Bases de Datos

Clase 14: Data Engineering

Hasta ahora

- Bases de datos relacionales
- SQL
- Data analytics (Pandas)
- NoSQL

¿Son estos todos los conceptos que necesitamos manejar para construir un sistema que maneja bien los datos?

- En algunos casos si …
- Pero a medida que pasa el tiempo, la complejidad de los sistemas aumenta y en algunos casos también la carga.
- Esto impone nuevos requisitos sobre las bases de datos, los cuales suelen requerir esfuerzos de ingeniería dedicados para poder hacerlos funcionar

Data Engineering

¿Por qué?

- Cuando los sistemas (y las organizaciones) crecen, nace la necesidad de generar protocolos e infraestructura para mantener los datos consistentes y accesibles 24/7, en distintos formatos.

- Distintos stakeholders necesitan consumir los datos de distinta manera: *data scientists, business intelligence, developers, etc.* Todos necesitan acceder a los datos de distinta forma y no pueden estar todos consultando la bd SQL principal.

- Todos los datos generados por un sistema tienen que ser extraídos, procesados y guardados según como los objetivos de la organización lo vayan requiriendo.

Data Engineering

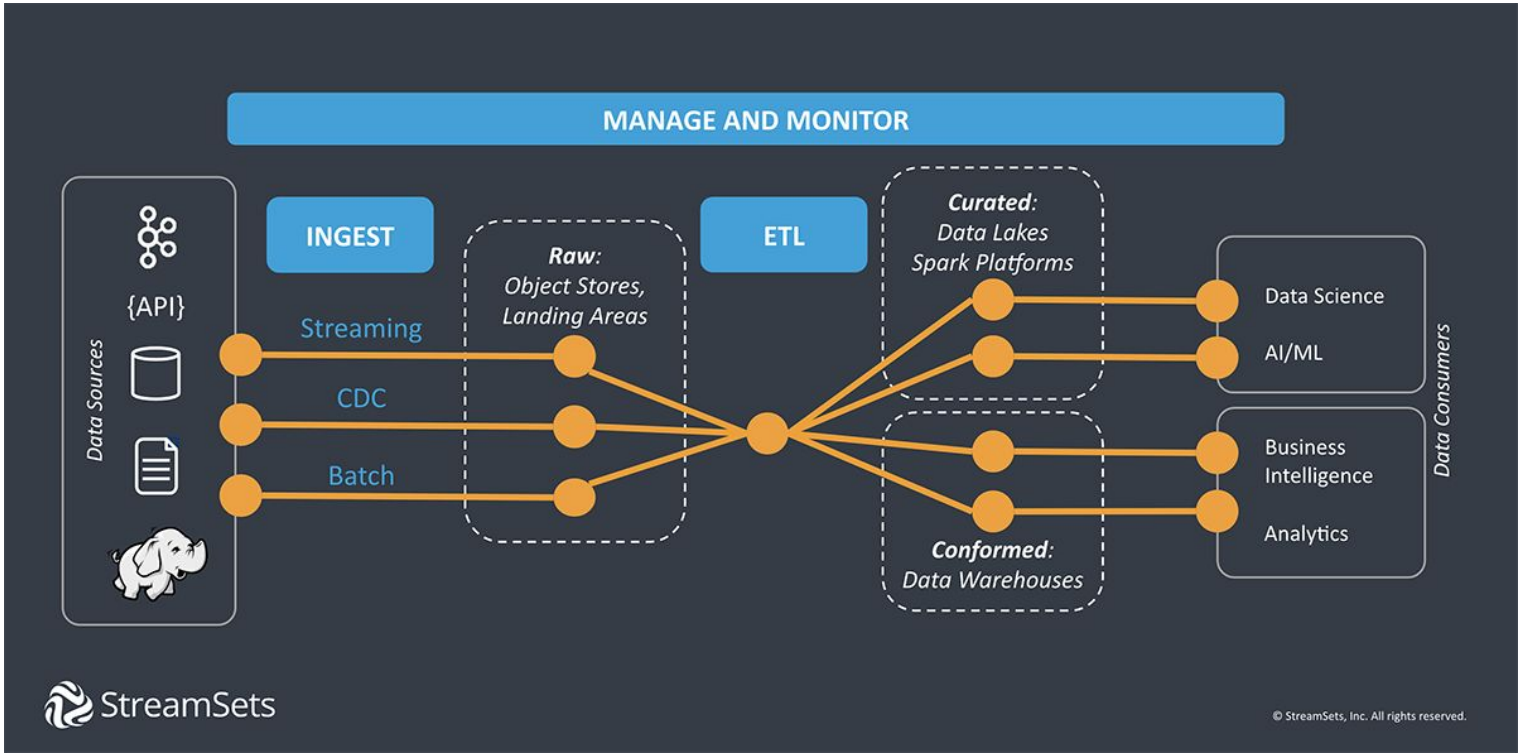
¿Qué es?

- Rama de la ingeniería de software dedicada al desarrollo y mantenimiento de software para transmisión, procesamiento, almacenamiento y acceso de los datos.

- El rol de los *data engineers* en una organización suele ser proveer al resto de los miembros de plataformas en donde puedan acceder a los datos de forma confiable y eficiente.

Data Engineering

¿Qué es?



Data Lakes & Data Warehouses

- Aplicaciones que funcionan basadas en una BD relacional suelen ejecutar muchas consultas tanto de escritura como de lectura, pero de baja complejidad. Se hacen pocos joins y agregaciones.

- Pero miembros de la organización que quieran analizar lo que ocurra en el sistema harán consultas con mucha agregación y joins. Cosas del tipo “Cuánto ha vendido en promedio cada tienda cada mes por los últimos 12 meses”

- Estas consultas son muy costosas para el DBMS y pueden afectar las disponibilidad de la aplicación para los usuarios reales!

Data Lakes & Data Warehouses

- A este tipo de consultas generalmente se les llama *On Line Analytic Processing* **(OLAP)**, y queremos que se hagan en otro lado.

- Para eso se generan copias de la(s) bases de datos del sistema, ya sea tal cual (*Data lake*) o pre procesada para su uso de una manera particular (*Data warehouse)*.

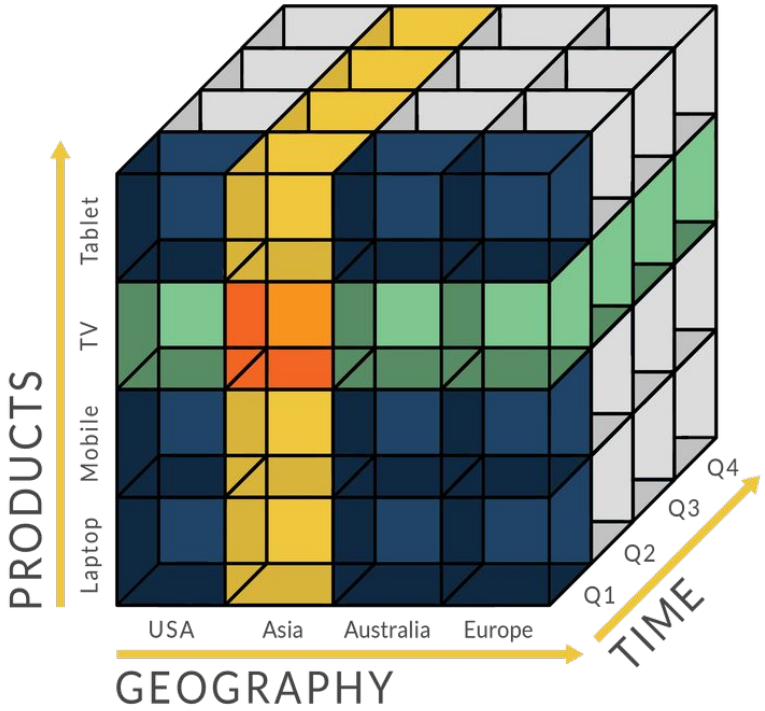
- Estas copias no suelen estar actualizadas en tiempo real, pero se pueden consultar libremente y suelen mantener toda la información histórica del sistema (incluso aquella que se deprecó de la BD original)

Data Warehouses

OLAP Cube

- Un warehouse suele tener los datos preprocesados de alguna forma que facilite las consultas que se van a hacer en él.

- Una estrategia clásica es guardar los datos de forma “multidimensional”.



Data Warehouses

Columnar Databases

- Tablas que guardan los datos en disco columna a columna y no fila por fila.
- Se benefician de una alta paralelización (la nube lo hace solo).
- Se consultan con SQL.
- Algunas se pueden particionar para acelerar consultas específicas.

ETLs

- Del inglés: *extract, transform, load*. Extraer, transformar y cargar.
- Son scripts o tareas que toman datos desde una o más fuentes, la procesan y transforman según sea necesario y finalmente la guardan en otro lugar (usualmente un warehouse).
- Los ETLs suelen programarse para ejecutarse de manera periódica (cada 1 hora, 1 día, etc). En algunos casos se hacen *event-driven,* para procesar datos que vengan de un *stream.*
- Son una parte muy importante de cómo se procesan los datos en una organización.

Paradigma Map Reduce

Computación Distribuida

- Datos tan grandes que no caben en un computador.
- Debemos repartirlos en varios servidores.
- Cada servidor no conoce lo que tiene el resto.
- No nos podemos dar el lujo de comunicar todos los datos por la red.

Map Reduce

- Algoritmo eficiente para computación paralela/distribuida.
- Comúnmente usado para el procesamiento y generación de set de datos grandes.
- El procesamiento sigue una estructura restringida.

Map Reduce

Ejemplo: ver cuántas veces ocurre cada palabra en un texto T

¿Cómo hacer esto en un archivo de 100 petabytes?

