

dataFrameOperationsBasics

February 9, 2026

0.1 Basic Data Frame Operations

```
[1]: from pyspark.sql import SparkSession  
  
spark=SparkSession.builder.appName("Data Frame").getOrCreate()  
  
sc=spark.sparkContext  
sc.stop()
```

the entry point to programming spark with the data frame is spark session
with a sql context, applications can create data frames from an existing RDD, from a hive table or
from data sources

```
[2]: from pyspark import SparkContext  
from pyspark.sql import SparkSession,SQLContext  
  
# old way of creating spark context  
sc1=SparkContext(master='local',appName='test1')  
  
# creating spark session  
spark1=SparkSession(sc1)  
  
# creating SQL context  
sqlcontext1=SQLContext(sc1)
```

```
c:\Users\emada\AppData\Local\Programs\Python\Python311\Lib\site-  
packages\pyspark\sql\context.py:112: FutureWarning: Deprecated in 3.0.0. Use  
SparkSession.builder.getOrCreate() instead.  
warnings.warn(
```

```
[ ]: spark=SparkSession.builder.appName("test").getOrCreate()  
  
# for rdd learning  
sc=spark.sparkContext  
rdd=sc.textFile("test.csv")  
  
# for dataframe purpose  
df=spark.read.csv('test.csv',header=True)
```

```
[5]: # Importing data using spark session
df1 = spark1.read.csv(path="iris/iris.csv", sep=",", header=True)
# full example
df1 = spark.read.csv(
    "iris/iris.csv",
    sep=",",
    header=True,
    inferSchema=True,
    nullValue="NA",
    mode="PERMISSIVE",
    ignoreLeadingWhiteSpace=True,
)

# importing data using sql context
df2 = sqlcontext1.read.csv(path="iris/iris.csv", sep=",", header=True)

# sqlcontext is old api, used for sql & dataframes(old style)

print(df1)
# print(df1.show())
print(df2)

iris1_df1 = spark1.read.json("iris/iris.json")

iris1_df1.show()
```

DataFrame[Sepal_Length: double, Sepal_Width: double, Petal_Length: double, Petal_Width: double, Species: string]

DataFrame[Sepal_Length: string, Sepal_Width: string, Petal_Length: string, Petal_Width: string, Species: string]

Petal_Length	Petal_Width	Sepal_Length	Sepal_Width	Species
1.4	0.2	5.1	3.5	setosa
1.4	0.2	4.9	3.0	setosa
1.3	0.2	4.7	3.2	setosa
1.5	0.2	4.6	3.1	setosa
1.4	0.2	5.0	3.6	setosa
1.7	0.4	5.4	3.9	setosa
1.4	0.3	4.6	3.4	setosa
1.5	0.2	5.0	3.4	setosa
1.4	0.2	4.4	2.9	setosa
1.5	0.1	4.9	3.1	setosa
1.5	0.2	5.4	3.7	setosa
1.6	0.2	4.8	3.4	setosa
1.4	0.1	4.8	3.0	setosa
1.1	0.1	4.3	3.0	setosa
1.2	0.2	5.8	4.0	setosa

```

|      1.5|      0.4|      5.7|      4.4| setosa|
|      1.3|      0.4|      5.4|      3.9| setosa|
|      1.4|      0.3|      5.1|      3.5| setosa|
|      1.7|      0.3|      5.7|      3.8| setosa|
|      1.5|      0.3|      5.1|      3.8| setosa|
+-----+-----+-----+-----+
only showing top 20 rows

```

```
[6]: # Convert RDD to Data Frame
# using createDataFrame function
iris1=sc1.textFile('iris/iris_site.csv')
iris1_split=iris1.map(lambda line:line.split(","))
df1=spark1.createDataFrame(iris1_split)
df1.show(10)
```

```

+---+---+---+---+---+
| _1| _2| _3| _4| _5|
+---+---+---+---+
|5.1|3.5|1.4|0.2|setosa|
|4.9|3.0|1.4|0.2|setosa|
|4.7|3.2|1.3|0.2|setosa|
|4.6|3.1|1.5|0.2|setosa|
|5.0|3.6|1.4|0.2|setosa|
|5.4|3.9|1.7|0.4|setosa|
|4.6|3.4|1.4|0.3|setosa|
|5.0|3.4|1.5|0.2|setosa|
|4.4|2.9|1.4|0.2|setosa|
|4.9|3.1|1.5|0.1|setosa|
+---+---+---+---+
only showing top 10 rows

