

BigData – Spark

Hands-On File IO - Parquet

Arnaud Nauwynck
Nov 2022

Objectives of TD2



1/ read CSV files, convert options



2/ read combine several files



3/ others file format: json, nd-json, xml, text



4/ write to PARQUET files dir
repartitionning

5/ Discover PARQUET format optimisations

Exercise 15: (similar to exercise 9)

save as parquet, load parquet file

a/ Convert Dataset addresses loaded (.. from CSV files)
into PARQUET files

b/ look at written file

compare file size of csv/ json with Parquet

c/ reload dataset from Parquet file

d/ notice more memory needs during conversion, logs example

WARN MemoryManager: Total allocation exceeds 95,00% ..of heap memory
Scaling row group sizes to 96,54% for 7 writers

Exercise 16 : (similar to exercise 10) analyze written PARQUET files

Repeat the same operation twice, and notice that

1/ you can not specify the file names .. Only an empty directory name !

2/ the file names are generated with a unique UUID
part-00{partNumber}-{UUID}-c000.snappy.parquet

3/ during write time... spark write to « _temporary » sub dir, then rename at the end

4/ spark has written several files ...

If using the full dataset of 3Giga .. Spark write 26 parquet files

Objectives of TD2



1/ read CSV files, convert options



2/ read combine several files



3/ others file format: json, nd-json, xml, text



4/ write to PARQUET files dir
repartitionning



5/ Discover PARQUET format optimisations

Exercise 17 : use repartition(1) before writing ... redo

a/ Use allAdressDs.repartition(1)
then write to parquet
(choose a new empty directory path)

b/ analyse written file

c/ Explain what it does

Exercise 18: use .repartition(N) ... check N files are written

Redo same as exercise 17, with

.repartition(2) => check 2 files are written
.repartition(3) => check 3 files are written
.repartition(4) => check 4 files are written

got the idea ?

Optionnally: Measure performances of writing for each scenario

Objectives of TD2



1/ read CSV files, convert options



2/ read combine several files



3/ others file format: json, nd-json, xml, text



4/ write to PARQUET files dir
repartitionning



5/ Discover PARQUET format optimisations

Exercise 19

Compare total size of parquet files

1/ file size when using repartition(1)

2/ total file sizes when using repartition(2)

3/ total file sizes when using repartition(3)

(Hint: on cygwin / linux, you can use « du -Sh »)

Questions:

a/ check that less files is better

b/ explain why

Exercise 20 : columnar file format

a/ execute 2 different queries on addressDs (from spark.read.parquet ...)

```
addressDs.select("commune_nom").distinct().count()
```

```
addressDs.select(« voie_nom").distinct().count()
```

b/ Check using Spark UI (<http://localhost:4000>)

that « Input » bytes size is different while reading only 1 or another column
(parquet is a « Columnar File format »)

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output
37	addressDs.select('commune_nom').distinct().count() + details	2022/11/22 21:43:35	2 s	<div><div></div></div> 8/8	677.6 KiB	

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output
43	addressDs.select('voie_nom').distinct().count() + details	2022/11/22 21:44:03	4 s	<div><div></div></div> 8/8	1509.6 KiB	

c/ repeat same 2 queries, on dataSet read from csv or json (not a « Columnar file format »)

... check in spark UI that the « Input » are same !

49	addressCsvDs.select('commune_nom').distinct().count() + details	2022/11/22 21:48:09	36 s	<div><div></div></div> 26/26	3.1 GiB	
55	addressCsvDs.select('voie_nom').distinct().count() + details	2022/11/22 21:49:55	38 s	<div><div></div></div> 26/26	3.1 GiB	

Exercise 21 : always use `.sortWithinPartition(..)`
after a `.repartition()` / before `.write()` !!

Redo `write()` ... by adding « `.sortWithinPartition()` »
which columns are good candidates to be sorted in « address » dataset?

