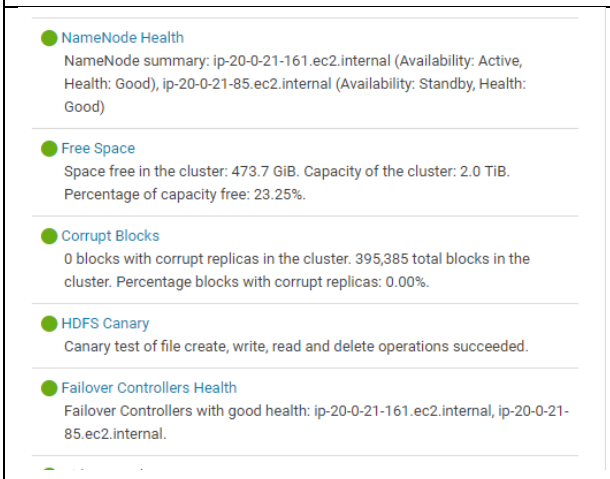
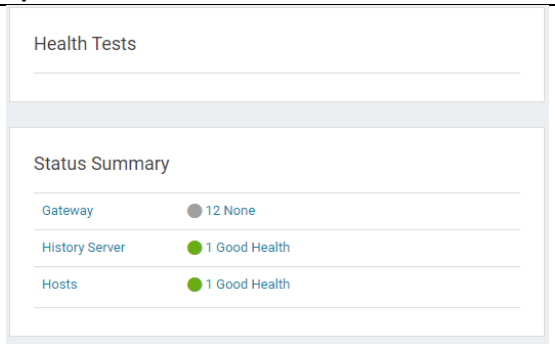
1. Verify the cluster health including HDFS and Spark

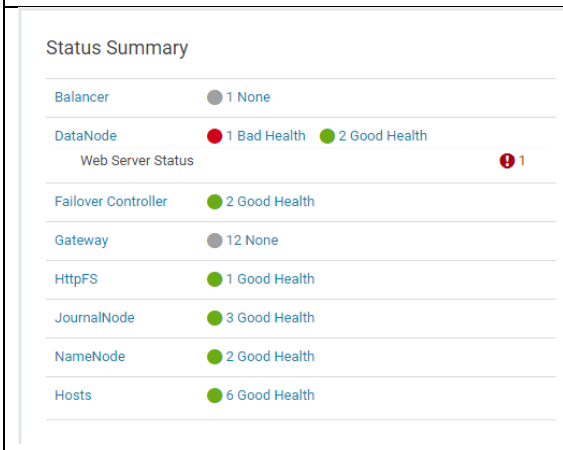
Cloudera manager contains everything in one place. There is one Hadoop cluster, educluster1. History Server Web UI shows submitted jobs. Name Node UI shows status of Data Nodes and Name Nodes.

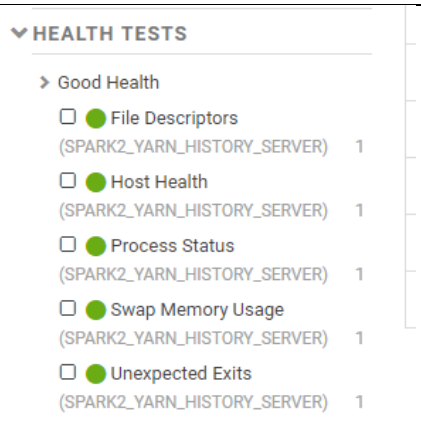
HDFS

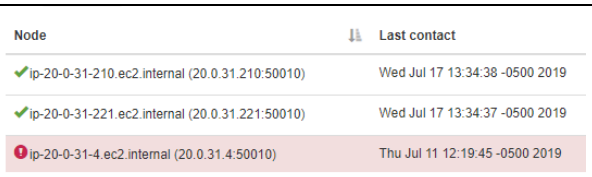


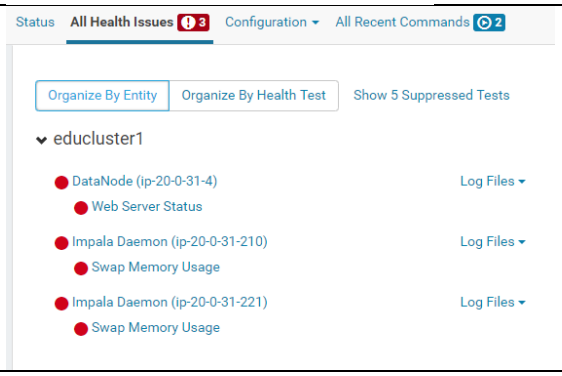
Spark2











2. Test the spark environment by executing the spark's sort.py example.

- List existing examples

```
ls /opt/cloudera/parcels/SPARK2/lib/spark2/examples/src/main/python/
```