Map Reduce

El proceso se puede entender en 3 pasos:

- **Map**: recibe datos y genera pares key – values
- **Shuffle**: transfiere los datos desde mappers a reducers
- **Reduce**: recibe pares con el mismo key y los agrega

Map Reduce

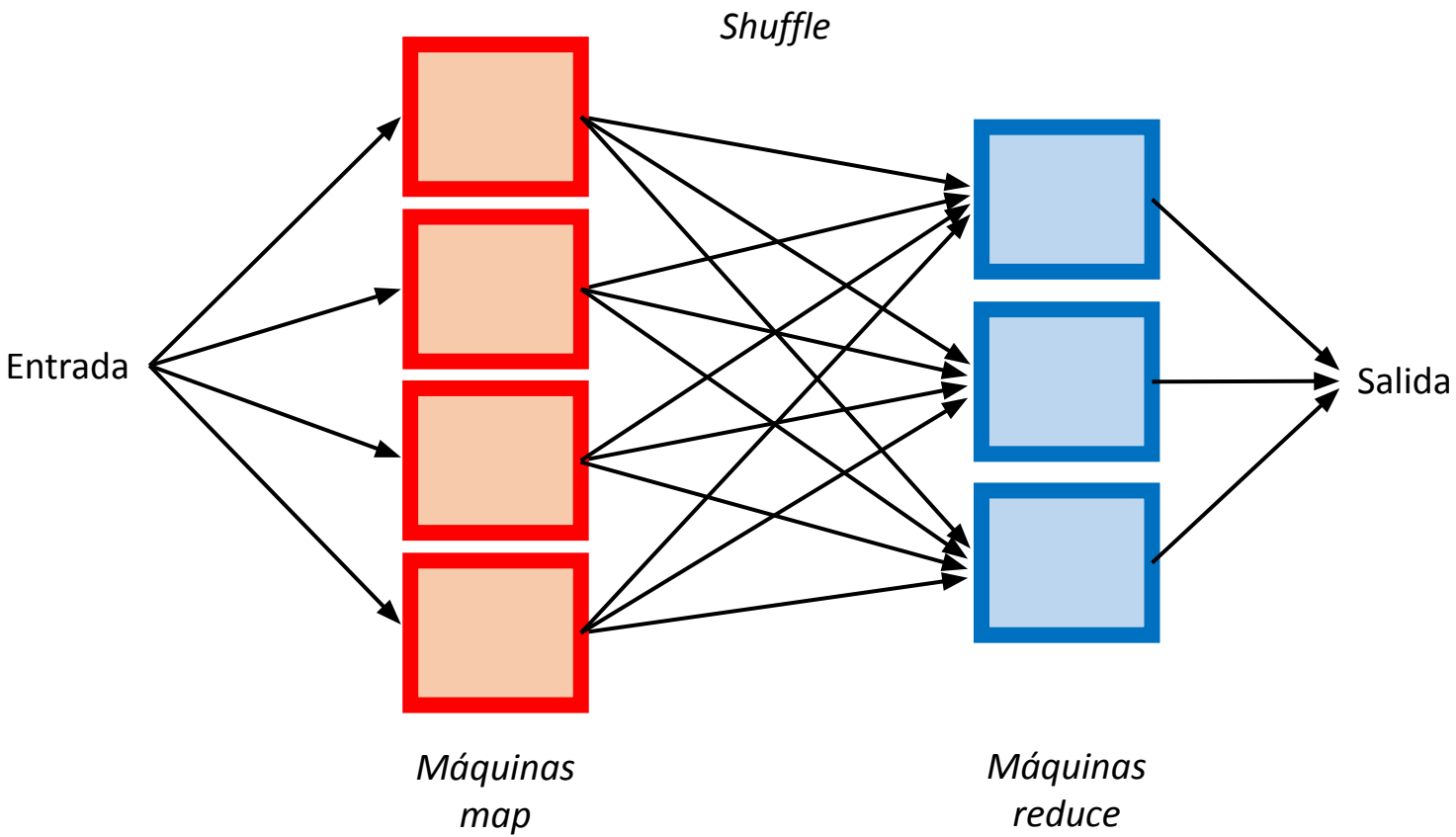
Arquitectura

Mappers:

- Nodos encargados de hacer Map
- Reciben parte del documento y lo envían a los reducers (a través de shuffle)

Reducers:

- Nodos encargados de hacer Reduce
- Reciben los Map y los agregan
- El output es la unión de cada Reducer


Shuffle
Entrada
Salida
Máquinas map
Máquinas reduce

Map Reduce

Ejemplo

¿Cuántas veces ocurre cada palabra en un archivo de texto grande?

Map Reduce

Ejemplo

Map:

- Recibe un pedazo de texto
- Por cada palabra, emite el par (palabra, número de ocurrencias)

Reduce:

- Cada reduce recibe todos los pares asociados a la misma palabra
- Junta todos estos pares y suma las ocurrencias

