

# Data extraction & processing while maintaining security, monitoring & standard on azure platform.

**Problem statements:** As data engineer usually will get different requirement for data processing & transformation of various sources, Azure offers a wide range of services and tools to help you achieve this.

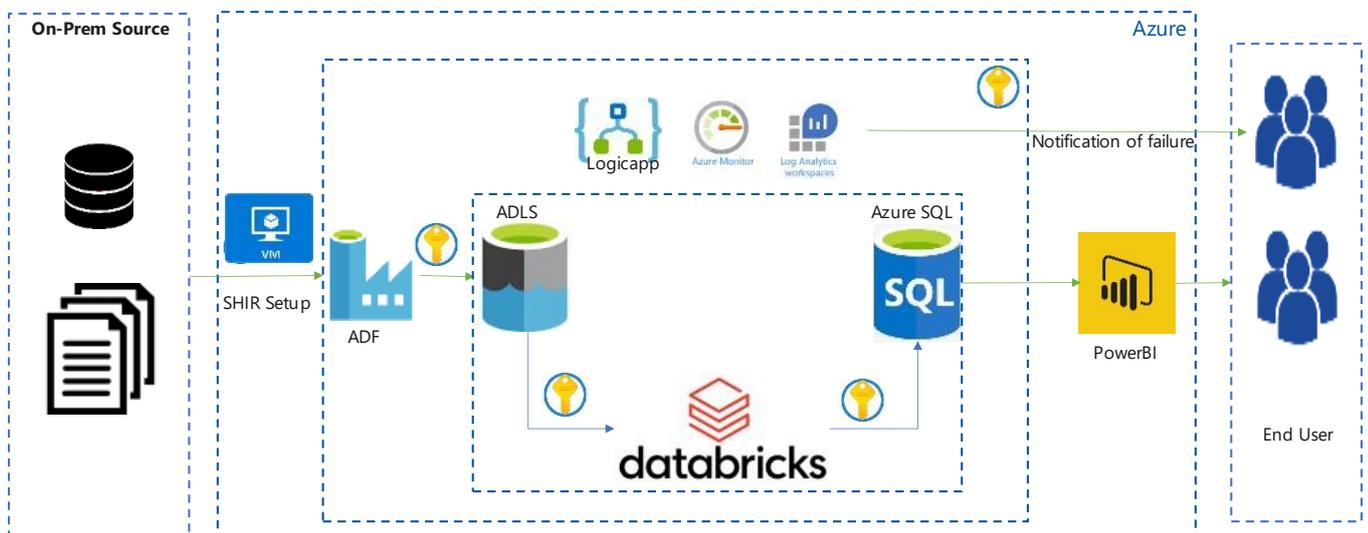
We received requirement to pick up files from on premises to azure cloud platform & perform checks on data quality while maintaining security, monitoring & data lake transformation layers before loading data in Data warehouse.

## Services use:

1. **Azure account:** For using cloud platform
2. **Azure Data lake :** - For storing data & maintaining transformation layer
3. **Azure DataFactory:-** For data ingestion from onprem to azure & data processing
4. **Azure Databricks:** For data transformation & delta tables
5. **Azure Synapse:** For DWH & data modelling purpose
6. **Azure Keyvault:** For maintaining credentials & secrets at centralize location.
7. **Azure monitor :** For monitor Azure resources
8. **Azure log analytics:** For querying logs
9. **Logic Apps:** For sending email on failure & success
10. **Azure VM :** For setup SHIR

## Architecture Diagram:

### Logical architecture



## Technical Requirement:

Setup azure environment with listed services

### 1. **Azure Data lake setup & maintain 3 different layer in ADLS**

- RAW: Store raw data from source
- Refined: store cleansed data after processing
- Processed : store transformed data

