How to Extract the Hour from a Timestamp in PySpark

Authored by

stats writer

January 19, 2026

In PySpark, the hour can be extracted from a timestamp by using the "hour" function, which returns an integer representing the hour of the day from a given timestamp. This function takes the timestamp as an input and extracts the hour value, making it easy to manipulate and analyze time-related data in PySpark. Additionally, the "hour" function can be used in conjunction with other functions, such as "date_trunc" or "from_unixtime," to further refine the extracted hour value based on specific criteria.

You can use the following methods to extract the hour from a timestamp in PySpark:

Method 1: Extract Hour from Timestamp

from pyspark.sql import functions as F

df_new = df.withColumn('hour', F.hour(df))

If the timestamp is **2023-01-15 04:14:22** then this syntax would return **4**.

Method 2: Extract Timestamp Truncated to Hour

from pyspark.sql import functions as F

df_new = df.withColumn('hour', F.date_trunc('hour', df))

If the timestamp is **2023-01-15 04:14:22** then this syntax would return **2023-01-15 04:00:00**.

The following example shows how to use each method in practice with the following PySpark DataFrame that contains information about sales made on various timestamps at some company:

from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()

from pyspark.sql import functions as F

#define data
data = ,
,
,
]

#define column names
columns =

```
#create dataframe using data and column names
df = spark.createDataFrame(data, columns)

#convert string column to timestamp
df = df.withColumn('ts', F.to_timestamp('ts', 'yyyy-MM-dd HH:mm:ss'))

#view dataframe
df.show()

+-------------------+-----+
| ts|sales|
+-------------------+-----+
|2023-01-15 04:14:22| 225|
|2023-02-24 10:55:01| 260|
|2023-07-14 18:34:59| 413|
|2023-10-30 22:20:05| 368|
+-------------------+-----+
```

Example 1: Extract Hour from Timestamp

We can use the following syntax to extract only the hour from each timestamp in the **ts** column of the DataFrame:

```
from pyspark.sql import functions as F

#extract hour from each timestamp in 'ts' column
df_new = df.withColumn('hour', F.hour(df))

#view new DataFrame
df_new.show()

+-------------------+-----+----+
| ts|sales|hour|
+-------------------+-----+----+
|2023-01-15 04:14:22| 225| 4|
|2023-02-24 10:55:01| 260| 10|
|2023-07-14 18:34:59| 413| 18|
|2023-10-30 22:20:05| 368| 22|
+-------------------+-----+----+
```

The new **hour** column shows only the hour from each timestamp in the **ts** column.

Example 2: Extract Timestamp Truncated to Hour

We can use the following syntax to return each timestamp from the **ts** column truncated to the hour:

```
from pyspark.sql import functions as F

#create new column that contains timestamp truncated to the hour
df_new = df.withColumn('hour', F.date_trunc('hour', df))

#view new DataFrame
df_new.show()
```

```
+-------------------+-----+-------------------+
| ts|sales| hour|
+-------------------+-----+-------------------+
|2023-01-15 04:14:22| 225|2023-01-15 04:00:00|
|2023-02-24 10:55:01| 260|2023-02-24 10:00:00|
|2023-07-14 18:34:59| 413|2023-07-14 18:00:00|
|2023-10-30 22:20:05| 368|2023-10-30 22:00:00|
+-------------------+-----+-------------------+
```

The new **hour** column shows each timestamp from the **ts** column truncated to the hour.

The following tutorials explain how to perform other common tasks in PySpark: