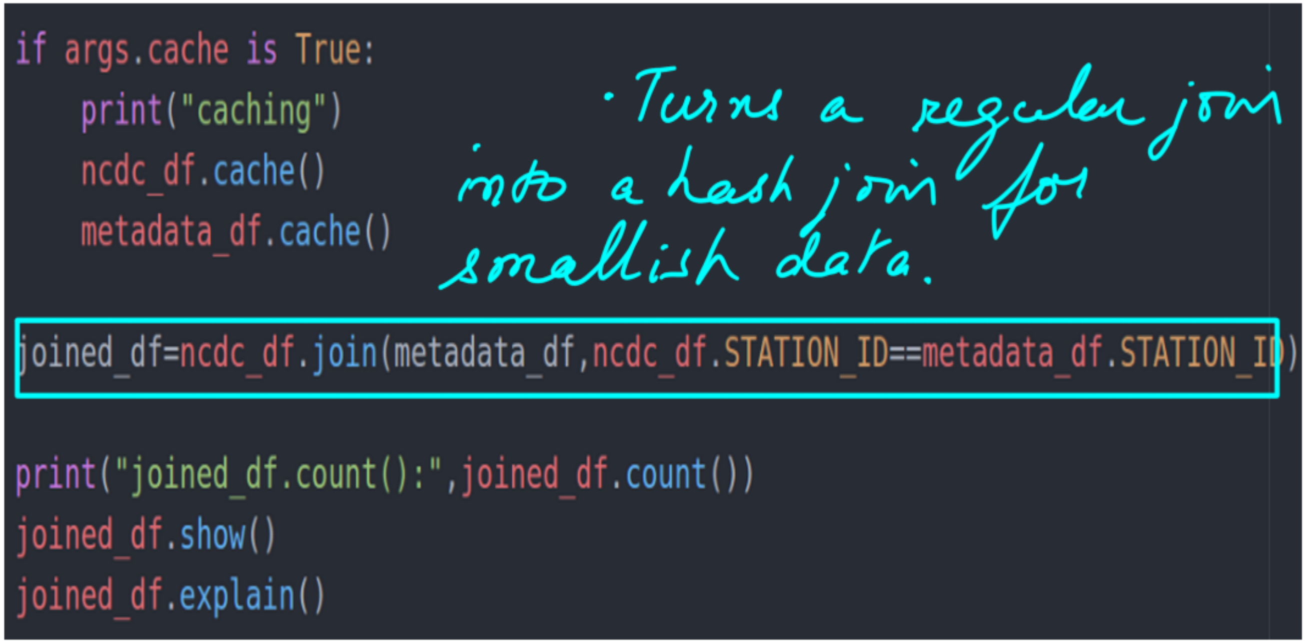
Optimizing Joins-2

By Mohit Kumar

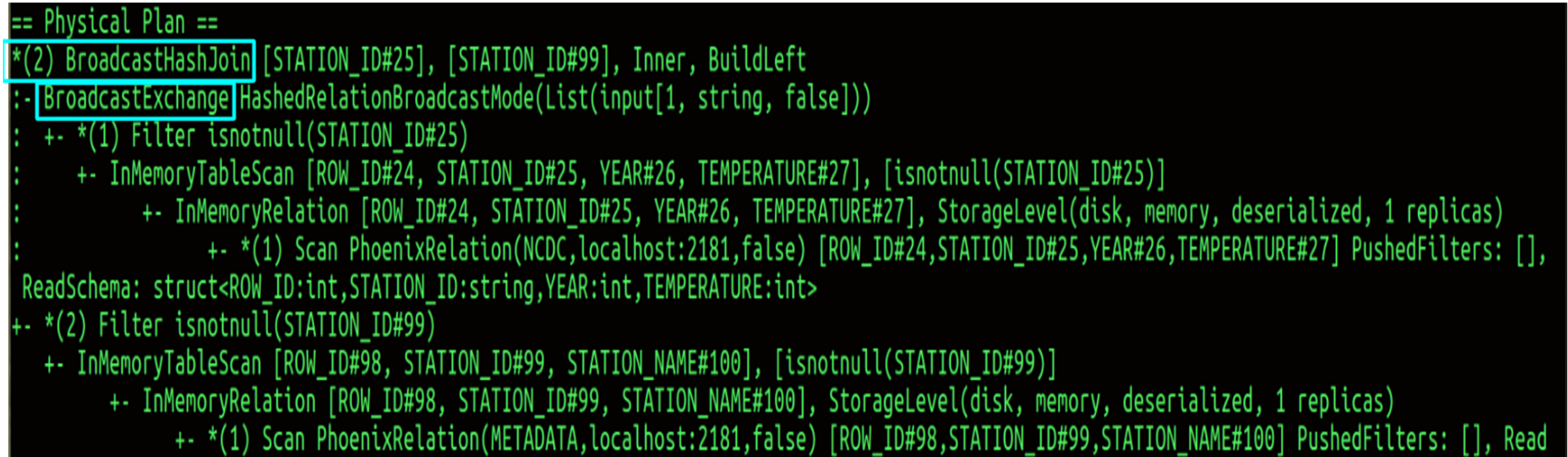
Dataframes:Optimizing Joins:join

- Regular Join with smallish data:



```
if args.cache is True:
    print("caching")
    ncdc_df.cache()
    metadata_df.cache()

joined_df=ncdc_df.join(metadata_df,ncdc_df.STATION_ID==metadata_df.STATION_ID)

print("joined_df.count():",joined_df.count())
joined_df.show()
joined_df.explain()
```

