Extraction and Ingestion, Transformation and Loading for Formula-1 data to understand and determine driver and constructor trends

This is a project that is fully executed on the cloud
Choice of service: Microsoft Azure
Resources created and used: Azure Data Lake Storage (Gen2), Azure Databricks, Azure Data Factory, Key Vault, Azure Active Directory for app registration.
Languages used: PySpark, SQL, Python

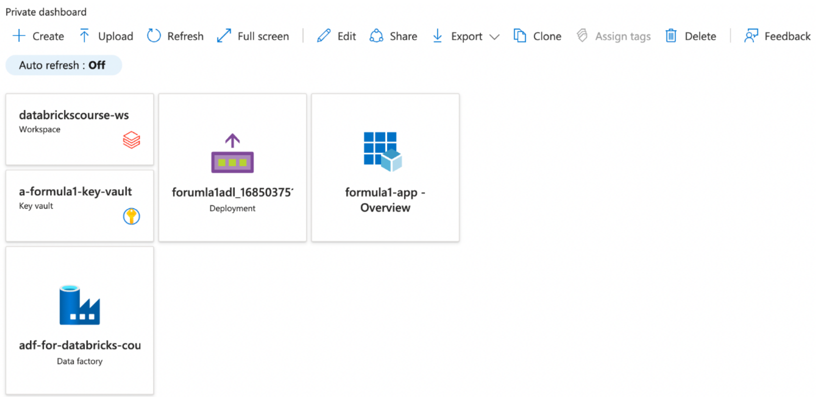
The access pattern used is Service Principal, to have a secure connection between databricks and data lake in order to avoid using credentials in the notebooks.

This also involves creating a secret scope in databricks with the details from the service principal, these details have been further stored in key-vault with variable names assigned to them and then used in the notebooks as it is confidential information (application-id, tenant-id and such..)

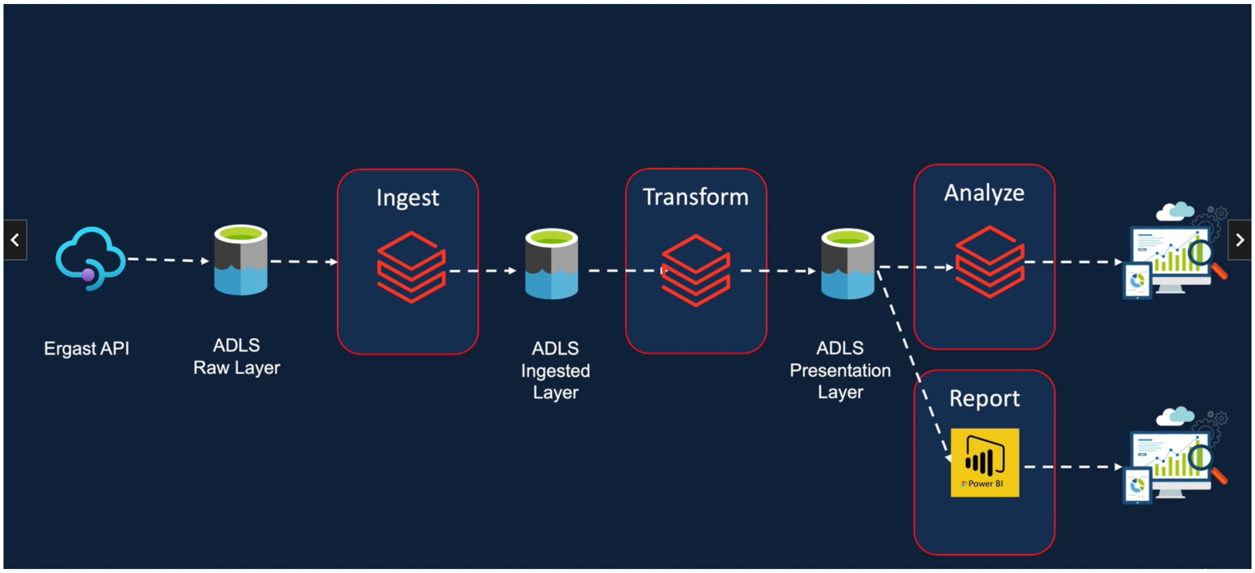
A role also has been assigned to the data lake as ‘Storage Blob Data Contributor’
Created containers to store data that has been initially used and processed for presentation with various trends

Finally, instead of using the abfss protocol for the paths, which is lengthy, exposing and redundant, used the file system utilities to mount the storage i.e., mounting ADLS to azure databricks workspace

Created a resource group for all these services and curated a dashboard for ease of use.



Solution architecture overview:



In this project , I have ingested files according to dates
Both Full Load and Incremental pattern of data was used.

The nature/pattern of the files used are:

1) Csv
2) Json
3) nested json
4) Multi-line json
5) Split csv
6) Multi-line split json

The full load files :
→ circuits
→ constructors
→ drivers
→ races

The Incremental load files :
→ results
→ pit-stops
→ lap-times
→ qualifying

These files were initially stored in the raw layer in delta tables with no changes/ transformation performed with delta tables created for each file.

After performing a few changes like selecting only the required columns, changing column names, I used intermediate dataframes and finally stored these in delta tables in the processed layer.

After performing further aggregation functions, joins and refining the data to get various trends like driver standings, constructor standings over the years this sport has emerged (1950 to present), stored them in delta tables in the presentation layer.

To automate this process an easy solution was to use databricks widgets for date entry for various dates in which files were named (race dates), and perform the same ingestion, transformation and load. This has its own demerits as each time we need to input the right date else we wouldn’t get the required output

But a much more elegant approach is to implement a pipeline in data factory with few parameters , linked services and a tumbling window for the date which will automatically search for the file in the mentioned path