```

```
[7]: # convert dataframe to rdd
iris1_df1=spark1.read.csv('iris/iris.csv',sep=',',header=True)
iris1_df1.rdd.map(tuple).take(10)
```

```
[7]: [('5.1', '3.5', '1.4', '0.2', 'setosa'),
 ('4.9', '3.0', '1.4', '0.2', 'setosa'),
 ('4.7', '3.2', '1.3', '0.2', 'setosa'),
 ('4.6', '3.1', '1.5', '0.2', 'setosa'),
 ('5.0', '3.6', '1.4', '0.2', 'setosa'),
 ('5.4', '3.9', '1.7', '0.4', 'setosa'),
 ('4.6', '3.4', '1.4', '0.3', 'setosa'),
 ('5.0', '3.4', '1.5', '0.2', 'setosa'),
 ('4.4', '2.9', '1.4', '0.2', 'setosa'),
 ('4.9', '3.1', '1.5', '0.1', 'setosa')]
```

```
[8]: # Display contents of data frame in table format
iris_df1=spark1.read.csv('iris/iris.csv',sep=',',header=True)
iris1_df1.show(5) # shows only top 5 rows

iris1_df1.collect() # display content of dataframe as a list of rows

iris1_df1.head(10) # shows 1st 10 rows of data frame as a list of rows
```

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa

only showing top 5 rows

```
[8]: [Row(Sepal_Length='5.1', Sepal_Width='3.5', Petal_Length='1.4',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='4.9', Sepal_Width='3.0', Petal_Length='1.4',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='4.7', Sepal_Width='3.2', Petal_Length='1.3',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='4.6', Sepal_Width='3.1', Petal_Length='1.5',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='5.0', Sepal_Width='3.6', Petal_Length='1.4',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='5.4', Sepal_Width='3.9', Petal_Length='1.7',
Petal_Width='0.4', Species='setosa'),
Row(Sepal_Length='4.6', Sepal_Width='3.4', Petal_Length='1.4',
Petal_Width='0.3', Species='setosa'),
Row(Sepal_Length='5.0', Sepal_Width='3.4', Petal_Length='1.5',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='4.4', Sepal_Width='2.9', Petal_Length='1.4',
Petal_Width='0.2', Species='setosa'),
Row(Sepal_Length='4.9', Sepal_Width='3.1', Petal_Length='1.5',
Petal_Width='0.1', Species='setosa')]
```

```
[11]: # display all columns
iris_df1.columns

iris_df1.printSchema()

root
|-- Sepal_Length: string (nullable = true)
|-- Sepal_Width: string (nullable = true)
```

```
|-- Petal_Length: string (nullable = true)
|-- Petal_Width: string (nullable = true)
|-- Species: string (nullable = true)
```

```
[26]: # Data Selection
# selecting any particular column
iris1_df1=spark1.read.csv('iris/iris.csv',sep=',',header=True)
iris1_df1.select("Sepal_Length","Species").show()
```

```
+-----+-----+
|Sepal_Length|Species|
+-----+-----+
|      5.1| setosa|
|      4.9| setosa|
|      4.7| setosa|
|      4.6| setosa|
|      5.0| setosa|
|      5.4| setosa|
|      4.6| setosa|
|      5.0| setosa|
|      4.4| setosa|
|      4.9| setosa|
|      5.4| setosa|
|      4.8| setosa|
|      4.8| setosa|
|      4.3| setosa|
|      5.8| setosa|
|      5.7| setosa|
|      5.4| setosa|
|      5.1| setosa|
|      5.7| setosa|
|      5.1| setosa|
+-----+-----+
only showing top 20 rows
```

```
[14]: # renaming columns while selecting
from pyspark.sql.functions import col
iris_df1.select(col("Sepal_Length").alias("SepalLength"),
                col("Species").alias("SpeciesType")).show(8)
```

```
+-----+-----+
|SepalLength|SpeciesType|
+-----+-----+
|      5.1|    setosa|
|      4.9|    setosa|
|      4.7|    setosa|
|      4.6|    setosa|
|      5.0|    setosa|
```

```

|      5.4|    setosa|
|      4.6|    setosa|
|      5.0|    setosa|
+-----+
only showing top 8 rows

