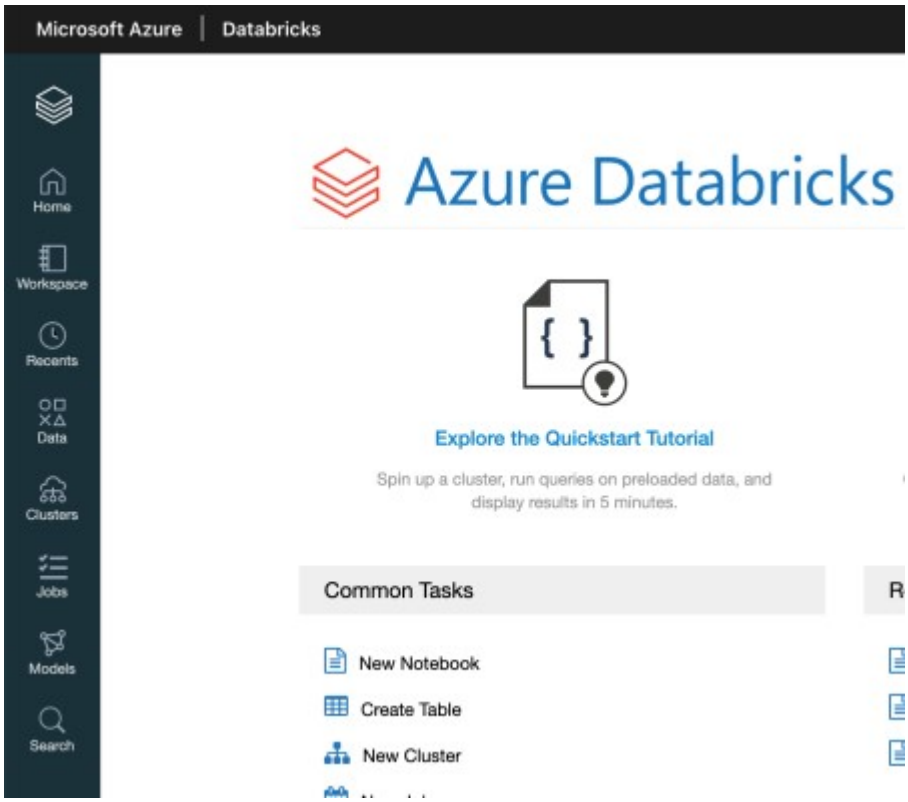
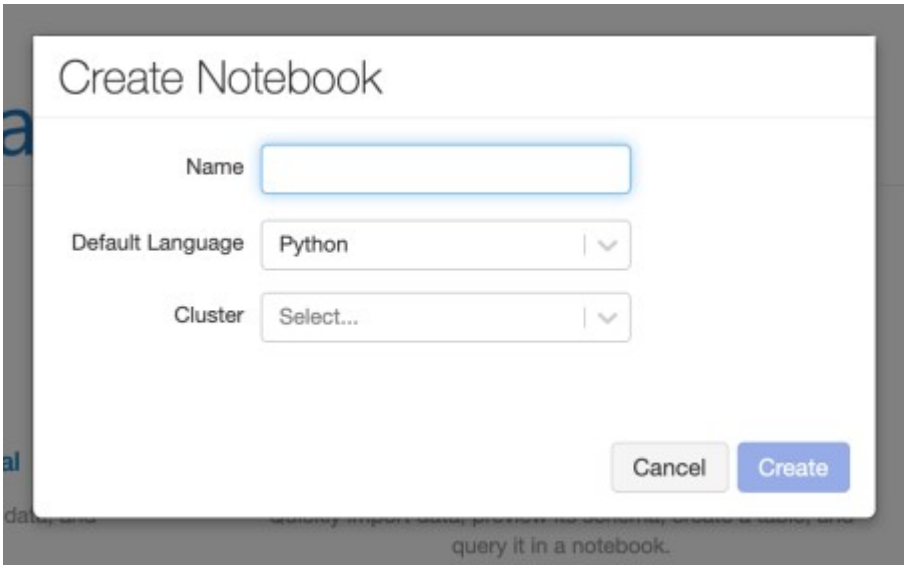
**1 . What are Notebooks?**

