CCA Spark and Hadoop Developer Exam v7.0 (CCA175)

Question 1

Problem Scenario 24 : You have been given below comma separated employee information.
Data Set:
name,salary,sex,age
alok,100000,male,29
jatin,105000,male,32
yogesh,134000,male,39
ragini,112000,female,35
jyotsana,129000,female,39
valmiki,123000,male,29
Requirements:
Use the netcat service on port 44444, and nc above data line by line. Please do the following activities.
1. Create a flume conf file using fastest channel, which write data in hive warehouse directory, in a table called flumemaleemployee (Create hive table as well tor given data).
2. While importing, make sure only male employee data is stored.

Expose Correct Answer

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

Explanation: Step 1 : Create hive table for flumeemployee.' CREATE TABLE flumemaleemployee (name string, salary int, sex string, age int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','; step 2 : Create flume configuration file, with below configuration for source, sink and channel and save it in flume4.conf. #Define source , sink, channel and agent. agent1 .sources = source1 agent1 .sinks = sink1 agent1 .channels = channel1 # Describe/configure source1 agent1 .sources.source1.type = netcat agent1 .sources.source1.bind = 127.0.0.1 agent1.sources.sourcel.port = 44444 #Define interceptors agent1.sources.source1.interceptors=il agent1 .sources.source1.interceptors.i1.type=regex_filter agent1 .sources.source1.interceptors.i1.regex=female agent1 .sources.source1.interceptors.i1.excludeEvents=true ## Describe sink1 agent1 .sinks, sinkl.channel = memory-channel agent1.sinks.sink1.type = hdfs agent1 .sinks, sinkl. hdfs. path =

/user/hive/warehouse/flumemaleemployee hdfs-agent.sinks.hdfs-write.hdfs.writeFormat=Text agentl .sinks.sink1.hdfs.fileType = Data Stream # Now we need to define channel1 property. agent1.channels.channel1.type = memory agent1.channels.channell.capacity = 1000 agent1.channels.channel1.transactionCapacity = 100 # Bind the source and sink to the channel agent1 .sources.source1.channels = channel1 agent1 .sinks.sink1.channel = channel1 step 3 : 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/flume4.conf --name agentl Step 4 : Open another terminal and use the netcat service, nc localhost 44444 Step 5 : Enter data line by line. alok,100000,male,29 jatin,105000,male,32 yogesh,134000,male,39 ragini,112000,female,35 jyotsana,129000,female,39 valmiki.123000.male.29 Step 6 : Open hue and check the data is available in hive table or not. Step 7 : Stop flume service by pressing ctrl+c Step 8 : Calculate average salary on hive table using below query. You can use either hive command line tool or hue. select avg(salary) from flumeemployee;

Next Question

Question discussion

You need to signup or login to add a comment

Question 2

Problem Scenario 27 : You need to implement near real time solutions for collecting information when submitted in file with below information.
Data
echo "IBM,100,20160104" >> /tmp/spooldir/bb/.bb.txt
echo "IBM,103,20160105" >> /tmp/spooldir/bb/.bb.txt
mv /tmp/spooldir/bb/.bb.txt /tmp/spooldir/bb/bb.txt
After few mins
echo "IBM,100.2,20160104" >> /tmp/spooldir/dr/.dr.txt
echo "IBM,103.1,20160105" >> /tmp/spooldir/dr/.dr.txt
mv /tmp/spooldir/dr/.dr.txt /tmp/spooldir/dr/dr.txt
Requirements:
You have been given below directory location (if not available than create it) /tmp/spooldir .
You have a finacial subscription for getting stock prices from BloomBerg as well as Reuters and using ftp you download every hour new files from their respective ftp site in directories /tmp/spooldir/bb and /tmp/spooldir/dr respectively.
As soon as file committed in this directory that needs to be available in hdfs in /tmp/flume/finance location in a single directory.
Write a flume configuration file named flume7.conf and use it to load data in hdfs with following additional properties .
1. Spool /tmp/spooldir/bb and /tmp/spooldir/dr
2. File prefix in hdfs sholuld be events

3. File suffix should be .log
4. If file is not commited and in use than it should have _ as prefix.
5. Data should be written as text to hdfs