```

```
[19]: # filter rows
iris_df1.filter(col("Sepal_Length")>4.5).show(8)

# multiple conditions
iris_df1.filter((col("Sepal_Length")>4.5) & (col("Species")=="setosa")).show(10)

+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+
|      5.1|      3.5|      1.4|      0.2|  setosa|
|      4.9|      3.0|      1.4|      0.2|  setosa|
|      4.7|      3.2|      1.3|      0.2|  setosa|
|      4.6|      3.1|      1.5|      0.2|  setosa|
|      5.0|      3.6|      1.4|      0.2|  setosa|
|      5.4|      3.9|      1.7|      0.4|  setosa|
|      4.6|      3.4|      1.4|      0.3|  setosa|
|      5.0|      3.4|      1.5|      0.2|  setosa|
+-----+-----+-----+-----+
only showing top 8 rows
+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+
|      5.1|      3.5|      1.4|      0.2|  setosa|
|      4.9|      3.0|      1.4|      0.2|  setosa|
|      4.7|      3.2|      1.3|      0.2|  setosa|
|      4.6|      3.1|      1.5|      0.2|  setosa|
|      5.0|      3.6|      1.4|      0.2|  setosa|
|      5.4|      3.9|      1.7|      0.4|  setosa|
|      4.6|      3.4|      1.4|      0.3|  setosa|
|      5.0|      3.4|      1.5|      0.2|  setosa|
|      4.9|      3.1|      1.5|      0.1|  setosa|
|      5.4|      3.7|      1.5|      0.2|  setosa|
+-----+-----+-----+-----+
only showing top 10 rows
```

```
[20]: # select + filter together
iris1_df1.select("Sepal_Length", "Species").filter(col("Sepal_Length")>4.7).
    show(8) # order doesnt matter
```

```
+-----+
|Sepal_Length|Species|
+-----+
|      5.1|  setosa|
```

```

|      4.9| setosa|
|      5.0| setosa|
|      5.4| setosa|
|      5.0| setosa|
|      4.9| setosa|
|      5.4| setosa|
|      4.8| setosa|
+-----+
only showing top 8 rows

```

```
[33]: # withColumn(add/modify columns)
# add new column
# lit() --> used to set constant value

from pyspark.sql.functions import col,lit
df2=iris1_df1.withColumn("NewColumn",lit("test"))
# df2.show()

# to modify a value
# df2.withColumn("Sepal_Length",col("Sepal_Length")+5).show() # this is string
# type so we need to convert to float
# dataframe is immutable,so we need to reassign
df2=df2.withColumn("Sepal_Length",col("Sepal_Length").cast("double")+5.0)
df2.show(10)
```

```

+-----+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|NewColumn|
+-----+-----+-----+-----+-----+
|      10.1|      3.5|      1.4|      0.2| setosa| test|
|      9.9|      3.0|      1.4|      0.2| setosa| test|
|      9.7|      3.2|      1.3|      0.2| setosa| test|
|      9.6|      3.1|      1.5|      0.2| setosa| test|
|     10.0|      3.6|      1.4|      0.2| setosa| test|
|     10.4|      3.9|      1.7|      0.4| setosa| test|
|      9.6|      3.4|      1.4|      0.3| setosa| test|
|     10.0|      3.4|      1.5|      0.2| setosa| test|
|      9.4|      2.9|      1.4|      0.2| setosa| test|
|      9.9|      3.1|      1.5|      0.1| setosa| test|
+-----+-----+-----+-----+-----+
only showing top 10 rows

