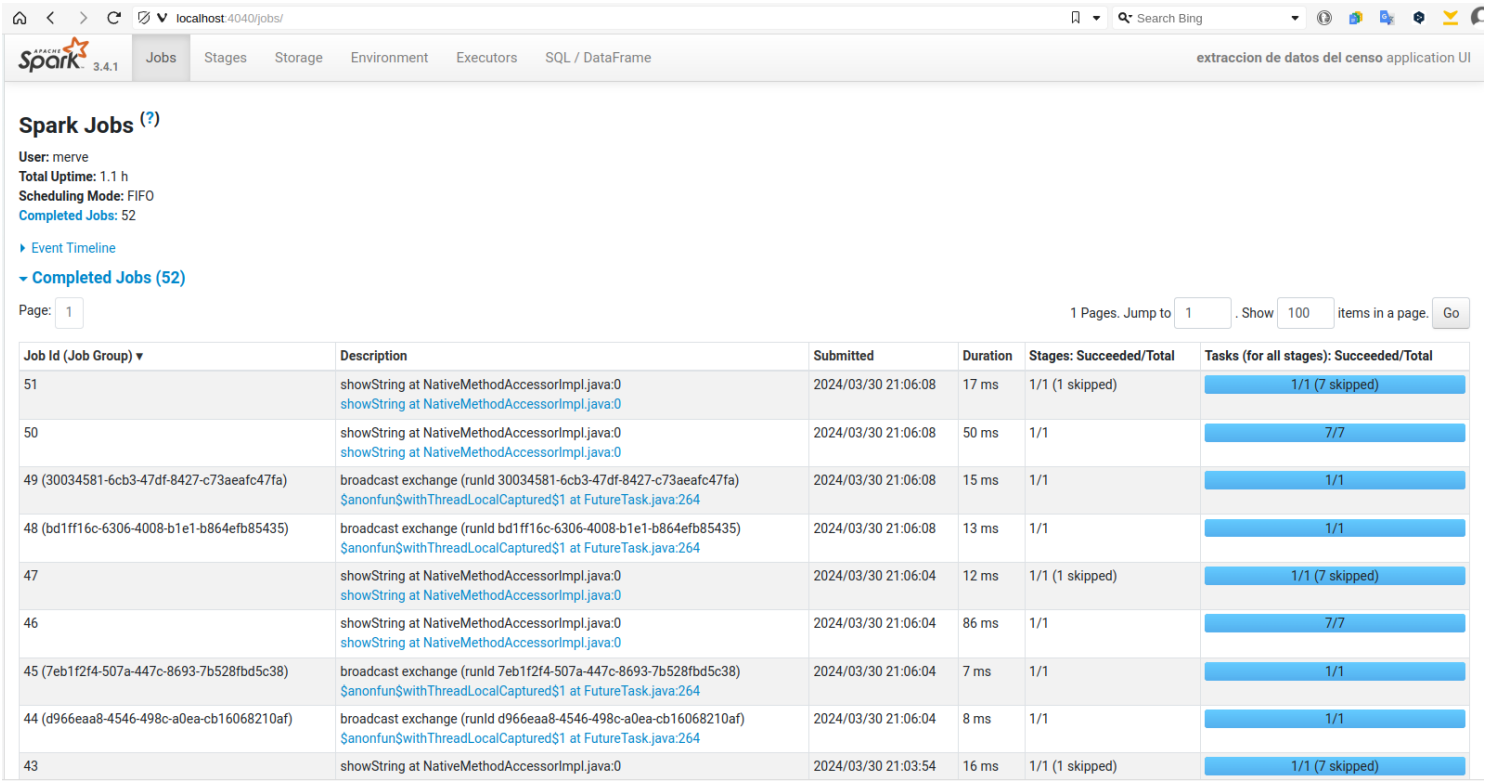
1. Los resultados de la operación (la lista ordenada de 10 provincias)

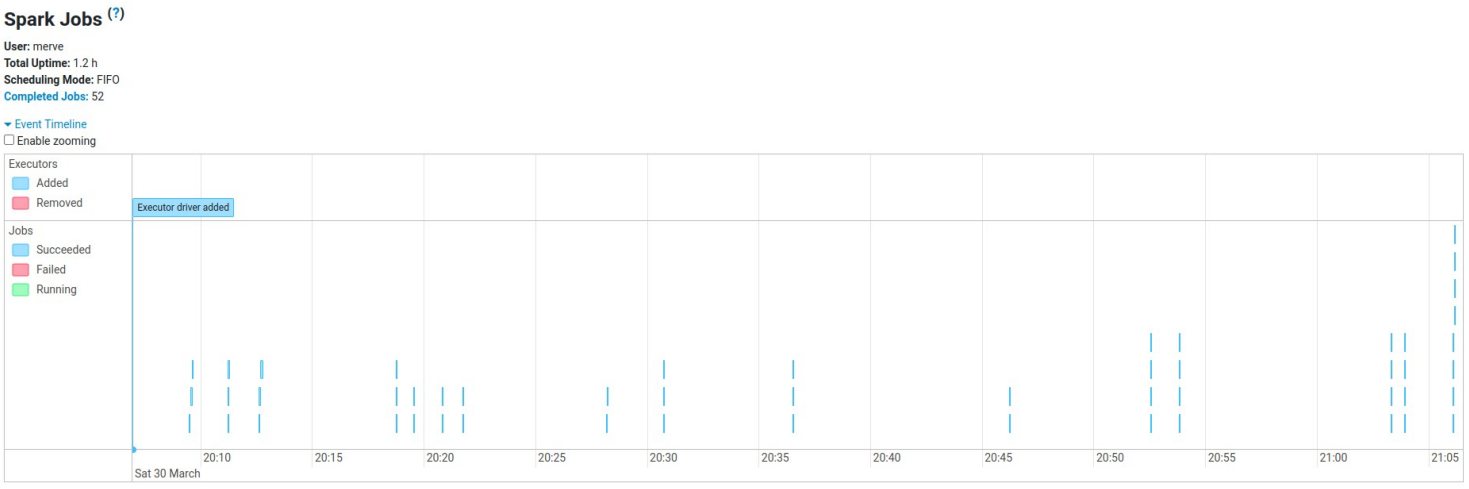
Número de personas en cada provincia con el nombre de la provincia:

Código de provincia (CPRO)	Nombre de Provincias (NOMBRE_PRO)	Número de personas (count)
28	Madrid	201
08	Barcelona	197
46	Valencia	181
03	Alicante	93
30	Murcia	79
17	Girona	79
32	Orense	70
36	Pontevedra	68
31	Navarra	67
24	León	62

2. Una descripción de los elementos identificados en la interfaz de monitorización de Spark, incluyendo las imágenes capturadas y su explicación.

- Capture algunas imágenes de flujo de datos, las lineas de tiempos y diagramas de tareas, y explique qué es lo que representan.