+ Container    🔒 Change access level    ↶ Restore containers

---

Search containers by prefix

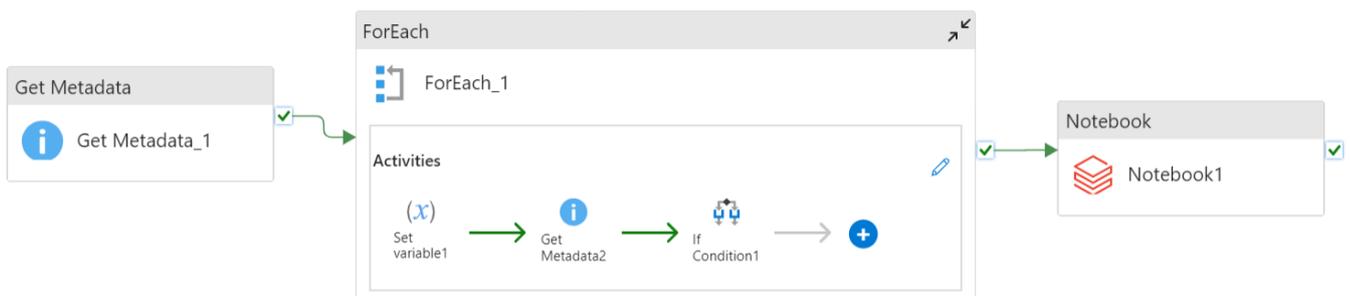
---

| Name                               |
|------------------------------------|
| <input type="checkbox"/> \$logs    |
| <input type="checkbox"/> processed |
| <input type="checkbox"/> raw       |
| <input type="checkbox"/> refined   |

### 2. **Setup vm for selfhosted IR** to connect on prem server to pickup file

| Name                       | Type        | Sub-type | Status  |
|----------------------------|-------------|----------|---------|
| AutoResolveIntegrationR... | Azure       | Public   | Running |
| integrationRuntime1        | Self-Hosted | ---      | Running |

### 3. **Setup Azure data factory** for data ingestion & processing :



Below checks has been performed in ADF for data consistency:

- Check if the file is available in the path. If it's not available, there should be timeout after 1 minute:

General Settings User properties

Name \*  [Learn more](#)

Description

Activity state (preview)  Active  Inactive

Timeout

Retry

- Check if the file size is greater than 20b or not. If the file size is greater than 20b then needs to process:

We have used 2 get metadata to check first list of files & then stored file name in variable to check size of each file



General Settings User properties

Dataset \*  [Open](#) [New](#) [Learn more](#)

Field list \* [New](#) [Delete](#)

Argument

Get metadata output passed to foreach loop activity to run loop for every file

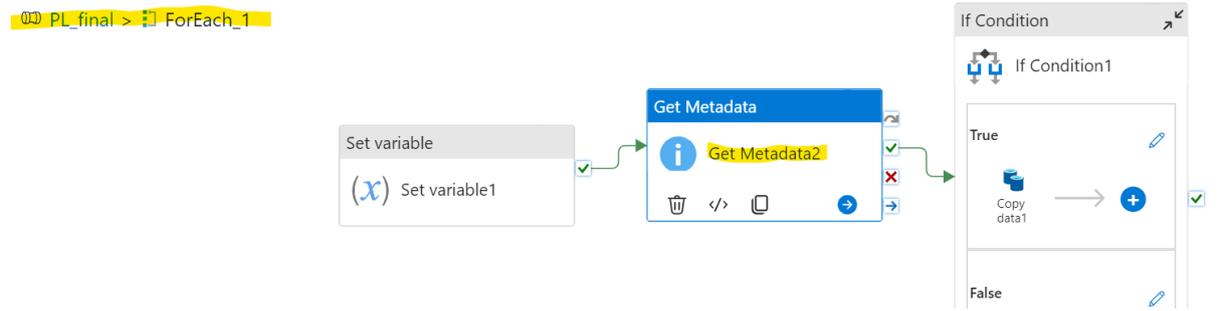
General **Settings** Activities (3) User properties

Sequential

Items

@activity('Get Metadata\_1').output.c... 

Inside for each loop used variable to store file name & with help of 2<sup>nd</sup> getmetadata pick size of file.