```

```
[26]: # modify existing column
df2.withColumn("Species",lit("newSpecies")).show(5) # Species column is
# modified to new species
# to just rename the column use
df2=df2.withColumnRenamed("Species","newSpecies")
df2.show()
```

```

+-----+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|  Species|NewColumn|
+-----+-----+-----+-----+-----+
|      5.1|      3.5|      1.4|      0.2|newSpecies|  test|
|      4.9|      3.0|      1.4|      0.2|newSpecies|  test|
|      4.7|      3.2|      1.3|      0.2|newSpecies|  test|
|      4.6|      3.1|      1.5|      0.2|newSpecies|  test|
|      5.0|      3.6|      1.4|      0.2|newSpecies|  test|
+-----+-----+-----+-----+-----+
only showing top 5 rows
+-----+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|newSpecies|NewColumn|
+-----+-----+-----+-----+-----+
|      5.1|      3.5|      1.4|      0.2|  setosa|  test|
|      4.9|      3.0|      1.4|      0.2|  setosa|  test|
|      4.7|      3.2|      1.3|      0.2|  setosa|  test|
|      4.6|      3.1|      1.5|      0.2|  setosa|  test|
|      5.0|      3.6|      1.4|      0.2|  setosa|  test|
|      5.4|      3.9|      1.7|      0.4|  setosa|  test|
|      4.6|      3.4|      1.4|      0.3|  setosa|  test|
|      5.0|      3.4|      1.5|      0.2|  setosa|  test|
|      4.4|      2.9|      1.4|      0.2|  setosa|  test|
|      4.9|      3.1|      1.5|      0.1|  setosa|  test|
|      5.4|      3.7|      1.5|      0.2|  setosa|  test|
|      4.8|      3.4|      1.6|      0.2|  setosa|  test|
|      4.8|      3.0|      1.4|      0.1|  setosa|  test|
|      4.3|      3.0|      1.1|      0.1|  setosa|  test|
|      5.8|      4.0|      1.2|      0.2|  setosa|  test|
|      5.7|      4.4|      1.5|      0.4|  setosa|  test|
|      5.4|      3.9|      1.3|      0.4|  setosa|  test|
|      5.1|      3.5|      1.4|      0.3|  setosa|  test|
|      5.7|      3.8|      1.7|      0.3|  setosa|  test|
|      5.1|      3.8|      1.5|      0.3|  setosa|  test|
+-----+-----+-----+-----+-----+
only showing top 20 rows

```

0.1.1 Sorting

sorting in spark -> distributed sort+shuffle

it reorders entire rows, not just columns

sort is just an alias for orderBy

```
[42]: # Sorting(orderBy)
iris1_df1.orderBy("Sepal_Length").show(10) # ascending

# iris1_df1.orderBy("Sepal_Length".desc()).show() # it gives error bcz str
    ↴should be converted
```

```

iris1_df1.orderBy(col("Sepal_Length").desc()).show(10)
# this can be written
iris_df1.orderBy("Sepal_Length",ascending=False).show(10)

# iris_df1.show()

```

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
4.3	3.0	1.1	0.1	setosa
4.4	2.9	1.4	0.2	setosa
4.4	3.0	1.3	0.2	setosa
4.4	3.2	1.3	0.2	setosa
4.5	2.3	1.3	0.3	setosa
4.6	3.1	1.5	0.2	setosa
4.6	3.4	1.4	0.3	setosa
4.6	3.6	1.0	0.2	setosa
4.6	3.2	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa

only showing top 10 rows

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
7.9	3.8	6.4	2.0	virginical
7.7	2.6	6.9	2.3	virginical
7.7	3.0	6.1	2.3	virginical
7.7	2.8	6.7	2.0	virginical
7.7	3.8	6.7	2.2	virginical
7.6	3.0	6.6	2.1	virginical
7.4	2.8	6.1	1.9	virginical
7.3	2.9	6.3	1.8	virginical
7.2	3.6	6.1	2.5	virginical
7.2	3.2	6.0	1.8	virginical

only showing top 10 rows

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
7.9	3.8	6.4	2.0	virginical
7.7	2.6	6.9	2.3	virginical
7.7	3.0	6.1	2.3	virginical
7.7	2.8	6.7	2.0	virginical
7.7	3.8	6.7	2.2	virginical
7.6	3.0	6.6	2.1	virginical
7.4	2.8	6.1	1.9	virginical
7.3	2.9	6.3	1.8	virginical
7.2	3.6	6.1	2.5	virginical
7.2	3.2	6.1	1.8	virginical

```

|      7.2|       3.2|       6.0|      1.8|virginica|
+-----+-----+-----+-----+
only showing top 10 rows