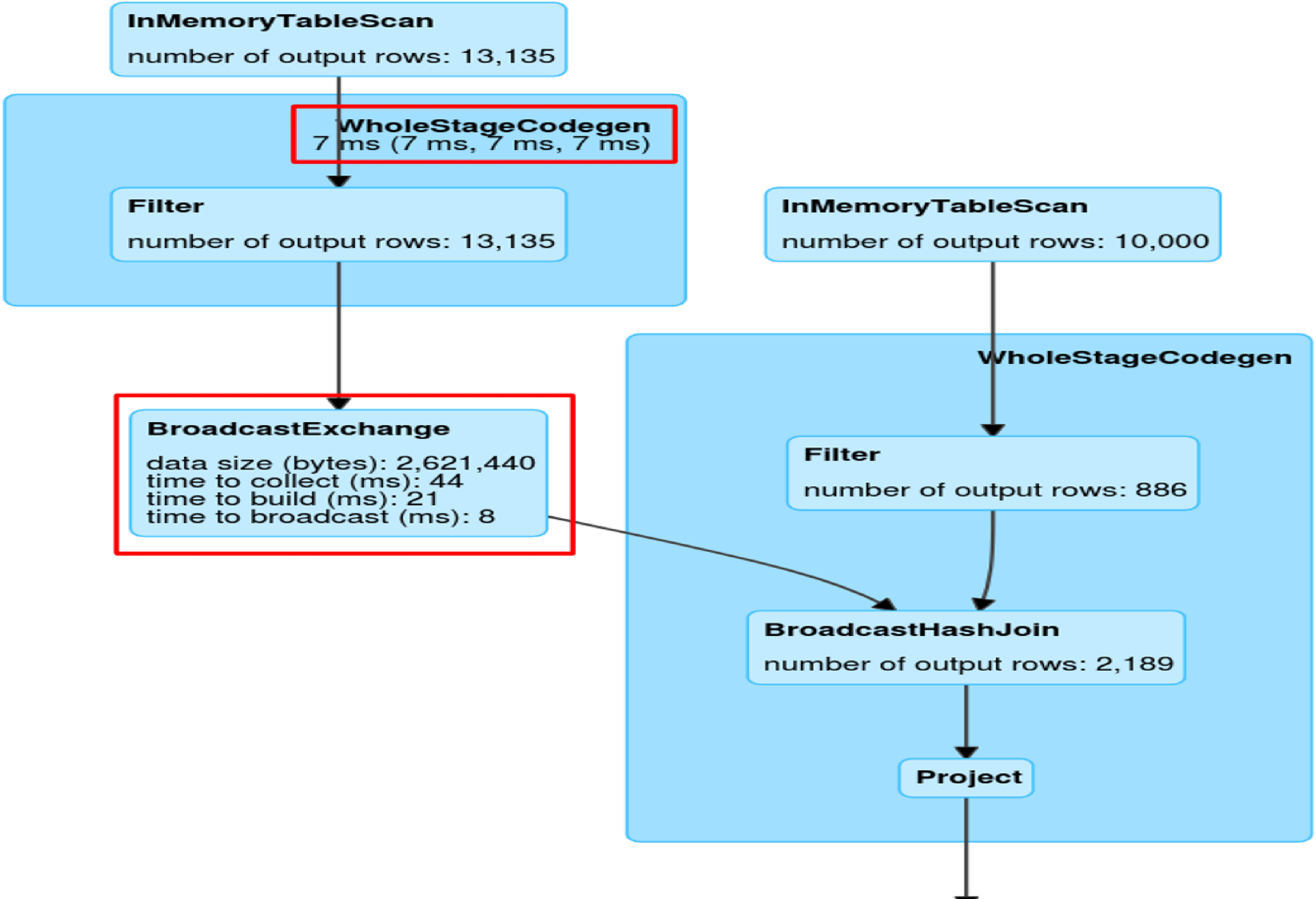
· Turns a regular join into a hash join for smallish data.



```
== Physical Plan ==
*(2) BroadcastHashJoin [STATION_ID#25], [STATION_ID#99], Inner, BuildLeft
:- BroadcastExchange HashedRelationBroadcastMode(List(input[1, string, false]))
:  +- *(1) Filter isnotnull(STATION_ID#25)
:     +- InMemoryTableScan [ROW_ID#24, STATION_ID#25, YEAR#26, TEMPERATURE#27], [isnotnull(STATION_ID#25)]
:           +- InMemoryRelation [ROW_ID#24, STATION_ID#25, YEAR#26, TEMPERATURE#27], StorageLevel(disk, memory, deserialized, 1 replicas)
:                 +- *(1) Scan PhoenixRelation(NCDC,localhost:2181,false) [ROW_ID#24,STATION_ID#25,YEAR#26,TEMPERATURE#27] PushedFilters: [],
 ReadSchema: struct<ROW_ID:int,STATION_ID:string,YEAR:int,TEMPERATURE:int>
+- *(2) Filter isnotnull(STATION_ID#99)
   +- InMemoryTableScan [ROW_ID#98, STATION_ID#99, STATION_NAME#100], [isnotnull(STATION_ID#99)]
         +- InMemoryRelation [ROW_ID#98, STATION_ID#99, STATION_NAME#100], StorageLevel(disk, memory, deserialized, 1 replicas)
               +- *(1) Scan PhoenixRelation(METADATA,localhost:2181,false) [ROW_ID#98,STATION_ID#99,STATION_NAME#100] PushedFilters: [], Read
```

Dataframes:Optimizing Joins:join

- Regular Join with smallish data:



InMemoryTableScan
number of output rows: 13,135

WholeStageCodegen
7 ms (7 ms, 7 ms, 7 ms)

Filter
number of output rows: 13,135

BroadcastExchange
data size (bytes): 2,621,440
time to collect (ms): 44
time to build (ms): 21
time to broadcast (ms): 8

