```
# Instalar SDK Java 8

!apt-get install openjdk-8-jdk-headless -qq > /dev/null

# Descargar Spark 3.2.2

!wget -q https://archive.apache.org/dist/spark/spark-3.2.3/spark-3.2.3-bin-hadoop3.2.tgz

# Descomprimir el archivo descargado de Spark

!tar xf spark-3.2.3-bin-hadoop3.2.tgz

# Establecer las variables de entorno

import os

os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.2.3-bin-hadoop3.2"

# Instalar la librería findspark

!pip install -q findspark

# Instalar pyspark

!pip install -q pyspark
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 281.4/281.4 MB 4.9 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 199.7/199.7 KB 18.4 MB/s eta 0:00:00
  Building wheel for pyspark (setup.py) ... done
```

```
# Explorando los datos


import findspark
findspark.init()
from pyspark.sql import SparkSession

spark = SparkSession.builder.getOrCreate()

df = spark.read.parquet('./data/dataframe.parquet')


df.printSchema()

df.show(20, truncate=False)
```

```
root
 |-- nombre: string (nullable = true)
 |-- color: string (nullable = true)
 |-- cantidad: long (nullable = true)

+------+-----+--------+
|nombre|color|cantidad|
+------+-----+--------+
|Jose  |azul |1900    |
|null  |null |1700    |
|null  |rojo |1300    |
|Juan  |rojo |1500    |
+------+-----+--------+
```

```
# Funciones count, countDistinct y approx_count_distinct

df = spark.read.parquet('./data/dataframe.parquet')

df.printSchema()

df.show()
```

```
root
 |-- nombre: string (nullable = true)
 |-- color: string (nullable = true)
 |-- cantidad: long (nullable = true)

+------+-----+--------+
|nombre|color|cantidad|
+------+-----+--------+
|  Jose| azul|    1900|
|  null| null|    1700|
|  null| rojo|    1300|
```

```
|  Juan| rojo|    1500|
+------+-----+--------+
```

```
# count

from pyspark.sql.functions import count

df.select(
    count('nombre').alias('conteo_nombre'),
    count('color').alias('conteo_color')
).show()

#los valores null no se cuentan
```

```
+-------------+------------+
|conteo_nombre|conteo_color|
+-------------+------------+
|            2|           3|
+-------------+------------+
```

```
df.select(
    count('nombre').alias('conteo_nombre'),
    count('color').alias('conteo_color'),
    count('*').alias('conteo_general')
).show()
```

```
+-------------+------------+--------------+
|conteo_nombre|conteo_color|conteo_general|
+-------------+------------+--------------+
|            2|           3|             4|
+-------------+------------+--------------+
```

```
# countDistinct

from pyspark.sql.functions import countDistinct

df.select(
    countDistinct('color').alias('colores_dif')
).show()

#los valores null no se cuentan
```

```
+-----------+
|colores_dif|
+-----------+
|          2|
+-----------+
```

```
# approx_count_distinct

from pyspark.sql.functions import approx_count_distinct
dataframe = spark.read.parquet('./data/vuelos.parquet')

dataframe.printSchema()

dataframe.select(
    countDistinct('AIRLINE'),
    approx_count_distinct('AIRLINE')
).show()
```

```
root
 |-- YEAR: integer (nullable = true)
 |-- MONTH: integer (nullable = true)
 |-- DAY: integer (nullable = true)
 |-- DAY_OF_WEEK: integer (nullable = true)
 |-- AIRLINE: string (nullable = true)
 |-- FLIGHT_NUMBER: integer (nullable = true)
 |-- TAIL_NUMBER: string (nullable = true)
 |-- ORIGIN_AIRPORT: string (nullable = true)
 |-- DESTINATION_AIRPORT: string (nullable = true)
 |-- SCHEDULED_DEPARTURE: integer (nullable = true)
 |-- DEPARTURE_TIME: integer (nullable = true)
 |-- DEPARTURE_DELAY: integer (nullable = true)