```

```
[47]: # sorting by multiple columns
# sort by species then by Sepal_Length
iris_df1.orderBy("Species","Sepal_Length").show(10)
# groups rows by species, and within each species sorts by Sepal_Length
```

```

+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+
|      4.3|       3.0|       1.1|      0.1| setosa|
|      4.4|       2.9|       1.4|      0.2| setosa|
|      4.4|       3.0|       1.3|      0.2| setosa|
|      4.4|       3.2|       1.3|      0.2| setosa|
|      4.5|       2.3|       1.3|      0.3| setosa|
|      4.6|       3.1|       1.5|      0.2| setosa|
|      4.6|       3.4|       1.4|      0.3| setosa|
|      4.6|       3.6|       1.0|      0.2| setosa|
|      4.6|       3.2|       1.4|      0.2| setosa|
|      4.7|       3.2|       1.3|      0.2| setosa|
+-----+-----+-----+-----+
only showing top 10 rows

```

```
[51]: # sorting+null values
# default behaviour --> nulls comes first in ascending, nulls come last in
# descending
# control it manually
iris_df1.orderBy(col("Sepal_Length").asc_nulls_last())
iris_df1.orderBy(col("Sepal_Length").asc_nulls_first())
```

```
[51]: DataFrame[Sepal_Length: string, Sepal_Width: string, Petal_Length: string,
Petal_Width: string, Species: string]
```

```
[46]: # mixed order directions
iris_df1.orderBy(
    col("Species").asc(),
    col("Sepal_Length").desc()
).show(10)
```

```

+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+
|      5.8|       4.0|       1.2|      0.2| setosa|
|      5.7|       4.4|       1.5|      0.4| setosa|
|      5.7|       3.8|       1.7|      0.3| setosa|
|      5.5|       4.2|       1.4|      0.2| setosa|
|      5.5|       3.5|       1.3|      0.2| setosa|
+-----+-----+-----+-----+

```

```

|      5.4|      3.9|      1.7|      0.4|  setosa|
|      5.4|      3.7|      1.5|      0.2|  setosa|
|      5.4|      3.9|      1.3|      0.4|  setosa|
|      5.4|      3.4|      1.7|      0.2|  setosa|
|      5.4|      3.4|      1.5|      0.4|  setosa|
+-----+-----+-----+-----+
only showing top 10 rows

```

```
[50]: # sorting +limit
iris1_df1.orderBy(col("Sepal_Length").desc()).limit(5).show(5)
```

```

+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|  Species|
+-----+-----+-----+-----+
|      7.9|      3.8|      6.4|      2.0|virginical|
|      7.7|      3.8|      6.7|      2.2|virginical|
|      7.7|      2.6|      6.9|      2.3|virginical|
|      7.7|      2.8|      6.7|      2.0|virginical|
|      7.7|      3.0|      6.1|      2.3|virginical|
+-----+-----+-----+-----+

```

```
[48]: iris1_df1.orderBy("Sepal_Length").explain(True)
```

```

== Parsed Logical Plan ==
'Sort ['Sepal_Length ASC NULLS FIRST], true
+- Relation
[Sepal_Length#137,Sepal_Width#138,Petal_Length#139,Petal_Width#140,Species#141]
csv

== Analyzed Logical Plan ==
Sepal_Length: string, Sepal_Width: string, Petal_Length: string, Petal_Width: string, Species: string
Sort [Sepal_Length#137 ASC NULLS FIRST], true
+- Relation
[Sepal_Length#137,Sepal_Width#138,Petal_Length#139,Petal_Width#140,Species#141]
csv

== Optimized Logical Plan ==
Sort [Sepal_Length#137 ASC NULLS FIRST], true
+- Relation
[Sepal_Length#137,Sepal_Width#138,Petal_Length#139,Petal_Width#140,Species#141]
csv

== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- Sort [Sepal_Length#137 ASC NULLS FIRST], true, 0
   +- Exchange rangepartitioning(Sepal_Length#137 ASC NULLS FIRST, 200),
ENSURE_REQUIREMENTS, [plan_id=538]
```

```

+- FileScan csv
[Sepal_Length#137,Sepal_Width#138,Petal_Length#139,Petal_Width#140,Species#141]
Batched: false, DataFilters: [], Format: CSV, Location: InMemoryFileIndex(1
paths)[file:/c:/Users/emada/Downloads/DSE Stream
Training/Pyspark/iris/iris.csv], PartitionFilters: [], PushedFilters: [],
ReadSchema: struct<Sepal_Length:string,Sepal_Width:string,Petal_Length:string,Pe
tal_Width:string,Species:string>