Notebook is a powerful text document that integrated interactive computing environment for data engineers, data scientists and machine learning engineers. It supports multiple kernels (compute environments) and multiple languages. Most common Notebook is Jupyter Notebook, name suggesting and providing acronyms for Julia, Python and R. Usually a notebook will consist of a text, rich text, HTML figures, photos, videos, and all sorts of engineering blocks of code or text. These blocks of code can be executed, since the notebooks are part of client-server web application. Azure Databricks notebooks are type of ipynb notebooks, same format as the Jupyter notebooks. Databricks environment provides client and server and you do not have to worry about installation not setup. Once the Databrick cluster is up an running, you are good to go.

On your home screen, select “New Notebook”



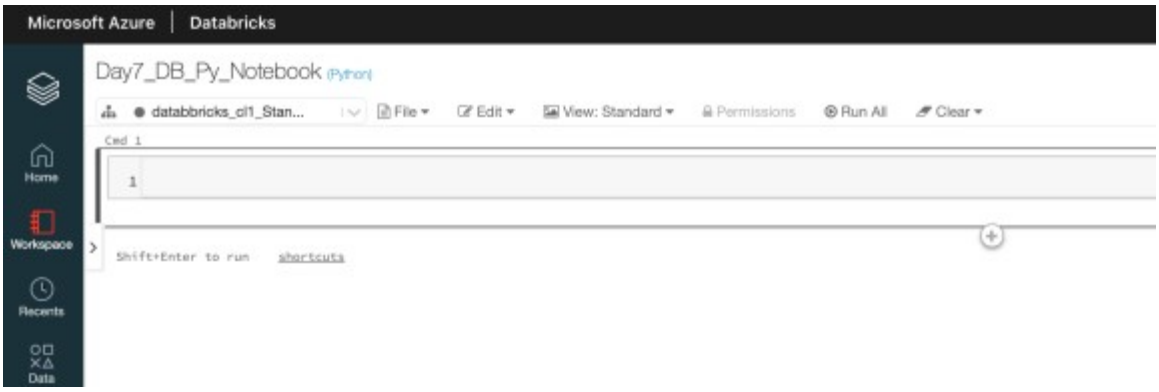

and give it a Name, Language and Cluster.




Databricks notebooks are support multi languages and you can seaminglessly switch the language in the notebook, without the need to switching the languange. If the notebooks are instructions of operations an what to do, is the cluster the engine that will execute all the instructions. Select cluster, that you have created on Day 4. I am inserting following:

| | |
|---|---|
| Name: | Day7_DB_Py_Notebook |
| Default Language: | Python |
| Cluster: | databricks_cl1_standard |

If your clusters are not started, you can still create a notebook and later attach selected cluster to notebook. This is how empty Databricks notebook looks like:



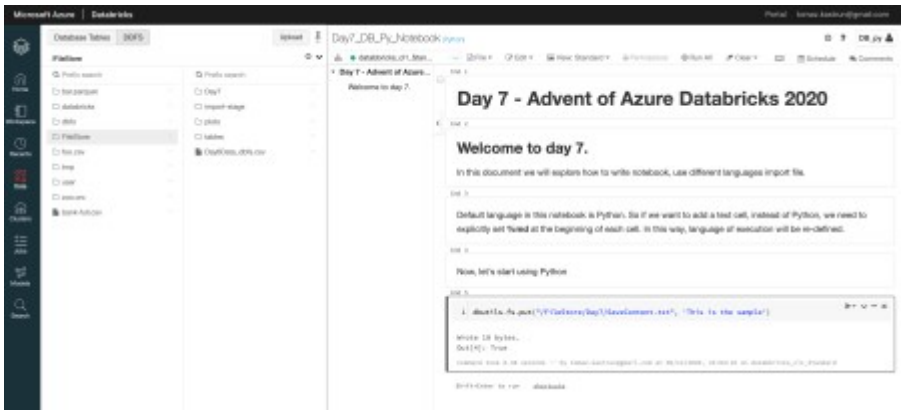

Notebook consists of cells that can be either formatted text or code. Notebooks are saved automatically. Under File, you will find useful functions to manage your notebooks, as: Move, Clone, Rename, Upload, Export. Under menu Edit, you will be able to work with cells, and code blocks. Run all is a quick function to execute all cells at one time (or if you prefer you can run a cell one by one, or selected cell all below or above). Once you start writing formatted text (Markdown, HTML, others), Databricks will automatically start building *Table of content*, giving you better overview of your content.

