```
#check that java is installed
!java -version
```

```
openjdk version "11.0.27" 2025-04-15
OpenJDK Runtime Environment (build 11.0.27+6-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.27+6-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
```

```
#install pyspark
!pip install pyspark
```

```
Requirement already satisfied: pyspark in /usr/local/lib/python3.11/dist-packages (3.5.1)
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.11/dist-packages (from pyspark) (0.10.9.7)
```

```
import os
import pandas as pd
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DoubleType, LongType, TimestampType
from pyspark.sql.functions import spark_partition_id
```

```
spark = SparkSession.builder.appName('AutomotiveData').getOrCreate()

print(f'The Spark version is {spark.version}')
```

```
The Spark version is 3.5.1
```

```
#Table creation for:
#transactions_df (large table)
#countries_df (small table)

from pyspark.sql import Row

#Broadcast join
transactions_data = [
    Row(id=1, countrycode="IN", amount=1000),    Row(id=2, countrycode="US", amount=1500),    Row(id=3, countrycode="FR", amount=900),
    Row(id=4, countrycode="IN", amount=700),    Row(id=5, countrycode="US", amount=1200),    Row(id=6, countrycode="FR", amount=1300),
    Row(id=7, countrycode="DE", amount=1100),   Row(id=8, countrycode="IN", amount=1400),    Row(id=9, countrycode="CA", amount=800),
    Row(id=10, countrycode="US", amount=1700),    Row(id=11, countrycode="DE", amount=950),    Row(id=12, countrycode="FR", amount=1150),
    Row(id=13, countrycode="IN", amount=1600),    Row(id=14, countrycode="CA", amount=1250)
]
transactions_df = spark.createDataFrame(transactions_data)

countries_data = [Row(countrycode="IN", countryname="India"),
                  Row(countrycode="US", countryname="USA"),
                  Row(countrycode="FR", countryname="France")]
countries_df = spark.createDataFrame(countries_data)
```

```
spark.conf.set("spark.sql.adaptive.enabled","false")
spark.conf.get("spark.sql.adaptive.enabled")
```

```
'false'
```

```
joined_df = transactions_df.join((countries_df),on="countrycode",how="inner")
joined_df.show()
```

```
+-----------+---+------+-----------+
|countrycode| id|amount|countryname|
+-----------+---+------+-----------+
|         US|  2|  1500|        USA|
|         US|  5|  1200|        USA|
|         US| 10|  1700|        USA|
|         IN|  1|  1000|      India|
|         IN|  4|   700|      India|
|         IN|  8|  1400|      India|
|         IN| 13|  1600|      India|
|         FR|  3|   900|     France|
|         FR|  6|  1300|     France|
|         FR| 12|  1150|     France|
+-----------+---+------+-----------+
```

```
joined_df.explain(True)
```

```
== Parsed Logical Plan ==
'Join UsingJoin(Inner, [countrycode])
:- LogicalRDD [id#0L, countrycode#1, amount#2L], false
+- LogicalRDD [countrycode#6, countryname#7], false

== Analyzed Logical Plan ==
countrycode: string, id: bigint, amount: bigint, countryname: string
Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- Join Inner, (countrycode#1 = countrycode#6)
   :- LogicalRDD [id#0L, countrycode#1, amount#2L], false
   +- LogicalRDD [countrycode#6, countryname#7], false

== Optimized Logical Plan ==
Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- Join Inner, (countrycode#1 = countrycode#6)
   :- Filter isnotnull(countrycode#1)
   :  +- LogicalRDD [id#0L, countrycode#1, amount#2L], false
   +- Filter isnotnull(countrycode#6)
      +- LogicalRDD [countrycode#6, countryname#7], false

== Physical Plan ==
*(5) Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- *(5) SortMergeJoin [countrycode#1], [countrycode#6], Inner
   :- *(2) Sort [countrycode#1 ASC NULLS FIRST], false, 0
   :  +- Exchange hashpartitioning(countrycode#1, 200), ENSURE_REQUIREMENTS, [plan_id=93]
   :     +- *(1) Filter isnotnull(countrycode#1)
   :        +- *(1) Scan ExistingRDD[id#0L,countrycode#1,amount#2L]
   +- *(4) Sort [countrycode#6 ASC NULLS FIRST], false, 0
      +- Exchange hashpartitioning(countrycode#6, 200), ENSURE_REQUIREMENTS, [plan_id=99]
         +- *(3) Filter isnotnull(countrycode#6)
            +- *(3) Scan ExistingRDD[countrycode#6,countryname#7]
