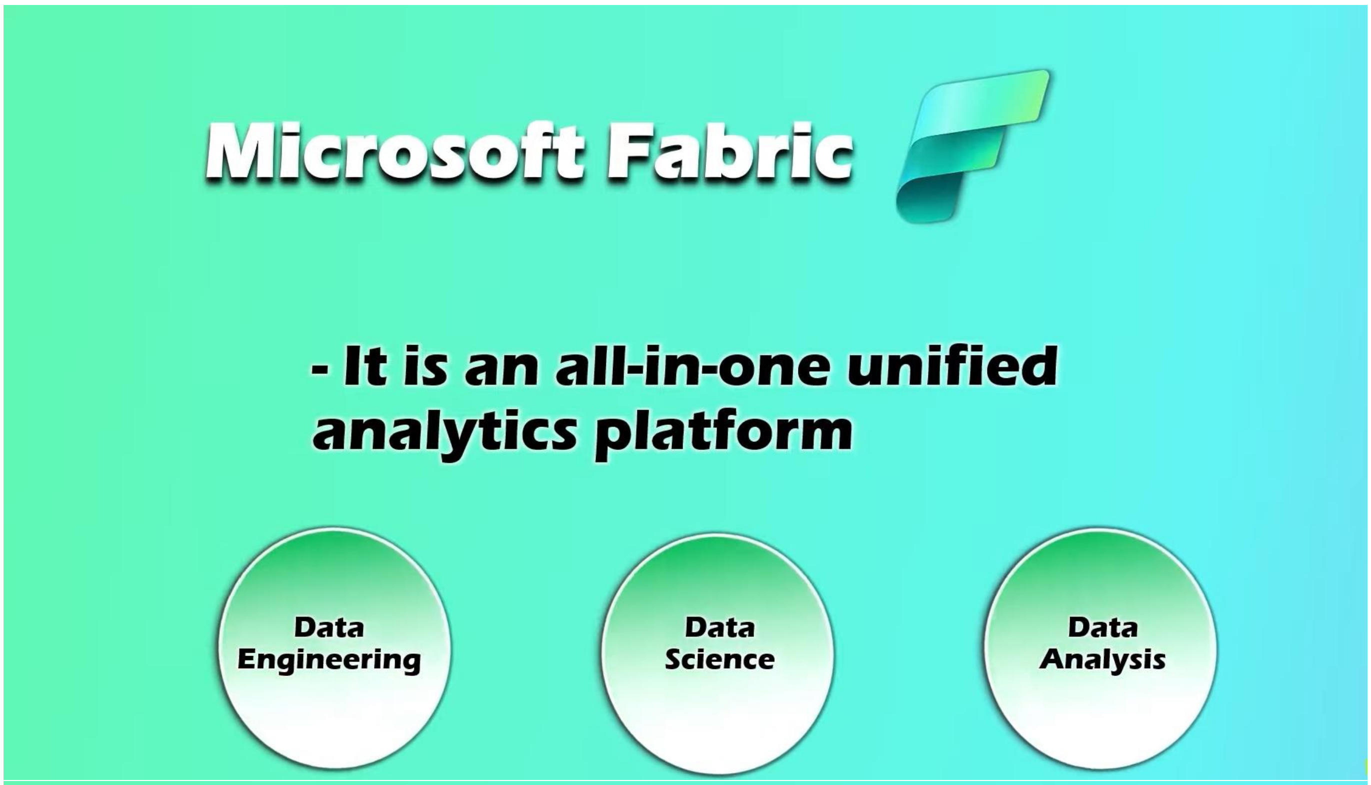
Azure Data Engineer Microsoft Fabric Project