INPUT

hola que hola año zzz hola que zzz que

Máquina map1

Máquina map2

Máquina map3

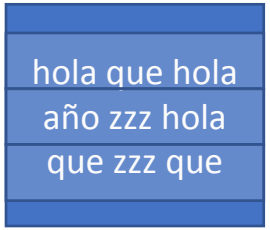
Máquina reduce1

Palabras A-M

Máquina reduce2

Palabras N-Z

SEPARAR INPUT



Máquina map1

Máquina map2

Máquina map3

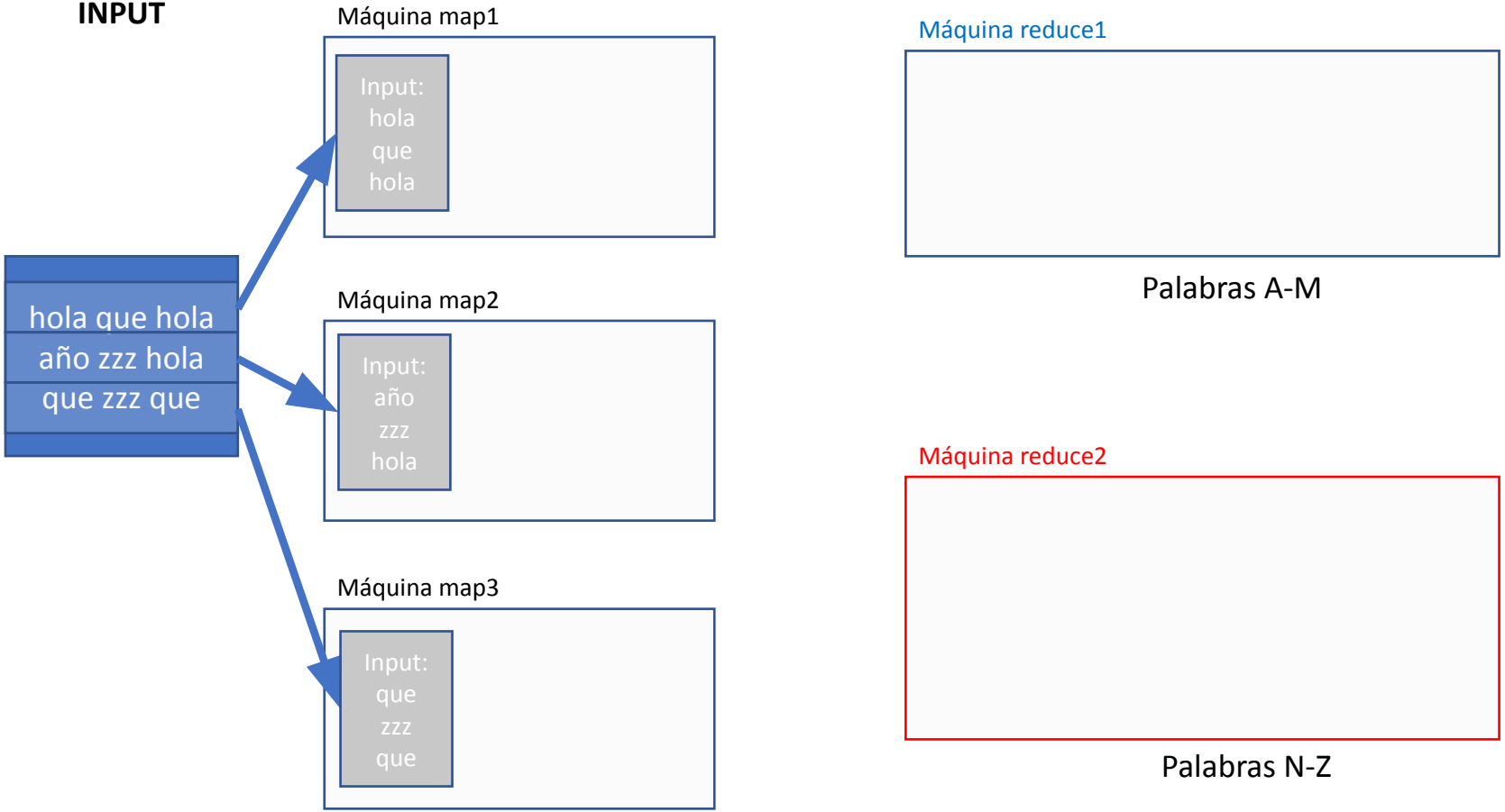
Máquina reduce1

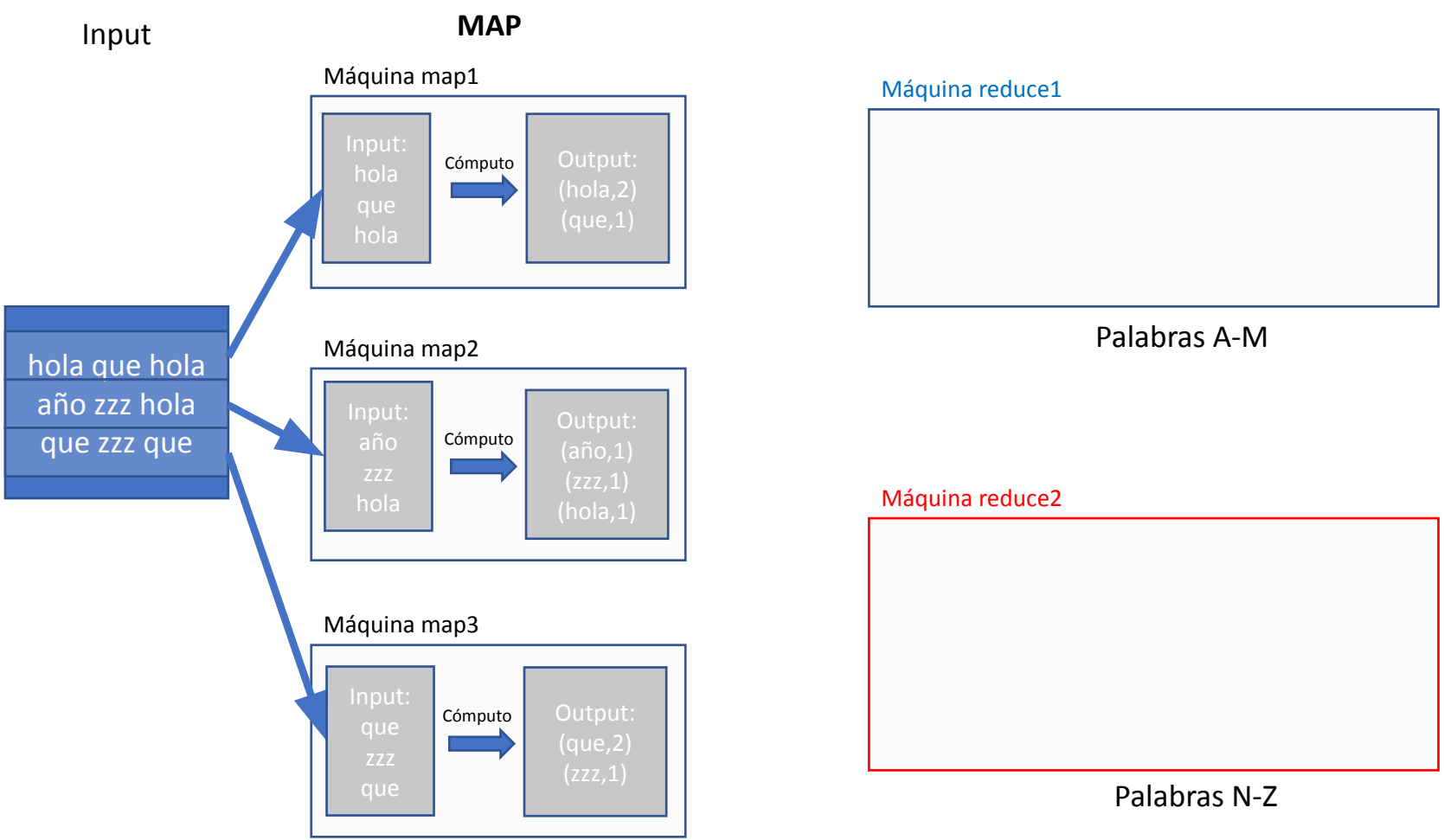
Palabras A-M

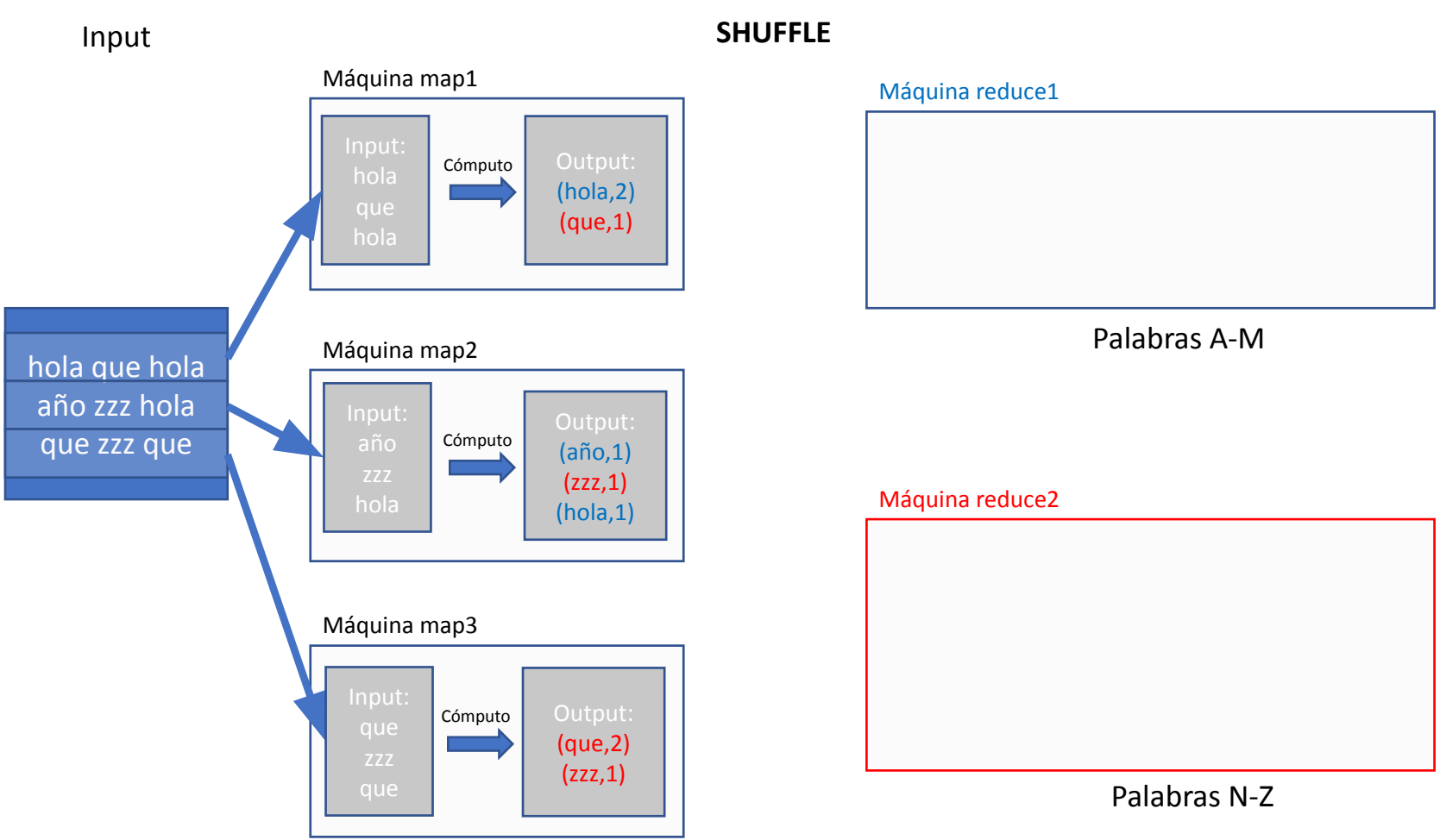
Máquina reduce2

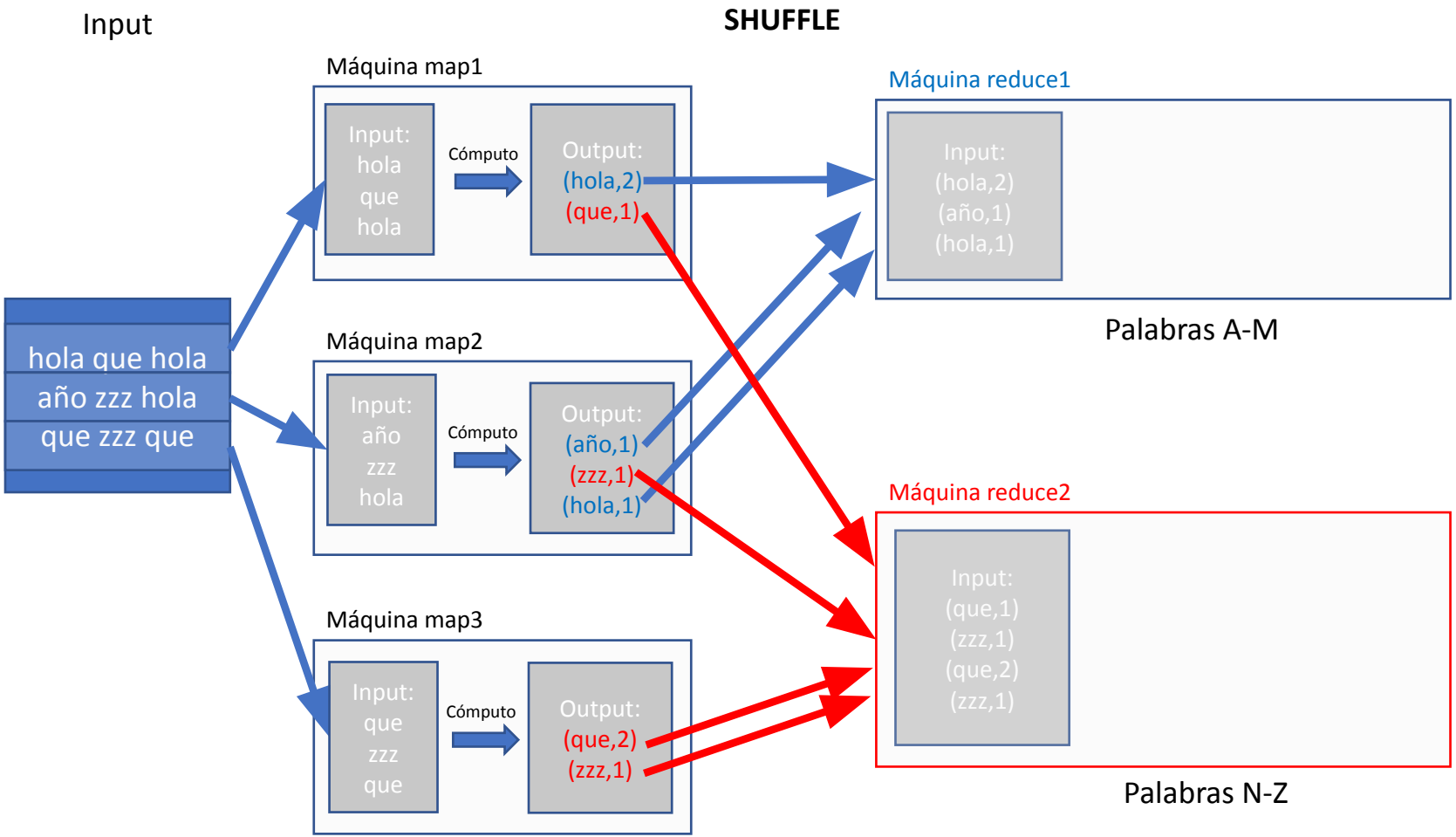
Palabras N-Z

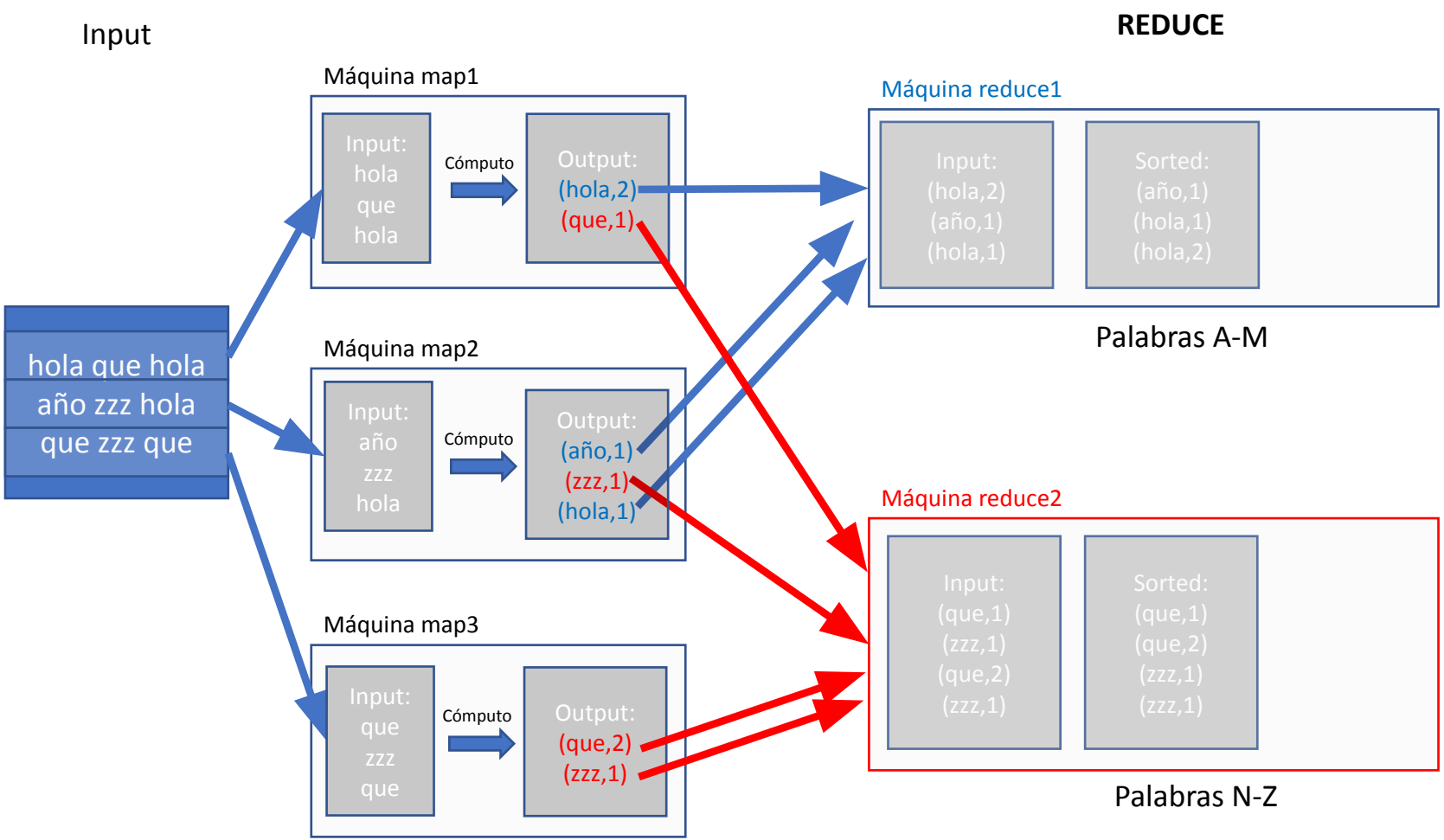
SEPARAR INPUT

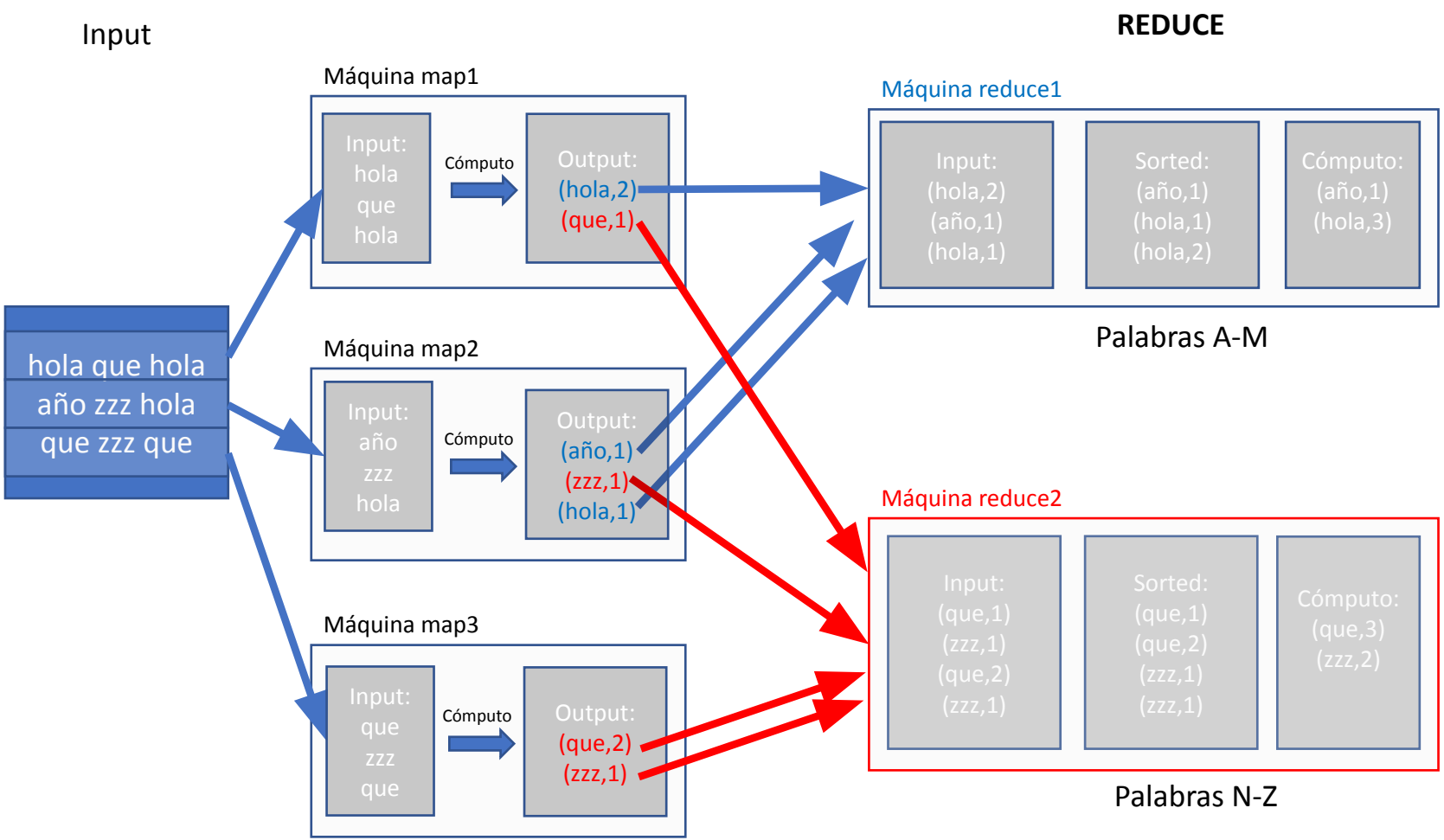


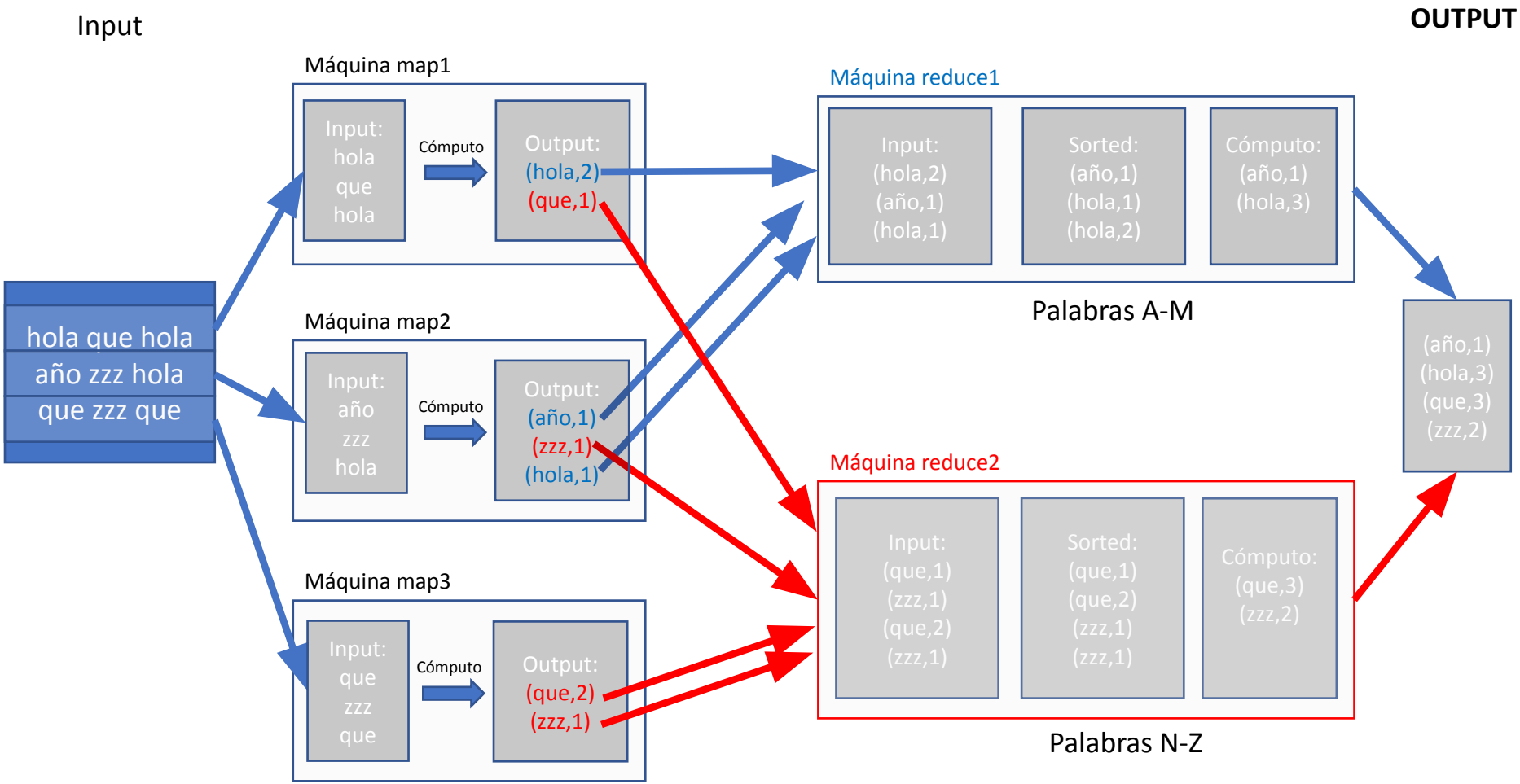

Input
MAP
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
hola que hola
año zzz hola
que zzz que
Máquina reduce1
Palabras A-M
Máquina reduce2
Palabras N-Z

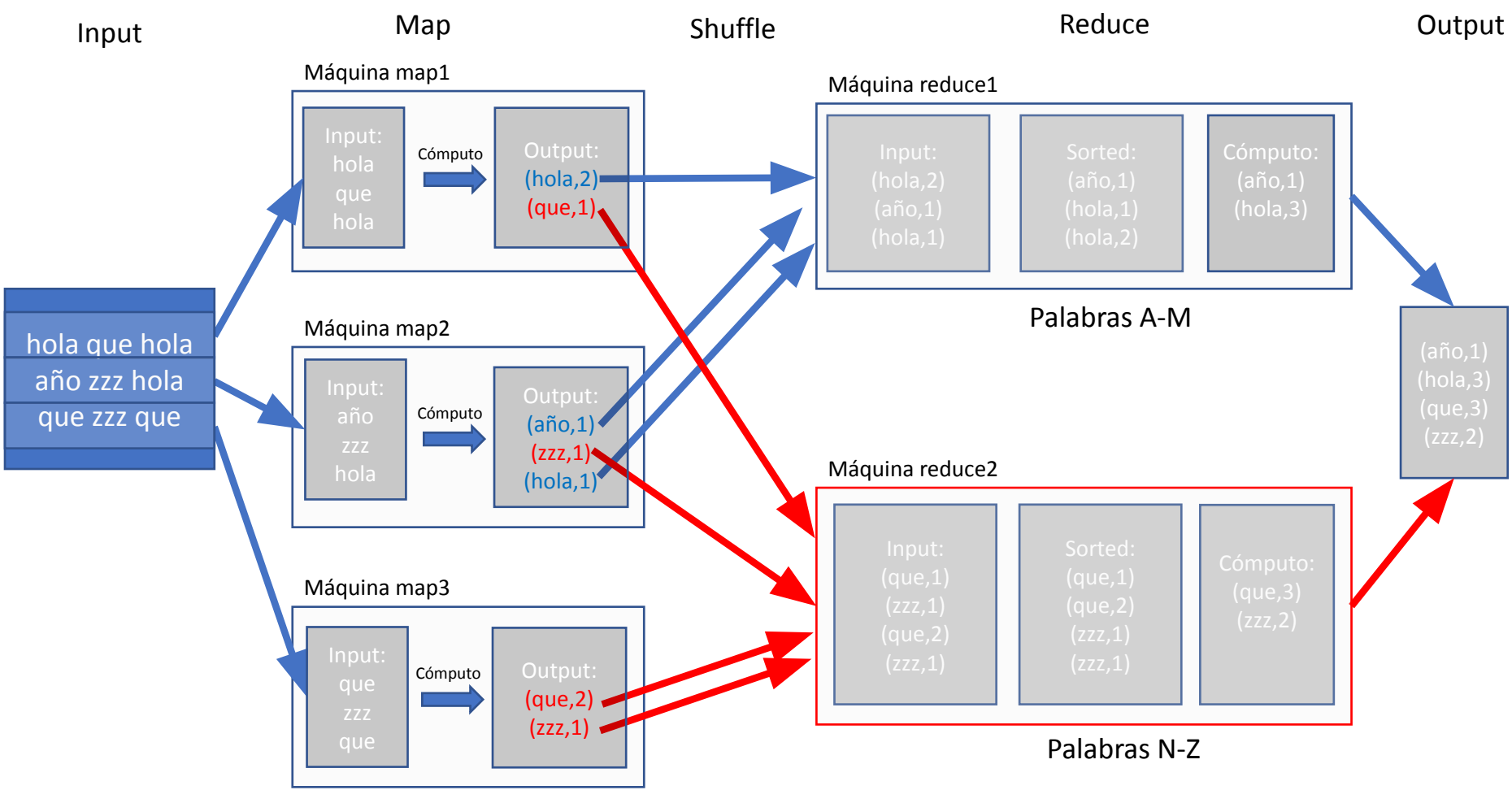

Input
SHUFFLE
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
hola que hola
año zzz hola
que zzz que
Máquina reduce1
Palabras A-M
Máquina reduce2