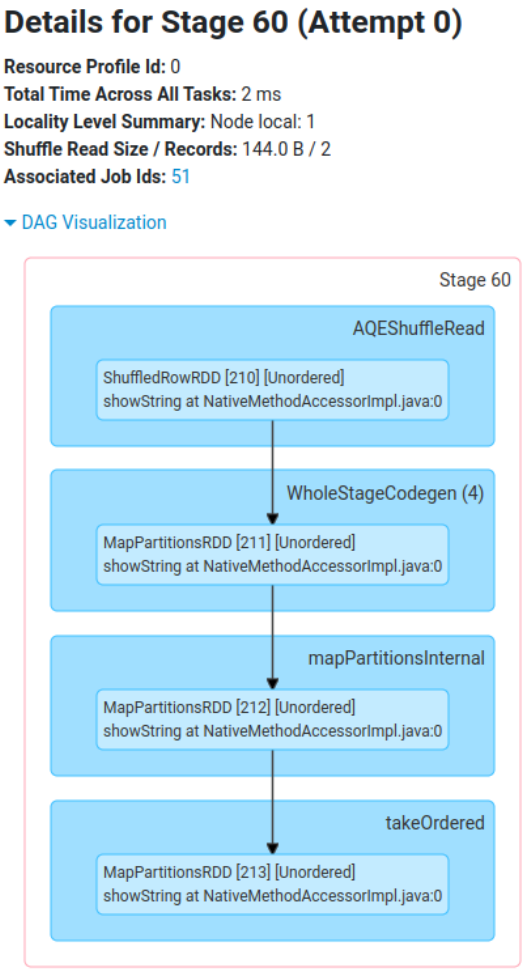


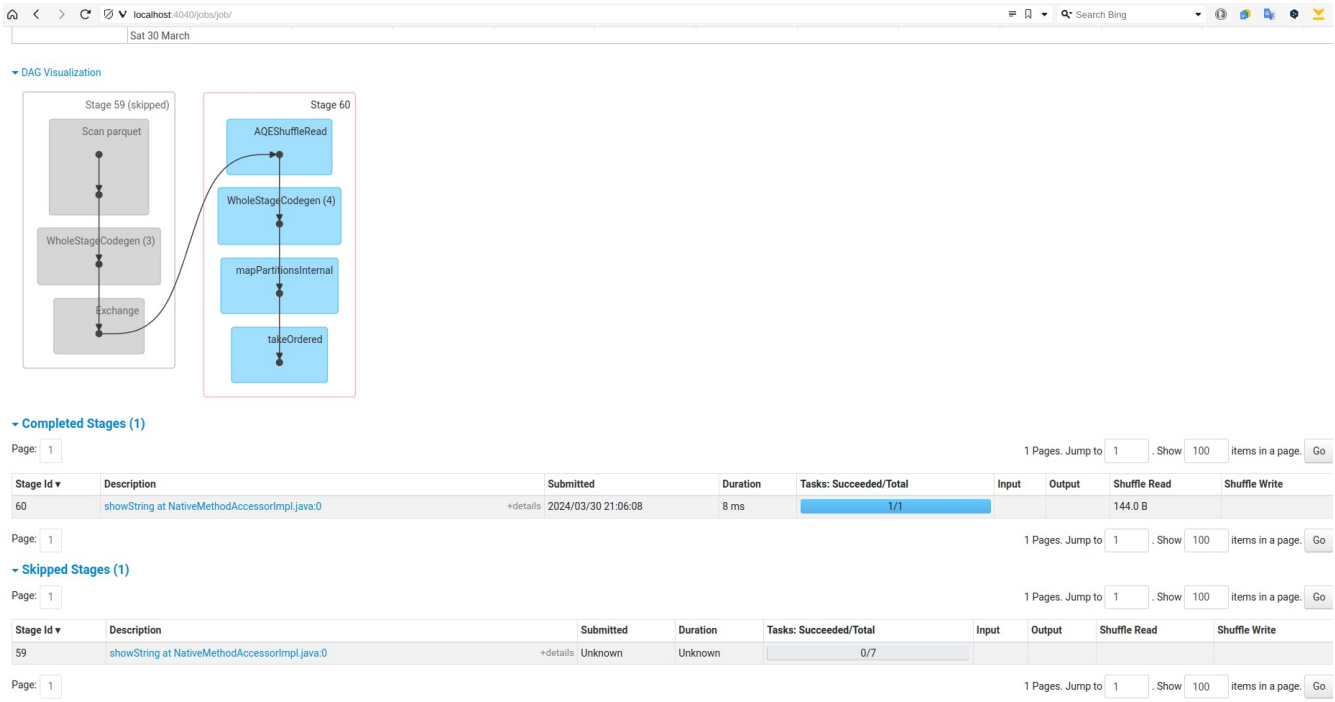
Spark Jobs (Trabajos de Spark): Este texto indica que es la sección de trabajos de Spark en la interfaz de usuario de Spark.

Completed Jobs (Trabajos completados) (52): Esta sección muestra los trabajos completados, y el número (52) indica que hay 52 trabajos completados mostrados en esta página.

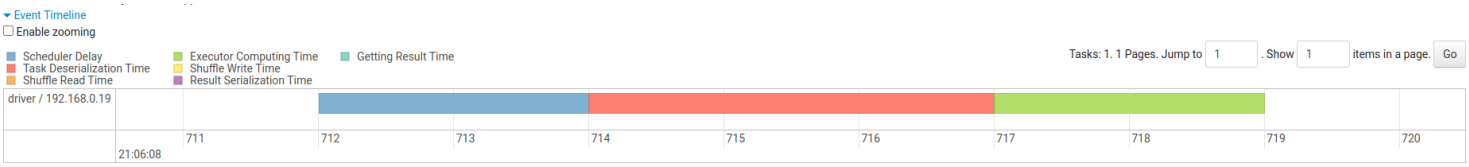
Event Timeline : Esta sección proporciona una línea de tiempo de los eventos relacionados con la ejecución de la aplicación Spark. La línea de tiempo muestra que la ejecución se realizó con éxito entre las 8 y las 9 de la tarde del 30 de marzo.

Job table : Esta tabla contiene detalles sobre los trabajos completados. Algunas columnas visibles:
Job Id: Esta columna muestre identificadores únicos para cada trabajo
Description: Esta columna puede contener descripciones breves de los trabajos.
Submitted: Esta columna muestra la hora de envío de cada trabajo.
Duration: Esta columna indica el tiempo de ejecución que tomó cada trabajo.
Stages: Succeeded/Total: Esta columna proporciona información sobre las etapas del trabajo. Aquí, muestra el número de etapas completadas con éxito (Succeeded) del número total de etapas (Total).
Tasks (for all stages): Succeeded/Total: Esta columna muestra las tareas exitosas (Succeeded) del número total de tareas (Total) ejecutadas en todas las etapas del trabajo.





Visualización DAG: Es una visualización de un gráfico acíclico dirigido (DAG). Representa el plan de ejecución del trabajo Spark, mostrando cómo están conectadas las diferentes etapas y tareas.
Nodes: Los círculos en el DAG representan etapas en la aplicación Spark. Las etapas son grupos de tareas relacionadas que pueden ejecutarse en paralelo.
Edges: Las flechas entre etapas representan dependencias de datos. Una flecha de la etapa A a la etapa B indica que la etapa B depende de los datos de salida de la etapa A. La etapa B no puede iniciar la ejecución hasta que la etapa A finalice con éxito.
Stage Labels: Algunas etapas tienen etiquetas que indican sus funciones.



Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	2.0 ms	2.0 ms	2.0 ms	2.0 ms	2.0 ms
GC Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Shuffle Read Size / Records	144 B / 2	144 B / 2	144 B / 2	144 B / 2	144 B / 2
Scheduler Delay	2.0 ms	2.0 ms	2.0 ms	2.0 ms	2.0 ms
Task Deserialization Time	3.0 ms	3.0 ms	3.0 ms	3.0 ms	3.0 ms
Shuffle Read Fetch Wait Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Shuffle Remote Reads	0.0 B	0.0 B	0.0 B	0.0 B	0.0 B
Result Serialization Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Getting Result Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Peak Execution Memory	2.2 MiB	2.2 MiB	2.2 MiB	2.2 MiB	2.2 MiB

Event Timeline: Esta sección proporciona una línea de tiempo de las diferentes etapas de la ejecución de la aplicación Spark. Incluye información sobre el retraso del planificador, el tiempo de computación del ejecutor, el tiempo de obtención de resultados, el tiempo de deserialización de la tarea, el tiempo de escritura de la mezcla, el tiempo de lectura de la mezcla y el tiempo de serialización de los resultados. En esta sección, nos muestra específicamente el retraso del planificador, el tiempo de deserialización de la tarea y el tiempo de computación del ejecutor.

Summary Metrics for 1 Completed Tasks: Esta sección proporciona estadísticas resumidas de las tareas completadas en la ejecución de la aplicación Spark. Incluye métricas como la duración, el tiempo de GC, el tamaño de lectura aleatoria/registros, el retraso del programador, el tiempo de deserialización de la tarea, el tiempo de espera de obtención de lectura aleatoria, las lecturas remotas aleatorias, el tiempo de serialización de resultados, el tiempo de obtención de resultados y la memoria de ejecución máxima.

Aggregated Metrics by Executor

Show 20 entries Search:

Executor ID	Logs	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Excluded	Shuffle Read Size / Records	Peak JVM Memory OnHeap / OffHeap	Peak Execution Memory OnHeap / OffHeap	Peak Storage Memory OnHeap / OffHeap	Peak Pool Memory Direct / Mapped
driver		192.168.0.19:45173	7.0 ms	1	0	0	1	false	144 B / 2	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B

Showing 1 to 1 of 1 entries Previous 1 Next

Tasks (1)

Show 20 entries Search:

Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Scheduler Delay	Task Deserialization Time	Shuffle Read Fetch Wait Time	Shuffle Remote Reads	Result Serialization Time	Getting Result Time	Peak Execution Memory	Shuffle Read Size / Records	Errors
0	110	0	SUCCESS	NODE_LOCAL	driver	192.168.0.19		2024-03-30 21:06:08	2.0 ms		2.0 ms	3.0 ms	0.0 ms	0.0 B			2.2 MiB	144 B / 2	

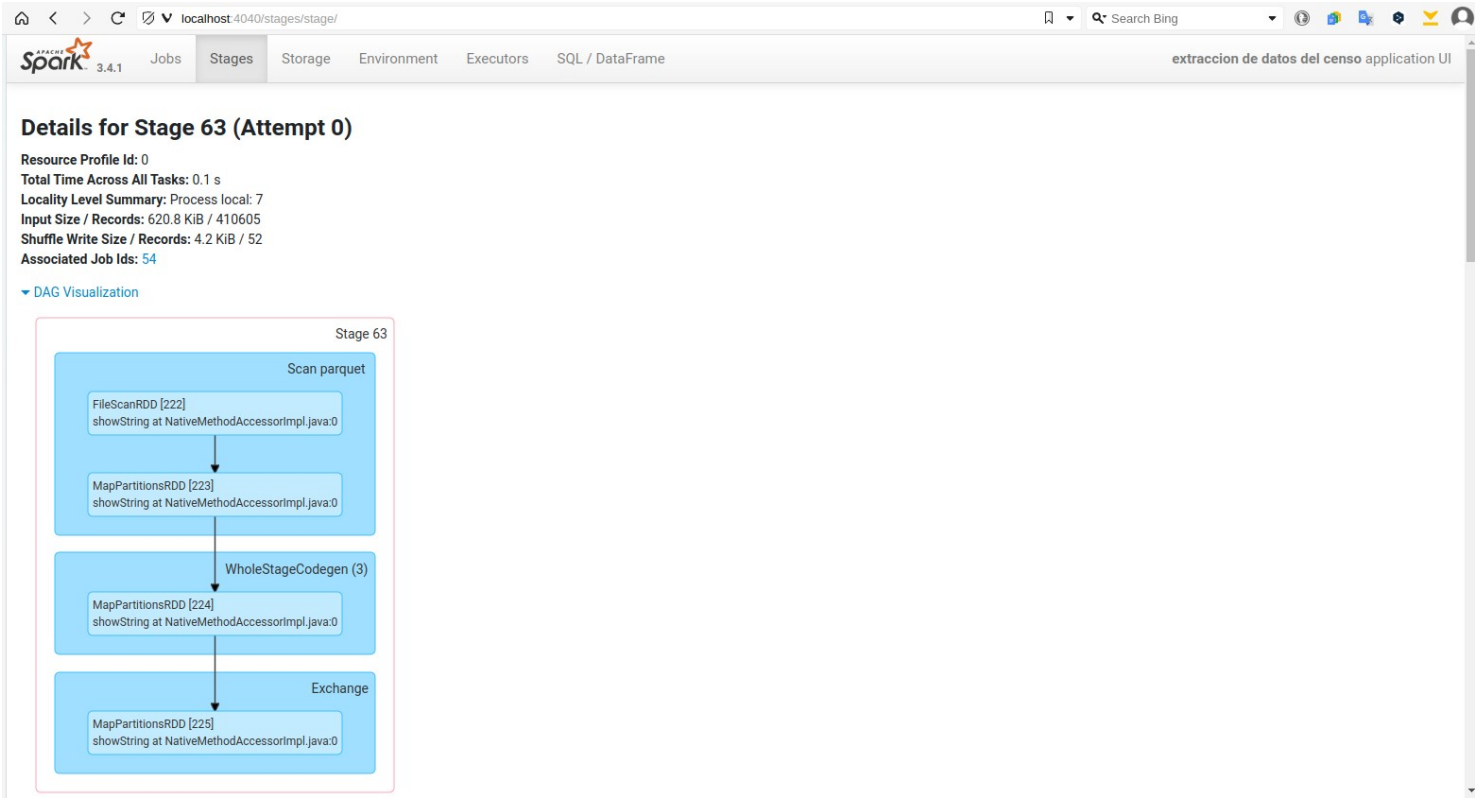
Showing 1 to 1 of 1 entries Previous 1 Next

