

El Quijote en el Data Center



Un Viaje Práctico con MapReduce

El Reto: Contar Cada Palabra de *El Quijote*

El Ingenioso Hidalgo Don Quijote de la Mancha contiene más de 380,000 palabras. ¿Cómo podríamos contarlas todas, una por una, de forma rápida y eficiente?

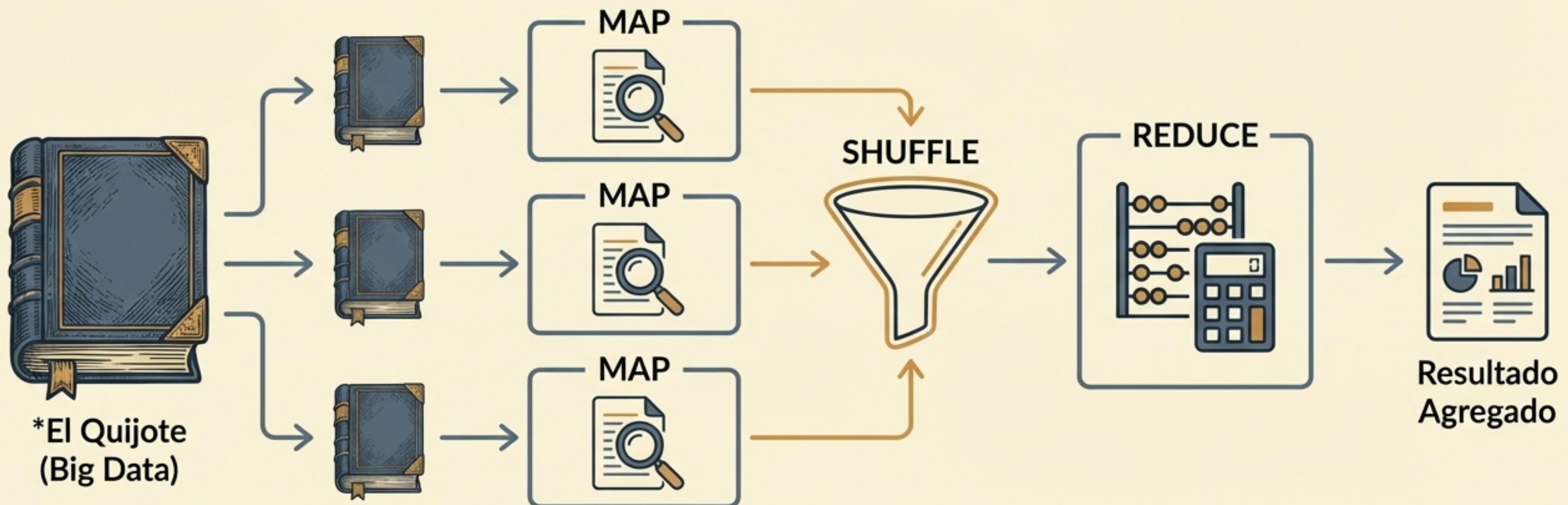
Un simple script en un solo ordenador podría tardar mucho tiempo y fallaría con un texto de tamaño terabytes. Necesitamos un enfoque diferente para el análisis a gran escala.



