Azure Data Engineer Project- Adventure Works (GitHub files)

**Architecture diagram: -**

A diagram of a process

AI-generated content may be incorrect.

**Phase 1: - Dynamic ADF Pipeline**

**Goal:** Dynamically load CSV files from GitHub into the Bronze container in ADLS Gen2 using Azure Data Factory (ADF).

**Steps: -**

* In GitHub we have 10 csv files we need to dynamically copy all the CSV files and load into bronze container in storage account using Azure data factory.
* Create one JSON file which is required to all the file name along with URL which is required to dynamically fetch the data. (Please find below JSON code)
* Create new container name is “Parameter” and save below JSON file into this container.

**JSON file details: -**

[

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Product\_Categories.csv",

"p\_sink\_folder" : "AdventureWorks\_Product\_Categories",

"p\_sink\_file" : "AdventureWorks\_Product\_Categories.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Calendar.csv",

"p\_sink\_folder" : "AdventureWorks\_Calendar",

"p\_sink\_file" : "AdventureWorks\_Calendar.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Customers.csv",

"p\_sink\_folder" : "AdventureWorks\_Customers",

"p\_sink\_file" : "AdventureWorks\_Customers.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Product\_Subcategories.csv",

"p\_sink\_folder" : "AdventureWorks\_Product\_Subcategories",

"p\_sink\_file" : "AdventureWorks\_Product\_Subcategories.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Products.csv",

"p\_sink\_folder" : "AdventureWorks\_Products",

"p\_sink\_file" : "AdventureWorks\_Products.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Returns.csv",

"p\_sink\_folder" : "AdventureWorks\_Returns",

"p\_sink\_file" : "AdventureWorks\_Returns.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Sales\_2015.csv",

"p\_sink\_folder" : "AdventureWorks\_Sales\_2015",

"p\_sink\_file" : "AdventureWorks\_Sales\_2015.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Sales\_2016.csv",

"p\_sink\_folder" : "AdventureWorks\_Sales\_2016",

"p\_sink\_file" : "AdventureWorks\_Sales\_2016.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Sales\_2017.csv",

"p\_sink\_folder" : "AdventureWorks\_Sales\_2017",

"p\_sink\_file" : "AdventureWorks\_Sales\_2017.csv"

},

{

"p\_rel\_url": "tanajisulagave/AdventureWorks-Data-Engineering-Project/refs/heads/main/Data/AdventureWorks\_Territories.csv",

"p\_sink\_folder" : "AdventureWorks\_Territories",

"p\_sink\_file" : "AdventureWorks\_Territories.csv"

}

]

**Diagram: -**

A diagram of a diagram of a company

AI-generated content may be incorrect.