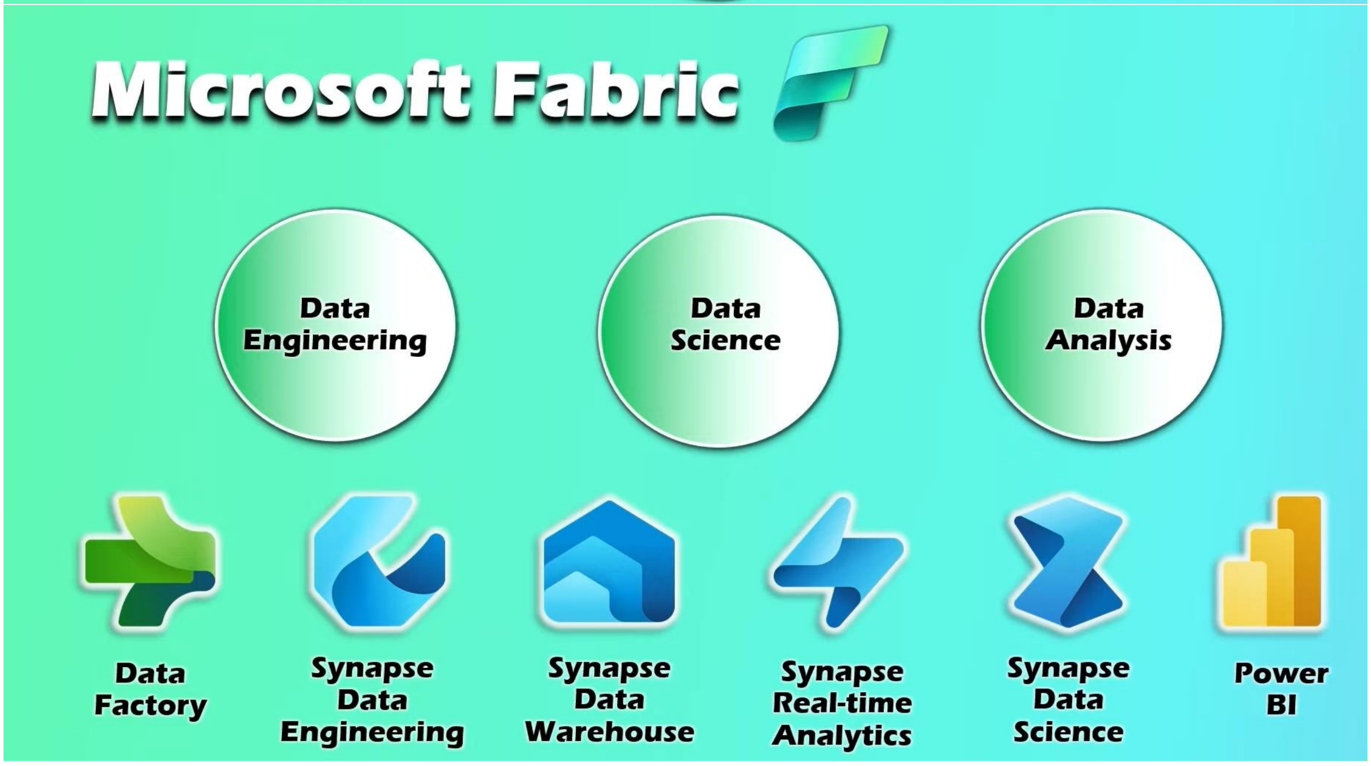
Microsoft Fabric

- It is an all-in-one unified analytics platform

Data Engineering

Data Science

Data Analysis



Microsoft Fabric

Data Engineering

Data Science

Data Analysis

Data Factory

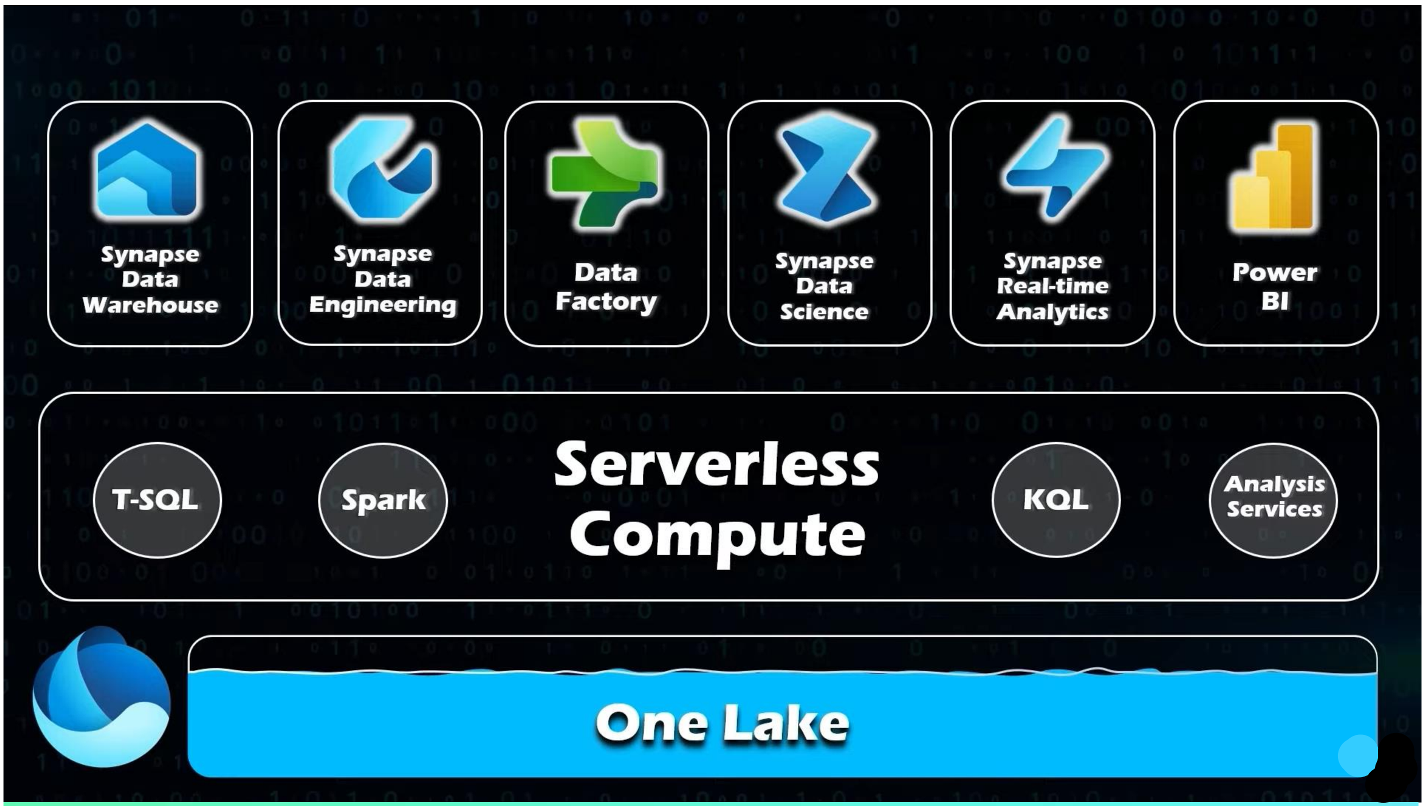
Synapse Data Engineering

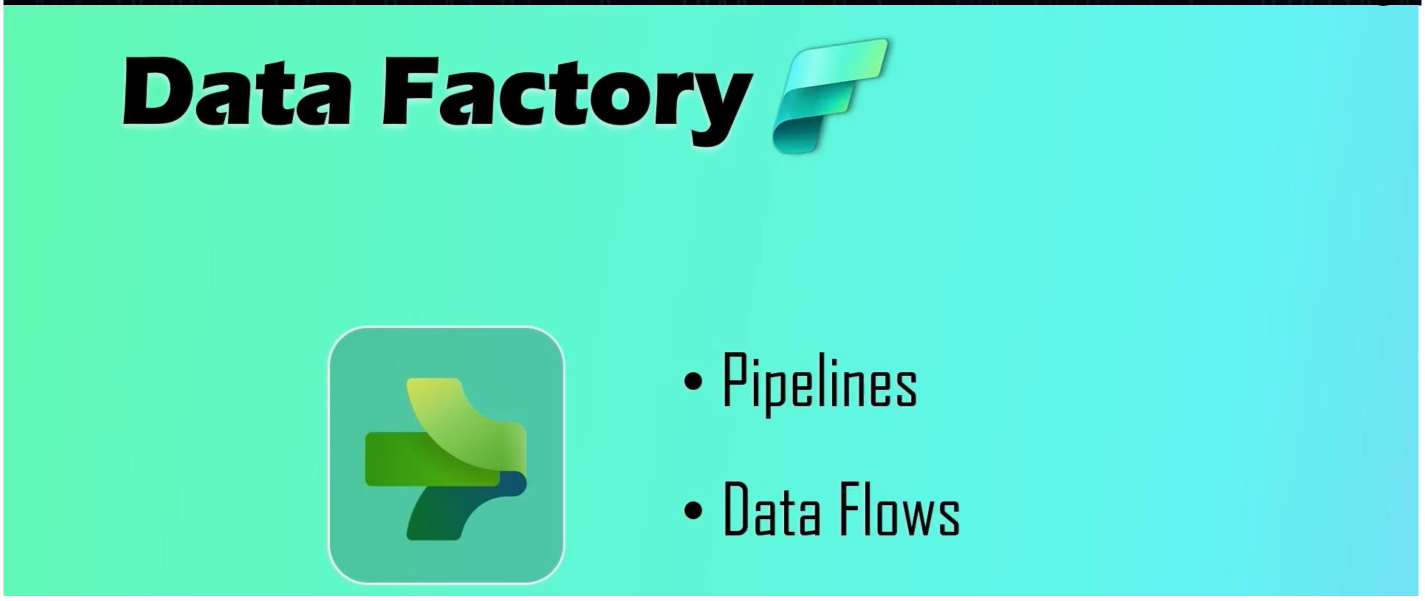
Synapse Data Warehouse

Synapse Real-time Analytics

Synapse Data Science

Power BI





Data Factory

- Pipelines
- Data Flows

Synapse Data Engineering

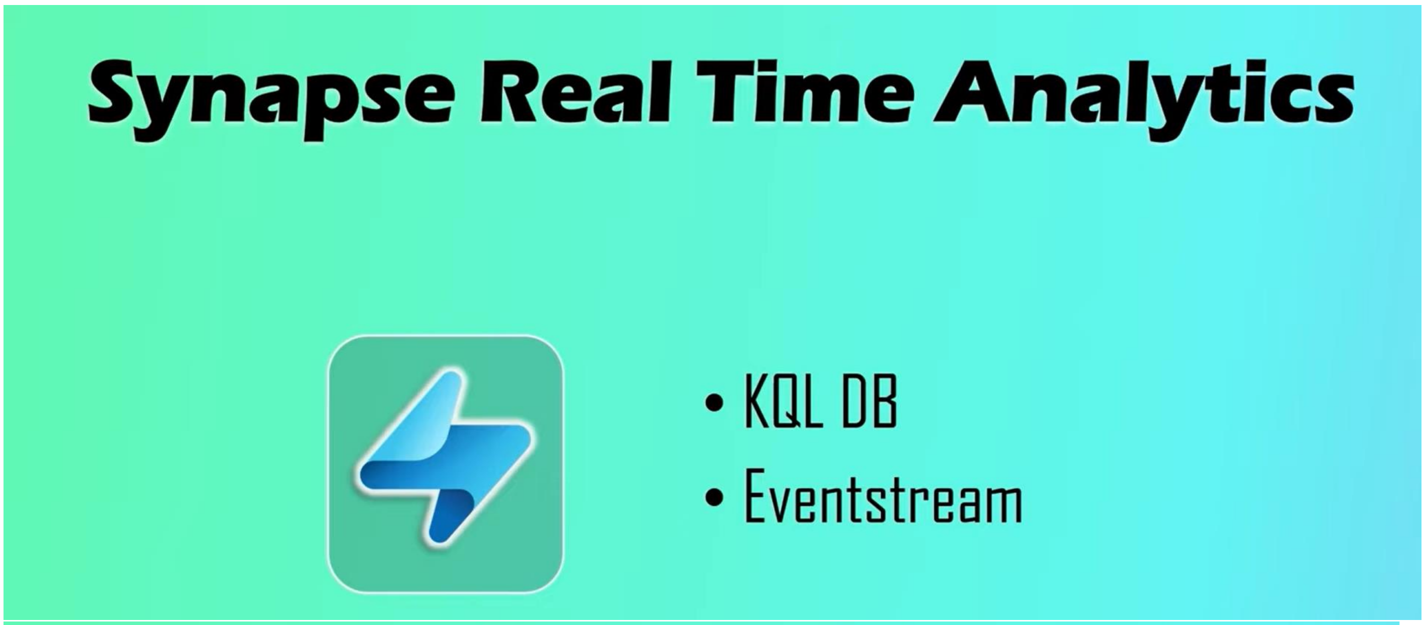
- Notebooks
- Spark Job Definition
- Lakehouse
- Pipeline

