Ex. No. 5	**Creating DataFrames in Spark SQL**
Youtube Link	**https://youtu.be/Ny4mBOGDHe4**
Date of Exercise	20.10.25

AIM

To learn various methods of creating DataFrames in Apache Spark using Spark SQL and the DataFrame API.

Procedure:

1. Setup

1. Initialize a SparkSession:
    - **Scala/Java**: val spark = SparkSession.builder().appName("CreateDF").getOrCreate()
    - **PySpark**: from pyspark.sql import SparkSession; spark = SparkSession.builder.appName("CreateDF").getOrCreate() GeeksforGeeks

2. Create DataFrames from Collections / RDDs

2.1 From a list or tuple

- PySpark example:

```
data = [("Alice", 1), ("Bob", 2)]

df = spark.createDataFrame(data, ["name","age"])

df.show()
```

2.2 From RDD

```
val rdd = sc.parallelize(Seq(("John",25),("Mary",30)))

import spark.implicits._

val df = rdd.toDF("name","age")

df.show()
```

2.4 Or using createDataFrame(rdd, schema) with explicit schema definitions (StructType, StructField) Wikipedia+15Baeldung+15GeeksforGeeks+15

C. From External Files

df_csv = spark.read.csv("people.csv", header=True, inferSchema=True)

df_csv.show()

df_csv.printSchema()

- JSON similarly via .read.json(...) or .read.json(..., multiLine=True) Reddit+3Analytics Vidhya+3Wikipedia+3

D. Inspecting the DataFrame

- Use .show(), .printSchema() to display rows and schema Analytics Vidhya+4Apache Spark+4Apache Spark+4

E. Registering and Querying via SQL

- Register table/view:

df.createOrReplaceTempView("people")

spark.sql("SELECT age, count(*) FROM people GROUP BY age").show()

PySpark as well and in Spark 3.4+, DFs can be queried directly via named parameters in SQL API Reddit+13Reddit+13Apache Spark+13

Program:

```
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("CreateDF").getOrCreate()
# From list of tuples
data = [("John", 28), ("Alice", 30), ("Bob", 25)]
df1 = spark.createDataFrame(data, ["name", "age"])
df1.show(); df1.printSchema()

# From RDD with schema
from pyspark.sql.types import StructType, StructField, StringType, IntegerType
rdd = spark.sparkContext.parallelize([("John", 25), ("Mary", 30)])
schema = StructType([StructField("name", StringType(), True), StructField("age", IntegerType(), True)])
df2 = spark.createDataFrame(rdd.map(lambda x: (x[0], x[1])), schema)
df2.show(); df2.printSchema()

# From CSV
dfcsv = spark.read.csv("people.csv", header=True, inferSchema=True)
dfcsv.show(); dfcsv.printSchema()

df1.createOrReplaceTempView("people")
spark.sql("SELECT age, COUNT(*) AS cnt FROM people GROUP BY age").show()

spark.stop()
```

Output:

```
+-----+---+
| name|age|
```

```
+-----+---+
| John| 28|
|Alice| 30|
|  Bob| 25|
| Maria| 27|
|  ...|
+-----+---+

root
 |-- name: string (nullable = true)
 |-- age: integer (nullable = true)

+----+---+-------------+
|name|age|       city  |
+----+---+-------------+
|John| 28|     New York|
|Alice|30|San Francisco|
|... |...|         ... |
+----+---+-------------+

root
 |-- name: string (nullable = true)
 |-- age: integer (nullable = true)
 |-- city: string (nullable = true)
```

```
+---+---+
|age|cnt|
+---+---+
| 28|  1|
| 30|  1|
| 25|  1|
| ...|
+---+---+
```

Result :

The Program was executed Succesfully.