```

0.1.2 Handling Null Values

```
[57]: # count nulls in a column
from pyspark.sql.functions import col
iris1_df1.filter(col("Sepal_Length").isNull()).show()

# not null
iris1_df1.filter(col("Sepal_Length").isNotNull()).show(10)

# drop rows with nulls

# drop rows having any null
iris_df1.na.drop()

# drop rows only if all columns are null
iris1_df1.na.drop(how="all")

# drop rows if null in specific columns
iris_df1=iris1_df1.na.drop(subset=["Sepal_Length","Species"])
```

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

only showing top 10 rows

```
[60]: # fill null values

# fill all numeric nulls with 0
iris1_df1.na.fill(0)

# fill specific columns
iris1_df1.na.fill({"Sepal_Length":0.0,"Species":"unknown"}).show(5) # replaces nulls with these
```

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa

only showing top 5 rows

```
[72]: # replace values(Not only nulls)
iris1_df1.na.replace("setosa","SETOSA").show(5)

# multiple
# replace in Species
from pyspark.sql.functions import when, col

iris1_df1 = iris1_df1.withColumn(
    "Species", when(col("Species") == "setosa", "SETOSA").
    otherwise(col("Species")))
)

iris1_df1 = iris1_df1.withColumn(
    "Sepal_Length",
    when(col("Sepal_Length") == "1.3", "10.0").otherwise(col("Sepal_Length")),
)

iris1_df1.show(5)
```

Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species
5.1	3.5	1.4	0.2	SETOSA
4.9	3.0	1.4	0.2	SETOSA
4.7	3.2	1.3	0.2	SETOSA
4.6	3.1	1.5	0.2	SETOSA
5.0	3.6	1.4	0.2	SETOSA

```

+-----+-----+-----+-----+
only showing top 5 rows
+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+
|      5.1|      3.5|      1.4|      0.2| SETOSA|
|      4.9|      3.0|      1.4|      0.2| SETOSA|
|      4.7|      3.2|      1.3|      0.2| SETOSA|
|      4.6|      3.1|      1.5|      0.2| SETOSA|
|      5.0|      3.6|      1.4|      0.2| SETOSA|
+-----+-----+-----+-----+
only showing top 5 rows

```

```
[73]: # nulls us empty strings
# df.filter(col("Species")=="")
from pyspark.sql.functions import when
iris_df1=iris_df1.withColumn(
    "Species",
    when(col("Species")=="",None).otherwise(col("Species"))
)
```

```
[30]: # joining two tables where the joining columns present in the two
# tables have a different name
iris1_df1.join(other=iris1_df2,on=(iris1_df1.ID==iris1_df2.ID),how='inner').
show()
```

```

+-----+-----+-----+-----+-----+
|Sepal_Length|Sepal_Width| ID| ID|Petal_Length|Petal_Width|Species|
+-----+-----+-----+-----+-----+
|      5.1|      3.5|  1|  1|      1.4|      0.2| setosa|
|      4.9|      3.0|  3|  2|      1.4|      0.2| setosa|
|      4.7|      3.2|  3|  3|      1.3|      0.2| setosa|
|      4.6|      3.1|  4|  4|      1.5|      0.2| setosa|
|      5|      3.6|  5|  5|      1.4|      0.2| setosa|
|      5.4|      3.9|  6|  6|      1.7|      0.4| setosa|
|      4.6|      3.4|  7|  7|      1.4|      0.3| setosa|
|      5|      3.4|  8|  8|      1.5|      0.2| setosa|
|      4.4|      2.9|  9|  9|      1.4|      0.2| setosa|
|      4.9|      3.1| 10| 10|      1.5|      0.1| setosa|
|      5.4|      3.7| 11| 11|      1.5|      0.2| setosa|
|      4.8|      3.4| 12| 12|      1.6|      0.2| setosa|
|      4.8|      3| 13| 13|      1.4|      0.1| setosa|
|      4.3|      3| 14| 14|      1.1|      0.1| setosa|
|      5.8|      4| 15| 15|      1.2|      0.2| setosa|
|      5.7|      4.4| 16| 16|      1.5|      0.4| setosa|
|      5.4|      3.9| 17| 17|      1.3|      0.4| setosa|
|      5.1|      3.5| 18| 18|      1.4|      0.3| setosa|
|      5.7|      3.8| 19| 19|      1.7|      0.3| setosa|

```

	5.1	3.8	20	20	1.5	0.3	setosa
+	-	-	-	-	-	-	-

only showing top 20 rows