100% Money Back Guarantee

Vendor: Cloudera

Exam Code:CCA175

Exam Name:CCA Spark and Hadoop Developer Exam

Version: Demo

QUESTION 1

Problem Scenario 49 : You have been given below code snippet (do a sum of values by

key}, with intermediate output.

val keysWithValuesList = Array("foo=A", "foo=A", "foo=A", "foo=A", "foo=B", "bar=C",

"bar=D", "bar=D")

val data = sc.parallelize(keysWithValuesl_ist}

//Create key value pairs

val kv = data.map(_.split("=")).map(v => (v(0), v(l))).cache()

val initialCount = 0;

val countByKey = kv.aggregateByKey(initialCount)(addToCounts, sumPartitionCounts)

Now define two functions (addToCounts, sumPartitionCounts) such, which will

produce following results.

Output 1

countByKey.collect

res3: Array[(String, Int)] = Array((foo,5), (bar,3))

import scala.collection._

val initialSet = scala.collection.mutable.HashSet.empty[String]

val uniqueByKey = kv.aggregateByKey(initialSet)(addToSet, mergePartitionSets)

Now define two functions (addToSet, mergePartitionSets) such, which will produce

following results.

Output 2:

uniqueByKey.collect

res4: Array[(String, scala.collection.mutable.HashSet[String])] = Array((foo,Set(B, A}},

(bar,Set(C, D}}}

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

Solution : val addToCounts = (n: Int, v: String) => n + 1 val sumPartitionCounts = (p1: Int, p2: Int} => p1 + p2 val addToSet = (s: mutable.HashSet[String], v: String) => s += v val mergePartitionSets = (p1: mutable.HashSet[String], p2: mutable.HashSet[String]) => p1 ++= p2

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 59 : You have been given below code snippet.

val x = sc.parallelize(1 to 20)

val y = sc.parallelize(10 to 30) operationl

z.collect

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

Array[Int] = Array(16,12, 20,13,17,14,18,10,19,15,11)

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

Solution :

val z = x.intersection(y)

intersection : Returns the elements in the two RDDs which are the same.

QUESTION 4

Problem Scenario 20 : 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. Write a Sqoop Job which will import "retaildb.categories" table to hdfs, in a directory name "categories_targetJob".

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

Solution :

Step 1 : Connecting to existing MySQL Database mysql -user=retail_dba -password=cloudera retail_db

Step 2 : Show all the available tables show tables;

Step 3 : Below is the command to create Sqoop Job (Please note that - import space is

mandatory)

sqoop job -create sqoopjob \ -- import \

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

-username=retail_dba \

-password=cloudera \

-table categories \

-target-dir categories_targetJob \

-fields-terminated-by \\'|\\' \ -lines-terminated-by \\'\n\\' Step 4 : List all the Sqoop Jobs sqoop job --list Step 5 : Show details of the Sqoop Job sqoop job --show sqoopjob Step 6 : Execute the sqoopjob sqoopjob --exec sqoopjob Step 7 : Check the output of import job hdfs dfs -Is categories_target_job hdfs dfs -cat categories_target_job/part*

QUESTION 5

Problem Scenario 58 : You have been given below code snippet.

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

a.keyBy(_.length)

operation1

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

Array[(lnt, Seq[String])] = Array((4,ArrayBuffer(lion)), (6,ArrayBuffer(spider)),

(3,ArrayBuffer(dog, cat)), (5,ArrayBuffer(tiger, eagle}}}

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

Solution :

b.groupByKey.collect

groupByKey [Pair]

Very similar to groupBy, but instead of supplying a function, the key-component of each

pair will automatically be presented to the partitioner.

Listing Variants

def groupByKeyQ: RDD[(K, lterable[V]}]

def groupByKey(numPartittons: Int): RDD[(K, lterable[V])]

def groupByKey(partitioner: Partitioner): RDD[(K, lterable[V])]

QUESTION 6

Problem Scenario 78 : You have been given MySQL DB with following details.

user=retail_dba

password=cloudera

database=retail_db

table=retail_db.orders

table=retail_db.order_items

jdbc URL = jdbc:mysql://quickstart:3306/retail_db

Columns of order table : (orderid , order_date , order_customer_id, order_status)

Columns of ordeMtems table : (order_item_td , order_item_order_id ,

order_item_product_id,

order_item_quantity,order_item_subtotal,order_item_product_price)

Please accomplish following activities.

1.

Copy "retail_db.orders" and "retail_db.order_items" table to hdfs in respective directory p92_orders and p92_order_items .

2.

Join these data using order_id in Spark and Python

3.

Calculate total revenue perday and per customer

4.

Calculate maximum revenue customer

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

Solution : Step 1 : Import Single table . sqoop import --connect jdbc:mysql://quickstart:3306/retail_db -username=retail_dba password=cloudera -table=orders --target-dir=p92_orders –m 1 sqoop import -connect jdbc:mysql://quickstart:3306/retail_db -username=retail_dba password=cloudera -table=order_items --target-dir=p92_order_orderitems --m 1 Note : Please check you dont have space between before or after \\'=\\' sign. Sqoop uses the MapReduce framework to copy data from RDBMS to hdfs Step 2 : Read the data from one of the partition, created using above command, hadoop fs -cat p92_orders/part-m-00000 hadoop fs -cat p92 orderitems/part-m-00000 Step 3 : Load these above two directory as RDD using Spark and Python (Open pyspark terminal and do following). orders = sc.textFile(Mp92_orders") orderitems = sc.textFile("p92_order_items") Step 4 : Convert RDD into key value as (orderjd as a key and rest of the values as a value)

#First value is orderjd

orders Key Value = orders.map(lambda line: (int(line.split(",")[0]), line))

#Second value as an Orderjd

orderltemsKeyValue = orderltems.map(lambda line: (int(line.split(",")[1]), line))

Step 5 : Join both the RDD using orderjd

joinedData = orderltemsKeyValue.join(ordersKeyValue)

#print the joined data

for line in joinedData.collect():

print(line)

#Format of joinedData as below.