- Copy example files to data

```
cd data
cp /opt/cloudera/parcels/SPARK2/lib/spark2/examples/src/main/python/sort.py sort.py
```

- Execute it

```
spark2-submit data/sort.py use_cases/input_sort_py.txt
```

The input file contains series of numbers in multiple lines. The above job sorted them and printed the final result to the terminal.

```
0 0 0 1 1 1 2 2 2 2 2 3 3 3 7 7 7 8 8 8 8 8 10 10 14 14 15 15
```

3. Try to implement the same example in scala and perform spark-submit.

```
mkdir scala_directory
cd scala_directory
create build file vi build.sbt
sbt compile
mkdir -p src/main/scala/com/edureka/training
src/main/scala/com/edureka/training
mkdir -p src/main/scala/com/edureka/training/m4/
```

- Create a new file sort.scala and add scala code

```
import org.apache.spark.sql.SparkSession
object ScalaSort {
  /** Usage: HdfsTest [file] */
  def main(args: Array[String]) {
    if (args.length < 1) {
      System.err.println("Usage: HdfsTest <file>")
      System.exit(1)
    }
    val spark = SparkSession.builder.appName("ScalaSortTGA").getOrCreate()
```

```
  val lines = spark.read.text(args(0)).rdd.map(r => r(0))
  val sortedCount = lines.flatMap(line => line.split(' ')).map(_.toInt).sortByKey()
  val output = sortedCount.collect()
  for (x <- output){
    println(x)
  }
  spark.stop()
 }
}
```

- Compile, Package

```
sbt compile
sbt package
```

- Submit

```
spark2-submit --class com.edureka.training.m4.ScalaSort --deploy-mode client target/scala-2.11/sparkme-project_2.11-1.0.jar use_cases/input_sort_py.txt
```

4. Analyze the behavior of spark application on Spark web UI

I went to spark web UI http://bdlabs.edureka.co:50014 and checked the Jobs, Stages, and Executors

5. Add custom logs in your application and re-execute the application. Once executed check the Spark logs.

- Create source directory “m4” in project source folder

```
mkdir -p src/main/scala/com/edureka/training/m4/
```

- Copy HdfsTest.scala in “m4”

```
cp/opt/cloudera/parcels/SPARK2/lib/spark2/examples/src/main/scala/org/apache/spark/examples/HdfsTest.scala src/main/scala/com/edureka/training/m4/
```

- Change package to

```
com.edureka.training.m4
```

- Add this dependency to build.sbt

```
libraryDependencies += "com.typesafe.scala-logging" %% "scala-logging" % "3.9.0"
```

- Add this import on top of HdfsTest.scala and add custom logs

```
import com.typesafe.scalalogging.Logger
```

logger.info("Hello there!")

- Compile, Package

sbt compile
sbt package

- Submit

spark2-submit --class com.edureka.training.m4.HdfsTest --deploy-mode client target/scala-2.11/sparkme-project_2.11-1.0.jar use_cases/input_sort_py.txt

- Collect logs from yarn

yarn logs --applicationId application_1528714825862_137643

5. Transfer complete dataset from RDBMS to HDFS

- Upload 6 csv files using ftp
- Login to mysql, change database, create table

```
mysql -h mysqldb.edu.cloudlab.com -u labuser --password=edureka

create database instacart672184

use instacart672184
```