General **Settings** User properties

Dataset \*

DS\_dynamicFile\_blob   

> Dataset properties 

Field list \*

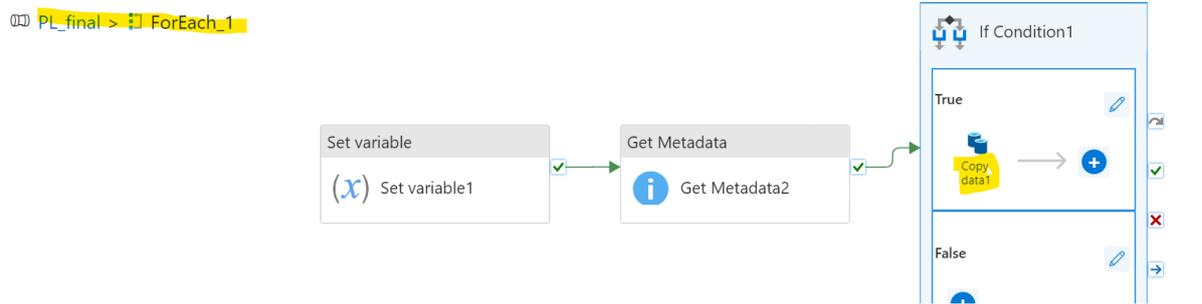
 

Argument

Size

Passed size condition in ifcondition activity to pick files which are greater then 20b

```
@greaterOrEquals(activity('Get Metadata2').output.size,21)
```



General **Activities (1)** User properties

Expression 

@greaterOrEquals(activity('Get Meta... 

Case

Activity

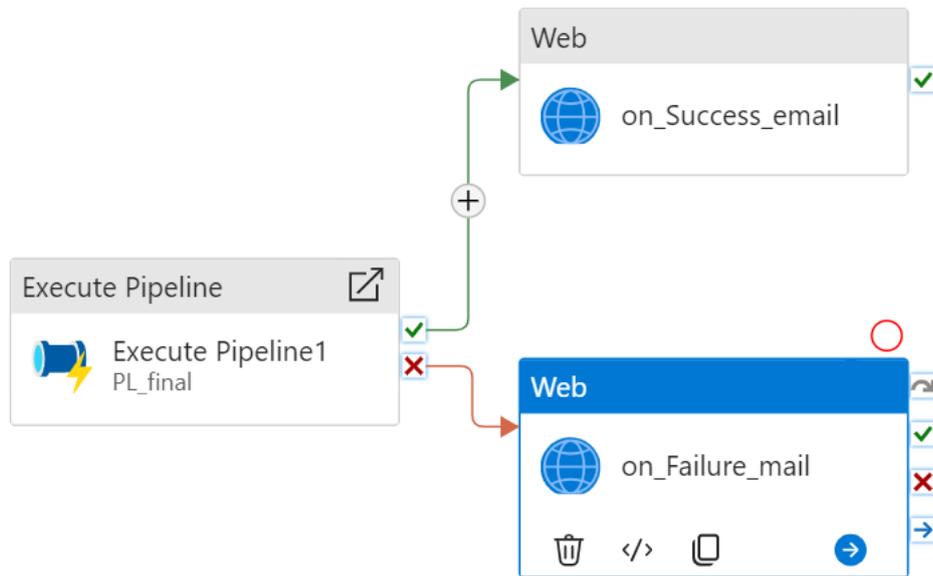
True

 Copy data1

1 Activity 

- Setup email configuration on failure & success of pipeline to get notify

Create master pipeline to call logic app to get notified on pipeline failure



4. **Configured Logicapp to trigger email** & passed logicapp url in be activity:

```
{  
  "message": "Pipeline with run ID @pipeline().RunId is  
  successfully executed.",  
  "dataFactoryName": "@pipeline().DataFactory",  
  "pipelineName": "@pipeline().Pipeline",  
  "receiver": "@pipeline().parameters.receiver"  
}
```

5. **Setup keyvault to store credentials**

| Name      | Type   | Status  |
|-----------|--------|---------|
| adobroken | Secret | Enabled |

