark12/technology.txt) first,last,technology Amit,Jain,java Lokesh,kumar,unix Mithun,kale,spark Rajni,vekat,hadoop Rahul,Yadav,scala (spark12/salary.txt) first,last,salary Amit,Jain,100000 Lokesh,kumar,95000 Mithun,kale,150000 Rajni,vekat,154000 Rahul,Yadav,120000 Write a Spark program, which will join the data based on first and last name and save the joined results in following format, first Last.technology.salary

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create 2 files first using Hue in hdfs.

Step 2 : Load all file as an RDD

val technology = sc.textFile(Msparkl2/technology.txt").map(e => e.splitf\\',"))

val salary = sc.textFile("spark12/salary.txt").map(e => e.split("."))

Step 3 : Now create Key.value pair of data and join them.

val joined = technology.map(e=>((e(0),e(1)),e(2))).join(salary.map(e=>((e(0),e(1)),e(2))))

Step 4 : Save the results in a text file as below.

joined.repartition(1).saveAsTextFile("spark12/multiColumn Joined.txt")

QUESTION 6

Problem Scenario 86 : In Continuation of previous question, please accomplish following activities.

1.

Select Maximum, minimum, average , Standard Deviation, and total quantity.

2.

Select minimum and maximum price for each product code.

3.

Select Maximum, minimum, average , Standard Deviation, and total quantity for each product code, hwoever make sure Average and Standard deviation will have maximum two decimal values.

4.

Select all the product code and average price only where product count is more than or equal to 3.

5.

Select maximum, minimum , average and total of all the products for each code. Also produce the same across all the products.

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Select Maximum, minimum, average , Standard Deviation, and total quantity.

val results = sqlContext.sql(\\'.....SELECT MAX(price) AS MAX , MIN(price) AS MIN ,

AVG(price) AS Average, STD(price) AS STD, SUM(quantity) AS total_products FROM

products......)

results. showQ

Step 2 : Select minimum and maximum price for each product code.

val results = sqlContext.sql(.....SELECT code, MAX(price) AS Highest Price\\', MIN(price)

AS Lowest Price\\'

FROM products GROUP BY code.....)

results. showQ

Step 3 : Select Maximum, minimum, average , Standard Deviation, and total quantity for

each product code, hwoever make sure Average and Standard deviation will have

maximum two decimal values.

val results = sqlContext.sql(.....SELECT code, MAX(price), MIN(price),

CAST(AVG(price} AS DECIMAL(7,2)) AS Average\\', CAST(STD(price) AS DECIMAL(7,2))

AS \\'Std Dev\ SUM(quantity) FROM products

GROUP BY code.....)

results. showQ

Step 4 : Select all the product code and average price only where product count is more

than or equal to 3.

val results = sqlContext.sql(.....SELECT code AS Product Code\\',

COUNTf) AS Count\\',

CAST(AVG(price) AS DECIMAL(7,2)) AS Average\\' FROM products GROUP BY code

HAVING Count >=3"M") results. showQ

Step 5 : Select maximum, minimum , average and total of all the products for each code.

Also produce the same across all the products.

val results = sqlContext.sql("""SELECT

code,

MAX(price),

MIN(pnce),

CAST(AVG(price) AS DECIMAL(7,2)) AS Average\\',

SUM(quantity)FROM products

GROUP BY code

WITH ROLLUP""")

results. show()

QUESTION 7

Problem Scenario 70 : Write down a Spark Application using Python, In which it read a

file "Content.txt" (On hdfs) with following content. Do the word count and save the

results in a directory called "problem85" (On hdfs)

Content.txt

Hello this is ABCTECH.com

This is XYZTECH.com

Apache Spark Training

This is Spark Learning Session Spark is faster than MapReduce

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create an application with following code and store it in problem84.py

Import SparkContext and SparkConf

from pyspark import SparkContext, SparkConf

Create configuration object and set App name

conf = SparkConf().setAppName("CCA 175 Problem 85") sc = sparkContext(conf=conf)

#load data from hdfs

contentRDD = sc.textFile(MContent.txt")

#filter out non-empty lines

nonemptyjines = contentRDD.filter(lambda x: len(x) > 0)

