DUMPSBOSS.

CCA Spark and Hadoop Developer Exam

Cloudera CCA175

Version Demo

Total Demo Questions: 7

Total Premium Questions: 96

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

QUESTION NO: 1 - (SIMULATION)

SIMULATION9

SIMULATION

Problem Scenario 19 : 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

Now accomplish following activities.

1. Import departments table from mysql to hdfs as textfile in departments_text directory.

2. Import departments table from mysql to hdfs as sequncefile in departments_sequence directory.

3. Import departments table from mysql to hdfs as avro file in departments avro directory.

4. Import departments table from mysql to hdfs as parquet file in departments_parquet directory.

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

Explanation:

Solution :

Step 1 : Import departments table from mysql to hdfs as textfile

sqoop import \

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

~username=retail_dba \

-password=cloudera \

-table departments \

-as-textfile \

-target-dir=departments_text

verify imported data

hdfs dfs -cat departments_text/part"

Step 2 : Import departments table from mysql to hdfs as sequncetlle

sqoop import \

-connect jdbc:mysql://quickstart:330G/retail_db \

~username=retail_dba \

-password=cloudera \

--table departments \

-as-sequencetlle \

-~target-dir=departments sequence

verify imported data

hdfs dfs -cat departments_sequence/part*

Step 3 : Import departments table from mysql to hdfs as sequncetlle

sqoop import \

-connect jdbc:mysql://quickstart:330G/retail_db \

~username=retail_dba \

--password=cloudera \

--table departments \

--as-avrodatafile \

--target-dir=departments_avro

verify imported data

hdfs dfs -cat departments avro/part*

Step 4 : Import departments table from mysql to hdfs as sequncetlle

sqoop import \

-connect jdbc:mysql://quickstart:330G/retail_db \

~username=retail_dba \

--password=cloudera \

-table departments \

-as-parquetfile \

-target-dir=departments_parquet

verify imported data

hdfs dfs -cat departmentsparquet/part*

QUESTION NO: 2 - (SIMULATION)

SIMULATION7

SIMULATION

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)))

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

Explanation:

Solution :

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

QUESTION NO: 3 - (SIMULATION)

SIMULATION4

SIMULATION

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.

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

Explanation:

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 NO: 4 - (SIMULATION)

SIMULATION5

SIMULATION

Problem Scenario 65 : You have been given below code snippet.

val a = sc.parallelize(List("dog", "cat", "owl", "gnu", "ant"), 2)

val b = sc.parallelize(1 to a.count.toInt, 2)

val c = a.zip(b)

operation1

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

Array[(String, Int)] = Array((owl,3), (gnu,4), (dog,1), (cat,2>, (ant,5))

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

Explanation:

Solution : c.sortByKey(false).collect

sortByKey [Ordered] : This function sorts the input RDD's data and stores it in a new RDD. "The output RDD is a shuffled RDD because it stores data that is output by a reducer which has been shuffled. The implementation of this function is actually very clever. First, it uses a range partitioner to partition the data in ranges within the shuffled RDD.

Then it sorts these ranges individually with mapPartitions using standard sort mechanisms.

QUESTION NO: 5 - (SIMULATION)

SIMULATION0

SIMULATION

Problem Scenario 90 : You have been given below two files

course.txt

id,course

1,Hadoop

2,Spark

3,HBase

fee.txt

id,fee

2,3900

3,4200

4,2900

Accomplish the following activities.

1. Select all the courses and their fees , whether fee is listed or not.

2. Select all the available fees and respective course. If course does not exists still list the fee

3. Select all the courses and their fees , whether fee is listed or not. However, ignore records having fee as null.

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

Explanation:

Solution :

Step 1:

hdfs dfs -mkdir sparksql4

hdfs dfs -put course.txt sparksql4/

hdfs dfs -put fee.txt sparksql4/

Step 2 : Now in spark shell

// load the data into a new RDD

val course = sc.textFile("sparksql4/course.txt")

val fee = sc.textFile("sparksql4/fee.txt")

// Return the first element in this RDD

course.fi rst()

fee.fi rst()