Aggregated Metrics by Executor: Esta tabla muestra ID para cada ejecutor en la aplicación Spark. La tabla incluye las informaciones como la dirección del ejecutor (observamos que IP address de nuestro ordenador), la duración de tareas que ha ejecutado, total de tareas, tareas fallidas, tareas eliminadas, tareas completadas.

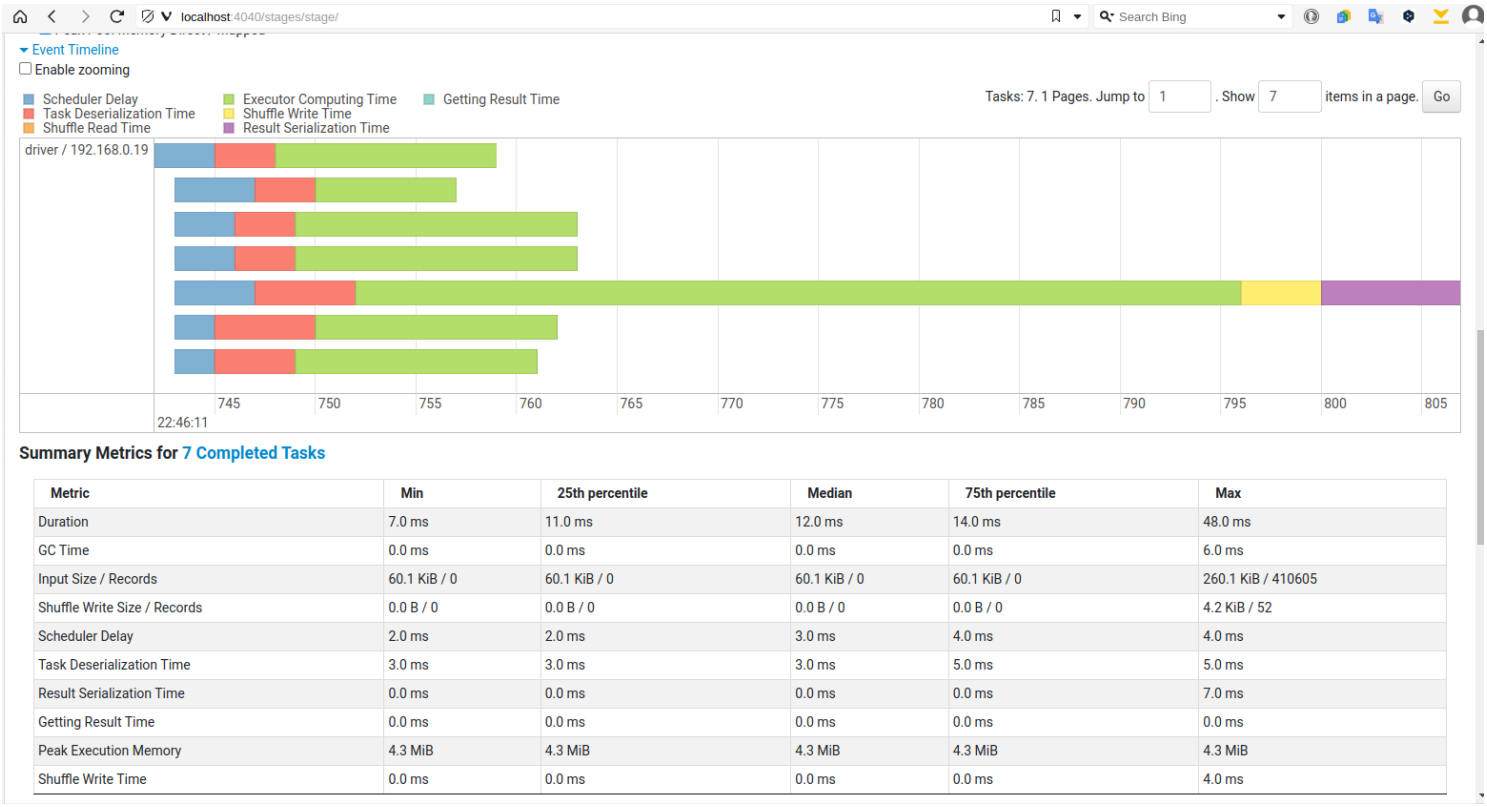
Tasks: Esta tabla muestra detalles sobre las tareas que se están ejecutando o se han ejecutado en la aplicación Spark. La tabla incluye el ID de la tarea, el host donde se ejecutó la tarea (observamos que IP address de nuestro ordenador), la hora de lanzamiento, la duración de la tarea y shuffle read size.

- Identifique en la interfaz las tareas (tasks) y fases (stages) que corresponden a las operaciones qué ha realizado (no es necesario identificar todas las tareas, basta con una muestra de las tareas más representativas del proceso que ha realizado).

En los datos de censo, que filtramos sólo para Francia y Portugal, vimos cuántos inmigrantes hay en cada provincia. Cuando ejecutemos esto, vamos a explicar las tareas y fases en la interfaz de Spark.



Stage ID: 63
Total Time Across All Tasks: 0.1 s (muestra que tiempo total de todas las tareas)
Locality Level Summary: Process local: 7 (Esto indica que la mayoría de las tareas de procesamiento de datos para esta etapa se realizaron en los mismos nodos trabajadores que almacenaron los datos, lo que puede mejorar el rendimiento)
Input Size/Records: 620.8 KiB/410605 (El tamaño de los datos de entrada a esta etapa es de 620,8 KiB y contiene 410.605 registros)
Shuffle Write Size / Records: La cantidad de datos shuffled entre etapas es de 4,2 KiB y contiene 52 registros. Shuffling es el proceso de mover datos por la red a diferentes nodos trabajadores para su procesamiento.
Associated Job Ids: Esta etapa está asociada al job ID 54. (Porque ejecutamos unas veces)
DAG Visualization: Esta sección proporciona una representación visual del Directed Acyclic Graph (DAG) para esta etapa. El DAG muestra el linaje de operaciones que se ejecutan en el trabajo Spark. En general, este imagen sugiere que la etapa 63 es una etapa de corta ejecución que lee datos de un archivo Parquet (indicado por la presencia del operador Scan Parquet) y luego los procesa localmente en el nodo trabajador. Por ejemplo, MapPartitionsRDD, es una transformación que aplica una función a cada partición del RDD.