Palabras N-Z


Input
SHUFFLE
hola que hola
año zzz hola
que zzz que
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
Máquina reduce1
Input:
(hola,2)
(año,1)
(hola,1)
Palabras A-M
Máquina reduce2
Input:
(que,1)
(zzz,1)
(que,2)
(zzz,1)
Palabras N-Z


Input
REDUCE
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
hola que hola
año zzz hola
que zzz que
Máquina reduce1
Input:
(hola,2)
(año,1)
(hola,1)
Sorted:
(año,1)
(hola,1)
(hola,2)
Palabras A-M
Máquina reduce2
Input:
(que,1)
(zzz,1)
(que,2)
(zzz,1)
Sorted:
(que,1)
(que,2)
(zzz,1)
(zzz,1)
Palabras N-Z


Input
REDUCE
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
hola que hola
año zzz hola
que zzz que
Máquina reduce1
Input:
(hola,2)
(año,1)
(hola,1)
Sorted:
(año,1)
(hola,1)
(hola,2)
Cómputo:
(año,1)
(hola,3)
Palabras A-M
Máquina reduce2
Input:
(que,1)
(zzz,1)
(que,2)
(zzz,1)
Sorted:
(que,1)
(que,2)
(zzz,1)
(zzz,1)
Cómputo:
(que,3)
(zzz,2)
Palabras N-Z


Input
OUTPUT
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