InMemoryTableScan
number of output rows: 10,000

WholeStageCodegen

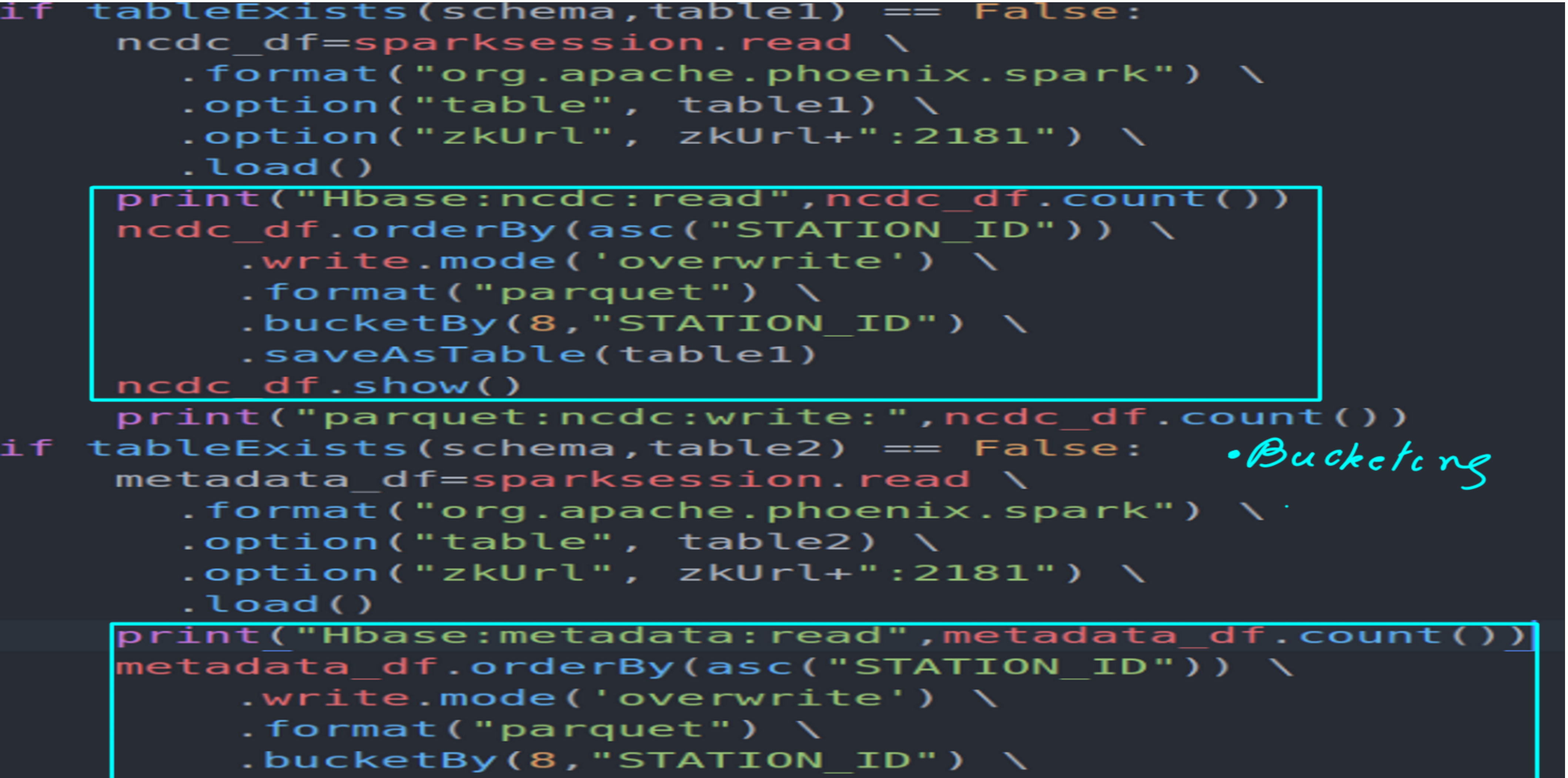
Filter
number of output rows: 886

BroadcastHashJoin
number of output rows: 2,189

Project

Dataframes:Optimizing Joins:bucketed SortMerge join

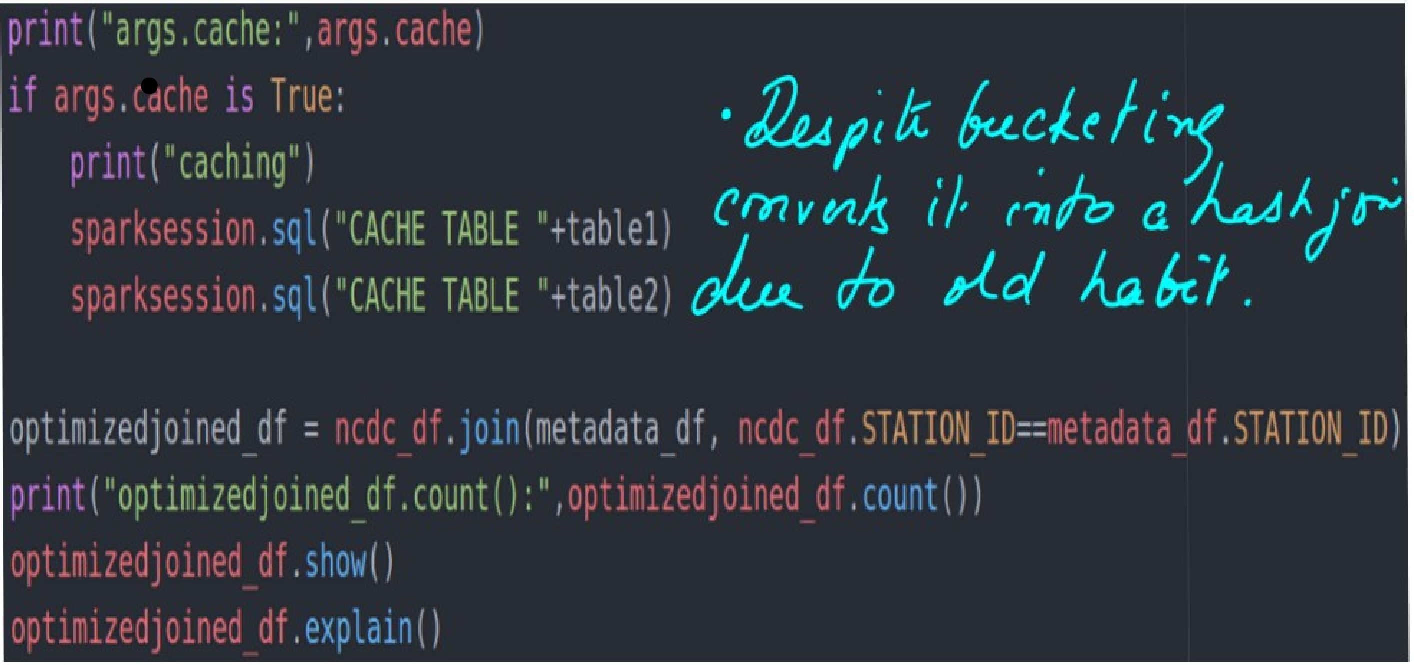
- Regular Join with smallish data:
  - we sort and bucket by the users_id and uid columns on which we’ll join, and save the buckets as Spark managed tables in Parquet format:



```
if tableExists(schema,table1) == False:
    ncdc_df=sparksession.read \
      .format("org.apache.phoenix.spark") \
      .option("table", table1) \
      .option("zkUrl", zkUrl+":2181") \
      .load()
    print("Hbase:ncdc:read",ncdc_df.count())
    ncdc_df.orderBy(asc("STATION_ID")) \
        .write.mode('overwrite') \
        .format("parquet") \
        .bucketBy(8,"STATION_ID") \
        .saveAsTable(table1)
    ncdc_df.show()
    print("parquet:ncdc:write:",ncdc_df.count())
if tableExists(schema,table2) == False:
    metadata_df=sparksession.read \
      .format("org.apache.phoenix.spark") \
      .option("table", table2) \
      .option("zkUrl", zkUrl+":2181") \
      .load()
    print("Hbase:metadata:read",metadata_df.count())
    metadata_df.orderBy(asc("STATION_ID")) \
        .write.mode('overwrite') \
        .format("parquet") \
        .bucketBy(8,"STATION_ID") \
```