**Edit linked service**  
 Azure Data Lake Storage Gen2 [Learn more](#)

**Authentication type**  
 Account key

**Account selection method**  
 From Azure subscription  Enter manually

**URL \***  
 https://[redacted].core.windows.net/

**Storage account key** **Azure Key Vault**

**AKV linked service \***  
 LS\_kv\_ [redacted]

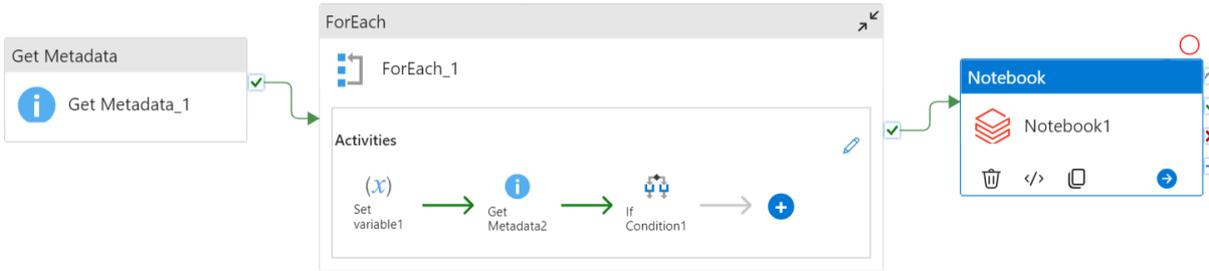
**Secret name \***  
 [redacted]-datalake

Edit

**Secret version**  
 [redacted]

Edit

Once all checks has performed databricks notebook will be executed from datafactory pipeline to maintain transformation layer in adls



General **Azure Databricks** Settings User properties

Databricks linked service \* LS\_capstone\_adb [Test connection](#) [Edit](#) [New](#)

**Setup Azure databricks :**

in Databricks couple of transformation applied , I am attaching some of them only as per confidentiality,

- If date is NULL or blank, give default date as '2020-11-28'. Format of date column should be YYYY-MM-DD.
- Remove the entries which has URL field value as 'ERROR'.
- Transform the values of column country with their acronyms. For eg: Austria would be replaced by 'AUST', Belgium by 'BELG' etc. Y

```

1 #read the parquet file from adls
2 source_file="abfss://refined@chaitanyacapastonelake.dfs.core.windows.net"
3 df_source=spark.read\
4     .format('parquet')\
5     .option('inferschema',True)\
6     .load(source_file)
7
8 adf=df_source

```

Cmd 3

```

1 # Apply transformations
2
3 target_df = adf.withColumn("date", when(col("date").isNull(), "2020-11-28").otherwise(col("date")))
4 target_df = target_df.filter(col("url") != "ERROR")
5

```

```

24 #run the loop
25 for rows in table1.find_all('tr'):
26     column=rows.find_all('td')
27     if len(column)>=2:
28         country_name=column[0].text.strip()
29         acronym=column[1].text.strip()
30         country_acronym[country_name]=acronym
31
32 #covert the column 'country' of transformed table to upper-case
33 target_adf=target_df.withColumn("country",upper(col("country")))
34
35 # Create a DataFrame from the acronym data
36 acronym_df = spark.createDataFrame(country_acronym.items(), ["country", "acronym"])
37
38 # Join the original DataFrame with the acronym DataFrame to replace values
39 adf1 = target_adf.join(acronym_df, "country", "left").select("*")
40
41 # Interchange the positions of two columns (e.g., swap "Age" and "Country") and then drop country table
42 result_df = adf1.withColumnRenamed("country", "temp").withColumnRenamed("acronym", "country").withColumnRenamed("temp", "acronym").drop("acronym")
43

```

Complete code & ARM template added in github.

Once data transformation done loading data in sql to build pbi dashboard on so can be consumed be end user.

## Challenges in implementing solution:

integrating all azure services together in secure way with help of vnet & keyvault so data can be processed from on prem to azure & load successfully in database to consume.

In while process criticality of data is key things to protect so overall flow will work smoothly as per architecture diagram.

## Business Benefits:

In all project as data engineering will play a key role to perfoem various data operation & maintain security so its help of this high level flow can implement security best practices can be followed.

**Github link:**

<https://github.com/deeksharm/azuredataengineer>