Expose Correct Answer

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

Explanation: Solution : Step 1 : Create directory mkdir /tmp/spooldir/bb mkdir /tmp/spooldir/dr Step 2 : Create flume configuration file, with below configuration for agent1.sources = source1 source2 agent1 .sinks = sink1 agent1.channels = channel1 agent1 .sources.source1.channels = channel1 agentl .sources.source2.channels = channell agent1 .sinks.sinkl.channel = channell agent1 .sources.source1.type = spooldir agent1 .sources.sourcel.spoolDir = /tmp/spooldir/bb agent1 .sources.source2.type = spooldir agent1 .sources.source2.spoolDir = /tmp/spooldir/dr agent1 .sinks.sink1.type = hdfs agent1 .sinks.sink1.hdfs.path = /tmp/flume/finance agent1-sinks.sink1.hdfs.filePrefix = events agent1.sinks.sink1.hdfs.fileSuffix = .log agent1 .sinks.sink1.hdfs.inUsePrefix = _ agent1 .sinks.sink1.hdfs.fileType = Data Stream agent1.channels.channel1.type = file 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/fIumeconf/fIume7.conf --name agent1 Step 5 : Open another terminal and create a file in /tmp/spooldir/ echo "IBM,100,20160104" /tmp/spooldir/bb/.bb.txt echo "IBM,103,20160105" /tmp/spooldir/bb/.bb.txt mv /tmp/spooldir/bb/.bb.txt /tmp/spooldir/bb/bb.txt After few mins echo "IBM,100.2,20160104" /tmp/spooldir/dr/.dr.txt echo "IBM,103.1,20160105" /tmp/spooldir/dr/.dr.txt mv /tmp/spooldir/dr/.dr.txt /tmp/spooldir/dr/dr.txt

Next Question

Question discussion

You need to signup or login to add a comment

Question 3

Problem Scenario 57 : You have been given below code snippet.
val a = sc.parallelize(1 to 9, 3) operationl
Write a correct code snippet for operationl which will produce desired output, shown below.
Array[(String, Seq[lnt])] = Array((even,ArrayBuffer(2, 4, G, 8)), (odd,ArrayBuffer(1, 3, 5, 7, 9)))

Expose Correct Answer

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

Explanation: Solution : a.groupBy(x => {if (x % 2 == 0) "even" else "odd" }).collect

Next Question

Question discussion

You need to signup or login to add a comment

Question 4

Problem Scenario 34 : You have given a file named spark6/user.csv.
Data is given below:
user.csv
id,topic,hits
Rahul,scala,120
Nikita,spark,80
Mithun,spark,1
myself,cca175,180
Now write a Spark code in scala which will remove the header part and create RDD of values as below, for all rows. And also if id is myself" than filter out row.
Map(id -> om, topic -> scala, hits -> 120)

Expose Correct Answer

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

Explanation: 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 user.csv file from hdfs and create PairRDDs val csv = sc.textFile("spark6/user.csv") Step 3 : split and clean data val headerAndRows = csv.map(line => line.split(",").map(_.trim)) Step 4 : Get header row val header = headerAndRows.first Step 5 : Filter out header (We need to check if the first val matches the first header name) val data = headerAndRows.filter(_(0) != header(O)) Step 6 : Splits to map (header/value pairs) val maps = data.map(splits => header.zip(splits).toMap) step 7: Filter out the user "myself val result = maps.filter(map => mapf'id") != "myself") Step 8 : Save the output as a Text file. result.saveAsTextFile("spark6/result.txt")

Next Question

Question discussion

You need to signup or login to add a comment

Question 5

Problem Scenario 91 : You have been given data in json format as below.
{"first_name":"Ankit", "last_name":"Jain"}
{"first_name":"Amir", "last_name":"Khan"}
{"first_name":"Rajesh", "last_name":"Khanna"}
{"first_name":"Priynka", "last_name":"Chopra"}
{"first_name":"Kareena", "last_name":"Kapoor"}
{"first_name":"Lokesh", "last_name":"Yadav"}
Do the following activity
1. create employee.json tile locally.
2. Load this tile on hdfs
3. Register this data as a temp table in Spark using Python.
4. Write select query and print this data.
5. Now save back this selected data in json format.

