Overview

This notebook will show you how to create and query a table or DataFrame that you uploaded to DBFS. DBFS (https://docs.databricks.com/user-guide/dbfs-databricks-file-system.html) 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.

```
# File location and type
file_location = "/FileStore/tables/WA_Fn_UseC__Telco_Customer_Churn.csv"
file_type = "csv"

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

# The applied options are for CSV files. For other file types, these will be
ignored.
df = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location)

display(df)
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenu
1	7590-VHVEG	Female	0	Yes	No	1
2	5575-GNVDE	Male	0	No	No	34
3	3668-QPYBK	Male	0	No	No	2
4	7795-CFOCW	Male	0	No	No	45
5	9237-HQITU	Female	0	No	No	2
6	9305-CDSKC	Female	0	No	No	8
7	1452-KIOVK	Male	0	No	Yes	22

Showing the first 1000 rows.

Churn is the column we need to predict

```
from pyspark.sql.types import DoubleType
df=df.withColumn("TotalCharges",df["TotalCharges"].cast(DoubleType()))
```

```
df.printSchema()
```

```
root
 |-- customerID: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- SeniorCitizen: integer (nullable = true)
 |-- Partner: string (nullable = true)
 |-- Dependents: string (nullable = true)
 |-- tenure: integer (nullable = true)
 |-- PhoneService: string (nullable = true)
 |-- MultipleLines: string (nullable = true)
 |-- InternetService: string (nullable = true)
 |-- OnlineSecurity: string (nullable = true)
 |-- OnlineBackup: string (nullable = true)
 |-- DeviceProtection: string (nullable = true)
 |-- TechSupport: string (nullable = true)
 |-- StreamingTV: string (nullable = true)
 |-- StreamingMovies: string (nullable = true)
 |-- Contract: string (nullable = true)
 |-- PaperlessBilling: string (nullable = true)
 |-- PaymentMethod: string (nullable = true)
 |-- MonthlyCharges: double (nullable = true)
 |-- TotalCharges: double (nullable = true)
```

```
from pyspark.sql.functions import isnan, when, count,col
display(df.select([count(when(isnan(c) | col(c).isNull(), c)).alias(c) for c in
df.columns] ))
```

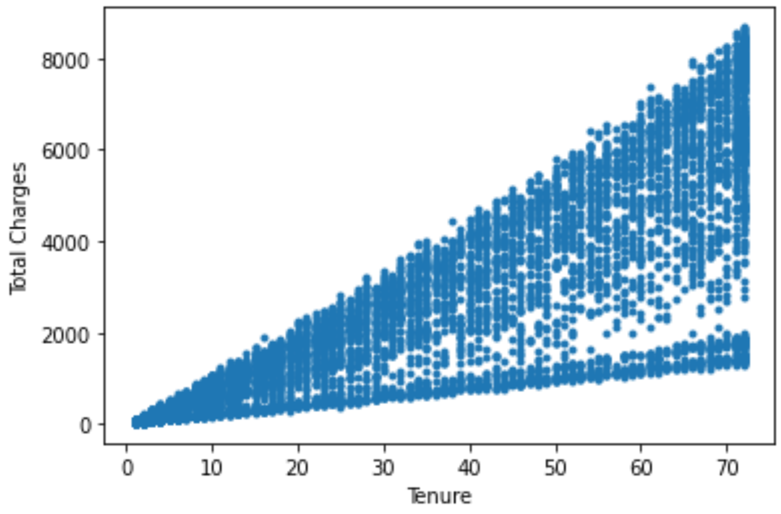
	customerID	gender	SeniorCitizen	Partner	Dependents	tenur
1	0	0	0	0	0	0

Showing all 1 rows.

Converting to Pandas df can help visualisation if the dataset is not too large

```
pd_df = df.toPandas()
```

```
import matplotlib.pyplot as plt
plt.clf()
plt.plot(pd_df["tenure"],pd_df["TotalCharges"],'.')
plt.xlabel("Tenure")
plt.ylabel("Total Charges")
display()
```



SQL based analysis

```
# Create a view or table

temp_table_name = "churn_analysis"

df.createOrReplaceTempView(temp_table_name)


%sql      /* %sql is used when you have to run sql queries */

/* Query the created temp table in a SQL cell */

select * from churn_analysis
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenu
1	7590-VHVEG	Female	0	Yes	No	1
2	5575-GNVDE	Male	0	No	No	34
3	3668-QPYBK	Male	0	No	No	2
4	7795-CFOCW	Male	0	No	No	45
5	9237-HQITU	Female	0	No	No	2
6	9305-CDSKC	Female	0	No	No	8
7	1452-KIOVK	Male	0	No	Yes	22

Showing the first 1000 rows.

```
df.groupBy("Churn").count().show()

+-----+-----+
|Churn|count|
+-----+-----+
|   No| 5174|
|  Yes| 1869|
+-----+-----+


df.select("tenure","MonthlyCharges","TotalCharges").describe().show()
##descriptive statistics of numerical columns

+-------+------------------+------------------+------------------+
|summary|            tenure|    MonthlyCharges|      TotalCharges|
+-------+------------------+------------------+------------------+
|  count|              7043|              7043|              7032|
|   mean| 32.37114865824223| 64.76169246059922|2283.3004408418697|
```