Master PySpark: From Zero to Big Data Hero!!

Date Function in Dataframe – Part 1

In PySpark, you can use various date functions to manipulate and analyze date and timestamp columns. Below, I'll provide a sample dataset and demonstrate key date functions like current_date, current_timestamp, date_add, date_sub, datediff, and months_between.

Code Explanation with Notes

1. **Creating a Spark Session:**
    - We begin by creating a Spark session to run the PySpark operations.
2. **Generating a DataFrame:**
    - Using spark.range(10) creates a DataFrame with 10 rows and a single column (id) with numbers ranging from 0 to 9.
    - Two additional columns are added:
        - **today:** Contains the current date using current_date().
        - **now:** Contains the current timestamp using current_timestamp().
3. **Date Manipulation Functions:**
    - **date_add:** Adds a specified number of days to the date.
    - **date_sub:** Subtracts a specified number of days from the date.
    - **datediff:** Returns the difference in days between two dates.
    - **months_between:** Returns the number of months between two dates.

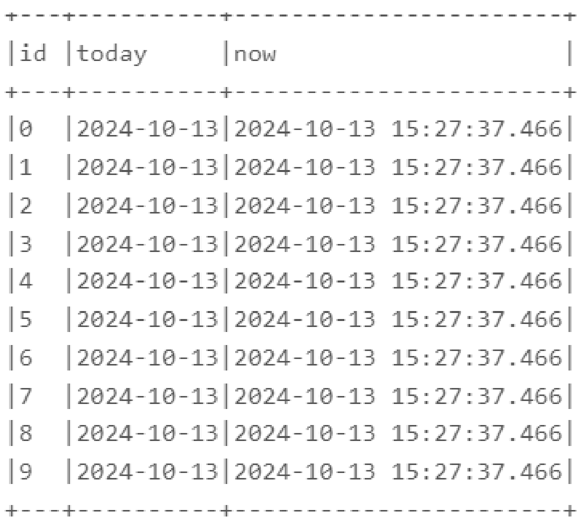
Code Example

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import current_date, current_timestamp,
date_add, date_sub, col, datediff, months_between, to_date, lit

# Generate a DataFrame with 10 rows, adding "today" and "now"
columns
dateDF = spark.range(10).withColumn("today",
current_date()).withColumn("now", current_timestamp())

# Show the DataFrame with today and now columns
dateDF.show(truncate=False)
```

Follow me on LinkedIn – Shivakiran kotur



