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.pass2lead.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 43 : You have been given following code snippet.

val grouped = sc.parallelize(Seq(((1,"twoM), List((3,4), (5,6)))))

val flattened = grouped.flatMap {A =>

groupValues.map { value => B }

}

You need to generate following output.

Hence replace A and B

Array((1,two,3,4),(1,two,5,6))

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

Solution :

A case (key, groupValues)

B (key._1, key._2, value._1, value._2)

QUESTION 2

Problem Scenario 81 : You have been given MySQL DB with following details. You have been given following product.csv file product.csv productID,productCode,name,quantity,price 1001,PEN,Pen Red,5000,1.23 1002,PEN,Pen Blue,8000,1.25 1003,PEN,Pen Black,2000,1.25 1004,PEC,Pencil 2B,10000,0.48 1005,PEC,Pencil 2H,8000,0.49 1006,PEC,Pencil HB,0,9999.99 Now accomplish following activities.

1.

Create a Hive ORC table using SparkSql

2.

Load this data in Hive table.

3.

Create a Hive parquet table using SparkSQL and load data in it.

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

Solution :

Step 1 : Create this tile in HDFS under following directory (Without header}

/user/cloudera/he/exam/task1/productcsv

Step 2 : Now using Spark-shell read the file as RDD

// load the data into a new RDD

val products = sc.textFile("/user/cloudera/he/exam/task1/product.csv")

// Return the first element in this RDD

prod u cts.fi rst()

Step 3 : Now define the schema using a case class

case class Product(productid: Integer, code: String, name: String, quantity:lnteger, price:

Float)

Step 4 : create an RDD of Product objects

val prdRDD = products.map(_.split(",")).map(p =>

Product(p(0).tolnt,p(1),p(2),p(3}.tolnt,p(4}.toFloat))

prdRDD.first()

prdRDD.count()

Step 5 : Now create data frame val prdDF = prdRDD.toDF()

Step 6 : Now store data in hive warehouse directory. (However, table will not be created }

import org.apache.spark.sql.SaveMode

prdDF.write.mode(SaveMode.Overwrite).format("orc").saveAsTable("product_orc_table")

step 7: Now create table using data stored in warehouse directory. With the help of hive.

hive

show tables

CREATE EXTERNAL TABLE products (productid int,code string,name string .quantity int,

price float}

STORED AS ore

LOCATION 7user/hive/warehouse/product_orc_table\\';

Step 8 : Now create a parquet table

import org.apache.spark.sql.SaveMode

prdDF.write.mode(SaveMode.Overwrite).format("parquet").saveAsTable("product_parquet_

table")

Step 9 : Now create table using this

CREATE EXTERNAL TABLE products_parquet (productid int,code string,name string

.quantity int, price float}

STORED AS parquet

LOCATION 7user/hive/warehouse/product_parquet_table\\';

Step 10 : Check data has been loaded or not.

Select * from products;

Select * from products_parquet;

QUESTION 3

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 4

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.

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

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;

QUESTION 5

Problem Scenario 60 : 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(List("dog","cat","gnu","salmon","rabbit","turkey","woif","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, (String, String))] = Array((6,(salmon,salmon)), (6,(salmon,rabbit)),

(6,(salmon,turkey)), (6,(salmon,salmon)), (6,(salmon,rabbit)),

(6,(salmon,turkey)), (3,(dog,dog)), (3,(dog,cat)), (3,(dog,gnu)), (3,(dog,bee)), (3,(rat,dog)),

(3,(rat,cat)), (3,(rat,gnu)), (3,(rat,bee)))

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

solution: b.join(d).collect join [Pair]: Performs an inner join using two key-value RDDs. Please note that the keys must be generally comparable to make this work. keyBy : Constructs two-component tuples (key-value pairs) by applying a function on each data item. The result of the function becomes the data item becomes the key and the original value of the newly created tuples.