La Solución Conceptual: El Paradigma MapReduce

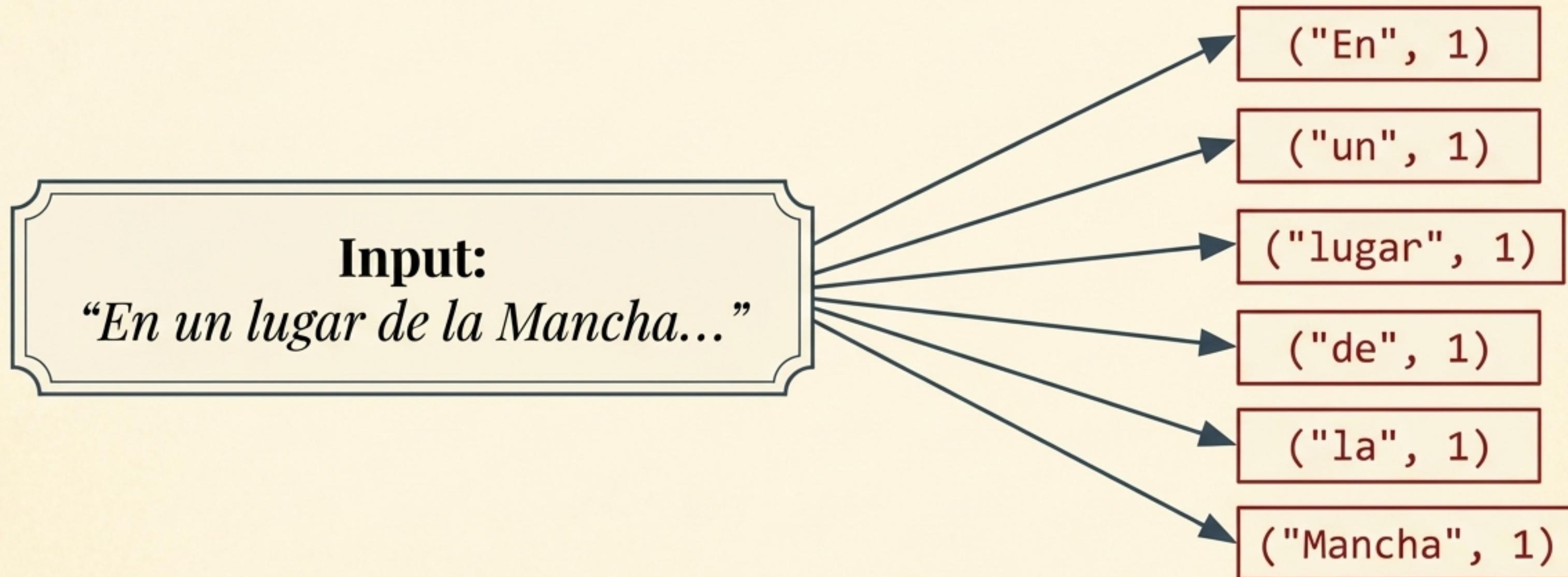
MapReduce es un modelo de programación para procesar grandes conjuntos de datos de forma paralela y distribuida. En lugar de mover los datos a un único punto para su cómputo, MapReduce lleva el cómputo a donde residen los datos.

Se divide en dos fases principales y una intermedia: **Map**, **Shuffle** y **Reduce**.



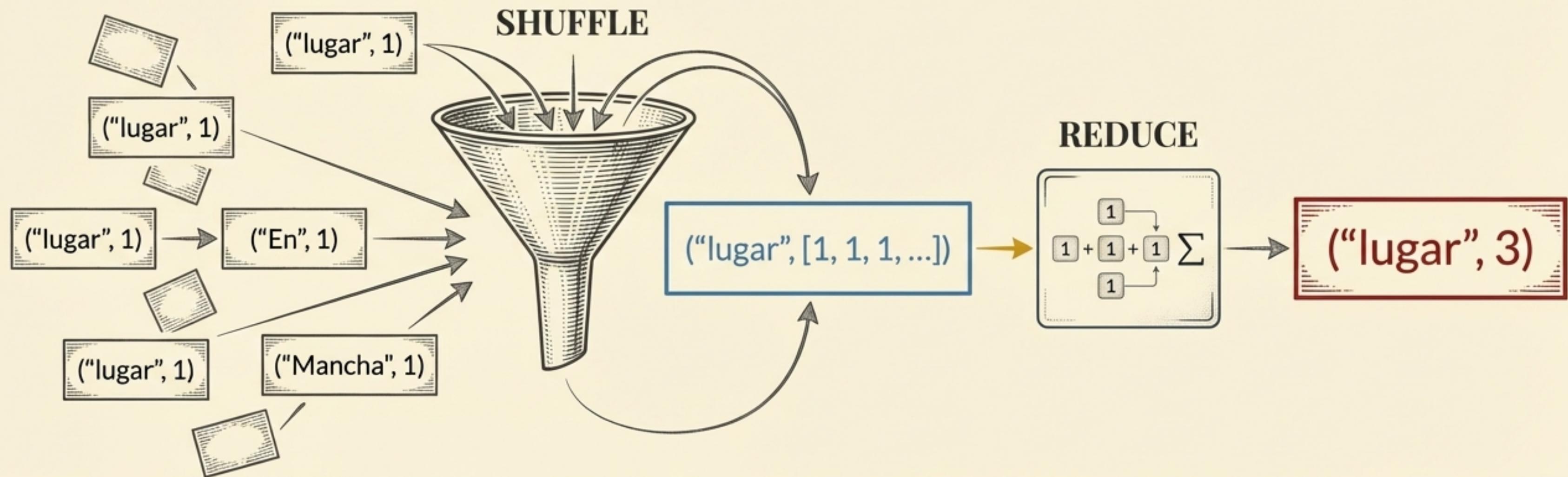
Fase 1: MAP (Mapear)

Cada “Mapper” recibe un fragmento del texto, lo divide en palabras y emite un par clave-valor por cada una: `(palabra, 1)`. Actúa como un censo individual.