//define the schema using a case class case class Course(id: Integer, name: String) case class Fee(id: Integer, fee: Integer)

// create an RDD of Product objects

val courseRDD = course.map(_.split(",")).map(c => Course(c(0).toInt,c(1)))

val feeRDD =fee.map(_.split(",")).map(c => Fee(c(0}.toInt,c(1}.toInt))

courseRDD.first()

courseRDD.count(}

feeRDD.first()

feeRDD.countQ

// change RDD of Product objects to a DataFrame val courseDF = courseRDD.toDF(} val feeDF = feeRDD.toDF{)

// register the DataFrame as a temp table courseDF. registerTempTable("course") feeDF. registerTempTablef'fee")

// Select data from table

val results = sqlContext.sql(.....SELECT' FROM course """)

results. showQ

val results = sqlContext.sql(.....SELECT' FROM fee.....)

results. showQ

val results = sqlContext.sql(.....SELECT * FROM course LEFT JOIN fee ON course.id = fee.id.....)

results-showQ

val results ="sqlContext.sql(.....SELECT * FROM course RIGHT JOIN fee ON course.id = fee.id "MM)

results. showQ

val results = sqlContext.sql(.....SELECT' FROM course LEFT JOIN fee ON course.id = fee.id where fee.id IS NULL"

results. show()

QUESTION NO: 6 - (SIMULATION)

Problem Scenario 92 : You have been given a spark scala application, which is bundled in jar named hadoopexam.jar.

Your application class name is com.hadoopexam.MyTask

You want that while submitting your application should launch a driver on one of the cluster node.

Please complete the following command to submit the application.

spark-submit XXX -master yarn \

YYY SSPARK HOME/lib/hadoopexam.jar 10

ANSWER: See the explanation for Step by Step Solution and configuration

Explanation:

Solution

XXX: -class com.hadoopexam.MyTask

YYY : --deploy-mode cluster

QUESTION NO: 7 - (SIMULATION)

SIMULATION2

SIMULATION

Problem Scenario 12 : 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_new (department_id int(11), department_name varchar(45), created_date T1MESTAMP DEFAULT NOW());

2. Now isert records from departments table to departments_new

3. Now import data from departments_new table to hdfs.

4. Insert following 5 records in departmentsnew table. Insert into departments_new values(110, "Civil" , null); Insert into departments_new values(111, "Mechanical" , null); Insert into departments_new values(112, "Automobile" , null); Insert into departments_new values(113, "Pharma" , null);

Insert into departments_new values(114, "Social Engineering" , null);

5. Now do the incremental import based on created_date column.

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

Explanation:

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_new (department_id int(11), department_name varchar(45), createddate T1MESTAMP DEFAULT NOW());

show tables;

Step 3 : isert records from departments table to departments_new insert into departments_new select a.", null from departments a;

Step 4 : Import data from departments new table to hdfs.

sqoop import \

-connect jdbc:mysql://quickstart:330G/retail_db \

~username=retail_dba \

-password=cloudera \

-table departments_new\

--target-dir /user/cloudera/departments_new \

--split-by departments

Stpe 5 : Check the imported data.

hdfs dfs -cat /user/cloudera/departmentsnew/part"

Step 6 : Insert following 5 records in departmentsnew table.

Insert into departments_new values(110, "Civil" , null);

Insert into departments_new values(111, "Mechanical" , null);

Insert into departments_new values(112, "Automobile" , null);

Insert into departments_new values(113, "Pharma" , null);

Insert into departments_new values(114, "Social Engineering" , null);

commit;

Stpe 7 : Import incremetal data based on created_date column.

sqoop import \

-connect jdbc:mysql://quickstart:330G/retail_db \

-username=retail_dba \

-password=cloudera \

--table departments_new\

-target-dir /user/cloudera/departments_new \

-append \

-check-column created_date \

-incremental lastmodified \

-split-by departments \

-last-value "2016-01-30 12:07:37.0"

Step 8 : Check the imported value.

hdfs dfs -cat /user/cloudera/departmentsnew/part"