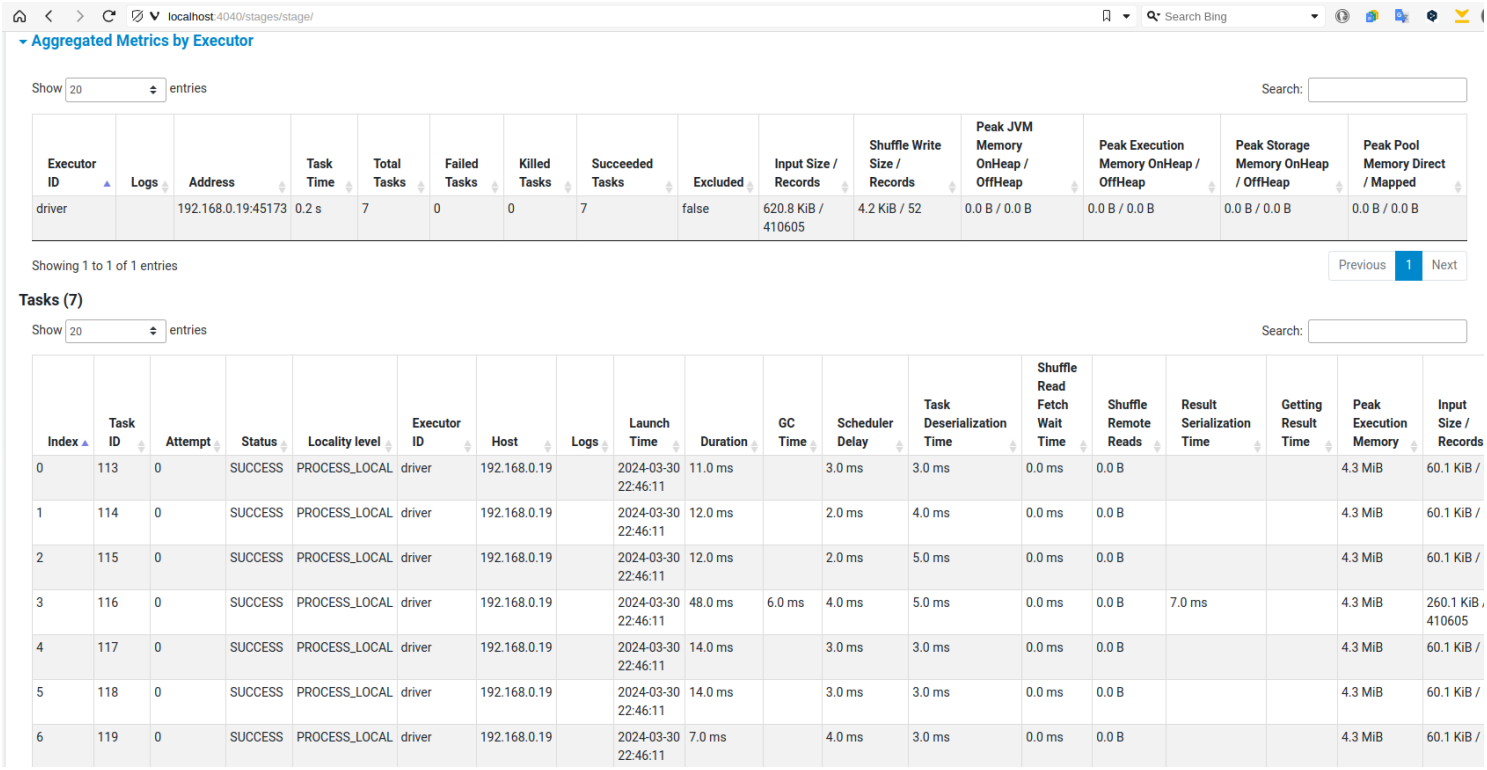
Summary Metrics for 7 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	7.0 ms	11.0 ms	12.0 ms	14.0 ms	48.0 ms
GC Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	6.0 ms
Input Size / Records	60.1 KiB / 0	60.1 KiB / 0	60.1 KiB / 0	60.1 KiB / 0	260.1 KiB / 410605
Shuffle Write Size / Records	0.0 B / 0	0.0 B / 0	0.0 B / 0	0.0 B / 0	4.2 KiB / 52
Scheduler Delay	2.0 ms	2.0 ms	3.0 ms	4.0 ms	4.0 ms
Task Deserialization Time	3.0 ms	3.0 ms	3.0 ms	5.0 ms	5.0 ms
Result Serialization Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	7.0 ms
Getting Result Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Peak Execution Memory	4.3 MiB	4.3 MiB	4.3 MiB	4.3 MiB	4.3 MiB
Shuffle Write Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	4.0 ms

En esta imagen observamos la sección Summary Metrics for 7 Completed Tasks de la interfaz de usuario de Spark. Esta sección proporciona estadísticas resumidas de siete tareas que se han ejecutado con éxito.

Duration: La cantidad total de tiempo que se tardó en completar la tarea.
GC Time: La cantidad de tiempo usado en realizar la recolección de basura.
Input Size/Records: La cantidad de datos leídos por la tarea.
Shuffle Write Size/Records: La cantidad de datos que fueron escritos por la tarea para ser shuffled.
Scheduler Delay: La cantidad de tiempo que tardó el programador de Spark en asignar una tarea a un ejecutor.
Task Deserialization Time: El tiempo que tardan los ejecutores en deserializar las tareas asignadas.
Result Serialization Time: Tiempo que tarda en serializar los resultados de la tarea en el ejecutor.
Getting Result Time: Tiempo necesario para obtener los resultados de la tarea del ejecutor.
Peak Execution Memory: La cantidad de memoria utilizada por la tarea en su peak.
Shuffle Write Time: El tiempo que se tarda en escribir los resultados de la tarea en la carpeta shuffle.

El **Event Timeline** muestra Scheduler Delay, Task Deserialization Time, Executor Computing Time para todas las tareas. Además ShuffleWrite Time and Result Serialization Time para una tarea específica.



Aggregated Metrics by Executor

Show 20 entries Search:

Executor ID	Logs	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Excluded	Input Size / Records	Shuffle Write Size / Records	Peak JVM Memory OnHeap / OffHeap	Peak Execution Memory OnHeap / OffHeap	Peak Storage Memory OnHeap / OffHeap	Peak Pool Memory Direct / Mapped
driver		192.168.0.19:45173	0.2 s	7	0	0	7	false	620.8 KiB / 410605	4.2 KiB / 52	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B

Showing 1 to 1 of 1 entries Previous 1 Next

Tasks (7)

Show 20 entries Search:

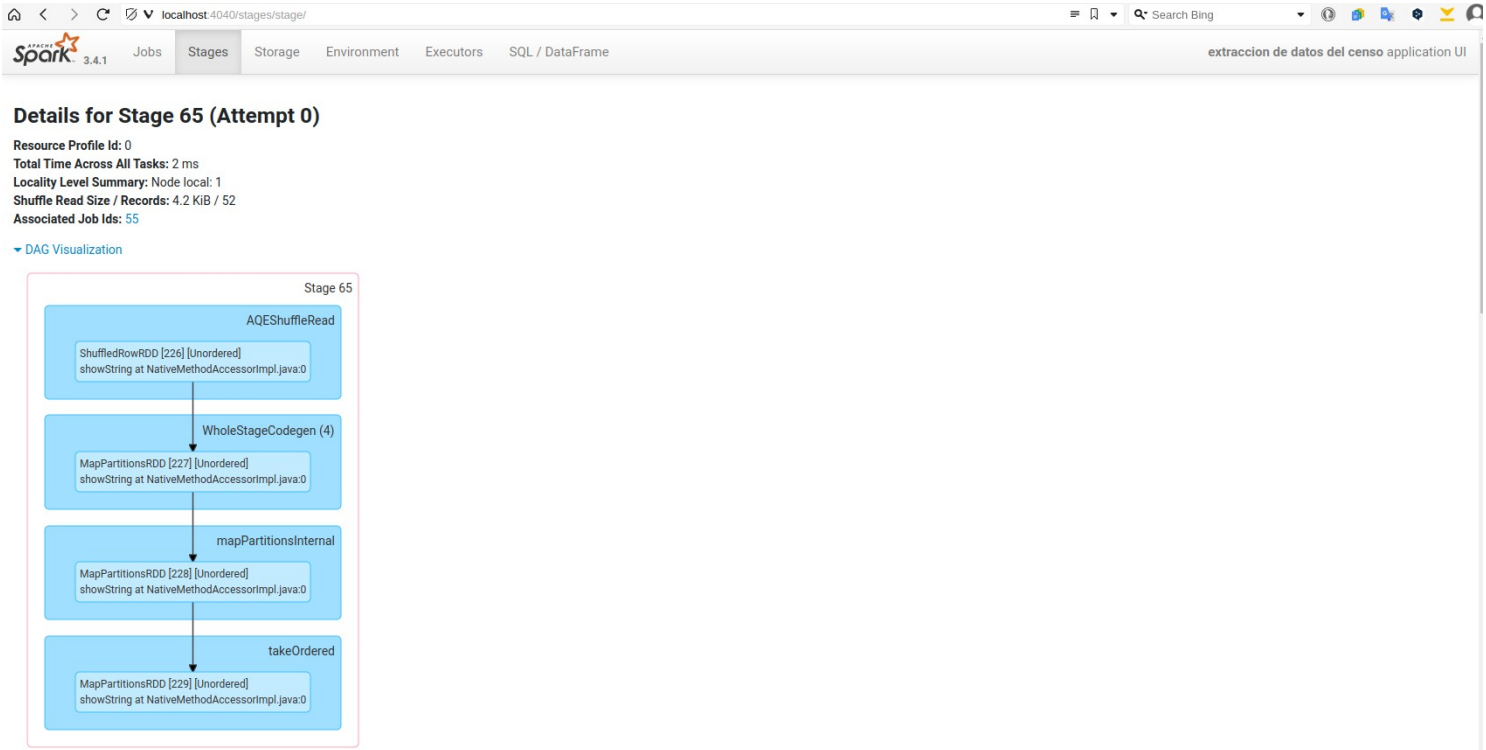
Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Scheduler Delay	Task Deserialization Time	Shuffle Read Fetch Wait Time	Shuffle Remote Reads	Result Serialization Time	Getting Result Time	Peak Execution Memory	Input Size / Records
0	113	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	11.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /
1	114	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	12.0 ms		2.0 ms	4.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /
2	115	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	12.0 ms		2.0 ms	5.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /
3	116	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	48.0 ms	6.0 ms	4.0 ms	5.0 ms	0.0 ms	0.0 B	7.0 ms		4.3 MiB	260.1 KiB / 410605
4	117	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	14.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /
5	118	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	14.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /
6	119	0	SUCCESS	PROCESS_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	7.0 ms		4.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB /

Search:

Launch Time	Duration	GC Time	Scheduler Delay	Task Deserialization Time	Shuffle Read Fetch Wait Time	Shuffle Remote Reads	Result Serialization Time	Getting Result Time	Peak Execution Memory	Input Size / Records	Shuffle Write Time	Shuffle Write Size / Records	Errors
2024-03-30 22:46:11	11.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			
2024-03-30 22:46:11	12.0 ms		2.0 ms	4.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			
2024-03-30 22:46:11	12.0 ms		2.0 ms	5.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			
2024-03-30 22:46:11	48.0 ms	6.0 ms	4.0 ms	5.0 ms	0.0 ms	0.0 B	7.0 ms		4.3 MiB	260.1 KiB / 410605	4.0 ms	4.2 KiB / 52	
2024-03-30 22:46:11	14.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			
2024-03-30 22:46:11	14.0 ms		3.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			
2024-03-30 22:46:11	7.0 ms		4.0 ms	3.0 ms	0.0 ms	0.0 B			4.3 MiB	60.1 KiB / 0			

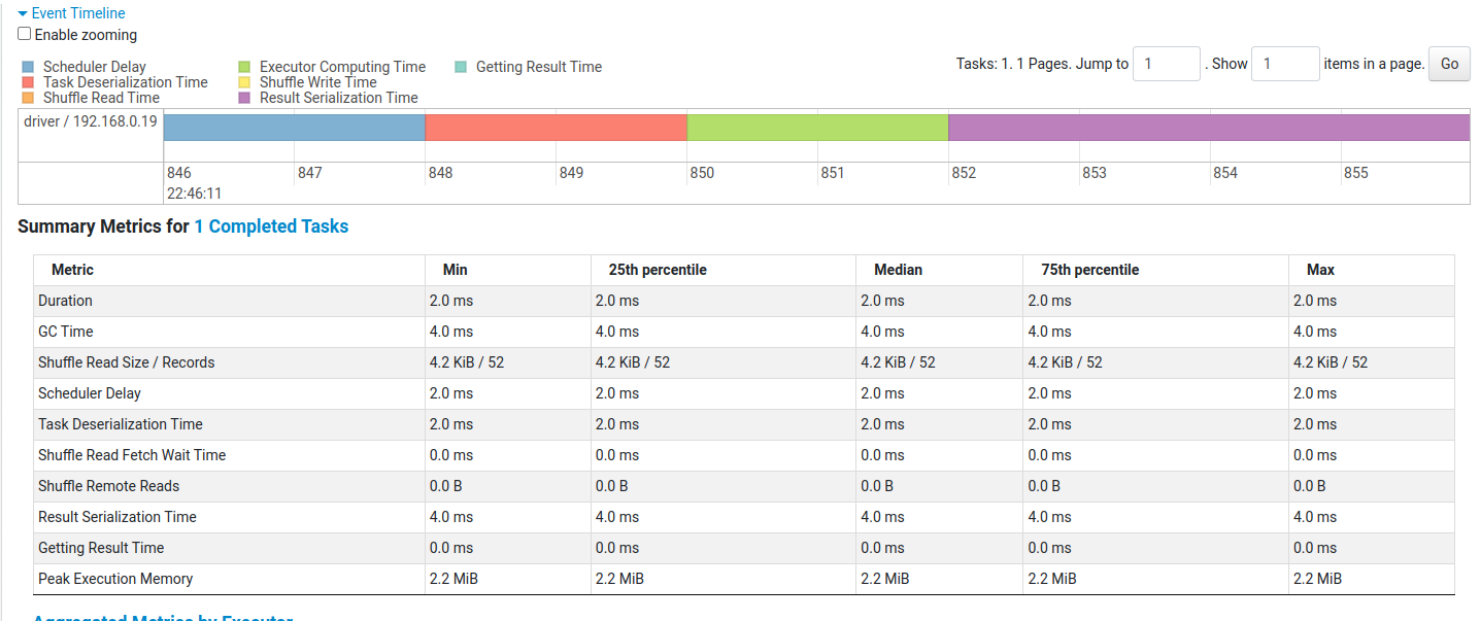
Aggregated Metrics by Executor: Como hemos mencionado antes, esta tabla muestra ID para cada ejecutor en la aplicación Spark. También incluye las informaciones como la dirección del ejecutor (observamos que IP address de nuestro ordenador), la duración de tareas que ha ejecutado, total de tareas, tareas fallidas, tareas eliminadas, tareas completadas.

Tasks: Esta tabla muestra detalles sobre las tareas que se están ejecutando o se han ejecutado en la aplicación Spark. Podemos ver lo que hemos dicho anteriormente (Duration,
GC Time, Input Size/Records, Shuffle Write Size/Records, Scheduler Delay, Task Deserialization Time, Result Serialization Time, Getting Result Time, Peak Execution Memory, Shuffle Write Time) por separado para cada tarea. También, observamos que no hay errores en ninguna de las tareas.
Además de eso, la tabla nos muestra ID de las tareas, Status (la tarea se realiza con éxito), Locality Level (Esto indica que las tareas se realizaron en los mismos nodos trabajadores), ID de ejecutor (es driver), Host (IP de ordenador), Launch time (el tiempo de ejecución).



La imagen muestra una visualización de Directed Acyclic Graph (DAG) de la Etapa 65. En las visualizaciones de DAG podemos ver las dependencias de las tareas dentro de una aplicación Spark. En este caso, el DAG muestra que la Etapa 65 consta de varias transformaciones aplicadas a un ShuffleRowRDD.

AQEShuffleRead: Es una operación shuffle que lee datos de un almacenamiento externo.
ShowString: Esta transformación convierte los datos a un formato de cadena.
Whole StageCodegen: Esto indica que Spark pudo generar código optimizado para esta etapa.
MapPartitionsRDD: Esta es una transformación que aplica una función a cada partición del RDD. En este caso, la función aplicada es la función showString mencionada anteriormente.



Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	2.0 ms	2.0 ms	2.0 ms	2.0 ms	2.0 ms
GC Time	4.0 ms	4.0 ms	4.0 ms	4.0 ms	4.0 ms
Shuffle Read Size / Records	4.2 KiB / 52	4.2 KiB / 52	4.2 KiB / 52	4.2 KiB / 52	4.2 KiB / 52
Scheduler Delay	2.0 ms	2.0 ms	2.0 ms	2.0 ms	2.0 ms
Task Deserialization Time	2.0 ms	2.0 ms	2.0 ms	2.0 ms	2.0 ms
Shuffle Read Fetch Wait Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Shuffle Remote Reads	0.0 B	0.0 B	0.0 B	0.0 B	0.0 B
Result Serialization Time	4.0 ms	4.0 ms	4.0 ms	4.0 ms	4.0 ms
Getting Result Time	0.0 ms	0.0 ms	0.0 ms	0.0 ms	0.0 ms
Peak Execution Memory	2.2 MiB	2.2 MiB	2.2 MiB	2.2 MiB	2.2 MiB

Aggregated Metrics by Executor

Show 20 entries Search:

Executor ID	Logs	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Excluded	Shuffle Read Size / Records	Peak JVM Memory OnHeap / OffHeap	Peak Execution Memory OnHeap / OffHeap	Peak Storage Memory OnHeap / OffHeap	Peak Pool Memory Direct / Mapped
driver		192.168.0.19:45173	10.0 ms	1	0	0	1	false	4.2 KiB / 52	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B	0.0 B / 0.0 B

Showing 1 to 1 of 1 entries Previous 1 Next

Tasks (1)

Show 20 entries Search:

Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Scheduler Delay	Task Deserialization Time	Shuffle Read Fetch Wait Time	Shuffle Remote Reads	Result Serialization Time	Getting Result Time	Peak Execution Memory	Shuffle Write Time	Shuffle Read Size / Records	Errors
0	120	0	SUCCESS	NODE_LOCAL	driver	192.168.0.19		2024-03-30 22:46:11	2.0 ms	4.0 ms	2.0 ms	2.0 ms	0.0 ms	0.0 B	4.0 ms		2.2 MiB		4.2 KiB / 52	

Showing 1 to 1 of 1 entries Previous 1 Next

Como hemos mencionado anteriormente:

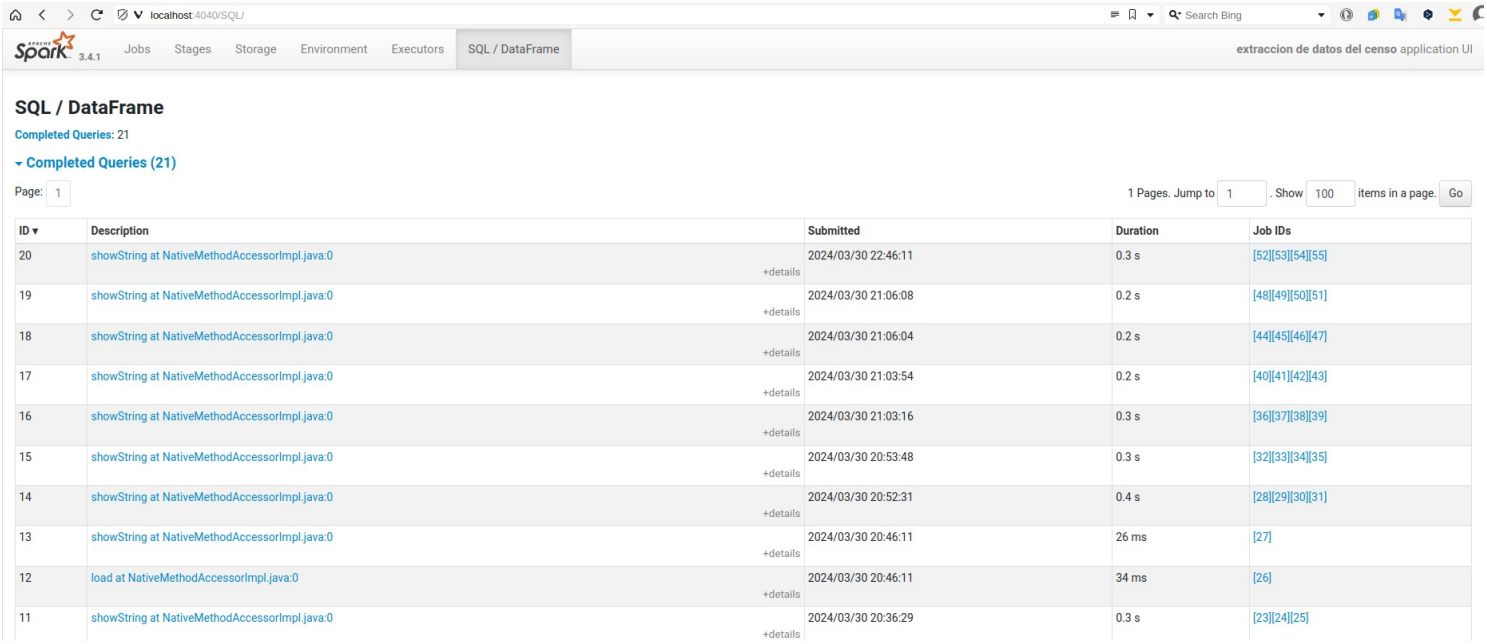
Aquí esta tabla de Summary Metrics también nos muestra estadísticas resumidas de la tarea que se ha ejecutado con éxito.