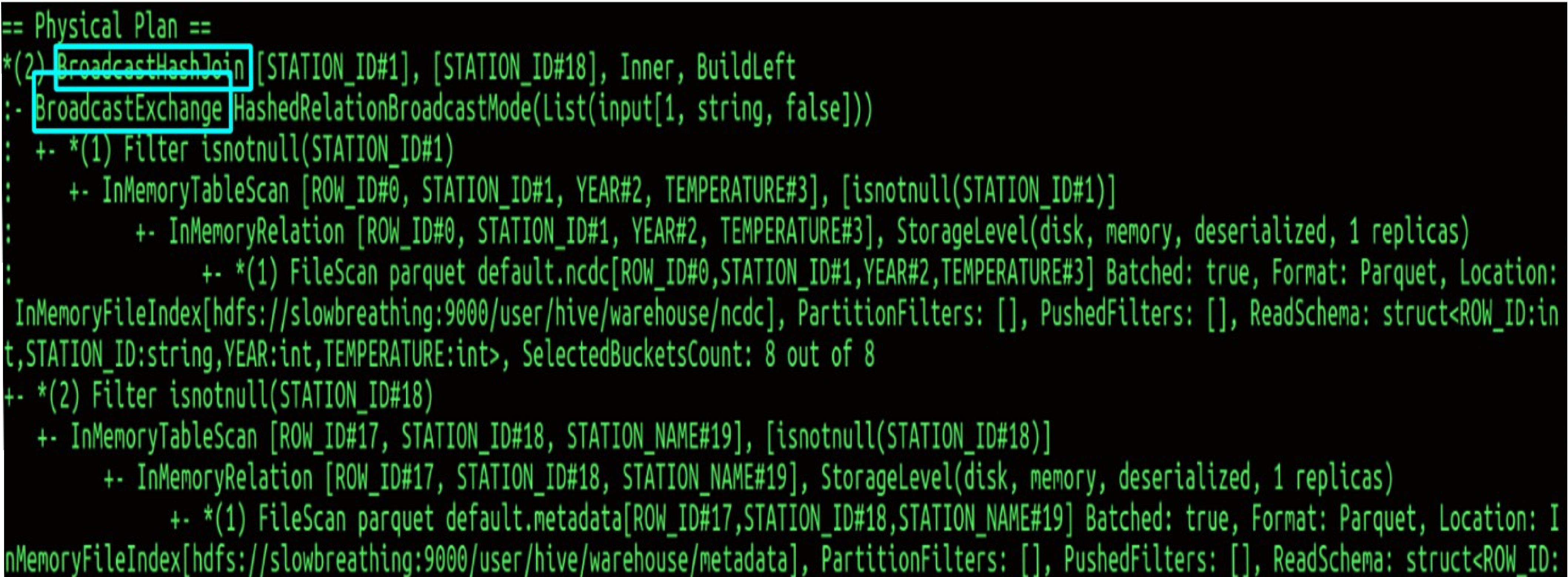
•Bucketing

Dataframes:Optimizing Joins:bucketed SortMerge join



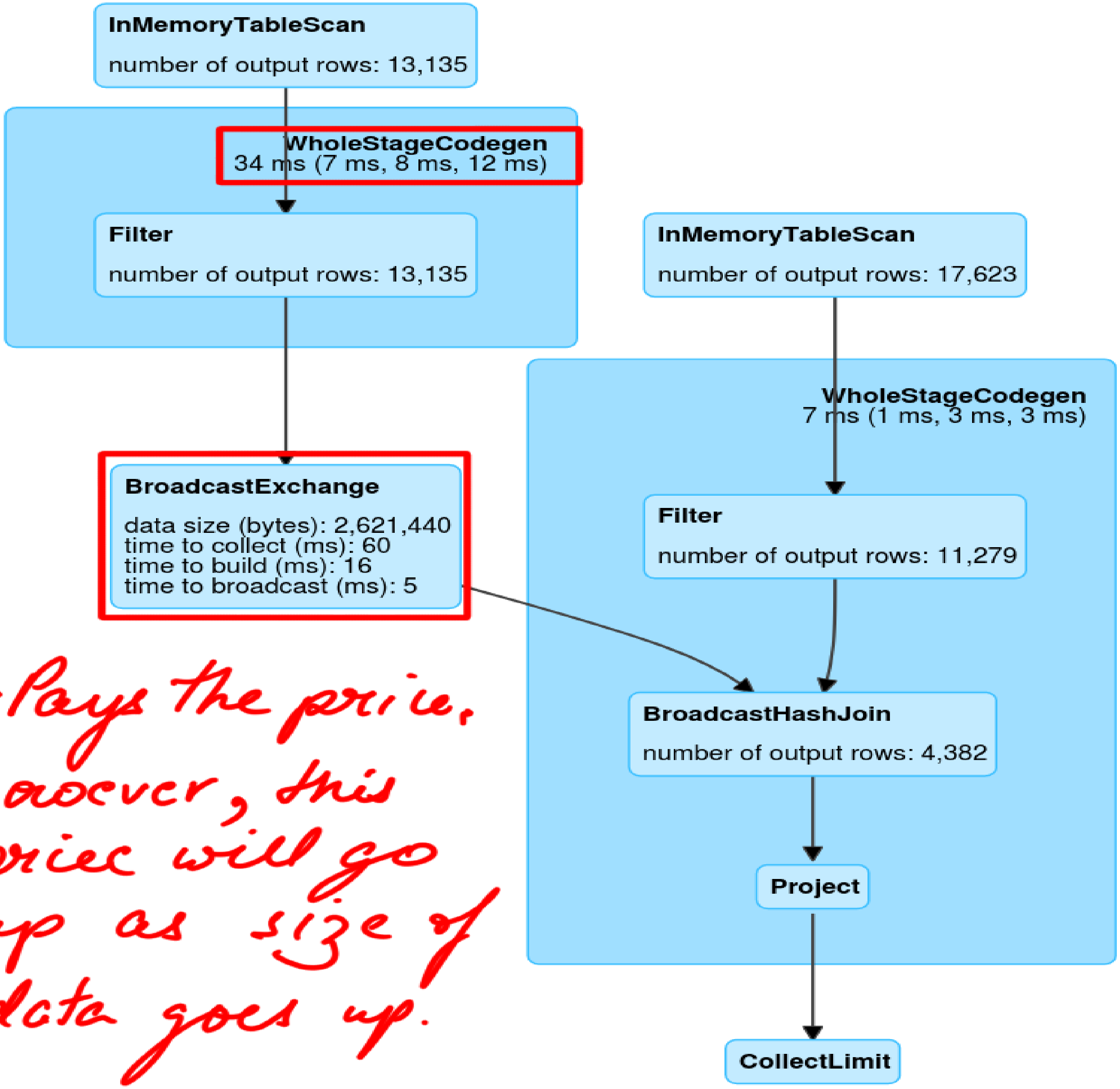
```python
print("args.cache:",args.cache)
if args.cache is True:
    print("caching")
    sparksession.sql("CACHE TABLE "+table1)
    sparksession.sql("CACHE TABLE "+table2)

optimizedjoined_df = ncdc_df.join(metadata_df, ncdc_df.STATION_ID==metadata_df.STATION_ID)
print("optimizedjoined_df.count():",optimizedjoined_df.count())
optimizedjoined_df.show()
optimizedjoined_df.explain()
```

• Despite bucketing converts it into a hash join due to old habit.



```
== Physical Plan ==
*(2) BroadcastHashJoin [STATION_ID#1], [STATION_ID#18], Inner, BuildLeft
:- BroadcastExchange HashedRelationBroadcastMode(List(input[1, string, false]))
:  +- *(1) Filter isnotnull(STATION_ID#1)
:     +- InMemoryTableScan [ROW_ID#0, STATION_ID#1, YEAR#2, TEMPERATURE#3], [isnotnull(STATION_ID#1)]
:           +- InMemoryRelation [ROW_ID#0, STATION_ID#1, YEAR#2, TEMPERATURE#3], StorageLevel(disk, memory, deserialized, 1 replicas)
:                 +- *(1) FileScan parquet default.ncdc[ROW_ID#0,STATION_ID#1,YEAR#2,TEMPERATURE#3] Batched: true, Format: Parquet, Location:
 InMemoryFileIndex[hdfs://slowbreathing:9000/user/hive/warehouse/ncdc], PartitionFilters: [], PushedFilters: [], ReadSchema: struct<ROW_ID:in
t,STATION_ID:string,YEAR:int,TEMPERATURE:int>, SelectedBucketsCount: 8 out of 8
+- *(2) Filter isnotnull(STATION_ID#18)
   +- InMemoryTableScan [ROW_ID#17, STATION_ID#18, STATION_NAME#19], [isnotnull(STATION_ID#18)]
         +- InMemoryRelation [ROW_ID#17, STATION_ID#18, STATION_NAME#19], StorageLevel(disk, memory, deserialized, 1 replicas)
               +- *(1) FileScan parquet default.metadata[ROW_ID#17,STATION_ID#18,STATION_NAME#19] Batched: true, Format: Parquet, Location: I
nMemoryFileIndex[hdfs://slowbreathing:9000/user/hive/warehouse/metadata], PartitionFilters: [], PushedFilters: [], ReadSchema: struct<ROW_ID:
```

Dataframes:Optimizing Joins:bucketed SortMerge join



InMemoryTableScan

number of output rows: 13,135

WholeStageCodegen
34 ms (7 ms, 8 ms, 12 ms)

Filter

number of output rows: 13,135

BroadcastExchange

data size (bytes): 2,621,440
time to collect (ms): 60
time to build (ms): 16
time to broadcast (ms): 5

InMemoryTableScan

number of output rows: 17,623

WholeStageCodegen
7 ms (1 ms, 3 ms, 3 ms)