Expose Correct Answer

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

Explanation: Solution : Step 1 : create employee.json tile locally. vi employee.json (press insert) past the content. Step 2 : Upload this tile to hdfs, default location hadoop fs -put employee.json val employee = sqlContext.read.json("/user/cloudera/employee.json") employee.write.parquet("employee. parquet") val parq_data = sqlContext.read.parquet("employee.parquet") parq_data.registerTempTable("employee") val allemployee = sqlContext.sql("SELeCT' FROM employee") all_employee.show() import org.apache.spark.sql.SaveMode prdDF.write..format("orc").saveAsTable("product ore table"} //Change the codec. sqlContext.setConf("spark.sql.parquet.compression.codec","snappy") employee.write.mode(SaveMode.Overwrite).parquet("employee.parquet")

Next Question

Question discussion

You need to signup or login to add a comment

Question 6

Problem Scenario 2 :
There is a parent organization called "ABC Group Inc", which has two child companies named Tech Inc and MPTech.
Both companies employee information is given in two separate text file as below. Please do the following activity for employee details.
Tech Inc.txt
1,Alok,Hyderabad
2,Krish,Hongkong
3,Jyoti,Mumbai
4,Atul,Banglore
5,Ishan,Gurgaon
MPTech.txt
6,John,Newyork
7,alp2004,California
8,tellme,Mumbai
9,Gagan21,Pune
10,Mukesh,Chennai
1. Which command will you use to check all the available command line options on HDFS and How will you get the Help for individual command.
2. Create a new Empty Directory named Employee using Command line. And also create an empty file named in it Techinc.txt
3. Load both companies Employee data in Employee directory (How to override existing file in HDFS).
4. Merge both the Employees data in a Single tile called MergedEmployee.txt, merged tiles should have new line character at the end of each file content.
5. Upload merged file on HDFS and change the file permission on HDFS merged file, so that owner and group member can read and write, other user can read the file.
6. Write a command to export the individual file as well as entire directory from HDFS to local file System.

Expose Correct Answer

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

Explanation: Solution : Step 1 : Check All Available command hdfs dfs Step 2 : Get help on Individual command hdfs dfs -help get Step 3 : Create a directory in HDFS using named Employee and create a Dummy file in it called e.g. Techinc.txt hdfs dfs -mkdir Employee Now create an emplty file in Employee directory using Hue. Step 4 : Create a directory on Local file System and then Create two files, with the given data in problems. Step 5 : Now we have an existing directory with content in it, now using HDFS command line , overrid this existing Employee directory. While copying these files

from local file System to HDFS. cd /home/cloudera/Desktop/ hdfs dfs -put -f Employee Step 6 : Check All files in directory copied successfully hdfs dfs -Is Employee Step 7 : Now merge all the files in Employee directory, hdfs dfs -getmerge -nl Employee MergedEmployee.txt Step 8 : Check the content of the file. cat MergedEmployee.txt Step 9 : Copy merged file in Employeed directory from local file ssytem to HDFS. hdfs dfs - put MergedEmployee.txt Employee/ Step 10 : Check file copied or not. hdfs dfs -Is Employee Step 11 : Change the permission of the merged file on HDFS hdfs dfs -chmpd 664 Employee/MergedEmployee.txt Step 12 : Get the file from HDFS to local file system, hdfs dfs -get Employee Employee_hdfs

Next Question

Question discussion

You need to signup or login to add a comment

Question 7

Problem Scenario 9 : 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.
1. Import departments table in a directory.
2. Again import departments table same directory (However, directory already exist hence it should not overrride and append the results)
3. Also make sure your results fields are terminated by '|' and lines terminated by '\n\

Expose Correct Answer

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

Explanation: Solutions : Step 1 : Clean the hdfs file system, if they exists clean out. hadoop fs -rm -R departments hadoop fs -rm -R categories hadoop fs -rm -R products hadoop fs -rm -R orders hadoop fs -rm -R order_items hadoop fs -rm -R customers Step 2 : Now import the department table as per requirement. sqoop import \ -connect jdbc:mysql://quickstart:330G/retaiI_db \ --username=retail_dba \ -password=cloudera \ -table departments \ -target-dir=departments \ -fields-terminated-by '|' \ -lines-terminated-by '\n' \ -ml Step 3 : Check imported data. hdfs dfs -Is departments hdfs dfs -cat departments/part-m-00000 Step 4 : Now again import data and needs to appended. sqoop import \ -connect jdbc:mysql://quickstart:3306/retail_db \ --username=retail_dba