#Split line based on space

words = nonempty_lines.ffatMap(lambda x: x.split(\\'\\'}}

#Do the word count

wordcounts = words.map(lambda x: (x, 1)) \

reduceByKey(lambda x, y: x+y) \

map(lambda x: (x[1], x[0]}}.sortByKey(False}

for word in wordcounts.collect(): print(word)

#Save final data " wordcounts.saveAsTextFile("problem85")

step 2 : Submit this application

spark-submit -master yarn problem85.py

QUESTION 8

Problem Scenario 25 : You have been given below comma separated employee information. That needs to be added in /home/cloudera/flumetest/in.txt file (to do tail source) sex,name,city 1,alok,mumbai 1,jatin,chennai 1,yogesh,kolkata 2,ragini,delhi 2,jyotsana,pune 1,valmiki,banglore Create a flume conf file using fastest non-durable channel, which write data in hive warehouse directory, in two separate tables called flumemaleemployee1 and flumefemaleemployee1 (Create hive table as well for given data}. Please use tail source with /home/cloudera/flumetest/in.txt file. Flumemaleemployee1 : will contain only male employees data flumefemaleemployee1 : Will contain only woman employees data

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create hive table for flumemaleemployeel and .\\'

CREATE TABLE flumemaleemployeel

(

sex_type int, name string, city string) ROW FORMAT DELIMITED FIELDS TERMINATED BY \\',\\'; CREATE TABLE flumefemaleemployeel (sex_type int, name string, city string) ROW FORMAT DELIMITED FIELDS TERMINATED BY \\',\\'; Step 2 : Create below directory and file mkdir /home/cloudera/flumetest/ cd /home/cloudera/flumetest/ Step 3 : Create flume configuration file, with below configuration for source, sink and channel and save it in flume5.conf. agent.sources = tailsrc agent.channels = mem1 mem2 agent.sinks = stdl std2 agent.sources.tailsrc.type = exec agent.sources.tailsrc.command = tail -F /home/cloudera/flumetest/in.txt agent.sources.tailsrc.batchSize = 1 agent.sources.tailsrc.interceptors = i1 agent.sources.tailsrc.interceptors.i1.type = regex_extractor agent.sources.tailsrc.interceptors.il.regex = A(\\d} agent.sources.tailsrc. interceptors. M.serializers = t1 agent.sources.tailsrc. interceptors, i1.serializers.t1. name = type agent.sources.tailsrc.selector.type = multiplexing agent.sources.tailsrc.selector.header = type agent.sources.tailsrc.selector.mapping.1 = memi agent.sources.tailsrc.selector.mapping.2 = mem2 agent.sinks.std1.type = hdfs agent.sinks.stdl.channel = mem1 agent.sinks.stdl.batchSize = 1 agent.sinks.std1.hdfs.path = /user/hive/warehouse/flumemaleemployeei agent.sinks.stdl.rolllnterval = 0 agent.sinks.stdl.hdfs.tileType = Data Stream agent.sinks.std2.type = hdfs agent.sinks.std2.channel = mem2 agent.sinks.std2.batchSize = 1 agent.sinks.std2.hdfs.path = /user/hi ve/warehouse/flumefemaleemployee1 agent.sinks.std2.rolllnterval = 0 agent.sinks.std2.hdfs.tileType = Data Stream agent.channels.mem1.type = memory agent.channels.meml.capacity = 100 agent.channels.mem2.type = memory agent.channels.mem2.capacity = 100 agent.sources.tailsrc.channels = mem1 mem2 Step 4 : Run below command which will use this configuration file and append data in hdfs. Start flume service: flume-ng agent -conf /home/cloudera/flumeconf -conf-file /home/cloudera/flumeconf/flume5.conf --name agent Step 5 : Open another terminal create a file at /home/cloudera/flumetest/in.txt. Step 6 : Enter below data in file and save it. l.alok.mumbai 1 jatin.chennai 1,yogesh,kolkata 2,ragini,delhi 2,jyotsana,pune 1,valmiki,banglore Step 7 : Open hue and check the data is available in hive table or not. Step 8 : Stop flume service by pressing ctrl+c

QUESTION 9

Problem Scenario 14 : You have been given following mysql database details as well as other info. user=retail_dba password=cloudera database=retail_db jdbc URL = jdbc:mysql://quickstart:3306/retail_db Please accomplish following activities.