Filter

number of output rows: 11,279

BroadcastHashJoin

number of output rows: 4,382

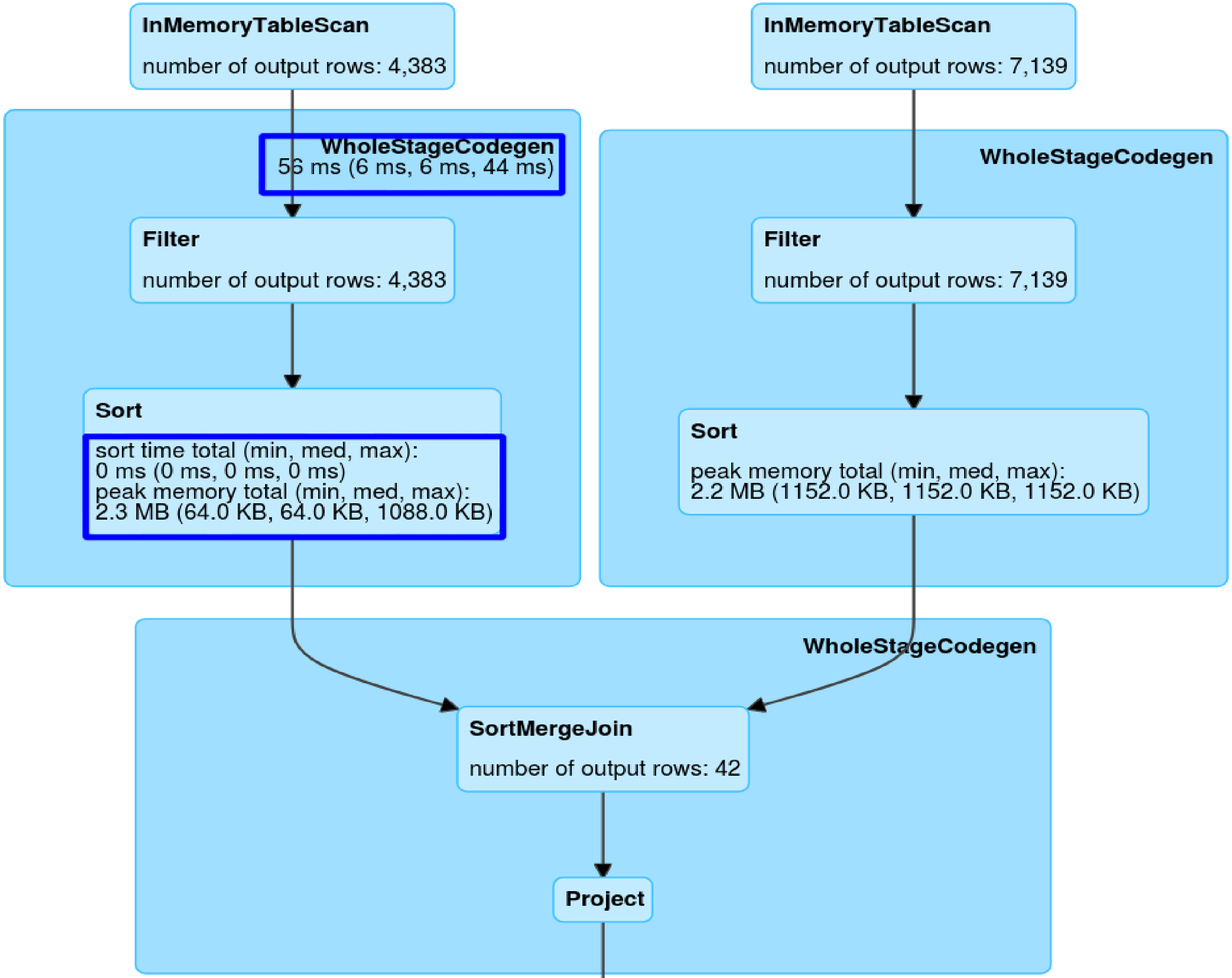
Project

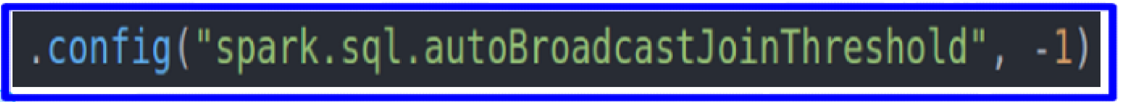
CollectLimit

·Pays the price, however, this price will go up as size of data goes up.

Dataframes:Optimizing Joins:bucketed SortMerge join:disablebc

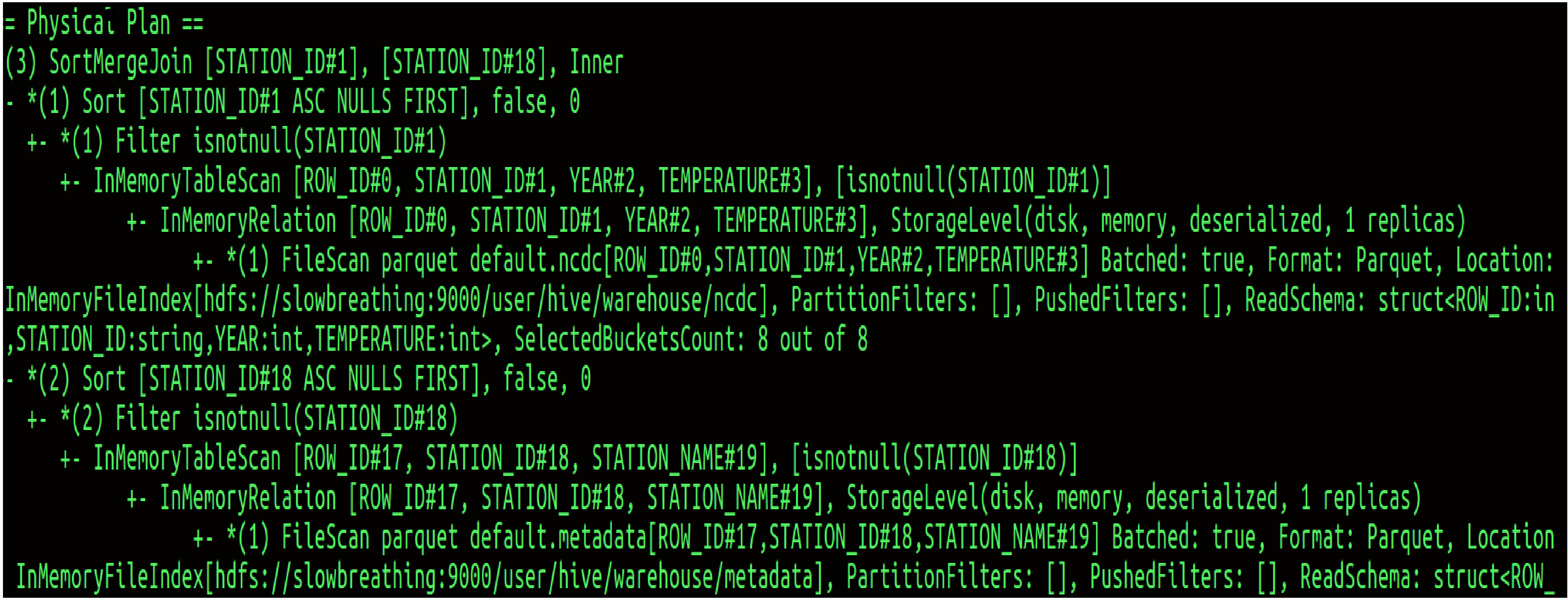
-


InMemoryTableScan
number of output rows: 4,383
InMemoryTableScan
number of output rows: 7,139
WholeStageCodegen
56 ms (6 ms, 6 ms, 44 ms)
WholeStageCodegen
Filter
number of output rows: 4,383
Filter
number of output rows: 7,139
Sort
sort time total (min, med, max):
0 ms (0 ms, 0 ms, 0 ms)
peak memory total (min, med, max):
2.3 MB (64.0 KB, 64.0 KB, 1088.0 KB)
Sort
peak memory total (min, med, max):
2.2 MB (1152.0 KB, 1152.0 KB, 1152.0 KB)
WholeStageCodegen
SortMergeJoin
number of output rows: 42
Project
CollectLimit



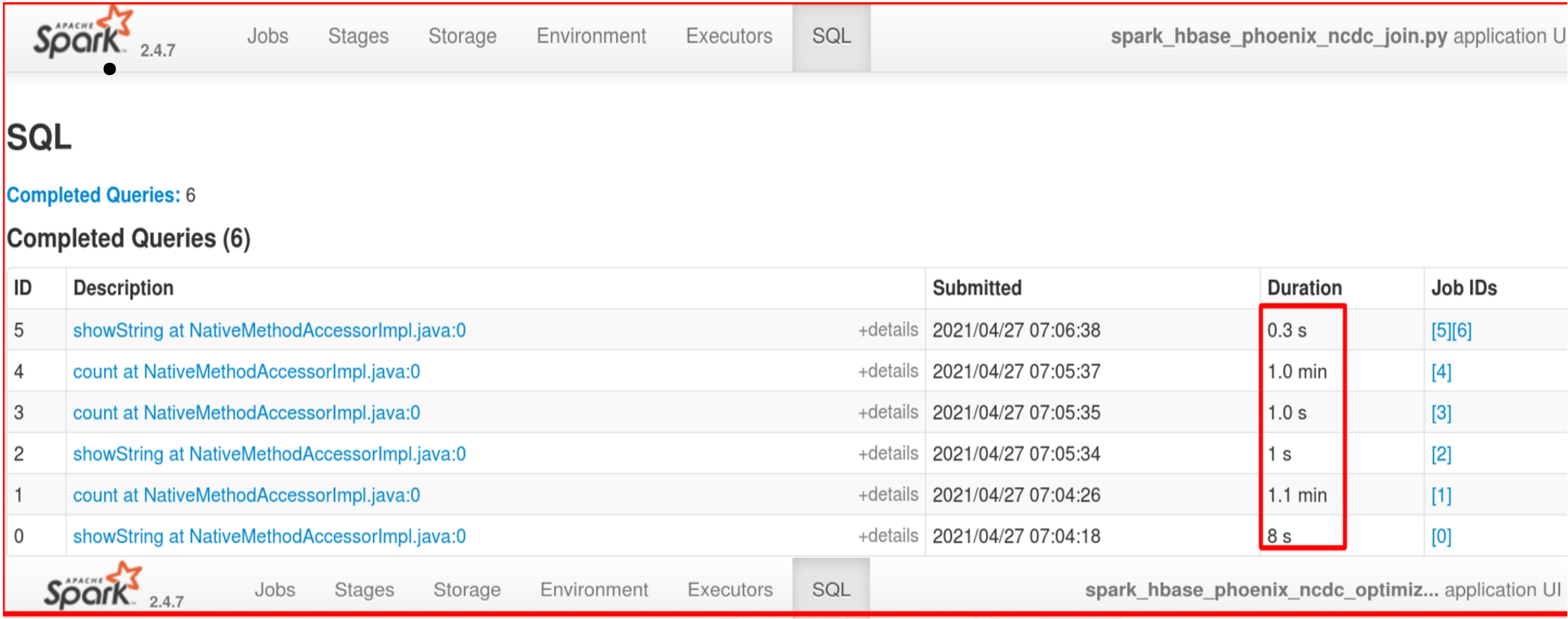
```
.config("spark.sql.autoBroadcastJoinThreshold", -1)
```