 |-- TAXI_OUT: integer (nullable = true)
 |-- WHEELS_OFF: integer (nullable = true)
 |-- SCHEDULED_TIME: integer (nullable = true)
 |-- ELAPSED_TIME: integer (nullable = true)
```

```
 |-- AIR_TIME: integer (nullable = true)
 |-- DISTANCE: integer (nullable = true)
 |-- WHEELS_ON: integer (nullable = true)
 |-- TAXI_IN: integer (nullable = true)
 |-- SCHEDULED_ARRIVAL: integer (nullable = true)
 |-- ARRIVAL_TIME: integer (nullable = true)
 |-- ARRIVAL_DELAY: integer (nullable = true)
 |-- DIVERTED: integer (nullable = true)
 |-- CANCELLED: integer (nullable = true)
 |-- CANCELLATION_REASON: string (nullable = true)
 |-- AIR_SYSTEM_DELAY: integer (nullable = true)
 |-- SECURITY_DELAY: integer (nullable = true)
 |-- AIRLINE_DELAY: integer (nullable = true)
 |-- LATE_AIRCRAFT_DELAY: integer (nullable = true)
 |-- WEATHER_DELAY: integer (nullable = true)

+----------------------+-----------------------------+
|count(DISTINCT AIRLINE)|approx_count_distinct(AIRLINE)|
+----------------------+-----------------------------+
|                    14|                           13|
+----------------------+-----------------------------+
```

```
# Funciones min y max

vuelos = spark.read.parquet('./data/vuelos.parquet')

vuelos.printSchema()

from pyspark.sql.functions import min, max, col

vuelos.select(
    min('AIR_TIME').alias('menor_timepo'),
    max('AIR_TIME').alias('mayor_tiempo')
).show()

vuelos.select(
    min('AIRLINE_DELAY'),
    max('AIRLINE_DELAY')
).show()
```

```
root
 |-- YEAR: integer (nullable = true)
 |-- MONTH: integer (nullable = true)
 |-- DAY: integer (nullable = true)
 |-- DAY_OF_WEEK: integer (nullable = true)
 |-- AIRLINE: string (nullable = true)
 |-- FLIGHT_NUMBER: integer (nullable = true)
 |-- TAIL_NUMBER: string (nullable = true)
 |-- ORIGIN_AIRPORT: string (nullable = true)
 |-- DESTINATION_AIRPORT: string (nullable = true)
 |-- SCHEDULED_DEPARTURE: integer (nullable = true)
 |-- DEPARTURE_TIME: integer (nullable = true)
 |-- DEPARTURE_DELAY: integer (nullable = true)
 |-- TAXI_OUT: integer (nullable = true)
 |-- WHEELS_OFF: integer (nullable = true)
 |-- SCHEDULED_TIME: integer (nullable = true)
 |-- ELAPSED_TIME: integer (nullable = true)
 |-- AIR_TIME: integer (nullable = true)
 |-- DISTANCE: integer (nullable = true)
 |-- WHEELS_ON: integer (nullable = true)
 |-- TAXI_IN: integer (nullable = true)
 |-- SCHEDULED_ARRIVAL: integer (nullable = true)
 |-- ARRIVAL_TIME: integer (nullable = true)
 |-- ARRIVAL_DELAY: integer (nullable = true)
 |-- DIVERTED: integer (nullable = true)
 |-- CANCELLED: integer (nullable = true)
 |-- CANCELLATION_REASON: string (nullable = true)
 |-- AIR_SYSTEM_DELAY: integer (nullable = true)
 |-- SECURITY_DELAY: integer (nullable = true)
 |-- AIRLINE_DELAY: integer (nullable = true)
 |-- LATE_AIRCRAFT_DELAY: integer (nullable = true)
 |-- WEATHER_DELAY: integer (nullable = true)

+------------+------------+
|menor_timepo|mayor_tiempo|
+------------+------------+
|           7|         690|
+------------+------------+

+------------------+------------------+
|min(AIRLINE_DELAY)|max(AIRLINE_DELAY)|
+------------------+------------------+
|                 0|              1971|
+------------------+------------------+
```

```
# Funciones sum, sumDistinct y avg


from pyspark.sql.functions import sum, sumDistinct, avg, count

# sum

vuelos.printSchema()
#suma de la columna distancia
vuelos.select(
    sum('DISTANCE').alias('sum_dis')
).show()
```

```
root
 |-- YEAR: integer (nullable = true)
 |-- MONTH: integer (nullable = true)
 |-- DAY: integer (nullable = true)