```

```python
from pyspark.sql.functions import broadcast

joined_df_broadcast= transactions_df.join(broadcast(countries_df),on="countrycode",how="inner")
joined_df_broadcast.show()
```

```
+-----------+---+------+-----------+
|countrycode| id|amount|countryname|
+-----------+---+------+-----------+
|         IN|  1|  1000|      India|
|         US|  2|  1500|        USA|
|         FR|  3|   900|     France|
|         IN|  4|   700|      India|
|         US|  5|  1200|        USA|
|         FR|  6|  1300|     France|
|         IN|  8|  1400|      India|
|         US| 10|  1700|        USA|
|         FR| 12|  1150|     France|
|         IN| 13|  1600|      India|
+-----------+---+------+-----------+
```

```python
joined_df_broadcast.explain(True)
```

```
== Parsed Logical Plan ==
'Join UsingJoin(Inner, [countrycode])
:- LogicalRDD [id#0L, countrycode#1, amount#2L], false
+- ResolvedHint (strategy=broadcast)
   +- LogicalRDD [countrycode#6, countryname#7], false

== Analyzed Logical Plan ==
countrycode: string, id: bigint, amount: bigint, countryname: string
Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- Join Inner, (countrycode#1 = countrycode#6)
   :- LogicalRDD [id#0L, countrycode#1, amount#2L], false
   +- ResolvedHint (strategy=broadcast)
      +- LogicalRDD [countrycode#6, countryname#7], false

== Optimized Logical Plan ==
Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- Join Inner, (countrycode#1 = countrycode#6), rightHint=(strategy=broadcast)
   :- Filter isnotnull(countrycode#1)
   :  +- LogicalRDD [id#0L, countrycode#1, amount#2L], false
   +- Filter isnotnull(countrycode#6)
      +- LogicalRDD [countrycode#6, countryname#7], false

== Physical Plan ==
*(2) Project [countrycode#1, id#0L, amount#2L, countryname#7]
+- *(2) BroadcastHashJoin [countrycode#1], [countrycode#6], Inner, BuildRight, false
```

```
        :- *(2) Filter isnotnull(countrycode#1)
        :  +- *(2) Scan ExistingRDD[id#0L,countrycode#1,amount#2L]
        +- BroadcastExchange HashedRelationBroadcastMode(List(input[0, string, false]),false), [plan_id=188]
           +- *(1) Filter isnotnull(countrycode#6)
              +- *(1) Scan ExistingRDD[countrycode#6,countryname#7]
```

SQL HINTS

```
countries_df.createOrReplaceTempView("countries")
transactions_df.createOrReplaceTempView("transactions")
```

```
spark.sql("select * from countries").show()
```

```
+-----------+-----------+
|countrycode|countryname|
+-----------+-----------+
|         IN|      India|
|         US|        USA|
|         FR|     France|
+-----------+-----------+
```

```
result = spark.sql("""
    SELECT
        t.id,
        t.countrycode,
        t.amount,
        c.countryname
    FROM
        transactions t
    INNER JOIN
        countries c
    ON
        t.countrycode = c.countrycode
""")
result.show()
```

```
+---+-----------+------+-----------+
| id|countrycode|amount|countryname|
+---+-----------+------+-----------+
|  2|         US|  1500|        USA|
|  5|         US|  1200|        USA|
| 10|         US|  1700|        USA|
|  1|         IN|  1000|      India|
|  4|         IN|   700|      India|
|  8|         IN|  1400|      India|
| 13|         IN|  1600|      India|
|  3|         FR|   900|     France|
|  6|         FR|  1300|     France|
| 12|         FR|  1150|     France|
+---+-----------+------+-----------+
```

```
result_SQLhint = spark.sql("""
    SELECT /*+ BROADCAST(c) */
        t.id,
        t.countrycode,
        t.amount,
        c.countryname
    FROM
        transactions t
    INNER JOIN
        countries c
    ON
        t.countrycode = c.countrycode
""")
result_SQLhint.show()
```

```
+---+-----------+------+-----------+
| id|countrycode|amount|countryname|
+---+-----------+------+-----------+
|  1|         IN|  1000|      India|
|  2|         US|  1500|        USA|
|  3|         FR|   900|     France|
|  4|         IN|   700|      India|
|  5|         US|  1200|        USA|
|  6|         FR|  1300|     France|
|  8|         IN|  1400|      India|
| 10|         US|  1700|        USA|
| 12|         FR|  1150|     France|
| 13|         IN|  1600|      India|
+---+-----------+------+-----------+
```