Let's start with Markdown and write the title and some text to notebook and adding some Python code. I have inserted:

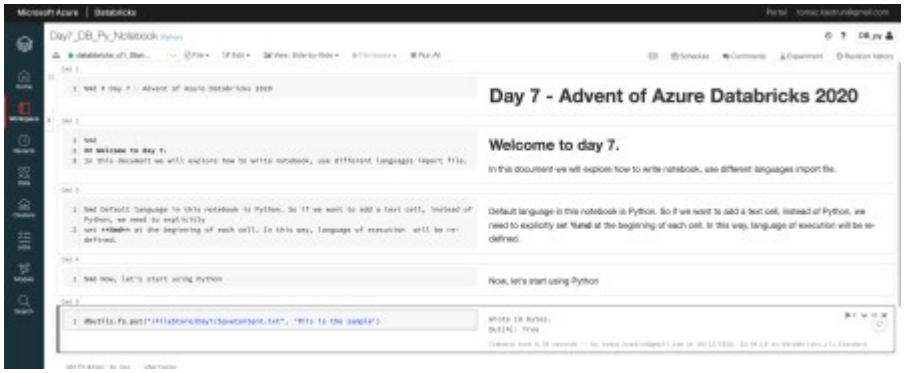
```
%md # Day 7 - Advent of Azure Databricks 2020

%md
## Welcome to day 7.
```

```
In this document we will explore how to write notebook, use different
languages import file.

%md Default language in this notebook is Python. So if we want to add a text
cell, instead of Python, we need to explicitly
set **%md** at the beginning of each cell. In this way, language of execution
will be re-defined.

%md Now, let's start using Python

dbutils.fs.put("/FileStore/Day7/SaveContent.txt", 'This is the sample')
```

And the result was a perfect Notebook with Heading, subtitle and text. In the middle the Table of content i generated automatically.




Under view, changing from Standard to side-by-side and you can see the code and converted code as notebook on the other-side. Useful for copying, changing or debugging the code.




Each cell text has **%md** at the beginning, for converting text to rich text – Markdown. The last cell is Python

```
dbutils.fs.put("/FileStore/Day7/SaveContent.txt", 'This is the sample')
```

That generated a txt file to Filestore. File can be also seen in the left pane. DbUtils – Databricks utils is a set of utility tools for efficiently working with object storage. dbUtils are available in R, Python and Scala.