 |-- DAY_OF_WEEK: integer (nullable = true)
 |-- AIRLINE: string (nullable = true)
 |-- FLIGHT_NUMBER: integer (nullable = true)
 |-- TAIL_NUMBER: string (nullable = true)
 |-- ORIGIN_AIRPORT: string (nullable = true)
 |-- DESTINATION_AIRPORT: string (nullable = true)
 |-- SCHEDULED_DEPARTURE: integer (nullable = true)
 |-- DEPARTURE_TIME: integer (nullable = true)
 |-- DEPARTURE_DELAY: integer (nullable = true)
 |-- TAXI_OUT: integer (nullable = true)
 |-- WHEELS_OFF: integer (nullable = true)
 |-- SCHEDULED_TIME: integer (nullable = true)
 |-- ELAPSED_TIME: integer (nullable = true)
 |-- AIR_TIME: integer (nullable = true)
 |-- DISTANCE: integer (nullable = true)
 |-- WHEELS_ON: integer (nullable = true)
 |-- TAXI_IN: integer (nullable = true)
 |-- SCHEDULED_ARRIVAL: integer (nullable = true)
 |-- ARRIVAL_TIME: integer (nullable = true)
 |-- ARRIVAL_DELAY: integer (nullable = true)
 |-- DIVERTED: integer (nullable = true)
 |-- CANCELLED: integer (nullable = true)
 |-- CANCELLATION_REASON: string (nullable = true)
 |-- AIR_SYSTEM_DELAY: integer (nullable = true)
 |-- SECURITY_DELAY: integer (nullable = true)
 |-- AIRLINE_DELAY: integer (nullable = true)
 |-- LATE_AIRCRAFT_DELAY: integer (nullable = true)
 |-- WEATHER_DELAY: integer (nullable = true)

+----------+
|   sum_dis|
+----------+
|4785357409|
+----------+
```

```
# sumDistinct
#suma solo los valores distintos de la columna, lo que no tiene mucho sentido en este caso
vuelos.select(
    sumDistinct('DISTANCE').alias('sum_dis_dif')
).show()
```

```
/content/spark-3.2.3-bin-hadoop3.2/python/pyspark/sql/functions.py:215: FutureWarning: Deprecated in 3.2, use sum_distinct instead
  warnings.warn("Deprecated in 3.2, use sum_distinct instead.", FutureWarning)
+-----------+
|sum_dis_dif|
+-----------+
|    1442300|
+-----------+
```

```
# avg

vuelos.select(
    avg('AIR_TIME').alias('promedio_aire'),
    (sum('AIR_TIME') / count('AIR_TIME')).alias('prom_manual')
).show()
```

```
+------------------+------------------+
|     promedio_aire|       prom_manual|
+------------------+------------------+
|113.51162809012519|113.51162809012519|
```

```
+------------------+------------------+
```

```
# Agregación con agrupación

vuelos.printSchema()

from pyspark.sql.functions import desc

(vuelos.groupBy('ORIGIN_AIRPORT')
    .count()
    .orderBy(desc('count'))
).show()

(vuelos.groupBy('ORIGIN_AIRPORT', 'DESTINATION_AIRPORT')
    .count()
    .orderBy(desc('count'))
).show()
```

```
 |-- TAXI_IN: integer (nullable = true)
 |-- SCHEDULED_ARRIVAL: integer (nullable = true)
 |-- ARRIVAL_TIME: integer (nullable = true)
 |-- ARRIVAL_DELAY: integer (nullable = true)
 |-- DIVERTED: integer (nullable = true)
 |-- CANCELLED: integer (nullable = true)
 |-- CANCELLATION_REASON: string (nullable = true)
 |-- AIR_SYSTEM_DELAY: integer (nullable = true)
 |-- SECURITY_DELAY: integer (nullable = true)
 |-- AIRLINE_DELAY: integer (nullable = true)
 |-- LATE_AIRCRAFT_DELAY: integer (nullable = true)
 |-- WEATHER_DELAY: integer (nullable = true)

+--------------+------+
|ORIGIN_AIRPORT| count|
+--------------+------+
|           ATL|346836|
|           ORD|285884|
|           DFW|239551|
|           DEN|196055|
|           LAX|194673|
|           SFO|148008|
|           PHX|146815|
|           IAH|146622|
|           LAS|133181|
|           MSP|112117|
|           MCO|110982|
|           SEA|110899|
|           DTW|108500|
|           BOS|107847|
|           EWR|101772|
|           CLT|100324|
|           LGA| 99605|
|           SLC| 97210|
|           JFK| 93811|