1.

Create a csv file named updated_departments.csv with the following contents in local file system. updated_departments.csv 2,fitness 3,footwear 12,fathematics 13,fcience 14,engineering 1000,management

2.

Upload this csv file to hdfs filesystem,

3.

Now export this data from hdfs to mysql retaildb.departments table. During upload make sure existing department will just updated and new departments needs to be inserted.

4.

Now update updated_departments.csv file with below content. 2,Fitness 3,Footwear 12,Fathematics 13,Science 14,Engineering 1000,Management 2000,Quality Check

5.

Now upload this file to hdfs.

6.

Now export this data from hdfs to mysql retail_db.departments table. During upload make sure existing department will just updated and no new departments needs to be inserted.

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create a csv tile named updateddepartments.csv with give content.

Step 2 : Now upload this tile to HDFS.

Create a directory called newdata.

hdfs dfs -mkdir new_data

hdfs dfs -put updated_departments.csv newdata/

Step 3 : Check whether tile is uploaded or not. hdfs dfs -Is new_data

Step 4 : Export this file to departments table using sqoop.

sqoop export --connect jdbc:mysql://quickstart:3306/retail_db \

-username retail_dba \

--password cloudera \

-table departments \

--export-dir new_data \

-batch \

-m 1 \

-update-key department_id \

-update-mode allowinsert

Step 5 : Check whether required data upsert is done or not. mysql --user=retail_dba password=cloudera

show databases;

use retail_db;

show tables;

select" from departments;

Step 6 : Update updated_departments.csv file.

Step 7 : Override the existing file in hdfs.

hdfs dfs -put updated_departments.csv newdata/

Step 8 : Now do the Sqoop export as per the requirement.

sqoop export --connect jdbc:mysql://quickstart:3306/retail_db \

-username retail_dba\

--password cloudera \

--table departments \

--export-dir new_data \

--batch \

-m 1 \

--update-key-department_id \

-update-mode updateonly

Step 9 : Check whether required data update is done or not. mysql --user=retail_dba password=cloudera

show databases;

use retail db;

show tables;

select" from departments;

QUESTION 10

Problem Scenario 56 : You have been given below code snippet.

val a = sc.parallelize(l to 100. 3)

operation1

Write a correct code snippet for operationl which will produce desired output, shown below.

Array [Array [I nt]] = Array(Array(1, 2, 3,4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19, 20,

21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33),

Array(34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,

56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66),

Array(67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88,

89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100))

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution : a.glom.collect glom Assembles an array that contains all elements of the partition and embeds it in an RDD.

Each returned array contains the contents of one panition

QUESTION 11

Problem Scenario 71 :

Write down a Spark script using Python,

In which it read a file "Content.txt" (On hdfs) with following content.

After that split each row as (key, value), where key is first word in line and entire line as

value.

Filter out the empty lines.

And save this key value in "problem86" as Sequence file(On hdfs)

Part 2 : Save as sequence file , where key as null and entire line as value. Read back the

stored sequence files.

Content.txt

Hello this is ABCTECH.com

This is XYZTECH.com

Apache Spark Training

This is Spark Learning Session Spark is faster than MapReduce

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 :

Import SparkContext and SparkConf

from pyspark import SparkContext, SparkConf

Step 2:

#load data from hdfs

contentRDD = sc.textFile(MContent.txt")

Step 3:

#filter out non-empty lines

nonemptyjines = contentRDD.filter(lambda x: len(x) > 0)

Step 4:

#Split line based on space (Remember : It is mandatory to convert is in tuple} words =

nonempty_lines.map(lambda x: tuple(x.split(\\'\\', 1)))

words.saveAsSequenceFile("problem86")

Step 5: Check contents in directory problem86 hdfs dfs -cat problem86/part*

Step 6 : Create key, value pair (where key is null)

nonempty_lines.map(lambda line: (None, Mne}).saveAsSequenceFile("problem86_1")

Step 7 : Reading back the sequence file data using spark. seqRDD =

sc.sequenceFile("problem86_1")

Step 8 : Print the content to validate the same.

for line in seqRDD.collect():

print(line)

QUESTION 12

Problem Scenario 17 : You have been given following mysql database details as well as other info. user=retail_dba password=cloudera database=retail_db jdbc URL = jdbc:mysql://quickstart:3306/retail_db Please accomplish below assignment.