Synapse Data Warehouse

- Warehouse
- Pipeline

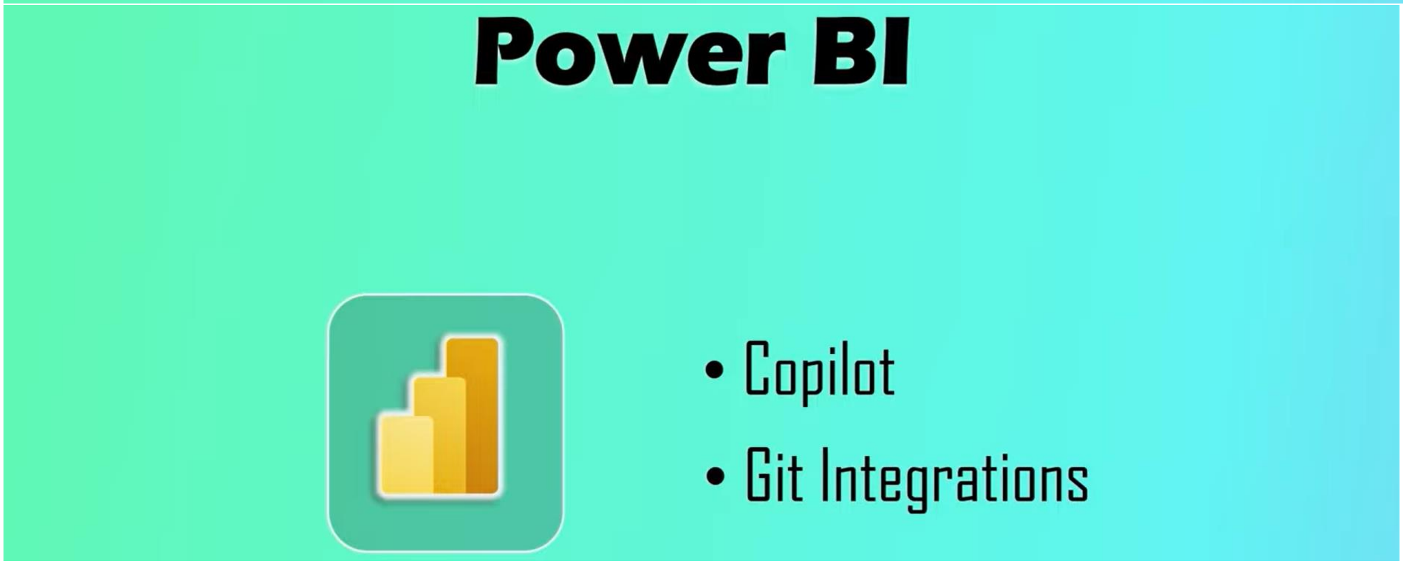
Synapse Data Science

- Model
- Experiment
- Notebook



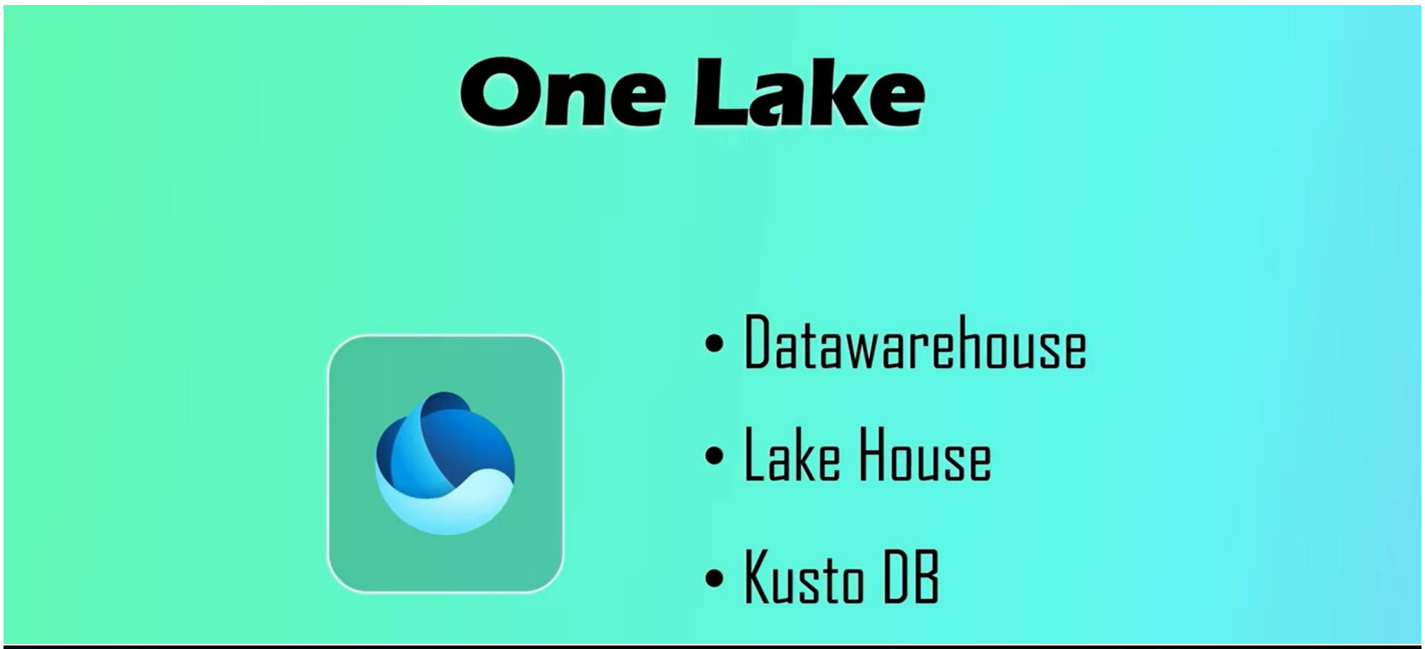
Synapse Real Time Analytics

- KQL DB
- Eventstream



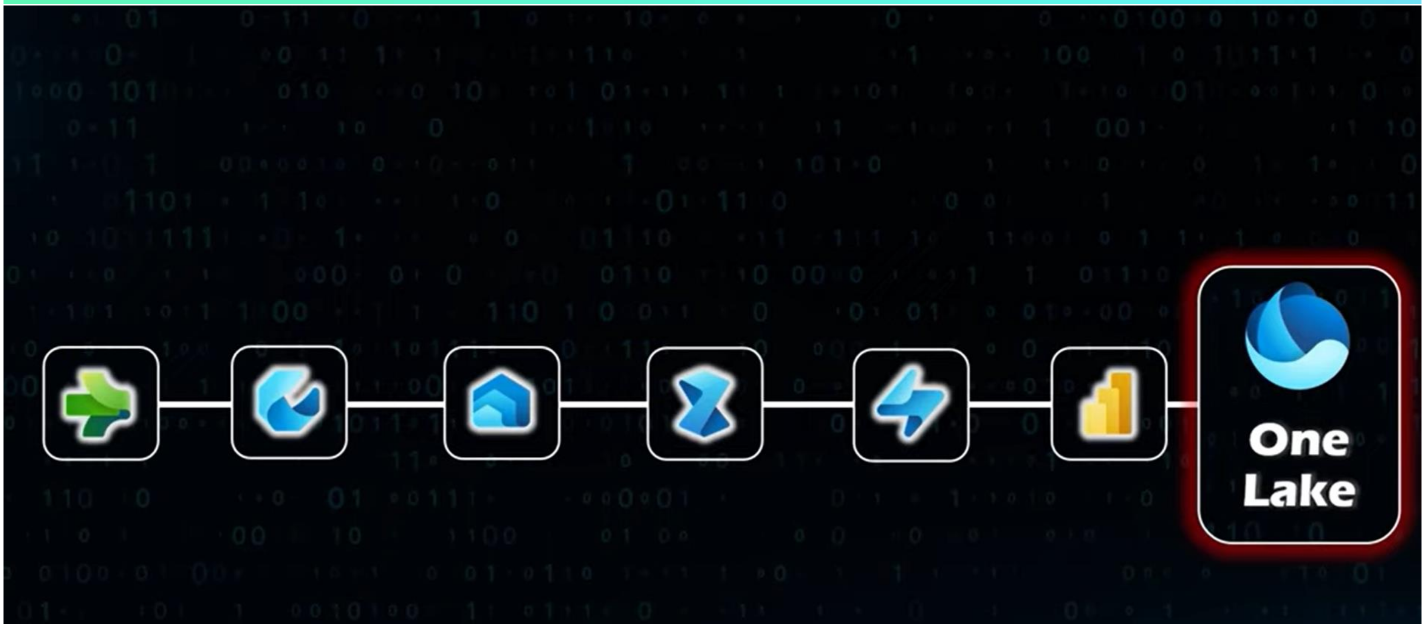
Power BI

- Copilot
- Git Integrations



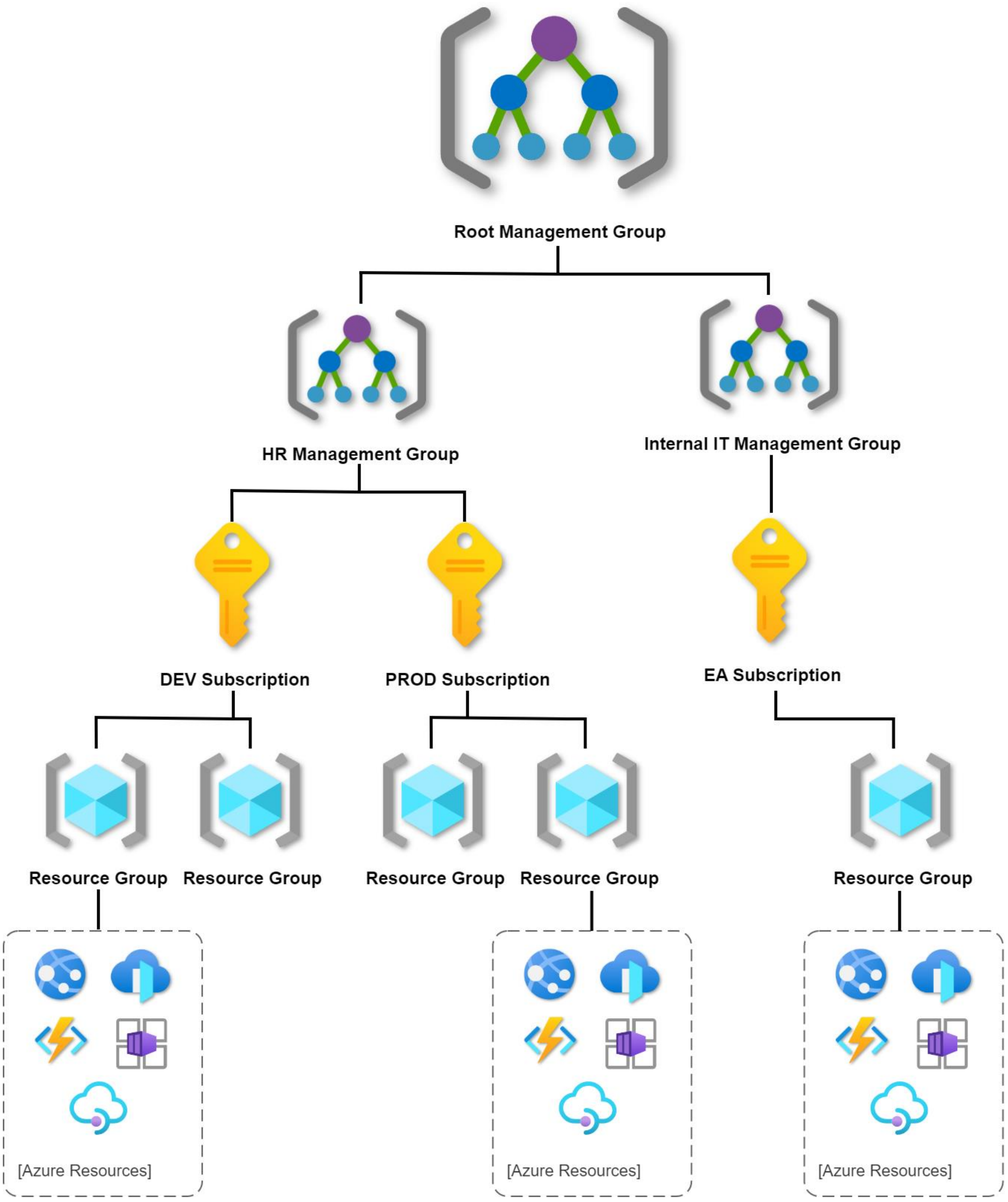
One Lake

- Datawarehouse
- Lake House
- Kusto DB



Part 1: How to create and enable Microsoft Fabric using Azure Portal?

Let’s go Azure Portal



Azure Tenant--->Subscription--->Resource Group--->Create--->Market Place--->Microsoft Fabric--->Click--->Create.

Getting an error Message to create Fabric.
We can easily fix this issue.
Open highest level permission login azure.

Azure Subscription--->Resource Provider--->Search(Fabric)--->Click on Register button.

