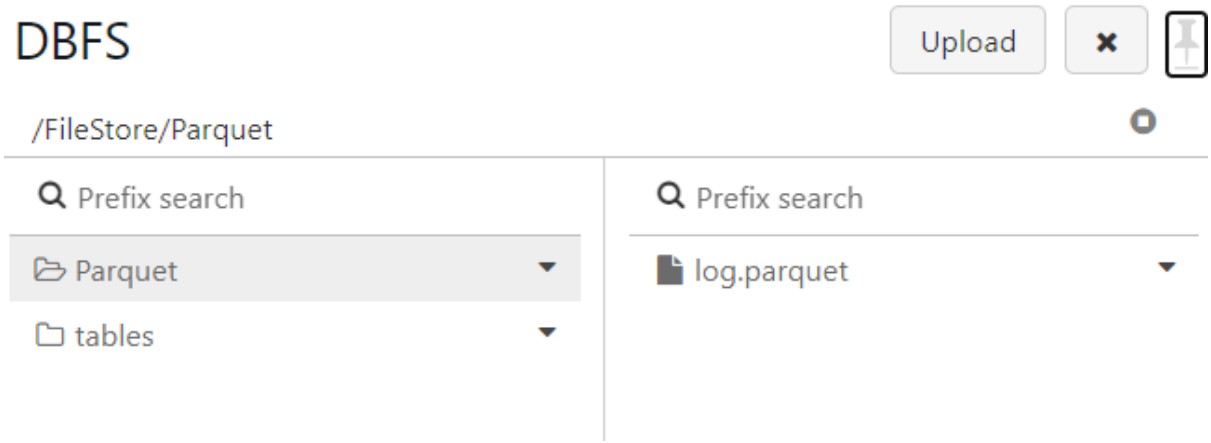
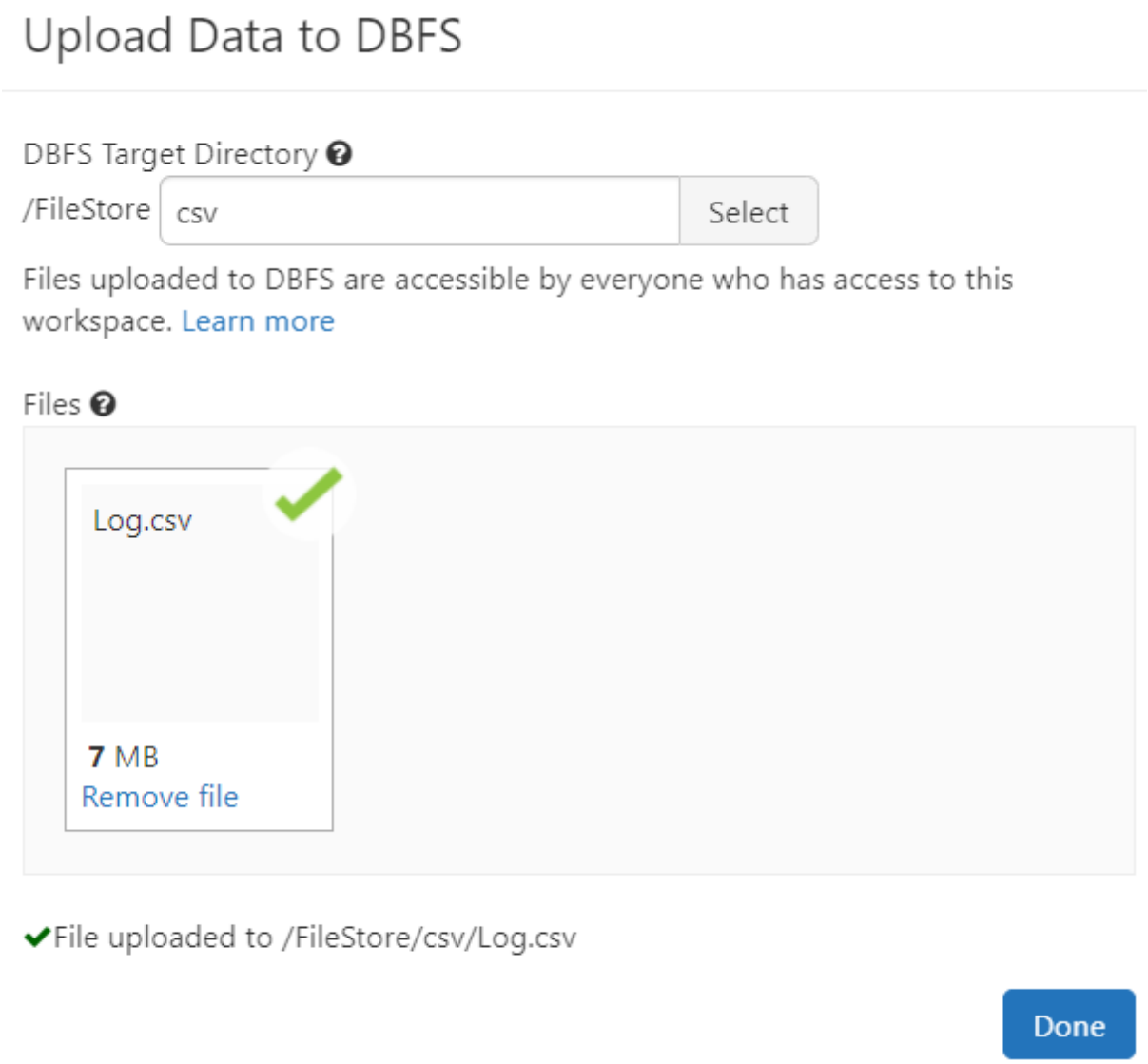
😊 Uploading CSV file