```
+---+----------+-----------------------+
|id |today     |now                    |
+---+----------+-----------------------+
|0  |2024-10-13|2024-10-13 15:27:37.466|
|1  |2024-10-13|2024-10-13 15:27:37.466|
|2  |2024-10-13|2024-10-13 15:27:37.466|
|3  |2024-10-13|2024-10-13 15:27:37.466|
|4  |2024-10-13|2024-10-13 15:27:37.466|
|5  |2024-10-13|2024-10-13 15:27:37.466|
|6  |2024-10-13|2024-10-13 15:27:37.466|
|7  |2024-10-13|2024-10-13 15:27:37.466|
|8  |2024-10-13|2024-10-13 15:27:37.466|
|9  |2024-10-13|2024-10-13 15:27:37.466|
+---+----------+-----------------------+
```

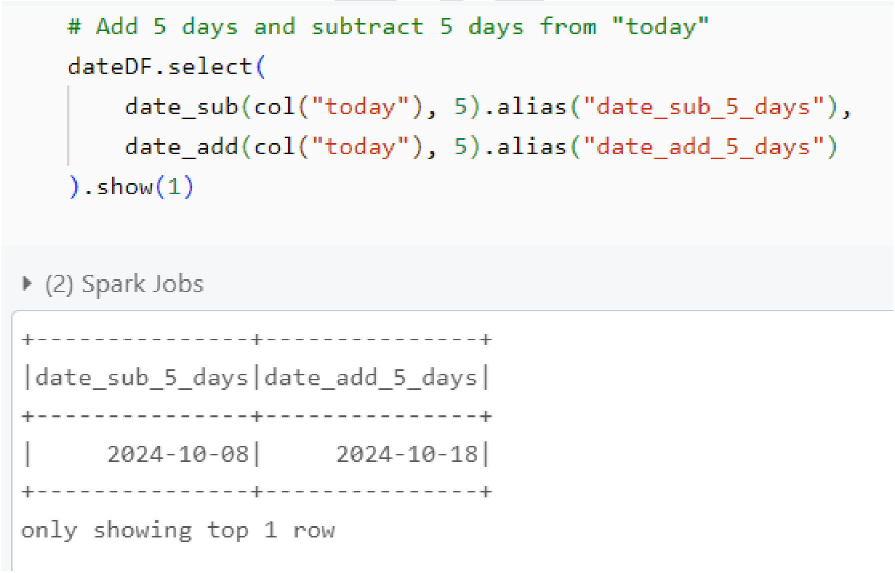
Explanation of Code and Output

1. current_date and current_timestamp:

- current_date() gives the current date (e.g., 2024-10-12).
- current_timestamp() provides the current timestamp, which includes both date and time (e.g., 2024-10-12 12:34:56).
- These are used to create columns today and now in the DataFrame.

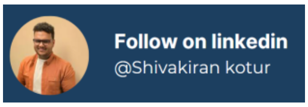
2. date_add and date_sub:

- **date_sub(col("today"), 5):** Subtracts 5 days from the current date, so if today is 2024-10-12, it returns 2024-10-07.
- **date_add(col("today"), 5):** Adds 5 days to the current date, returning 2024-10-17.



