pharma

November 13, 2024

0.1 Overview

This notebook will show you how to create and query a table or DataFrame that you uploaded to DBFS. DBFS is a Databricks File System that allows you to store data for querying inside of Databricks. This notebook assumes that you have a file already inside of DBFS that you would like to read from.

This notebook is written in **Python** so the default cell type is Python. However, you can use different languages by using the `%LANGUAGE` syntax. Python, Scala, SQL, and R are all supported.

[ ]:
```python
# Define file locations
file_location_daily = "/FileStore/tables/salesdaily.csv"
file_location_hourly = "/FileStore/tables/saleshourly.csv"
file_location_weekly = "/FileStore/tables/salesweekly.csv"
file_location_monthly = "/FileStore/tables/salesmonthly.csv"

file_type = "csv"

# CSV options
infer_schema = "true"
first_row_is_header = "true"
delimiter = ","

# Load each dataset into a separate DataFrame
df_daily = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location_daily)

df_hourly = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location_hourly)

df_weekly = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
```

```
    .option("sep", delimiter) \
    .load(file_location_weekly)

df_monthly = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location_monthly)

# Display a sample of each dataset
display(df_daily)
display(df_hourly)
display(df_weekly)
display(df_monthly)
```

[ ]:
```
# Print schemas to understand the structure
df_daily.printSchema()
df_hourly.printSchema()
df_weekly.printSchema()
df_monthly.printSchema()

# Show basic statistics
df_daily.describe().show()
df_hourly.describe().show()
df_weekly.describe().show()
df_monthly.describe().show()

# Check for missing values
from pyspark.sql.functions import col, sum

# Example for daily sales data
df_daily.select([sum(col(c).isNull().cast("int")).alias(c) for c in df_daily.
 ↪columns]).show()
```

```
root
 |-- datum: date (nullable = true)
 |-- M01AB: double (nullable = true)
 |-- M01AE: double (nullable = true)
 |-- N02BA: double (nullable = true)
 |-- N02BE: double (nullable = true)
 |-- N05B: double (nullable = true)
 |-- N05C: double (nullable = true)
 |-- R03: double (nullable = true)
 |-- R06: double (nullable = true)
 |-- Year: integer (nullable = true)
 |-- Month: integer (nullable = true)
 |-- Hour: integer (nullable = true)
```

```
 |-- Weekday Name: string (nullable = true)

root
 |-- datum: string (nullable = true)
 |-- M01AB: double (nullable = true)
 |-- M01AE: double (nullable = true)
 |-- N02BA: double (nullable = true)
 |-- N02BE: double (nullable = true)
 |-- N05B: double (nullable = true)
 |-- N05C: double (nullable = true)
 |-- R03: double (nullable = true)
 |-- R06: double (nullable = true)
 |-- Year: integer (nullable = true)
 |-- Month: integer (nullable = true)
 |-- Hour: integer (nullable = true)
 |-- Weekday Name: string (nullable = true)

root
 |-- datum: date (nullable = true)
 |-- M01AB: double (nullable = true)
 |-- M01AE: double (nullable = true)
 |-- N02BA: double (nullable = true)
 |-- N02BE: double (nullable = true)
 |-- N05B: double (nullable = true)
 |-- N05C: double (nullable = true)
 |-- R03: double (nullable = true)
 |-- R06: double (nullable = true)

root
 |-- datum: date (nullable = true)
 |-- M01AB: double (nullable = true)
 |-- M01AE: double (nullable = true)
 |-- N02BA: double (nullable = true)
 |-- N02BE: double (nullable = true)
 |-- N05B: double (nullable = true)
 |-- N05C: double (nullable = true)
 |-- R03: double (nullable = true)
 |-- R06: double (nullable = true)

+-------+-----------------+------------------+------------------+---------------
---+-----------------+------------------+----------------+------------------+--
---------------+-----------------+------------------+------------+
|summary|            M01AB|             M01AE|            N02BA|
N02BE|             N05B|             N05C|             R03|               R06|
Year|            Month|              Hour|Weekday Name|
+-------+-----------------+------------------+------------------+---------------
---+-----------------+------------------+----------------+------------------+--
---------------+-----------------+------------------+------------+
```

```
|  count|             2106|             2106|             2106|
2106|             2106|             2106|             2106|             2106|
2106|             2106|             2106|        2106|
|   mean|5.033683325419752|
3.895830316160029|3.8804411206082676|29.917095303105373|
8.85362654319611|0.5935224754667617| 5.51226159386135|2.9001982431149083|2016.40
12345679012|6.344254510921178|275.94586894586894|        null|
| stddev|2.737578507424412|2.1333365992423245|2.3840102366211324|15.590965539762
397|5.605604746929916|1.0929883200525332|6.428736372389848|2.4158159046504153|1.
6650603678435227|3.386953835651899|1.9705466716604123|        null|
|    min|              0.0|              0.0|              0.0|
0.0|              0.0|              0.0|              0.0|              0.0|
2014|               1|              190|      Friday|
|    max|             17.34|             14.463|             16.0|
161.0|      54.83333333|              9.0|             45.0|              15.0|
2019|              12|              276|   Wednesday|
+-------+-----------------+-----------------+-----------------+---------------