Fases 2 y 3: SHUFFLE (Agrupar) y REDUCE (Reducir)

- **Shuffle:** El framework agrupa y ordena todos los pares emitidos por los mappers, juntando todos los valores asociados a la misma clave.
- **Reduce:** Cada ‘Reducer’ procesa una clave y su lista de valores para generar un resultado final. En WordCount, simplemente suma los ‘1’ para obtener el conteo total.



De la Teoría a la Práctica: El Experimento WordCount

Ahora, apliquemos este paradigma para analizar la obra completa de *El Quijote* utilizando el ecosistema Hadoop.



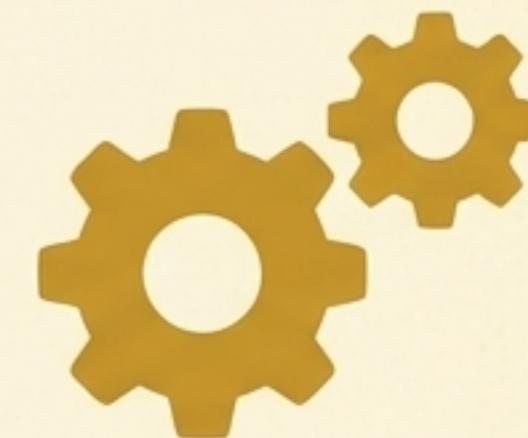
HDFS

Nuestro sistema de archivos distribuido donde almacenaremos el libro.



YARN

El gestor de recursos que coordinará la ejecución de nuestro trabajo.



MapReduce

El motor de cómputo que ejecutará la lógica de WordCount.

Paso 1: Preparando el Manuscrito en HDFS

Subimos el fichero `quijote.txt` a nuestro clúster Hadoop.

1. `hdfs dfs -mkdir -p /practicas`
2. `hdfs dfs -put ~/Descargas/quijote.txt /practicas/`

****Verificación:**** `hdfs dfs -ls /practicas`

```
● ● ●  
Found 1 items  
-rw-r--r-- 3 cloudera cloudera 2152865 2023-10-27 10:30 /practicas/quijote.txt
```

Entregable 1:
Fichero cargado y
visible en HDFS.

Paso 2: Lanzando el Job de MapReduce

Ejecutamos el programa WordCount, que viene incluido en los ejemplos de Hadoop, indicándole el fichero de entrada y el directorio de salida.

```
hadoop jar <path_to_examples>.jar wordcount \  
 /practicas/quijote.txt \  
 /practicas/resultado
```

El programa específico a ejecutar.

Fichero de entrada en HDFS.

Directorio de salida en HDFS (no debe existir previamente).



Pro Tip: Durante la ejecución, la consola muestra un 'Tracking URL'. Este es el enlace para monitorizar el progreso en tiempo real a través de la interfaz web de YARN.

El Resultado en HDFS: La Evidencia del Éxito

Una vez finalizado el job, verificamos el contenido del directorio de salida.

hdfs dfs -ls /practicas/resultado

```
Found 2 items
-rw-r--r-- 3 cloudera cloudera      0 2023-10-27 10:35 /practicas/resultado/_SUCCESS
-rw-r--r-- 3 cloudera cloudera 691498 2023-10-27 10:35 /practicas/resultado/part-r-00000
```

Este es el fichero que contiene nuestro resultado. El 'r' indica que es la salida de un Reducer.

Este fichero vacío confirma que el job terminó sin errores.

Esto cubre el Entregable 2.

Un Vistazo al Conteo: Las Primeras 10 Líneas

Inspeccionamos el contenido del fichero de resultados para ver las palabras más frecuentes (el resultado no está ordenado por frecuencia, sino alfabéticamente).

```
hdfs dfs -cat /practicas/resultado/part-r-00000 | head
```

"a"	21583
"abad"	14
"abadejo"	2
"abajo"	130
"abalanzó"	25
"abalorios"	3
"abarcaba"	2
"abarcar"	5
"abiertas"	55
"abierto"	72
...	

Esto responde al Entregable 3.

Profundizando el Análisis: El Job Overview en YARN

Usamos la interfaz web del History Server (<http://<hs>:19888/>) para analizar las métricas detalladas de la ejecución.