Questions

b/ explain why it will improve performances to find addresses

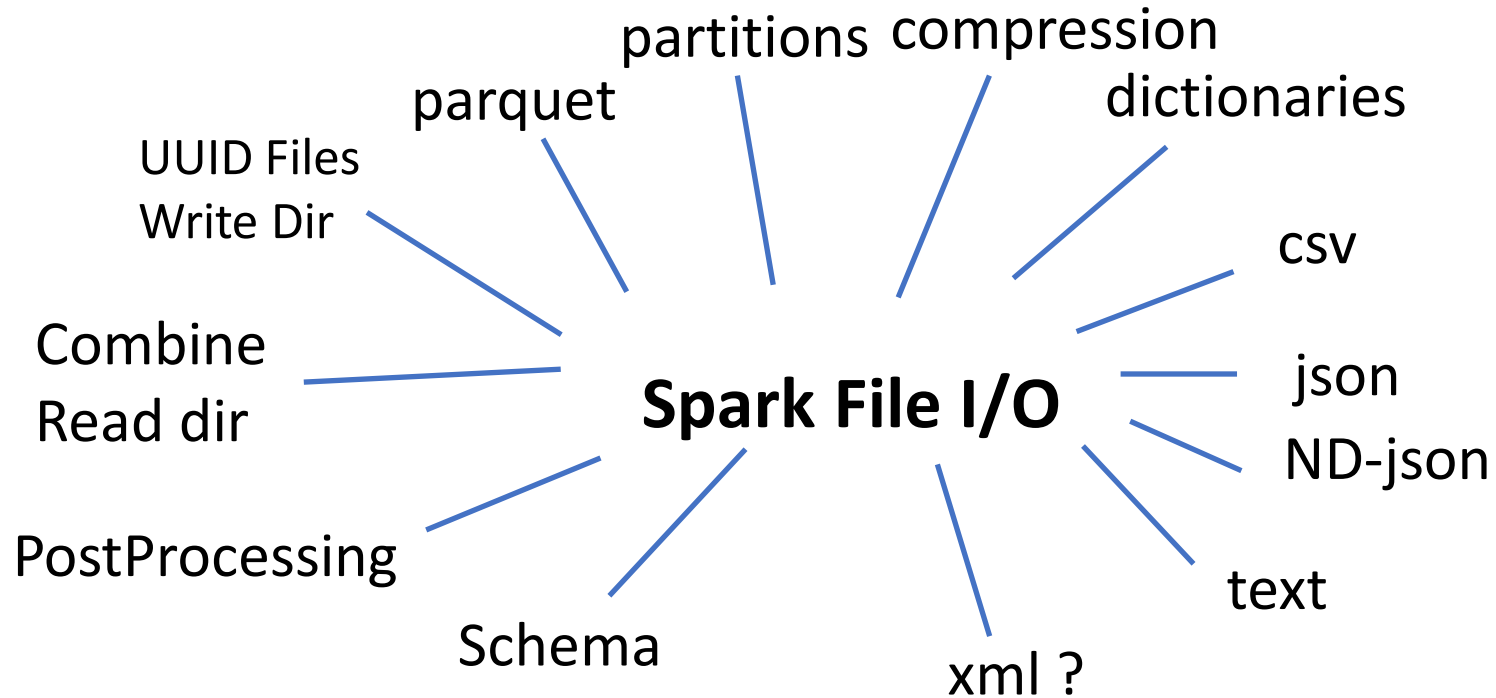
c/ explain why it will also improve Dictionary compression

Exercise 22 : MindMap

Draw a MindMap to summarize
what you did and learn from this TD session

Your MindMap should
start with word « Spark File I/O » in the middle
Then draw star edges to other word chapters and sub-chapters

Exercise 21



Bonus Exercise ...

Your friends are still struggling with spark parquet ?

Option 1/ Help your friends

Option 2/ do bonus exercise 22/ on parquet-cli File viewer

Exercise 22 : use paquet-cli to view internal file Page / ColumnChunk / Dictionaries ..

Checkout tool from official site:

<https://github.com/apache/parquet-mr/tree/master/parquet-cli>

(notice the tool was previously called parquet-tool)

Focus on

parquet « **pages** »

parquet « **dictionary** »

```
Usage: parquet [options] [command] [command options]
```

```
Options:
```

```
-v, --verbose, --debug
    Print extra debugging information
```

```
Commands:
```

```
help
    Retrieves details on the functions of other commands
meta
    Print a Parquet file's metadata
```

```
pages
    Print page summaries for a Parquet file
```

```
dictionary
    Print dictionaries for a Parquet column
```

```
check-stats
    Check Parquet files for corrupt page and column stats (PARQUET-251)
```

```
schema
    Print the Avro schema for a file
```

```
csv-schema
    Build a schema from a CSV data sample
```

```
convert-csv
    Create a file from CSV data
```

```
convert
    Create a Parquet file from a data file
```

```
to-avro
    Create an Avro file from a data file
```

```
cat
    Print the first N records from a file
```

```
head
    Print the first N records from a file
```

Questions ?

Take-Away

What You learned ?

Questions ?

Next Steps

More CMs

More TDs

Spark concepts:

- SQL
- Column Pruning / Partition Pruning / Predicate-Push-Down
- DAG, Distribution, Optimizations
- Spark Clustering
- Java binding, UDF, map
- Spark Streaming
- ...