PySpark Cheat Sheet for Data Engineers

✅ 1. Quickstart

Explanation: PySpark is the Python API for Apache Spark. Start a session using `SparkSession`.

Example:

```
from pyspark.sql import SparkSessionspark =
SparkSession.builder.appName("MyApp").getOrCreate()
```

✅ 2. Basics

Explanation: Create DataFrames from lists, CSV, JSON, or Parquet files.

Example:

```
# From listdata = [(1, "Alice"), (2, "Bob")]df =
spark.createDataFrame(data, ["id", "name"])# From CSVcsv_df =
spark.read.csv("path/file.csv", header=True, inferSchema=True)
```

✅ 3. Common Patterns

Explanation: View schema, show top records, and print DataFrame structure.

Example:

```
df.show()df.printSchema()df.describe().show()df.columns
```

✅ 4. Importing Functions & Types

Explanation: Use `pyspark.sql.functions` and `pyspark.sql.types` for most operations.

Example:

```
from pyspark.sql.functions import col, lit, whenfrom pyspark.sql.types import IntegerType, StringType
```

✅ 5. Filtering

Explanation: Filter rows using conditions.

Example:

```
df.filter(col("id") > 1).show()df.where(col("name") == "Alice").show()
```

✅ 6. Joins

Explanation: Combine DataFrames using join conditions.

Example:

```
df1.join(df2, df1.id == df2.id, "inner").show()df1.join(df2, on="id", how="left").show()
```

✅ 7. Column Operations

Explanation: Add, rename, or drop columns.

Example:

```
df = df.withColumn("new_col", col("id") + 10)df = df.withColumnRenamed("name", "full_name")df = df.drop("new_col")
```

✅ 8. Casting & Coalescing, Null Values & Duplicates

Explanation: Handle data types, nulls, and duplicates.

Example:

```
df = df.withColumn("id", col("id").cast(StringType()))df = df.fillna({"name": "Unknown"})df = df.dropDuplicates()
```

✅ 9. String Operations

Explanation: String manipulations like lower, upper, and trimming.

Example:

```
from pyspark.sql.functions import upper, lower, trimdf =
df.withColumn("name_upper", upper(col("name")))df =
df.withColumn("name_trimmed", trim(col("name")))
```

✅ 10. String Filters

Explanation: Use `contains`, `startswith`, or `endswith`.

Example:

```
df.filter(col("name").contains("Ali")).show()df.filter(col("name").sta
rtswith("A")).show()
```

✅ 11. String Functions

Explanation: Use built-in functions to manipulate strings.

Example:

```
from pyspark.sql.functions import concat_wsdf =
df.withColumn("greeting", concat_ws(" ", lit("Hello"), col("name")))
```

✅ 12. Number Operations

Explanation: Perform arithmetic operations on numeric columns.

Example:

```
df = df.withColumn("id_squared", col("id") ** 2)
```

✅ 13. Date & Timestamp Operations

Explanation: Work with dates, extract year, month, etc.

Example:

```
from pyspark.sql.functions import current_date, year, monthdf =
df.withColumn("today", current_date())df = df.withColumn("year",
year(col("today")))
```

✅ 14. Array Operations

Explanation: Explode, split, or manipulate array columns.

Example:

```
from pyspark.sql.functions import split, explodedf =
df.withColumn("words", split(col("name"), " "))df =
df.withColumn("word", explode(col("words")))
```

✅ 15. Struct Operations

Explanation: Work with nested fields.

Example:

```
from pyspark.sql.functions import structdf = df.withColumn("info",
struct("id", "name"))df.select("info.id", "info.name").show()
```

✅ 16. Aggregation Operations

Explanation: Use `groupBy` and aggregation functions.

Example:

```
from pyspark.sql.functions import count,
avgdf.groupBy("name").agg(count("id"), avg("id")).show()
```

✅ 17. Advanced Operations

Explanation: Use `window` functions for row-wise operations.

Example:

```
from pyspark.sql.window import Windowfrom pyspark.sql.functions import row_numberwindowSpec = Window.partitionBy("name").orderBy("id")df = df.withColumn("row_num", row_number().over(windowSpec))
```

✅ 18. Repartitioning

Explanation: Optimize parallelism using repartition or coalesce.

Example:

```
df = df.repartition(4)df = df.coalesce(1)
```

✅ 19. UDFs (User Defined Functions)

Explanation: Define custom logic not supported natively by Spark.

Example:

```
from pyspark.sql.functions import udffrom pyspark.sql.types import StringTypedef greeting(name):    return f"Hi {name}"greet_udf = udf(greeting, StringType())df = df.withColumn("greet", greet_udf(col("name")))
```

✅ 20. Useful Functions / Transformations

Explanation: Random, monotonically increasing ID, hash, etc.

Example:

```
from pyspark.sql.functions import rand, monotonically_increasing_iddf = df.withColumn("random_num", rand())df = df.withColumn("unique_id", monotonically_increasing_id())
```

🔥 Let’s build your Data Engineering journey together!

📞 Ready to Upskill?

Want to achieve the same results? Let’s get started:

📩 Call us directly at: 9989454737

📥 Join the info group here: **https://lnkd.in/gkVevx5w**

🌐 Visit: **https://lnkd.in/gfDiU3sk**