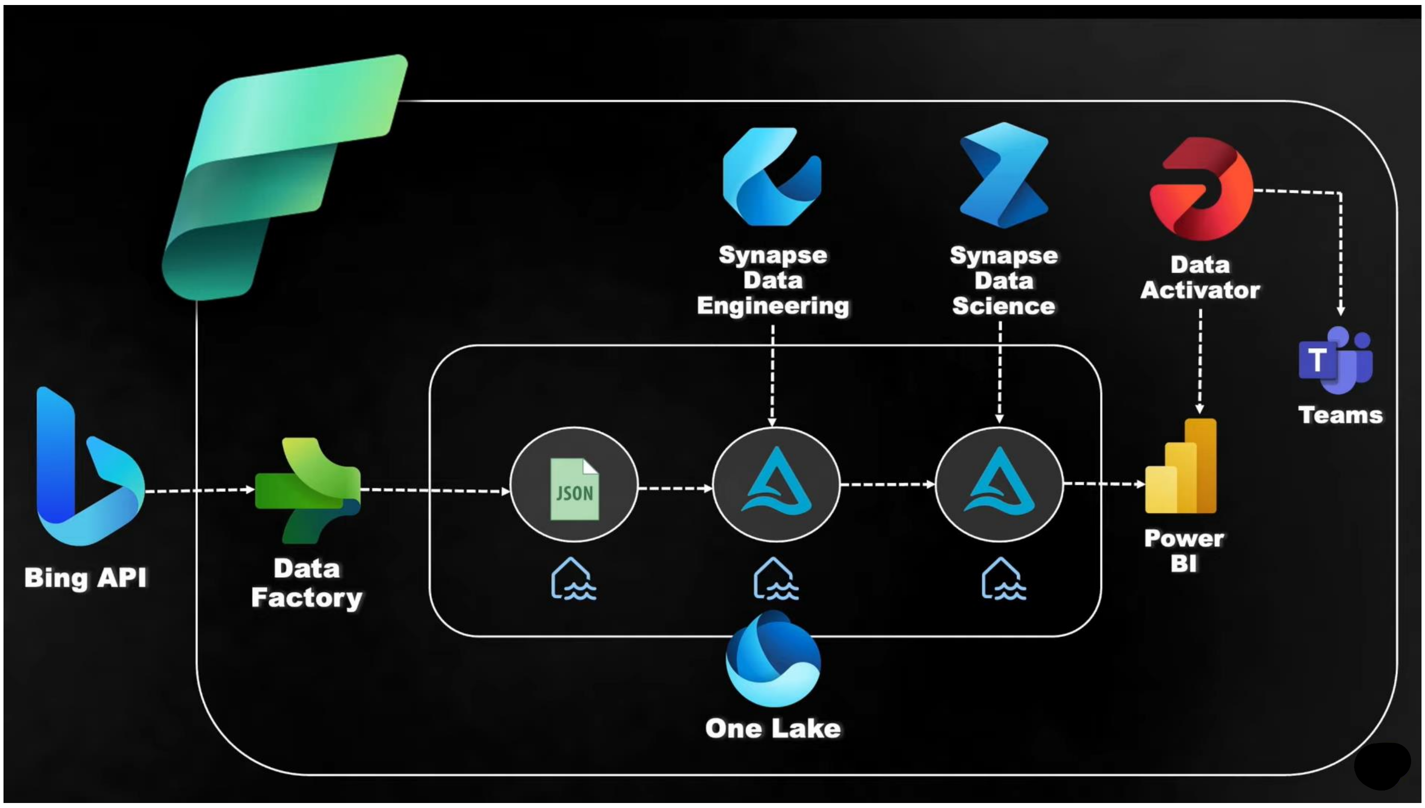
Goto Account--->Refresh

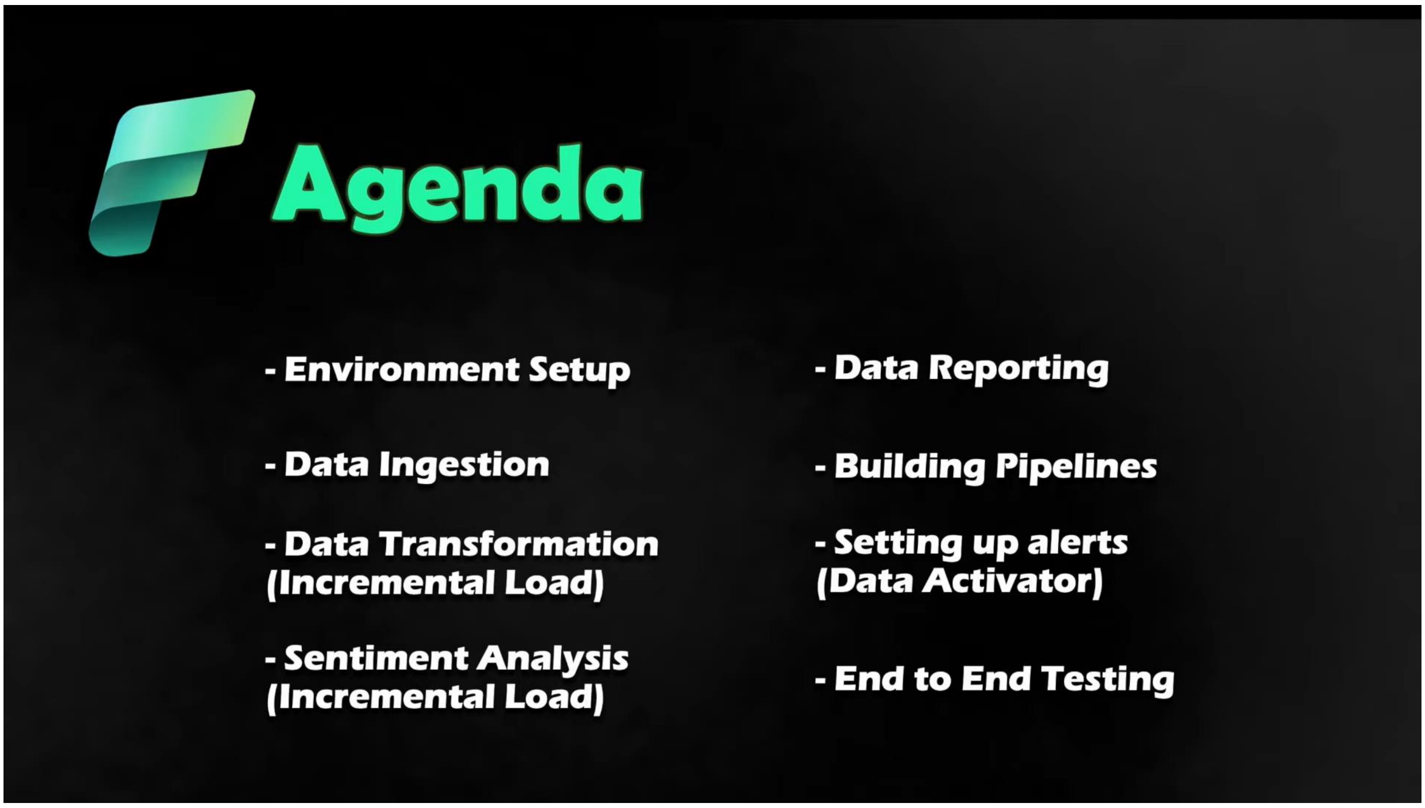
Subscription--->Resource Group--->Capacity Name--->Region (Nearest Location) --->Size(F64)---->Fabric Capacity Administrator--->click and create.

Click Goto Resource.

Goto PowerBI Service--->Workspace--->Workspace Settings--->Premium--->Fabric Capacity--->License Capacity(   )--->Click Apply Button.

Bing



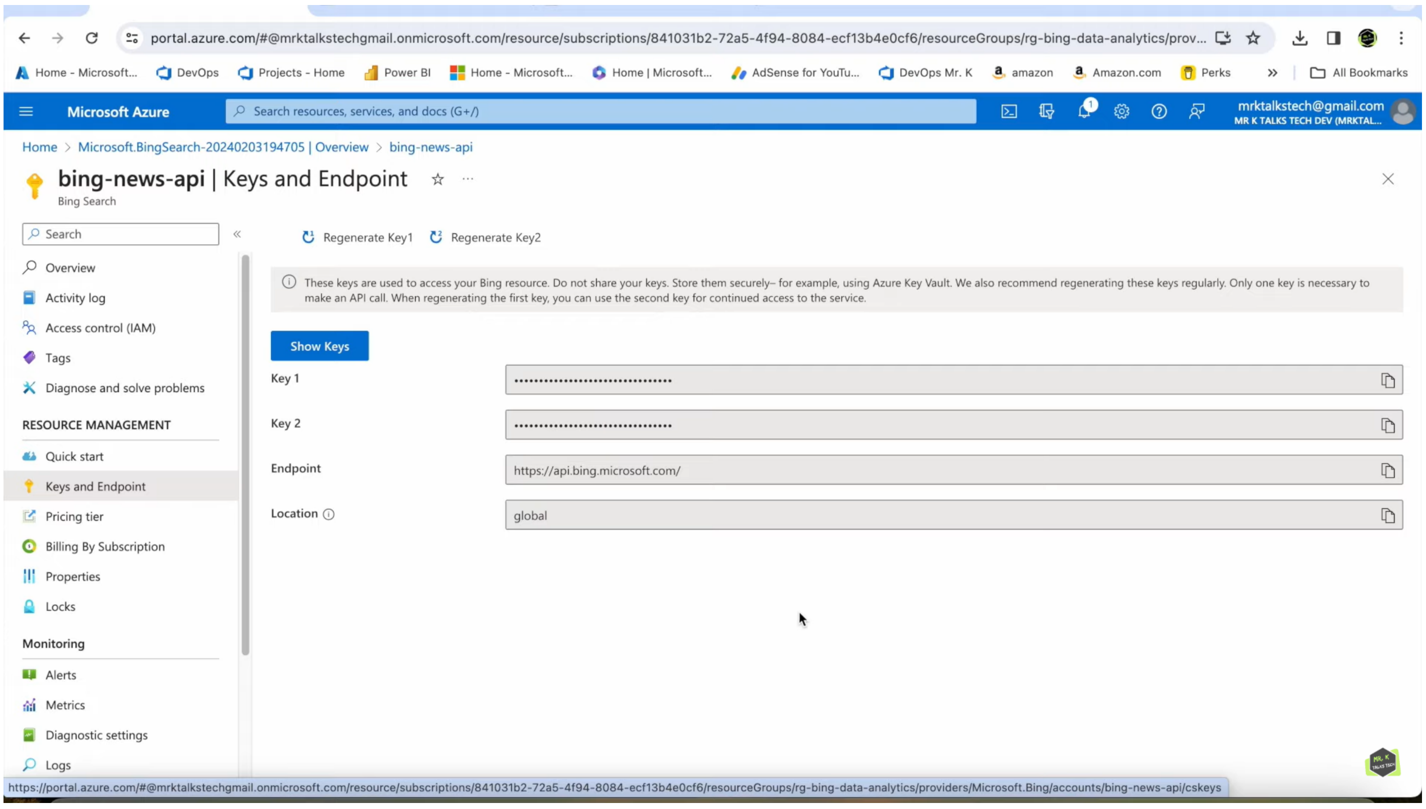


Agenda

- Environment Setup
- Data Ingestion
- Data Transformation (Incremental Load)
- Sentiment Analysis (Incremental Load)
- Data Reporting
- Building Pipelines
- Setting up alerts (Data Activator)
- End to End Testing

Part-2 Create your first Data Ingestion Pipeline in Microsoft Fabric

Create resource group--->Name(Bing Project)--->create RG--->goto overview--->create---> Market place--->in search--->lets type bing--->big search v7--->create.


portal.azure.com/#@mrktalkstechgmail.onmicrosoft.com/resource/subscriptions/841031b2-72a5-4f94-8084-ecf13b4e0cf6/resourceGroups/rg-bing-data-analytics/prov...

Home > Microsoft.BingSearch-20240203194705 | Overview > bing-news-api

bing-news-api | Keys and Endpoint

Bing Search

Regenerate Key1 Regenerate Key2

These keys are used to access your Bing resource. Do not share your keys. Store them securely– for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.

Show Keys

Key 1	••••••••••••••••••••••••••••••••
Key 2	••••••••••••••••••••••••••••••••
Endpoint	https://api.bing.microsoft.com/
Location	global

Goto--->App.powerBI--->Login with organization Account--->by default one Workspace---> Click on left corner use see all tools

In this we use Power BI

Create new workspace--->Premium(Dev)
Not everyone accesses to create workspace Admin only create.
If you have not access to create reach out to admin.

Click on settings icon--->admin portal--->workspace--->settings--->create workspaces--->enabled option(Disable).--->specific security group.

Goto--->data engineering--->lakesouse--->create lakehouse.

Part- 3 Data ingestion

Goto Data Factory--->Data pipeline--->Create pipeline (Name)--->Copy Data (Add to Canvas)
Goto Azure--->Container--->Properties--->URL(Copy)

Goto Azure.portal--->create Blob storage container(One LakeHouse)--->Properties--->URL
ADLS Gen2--->IAM--->Role assignment--->Contributor.

General Source Destination Mapping Settings

Data store Type: Workspace External Sample Dataset

Connection: New (ADLS Gen2)--->SConnection settings elect Refresh

Connection Settings: Test connection

URL: https://rgs.dfs.core.windows.net/test

Connection Credentials:

Connection: create new connection

Authentication Kind: organization Account (Sign in)
Account Key
Shared Access signature (SAS)
Service Principal

File path type: File Path Wildcard file path list of files

File Path: Folder/Directory/Filename Browse preview data
Recursively: Click
File Format: Delimetedtext Settings
Advanced:

Data Store Type:
Workspace External

Workspace data store type:
Lakehouse

Root Folder:
Tables Files
Filepath: Directory/FileName

File Format: Parquet
Advanced:
Copy Behavior:
Max Concurrent Connection:

Run the Pipeline.

Save changes
Run the pipeline

Click on workspace--->click test workspace--->Files(parquet cannot preview the data since it is parquet)
Click on …click on load to tables--->load file to new table.

Bing News Data Analytics-End to End Azure Data Engineering Project using Microsoft Fabric.