**Azure Services: -**

1. Storage Account (3 containers Bronze, Silver, Gold)
2. Azure Data factory

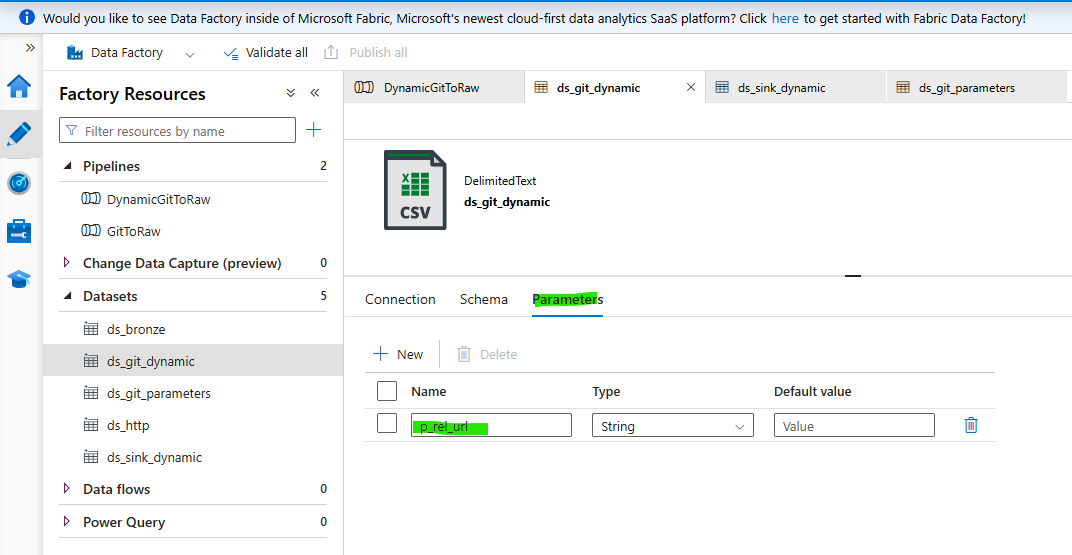
**Steps: -**

1. Create source and sink linked services with help connection of HTTP and ADLS gen 2.
2. Dataset Configuration
   1. Dataset with JSON connection (For lookup Activity): -

A screenshot of a computer

AI-generated content may be incorrect.

* 1. Source Dataset for copy activity with parameter



A screenshot of a computer

AI-generated content may be incorrect.

* 1. Sink Dataset for copy activity with parameter

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

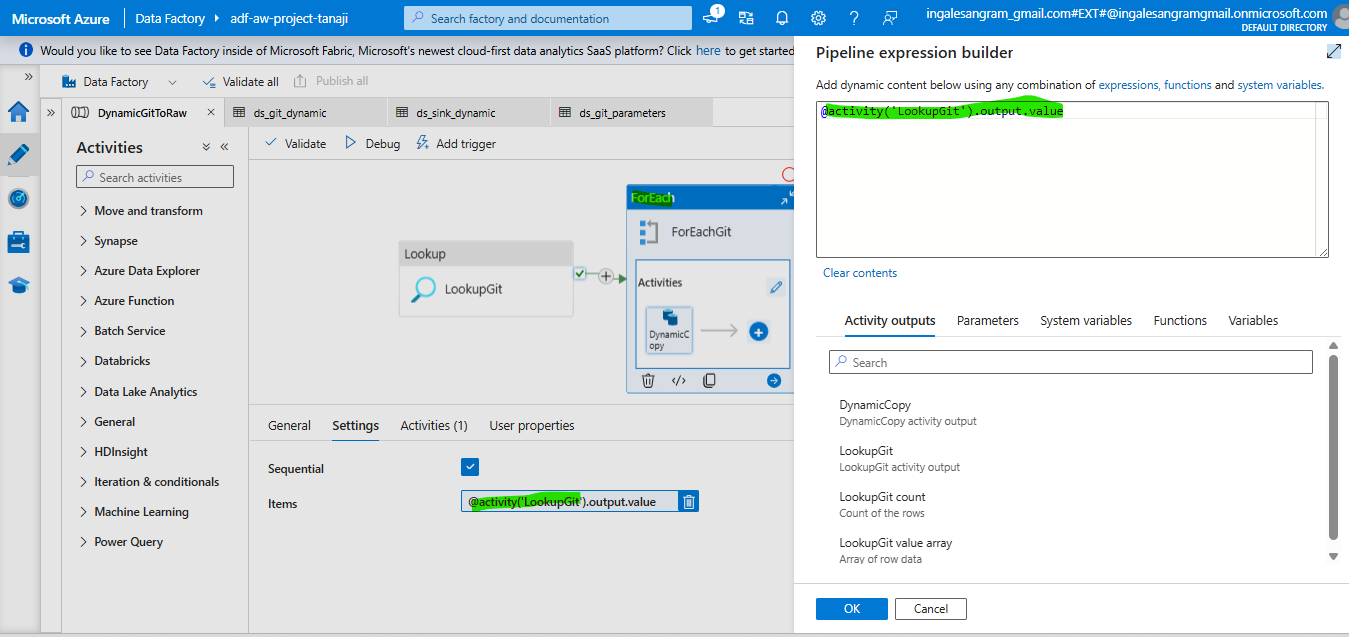
AI-generated content may be incorrect.