1.

Create a table in hive as below, create table departments_hiveOl(department_id int,

department_name string, avg_salary int);

2.

Create another table in mysql using below statement CREATE TABLE IF NOT EXISTS

departments_hive01(id int, department_name varchar(45), avg_salary int);

3.

Copy all the data from departments table to departments_hive01 using insert into

departments_hive01 select a.*, null from departments a;

Also insert following records as below

insert into departments_hive01 values(777, "Not known",1000);

insert into departments_hive01 values(8888, null,1000);

insert into departments_hive01 values(666, null,1100);

4.

Now import data from mysql table departments_hive01 to this hive table. Please make

sure that data should be visible using below hive command. Also, while importing if null

value found for department_name column replace it with "" (empty string) and for id column

with -999 select * from departments_hive;

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution :

Step 1 : Create hive table as below.

hive

show tables;

create table departments_hive01(department_id int, department_name string, avgsalary

int);

Step 2 : Create table in mysql db as well.

mysql -user=retail_dba -password=cloudera

use retail_db

CREATE TABLE IF NOT EXISTS departments_hive01(id int, department_name

varchar(45), avg_salary int);

show tables;

step 3 : Insert data in mysql table.

insert into departments_hive01 select a.*, null from departments a;

check data inserts

select\\' from departments_hive01;

Now iserts null records as given in problem. insert into departments_hive01 values(777,

"Not known",1000); insert into departments_hive01 values(8888, null,1000); insert into

departments_hive01 values(666, null,1100);

Step 4 : Now import data in hive as per requirement.

sqoop import \

-connect jdbc:mysql://quickstart:3306/retail_db \

~username=retail_dba \

--password=cloudera \

-table departments_hive01 \

--hive-home /user/hive/warehouse \

--hive-import \

-hive-overwrite \

-hive-table departments_hive0l \

--fields-terminated-by \\'\001\\' \

--null-string M"\

--null-non-strlng -999 \

-split-by id \

-m 1

Step 5 : Checkthe data in directory.

hdfs dfs -Is /user/hive/warehouse/departments_hive01

hdfs dfs -cat/user/hive/warehouse/departments_hive01/part"

Check data in hive table.

Select * from departments_hive01;

QUESTION 13

Problem Scenario 5 : You have been given following mysql database details.

user=retail_dba password=cloudera database=retail_db jdbc URL = jdbc:mysql://quickstart:3306/retail_db Please accomplish following activities.

1.

List all the tables using sqoop command from retail_db

2.

Write simple sqoop eval command to check whether you have permission to read database tables or not.

3.

Import all the tables as avro files in /user/hive/warehouse/retail cca174.db

4.

Import departments table as a text file in /user/cloudera/departments.

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution:

Step 1 : List tables using sqoop

sqoop list-tables --connect jdbc:mysql://quickstart:330G/retail_db --username retail dba password cloudera

Step 2 : Eval command, just run a count query on one of the table.

sqoop eval \

--connect jdbc:mysql://quickstart:3306/retail_db \

-username retail_dba \

-password cloudera \

--query "select count(1) from ordeMtems"

Step 3 : Import all the tables as avro file.

sqoop import-all-tables \

-connect jdbc:mysql://quickstart:3306/retail_db \

-username=retail_dba \

-password=cloudera \

-as-avrodatafile \

-warehouse-dir=/user/hive/warehouse/retail stage.db \

-ml

Step 4 : Import departments table as a text file in /user/cloudera/departments

sqoop import \

-connect jdbc:mysql://quickstart:3306/retail_db \

-username=retail_dba \

-password=cloudera \

-table departments \

-as-textfile \

-target-dir=/user/cloudera/departments

Step 5 : Verify the imported data.