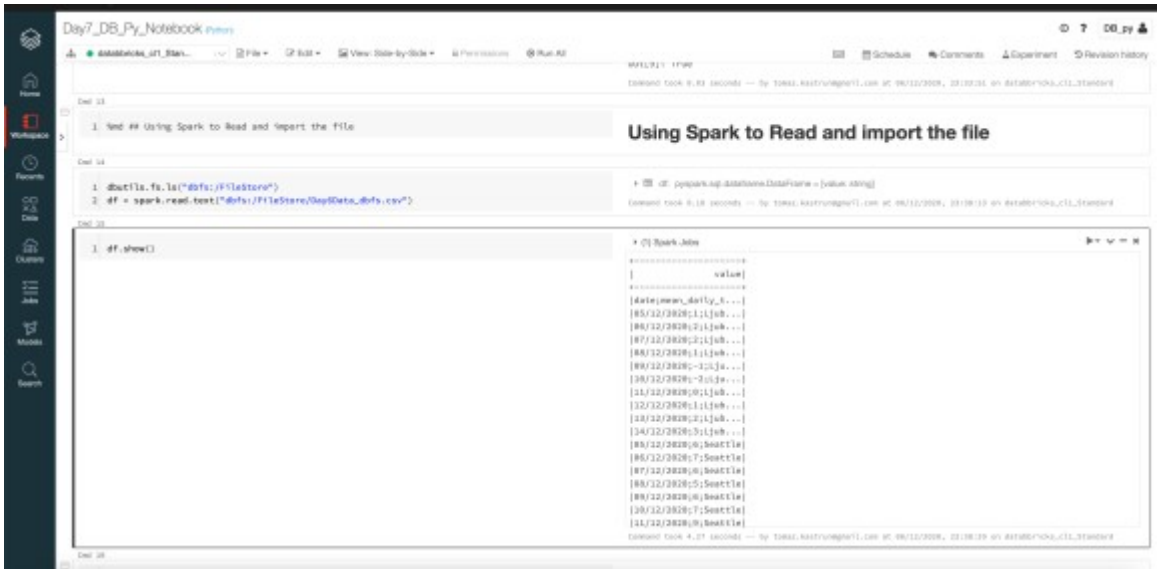
**2. Importing file**

In Notebook we have the ability to use multiple languages. Mixing Python with Bash and Spark and R is something common. But in this case, we will use DbUtils – powerful set of functions. Learn to like it, because it will be utterly helpful.

Let us explore the Bash and R to import the file into data.frame.

```
dbutils.fs.ls("dbfs:/FileStore")
df = spark.read.text("dbfs:/FileStore/Day6Data_dbfs.csv")

df.show()
```

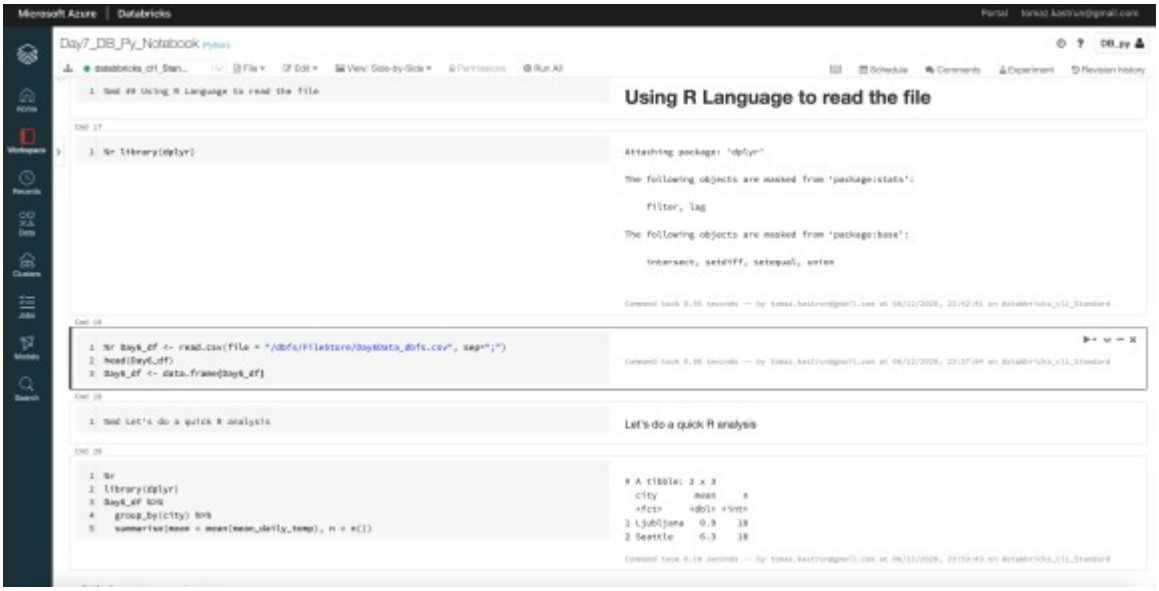
And the results is:




And do the same for R Language:

```
%r library(dplyr)

%r Day6_df <- read.csv(file = "/dbfs/FileStore/Day6Data_dbfs.csv", sep=";")
head(Day6_df)
Day6_df <- data.frame(Day6_df)

%md Let's do a quick R analysis

%r
library(dplyr)
Day6_df %>%
  group_by(city) %>%
  summarise(mean = mean(mean_daily_temp), n = n())
```

And the result is the same, just using R Language.


Microsoft Azure | Databricks
Day7_DB_Py_Notebook
Using R Language to read the file
Let's do a quick R analysis