hola que hola
año zzz hola
que zzz que
Máquina reduce1
Input:
(hola,2)
(año,1)
(hola,1)
Sorted:
(año,1)
(hola,1)
(hola,2)
Cómputo:
(año,1)
(hola,3)
Palabras A-M
Máquina reduce2
Input:
(que,1)
(zzz,1)
(que,2)
(zzz,1)
Sorted:
(que,1)
(que,2)
(zzz,1)
(zzz,1)
Cómputo:
(que,3)
(zzz,2)
Palabras N-Z
(año,1)
(hola,3)
(que,3)
(zzz,2)


Input
Map
Shuffle
Reduce
Output
hola que hola
año zzz hola
que zzz que
Máquina map1
Input:
hola
que
hola
Cómputo
Output:
(hola,2)
(que,1)
Máquina map2
Input:
año
zzz
hola
Cómputo
Output:
(año,1)
(zzz,1)
(hola,1)
Máquina map3
Input:
que
zzz
que
Cómputo
Output:
(que,2)
(zzz,1)
Máquina reduce1
Input:
(hola,2)
(año,1)
(hola,1)
Sorted:
(año,1)
(hola,1)
(hola,2)
Cómputo:
(año,1)
(hola,3)
Palabras A-M
Máquina reduce2
Input:
(que,1)
(zzz,1)
(que,2)
(zzz,1)
Sorted:
(que,1)
(que,2)
(zzz,1)
(zzz,1)
Cómputo:
(que,3)
(zzz,2)
Palabras N-Z
(año,1)
(hola,3)
(que,3)
(zzz,2)

Map Reduce

Ejemplo: Join

¿Cómo hago un join con Map Reduce?