|           BWI| 86079|
+--------------+------+
only showing top 20 rows

+--------------+-------------------+-----+
|ORIGIN_AIRPORT|DESTINATION_AIRPORT|count|
+--------------+-------------------+-----+
|           SFO|                LAX|13744|
|           LAX|                SFO|13457|
|           JFK|                LAX|12016|
|           LAX|                JFK|12015|
|           LAS|                LAX| 9715|
|           LGA|                ORD| 9639|
|           LAX|                LAS| 9594|
|           ORD|                LGA| 9575|
|           SFO|                JFK| 8440|
|           JFK|                SFO| 8437|
|           OGG|                HNL| 8313|
|           HNL|                OGG| 8282|
|           LAX|                ORD| 8256|
|           ATL|                LGA| 8234|
|           LGA|                ATL| 8215|
|           ATL|                MCO| 8202|
|           MCO|                ATL| 8202|
```

```
# Varias agregaciones por grupo con la función agg()

from pyspark.sql.functions import count, min, max, desc, avg

vuelos.groupBy('ORIGIN_AIRPORT').agg(
    count('AIR_TIME').alias('tiempo_aire'),
    min('AIR_TIME').alias('min'),
    max('AIR_TIME').alias('max')
```

```
).orderBy(desc('tiempo_aire')).show()

vuelos.groupBy('MONTH').agg(
    count('ARRIVAL_DELAY').alias('conteo_de_retrasos'),
    avg('DISTANCE').alias('prom_dist')
).orderBy(desc('conteo_de_retrasos')).show()
```

```
+--------------+-----------+---+---+
|ORIGIN_AIRPORT|tiempo_aire|min|max|
+--------------+-----------+---+---+
|           ATL|     343506| 15|614|
|           ORD|     276554| 13|571|
|           DFW|     232647| 11|534|
|           DEN|     193402| 12|493|
|           LAX|     192003| 14|409|
|           PHX|     145552| 19|444|
|           SFO|     145491|  8|389|
|           IAH|     144019| 15|524|
|           LAS|     131937| 25|429|
|           MSP|     111055| 14|537|
|           SEA|     110178| 17|412|
|           MCO|     109532| 25|395|
|           DTW|     106992| 15|341|
|           BOS|     104804| 16|432|
|           CLT|      99052| 17|379|
|           EWR|      98341| 21|683|
|           SLC|      96505| 18|419|
|           LGA|      94834| 19|311|
|           JFK|      91663| 29|690|
|           BWI|      84329| 19|398|
+--------------+-----------+---+---+
only showing top 20 rows
```

```
+-----+------------------+-----------------+
|MONTH|conteo_de_retrasos|        prom_dist|
+-----+------------------+-----------------+
|    7|            514384|841.4772794487611|
|    8|            503956|834.8244276603413|
|    6|            492847|835.6302716626612|
|    3|            492138|816.0553268611494|
|    5|            489641|823.3230588760807|
|   10|            482878|816.4436127652134|
|    4|            479251|817.0060476016745|
|   12|            469717|837.8018926194103|
|   11|            462367|820.2482434846529|
|    9|            462153|815.8487523282274|
|    1|            457013|803.2612794913696|
|    2|            407663| 800.785449834689|
+-----+------------------+-----------------+
```

```
# Agregación con pivote
estudiantes = spark.read.parquet('./data/estudiantes.parquet')

estudiantes.show()

from pyspark.sql.functions import min, max, avg, col

estudiantes.groupBy('graduacion').pivot('sexo').agg(avg('peso')).show()

estudiantes.groupBy('graduacion').pivot('sexo').agg(avg('peso'), min('peso'), max('peso')).show()

estudiantes.groupBy('graduacion').pivot('sexo', ['M']).agg(avg('peso'), min('peso'), max('peso')).show()

estudiantes.groupBy('graduacion').pivot('sexo', ['F']).agg(avg('peso'), min('peso'), max('peso')).show()