hdfs dfs -ls /user/cloudera/departments

hdfs dfs -ls /user/hive/warehouse/retailstage.db

hdfs dfs -ls /user/hive/warehouse/retail_stage.db/products

QUESTION 14

Problem Scenario 26 : You need to implement near real time solutions for collecting information when submitted in file with below information. You have been given below directory location (if not available than create it) /tmp/nrtcontent. Assume your departments upstream service is continuously committing data in this directory as a new file (not stream of data, because it is near real time solution). As soon as file committed in this directory that needs to be available in hdfs in /tmp/flume location Data

echo "I am preparing for CCA175 from ABCTECH.com" > /tmp/nrtcontent/.he1.txt mv /tmp/nrtcontent/.he1.txt /tmp/nrtcontent/he1.txt After few mins echo "I am preparing for CCA175 from TopTech.com" > /tmp/nrtcontent/.qt1.txt mv /tmp/nrtcontent/.qt1.txt /tmp/nrtcontent/qt1.txt

Write a flume configuration file named flumes.conf and use it to load data in hdfs with following additional properties.

1.

Spool /tmp/nrtcontent

2.

File prefix in hdfs sholuld be events

3.

File suffix should be Jog

4.

If file is not commited and in use than it should have as prefix.

5.

Data should be written as text to hdfs

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution : Step 1 : Create directory mkdir /tmp/nrtcontent Step 2 : Create flume configuration file, with below configuration for source, sink and channel and save it in flume6.conf. agent1 .sources = source1 agent1 .sinks = sink1 agent1.channels = channel1 agent1 .sources.source1.channels = channel1 agent1 .sinks.sink1.channel = channel1 agent1 .sources.source1.type = spooldir agent1 .sources.source1.spoolDir = /tmp/nrtcontent agent1 .sinks.sink1 .type = hdfs agent1 .sinks.sink1.hdfs.path = /tmp/flume agent1.sinks.sink1.hdfs.filePrefix = events agent1.sinks.sink1.hdfs.fileSuffix = .log agent1 .sinks.sink1.hdfs.inUsePrefix = _ agent1 .sinks.sink1.hdfs.fileType = Data Stream Step 4 : Run below command which will use this configuration file and append data in hdfs. Start flume service: flume-ng agent -conf /home/cloudera/flumeconf -conf-file /home/cloudera/flumeconf/flume6.conf --name agent1 Step 5 : Open another terminal and create a file in /tmp/nrtcontent echo "I am preparing for CCA175 from ABCTechm.com" > /tmp/nrtcontent/.he1.txt mv /tmp/nrtcontent/.he1.txt /tmp/nrtcontent/he1.txt After few mins echo "I am preparing for CCA175 from TopTech.com" > /tmp/nrtcontent/.qt1.txt mv /tmp/nrtcontent/.qt1.txt /tmp/nrtcontent/qt1.txt

QUESTION 15

Problem Scenario 84 : In Continuation of previous question, please accomplish following activities.

1.

Select all the products which has product code as null

2.

Select all the products, whose name starts with Pen and results should be order by Price descending order.

3.

Select all the products, whose name starts with Pen and results should be order by Price descending order and quantity ascending order.

4.

Select top 2 products by price

Correct Answer: See the explanation for Step by Step Solution and configuration.

Solution : Step 1 : Select all the products which has product code as null val results = sqlContext.sql(.....SELECT\\' FROM products WHERE code IS NULL.....) results. showQ val results = sqlContext.sql(.....SELECT * FROM products WHERE code = NULL ",,M) results.showQ Step 2 : Select all the products , whose name starts with Pen and results should be order by Price descending order. val results = sqlContext.sql(.....SELECT * FROM products WHERE name LIKE \\'Pen %\\' ORDER BY price DESC.....) results. showQ Step 3 : Select all the products , whose name starts with Pen and results should be order by Price descending order and quantity ascending order. val results = sqlContext.sql(\\'.....SELECT * FROM products WHERE name LIKE \\'Pen %\\' ORDER BY price DESC, quantity.....) results. showQ Step 4 : Select top 2 products by price val results = sqlContext.sql(.....SELECT\\' FROM products ORDER BY price desc LIMIT2.....} results. show()