---+-----------------+-----------------+-----------------+-----------------+--
---------------+-----------------+-----------------+------------+

+-------+-------------+-------------------+-------------------+----------------
--+-------------------+-------------------+-------------------+----------------
-+-------------------+-------------------+-----------------+-----------------+--
----------+
|summary|        datum|              M01AB|              M01AE|
N02BA|              N02BE|              N05B|              N05C|
R03|               R06|             Year|            Month|
Hour|Weekday Name|
+-------+-------------+-------------------+-------------------+----------------
--+-------------------+-------------------+-------------------+----------------
-+-------------------+-------------------+-----------------+-----------------+--
----------+
|  count|        50532|              50532|              50532|
50532|              50532|              50532|              50532|
50532|              50532|              50532|            50532|
50532|       50532|
|   mean|         null|0.20978661211426386|  0.162364811324844|0.161723442571400
48|1.2468416589160307|0.36898871012455564|0.02473597588334124|0.2297321086967664
4|0.12087029011374939|2016.4014090081532|6.344811208739017|11.500474946568511|
null|
| stddev|         null|
0.5560026652182646|0.41610874069440673|0.45321122380532103|2.3873916844461154|
0.9309337244775395|0.21787064801445225| 1.2405127649551182|
0.3919993906592702|1.6644443550164936|3.385761229604851| 6.921706105317404|
null|
|    min|1/1/2015 0:00|               0.0|               0.0|
0.0|               0.0|               0.0|               0.0|
0.0|               0.0|              2014|               1|                 0|
```

```
Friday|
|    max|9/9/2019 9:00|               7.0|               6.0|
6.5|              29.0|              15.0|               6.0|
25.0|               5.0|              2019|              12|
23|   Wednesday|
+-------+-------------+------------------+------------------+----------------
--+------------------+------------------+------------------+-----------------
-+------------------+-----------------+----------------+------------------+--
----------+

+-------+-----------------+-----------------+-----------------+--------------
---+------------------+-----------------+-----------------+------------------+
|summary|             M01AB|             M01AE|             N02BA|
N02BE|              N05B|              N05C|              R03|
R06|
+-------+-----------------+-----------------+-----------------+--------------
---+------------------+-----------------+-----------------+------------------+
|  count|              302|              302|              302|
302|               302|               302|               302|               302|
|
mean|35.102440673278174|27.167611410066222|27.06029470198675|208.62716128609253|
61.74085264877485|4.1389348785860935| 38.43981098238411|20.224561258245025|
| stddev| 8.617106216402073|7.0434911201017485|8.086458417593548| 76.06922114429
098|22.436969645571224|3.1292645522027094|22.900872676300924|11.381464397131417|
|    min|             7.67|             6.237|              3.5|
86.25|              18.0|               0.0|               2.0|
1.0|
|    max|             65.33|            53.571|            60.125|
546.899|             154.0|              17.0|             131.0|
65.0|
+-------+-----------------+-----------------+-----------------+--------------
---+------------------+-----------------+-----------------+------------------+

+-------+-----------------+-----------------+-----------------+--------------
----+-----------------+----------------+----------------+------------------+
|summary|             M01AB|             M01AE|             N02BA|
N02BE|              N05B|              N05C|              R03|               R06|
+-------+-----------------+-----------------+-----------------+--------------
----+-----------------+----------------+----------------+------------------+
|  count|               70|               70|               70|
70|               70|              70|              70|                70|
|   mean|149.99200000000005|116.51428571428576| 115.0208428571428|
892.5420714285715|262.11857142857144|17.84285714285714|         167.675|
86.66257142857144|
| stddev|31.485325341880444|27.889336387230223|31.245898650578052|338.8439082049
4855| 85.06093006790228|8.481242242696293|81.76797934186814|45.859335518098064|
|    min|              0.0|              0.0|              0.0|
0.0|               1.0|              0.0|              0.0|                0.0|
```

```
|
max|211.13000000000017|222.35100000000017|191.59999999999997|1856.8149999999994|
492.0|              50.0|              386.0|213.04000000000002|
+-------+------------------+------------------+------------------+-------------
----+------------------+-----------------+-----------------+------------------+

+-----+-----+-----+-----+-----+----+----+---+---+----+-----+----+------------+
|datum|M01AB|M01AE|N02BA|N02BE|N05B|N05C|R03|R06|Year|Month|Hour|Weekday Name|
+-----+-----+-----+-----+-----+----+----+---+---+----+-----+----+------------+
|    0|    0|    0|    0|    0|   0|   0|  0|  0|   0|    0|   0|           0|
+-----+-----+-----+-----+-----+----+----+---+---+----+-----+----+------------+
```