The screenshot shows the Cloudera History Server interface for a job named 'wordcount'. The 'Job Overview' tab is selected in the sidebar. The main content area displays the following information:

- Job Status:** State: SUCCEEDED, User: cloudera, Discordant: 0, Job Name: wordcount, Status: cloudwin, User Nation: 1.
- Timings:** Start Time: 2023-10-27 10:33:15, Finish Time: 2023-10-27 10:35:48, Elapsed: 2mins, 33sec.
- Task Counts:** Maps: Total 3, Succeeded 3, Failed 0, Killed 0, Running 0. Reduces: Total 1, Succeeded 1, Failed 0, Killed 0, Running 0.
- Counters:** (Table partially visible)

A red arrow points from a callout box on the left to the 'Counters' button in the sidebar, with the text: "Aquí se encuentran las métricas detalladas de E/S." A blue box highlights the 'Timings' section. A yellow box highlights the 'Task Counts' section. A yellow arrow points from a callout box on the right to the 'Task Counts' section, with the text: "Visión general del número de tareas."

Aquí se encuentran las métricas detalladas de E/S.

Visión general del número de tareas.

Esta vista general corresponde al Entregable 4.

Desglose del Job (1/2): Las Tareas MAP

Analizamos el rendimiento y los detalles de las tareas de mapeo.



Map Tasks				
Task ID	Status	Duration	HDFS_BYTES_READ	Retries
task_..._m_000000	SUCCEEDED	10s	1.05 MB	0
task_..._m_000001	SUCCEEDED	12s	1.00 MB	0

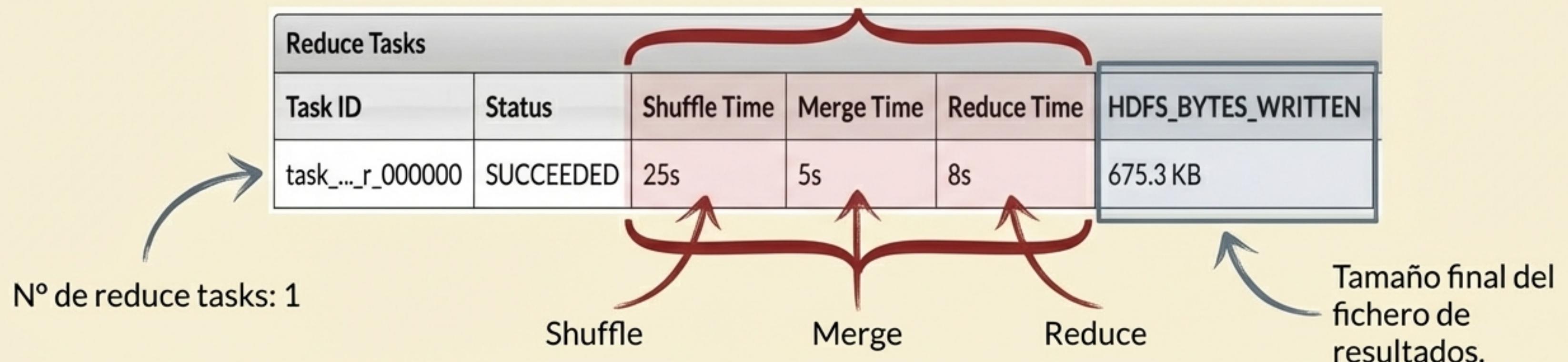
Nº de map tasks: 2

Cantidad de datos leídos por cada mapper.

****Insight**:** Cada tarea Map procesó una porción del fichero `quijote.txt` en paralelo.

Desglose del Job (2/2): Las Tareas REDUCE

Analizamos las fases de agregación que ocurren en las tareas de reducción.



****Insight**:** En este caso, un solo Reducer fue suficiente para agregar los resultados de todos los Mappers.

Resumen de Métricas Clave



****Reintentos de Tareas**: 0**

Estos datos cuantitativos nos permiten entender la escala y eficiencia de la operación.

La Pregunta Final: ¿Qué Fase Tardó Más y Por Qué?

¿Qué fase tardó más: map, shuffle o reduce?

Generalmente, la fase de **Shuffle** es la que consume más tiempo.

- La fase de **Map** es muy rápida: es una lectura local y un procesamiento en memoria.
- La fase de **Shuffle** es la más costosa: Implica mover grandes volúmenes de datos intermedios (los pares (palabra, 1)) a través de la red desde donde se ejecutaron los Mappers hasta el nodo donde se ejecutará el Reducer. Esta transferencia de datos y la posterior ordenación en disco es la **operación más intensiva**.