#[Orderld, \\'All columns from orderltemsKeyValue\\', \\'All columns from ordersKeyValue\\']

ordersPerDatePerCustomer = joinedData.map(lambda line: ((line[1][1].split(",")[1],

line[1][1].split(",M)[2]), float(line[1][0].split(",")[4]))) amountCollectedPerDayPerCustomer =

ordersPerDatePerCustomer.reduceByKey(lambda runningSum, amount: runningSum +

amount}

#(Out record format will be ((date,customer_id), totalAmount} for line in

amountCollectedPerDayPerCustomer.collect(): print(line)

#now change the format of record as (date,(customer_id,total_amount))

revenuePerDatePerCustomerRDD = amountCollectedPerDayPerCustomer.map(lambda

threeElementTuple: (threeElementTuple[0][0],

(threeElementTuple[0][1],threeElementTuple[1])))

for line in revenuePerDatePerCustomerRDD.collect():

print(line)

#Calculate maximum amount collected by a customer for each day

perDateMaxAmountCollectedByCustomer =revenuePerDatePerCustomerRDD.reduceByKey(lambda runningAmountTuple,newAmountTuple: (runningAmountTuple if runningAmountTuple[1] >= newAmountTuple[1] else newAmountTuple})

for line in perDateMaxAmountCollectedByCustomer\sortByKey().collect(): print(line)

QUESTION 7

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.

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

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

---

QUESTION 8

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 9

Problem Scenario 13 : 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.

 Create a table in retailedb with following definition.

CREATE table departments_export (department_id int(11), department_name varchar(45),

created_date T1MESTAMP DEFAULT NOWQ);

2.

 Now import the data from following directory into departments_export table,

/user/cloudera/departments new

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

Solution :

Step 1 : Login to musql db

mysql --user=retail_dba -password=cloudera

show databases; use retail_db; show tables;

step 2 : Create a table as given in problem statement.

CREATE table departments_export (departmentjd int(11), department_name varchar(45),

created_date T1MESTAMP DEFAULT NOW());

show tables;

Step 3 : Export data from /user/cloudera/departmentsnew to new table departments_export

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

-username retaildba \

--password cloudera \

--table departments_export \

-export-dir /user/cloudera/departments_new \

-batch

Step 4 : Now check the export is correctly done or not. mysql -user*retail_dba password=cloudera

show databases;

use retail _db;

show tables;

select\\' from departments_export;

QUESTION 10

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

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

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

---

QUESTION 11

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 12

Problem Scenario 37 : ABCTECH.com has done survey on their Exam Products feedback using a web based form. With the following free text field as input in web ui. Name: StringSubscription Date: String Rating : String And servey data has been saved in a file called spark9/feedback.txt Christopher|Jan 11, 2015|5 Kapil|11 Jan, 2015|5 Thomas|6/17/2014|5 John|22-08-2013|5 Mithun|2013|5 Jitendra||5 Write a spark program using regular expression which will filter all the valid dates and save in two separate file (good record and bad record)

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

Solution :

Step 1 : Create a file first using Hue in hdfs.

Step 2 : Write all valid regular expressions sysntex for checking whether records are having

valid dates or not.

```
val regl =......(\d+)\s(\w{3})(,)\s(\d{4}).......r//11 Jan, 2015

val reg2 =......(\d+)(U)(\d+)(U)(\d{4})......s II 6/17/2014

val reg3 =......(\d+)(-)(\d+)(-)(\d{4})"""".r//22-08-2013

val reg4 =......(\w{3})\s(\d+)(,)\s(\d{4})......s II Jan 11, 2015
```

Step 3 : Load the file as an RDD.

```
val feedbackRDD = sc.textFile("spark9/feedback.txt"}
```

Step 4 : As data are pipe separated , hence split the same. val feedbackSplit =

```
feedbackRDD.map(line => line.split(\\'|\\'))
```

Step 5 : Now get the valid records as well as , bad records.

```
val validRecords = feedbackSplit.filter(x =>

(reg1.pattern.matcher(x(1).trim).matches|reg2.pattern.matcher(x(1).trim).matches|reg3.patt
```

ern.matcher(x(1).trim).matches | reg4.pattern.matcher(x(1).trim).matches))

val badRecords = feedbackSplit.filter(x =>

!(reg1.pattern.matcher(x(1).trim).matches|reg2.pattern.matcher(x(1).trim).matches|reg3.pat

tern.matcher(x(1).trim).matches | reg4.pattern.matcher(x(1).trim).matches))

Step 6 : Now convert each Array to Strings

val valid =vatidRecords.map(e => (e(0),e(1),e(2)))

val bad =badRecords.map(e => (e(0),e(1),e(2)))

Step 7 : Save the output as a Text file and output must be written in a single tile,

valid.repartition(1).saveAsTextFile("spark9/good.txt")

bad.repartition(1).saveAsTextFile("sparkS7bad.txt")