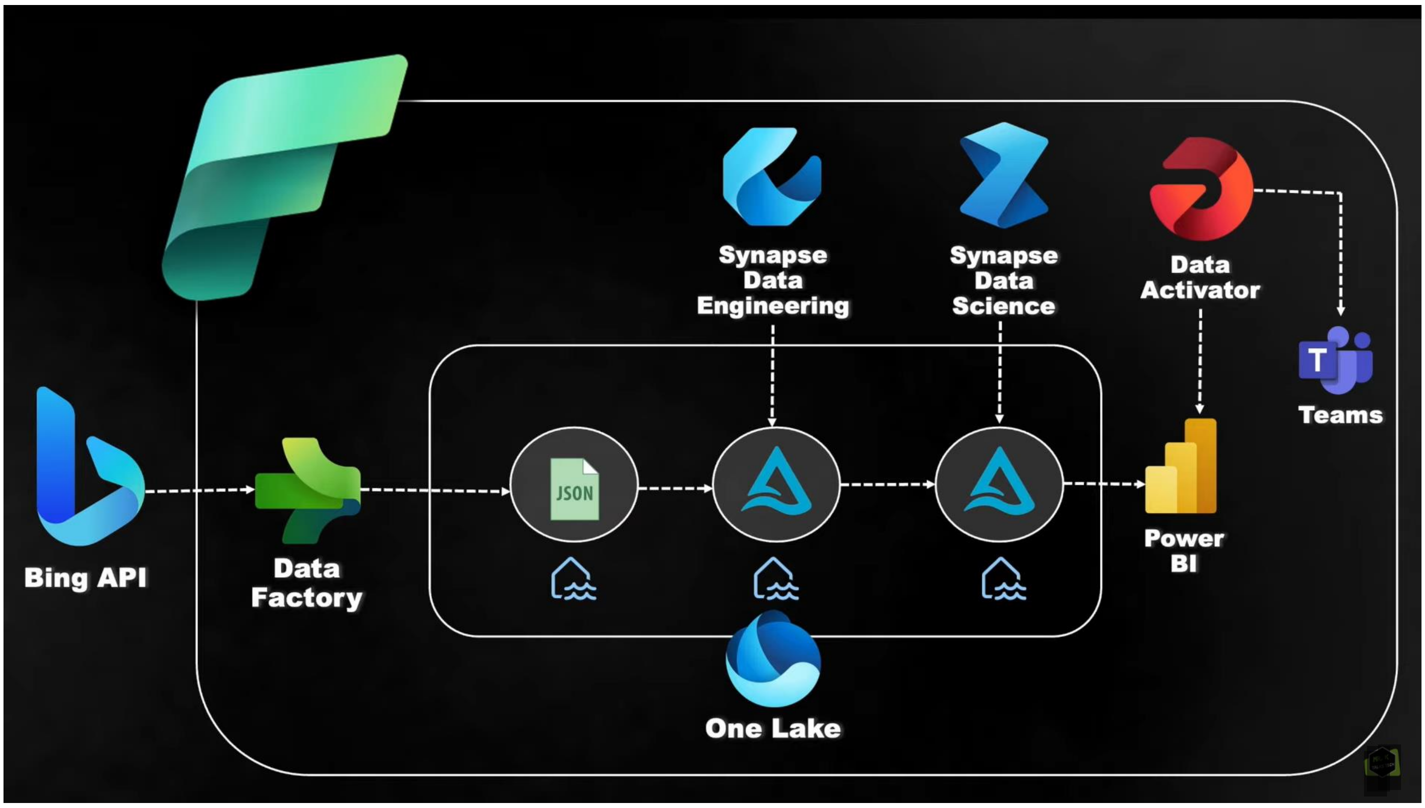
Part 1:

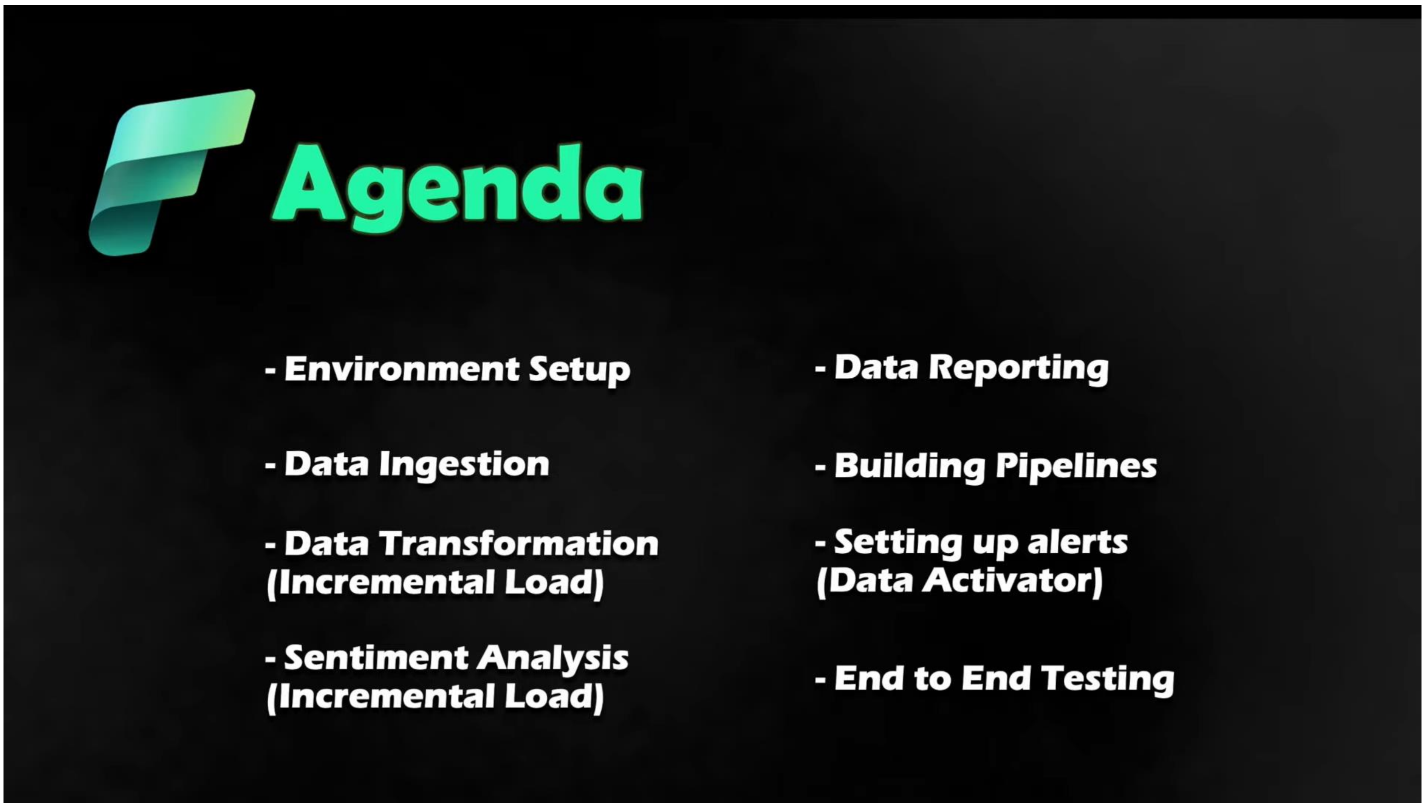
Prerequisites:

API
Python
SQL
Pyspark
ML
Power BI

Project Overview:

Bing News Data Analytics




Agenda

- Environment Setup
- Data Ingestion
- Data Transformation (Incremental Load)
- Sentiment Analysis (Incremental Load)
- Data Reporting
- Building Pipelines
- Setting up alerts (Data Activator)
- End to End Testing

Part 2- Environment setup

Goto Azure--->Resource Group--->Create--->Subscription--->Resource Group name (Bing Data Analytics)--->Region(Nearest Location)--->Create click button.

Goto resource group--->create--->market place--->resource(Bing Search V7)--->Create Resource group--->Name--->Region (Global)--->Pricing tier(F1(3 Calls per second 1k calls per month))--->next--->click (terms & conditions)--->Review+Create.

Goto App.PowerBI--->Workspace Tab--->by default every user access one workspace--->new workspace--->Create a workspace--->name--->description--->Trial--->click apply button.

Goto settings--->admin portal--->workspaces--->workspaces settings--->create workspaces(new workspace experience)--->un check enable--->apply to--->specific security groups.

#Not everyone creates a workspace only admin is required to access to create workspace. I am only the person admin to create workspace. If you don't have access then you can reach out to your admin team to get required access.

Create Lakehouse component
Goto data engineering--->Lakehouse--->new Lakehouse--->name--->create.

Tables
Files

Part 3- Data Ingestion using Data Factory

Step1: goto Azure--->Bing API--->Keys and endpoint

Overview--->make an API call

Bing news search API reference

- Endpoints--->choose second one(copy)
- Headers--->

go to Data Factory tap--->Data Pipeline--->New Data Pipeline

Copy Data Activity--->Add to canvas--->Name

Goto Azure portal--->RG-->Resource(bing-data-analytics)--->overview--->tutorials--->make a API call--->bing news search API((end points--->chose 2nd one)--->(headers--->ocp-Apim-subscription-Key))

General Source Destination Mapping Settings

Data store Type: Workspace External Sample Dataset

Connection: Select Refresh New (Rest)

Connection Settings: Test connection

Base URL: https://api.bing.microsoft.com/v7.0/news/search

Token Audience URL:

Connection credentials: create new connection

Authentication Kind: Anonymous(Sign in)

Basic

Service Principal

Relative URL: ?q==latest+news&count=100&freshness=Day&en-IN Preview Data

Advanced: **File path type: Directort/bing-**

Requested method: GET

Requested interval(ms): 10

Additional Headers: new (Name--->Ocp-Apim-Subscription-key)

--->Value(got key and secrets resource paste)

Data Store Type:

Workspace

External

Workspace data

store type:

Lakehouse: bing_lake_bb

refresh

New(Name)

Root Folder:

Tables Files

latest-news.json

File Format: json Settings

Advanced:

Save

Goto synapse data engineering--->bing_lake_db--->lets click on--->(Tables, Files)

Lets run the pipeline.

Goto lakehouse--->bing_lake_db--->Files(Refresh)

Part-4 Data Transformation using synapse Data Engineering

Goto synapse--->data engineering--->click notebook--->Name--->add Lakehouse(existing Lakehouse)--->bing_lake_db(Tables and Files)--->click ok.

Goto files--->bing-latest-news.json--->Load Data(spark and pandas)--->select spark.

```
df=spark.read.option(“multiline”,”true”).json(“File/bing-latest-news.json)

display(df)

#use serverless compute

df=df.select(“value”)

display(df)

from pyspark.sql.functions import explode

df2 = df.select(df[“value”]).alias(“json_object”))

df2.display()

json_list=df2.toJSON().collect()

print(json_list)

print(json_list[0])


import json

news_json =json.loads(json_list[25])

print(news_json[‘json_object’][‘description’])


#online JSON Parser Online

print(news_json[‘json_object’][‘name’])

print(news_json[‘json_object’][‘description’])

print(news_json[‘json_object’][‘category’])

print(news_json[‘json_object’][‘url’])

print(news_json[‘json_object’][‘image’][“thumbnail”][“contentUrl”])

print(news_json[‘json_object’][‘description’][“provider”][0][‘name’])

print(news_json[‘json_object’][‘datePublished’])
```

```
title=[]

description =[]

category=[]

url=[]

image=[]

provider=[]

datePublished=[]

#Process each Json object in the list

for json_str in json_str:

        try:

                #parse the Json string into a dictionary

                article= json.loads(json_str)

                if article[“json_object”].get(“category”) and

                 article[“json_object].get(“image”,{}).get(“thumbnail”, {}).get(“contentUrl”):

                #extract information from the dictionary

                        title.append(artcle[‘json_object’][‘name’])

                        desctrition.append(artcle[‘json_object’][‘description’])

                        category.append(artcle [‘json_object’][‘category’])

                        url.append(artcle [‘json_object’][‘url’])

                        image.append(artcle [‘json_object’][‘image’][“thumbnail”][“contentUrl”])

                        provider.append(artcle
                        [‘json_object’][‘description’][“provider”][0][‘name’])

                        dataPublished.append(artcle [‘json_object’][‘datePublished’])

        except Exception as e:

                print(f”Error processing JSON object: {e}”)



#defined schema

from pyspark.sql.types import StructType, StructField, StringType

#combine the lists
```

```
data =list(zip(title, description, category, Url, image, provider, datePublished)
Schema =  StructType([ \
    StructField("title",StringType(),True), \
    StructField("description", StringType(),True), \
    StructField("category",StringType(),True), \
    StructField("Url", StringType(), True), \
    StructField("image", StringType(), True), \
    StructField("provider", StringType(), True) \
    StructField("datePublished", StringType(), True) \
  ])
#Create DataFrame
df_cleaned = spark.createDataFrame(Data, schema=schema)

display(df_cleaned)

from pyspark.sql.functions import to_date, date_format
df_cleanned_final=                                                     df_cleaned.withcolumn("datePublished", date_format(to_date("datePublished"),  "dd-MM-yyyy"))
df_cleaned_final(display)
#write the final dataframe to the Lakehouse DB in a Delta Format
df_cleaned_final.write.format("delta").saveAsTable("bing_lake_db.tbl_latest_news")
#write the final dataframe to the Lakehouse DB in a Delta Format
from pyspark.sql.utils import AnalysisException
try:
          table_name=' bing_lake_db.tbl_latest_news'
          df_cleanced_final_write.format("delta").saveAsTable(table_name)
except AnalysisException:
          df_cleanced_final.creatOrReplaceTempView("vw_df_clenaced_final")
          spark.sql(f""" Merge into {table_name} target_table
                    using vw_df_cleanced_format_final source_view
```

```
                on source_view.url=target_tabe.url
                when matched and
                source_view.title<> target_table.title or
                source_view. description <> target_table. description or
                source_view.category<> target_table. category or
                source_view.image<> target_table. image or
                source_view.provider<> target_table. provider or
                source_view.datePublished<> target_table. datePublished
                Then update set *
                “””)
#refresh table

%sql
Select count(*) from bing_lake_db.tbl_latest_news
```

