

Reading data from a file

```
df = spark.read.csv("file.csv", header=True)
df = spark.read.option("header", "true").csv("file.csv", header=True)
```

Casting a column to a different data type

```
df.withColumn("col1", col("col1").cast("double"))
```

Displaying the schema of a Dataframe

```
df.printSchema()
```

Get distinct count of columns

```
df.select("col").distinct().count()
```

Filtering rows based on a condition

```
# Filter entries of age, only keep those records of
# which the values are >24
df.filter(df["age"] > 24).show()
```

Renaming columns of DataFrame

```
#Syntax
df.withColumnRenamed("old_name", "new_name")
#Example
df = df.withColumnRenamed('CallNumber', 'Phone Number')
```

Inspect Data

```
# Return first n rows
df.head()
# Return first row
df.first()
# Return the first n rows
df.take(2)
# Print the schema of df
df.printSchema()
# Print the (logical and physical) plans
df.explain()
#Get All column names from DataFrame
df.columns
```

Get count

```
# Get_row count
rows = empDF.count()
# Get_columns count
cols = len(empDF.columns)
```

Selecting specific columns of a Dataframe

```
df.select("col1", "col2").show()
# Select All columns
df.select("*").show()
```

Full content of the columns without truncation

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

Handling missing or null values

```
# Fill all null values with 0
df.fillna(0)
#Fill specific columns with specified values
df.fillna({'col1': 0, "col2": "missing"})
```

Joining two dataframes

```
#syntax:
joined_df = df1.join(df2, on="key_column", how="inner")
#example
joined_df = empDF.join(deptDF, empDF.employee_id == deptDF.dept_id, "inner") \
    .show(truncate=False)
```

Adding a new column to a DataFrame

```
df.withColumn("new_col", col("col1") + col("col2"))
```

Dropping columns from a Dataframe

```
df.drop("col1")
```



By shivprasadgadekar

Published 17th March, 2023.
Last updated 17th March, 2023.
Page 1 of 2.

Sponsored by [Readable.com](https://readable.com)
Measure your website readability!
<https://readable.com>

Grouping data by a colm and agg. with a function

```
#syntax
df.groupBy("column").agg({"col2": "mean"})
#examples
df.groupBy("department") \
    .agg(sum("salary").alias("sum_salary"), \
         avg("salary").alias("avg_salary"), \
         sum("bonus").alias("sum_bonus"), \
         max("bonus").alias("max_bonus")) \
    .show(truncate=False)
```

Stopping the SparkSession

```
spark.stop()
```



By **shivprasadgadekar**

Published 17th March, 2023.

Last updated 17th March, 2023.

Page 2 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>