- Input: archivo con el nombre de la tabla y sus tuplas
- Map: Agrupo por el atributo que hace el join
- Reduce: Hago el producto cruz para las tablas distintas

INPUT

R

A	B
1	1
1	3
3	2
3	3

S

B	C
2	4
2	7
3	8
3	9

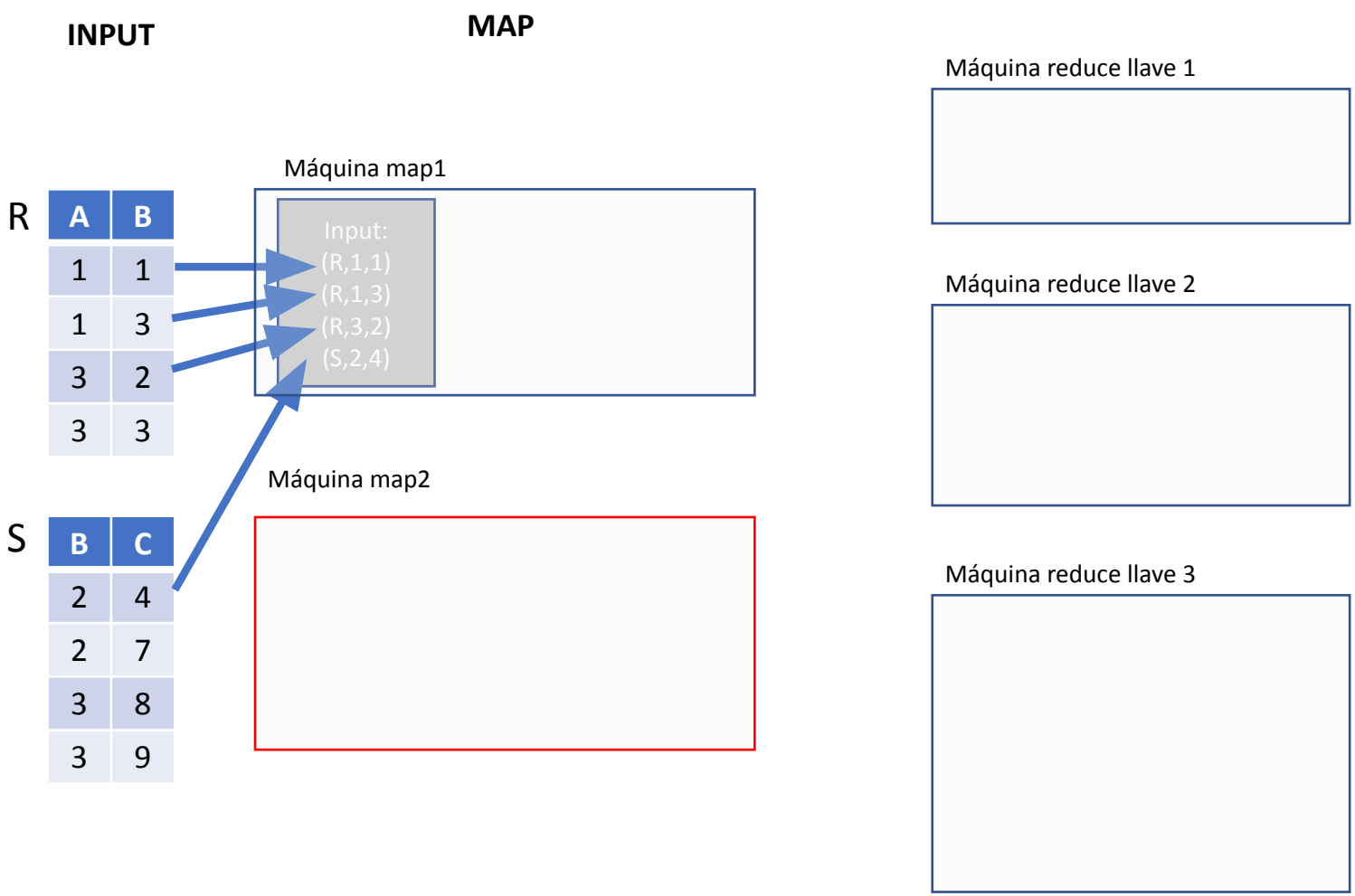
Máquina map1

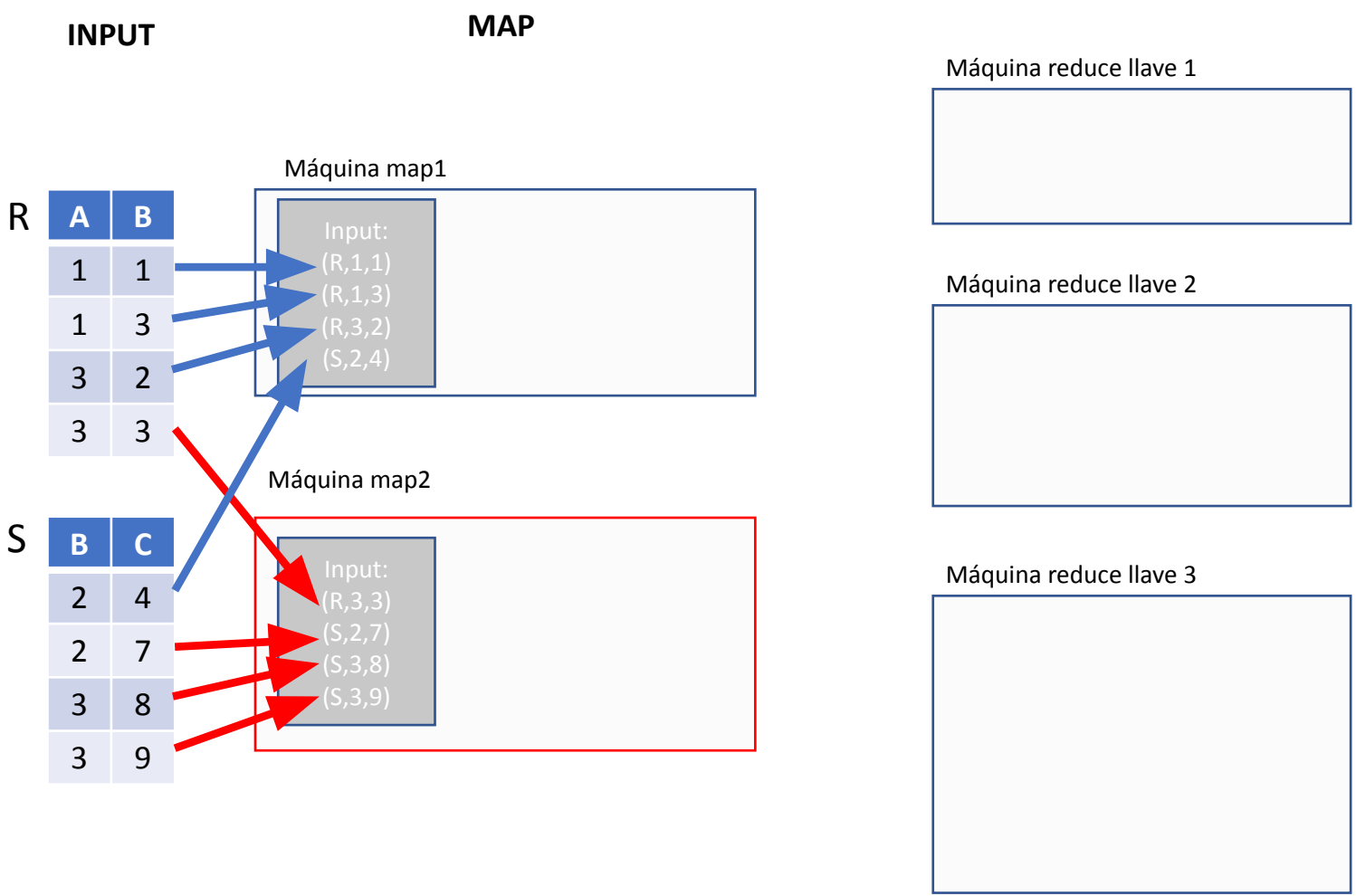
Máquina map2

Máquina reduce llave 1

Máquina reduce llave 2

Máquina reduce llave 3


INPUT
MAP
Máquina map1
R
A
B
1
1
1
3
3
2
3
3
Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)
Máquina map2
S
B
C
2
4
2
7
3
8
3
9
Máquina reduce llave 1
Máquina reduce llave 2
Máquina reduce llave 3


INPUT
MAP
R
A
B
1
1
1
3
3
2
3
3
S
B
C
2
4
2
7
3
8
3
9
Máquina map1
Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)
Máquina map2
Input:
(R,3,3)
(S,2,7)
(S,3,8)
(S,3,9)
Máquina reduce llave 1
Máquina reduce llave 2
Máquina reduce llave 3

INPUT

MAP

R

A	B
1	1
1	3
3	2
3	3

S

B	C
2	4
2	7
3	8
3	9

Máquina map1

Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)

Cómputo

Output:
1: (R,1)
3: (R,1)
2: (R,3)
2: (S,4)

Máquina map2

Input:
(R,3,3)
(S,2,7)
(S,3,8)
(S,3,9)