Part 5 - Incremental Load- Type 1

#If you run execute the query

df_cleaned_final.write.format("delta").saveAsTable("bing_lake_db.tbl_latest_news")

#getting an error because of data table already there

#Then overwrite the data

df_cleaned_final.write.format("delta").mode('overwrite').saveAsTable("bing_lake_db.tbl_latest_news")

#Instead of using overwrite use append

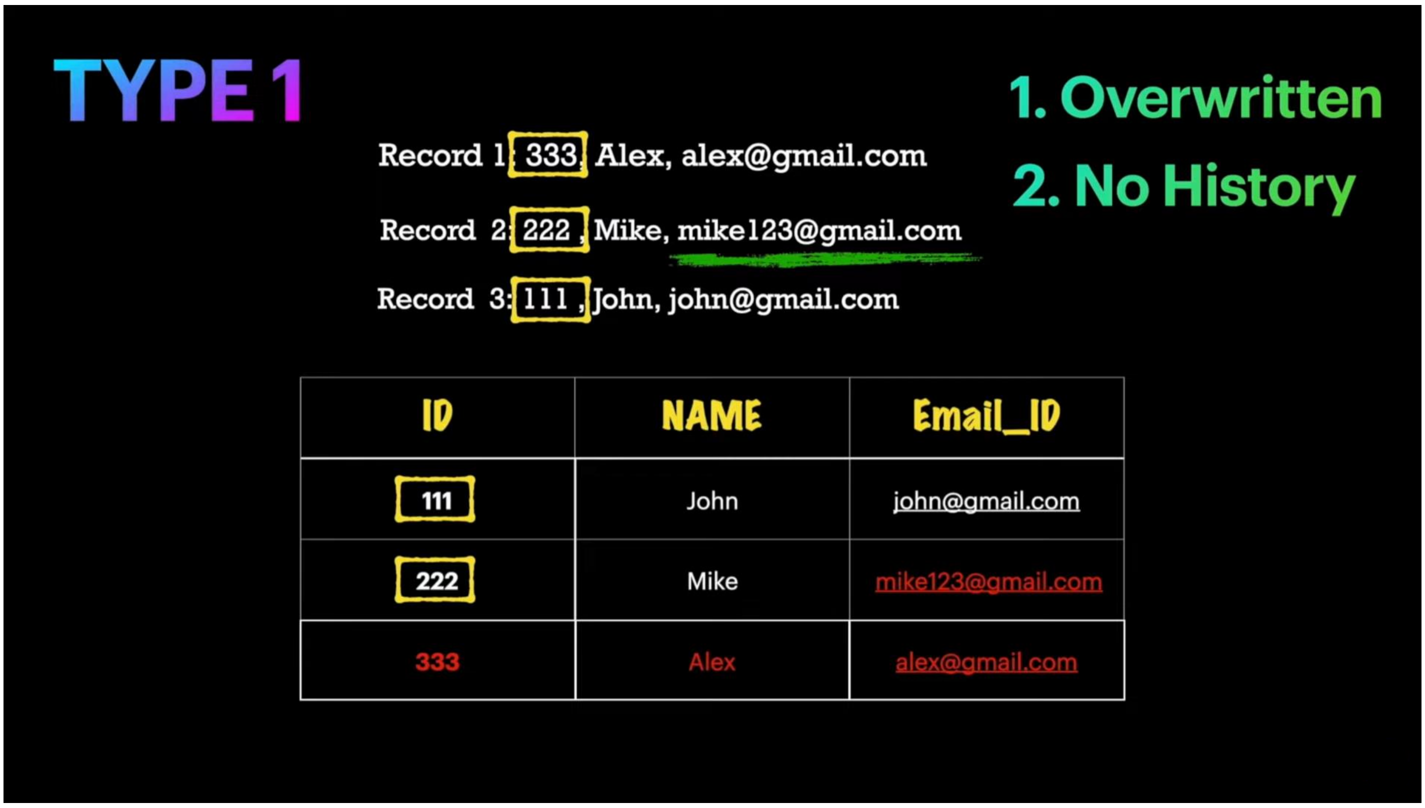
df_cleaned_final.write.format("delta").mode('overwrite').saveAsTable("bing_lake_db.tbl_latest_news")

%sql

Select count(*) from bing_lake_db.tbl_latest_news

#Incremental load using SQL merge data warehousing
Type1 and Type2

- **Type 1 – This model involves overwriting the old current value with the new current value. No history is maintained.**
- **Type 2 – The current and the historical records are kept and maintained in the same file or table.**

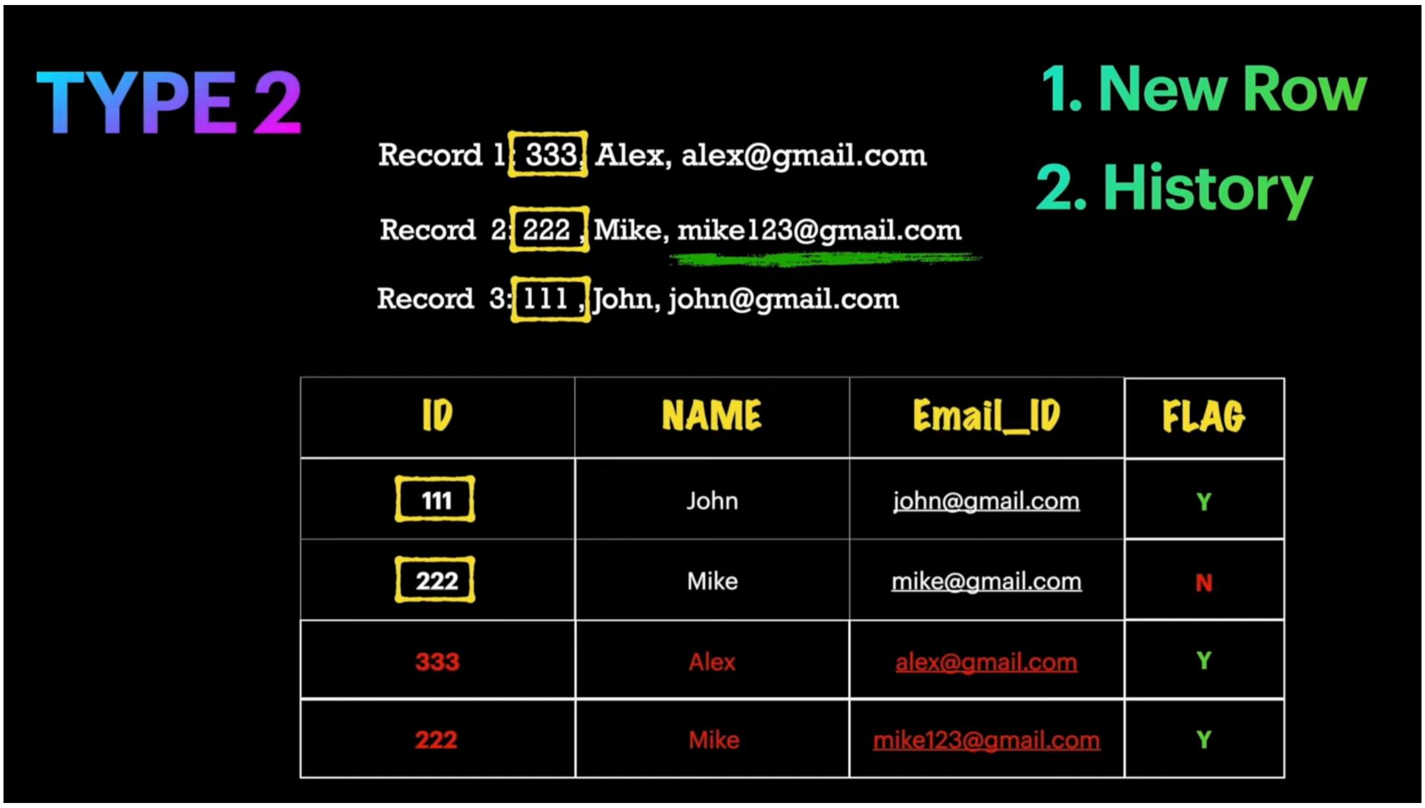


Record 1: 333, Alex, alex@gmail.com

Record 2: 222, Mike, mike123@gmail.com

Record 3: 111, John, john@gmail.com

ID	NAME	Email_ID
111	John	john@gmail.com
222	Mike	mike123@gmail.com
333	Alex	alex@gmail.com


TYPE 2
1. New Row
2. History
Record 1: 333, Alex, alex@gmail.com
Record 2: 222, Mike, mike123@gmail.com
Record 3: 111, John, john@gmail.com

ID	NAME	Email_ID	FLAG
111	John	john@gmail.com	Y
222	Mike	mike@gmail.com	N
333	Alex	alex@gmail.com	Y
222	Mike	mike123@gmail.com	Y

Here we going to use Type 1 is best approach

df_cleaned_final.write.format(“delta”).mode(‘overwrite’).saveAsTable(“bing_lake_db.tbl_latest_news”)

#Execute this one then it will raise exception

#Delete the Table.

from pyspark.sql.functions import to_date, date_format

df_cleanned_final= df_cleaned.withcolumn(“datePublished”, date_format(to_date(“datePublished”), “dd-MM-yyyy”))

df_cleaned_final(display)

#write the final DataFrame to the Lakehouse DB in a Delta Format

df_cleaned_final.write.format(“delta”).saveAsTable(“bing_lake_db.tbl_latest_news”)

#write the final dataframe to the Lakehouse DB in a Delta Format

```
from pyspark.sql.utils import AnalysisException

try:

        table_name=’ bing_lake_db.tbl_latest_news’

        df_cleanced_final.write.format(“delta”).saveAsTable(table_name)

except AnalysisException:

                print(“Table Already Exists”)
        df_cleanced_final.creatOrReplaceTempView(“vw_df_clenaced_final”)
        spark.sql(f””” Merge into {table_name} target_table
                using vw_df_cleanced_format_final source_view
                on source_view.url=target_tabe.url
                when matched and
                source_view.title<> target_table.title or
                source_view. description <> target_table. description or
                source_view.category<> target_table. category or
                source_view.image<> target_table. image or
                source_view.provider<> target_table. provider or
                source_view.datePublished<> target_table. datePublished
                Then update set * when not matched then insert *
                “””)
#refresh table

%sql
Select count(*) from bing_lake_db.tbl_latest_news

#This is actually Type 1 SCD
```

Part-6 Sentiment Analysis using Synapse ML.