1. Pipeline Activities
   1. Lookup Activity

A screenshot of a computer

AI-generated content may be incorrect.

* 1. Foreach Activity



* 1. Copy Activity
     1. Source

A screenshot of a computer

AI-generated content may be incorrect.

* + 1. Sink

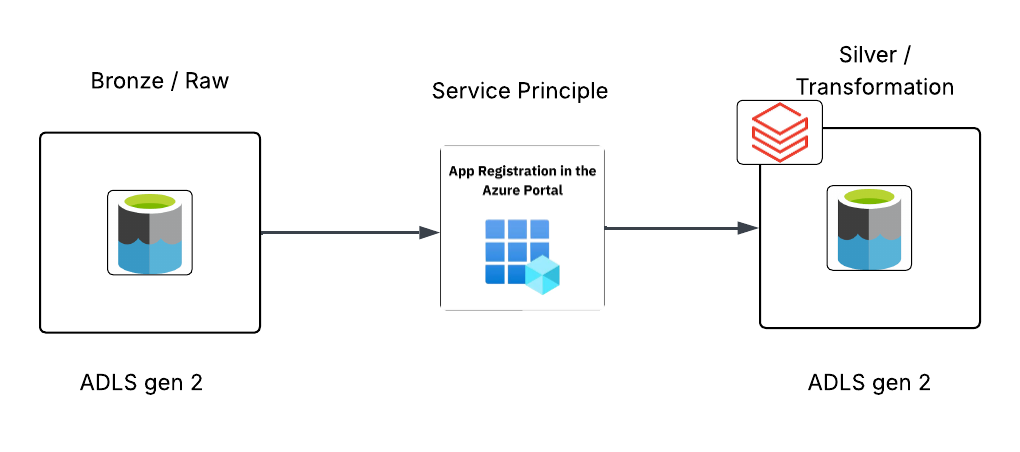
A screenshot of a computer

AI-generated content may be incorrect.

**Phase 2: - Data Transformation**

**Goal: -** We need to connect data which is available in the bronze container in ADLS gen 2 in databricks and we need to do some transformations on this data and load into silver container in ADLS gen 2 using pyspark notebook

**Diagram:-**

****

**Azure Service: -** Azure Data Bricks.

**Allow ADLS access to Azure data bricks: -**

1. Create service principle using Microsoft entra id

A screenshot of a computer

AI-generated content may be incorrect.

1. Creating secret key

A screenshot of a computer

AI-generated content may be incorrect.

1. Note down
   * 1. Storage Account Name
     2. Application (client) ID
     3. Secret value of certificate (one time visible)
     4. Directory (tenant) ID
2. Storage account access: -

Go to Storage account (ADLS gen 2) and open Access Control (IAM) provide access to this created service principle (App) in this ADLS Gen 2 as Storage Blob Data Container

A screenshot of a computer

AI-generated content may be incorrect.

1. Azure Data bricks flow

Create compute / cluster and assign this in new notebook and use below code for allow access to ADLS into notebook

spark.conf.set("fs.azure.account.auth.type.<StorageAccount>.dfs.core.windows.net", "OAuth")

spark.conf.set("fs.azure.account.oauth.provider.type. <StorageAccount>.dfs.core.windows.net", "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider")

spark.conf.set("fs.azure.account.oauth2.client.id. <StorageAccount>.dfs.core.windows.net","<Application (client) ID>")

spark.conf.set("fs.azure.account.oauth2.client.secret. <StorageAccount>.dfs.core.windows.net", "<secret Key>")

spark.conf.set("fs.azure.account.oauth2.client.endpoint. <StorageAccount>.dfs.core.windows.net", "https://login.microsoftonline.com/<Directory (tenant) ID>/oauth2/token")

1. Reading the data from the bronze layer

df\_cal=spark.read.format('csv')\

.option("header",True)\

.option("inferSchema",True)\

.load('abfss://bronze@<StorageAccount>..dfs.core.windows.net/AdventureWorks\_Calendar')

The above script is a sample script. You can read all the folders of csv files and save them into data frame using above script from the bronze layer.