\ -password=cloudera \ -table departments \ -target-dir departments \ -append \ -tields-terminated-by '|' \ -lines-termtnated-by '\n' \ -ml Step 5 : Again Check the results hdfs dfs -Is departments hdfs dfs -cat departments/part-m-00001

Next Question

Question discussion

You need to signup or login to add a comment

Question 8

Problem Scenario 68 : You have given a file as below.
spark75/f ile1.txt
File contain some text. As given Below
spark75/file1.txt
Apache Hadoop is an open-source software framework written in Java for distributed storage and distributed processing of very large data sets on computer clusters built from commodity hardware. All the modules in Hadoop are designed with a fundamental assumption that hardware failures are common and should be automatically handled by the framework
The core of Apache Hadoop consists of a storage part known as Hadoop Distributed File System (HDFS) and a processing part called MapReduce. Hadoop splits files into large blocks and distributes them across nodes in a cluster. To process data, Hadoop transfers packaged code for nodes to process in parallel based on the data that needs to be processed.
his approach takes advantage of data locality nodes manipulating the data they have access to to allow the dataset to be processed faster and more efficiently than it would be in a more conventional supercomputer architecture that relies on a parallel file system where computation and data are distributed via high-speed networking
For a slightly more complicated task, lets look into splitting up sentences from our documents into word bigrams. A bigram is pair of successive tokens in some sequence. We will look at building bigrams from the sequences of words in each sentence, and then try to find the most frequently occuring ones.
The first problem is that values in each partition of our initial RDD describe lines from the file rather than sentences. Sentences may be split over multiple lines. The glom() RDD method is used to create a single entry for each document containing the list of all lines, we can then join the lines up, then resplit them into sentences using "." as the separator, using flatMap so that every object in our RDD is now a sentence.
A bigram is pair of successive tokens in some sequence. Please build bigrams from the sequences of

Expose Correct Answer

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

Explanation: Solution : Step 1 : Create all three tiles in hdfs (We will do using Hue}. However, you can first create in local filesystem and then upload it to hdfs. Step 2 : The first problem is that values in each partition of our initial RDD describe lines from the file rather than sentences. Sentences may be split over multiple lines. The glom() RDD method is used to create a single entry for each document containing the list of all lines, we can then join the lines up, then resplit them into sentences using "." as the separator, using flatMap so that every object in our RDD is now a sentence. sentences = sc.textFile("spark75/file1.txt") \ .glom() \ map(lambda x: " ".join(x)) \ .flatMap(lambda x: x.spllt(".")) Step 3 : Now we have isolated each sentence we can split it into a list of words and extract the word bigrams from it. Our new RDD contains tuples containing the word bigram (itself a tuple containing the first and second word) as the first value and the number 1 as the second value. bigrams = sentences.map(lambda x:x.split()) \ .flatMap(lambda x: [((x[i],x[i+1]),1)for i in range(0,len(x)-1)]) Step 4 : Finally we can apply the same reduceByKey and sort steps that we used in the wordcount example, to count up the bigrams and sort them in order of descending frequency. In reduceByKey the key is not an individual word but a bigram. freq_bigrams = bigrams.reduceByKey(lambda x,y:x+y)\ map(lambda x:(x[1],x[0])) \ sortByKey(False) freq_bigrams.take(10)