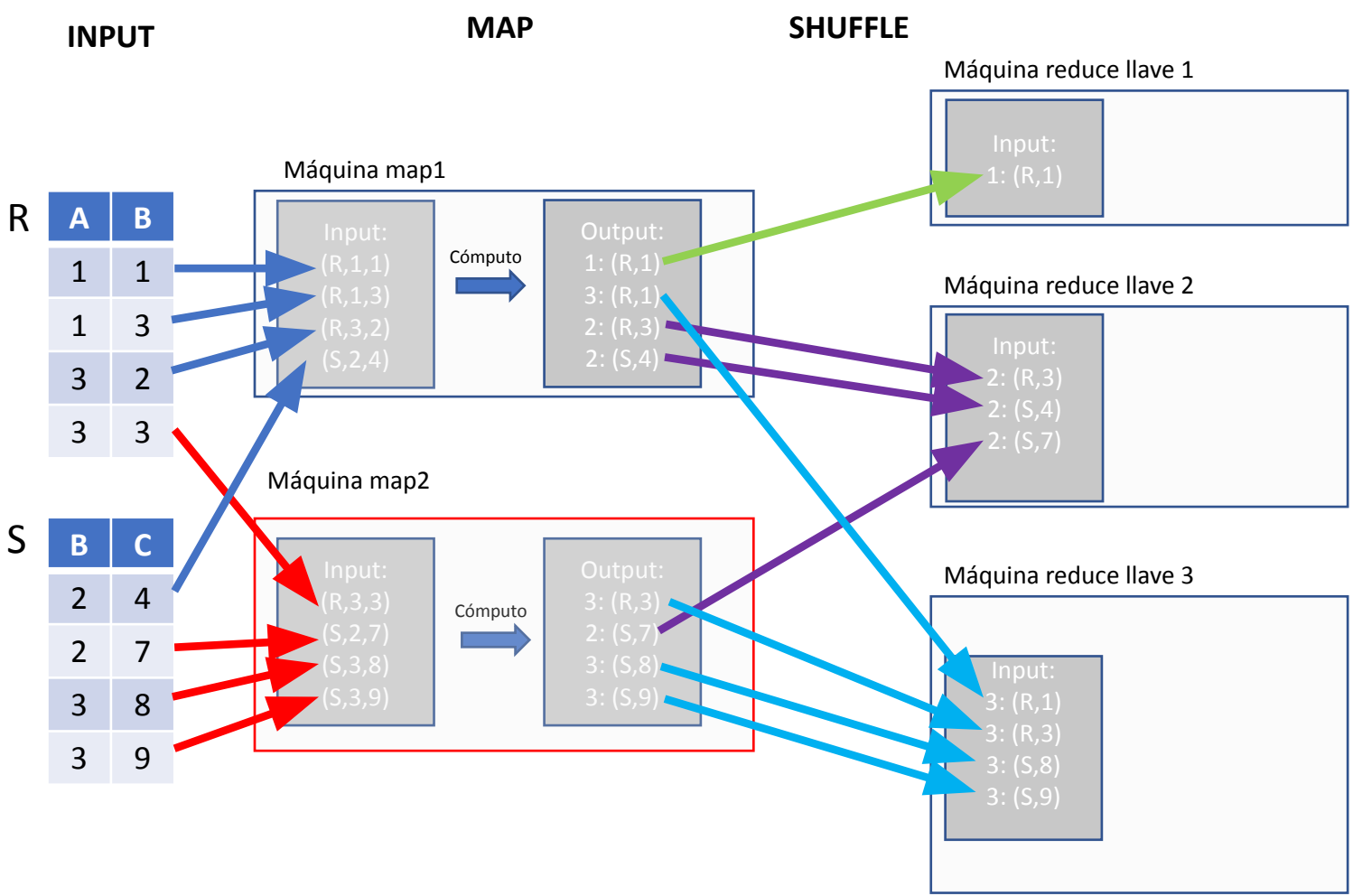
Cómputo

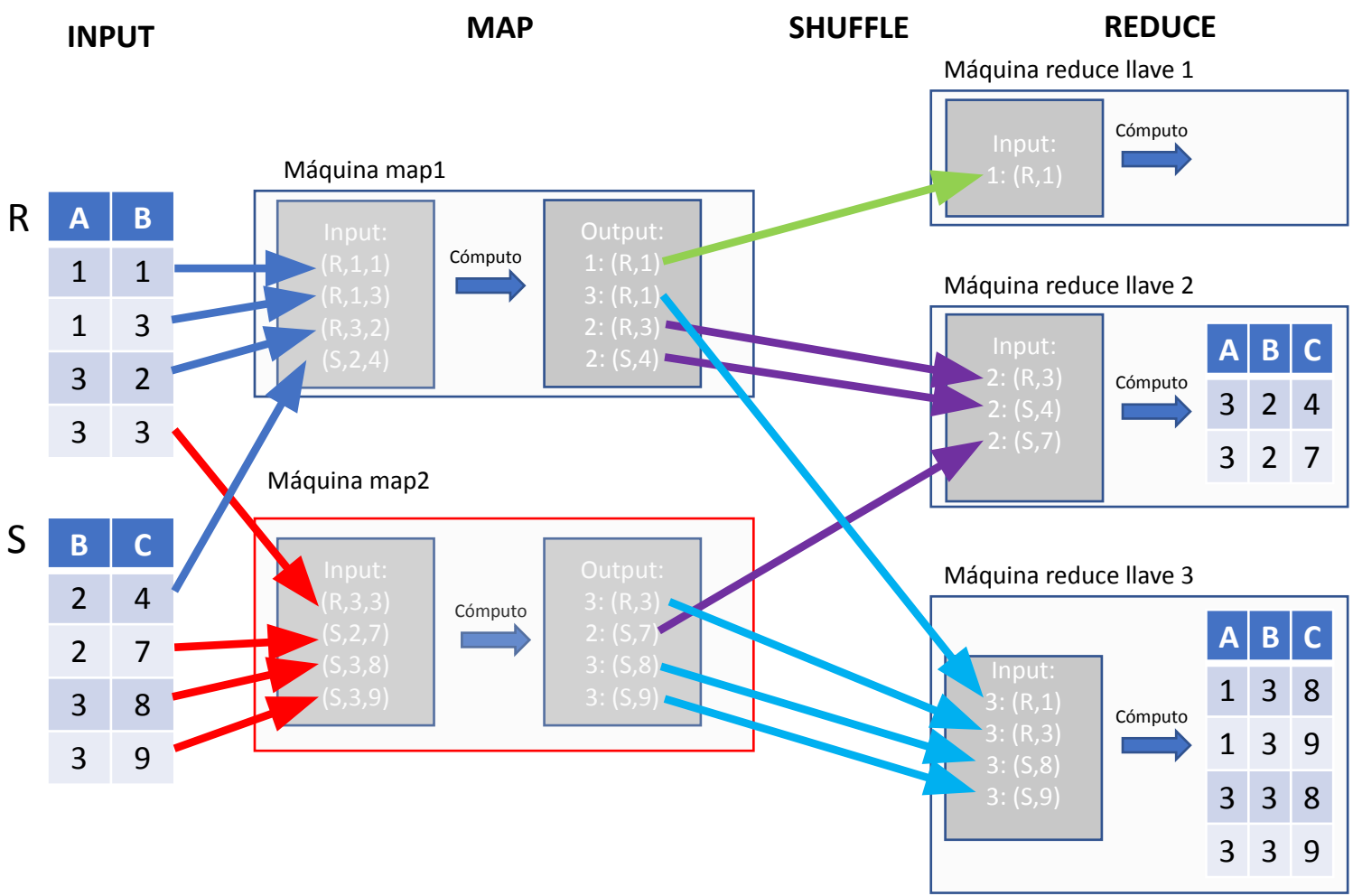
Output:
3: (R,3)
2: (S,7)
3: (S,8)
3: (S,9)

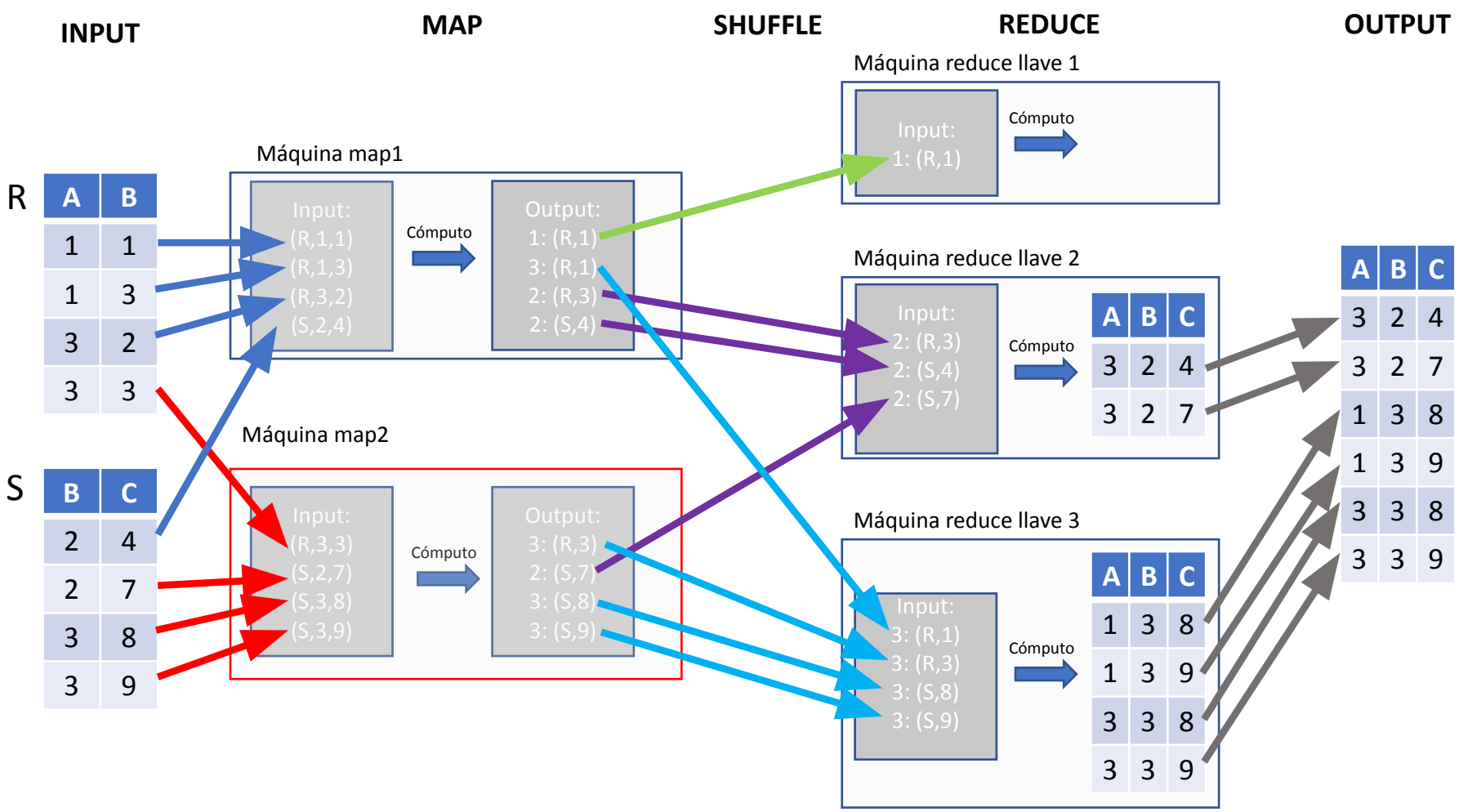
Máquina reduce llave 1

Máquina reduce llave 2

Máquina reduce llave 3


INPUT
MAP
SHUFFLE
R
A
B
1
1
1
3
3
2
3
3
S
B
C
2
4
2
7
3
8
3
9
Máquina map1
Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)
Cómputo
Output:
1: (R,1)
3: (R,1)
2: (R,3)
2: (S,4)
Máquina map2
Input:
(R,3,3)
(S,2,7)
(S,3,8)
(S,3,9)
Cómputo
Output:
3: (R,3)
2: (S,7)
3: (S,8)
3: (S,9)
Máquina reduce llave 1
Input:
1: (R,1)
Máquina reduce llave 2
Input:
2: (R,3)
2: (S,4)
2: (S,7)
Máquina reduce llave 3
Input:
3: (R,1)
3: (R,3)
3: (S,8)
3: (S,9)


INPUT
MAP
SHUFFLE
REDUCE
R
A B
1 1
1 3
3 2
3 3
S
B C
2 4
2 7
3 8
3 9
Máquina map1
Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)
Cómputo
Output:
1: (R,1)
3: (R,1)
2: (R,3)
2: (S,4)
Máquina map2
Input:
(R,3,3)
(S,2,7)
(S,3,8)
(S,3,9)
Cómputo
Output:
3: (R,3)
2: (S,7)
3: (S,8)
3: (S,9)
Máquina reduce llave 1
Input:
1: (R,1)
Cómputo
Máquina reduce llave 2
Input:
2: (R,3)
2: (S,4)
2: (S,7)
Cómputo
A B C
3 2 4
3 2 7
Máquina reduce llave 3
Input:
3: (R,1)
3: (R,3)
3: (S,8)
3: (S,9)
Cómputo
A B C
1 3 8
1 3 9
3 3 8
3 3 9


INPUT
MAP
SHUFFLE
REDUCE
OUTPUT
R
A B
1 1
1 3
3 2
3 3
S
B C
2 4
2 7
3 8
3 9
Máquina map1
Input:
(R,1,1)
(R,1,3)
(R,3,2)
(S,2,4)
Cómputo
Output:
1: (R,1)
3: (R,1)
2: (R,3)
2: (S,4)
Máquina map2
Input:
(R,3,3)
(S,2,7)
(S,3,8)
(S,3,9)
Cómputo
Output:
3: (R,3)
2: (S,7)
3: (S,8)
3: (S,9)
Máquina reduce llave 1
Input:
1: (R,1)
Cómputo
Máquina reduce llave 2
Input:
2: (R,3)
2: (S,4)
2: (S,7)
Cómputo
A B C
3 2 4
3 2 7
Máquina reduce llave 3
Input:
3: (R,1)
3: (R,3)
3: (S,8)
3: (S,9)
Cómputo
A B C
1 3 8
1 3 9
3 3 8
3 3 9
A B C
3 2 4
3 2 7
1 3 8
1 3 9
3 3 8
3 3 9

Data Engineering en tiempos de la nube

- Para la implementación de ETLs, data warehouses, lakes, streams de datos, etc. existen herramientas tanto open source como de proveedores cloud (pagadas).

- El trabajo de data engineering en una organización grande suele consistir en combinar las herramientas open source con los servicios en la nube para generar *pipelines* para el procesamiento, guardado y uso de los datos.

Ejemplos open source

Plataforma para streaming distribuido de datos.

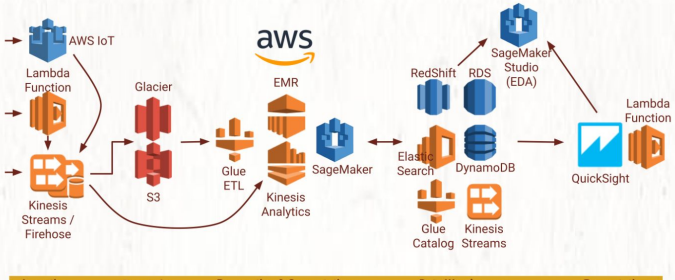
Procesamiento distribuido de datos (como map reduce)

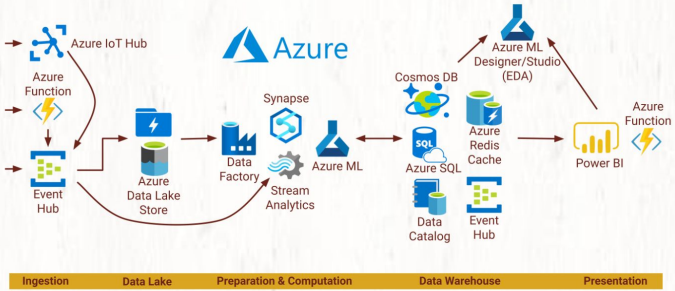
Orquestación y ejecución de ETLs.

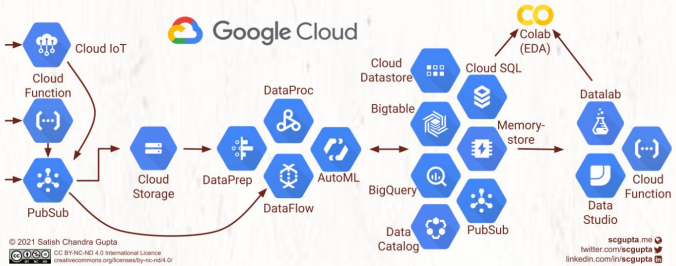
Procesamiento y analytics para datasets grandes.

Big Data Pipelines on AWS, Microsoft Azure, and GCP

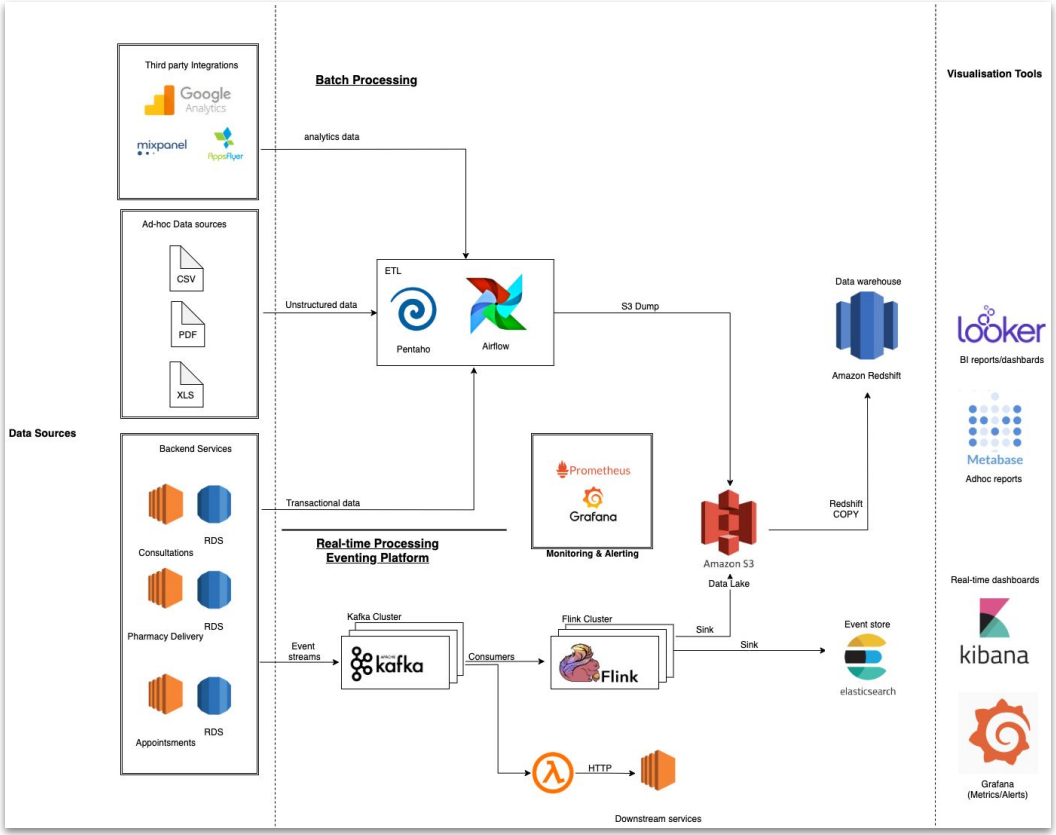
scgupta.link/big-data-pipeline







Pipeline real:



Links para los curiosos

- https://www.intermix.io/blog/14-data-pipelines-amazon-redshift/
- https://www.freecodecamp.org/news/scalable-data-analytics-pipeline/
- https://aws.amazon.com/blogs/big-data/aws-serverless-data-analytics-pipeline-reference-architecture/
- https://www.slideshare.net/legoboku/building-a-data-pipeline-using-apache-airflow-on-aws-gcp