[ ]:
```python
from pyspark.sql.functions import to_date

# Convert 'datum' column to date type for daily dataset (repeat for other datasets)
df_daily = df_daily.withColumn("datum", to_date(df_daily["datum"], "yyyy-MM-dd"))
```

[ ]:
```python
# Fill missing values with 0
df_daily = df_daily.fillna(0)
```

[ ]:
```python
# Monthly total sales for each drug category in the daily dataset
monthly_sales = df_daily.groupBy("Year", "Month") \
    .sum("M01AB", "M01AE", "N02BA", "N02BE", "N05B", "N05C", "R03", "R06") \
    .orderBy("Year", "Month")
display(monthly_sales)
```

[ ]:
```python
# Weekly trend analysis
weekly_sales_trends = df_weekly.groupBy("datum").sum("M01AB", "M01AE", "N02BA", "N02BE", "N05B", "N05C", "R03", "R06")
display(weekly_sales_trends)
```

[ ]:
```python
from statsmodels.tsa.arima.model import ARIMA
import pandas as pd

# Convert to pandas DataFrame
category_df = df_daily.select("datum", "M01AB").toPandas()
category_df.set_index('datum', inplace=True)

# Fit ARIMA model
model = ARIMA(category_df['M01AB'], order=(1, 1, 1))  # You may need to adjust the order
model_fit = model.fit()

# Forecast
```

```
forecast = model_fit.forecast(steps=30)
print(forecast)
```

```
/databricks/python/lib/python3.9/site-
packages/statsmodels/tsa/base/tsa_model.py:471: ValueWarning: No frequency
information was provided, so inferred frequency D will be used.
  self._init_dates(dates, freq)
/databricks/python/lib/python3.9/site-
packages/statsmodels/tsa/base/tsa_model.py:471: ValueWarning: No frequency
information was provided, so inferred frequency D will be used.
  self._init_dates(dates, freq)
/databricks/python/lib/python3.9/site-
packages/statsmodels/tsa/base/tsa_model.py:471: ValueWarning: No frequency
information was provided, so inferred frequency D will be used.
  self._init_dates(dates, freq)
```

```
2019-10-09    5.485714
2019-10-10    5.501847
2019-10-11    5.501897
2019-10-12    5.501898
2019-10-13    5.501898
2019-10-14    5.501898
2019-10-15    5.501898
2019-10-16    5.501898
2019-10-17    5.501898
2019-10-18    5.501898
2019-10-19    5.501898
2019-10-20    5.501898
2019-10-21    5.501898
2019-10-22    5.501898
2019-10-23    5.501898
2019-10-24    5.501898
2019-10-25    5.501898
2019-10-26    5.501898
2019-10-27    5.501898
2019-10-28    5.501898
2019-10-29    5.501898
2019-10-30    5.501898
2019-10-31    5.501898
2019-11-01    5.501898
2019-11-02    5.501898
2019-11-03    5.501898
2019-11-04    5.501898
2019-11-05    5.501898
2019-11-06    5.501898
2019-11-07    5.501898
Freq: D, Name: predicted_mean, dtype: float64
```

[ ]:
```
import matplotlib.pyplot as plt

# Plot the historical data
plt.figure(figsize=(12, 6))
plt.plot(category_df['M01AB'], label="Historical Data")

# Plot the forecast
forecast_dates = pd.date_range(start=category_df.index[-1], periods=30,␣
 ↪freq='D')
plt.plot(forecast_dates, forecast, label="Forecast", color='red')

# Add labels and title
plt.xlabel("Date")
plt.ylabel("M01AB Sales Volume")
plt.title("M01AB Sales Forecast")
plt.legend()
plt.show()
```