**Transformations and Write into Silver containers: -**

1. Transform Calendar Table Data and load into silver container

We can add two columns on this calendar table

df\_cal=df\_cal.withColumn('month',month('Date'))\

.withColumn('year',year('Date'))

df\_cal.display()

df\_cal.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Calendar")\

.save()

1. Transform Customer Table Data

Creating "FullName" column using **contact()** and **concat\_ws()** functions

df\_cus.withColumn("fullName",concat(col('Prefix'),lit(' '),col('FirstName'),lit(' '),col('LastName'))).display()

**OR**

df\_cus=df\_cus.withColumn("FullName",concat\_ws(' ',col('prefix'),col('Firstname'),col('LastName')))

df\_cus.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Customers")\

.save()

1. Transform Product Sub-Category data

We can read and load it into a silver container, there is no transformation

df\_subcat.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_SubCategories")\

.save()

1. Transform Product data
   1. We can use **SPLIT()** function on ProductSKU and ProductName column using Index.
   2. We can Transform the existing column instead of creating new column

df\_pro = df\_pro.withColumn('ProductSKU',split('ProductSKU','-')[0])\

.withColumn('ProductName',split('ProductName',' ')[0])

df\_pro.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Products")\

.save()

1. Transform Returns Data

df\_ret.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Returns")\

.save()

1. Transform Territories Data

df\_ret.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Returns")\

.save()

1. Transform Sales Data

df\_Sales = df\_Sales.withColumn('StockDate',to\_timestamp('StockDate'))

df\_Sales = df\_Sales.withColumn('OrderNumber', regexp\_replace('OrderNumber', 'S', 'T'))

df\_Sales = df\_Sales.withColumn('multiply',col('OrderLineItem') \* col('OrderQuantity'))

df\_Sales.write.format('parquet')\

.mode('append')\

.option("path","abfss://silver@awstoragedatalaketanaji.dfs.core.windows.net/AdventureWorks\_Sales")\

.save()

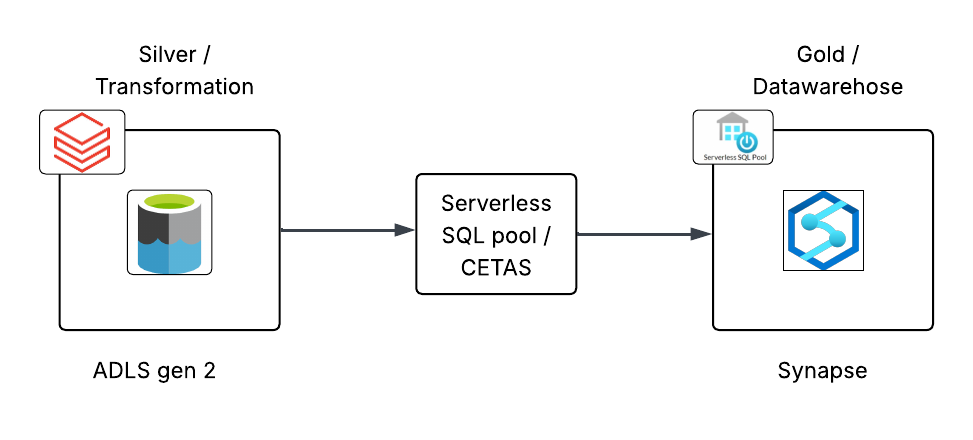
**Sales Analysis**

df\_Sales.groupBy('OrderDate').agg(count('OrderNumber').alias('total\_order')).display()