1. In this lab we are going to upload our CSV file onto our data bricks workspace.
2. For that you need to go to Catalog and click and browse DBFS. Inside DBFS you have to click on upload.



3. Then you should name your directory and upload your CSV file in it.



4. Then you need to delete you previous cells and in a new cell paste this code and run it. By this, we have read the data in our CSV file.

Just now (1s) 1 Scala

```scala
import org.apache.spark.sql.types._
import org.apache.spark.sql.functions._

val file_location = "/FileStore/csv/Log.csv"
val file_type = "csv"

val dataSchema = StructType(Array(
    StructField("Correlationid", StringType, true),
    StructField("Operationname", StringType, true),
    StructField("Status", StringType, true),
    StructField("Eventcategory", StringType, true),
    StructField("Level", StringType, true),
    StructField("Time", TimestampType, true),
    StructField("Subscription", StringType, true),
    StructField("Eventinitiatedby", StringType, true),
    StructField("Resourcetype", StringType, true),
    StructField("Resourcegroup", StringType, true),
    StructField("Resource", StringType, true)))


val df = spark.read.format(file_type).
options(Map("header"->"true")).
schema(dataSchema).
load(file_location)
```

```
df: org.apache.spark.sql.DataFrame = [Correlationid: string, Operationname: string ... 9 more fields]
import org.apache.spark.sql.types._
import org.apache.spark.sql.functions._
file_location: String = /FileStore/csv/Log.csv
file_type: String = csv
dataSchema: org.apache.spark.sql.types.StructType = StructType(StructField(Correlationid,StringType,true),StructField(Operationname,StringType,tr
ue),StructField(Status,StringType,true),StructField(Eventcategory,StringType,true),StructField(Level,StringType,true),StructField(Time,TimestampT
ype,true),StructField(Subscription,StringType,true),StructField(Eventinitiatedby,StringType,true),StructField(Resourcetype,StringType,true),Struc
```

5. Now we use this code below in a new cell to see the dates from our data.

Just now (3s) 2 Scala

```scala
import org.apache.spark.sql.functions._
display(df.select(year(col("time")),month(col("time")),dayofyear(col("time"))))
```

(1) Spark Jobs

Table + New result table: ON

	year(time)	month(time)	dayofyear(time)
1	2023	4	115
2	2023	4	115
3	2023	4	115
4	2023	4	115
5	2023	4	115
6	2023	4	115
7	2023	4	115
8	2023	4	115
9	2023	4	115
10	2023	4	115
11	2023	4	115
12	2023	4	115
13	2023	4	115
14	2023	4	115
15	2023	4	115

10,000+ rows | Truncated data | 3.20 seconds runtime Refreshed now

6. Now this code below shows our data in more meaningful way.

Just now (1s) 3 Scala

```
display(df.select(year(col("time")).alias("Year"),month(col("time")).alias("Month"),dayofyear(col("time")).alias("Day of Year")))
```

(1) Spark Jobs

Table + New result table: ON

	Year	Month	Day of Year
1	2023	4	115
2	2023	4	115
3	2023	4	115
4	2023	4	115
5	2023	4	115
6	2023	4	115
7	2023	4	115
8	2023	4	115
9	2023	4	115
10	2023	4	115
11	2023	4	115
12	2023	4	115
13	2023	4	115
14	2023	4	115
15	2023	4	115

10,000+ rows | Truncated data | 1.02 seconds runtime Refreshed now

7. If you want to convert the date to a particular format.

Just now (1s) 4

```
display(df.select(to_date(col("time"),"dd-mm-yyyy").alias("Date")))
```

(1) Spark Jobs

Table

	Date
1	2023-04-25
2	2023-04-25
3	2023-04-25
4	2023-04-25
5	2023-04-25
6	2023-04-25
7	2023-04-25
8	2023-04-25
9	2023-04-25
10	2023-04-25
11	2023-04-25
12	2023-04-25
13	2023-04-25
14	2023-04-25
15	2023-04-25

10,000+ rows | Truncated data | 0.92 seconds runtime