Lab 8 RDDS

- Repositorio

Librerias

In [13]:
```
from pyspark.sql import SparkSession
import nltk
from nltk.corpus import stopwords
nltk.download("stopwords", quiet=True) # Descargar stopwords si no están descargada
# docker compose up -d
```

Out[13]: True

Spark Session

In [14]:
```
spark = SparkSession.builder.appName("MiApp").master("local[*]").getOrCreate()
sc = spark.sparkContext
working_dir = "/opt/app/working_dir/"
rdd = sc.textFile(working_dir + "constitution.txt")
```

Carga de RDD y exploración

In [15]:
```
primeras_lineas = rdd.take(3)
num_lineas = rdd.count()

print("=== Exploración inicial ===")
for i, l in enumerate(primeras_lineas, 1):
    print(f"Línea {i}: {l}")
print(f"\nTotal de líneas en el documento: {num_lineas:,}")
```

```
=== Exploración inicial ===
Línea 1: We the People of the United States, in Order to form a more perfect
Línea 2: Union, establish Justice, insure domestic Tranquility, provide for the
Línea 3: common defence, promote the general Welfare, and secure the Blessings of

Total de líneas en el documento: 649
```

Word Count

In [16]:
```
splitted_lines = rdd.map(lambda line: line.split(' '))
print("\nEjemplo con map (3 elementos):")
print(splitted_lines.take(3))

# Pipeline de limpieza + normalización
```

```
words_rdd = (
    rdd.flatMap(lambda line: line.strip().split(' '))
       .map(lambda w: w.strip())
       .filter(lambda w: w != '' and w.isalnum())
       .map(lambda w: w.lower())
)

num_palabras = words_rdd.count()
print(f"\nTotal de 'tokens' limpios: {num_palabras:,}")

# Palabra más larga (reduce)
mas_larga = words_rdd.reduce(lambda a, b: a if len(a) > len(b) else b)
print(f"Palabra más larga: '{mas_larga}'")
```

```
Ejemplo con map (3 elementos):
[['We', 'the', 'People', 'of', 'the', 'United', 'States,', 'in', 'Order', 'to', 'for
m', 'a', 'more', 'perfect', ''], ['Union,', 'establish', 'Justice,', 'insure', 'dome
stic', 'Tranquility,', 'provide', 'for', 'the', ''], ['common', 'defence,', 'promot
e', 'the', 'general', 'Welfare,', 'and', 'secure', 'the', 'Blessings', 'of', '']]

Total de 'tokens' limpios: 6,701
Palabra más larga: 'constitutionally'
```

Conteos y Top

In [17]:
```
keyval_rdd = words_rdd.map(lambda w: (w, 1))
wordcount = keyval_rdd.reduceByKey(lambda a, b: a + b)

top5 = (wordcount
        .map(lambda kv: (kv[1], kv[0]))
        .sortByKey(ascending=False)
        .take(5))

print("\nTop 5 (incluyendo stopwords):")
for rank, (freq, word) in enumerate(top5, 1):
    print(f"{rank}. {word} -> {freq}")
```

```
Top 5 (incluyendo stopwords):
1. the -> 726
2. of -> 493
3. shall -> 293
4. and -> 262
5. to -> 201
```

Top-N sin stopwords

In [18]:
```
# Stopwords base + términos frecuentes del dominio legal/constitucional
sw = set(stopwords.words("english"))
domain_sw = {"shall", "section", "sections", "article", "articles", "state", "state
stopwords_all = sw.union(domain_sw)

# Filtrar y obtener top 5 sin stopwords
top5_no_stop = (
    wordcount
```

```
    .filter(lambda kv: kv[0] not in stopwords_all)
    .map(lambda kv: (kv[1], kv[0]))
    .sortByKey(ascending=False)
    .take(5)
)

print("Top 5 sin stopwords:")
for i, (freq, word) in enumerate(top5_no_stop, 1):
    print(f"{i}. {word} -> {freq}")
```

```
Top 5 sin stopwords:
1. united -> 85
2. president -> 72
3. may -> 42
4. congress -> 39
5. amendment -> 34
```

In [19]:
```
spark.stop()
```

Discusión y Conclusiones

Se observó que el documento contenía **649 líneas**.

Al aplicar transformaciones con `map` y `flatMap`, se pudo observar la diferencia entre ambas operaciones: `map` devolvió listas anidadas, por lo que un `splitted_lines.count()` solo devolvería el número de **líneas** y no el total de palabras, lo cual no cumplía con nuestro propósito de análisis léxico. En cambio, `flatMap` permitió generar un flujo continuo de palabras, haciendo posible un conteo real de tokens en el texto.

Posteriormente, se aplicó un proceso de limpieza eliminando cadenas vacías, normalizando todas las palabras a minúsculas y utilizando el método `.isalnum()` para asegurar que el RDD `words_rdd` contuviera únicamente palabras formadas por caracteres alfanuméricos. Gracias a esto se contabilizaron **6,701 palabras**.

Se aplicó la operación `reduce` para identificar la palabra más larga, encontrándose **"constitutionally"** como resultado.

Se construyó un RDD de pares clave-valor y se calcularon las frecuencias. Al ordenar los resultados se obtuvo el **Top 5 incluyendo stopwords**, conformado por:

1. *the* (726)
2. *of* (493)
3. *shall* (293)
4. *and* (262)
5. *to* (201)

Pregunta: Muestre las 5 palabras más repetidas excluyendo las stopwords.

Para dar respuesta, se aplicó un filtro adicional con las stopwords provistas por NLTK, ampliadas con términos del dominio constitucional como *shall*, *section* o *article*. Tras este proceso, el **Top 5 sin stopwords** fue:

1. *united* (85)
2. *president* (72)
3. *may* (42)
4. *congress* (39)
5. *amendment* (34)