**Phase 3: - Datawarehouse**

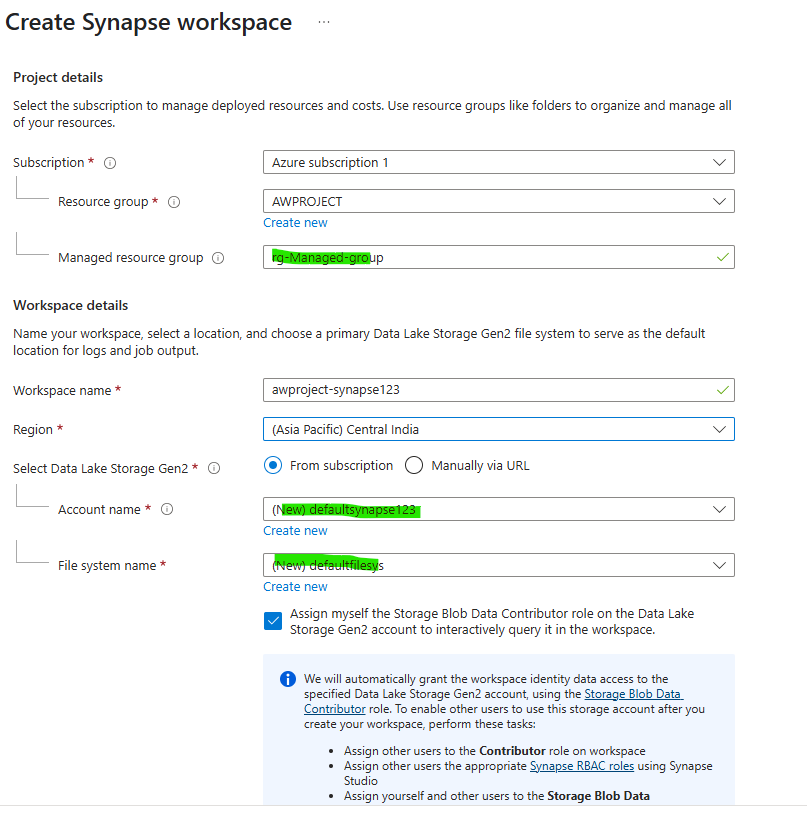
**Goal: -** We can fetch the data from silver containers and load into server less SQL pool for connecting the data to Power BI for visualization.

**Diagram:-**

****

**Azure Service: - Azure Synapse Analytics**

1. Create Service Azure Synapse Analytics



A screenshot of a computer

AI-generated content may be incorrect.

1. Provide access to storage account for Azure Synapse Analytics

**Flow: -** Go to Storage Account 🡪 Access Control (IAM) 🡪 Add role assignment 🡪 select role as a “Storage Blob Data Contributor” 🡪 members = Managed identity 🡪 select members 🡪 select Synapse Workspace (1) 🡪 select synapse managed identity 🡪 select 🡪Review + Assign and wait for 15 minutes to enable the access.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create server less SQL pool

**Flow**: - Create database for accessing the data. Go to data Tab and select plus symbol 🡪 SQL Database 🡪provide database name 🡪Create serverless SQL pool

**Lakehouse concept: -** data is stored in storage location and we can access it through SQL query using database.

So, data is in storage location, but I want to be querying the data so it will create an abstraction layer it’s kind of metadata layer to fetch the data.

So, I am querying about the metadata and serverless SQL pool apply this metadata to actual data. Serverless SQL pool do all the work behind the scenes.\

So why is this required because of the cost. storing data in data lake is very chip cost as compare traditional SQL server.

1. Develop the SQL script for accessing the data from datalake using **OPENROWSET** function.
2. Assign the role into Storage Account of specific user to query execution

A screenshot of a computer

AI-generated content may be incorrect.

1. Read the silver layer data into synapse workspace using SQL query

File Reading sample script

SELECT

    \*

FROM OPENROWSET(

    BULK 'https://awstoragedatalaketanaji.dfs.core.windows.net/silver/AdventureWorks\_Calendar/',

    FORMAT='PARQUET'

) as query1

1. BULK is location of file = you can go to storage Account container file 🡪 Three dots 🡪 then properties 🡪 copy URL
2. So here we do not need main URL, we need only till folder name and pass format of the file, so our format is “Parquet”
3. Please find the above sample script.
4. Note down in URL by default storage account create “blob” keyword in URL we need to change it as “dfs”

**Old URL**

'https://awstoragedatalaketanaji.blob.core.windows.net/silver/AdventureWorks\_Calendar/

**Changed URL**

'https://awstoragedatalaketanaji.dfs.core.windows.net/silver/AdventureWorks\_Calendar/

A screenshot of a computer

AI-generated content may be incorrect.

