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

- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

Problem Scenario 7 : 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 department tables using your custom boundary query, which import departments between 1 to 25.

2.

Also make sure each tables file is partitioned in 2 files e.g. part-00000, part-00002

3.

Also make sure you have imported only two columns from table, which are department_id,department_name

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

Solutions :

Step 1 : Clean the hdfs tile 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_itmes

hadoop fs -rm -R customers

Step 2 : Now import the department table as per requirement.

sqoop import \

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

--username=retail_dba \

-password=cloudera \

-table departments \

-target-dir /user/cloudera/departments \

-m2\

-boundary-query "select 1, 25 from departments" \

-columns department_id,department_name

Step 3 : Check imported data.

hdfs dfs -ls departments

hdfs dfs -cat departments/part-m-00000

hdfs dfs -cat departments/part-m-00001

QUESTION 2

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

val a = sc.parallelize(List("dog", "tiger", "lion", "cat", "spider", "eagle"), 2)

val b = a.keyBy(_.length)

val c = sc.parallelize(List("ant", "falcon", "squid"), 2)

val d = c.keyBy(.length)

operation 1

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

Array[(lnt, String)] = Array((4,lion))

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

Solution : b.subtractByKey(d).collect subtractByKey [Pair] : Very similar to subtract, but instead of supplying a function, the keycomponent of each pair will be automatically used as criterion for removing items from the first RDD.

QUESTION 3

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 4

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 5

Problem Scenario 73 : 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 file locally.

2.

Load this file 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.

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

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

Step 3 : Write spark script

#lmport SQLContext

from pyspark import SQLContext

#Create instance of SQLContext sqIContext = SQLContext(sc)

#Load json file

employee = sqlContext.jsonFile("employee.json")

#Register RDD as a temp table employee.registerTempTablef\\'EmployeeTab"}

#Select data from Employee table

employeelnfo = sqlContext.sql("select * from EmployeeTab"}

#lterate data and print

for row in employeelnfo.collect():

print(row)

Step 4 : Write dataas a Text file employeelnfo.toJSON().saveAsTextFile("employeeJson1") Step 5: Check whether data has been created or not hadoop fs -cat employeeJsonl/part"