```
# Add 5 days and subtract 5 days from "today"
dateDF.select(
    date_sub(col("today"), 5).alias("date_sub_5_days"),
    date_add(col("today"), 5).alias("date_add_5_days")
).show(1)
```

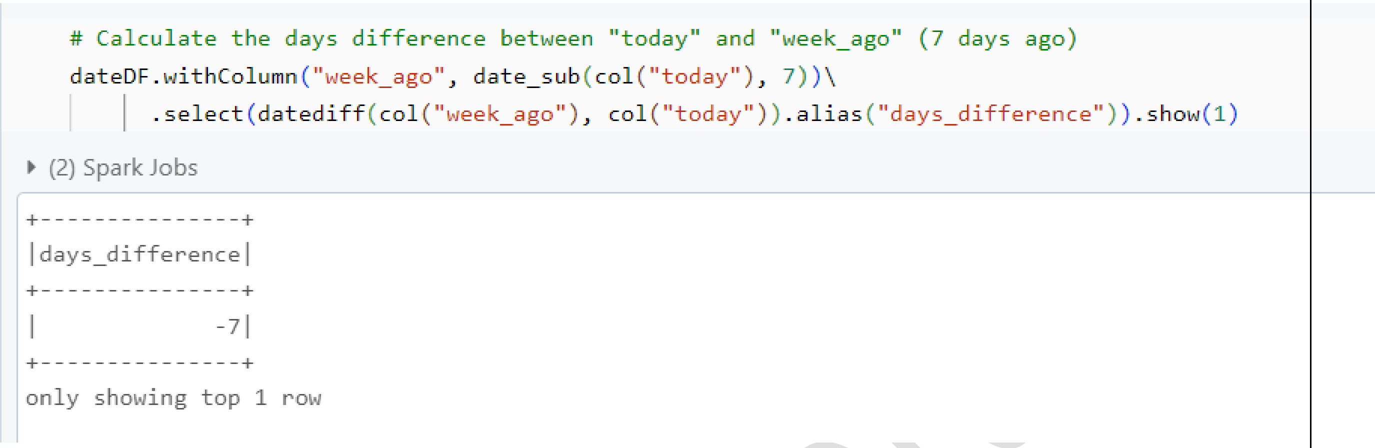
```
+---------------+---------------+
|date_sub_5_days|date_add_5_days|
+---------------+---------------+
|     2024-10-08|     2024-10-18|
+---------------+---------------+
only showing top 1 row
```



Follow me on LinkedIn – Shivakiran kotur

3. datediff:

- **datediff(col("week_ago"), col("today")):** Calculates the difference in days between the current date and 7 days ago (i.e., -7).

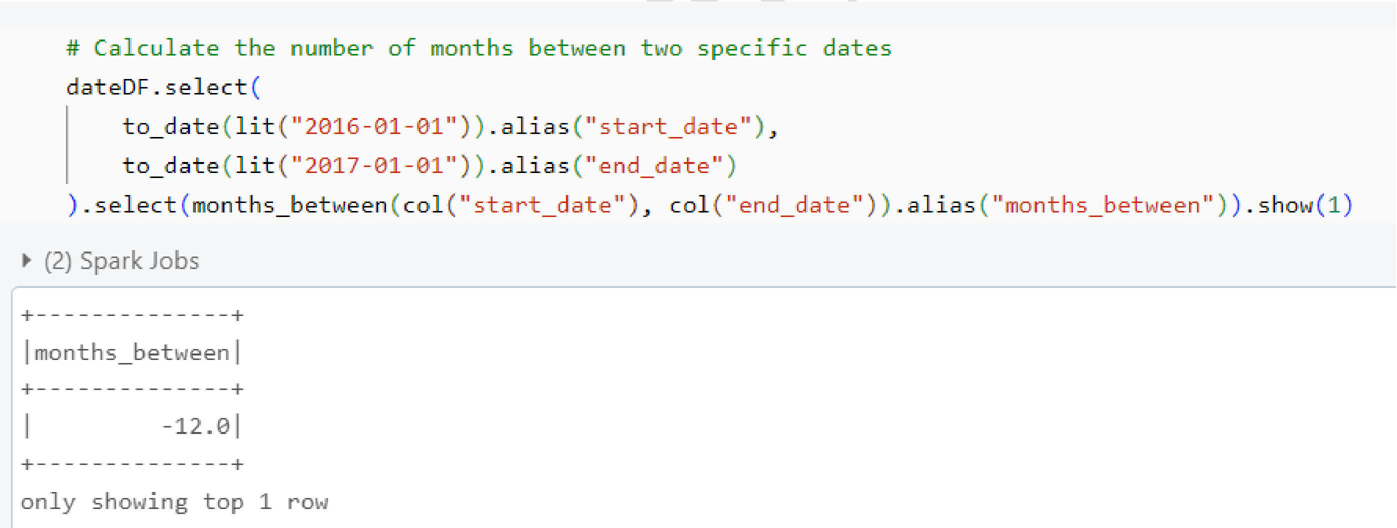


```
# Calculate the days difference between "today" and "week_ago" (7 days ago)
dateDF.withColumn("week_ago", date_sub(col("today"), 7))\
    .select(datediff(col("week_ago"), col("today")).alias("days_difference")).show(1)
```

```
+---------------+
|days_difference|
+---------------+
|             -7|
+---------------+
only showing top 1 row
```

4. months_between:

- **months_between(to_date(lit("2016-01-01")), to_date(lit("2017-01-01"))):** Calculates the number of months between January 1, 2016, and January 1, 2017, which is -12 months because start_date is earlier than end_date.



```
# Calculate the number of months between two specific dates
dateDF.select(
    to_date(lit("2016-01-01")).alias("start_date"),
    to_date(lit("2017-01-01")).alias("end_date")
).select(months_between(col("start_date"), col("end_date")).alias("months_between")).show(1)
```

```
+--------------+
|months_between|
+--------------+
|         -12.0|
+--------------+
only showing top 1 row
```

Follow me on LinkedIn – Shivakiran kotur