Dataframes:Optimizing Joins:bucketed SortMerge join:disablebc



```
= Physical Plan ==
(3) SortMergeJoin [STATION_ID#1], [STATION_ID#18], Inner
- *(1) Sort [STATION_ID#1 ASC NULLS FIRST], false, 0
  +- *(1) Filter isnotnull(STATION_ID#1)
     +- InMemoryTableScan [ROW_ID#0, STATION_ID#1, YEAR#2, TEMPERATURE#3], [isnotnull(STATION_ID#1)]
           +- InMemoryRelation [ROW_ID#0, STATION_ID#1, YEAR#2, TEMPERATURE#3], StorageLevel(disk, memory, deserialized, 1 replicas)
                 +- *(1) FileScan parquet default.ncdc[ROW_ID#0,STATION_ID#1,YEAR#2,TEMPERATURE#3] Batched: true, Format: Parquet, Location:
InMemoryFileIndex[hdfs://slowbreathing:9000/user/hive/warehouse/ncdc], PartitionFilters: [], PushedFilters: [], ReadSchema: struct<ROW_ID:in
,STATION_ID:string,YEAR:int,TEMPERATURE:int>, SelectedBucketsCount: 8 out of 8
- *(2) Sort [STATION_ID#18 ASC NULLS FIRST], false, 0
  +- *(2) Filter isnotnull(STATION_ID#18)
     +- InMemoryTableScan [ROW_ID#17, STATION_ID#18, STATION_NAME#19], [isnotnull(STATION_ID#18)]
           +- InMemoryRelation [ROW_ID#17, STATION_ID#18, STATION_NAME#19], StorageLevel(disk, memory, deserialized, 1 replicas)
                 +- *(1) FileScan parquet default.metadata[ROW_ID#17,STATION_ID#18,STATION_NAME#19] Batched: true, Format: Parquet, Location
 InMemoryFileIndex[hdfs://slowbreathing:9000/user/hive/warehouse/metadata], PartitionFilters: [], PushedFilters: [], ReadSchema: struct<ROW_
```

Dataframes:Optimizing Joins:comparision

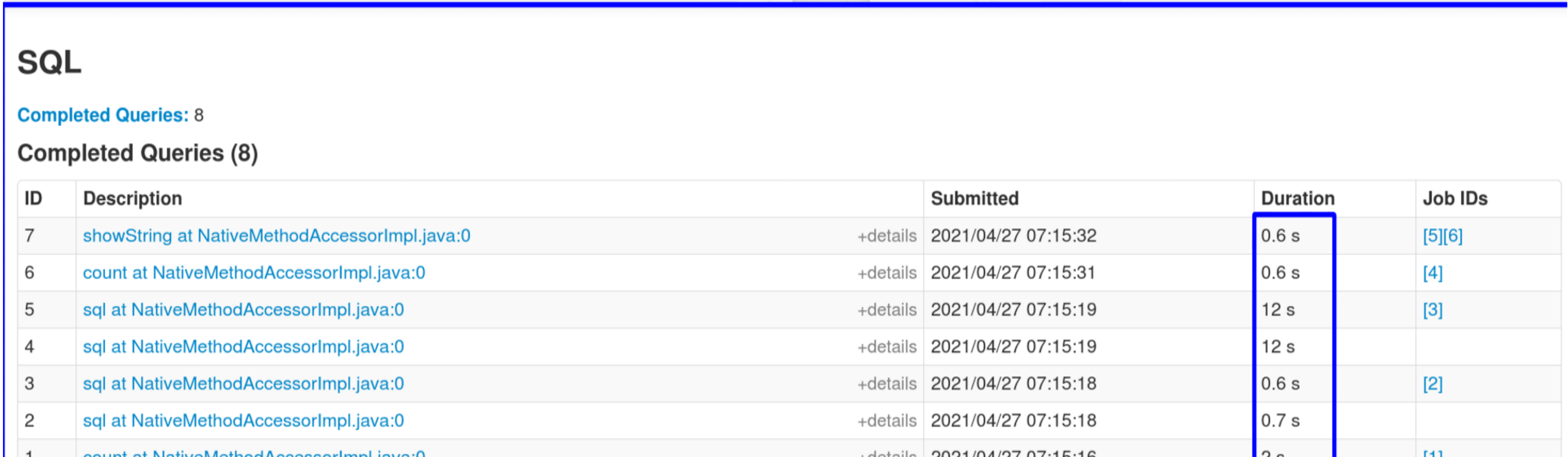


SQL

Completed Queries: 6

Completed Queries (6)

ID	Description		Submitted	Duration	Job IDs
5	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:06:38	0.3 s	[5][6]
4	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:05:37	1.0 min	[4]
3	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:05:35	1.0 s	[3]
2	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:05:34	1 s	[2]
1	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:04:26	1.1 min	[1]
0	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:04:18	8 s	[0]