Goto power BI service--->click down bottom left---> synapse--->Data Science--->workspace.

Create new notebook--->Name(news-sentiment-analysis)

Lakehouse--->add--->existing lakehouse(add)--->tbl_latest_news

Read the data--->click three dots--->load data--->spark.

```
df=spark.sql(“select * from bing_lake_db.tbl_latest_news limit 1000”)
display(df)
#import setup libraries to perform sentiment analysis


Import synapse.ml.core
from synapse.ml.services import AnalyzeText

#import the model and configure the input and output coulmns

model = (AnalyzeText()
                .setTextCol(“description”)
                .setKind(“SentimentAnalysis”)
                .setOutputCol(“response”)
                .setErrorCol(“error”))

#now successfully configure model

#Apply the model to our dataframe

Result= model.transform(df)

display(result)


#Create sentiment Column

from pyspark.sq.functions import col

sentiment_df = result.withColumn(“sentiment”,
col(“response.documnents.sentiment”))
```

```
display(sentiment_df)

#drop the columns

sentiment_df_final = sentiment_df.drop(“error”,”response”)

display(sentiment_df_final)
```

#incremental load

```
from pyspark.sql.utils import AnalysisException

try:

        table_name=’ bing_lake_db.tbl_sentiment_analysis’

        sentiment_df_final.write.format(“delta”).saveAsTable(table_name)

except AnalysisException:

        print(“Table Already Exists”)

        df_cleanced_final.creatOrReplaceTempView(“vw_df_sentiment_final”)

        spark.sql(f””” Merge into {table_name} target_table

                using vw_df_cleanced_format_final source_view

                on source_view.url=target_tabe.url

                when matched and

                source_view.title<> target_table.title or

                source_view. description <> target_table. description or

                source_view.category<> target_table. category or

                source_view.image<> target_table. image or

                source_view.provider<> target_table. provider or

                source_view.datePublished<> target_table. datePublished

                Then update set * when not matched then insert *

                “””)
```

#lets run cell

Part-7 Building Reports using Power BI.

Goto Lakehouse--->new semantic model--->name(news-dashboard-dataset)

#let's click bottom left--->power BI

In semantic model dataset--->click on it--->discover business insights--->explore this data--->auto create report.--->edit and continue.

--->Create new page instead of changes

Click **title, provider, url, category, datePublished columns** to Table visual

Add slice in canvas and choose(datePublished)

In options change style(dropdown)

Save the report give appropriate name.

Now ok saved successfully saved report now we see this url column web URL format for that firstly we need to go to the semantic model that.

#goto news-dashboard-pipeline

Open semantic model--->click three dots open dataset--->click url column--->advanced--->data category(change to web URL)--->lets go to report--->now url has been changed.

#filtering the data latest

Goto powerbi--->page--->basic filtering--->by value(add column from first dataPublished)--->show items(1)--->apply filter.

Add sentiment column in table.

#what is sentiment percentage of latest particular day

Save it reports.

Create measures goto data model.

Click on semantic model--->open data model.

Create new measure.

Dax code:

```
Positive sentiment %=

If(

        Countrows(filter (‘tbl_sentiment_analysis’, ‘tbl_sentiment_analysis’[sentiment]=

        “Positive”) )>0,

        Divide(

                Calulate (filter (‘tbl_sentiment_analysis’, tbl_sentiment_analysis’[sentiment]=

                “Positive”))),

        Countrows (‘tbl_sentiment_analysis’)

        ),

        0

)

Negative sentiment %=

If(

        Countrows(filter (‘tbl_sentiment_analysis’, ‘tbl_sentiment_analysis’[sentiment]=

        “Negative”) )>0,

        Divide(

                Calulate (filter (‘tbl_sentiment_analysis’, tbl_sentiment_analysis’[sentiment]=

                “Negative”))),

        Countrows (‘tbl_sentiment_analysis’)

        ),

        0

)
```

```
Neutral sentiment %=
If(
        Countrows(filter ('tbl_sentiment_analysis', 'tbl_sentiment_analysis'[sentiment]=
        "Neutral") )>0,
        Divide(
                Calulate (filter ('tbl_sentiment_analysis', tbl_sentiment_analysis'[sentiment]=
                "Neutral"))),
        Countrows ('tbl_sentiment_analysis')
        ),
        0
)
```

Goto dashboard--->edit--->lets add card visual to canvas

3 card visuals---->in fields drag and drop it.

You apply slice with latest sentiment.

Add filters--->filter type--->basic filtering--->Top N(1)--->by value(first datePublished)--->apply filter

Same apply all cases (positive, neutal)

Let's save change

The percentage must be 100%

Let's connect semantic model.

Change with multiply *100.

Part-8 Building Pipelines using Data Factory

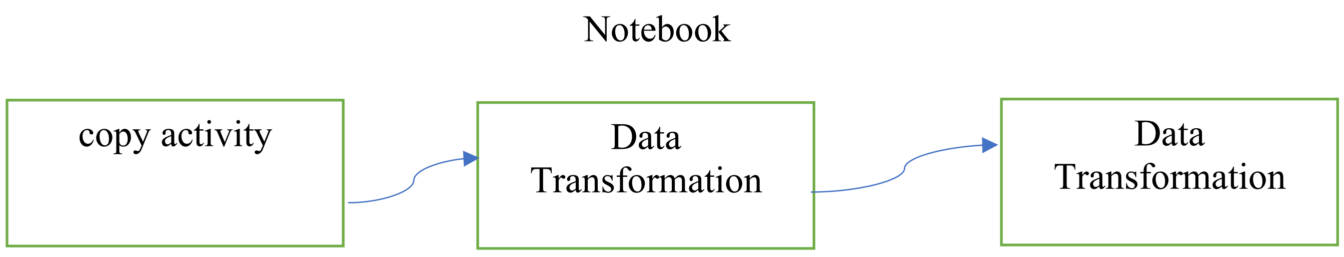
In power BI dev--->workspace--->pipeline--->news-ingestion--->pipeline.

Open

We have already copy activity

General	Settings
Name: Data Transformation	**Workspace:** Sentiment Analysis
	Notebook: process_bing_news
	Base Parameters:

General	Settings
Name: Sentiment_Analysis	**Workspace:** Sentiment_Analysis
	Notebook: news_sentiment_analysis
	Base Parameters:



Save the pipeline.

Before testing the pipeline, I would like to do one thing which automate this workflow even better so for that and I will walk through copy activity.

In source tab as you can see here in the relative URL I have explicitly hot coded search term as latest news during our first injection step so since to make even better

And goto pipeline click on white canvas--->parameters--->name(search_term), type(string), default value(latest news).

Goto pipeline--->source--->relative URL-->add dynamic content--->pipeline expression builder(parameters--->search_term(?q=@{pipeline().parameters.search_term}&count=100&freshness=days&mkt=en-IN).

And click preview data---->click ok for preview the data.

Now we successfully configure pipeline.

And schedule the pipeline.

This pipeline should exactly 6'o clock every morning.

Now click on schedule button--->click on schedule run option(ON)---> repeat (Daily)--->time(06:00)--->start date and time (Today)--->end date and time(next year)--->apply changes.

We don't want wait the pipeline next date pipeline then we can run manually--->click Run Option in pipeline.

After clicking run option --->pipeline run(Name--->Type--->Value(sports)--->lets click on running button.

Lets wait the pipeline running.

Then lets goto power bi--->reports page--->refersh option--->as u can see no latest news update.

Edit option--->as you can see(in data model)--->datePublished column is string type.--->because in filtering option is taking randomly picking up instead of latest news.

Then we need to do

Lets goto workspace--->new-sentiment-analysis(notebook)--->lets run complete notebook

If you run N no. of times merge logic will ignore data again and again.

All the columns are string type

```
#Create another cell

#temporary code – delete after running

df= spark.sql("select * from bing_lake_db.tbl_sentiment_analysis")

from pyspark.sql.functions import col, to_date

df=df.withColumn("datePublished", to_date(col("datePublished"), "dd-MMM-yyyy"))

df.printSchema()

df.write.format('delta').mode("overwrite").option("overwriteSchema","
True").saveAsTable(table_name)

#after that refresh the table.

#Now as we discussed earlier delete now newly added code. Before that
```

Lets goto power bi reports

Click edit option--->open data model option--->edit tables--->refresh button--->you can the latest schema changes in semantic model.--->confim--->it is click datepublished column u see currently in date time format.

Lets jumped into power BI reports--->click on the table--->expand the--->(previously is first datePubished) now its (earliest datePublished)--->you can see different options(earliest, latest, count, count(distinct))--->choose latest option).

Same u can configure in the card visuals.

Click on card visual--->filters--->By value(latest datePublished)--->show items(Top 2).

Lets save changes.