1. SQL commands for data reading from Silver Layer
2. Create Schema: -

-----------------------

-- create schema in databse

-----------------------

CREATE SCHEMA gold;

1. Create Views (Below is the sample view creation statement)

You can refer to uploaded script file for view creation

-----------------------

-- CREATE VIEW CALENDER

-----------------------

CREATE VIEW gold.calender

AS

SELECT

    \*

FROM OPENROWSET(

    BULK 'https://awstoragedatalaketanaji.dfs.core.windows.net/silver/AdventureWorks\_Calendar/',

    FORMAT='PARQUET'

) as query1

1. Creating External Tables (External tables keep the data on their side)

Before creating an External table, we need to set 1) credentials, 2) External Data Source, 3) External File format

1. Credential

--------------------

-- Create master Key

--------------------

CREATE MASTER KEY ENCRYPTION BY PASSWORD ='Abcd1234\*\*';

------------------

-- Database scope credentials

------------------

CREATE DATABASE SCOPED CREDENTIAL cred\_tanaji

WITH

    IDENTITY='Managed Identity' ;

1. External data source

------------------

-- CREATE External Data Source

------------------

---- For silver Layer

CREATE EXTERNAL DATA SOURCE source\_silver

WITH

(

    LOCATION='https://awstoragedatalaketanaji.dfs.core.windows.net/silver',

    CREDENTIAL = cred\_tanaji

)

---- For Gold Layer

CREATE EXTERNAL DATA SOURCE source\_gold

WITH

(

    LOCATION='https://awstoragedatalaketanaji.dfs.core.windows.net/gold',

    CREDENTIAL = cred\_tanaji

)

1. External file format

------------------

-- CREATE External File Format

------------------

CREATE EXTERNAL FILE FORMAT format\_parquet

WITH

(

    FORMAT\_TYPE = PARQUET,

    DATA\_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'

)

1. Create External Table (CETAS)

CETAS= create external tables as Select

Below is the sample script, you can refer to uploaded script file for CETAS creation

------------------

-- CREATE EXTERNAL TABLE

------------------

CREATE EXTERNAL TABLE gold.extsales

WITH

(

    LOCATION='extsales', -- ADLS Location

    DATA\_SOURCE = source\_gold,

    FILE\_FORMAT= format\_parquet

) AS

SELECT \* from gold.Sales  --- view execute (means we call view definition code)

-------- Execute CETAS

select \* from gold.extsales

A screenshot of a computer

AI-generated content may be incorrect.

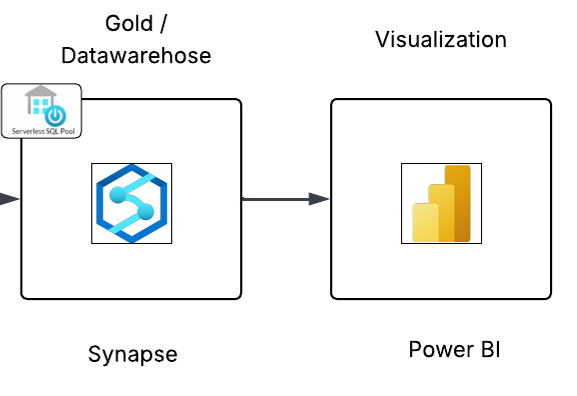
**A screenshot of a computer

AI-generated content may be incorrect.**

**Phase 4: - Power BI (Synapse workspace connection)**

* You can copy serverless SQL endpoint connection from Azure synapse workspace and add into server textbox.
* It needs connection you can pass database connection when we have created synapse analytics, we have provided user and password you can pass here otherwise you can access though Microsoft account.
* Once you provide a connection you can check the external tables and load the data into Power BI for generating the power Bi report.

**Diagram:-**



**Power Bi Report:-**

A screenshot of a computer

AI-generated content may be incorrect.