```

```
+------+----+----+----------+
|nombre|sexo|peso|graduacion|
+------+----+----+----------+
|  Jose|   M|  80|      2000|
| Hilda|   F|  50|      2000|
|  Juan|   M|  75|      2000|
| Pedro|   M|  76|      2001|
|Katia+|   F|  65|      2001|
+------+----+----+----------+
```

```
+----------+----+----+
|graduacion|   F|   M|
+----------+----+----+
|      2001|65.0|76.0|
|      2000|50.0|77.5|
+----------+----+----+
```

```
+----------+-----------+-----------+-----------+-----------+-----------+-----------+
|graduacion|F_avg(peso)|F_min(peso)|F_max(peso)|M_avg(peso)|M_min(peso)|M_max(peso)|
+----------+-----------+-----------+-----------+-----------+-----------+-----------+
|      2001|       65.0|         65|         65|       76.0|         76|         76|
|      2000|       50.0|         50|         50|       77.5|         75|         80|
```

```
+----------+-----------+-----------+-----------+-----------+-----------+-----------+

+----------+-----------+-----------+-----------+
|graduacion|M_avg(peso)|M_min(peso)|M_max(peso)|
+----------+-----------+-----------+-----------+
|      2001|       76.0|         76|         76|
|      2000|       77.5|         75|         80|
+----------+-----------+-----------+-----------+

+----------+-----------+-----------+-----------+
|graduacion|F_avg(peso)|F_min(peso)|F_max(peso)|
+----------+-----------+-----------+-----------+
|      2001|       65.0|         65|         65|
|      2000|       50.0|         50|         50|
+----------+-----------+-----------+-----------+
```

```
# Inner Join

empleados = spark.read.parquet('./data/empleados.parquet')

departamentos = spark.read.parquet('./data/departamentos.parquet')

empleados.show()

departamentos.show()
```

```
+------+--------+
|nombre|num_dpto|
+------+--------+
|  Luis|      33|
| Katia|      33|
|  Raul|      34|
| Pedro|       0|
| Laura|      34|
|Sandro|      31|
+------+--------+

+---+-----------+
| id|nombre_dpto|
+---+-----------+
| 31|     letras|
| 33|    derecho|
| 34| matemática|
| 35|informática|
+---+-----------+
```

```
from pyspark.sql.functions import col

join_df = empleados.join(departamentos, col('num_dpto') == col('id'))

join_df.show()

join_df = empleados.join(departamentos, col('num_dpto') == col('id'), 'inner')

join_df.show()

join_df = empleados.join(departamentos).where(col('num_dpto') == col('id'))

join_df.show()
```

```
+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|  Luis|      33| 33|    derecho|
| Katia|      33| 33|    derecho|
|  Raul|      34| 34| matemática|
| Laura|      34| 34| matemática|
|Sandro|      31| 31|     letras|
+------+--------+---+-----------+

+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|  Luis|      33| 33|    derecho|
| Katia|      33| 33|    derecho|
|  Raul|      34| 34| matemática|
| Laura|      34| 34| matemática|
|Sandro|      31| 31|     letras|
+------+--------+---+-----------+

+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
```

```
|  Luis|      33| 33|    derecho|
| Katia|      33| 33|    derecho|
|  Raul|      34| 34| matemática|
| Laura|      34| 34| matemática|
|Sandro|      31| 31|     letras|
+------+--------+---+-----------+
```

```
# Left Outer Join

empleados.join(departamentos, col('num_dpto') == col('id'), 'leftouter').show()

empleados.join(departamentos, col('num_dpto') == col('id'), 'left_outer').show()

empleados.join(departamentos, col('num_dpto') == col('id'), 'left').show()
```

```
+------+--------+----+-----------+
|nombre|num_dpto|  id|nombre_dpto|