La tabla de Aggregated Metrics by Executor muestra ID para cada ejecutor en la aplicación Spark. También incluye las informaciones como la dirección del ejecutor, la duración de tareas que ha ejecutado, total de tareas, tareas fallidas, tareas eliminadas, tareas completadas.

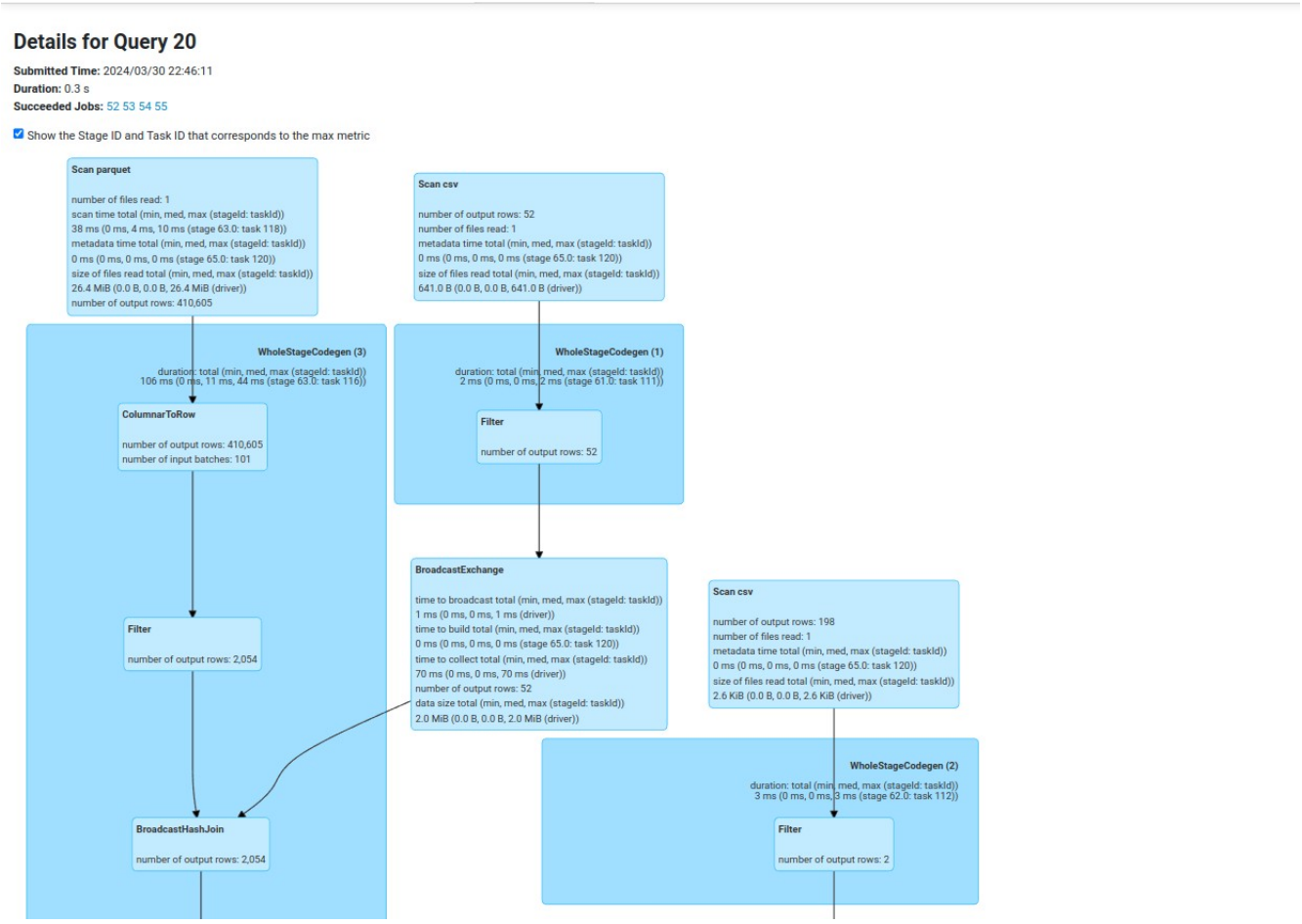
La tabla de Tasks muestra detalles sobre las tareas que se están ejecutando o se han ejecutado en la aplicación Spark. La tabla incluye Status de la tarea, el ID de la tarea, el host donde se ejecutó la tarea, la hora de ejecución, la duración de la tarea y shuffle read size.

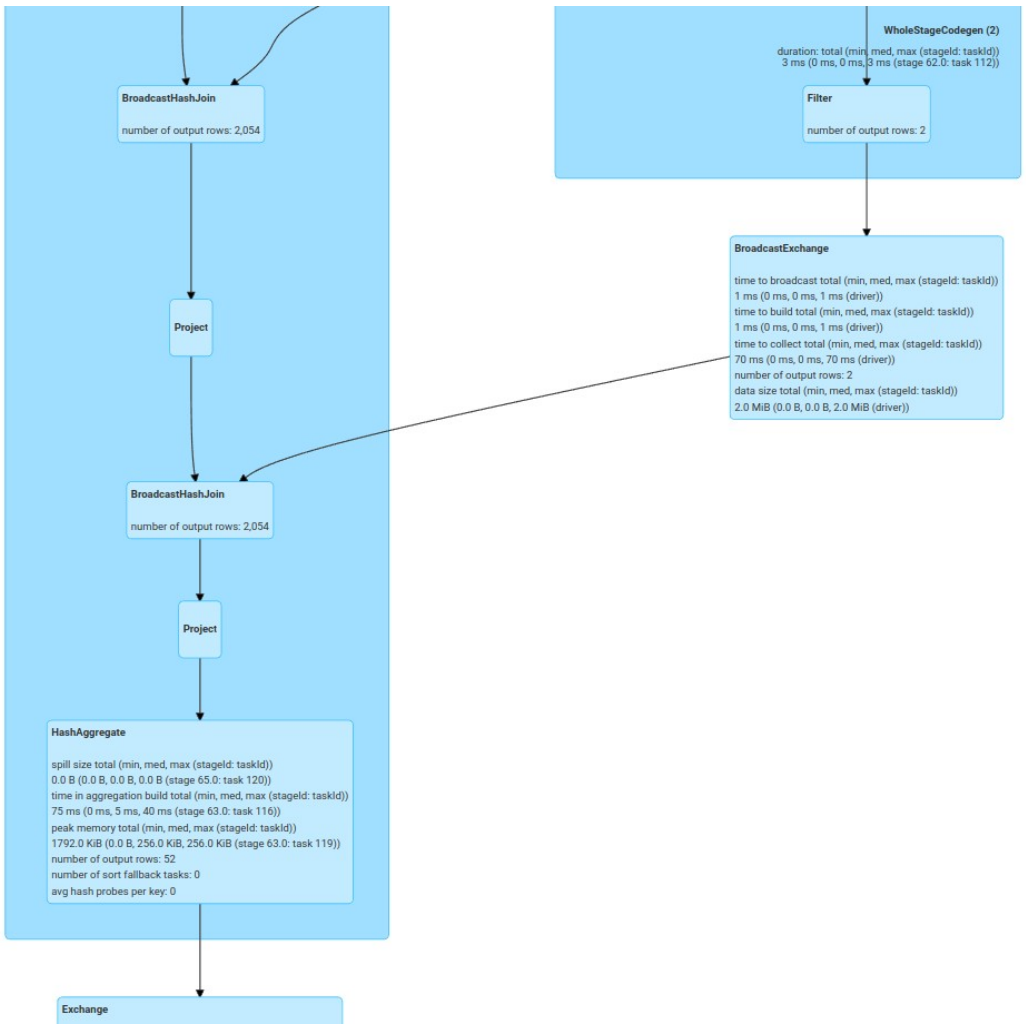
- Incluya y explique también resultados de la pestaña de SQL, que describe las operaciones con DataFrames, incluyendo diagramas de flujo de las consultas.