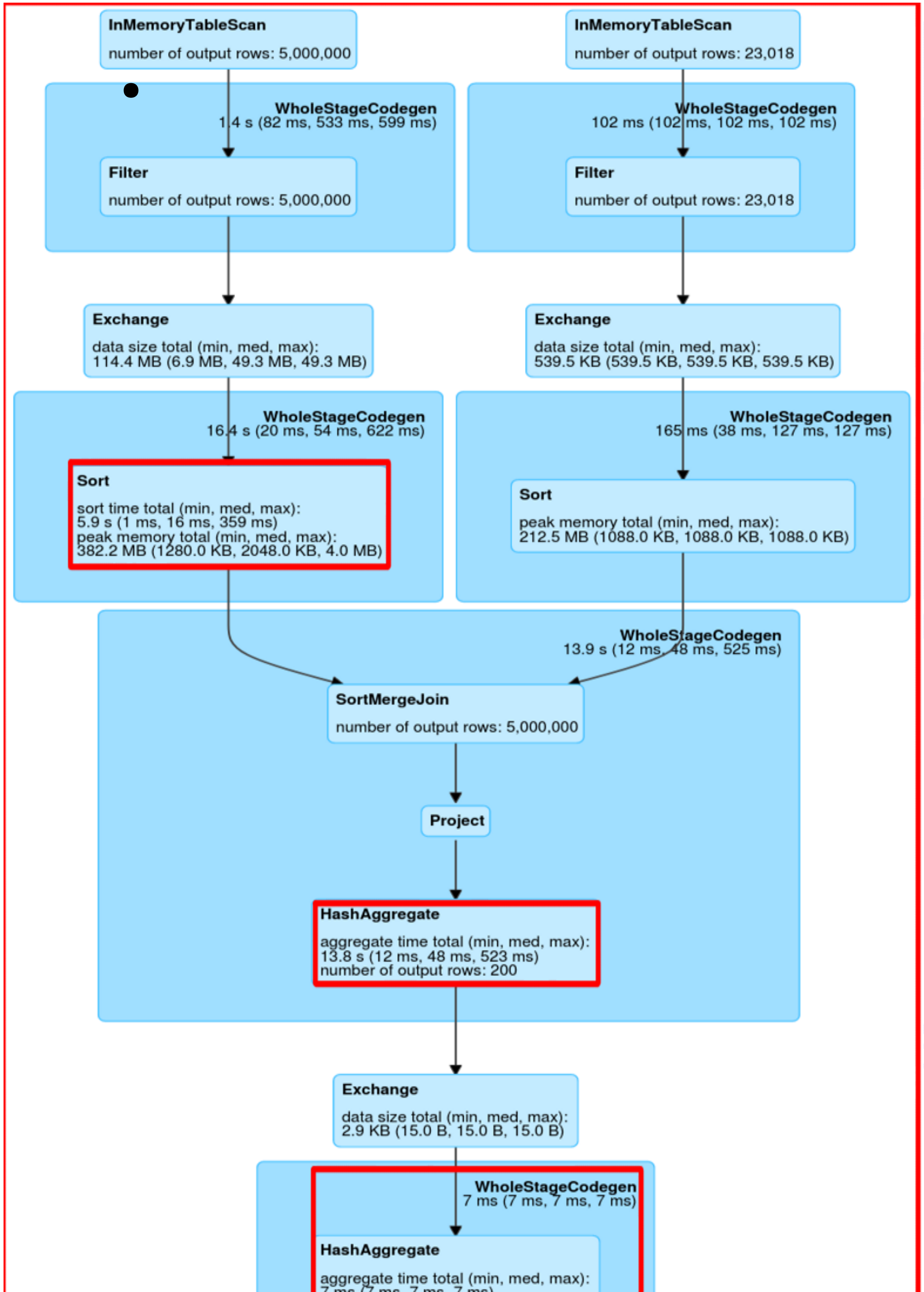
SQL

Completed Queries: 8

Completed Queries (8)

ID	Description		Submitted	Duration	Job IDs
7	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:32	0.6 s	[5][6]
6	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:31	0.6 s	[4]
5	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:19	12 s	[3]
4	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:19	12 s	
3	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:18	0.6 s	[2]
2	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:18	0.7 s	
1	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 07:15:16	2 s	[1]

Dataframes:Optimizing Joins:comparision



InMemoryTableScan
number of output rows: 5,000,000

WholeStageCodegen
1.4 s (82 ms, 533 ms, 599 ms)

Filter
number of output rows: 5,000,000

Exchange
data size total (min, med, max):
114.4 MB (6.9 MB, 49.3 MB, 49.3 MB)

WholeStageCodegen
16.4 s (20 ms, 54 ms, 622 ms)

Sort
sort time total (min, med, max):
5.9 s (1 ms, 16 ms, 359 ms)
peak memory total (min, med, max):
382.2 MB (1280.0 KB, 2048.0 KB, 4.0 MB)

InMemoryTableScan
number of output rows: 23,018

WholeStageCodegen
102 ms (102 ms, 102 ms, 102 ms)

Filter
number of output rows: 23,018

Exchange
data size total (min, med, max):
539.5 KB (539.5 KB, 539.5 KB, 539.5 KB)

WholeStageCodegen
165 ms (38 ms, 127 ms, 127 ms)

Sort
peak memory total (min, med, max):
212.5 MB (1088.0 KB, 1088.0 KB, 1088.0 KB)

WholeStageCodegen
13.9 s (12 ms, 48 ms, 525 ms)

SortMergeJoin
number of output rows: 5,000,000

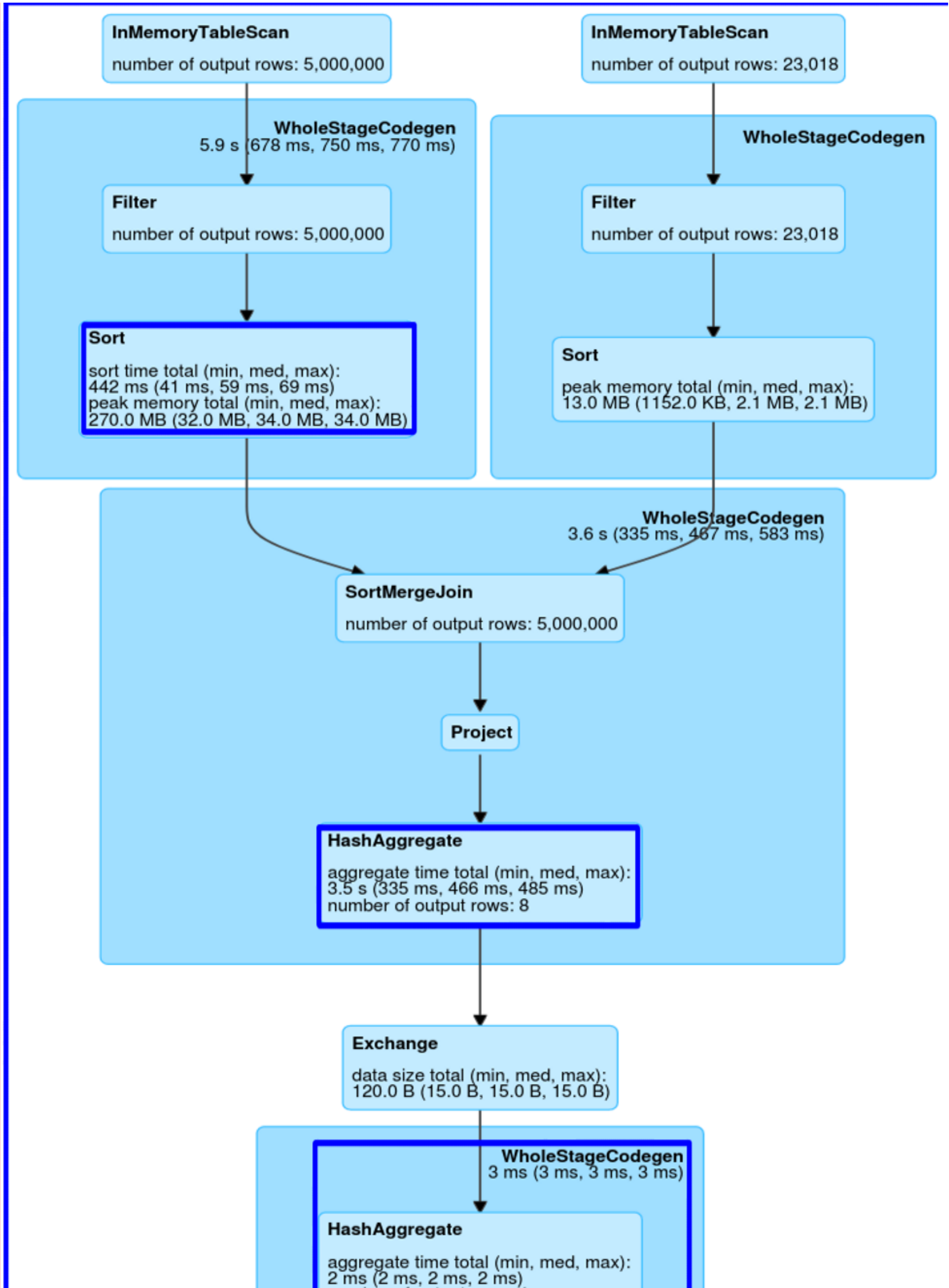
Project

HashAggregate
aggregate time total (min, med, max):
13.8 s (12 ms, 48 ms, 523 ms)
number of output rows: 200

Exchange
data size total (min, med, max):
2.9 KB (15.0 B, 15.0 B, 15.0 B)

WholeStageCodegen
7 ms (7 ms, 7 ms, 7 ms)

HashAggregate
aggregate time total (min, med, max):
7 ms (7 ms, 7 ms, 7 ms)



InMemoryTableScan
number of output rows: 5,000,000

WholeStageCodegen
5.9 s (678 ms, 750 ms, 770 ms)

Filter
number of output rows: 5,000,000

Sort
sort time total (min, med, max):
442 ms (41 ms, 59 ms, 69 ms)
peak memory total (min, med, max):
270.0 MB (32.0 MB, 34.0 MB, 34.0 MB)