Next Question

Question discussion

You need to signup or login to add a comment

Question 9

Problem Scenario 64 : You have been given below code snippet.
val a = sc.parallelize(List("dog", "salmon", "salmon", "rat", "elephant"), 3)
val b = a.keyBy(_.length)
val c = sc.parallelize(Ust("dog","cat","gnu","salmon","rabbit","turkey","wolf","bear","bee"), 3)
val d = c.keyBy(_.length)
operation1
Write a correct code snippet for operationl which will produce desired output, shown below.
Array[(lnt, (Option[String], String))] = Array((6,(Some(salmon),salmon)),
(6,(Some(salmon),rabbit}}, (6,(Some(salmon),turkey)), (6,(Some(salmon),salmon)),
(6,(Some(salmon),rabbit)), (6,(Some(salmon),turkey)), (3,(Some(dog),dog)),
(3,(Some(dog),cat)), (3,(Some(dog),gnu)), (3,(Some(dog),bee)), (3,(Some(rat),
(3,(Some(rat),cat)), (3,(Some(rat),gnu)), (3,(Some(rat),bee)), (4,(None,wo!f)),
(4,(None,bear)))

Expose Correct Answer

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

Explanation: solution : b.rightOuterJqin(d).collect rightOuterJoin [Pair] : Performs an right outer join using two key-value RDDs. Please note that the keys must be generally comparable to make this work correctly.

Next Question

Question discussion

You need to signup or login to add a comment

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))

Expose Correct Answer

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

Explanation: 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

Next Question

Question discussion

You need to signup or login to add a comment

Question 11

Problem Scenario 31 : You have given following two files
1. Content.txt: Contain a huge text file containing space separated words.
2. Remove.txt: Ignore/filter all the words given in this file (Comma Separated).
Write a Spark program which reads the Content.txt file and load as an RDD, remove all the words from a broadcast variables (which is loaded as an RDD of words from Remove.txt).
And count the occurrence of the each word and save it as a text file in HDFS.
Content.txt
Hello this is ABCTech.com
This is TechABY.com
Apache Spark Training
This is Spark Learning Session
Spark is faster than MapReduce
Remove.txt
Hello, is, this, the

Expose Correct Answer

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

Explanation: Solution : Step 1 : Create all three files in hdfs in directory called spark2 (We will do using Hue). However, you can first create in local filesystem and then upload it to hdfs Step 2 : Load the Content.txt file val content = sc.textFile("spark2/Content.txt") //Load the text file Step 3 : Load the Remove.txt file val remove = sc.textFile("spark2/Remove.txt") //Load the text file Step 4 : Create an RDD from remove, However, there is a possibility each word could have trailing spaces, remove those whitespaces as well. We have used two functions here flatMap, map and trim. val removeRDD= remove.flatMap(x=> x.splitf',")).map(word=>word.trim)//Create an array of words Step 5 : Broadcast the variable, which you want to ignore val bRemove = sc.broadcast(removeRDD.collect().toList) // It should be array of Strings Step 6 : Split the content RDD, so we can have Array of String. val words = content.flatMap(line => line.split(" ")) Step 7 : Filter the RDD, so it can have only content which are not present in "Broadcast Variable". val filtered = words.filter{case (word) => !bRemove.value.contains(word)} Step 8 : Create a PairRDD, so we can have (word,1) tuple or PairRDD. val pairRDD = filtered.map(word => (word,1)) Step 9 : Nowdo the word count on PairRDD. val wordCount = pairRDD.reduceByKey(_ + _) Step 10 : Save the output as a Text file. wordCount.saveAsTextFile("spark2/result.txt")

Next Question

Question discussion

You need to signup or login to add a comment

Question 12

Problem Scenario 3: You have been given MySQL DB with following details.
user=retail_dba
password=cloudera
database=retail_db
table=retail_db.categories
jdbc URL = jdbc:mysql://quickstart:3306/retail_db
Please accomplish following activities.
1. Import data from categories table, where category=22 (Data should be stored in categories subset)
2. Import data from categories table, where category>22 (Data should be stored in categories_subset_2)
3. Import data from categories table, where category between 1 and 22 (Data should be stored in categories_subset_3)
4. While importing catagories data change the delimiter to '|' (Data should be stored in categories_subset_S)
5. Importing data from catagories table and restrict the import to category_name,category id columns only with delimiter as '|'
6. Add null values in the table using below SQL statement ALTER TABLE categories modify category_department_id int(11); INSERT INTO categories values (eO.NULL.'TESTING');
7. Importing data from catagories table (In categories_subset_17 directory) using '|' delimiter and categoryjd between 1 and 61 and encode null values for both string and non string columns.
8. Import entire schema retail_db in a directory categories_subset_all_tables