+------+--------+----+-----------+
|  Luis|      33|  33|    derecho|
| Katia|      33|  33|    derecho|
|  Raul|      34|  34| matemática|
| Pedro|       0|null|       null|
| Laura|      34|  34| matemática|
|Sandro|      31|  31|     letras|
+------+--------+----+-----------+

+------+--------+----+-----------+
|nombre|num_dpto|  id|nombre_dpto|
+------+--------+----+-----------+
|  Luis|      33|  33|    derecho|
| Katia|      33|  33|    derecho|
|  Raul|      34|  34| matemática|
| Pedro|       0|null|       null|
| Laura|      34|  34| matemática|
|Sandro|      31|  31|     letras|
+------+--------+----+-----------+

+------+--------+----+-----------+
|nombre|num_dpto|  id|nombre_dpto|
+------+--------+----+-----------+
|  Luis|      33|  33|    derecho|
| Katia|      33|  33|    derecho|
|  Raul|      34|  34| matemática|
| Pedro|       0|null|       null|
| Laura|      34|  34| matemática|
|Sandro|      31|  31|     letras|
+------+--------+----+-----------+
```

```
# Right Outer Join

empleados.join(departamentos, col('num_dpto') == col('id'), 'rightouter').show()

empleados.join(departamentos, col('num_dpto') == col('id'), 'right_outer').show()

empleados.join(departamentos, col('num_dpto') == col('id'), 'right').show()
```

```
+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|Sandro|      31| 31|     letras|
| Katia|      33| 33|    derecho|
|  Luis|      33| 33|    derecho|
| Laura|      34| 34| matemática|
|  Raul|      34| 34| matemática|
|  null|    null| 35|informática|
+------+--------+---+-----------+

+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|Sandro|      31| 31|     letras|
| Katia|      33| 33|    derecho|
|  Luis|      33| 33|    derecho|
| Laura|      34| 34| matemática|
|  Raul|      34| 34| matemática|
|  null|    null| 35|informática|
+------+--------+---+-----------+

+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|Sandro|      31| 31|     letras|
| Katia|      33| 33|    derecho|
|  Luis|      33| 33|    derecho|
| Laura|      34| 34| matemática|
```

```
|  Raul|      34| 34| matemática|
|  null|    null| 35|informática|
+------+--------+---+-----------+
```

```
# Full Outer Join

empleados.join(departamentos, col('num_dpto') == col('id'), 'outer').show()
```

```
+------+--------+----+-----------+
|nombre|num_dpto|  id|nombre_dpto|
+------+--------+----+-----------+
| Pedro|       0|null|       null|
|Sandro|      31|  31|     letras|
|  Luis|      33|  33|    derecho|
| Katia|      33|  33|    derecho|
|  Raul|      34|  34| matemática|
| Laura|      34|  34| matemática|
|  null|    null|  35|informática|
+------+--------+----+-----------+
```

```
# Left Anti Join

#que filas del conjunto de datos de la izquierda no tienen datos coincidentes con el conjunto de la derecha

empleados.join(departamentos, col('num_dpto') == col('id'), 'left_anti').show() #pedro no esta asignado a ningún departamento

departamentos.join(empleados, col('num_dpto') == col('id'), 'left_anti').show()
```

```
+------+--------+
|nombre|num_dpto|
+------+--------+
| Pedro|       0|
+------+--------+
```

```
+---+-----------+
| id|nombre_dpto|
+---+-----------+
| 35|informática|
+---+-----------+
```

```
# Left Semi Join
#similar al anti join pero no muestra los datos de la derecha

empleados.join(departamentos, col('num_dpto') == col('id'), 'left_semi').show() #que trabajadores si estan asignados a algun departament
```

```
+------+--------+
|nombre|num_dpto|
+------+--------+
|  Luis|      33|
| Katia|      33|
|  Raul|      34|
| Laura|      34|
|Sandro|      31|