- create tables using this script

```
CREATE TABLE `aisles` (
  `aisle_id` int(11) NOT NULL AUTO_INCREMENT,
  `aisle` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`aisle_id`)
);
CREATE TABLE `departments` (
  `department_id` int(11) NOT NULL AUTO_INCREMENT,
  `department` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`department_id`)
);
CREATE TABLE `order_products_prior` (
  `order_id` int(11) DEFAULT NULL,
  `product_id` int(11) DEFAULT NULL,
  `add_to_cart_order` int(11) DEFAULT NULL,
  `reordered` int(11) DEFAULT NULL
);
CREATE TABLE `order_products_train` (
  `order_id` int(11) DEFAULT NULL,
  `product_id` int(11) DEFAULT NULL,
```

```
  `add_to_cart_order` int(11) DEFAULT NULL,
  `reordered` int(11) DEFAULT NULL
);
CREATE TABLE `orders` (
  `order_id` int(11) NOT NULL,
  `user_id` int(11) DEFAULT NULL,
  `eval_set` varchar(20) DEFAULT NULL,
  `order_number` int(11) DEFAULT NULL,
  `order_dow` int(11) DEFAULT NULL,
  `order_hour_of_day` int(11) DEFAULT NULL,
  `days_since_prior_order` float(3,1) DEFAULT NULL,
  PRIMARY KEY (`order_id`)
);
CREATE TABLE `products` (
  `product_id` int(11) NOT NULL,
  `product_name` varchar(100) DEFAULT NULL,
  `aisle_id` int(11) DEFAULT NULL,
  `department_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`product_id`)
);
CREATE TABLE `sample_submission` (
  `order_id` int(11) NOT NULL,
  `products` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`order_id`)
) ;
```

```
show tables;
```

Tables_in_instacart672184

| aisles |

| departments |

| order_products_prior |

| order_products_train |

| orders |

| products |

| sample_submission |

+--------------------------+

- Load csv into table

load data local infile '/mnt/home/edureka_672184/data/aisles.csv' into table aisles FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/departments.csv' into table departments FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/order_products__prior.csv' into table order_products_prior FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/order_products__train.csv' into table order_products_train FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/orders.csv' into table orders FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/products.csv' into table products FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

load data local infile '/mnt/home/edureka_672184/data/sample_submission.csv' into table sample_submission FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n' IGNORE 1 LINES;

- Sqoop import

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table aisles -m 1 --target-dir /user/edureka_672184/use_cases/instacart/aisles/

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table departments -m 1 --target-dir /user/edureka_672184/use_cases/instacart/departments /

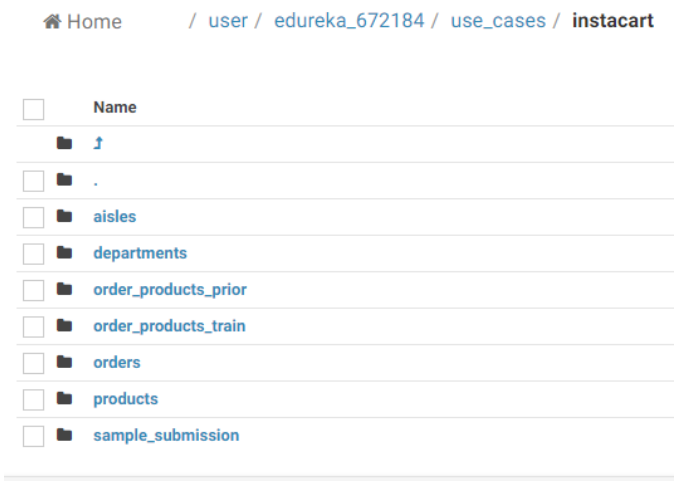
sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table order_products_prior -m 1 --target-dir /user/edureka_672184/use_cases/instacart/order_products_prior/

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table order_products_train -m 1 --target-dir /user/edureka_672184/use_cases/instacart/order_products_train/

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table orders -m 1 --target-dir /user/edureka_672184/use_cases/instacart/orders/

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table products -m 1 --target-dir /user/edureka_672184/use_cases/instacart/products/

sqoop import --connect jdbc:mysql://sqoopdb.edu.cloudlab.com/instacart672184 --username labuser -password edureka --table sample_submission -m 1 --target-dir /user/edureka_672184/use_cases/instacart/sample_submission/