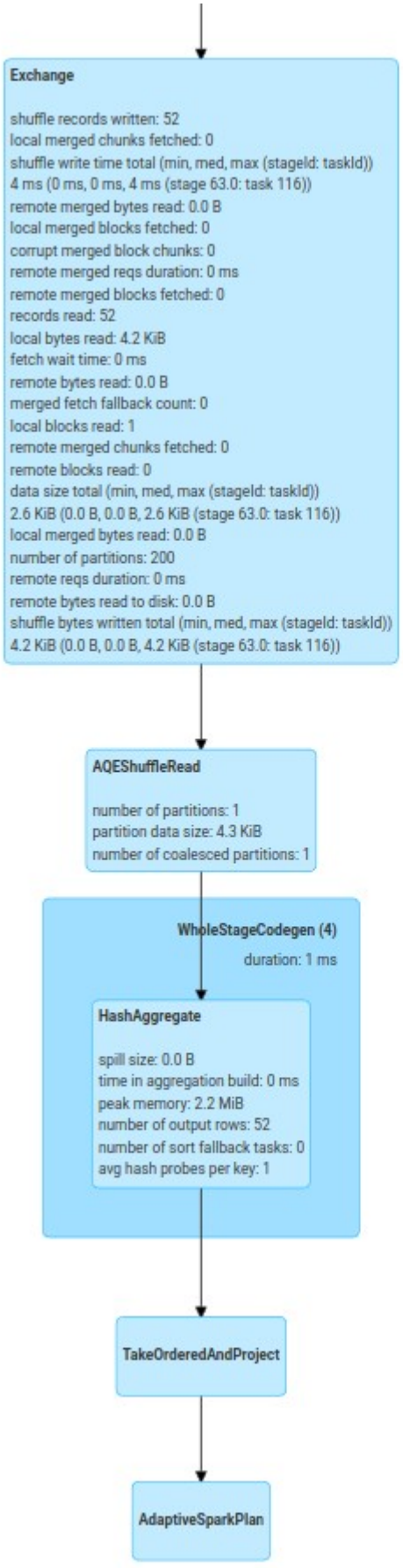
En la pestaña SQL/DataFrame observamos nuestros consultas y totalmente tenemos 21 consultas con la hora de ejecución, la duración de ejecución y IDs de las tareas.



ID	Description	Submitted	Duration	Job IDs
20	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 22:46:11	0.3 s	[52][53][54][55]
19	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 21:06:08	0.2 s	[48][49][50][51]
18	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 21:06:04	0.2 s	[44][45][46][47]
17	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 21:03:54	0.2 s	[40][41][42][43]
16	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 21:03:16	0.3 s	[36][37][38][39]
15	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 20:53:48	0.3 s	[32][33][34][35]
14	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 20:52:31	0.4 s	[28][29][30][31]
13	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 20:46:11	26 ms	[27]
12	load at NativeMethodAccessorImpl.java:0 +details	2024/03/30 20:46:11	34 ms	[26]
11	showString at NativeMethodAccessorImpl.java:0 +details	2024/03/30 20:36:29	0.3 s	[23][24][25]




BroadcastHashJoin
number of output rows: 2,054
Project
BroadcastHashJoin
number of output rows: 2,054
Project
HashAggregate
spill size total (min, med, max (stageId: taskId))
0.0 B (0.0 B, 0.0 B, 0.0 B (stage 65.0: task 120))
time in aggregation build total (min, med, max (stageId: taskId))
75 ms (0 ms, 5 ms, 40 ms (stage 63.0: task 116))
peak memory total (min, med, max (stageId: taskId))
1792.0 KiB (0.0 B, 256.0 KiB, 256.0 KiB (stage 63.0: task 119))
number of output rows: 52
number of sort fallback tasks: 0
avg hash probes per key: 0
Exchange
WholeStageCodegen (2)
duration: total (min, med, max (stageId: taskId))
3 ms (0 ms, 0 ms, 3 ms (stage 62.0: task 112))
Filter
number of output rows: 2
BroadcastExchange
time to broadcast total (min, med, max (stageId: taskId))
1 ms (0 ms, 0 ms, 1 ms (driver))
time to build total (min, med, max (stageId: taskId))
1 ms (0 ms, 0 ms, 1 ms (driver))
time to collect total (min, med, max (stageId: taskId))
70 ms (0 ms, 0 ms, 70 ms (driver))
number of output rows: 2
data size total (min, med, max (stageId: taskId))
2.0 MiB (0.0 B, 0.0 B, 2.0 MiB (driver))


Exchange
shuffle records written: 52
local merged chunks fetched: 0
shuffle write time total (min, med, max (stageId: taskId))
4 ms (0 ms, 0 ms, 4 ms (stage 63.0: task 116))
remote merged bytes read: 0.0 B
local merged blocks fetched: 0
corrupt merged block chunks: 0
remote merged reqs duration: 0 ms
remote merged blocks fetched: 0
records read: 52
local bytes read: 4.2 KiB
fetch wait time: 0 ms
remote bytes read: 0.0 B
merged fetch fallback count: 0
local blocks read: 1
remote merged chunks fetched: 0
remote blocks read: 0
data size total (min, med, max (stageId: taskId))
2.6 KiB (0.0 B, 0.0 B, 2.6 KiB (stage 63.0: task 116))
local merged bytes read: 0.0 B
number of partitions: 200
remote reqs duration: 0 ms
remote bytes read to disk: 0.0 B
shuffle bytes written total (min, med, max (stageId: taskId))
4.2 KiB (0.0 B, 0.0 B, 4.2 KiB (stage 63.0: task 116))
AQEShuffleRead
number of partitions: 1
partition data size: 4.3 KiB
number of coalesced partitions: 1
WholeStageCodegen (4)
duration: 1 ms
HashAggregate
spill size: 0.0 B
time in aggregation build: 0 ms
peak memory: 2.2 MiB
number of output rows: 52
number of sort fallback tasks: 0
avg hash probes per key: 1
TakeOrderedAndProject
AdaptiveSparkPlan

Al observar el diagrama de flujo de nuestra última consulta sobre el número de inmigrantes en 10 provincias, ordenados de mayor a menor:

Scan parquet: Esta etapa lee datos de un archivo Parquet. El total del tiempo de escaneo muestra el tiempo mínimo, mediano y máximo que tardó en escanear el archivo en diferentes tareas. Podemos ver que la etapa 63.0 tarea 118 tardó más tiempo (10 milisegundos) en escanear el archivo. También nos muestra los numeros de filas salidas y tamaño de ficheros leídos total.
Scan csv: Esta etapa lee datos de un archivo CSV y muestra tamaño de ficheros leídos total y los numeros de filas salidas. También el total del tiempo de escaneo muestra el tiempo mínimo, mediano y máximo que tardó en escanear el archivo en tarea.

WholeStageCodegen: Esta etapa genera código para una tarea específica. Para **WholeStageCodegen (3)**, esta generando código para convertir datos a un formato basado en filas.
ColumnarToRow: Esta etapa convierte datos a un formato basado en filas.
Filter: Esta etapa filtra datos según una condición específica.
BroadcastExchange: Esta etapa shuffles datos en el clúster.
BroadcastHashJoin: Esta etapa une dos conjuntos de datos mediante una unión hash (con numero de filas salidas).
HashAggregate: Esta etapa agrega los datos. La agregación hash es un tipo de agregación que podemos utilizar para agregar datos en una tabla grande.

Project: Esta etapa proyecta un conjunto específico de columnas de un DataFrame o tabla SQL.

Exchange: En esta etapa, los datos se barajan a través del clúster. El barajado (shuffle) es necesario cuando los datos deben redistribuirse entre diferentes ejecutores para su procesamiento. Por ejemplo, puede ser necesario barajar los datos después de una operación de join.

AQEShuffleRead: Esta etapa lee los datos barajados (shuffled) del gestor de bloques.

TakeOrderedAndProject: Esta etapa toma y ordena un conjunto específico de columnas a partir de la salida de la etapa anterior.
shuffle records written: Esta métrica representa la cantidad total de registros escritos en la etapa de barajado.
local merged chunks fetched: Esta métrica indica la cantidad de fragmentos de datos que se obtuvieron del servicio de barajado local. El servicio de barajado local almacena datos que se están barajando dentro del mismo ejecutor o entre ejecutores en el mismo nodo.
shuffle write time total: Esta métrica muestre el tiempo mínimo, mediano y máximo que tomó escribir datos en la etapa de barajado en diferentes tareas (identificadas por ID de etapa e ID de tarea).
records read: Esta métrica representa la cantidad total de registros que se leyeron de la etapa de barajado.
local bytes read: Esta métrica indica la cantidad total de bytes de datos que se leyeron del servicio de barajado local.
remote bytes read: Esta métrica indica la cantidad total de bytes de datos que se leyeron del servicio de barajado remoto.

local blocks read: Esta métrica indica la cantidad de bloques de datos que se leyeron del servicio de barajado local.
remote blocks read: Esta métrica representa la cantidad de bloques de datos que se leyeron del servicio de barajado remoto.

Data size total: Esta métrica muestra el tamaño mínimo, mediano y máximo de los datos que se barajaron en diferentes tareas (identificadas por ID de etapa e ID de tarea).
number of partitions: Esta métrica representa la cantidad de particiones en las que se dividieron los datos para el barajado.
partition data size: Esta métrica indica el tamaño de una.

Al observar Details en la pestaña SQL/DataFrames, vemos un plan de consultas. Este plan de consulta muestra las etapas físicas implicadas en la ejecución de una consulta. Las etapas físicas son los pasos reales que dará Spark para ejecutar la consulta.

