DumpsPedia

Cloudera

Exam CCA175

CCA Spark and Hadoop Developer Exam

Verson: Demo

[Total Questions: 10]

Question No : 1 CORRECT TEXT

Problem Scenario 95 : You have to run your Spark application on yarn with each executor Maximum heap size to be 512MB and Number of processor cores to allocate on each executor will be 1 and Your main application required three values as input arguments V1 V2 V3.

Please replace XXX, YYY, ZZZ

./bin/spark-submit -class com.hadoopexam.MyTask --master yarn-cluster--num-executors 3 --driver-memory 512m XXX YYY lib/hadoopexam.jarZZZ

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

Explanation:

Solution

XXX: -executor-memory 512m YYY: -executor-cores 1

ZZZ : V1 V2 V3

Notes : spark-submit on yarn options Option Description

archives Comma-separated list of archives to be extracted into the working directory of each executor. The path must be globally visible inside your cluster; see Advanced Dependency Management.

executor-cores Number of processor cores to allocate on each executor. Alternatively, you can use the spark.executor.cores property, executor-memory Maximum heap size to allocate to each executor. Alternatively, you can use the spark.executor.memory-property.

num-executors Total number of YARN containers to allocate for this application. Alternatively, you can use the spark.executor.instances property. queue YARN queue to submit to. For more information, see Assigning Applications and Queries to Resource Pools. Default: default.

Question No : 2 CORRECT TEXT

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\

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/retail_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

Question No : 3 CORRECT TEXT

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))

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

Question No : 4 CORRECT TEXT

Problem Scenario 75 : 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

Please accomplish following activities.

1. Copy "retail_db.order_items" table to hdfs in respective directory p90_order_items .

2. Do the summation of entire revenue in this table using pyspark.

3. Find the maximum and minimum revenue as well.

4. Calculate average revenue

Columns of ordeMtems table : (order_item_id , order_item_order_id , order_item_product_id, order_item_quantity,order_item_subtotal,order_

item_subtotal,order_item_product_price)

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

Explanation:

Solution :

Step 1 : Import Single table .

sqoop import --connect jdbc:mysql://quickstart:3306/retail_db -username=retail_dba -password=cloudera -table=order_items --target -dir=p90 ordeMtems --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 p90_order_items/part-m-00000

Step 3 : In pyspark, get the total revenue across all days and orders. entire TableRDD = sc.textFile("p90_order_items")

#Cast string to float

extractedRevenueColumn = entireTableRDD.map(lambda line: float(line.split(",")[4]))

Step 4 : Verify extracted data

for revenue in extractedRevenueColumn.collect():

print revenue

#use reduce'function to sum a single column vale

totalRevenue = extractedRevenueColumn.reduce(lambda a, b: a + b)

Step 5 : Calculate the maximum revenue

maximumRevenue = extractedRevenueColumn.reduce(lambda a, b: (a if a>=b else b))

Step 6 : Calculate the minimum revenue

minimumRevenue = extractedRevenueColumn.reduce(lambda a, b: (a if a<=b else b))

Step 7 : Caclculate average revenue

count=extractedRevenueColumn.count()

averageRev=totalRevenue/count

Question No : 5 CORRECT TEXT

Problem Scenario 36 : You have been given a file named spark8/data.csv (type,name).

data.csv

1,Lokesh

2,Bhupesh

2,Amit

2,Ratan

2,Dinesh

1,Pavan

1,Tejas

2,Sheela

1,Kumar

1,Venkat

1. Load this file from hdfs and save it back as (id, (all names of same type)) in results directory. However, make sure while saving it should be

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 data.csv file from hdfs and create PairRDDs

```
val name = sc.textFile("spark8/data.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 combine the rdd by key.
val combinedOutput = namePairRDD.combineByKey(List(_), (x:List[String], y:String) => y :: x, (x:List[String], y:List[String]) => x ::: y)

Step 5 : Save the output as a Text file and output must be written in a single file.
:ombinedOutput.repartition(1).saveAsTextFile("spark8/result.txt")

Question No : 6 CORRECT TEXT

Problem Scenario 40 : You have been given sample data as below in a file called spark15/file1.txt

3070811,1963,1096,,"US","CA",,1,

3022811,1963,1096,,"US","CA",,1,56

3033811,1963,1096,,"US","CA",,1,23

Below is the code snippet to process this tile.

val field= sc.textFile("spark15/f ilel.txt")

val mapper = field.map(x=> A)

mapper.map(x => x.map(x=> {B})).collect

Please fill in A and B so it can generate below final output

Array(Array(3070811,1963,109G, 0, "US", "CA", 0,1, 0)

,Array(3022811,1963,1096, 0, "US", "CA", 0,1, 56)

,Array(3033811,1963,1096, 0, "US", "CA", 0,1, 23)

)

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

Explanation:

Solution :

A. x.split(",”-1)

B. if (x. isEmpty) 0 else x

Question No : 7 CORRECT TEXT

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

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

Explanation:

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 No : 8 CORRECT TEXT

Problem Scenario 51 : You have been given below code snippet.

val a = **sc.parallelize(List(1, 2,**1, 3), **1)**

val b = **a.map((_, "b"))**

val c = **a.map((_, "c"))**

Operation_xyz

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

Output:

Array[(lnt, **(Iterable[String], Iterable[String]))]** = Array(

(2,(ArrayBuffer(b),ArrayBuffer(c))),

(3,(ArrayBuffer(b),ArrayBuffer(c))),

(1,(ArrayBuffer(b, b),ArrayBuffer(c, c)))

)

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

Explanation:

Solution :

b.cogroup(c).collect

cogroup [Pair], groupWith [Pair]

A very powerful set of functions that allow grouping up to 3 key-value RDDs together using their keys.

Another example

val x = sc.parallelize(List((1, "apple"), (2, "banana"), (3, "orange"), (4, "kiwi")), 2)

val y = sc.parallelize(List((5, "computer"), (1, "laptop"), (1, "desktop"), (4, "iPad")), 2)

x.cogroup(y).collect

Array[(lnt, (Iterable[String], Iterable[String]))] = Array(

(4,(ArrayBuffer(kiwi),ArrayBuffer(iPad))),
(2,(ArrayBuffer(banana),ArrayBuffer())),
(3,(ArrayBuffer(orange),ArrayBuffer())),
(1 ,(ArrayBuffer(apple),ArrayBuffer(laptop, desktop))),
(5,{ArrayBuffer(),ArrayBuffer(computer))))

Question No : 9 CORRECT TEXT

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.

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;

Question No : 10 CORRECT TEXT

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

Answer: See the explanation for Step by Step Solution and configuration.
Explanation:
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(UNIXJTlMESTAMP(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 TlMESTAMP}}}/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(UNIXJTlMESTAMP(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();