InMemoryTableScan
number of output rows: 23,018

WholeStageCodegen

Filter
number of output rows: 23,018

Sort
peak memory total (min, med, max):
13.0 MB (1152.0 KB, 2.1 MB, 2.1 MB)

WholeStageCodegen
3.6 s (335 ms, 467 ms, 583 ms)

SortMergeJoin
number of output rows: 5,000,000

Project

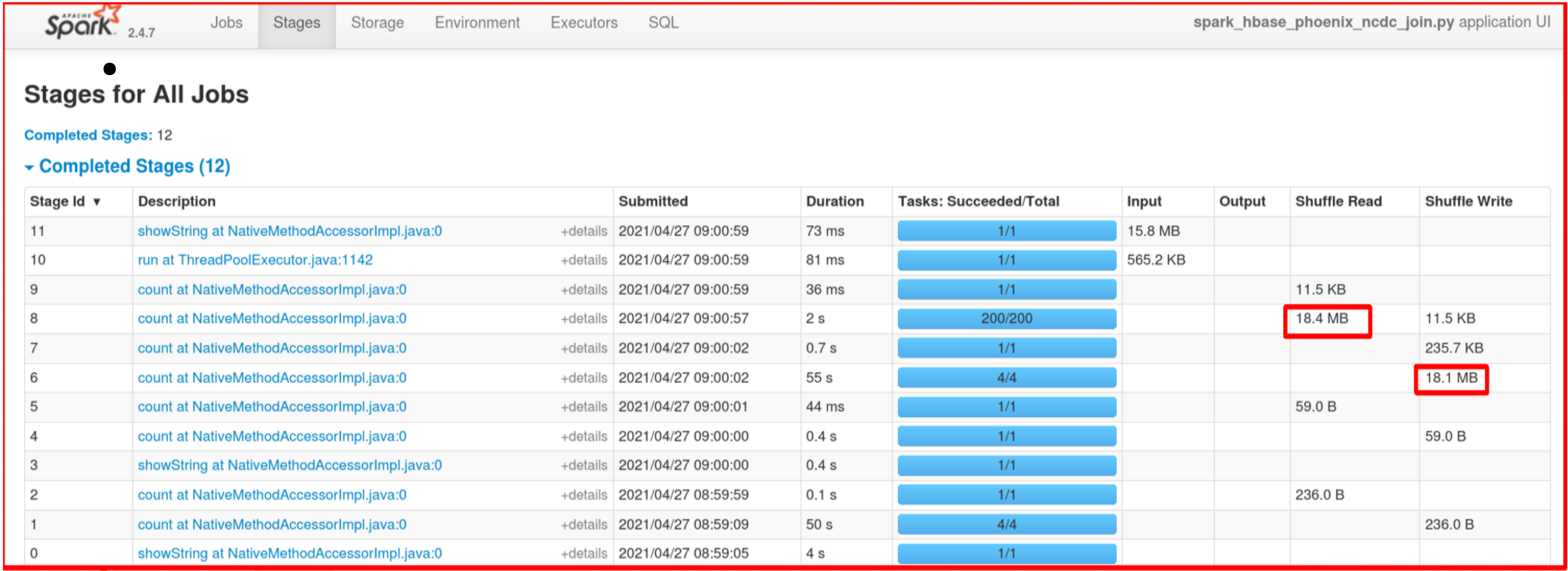
HashAggregate
aggregate time total (min, med, max):
3.5 s (335 ms, 466 ms, 485 ms)
number of output rows: 8

Exchange
data size total (min, med, max):
120.0 B (15.0 B, 15.0 B, 15.0 B)

WholeStageCodegen
3 ms (3 ms, 3 ms, 3 ms)

HashAggregate
aggregate time total (min, med, max):
2 ms (2 ms, 2 ms, 2 ms)

Dataframes:Optimizing Joins:comparision

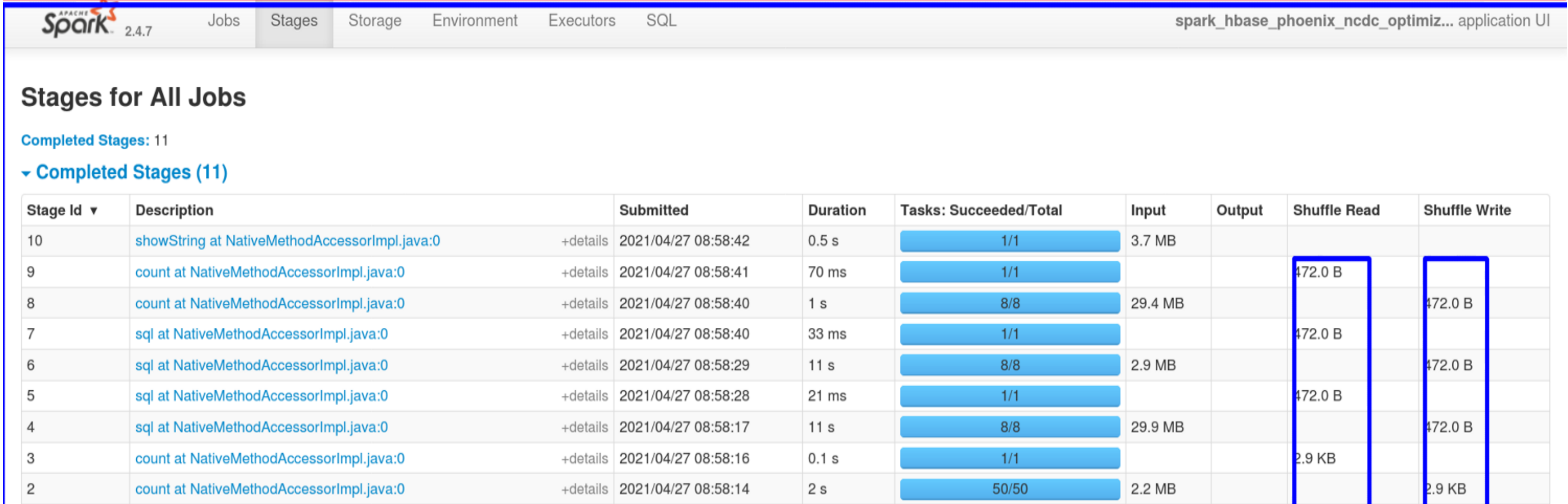


Stages for All Jobs

Completed Stages: 12

Completed Stages (12)

Stage Id ▾	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
11	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:59	73 ms	1/1	15.8 MB			
10	run at ThreadPoolExecutor.java:1142	+details	2021/04/27 09:00:59	81 ms	1/1	565.2 KB			
9	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:59	36 ms	1/1			11.5 KB	
8	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:57	2 s	200/200			18.4 MB	11.5 KB
7	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:02	0.7 s	1/1				235.7 KB
6	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:02	55 s	4/4				18.1 MB
5	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:01	44 ms	1/1			59.0 B	
4	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:00	0.4 s	1/1				59.0 B
3	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 09:00:00	0.4 s	1/1				
2	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:59:59	0.1 s	1/1			236.0 B	
1	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:59:09	50 s	4/4				236.0 B
0	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:59:05	4 s	1/1				



Stages for All Jobs

Completed Stages: 11

Completed Stages (11)

Stage Id ▾	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
10	showString at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:42	0.5 s	1/1	3.7 MB			
9	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:41	70 ms	1/1			472.0 B	
8	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:40	1 s	8/8	29.4 MB			472.0 B
7	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:40	33 ms	1/1			472.0 B	
6	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:29	11 s	8/8	2.9 MB			472.0 B
5	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:28	21 ms	1/1			472.0 B	
4	sql at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:17	11 s	8/8	29.9 MB			472.0 B
3	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:16	0.1 s	1/1			2.9 KB	
2	count at NativeMethodAccessorImpl.java:0	+details	2021/04/27 08:58:14	2 s	50/50	2.2 MB			2.9 KB