Como hemos mencionado anteriormente, observamos los pasos mismos con detalles.
Podemos ver en detalle el plan inicial, el plan final, qué columnas se unen, qué tipo de join se utiliza durante la operación join, la lectura y localización de los ficheros csv y parquet, qué columnas se filtran, qué columnas se ven cuando se realiza shuffle.

Details

```
== Physical Plan ==
AdaptiveSparkPlan (35)
+- == Final Plan ==
   TakeOrderedAndProject (21)
   +- * HashAggregate (20)
      +- AQEShuffleRead (19)
         +- ShuffleQueryStage (18), Statistics(sizeInBytes=2.6 KiB, rowCount=52)
            +- Exchange (17)
               +- * HashAggregate (16)
                  +- * Project (15)
                     +- * BroadcastHashJoin LeftOuter BuildRight (14)
                        :- * Project (9)
                        :  +- * BroadcastHashJoin LeftOuter BuildRight (8)
                        :     :- * Filter (3)
                        :     :  +- * ColumnarToRow (2)
                        :     :     +- Scan parquet  (1)
                        :     +- BroadcastQueryStage (7), Statistics(sizeInBytes=2.0 MiB, rowCount=52)
                        :        +- BroadcastExchange (6)
                        :           +- * Filter (5)
                        :              +- Scan csv  (4)
                        +- BroadcastQueryStage (13), Statistics(sizeInBytes=2.0 MiB, rowCount=2)
                           +- BroadcastExchange (12)
                              +- * Filter (11)
                                 +- Scan csv  (10)
+- == Initial Plan ==
   TakeOrderedAndProject (34)
   +- HashAggregate (33)
      +- Exchange (32)
         +- HashAggregate (31)
            +- Project (30)
               +- BroadcastHashJoin LeftOuter BuildRight (29)
                  :- Project (26)
                  :  +- BroadcastHashJoin LeftOuter BuildRight (25)
                  :     :- Filter (22)
                  :     :  +- Scan parquet  (1)
                  :     +- BroadcastExchange (24)
                  :        +- Filter (23)
                  :           +- Scan csv  (4)
                  +- BroadcastExchange (28)
                     +- Filter (27)
                        +- Scan csv  (10)


(1) Scan parquet
Output [2]: [CPRO#1330, CPAISN#1340]
Batched: true
Location: InMemoryFileIndex [file:/home/merve/Desktop/Infraestructura_para_big_data/Practica3/UAM/Spark-infra/data/censo-2011s.parquet]
ReadSchema: struct<CPRO:string,CPAISN:string>

(2) ColumnarToRow [codegen id : 3]
Input [2]: [CPRO#1330, CPAISN#1340]

(3) Filter [codegen id : 3]
Input [2]: [CPRO#1330, CPAISN#1340]
Condition : ((cast(CPAISN#1340 as int) = 110) OR (cast(CPAISN#1340 as int) = 123))

(4) Scan csv
Output [2]: [CPRO#2503, NOMBRE_PRO#2504]
Batched: false
Location: InMemoryFileIndex [file:/home/merve/Desktop/Infraestructura_para_big_data/Practica3/UAM/Spark-infra/data/codes-prov.csv]
PushedFilters: [IsNotNull(CPRO)]
ReadSchema: struct<CPRO:string,NOMBRE_PRO:string>

(5) Filter [codegen id : 1]
Input [2]: [CPRO#2503, NOMBRE_PRO#2504]
Condition : isnotnull(CPRO#2503)

(6) BroadcastExchange
Input [2]: [CPRO#2503, NOMBRE_PRO#2504]
Arguments: HashedRelationBroadcastMode(List(input[0, string, false]),false), [plan_id=1955]

(7) BroadcastQueryStage
Output [2]: [CPRO#2503, NOMBRE_PRO#2504]
```

```
(7) BroadcastQueryStage
Output [2]: [CPRO#2503, NOMBRE_PRO#2504]
Arguments: 0

(8) BroadcastHashJoin [codegen id : 3]
Left keys [1]: [CPRO#1330]
Right keys [1]: [CPRO#2503]
Join type: LeftOuter
Join condition: None

(9) Project [codegen id : 3]
Output [3]: [CPRO#1330, CPAISN#1340, NOMBRE_PRO#2504]
Input [4]: [CPRO#1330, CPAISN#1340, CPRO#2503, NOMBRE_PRO#2504]

(10) Scan csv
Output [1]: [COD_PAIS#4128]
Batched: false
Location: InMemoryFileIndex [file:/home/merve/Desktop/Infraestructura_para_big_data/Practica3/UAM/Spark-infra/data/codes-cpais.csv]
PushedFilters: [IsNotNull(COD_PAIS)]
ReadSchema: struct<COD_PAIS:string>

(11) Filter [codegen id : 2]
Input [1]: [COD_PAIS#4128]
Condition : (((cast(COD_PAIS#4128 as int) = 110) OR (cast(COD_PAIS#4128 as int) = 123)) AND isnotnull(COD_PAIS#4128))

(12) BroadcastExchange
Input [1]: [COD_PAIS#4128]
Arguments: HashedRelationBroadcastMode(List(input[0, string, false]),false), [plan_id=1967]

(13) BroadcastQueryStage
Output [1]: [COD_PAIS#4128]
Arguments: 1

(14) BroadcastHashJoin [codegen id : 3]
Left keys [1]: [CPAISN#1340]
Right keys [1]: [COD_PAIS#4128]
Join type: LeftOuter
Join condition: None

(15) Project [codegen id : 3]
Output [2]: [CPRO#1330, NOMBRE_PRO#2504]
Input [4]: [CPRO#1330, CPAISN#1340, NOMBRE_PRO#2504, COD_PAIS#4128]

(16) HashAggregate [codegen id : 3]
Input [2]: [CPRO#1330, NOMBRE_PRO#2504]
Keys [2]: [CPRO#1330, NOMBRE_PRO#2504]
Functions [1]: [partial_count(1)]
Aggregate Attributes [1]: [count#6296L]
Results [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]

(17) Exchange
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Arguments: hashpartitioning(CPRO#1330, NOMBRE_PRO#2504, 200), ENSURE_REQUIREMENTS, [plan_id=2069]

(18) ShuffleQueryStage
Output [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Arguments: 2

(19) AQEShuffleRead
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Arguments: coalesced

(20) HashAggregate [codegen id : 4]
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Keys [2]: [CPRO#1330, NOMBRE_PRO#2504]
Functions [1]: [count(1)]
Aggregate Attributes [1]: [count(1)#6282L]
Results [3]: [CPRO#1330, NOMBRE_PRO#2504, count(1)#6282L AS count#6283L]

(21) TakeOrderedAndProject
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6283L]
Arguments: 11, [count#6283L DESC NULLS LAST], [CPRO#1330, NOMBRE_PRO#2504, cast(count#6283L as string) AS count#6292]
```

```
(22) Filter
Input [2]: [CPRO#1330, CPAISN#1340]
Condition : ((cast(CPAISN#1340 as int) = 110) OR (cast(CPAISN#1340 as int) = 123))

(23) Filter
Input [2]: [CPRO#2503, NOMBRE_PRO#2504]
Condition : isnotnull(CPRO#2503)

(24) BroadcastExchange
Input [2]: [CPRO#2503, NOMBRE_PRO#2504]
Arguments: HashedRelationBroadcastMode(List(input[0, string, false]),false), [plan_id=1917]

(25) BroadcastHashJoin
Left keys [1]: [CPRO#1330]
Right keys [1]: [CPRO#2503]
Join type: LeftOuter
Join condition: None

(26) Project
Output [3]: [CPRO#1330, CPAISN#1340, NOMBRE_PRO#2504]
Input [4]: [CPRO#1330, CPAISN#1340, CPRO#2503, NOMBRE_PRO#2504]

(27) Filter
Input [1]: [COD_PAIS#4128]
Condition : (((cast(COD_PAIS#4128 as int) = 110) OR (cast(COD_PAIS#4128 as int) = 123)) AND isnotnull(COD_PAIS#4128))

(28) BroadcastExchange
Input [1]: [COD_PAIS#4128]
Arguments: HashedRelationBroadcastMode(List(input[0, string, false]),false), [plan_id=1921]

(29) BroadcastHashJoin
Left keys [1]: [CPAISN#1340]
Right keys [1]: [COD_PAIS#4128]
Join type: LeftOuter
Join condition: None

(30) Project
Output [2]: [CPRO#1330, NOMBRE_PRO#2504]
Input [4]: [CPRO#1330, CPAISN#1340, NOMBRE_PRO#2504, COD_PAIS#4128]

(31) HashAggregate
Input [2]: [CPRO#1330, NOMBRE_PRO#2504]
Keys [2]: [CPRO#1330, NOMBRE_PRO#2504]
Functions [1]: [partial_count(1)]
Aggregate Attributes [1]: [count#6296L]
Results [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]

(32) Exchange
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Arguments: hashpartitioning(CPRO#1330, NOMBRE_PRO#2504, 200), ENSURE_REQUIREMENTS, [plan_id=1926]

(33) HashAggregate
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6297L]
Keys [2]: [CPRO#1330, NOMBRE_PRO#2504]
Functions [1]: [count(1)]
Aggregate Attributes [1]: [count(1)#6282L]
Results [3]: [CPRO#1330, NOMBRE_PRO#2504, count(1)#6282L AS count#6283L]

(34) TakeOrderedAndProject
Input [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6283L]
Arguments: 11, [count#6283L DESC NULLS LAST], [CPRO#1330, NOMBRE_PRO#2504, cast(count#6283L as string) AS count#6292]

(35) AdaptiveSparkPlan
Output [3]: [CPRO#1330, NOMBRE_PRO#2504, count#6292]
Arguments: isFinalPlan=true
```