**MIT xPRO Data Engineering Certificate**

**Parquet & Feather**

Feather and Parquet are formats that can be used to handle very large volumes of data.

**Parquet**

Reading and writing large data files using the pandas *library* can be time consuming, and it can use a large *chunk* of your computer’s memory.

To overcome these issues, the Apache Software Foundation created Parquet, an open-source columnar storage format, for use in big data analysis. The main difference between traditional storage and the type of storage offered by the Parquet format resides in the way that data is processed and stored. Parquet stores data by column, thus the term columnar storage format, rather than by row. This offers far more efficient access to data because having the data stored in columns keeps the data of the same type together, thus reducing the *querying* time.

For example, traditional databases are row-oriented databases that store data by row. In these databases, the fields for each record are stored in a long row — for example, “Customer 1: name, address, date of birth, etc.” In a columnar database, the names of every customer appear in a “name” column and all the addresses appear in an “address” column, etc.

**Reading and Writing Files Using the Parquet Format**

Before you start reading and writing files using the Parquet format, you must install theApache Arrow *library*, which is used by pandas to process Parquet files.

To install the Apache Arrow *library*, run the following command in a Terminal window:

pip install pyarrow

Suppose that you have a *dataframe* titled df that is defined by the syntax below:

import pandas as pd  
df = pd.DataFrame(data={'col1': [1, 2], 'col2': [3, 4]})

You can write the df *dataframe* in the Parquet format by using theto\_parquet()*function*. This *function* takes, as an *argument*, the name of the file where you want to save your data, with the .parquet extension, as shown in the command below:

df.to\_parquet('df.parquet')

Running the commands above in a Jupyter Notebook will create a df.parquet file in the same *directory* as your Jupyter Notebook. The df.parquet file contains your *dataframe* written in the Parquet format.

You can verify that the df.parquet file has been created by running the ls command in your Terminal window.

You can also read files from a Parquet formatted *dataframe* by using the read\_parquet()*function*. This *function* takes, as an *argument*, the name of the file that you want to read with the .parquet extension using the command below:

pd.read\_parquet('df.parquet')

By running the above command in a Jupyter Notebook, you will be able to see your original *dataframe*,as illustrated in the image below:



Now that you have a basic understanding of how the Parquet format works, you are ready to start reading and writing files with Parquet.

**Feather**

The Apache Software Foundation offers another type of open-source columnar storage format called Feather.

The main differences between the Parquet and Feather formats can be summarized as follows:

**Parquet vs. Feather Format**

| **Parquet Format** | **Feather Format** |
| --- | --- |
| The Parquet format is designed for long-term storage. | The Feather format is better suited to short-term storage. |
| The Parquet format is more expensive to write in terms of memory than the Feather format, as it features more layers of encoding and compression. | Using the Feather format enables faster input/output speeds and uses less memory. |
| The Parquet format is a standard storage format for analytics that is supported by many different systems: Spark, various AWS services, BigQuery, etc. So, if you are doing analytics, Parquet is a good option as a reference storage format for *querying* multiple systems. | In the Feather format, data is written using a binary format, which enables faster processing speeds. This reduces memory cost, but it is not ideal for long-term storage. |

**Reading and Writing Files Using the Feather Format**

Before you start reading and writing files using the Feather format, you must install the Apache Feather *library*, which is used by pandas to process Feather files.

To install the Feather *library*, run the following command in a Terminal window:

pip install feather

Suppose that you have a *dataframe* titled df that is defined by the syntax below:

import pandas as pd  
df = pd.DataFrame(data={'col1': [‘a’, ‘b’], 'col2': [‘c’, ‘d’]})

You can write the df *dataframe* in the Feather format by using theto\_feather()*function*. This *function* takes, as an *argument*, the name of the file where you want to save your data, with the .ftr extension, as shown in the command below:

df.to\_feather('df.ftr')

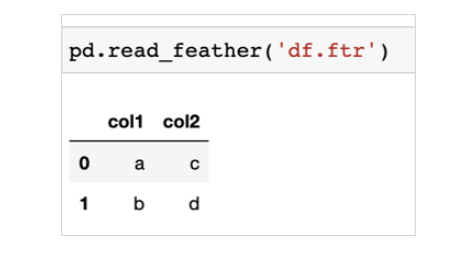
Running the above command in a Jupyter Notebook will create a df.ftr file in the same *directory* as your Jupyter Notebook. This file will have the df *dataframe* written in the Feather format.

You can verify that the df.ftr file exists by running the ls command in your Terminal window.

You can also read files from a Feather formatted *dataframe* by using the read\_feather()*function*. This *function* takes, as an *argument*, the name of the file you want to read, with the .ftr extension, as shown below:

pd.read\_feather('df.ftr')

By running the above command in a Jupyter Notebook, you should be able to see the original df *dataframe*:



Now that you have a basic understanding of how the Feather format works, you are ready to start reading and writing files using it.