+------+--------+
```

```
# Cross Join

df = empleados.crossJoin(departamentos)

df.show()

df.count()
```

```
+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|  Luis|      33| 31|     letras|
|  Luis|      33| 33|    derecho|
|  Luis|      33| 34| matemática|
|  Luis|      33| 35|informática|
| Katia|      33| 31|     letras|
| Katia|      33| 33|    derecho|
| Katia|      33| 34| matemática|
| Katia|      33| 35|informática|
|  Raul|      34| 31|     letras|
|  Raul|      34| 33|    derecho|
|  Raul|      34| 34| matemática|
|  Raul|      34| 35|informática|
| Pedro|       0| 31|     letras|
| Pedro|       0| 33|    derecho|
```

```
| Pedro|       0| 34| matemática|
| Pedro|       0| 35|informática|
| Laura|      34| 31|     letras|
| Laura|      34| 33|    derecho|
| Laura|      34| 34| matemática|
| Laura|      34| 35|informática|
+------+--------+---+-----------+
only showing top 20 rows

24
```

```
# Manejo de nombres de columnas duplicados

depa = departamentos.withColumn('num_dpto', col('id'))

depa.printSchema()

empleados.printSchema()

# Devuelve un error
#empleados.join(depa, col('num_dpto') == col('num_dpto'))
```

```
root
 |-- id: long (nullable = true)
 |-- nombre_dpto: string (nullable = true)
 |-- num_dpto: long (nullable = true)

root
 |-- nombre: string (nullable = true)
 |-- num_dpto: long (nullable = true)
```

```
# Forma correcta

df_con_duplicados = empleados.join(depa, empleados['num_dpto'] == depa['num_dpto'])

df_con_duplicados.printSchema()

df_con_duplicados.select(empleados['num_dpto']).show()

df2 = empleados.join(depa, 'num_dpto')

df2.printSchema()

empleados.join(depa, ['num_dpto']).printSchema()
```

```
root
 |-- nombre: string (nullable = true)
 |-- num_dpto: long (nullable = true)
 |-- id: long (nullable = true)
 |-- nombre_dpto: string (nullable = true)
 |-- num_dpto: long (nullable = true)

+--------+
|num_dpto|
+--------+
|      33|
|      33|
|      34|
|      34|
|      31|
+--------+

root
 |-- num_dpto: long (nullable = true)
 |-- nombre: string (nullable = true)
 |-- id: long (nullable = true)
 |-- nombre_dpto: string (nullable = true)

root
 |-- num_dpto: long (nullable = true)
 |-- nombre: string (nullable = true)
 |-- id: long (nullable = true)
 |-- nombre_dpto: string (nullable = true)
```

```
# Shuffle Hash Join y Broadcast Hash Join

#Broadcast es para uniones pequeñas, menos memoria
# Shuffle Hash Join para conjuntos más grandes
```

```
from pyspark.sql.functions import col, broadcast

empleados.join(broadcast(departamentos), col('num_dpto') == col('id')).show()

empleados.join(broadcast(departamentos), col('num_dpto') == col('id')).explain()
```

```
+------+--------+---+-----------+
|nombre|num_dpto| id|nombre_dpto|
+------+--------+---+-----------+
|  Luis|      33| 33|    derecho|
| Katia|      33| 33|    derecho|
|  Raul|      34| 34| matemática|
| Laura|      34| 34| matemática|
|Sandro|      31| 31|     letras|
+------+--------+---+-----------+

== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- BroadcastHashJoin [num_dpto#7762L], [id#7765L], Inner, BuildRight, false
   :- Filter isnotnull(num_dpto#7762L)
   :  +- FileScan parquet [nombre#7761,num_dpto#7762L] Batched: true, DataFilters: [isnotnull(num_dpto#7762L)], Format: Parquet, Lo
   +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, bigint, false]),false), [plan_id=3080]
      +- Filter isnotnull(id#7765L)
         +- FileScan parquet [id#7765L,nombre_dpto#7766] Batched: true, DataFilters: [isnotnull(id#7765L)], Format: Parquet, Locati
```

✓ 0 s completado a las 19:04