2024 Spring Session 4: Python for SAS Programmers

Date: Wednesday, 5/15 3:30-4:15pm ET
Speaker: Colton Rathe

1. **Q- How do macros work in Python?**
   A- There are not macros in Python; however, you can achieve similar functionality. Like a user defined SAS macro, a user-defined function is defined to perform a specific task. A user-defined function is defined using the 'def' key word followed by a unique function name, a pair of parentheses that may contain parameters, and a colon. To call a function, use the function name followed by parentheses enclosing any arguments that should be passed to the parameters of function. Similarly, to calling a macro variable, you can set a variable to a value and directly embed the variable within a formatted string. See User-Defined Functions in the example for more information.

2. **Q- How to import a SAS7BDAT file into Databricks?**
   A- Users can use either the SAS7BDAT package or the PYREADSTAT package to be able to read a SAS dataset into a Pandas DataFrame. Refer to the User Defined Functions: Example 2: Import SAS dataset into a Pandas DataFrame.

3. **Q- Where should we be saving our files**
   A- You should be saving your files on S3 (SAS workbench) or in a Centralized Data Repository (CDR) database table.

4. **Q- What is the best way to store logs and results from Databricks?**
   A- There is a built in Python 'logging' module recommended for logging. The module gives programmers a flexible toolkit to manage log messages. It lets you set up loggers, handlers, and formatters to customize how your logs are recorded and formatted.

   After your logs are created, copy your logs to S3. One way to do this is using dbutils.fs.cp.

   For example, dbutils.fs.cp("file:local_log_path/logfile.log", "s3_path/logfile.log)
   For more information on logging: https://sematext.com/blog/python-logging/

   The best way to save results is to utilize Spark's DataFrame "write" method and directly write to S3 (SAS workbench).
   For example, df.write.format("csv").save(s3_path)

5. **Q- What are some additional resources you would recommend?**
   A- We recommend the following resources:

   Databricks documentation on Databricks Utilities: https://docs.databricks.com/en/dev-tools/databricks-utils.html

   Official documentation for
   PySpark: https://spark.apache.org/docs/latest/api/python/index.html

Official documentation for PySpark
Pandas: https://spark.apache.org/docs/latest/api/python/user_guide/pandas_on_spark/index.html

Official documentation for Pandas: https://pandas.pydata.org/docs/index.html

Spark - The Definitive Guide: https://analyticsdata24.wordpress.com/wp-content/uploads/2020/02/spark-the-definitive-guide40www.bigdatabugs.com_.pdf

Spark by example: https://sparkbyexamples.com/

101 PySpark exercises for data
analysis: https://www.machinelearningplus.com/pyspark/pyspark-exercises-101-pyspark-exercises-for-data-analysis

6. **Q- Can PySpark do everything Pandas can? Or is there anything we can only do in Pandas?**
   A- They have a lot of overlapping capabilities. Pandas has some built in visualizations that PySpark may not include, but for the most part when working with DataFrames, PySpark can do everything Pandas does, but much faster.

7. **Q- I assume the SAS workbench is going away when Viya goes away? If that is correct, what will be the equivalent replacement for storing data to use among colleagues (e.g., that all of us in the Division of Nursing Homes could access)?**
   A- The SAS workbench is just a mounted Simple Storage Service (S3) bucket on the SAS servers. That same S3 bucket is available to read and write with using your org/group compute cluster. You can also utilize the new Unified File Management (UFM) application and the Network Shares to be able to upload files onto that S3 bucket/workbench just like you could with FileCloud.

8. **Q- What is the advantage of using temporary storage (with Package operating system (OS)) vs. creating a temporary view in PySpark?**
   A- The temporary storage on the local filesystem is a way to be able to use Pandas or other things to write files out and then move them to the S3 workbench. A temporary view is creating a temporary call data records (CDR) database table/view to be able to utilize with Spark SQL. One is an S3 workbench unstructured storage, and the other is a SQL-specific database structured storage.

9. **Q- Will documentation on how to use UFM be available, and if so where? And how does one get access?**
   A- We do not manage the UFM tool, but we will be creating training materials around how to utilize the application to get data onto the S3 workbench and accessible with Databricks.

10. **Q- Where will that documentation live?**
    A- It will be within our SAS Viya Transition page within the Portal Support space.