Expose Correct Answer

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

Explanation: Solution: Step 1: Import Single table (Subset data} Note: Here the ' is the same you find on - key sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba -password=cloudera -table=categories ~warehouse-dir= categories_subset --where \'category_id\=22 --m 1 Step 2 : Check the output partition hdfs dfs -cat categoriessubset/categories/part-m-00000 Step 3 : Change the selection criteria (Subset data) sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba -password=cloudera -table=categories ~warehouse-dir= categories_subset_2 --where \category_id\\>22 -m 1 Step 4 : Check the output partition hdfs dfs -cat categories_subset_2/categories/part-m-00000 Step 5 : Use between clause (Subset data) sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba -password=cloudera -table=categories ~warehouse-dir=categories_subset_3 --where

"\category_id\' between 1 and 22" --m 1 Step 6 : Check the output partition hdfs dfs -cat categories_subset_3/categories/part-m-00000 Step 7 : Changing the delimiter during import. sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail dba - password=cloudera -table=categories -warehouse-dir=:categories_subset_6 --where "/categoryjd / between 1 and 22" -fields-terminated-by='|' -m 1 Step 8 : Check the.output partition hdfs dfs -cat categories_subset_6/categories/part-m-00000 Step 9 : Selecting subset columns sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail_dba - password=cloudera - table=categories --warehouse-dir=categories subset col -where "/category id/ between 1 and 22" - fields-terminated-by=T -columns=category name,category id --m 1 Step 10 : Check the output partition hdfs dfs -cat categories_subset_col/categories/part-m-00000 Step 11 : Inserting record with null values (Using mysql} ALTER TABLE categories modify category_department_id int(11); INSERT INTO categories values ^NULL/TESTING'); select" from categories; Step 12 : Encode non string null column sqoop import --connect jdbc:mysql://quickstart:3306/retail_db --username=retail dba - password=cloudera -table=categories --warehouse-dir=categortes_subset_17 -where "\"category_id\" between 1 and 61" -fields-terminated-by=,|' --null-string-N' -null-non- string=,N' - -m 1 Step 13 : View the content hdfs dfs -cat categories_subset_17/categories/part-m-00000 Step 14 : Import all the tables from a schema (This step will take little time) sqoop import-all-tables - connect jdbc:mysql://quickstart:3306/retail_db -- username=retail_dba -password=cloudera - warehouse-dir=categories_si Step 15 : View the contents hdfs dfs -Is categories_subset_all_tables Step 16 : Cleanup or back to originals. delete from categories where categoryid in (59,60); ALTER TABLE categories modify category_department_id int(11) NOTNULL; ALTER TABLE categories modify category_name varchar(45) NOT NULL; desc categories;

Next Question

Question discussion

You need to signup or login to add a comment

Question 13

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

Expose Correct Answer

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

Explanation: 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)

Next Question

Question discussion

You need to signup or login to add a comment

Question 14

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

Expose Correct Answer

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

Explanation: 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")

Next Question

Question discussion

You need to signup or login to add a comment

Question 15

Problem Scenario 54 : You have been given below code snippet.
val a = sc.parallelize(List("dog", "tiger", "lion", "cat", "panther", "eagle"))
val b = a.map(x => (x.length, x))
operation1
Write a correct code snippet for operationl which will produce desired output, shown below.
Array[(lnt, String)] = Array((4,lion), (7,panther), (3,dogcat), (5,tigereagle))

Expose Correct Answer

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

Explanation: Solution : b.foidByKey("")(_ + J.collect foldByKey [Pair] Very similar to fold, but performs the folding separately for each key of the RDD. This function is only available if the RDD consists of two-component tuples Listing Variants def foldByKey(zeroValue: V)(func: (V, V) => V):

RDD[(K, V)] def foldByKey(zeroValue: V, numPartitions: Int)(func: (V, V) => V): RDD[(K, V)] def foldByKey(zeroValue: V, partitioner: Partitioner)(func: (V, V) => V): RDD[(K, V)]

Next Question

Question discussion

You need to signup or login to add a comment