7. Validate the loaded data by comparing the statistics of data both in source and HDFS

hdfs dfs -cat /user/edureka_672184/use_cases/Instacart/aisles/*|wc -l

SELECT count(*) FROM aisles;

= 134

hdfs dfs -cat /user/edureka_672184/use_cases/Instacart/products/*|wc -l

SELECT count(*) FROM products;

= 49355, hdfs shows 49400

hdfs dfs -cat /user/edureka_672184/use_cases/Instacart/orders/*|wc -l

SELECT count(*) FROM orders;

= 3421083

hdfs dfs -cat /user/edureka_672184/use_cases/instacart/order_products_train/*|head -5

1,49302,1,1

1,11109,2,1

1,10246,3,0

1,49683,4,0

1,43633,5,1

SELECT * FROM order_products_train LIMIT 5;

order_id	product_id	add_to_cart_order	reordered
1	49302	1	1
1	11109	2	1
1	10246	3	0
1	49683	4	0
1	43633	5	1

8. Create a new directory in HDFS named cheeses and load only rows where aisle is "specialty cheeses"

- create directory

hdfs dfs -mkdir /user/edureka_672184/use_cases/instacart/cheeses

- Test this query on mysql by creating a view

CREATE VIEW chessy AS

SELECT a.aisle, op.order_id, p.product_id, p.product_name FROM aisles a JOIN products p ON a.aisle_id = p.aisle_id JOIN order_products_train op ON op.product_id = p.product_id WHERE a.aisle LIKE 'specialty cheeses'

SELECT * FROM chessy LIMIT 10;

aisle	order_id	product_id	product_name
specialty cheeses	36	39612	Grated Pecorino Romano Cheese
specialty cheeses	98	47333	Queso Fresco
specialty cheeses	1032	21901	Le Petite Fromage Parmesan & Basil
specialty cheeses	1620	13093	Camembert
specialty cheeses	2445	13409	Dubliner Wedge Pre Cut Cheese
specialty cheeses	2869	14511	Grated Pecorino Romano
specialty cheeses	3733	48969	Cheese Plate Spanish
specialty cheeses	3901	37524	Fresh Mozzarella Ball
specialty cheeses	4562	117	Petit Suisse Fruit
specialty cheeses	5466	39612	Grated Pecorino Romano Cheese

sqoop import --connect jdbc:mysql://mysqldb.edu.cloudlab.com/instacart672184 --username labuser --password edureka --query 'SELECT * FROM chessy WHERE $CONDITIONS' -m 1 -target-dir '/user/edureka_672184/instacart/cheeses'

9. update "specialty cheeses" to " specialty cheese" and transfer only updated rows in the above created directory.

- update table through the view

UPDATE cheesy SET aisle = 'specialty cheese' WHERE aisle LIKE 'specialty cheeses';

- Recreate the view

CREATE VIEW chessy AS

SELECT a.aisle, op.order_id, p.product_id, p.product_name FROM aisles a JOIN products p ON a.aisle_id = p.aisle_id JOIN order_products_train op ON op.product_id = p.product_id WHERE a.aisle LIKE 'specialty cheese'

SELECT * FROM chessy LIMIT 10;

aisle	order_id	product_id	product_name
specialty cheese	36	39612	Grated Pecorino Romano Cheese
specialty cheese	98	47333	Queso Fresco
specialty cheese	1032	21901	Le Petite Fromage Parmesan & Basil
specialty cheese	1620	13093	Camembert
specialty cheese	2445	13409	Dubliner Wedge Pre Cut Cheese
specialty cheese	2869	14511	Grated Pecorino Romano
specialty cheese	3733	48969	Cheese Plate Spanish
specialty cheese	3901	37524	Fresh Mozzarella Ball
specialty cheese	4562	117	Petit Suisse Fruit
specialty cheese	5466	39612	Grated Pecorino Romano Cheese

- Create a column that stores last modified date for each row: last_mod
- Transfer updated view based last modified date

sqoop import --connect jdbc:mysql://mysqldb.edu.cloudlab.com/instacart672184 --username=labuser --password=edureka --table chessy -target-dir '/user/edureka_672184/cheeses' --incremental lastmodified --check-column last_mod --merge-key order_id --last-value 2019.07.16 -m 1