11. **Q- If the S3 bucket is mounted on the SAS server, then where will the S3 bucket live once SAS is decommissioned?**
    A- S3 is a service within Amazon Web Services (AWS) and resides there. The physical files do not actually live on the SAS servers, but rather within AWS S3.

12. **Q- Everything must be in DataFrame means moving lots of data from here to there and many copies of the same data (in S3, DataFrame, etc.)? Will transferring large data from everywhere to DataFrame slow down the process?**
    A- Reading and writing from S3 can certainly take time and resources. Make sure to leverage Spark as much as possible as it leverages distributed processing capabilities, reducing time and resource consumption.

13. **Q- When reading in the SAS data from workbench in the Databricks, it shows the error "Unable to allocate 47.2 GiB for an array with shape (362, 17496514) and data type object". Is it normal?**
    A- We would recommend that you take a different approach and not try to import the SAS dataset of that size into a DataFrame. You can write the SAS dataset out to a CDR database table or to a text delimited (CSV) type of file onto your workbench, and then read that into a Spark DataFrame.

14. **Q- Do you mean I need to convert the big SAS data set to comma separated value (CSV) file format first?**
    A- Correct, or you can write it to a database table and then use Spark SQL/PySpark to read it into a DataFrame.

15. **Q- Do all three methods have efficient ways of importing SAS7BDAT format?**
    A- Users can use either the SAS7BDAT package or the PYREADSTAT package to be able to read a SAS dataset into a Pandas DataFrame. See User-Defined Functions Example 2: Import SAS dataset into a Pandas DataFrame.

16. **Q- How about text in specific columns like 1-8?**
    A- Reading data with column input in PySpark. You can define a schema and read specific columns from a CSV file.
    For example:

```
# Define schema for the specific columns you want to read
schema = StructType([
    StructField("Column1", StringType(), True),
    StructField("Column2", IntegerType(), True),
    StructField("Column3", StringType(), True),
    StructField("Column4", IntegerType(), True),
    StructField("Column5", StringType(), True),
    StructField("Column6", IntegerType(), True),
    StructField("Column7", StringType(), True),
    StructField("Column8", IntegerType(), True)
])# Read CSV file with the defined schema
df = spark.read.csv("s3_path/file.csv", schema=schema, header=True)
```

    Writing data with column input in PySpark:
    You can write the DataFrame to a CSV file, selecting only specific columns you want.
    For example:

```
# Select specific columns to write
selected_columns = df.select("Column1", "Column2", "Column3", "Column4", "Column5",
"Column6", "Column7", "Column8")# Write DataFrame to CSV
selected_columns.write.csv("s3_path/file.csv", header=True)
```

17. **Q- We usually use the workbench to save all our SAS code which allows other users on the team to run the code. Do we also save all the Databricks notebooks on the workbench as well so other team members can run or is there a different way?**
    A- We would recommend you save your Notebooks and code within the Databricks Workspace. Each contract/organization has their own specific folder within there and you can set up whatever filesystem you want to set up as everybody within your group will have access to that space.

18. **Q- If there are large SAS datasets we will need to use in the future, do we need to ensure that they are converted to csv and/or a CDR table within our project's database prior to the December 2025 decommission?**
    A- Yes. I would recommend that you write them to a CDR database table as that will be much more efficient to read in than an SAS dataset.

19. **Q- Will we have access to all of this?**
    A- Yes, you can access the notebook here: Python for SAS Programmers Notebook: https://da-cdr-prod.cloud.databricks.com/?o=8943086681571732#notebook/4197603606132855

    Then all of the materials (slide decks, notebook links, session recordings) will be available on our Confluence page here: CCSQ Data & Analytics Data Camp - Data - QualityNet Confluence (cms.gov)

20. **Q- Can a user defined function be used in a DataFrame, within a .withColumn for example?**
    A- You could extend the type of method for a DataFrame, but it is not straight forward. DataFrames have the following methods defined: https://spark.apache.org/docs/3.1.1/api/python/reference/api/pyspark.sql.DataFrame.html

21. **Q- As we are creating Spark DataFrame on top of parquet files in S3, keeping a fair assumption that if the parquet files reside in a different Application Development Organizations (ADOs) bucket, then those permission gateways need to be opened first, to be able to access the data inside the spark DataFrame inside Databricks.**
    A- Spark DataFrames are in memory created by the code that is run in the notebooks.

22. **Q- Can you export a DataFrame as a SAS dataset?**
    A- You can, but you may have issues with encoding if you are bringing it into SAS Viya.