Part 9: Setting up Alerts using Data Activator and End to End Pipeline Testing

Goto app.powerBI--->in the bottom left--->Microsoft fabric--->data Activator--->click on it--->reflex--->click on it--->get your data.(power BI, Simulator)

Lets goto power BI report--->page1 and page2--->goto page2

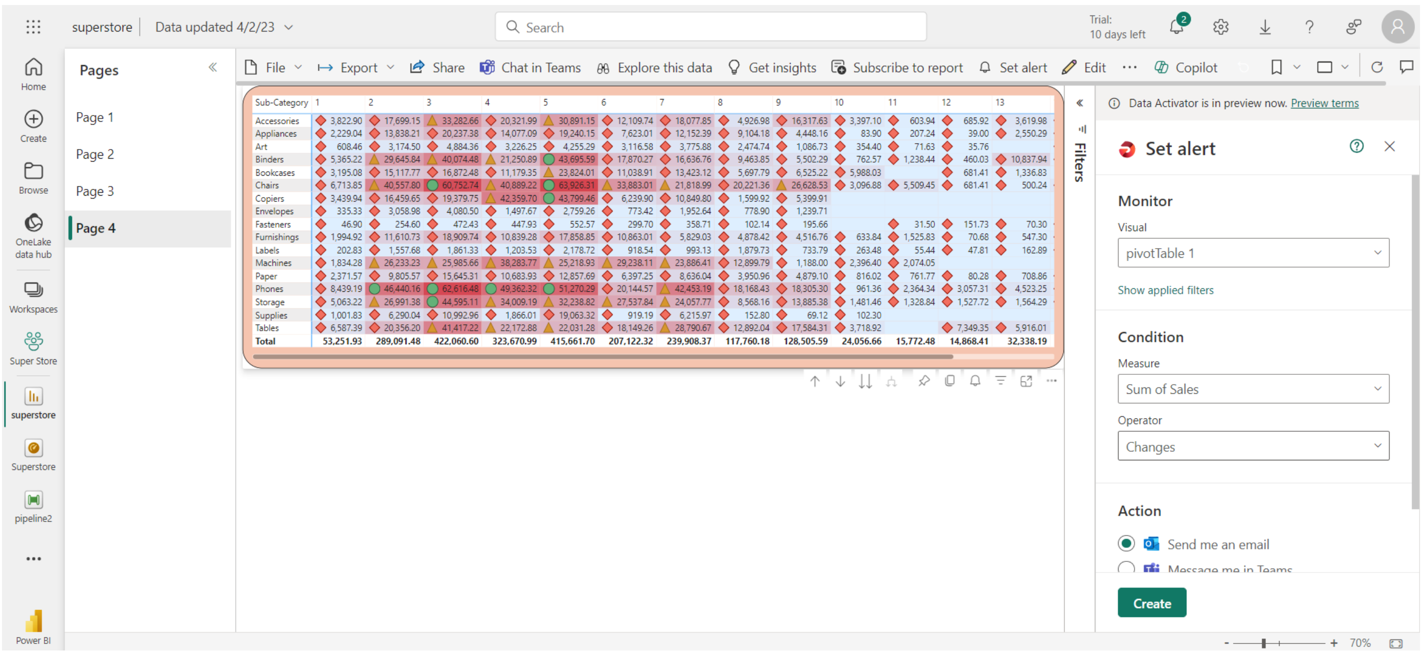
Before configuring alerting in the power BI lets go to data factory to view the pipeline run history.

Goto data factory--->view run history--->click on it.

We can also monitor pipeline run using goto monitoring hub with Microsoft fabric.

Goto power bi report--->to configure the alerts using data activator--->in terms of configure lets choose one visual .

Edit page--->choose positive sentiment--->you find three dots--->set alert--->right side--->



Sub-Category	1	2	3	4	5	6	7	8	9	10	11	12	13
Accessories	◆ 3,822.90	◆ 17,699.15	▲ 33,282.66	◆ 20,321.99	▲ 30,891.15	◆ 12,109.74	◆ 18,077.85	◆ 4,926.98	◆ 16,317.63	◆ 3,397.10	◆ 603.94	◆ 685.92	◆ 3,619.98
Appliances	◆ 2,229.04	◆ 13,838.21	◆ 20,237.38	◆ 14,077.09	◆ 19,240.15	◆ 7,623.01	◆ 12,152.39	◆ 9,104.18	◆ 4,448.16	◆ 83.90	◆ 207.24	◆ 39.00	◆ 2,550.29
Art	◆ 608.46	◆ 3,174.50	◆ 4,884.36	◆ 3,226.25	◆ 4,255.29	◆ 3,116.58	◆ 3,775.88	◆ 2,474.74	◆ 1,086.73	◆ 354.40	◆ 71.63	◆ 35.76	
Binders	◆ 5,365.22	▲ 29,645.84	▲ 40,074.48	▲ 21,250.89	● 43,695.59	◆ 17,870.27	◆ 16,636.76	◆ 9,463.85	◆ 5,502.29	◆ 762.57	◆ 1,238.44	◆ 460.03	◆ 10,837.94
Bookcases	◆ 3,195.08	◆ 15,117.77	◆ 16,872.48	◆ 11,179.35	▲ 23,824.01	◆ 11,038.91	◆ 13,423.12	◆ 5,697.79	◆ 6,525.22	◆ 5,988.03		◆ 681.41	◆ 1,336.83
Chairs	◆ 6,713.85	▲ 40,557.80	● 60,752.74	▲ 40,889.22	● 63,926.31	▲ 33,883.01	▲ 21,818.99	◆ 20,221.36	▲ 26,628.53	◆ 3,096.88	◆ 5,509.45	◆ 681.41	◆ 500.24
Copiers	◆ 3,439.94	◆ 16,459.65	◆ 19,379.75	▲ 42,359.70	● 43,799.46	◆ 6,239.90	◆ 10,849.80	◆ 1,599.92	◆ 5,399.91				
Envelopes	◆ 335.33	◆ 3,058.98	◆ 4,080.50	◆ 1,497.67	◆ 2,759.26	◆ 773.42	◆ 1,952.64	◆ 778.90	◆ 1,239.71				
Fasteners	◆ 46.90	◆ 254.60	◆ 472.43	◆ 447.93	◆ 552.57	◆ 299.70	◆ 358.71	◆ 102.14	◆ 195.66		◆ 31.50	◆ 151.73	◆ 70.30
Furnishings	◆ 1,994.92	◆ 11,610.73	◆ 18,909.74	◆ 10,839.28	◆ 17,858.85	◆ 10,863.01	◆ 5,829.03	◆ 4,878.42	◆ 4,516.76	◆ 633.84	◆ 1,525.83	◆ 70.68	◆ 547.30
Labels	◆ 202.83	◆ 1,557.68	◆ 1,861.33	◆ 1,203.53	◆ 2,178.72	◆ 918.54	◆ 993.13	◆ 1,879.73	◆ 733.79	◆ 263.48	◆ 55.44	◆ 47.81	◆ 162.89
Machines	◆ 1,834.28	▲ 26,233.23	▲ 25,985.66	▲ 38,283.77	▲ 25,218.93	▲ 29,238.11	▲ 23,886.41	◆ 12,899.79	◆ 1,188.00	◆ 2,396.40	◆ 2,074.05		
Paper	◆ 2,371.57	◆ 9,805.57	◆ 15,645.31	◆ 10,683.93	◆ 12,857.69	◆ 6,397.25	◆ 8,636.04	◆ 3,950.96	◆ 4,879.10	◆ 816.02	◆ 761.77	◆ 80.28	◆ 708.86
Phones	◆ 8,439.19	● 46,440.16	● 62,616.48	● 49,362.32	● 51,270.29	◆ 20,144.57	▲ 42,453.19	◆ 18,168.43	◆ 18,305.30	◆ 961.36	◆ 2,364.34	◆ 3,057.31	◆ 4,523.25
Storage	◆ 5,063.22	▲ 26,991.38	● 44,595.11	▲ 34,009.19	▲ 32,238.82	▲ 27,537.84	▲ 24,057.77	◆ 8,568.16	◆ 13,885.38	◆ 1,481.46	◆ 1,328.84	◆ 1,527.72	◆ 1,564.29
Supplies	◆ 1,001.83	◆ 6,290.04	◆ 10,992.96	◆ 1,866.01	◆ 19,063.32	◆ 919.19	◆ 6,215.97	◆ 152.80	◆ 69.12	◆ 102.30			
Tables	◆ 6,587.39	◆ 20,356.20	▲ 41,417.22	▲ 22,172.88	▲ 22,031.28	◆ 18,149.26	▲ 28,790.67	◆ 12,892.04	◆ 17,584.31	◆ 3,718.92		◆ 7,349.35	◆ 5,916.01
Total	**53,251.93**	**289,091.48**	**422,060.60**	**323,670.99**	**415,661.70**	**207,122.32**	**239,908.37**	**117,760.18**	**128,505.59**	**24,056.66**	**15,772.48**	**14,868.41**	**32,338.19**

Data Activator is in preview now. Preview terms

Set alert

Monitor
Visual: pivotTable 1
Show applied filters

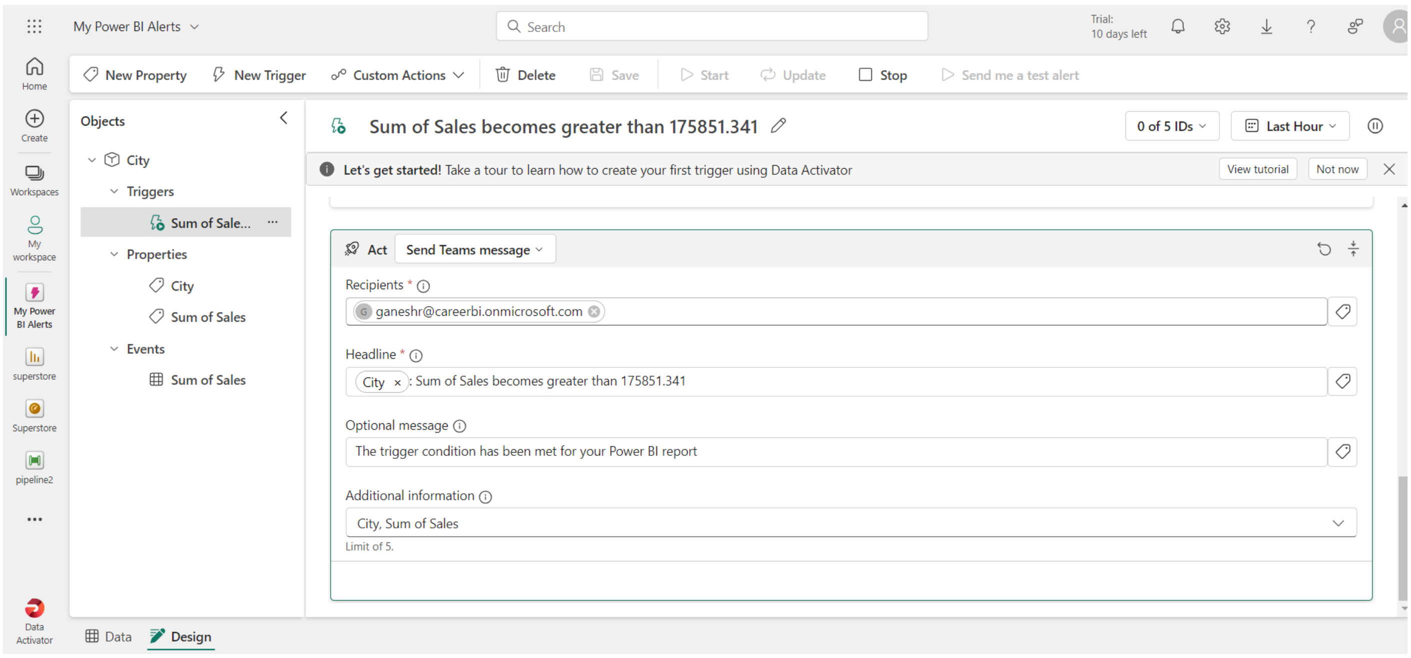
Condition
Measure: Sum of Sales
Operator: Changes

Action
[✓ Send me an email]
Message me in Teams

Create

Choose visual--->measure--->condition(greater than)--->threshold(0.00)--->notificationtype(teams)--->where to save(Workspace--->Item(create new item name(positive sentiment item)--->start my alert)--->create alert.

Lets view alert.


My Power BI Alerts

New Property | New Trigger | Custom Actions | Delete | Save | Start | Update | Stop | Send me a test alert

Objects
- City
  - Triggers
    - Sum of Sale...
  - Properties
    - City
    - Sum of Sales
  - Events
    - Sum of Sales

Sum of Sales becomes greater than 175851.341

Let's get started! Take a tour to learn how to create your first trigger using Data Activator

Act: Send Teams message

Recipients *: ganeshr@careerbi.onmicrosoft.com

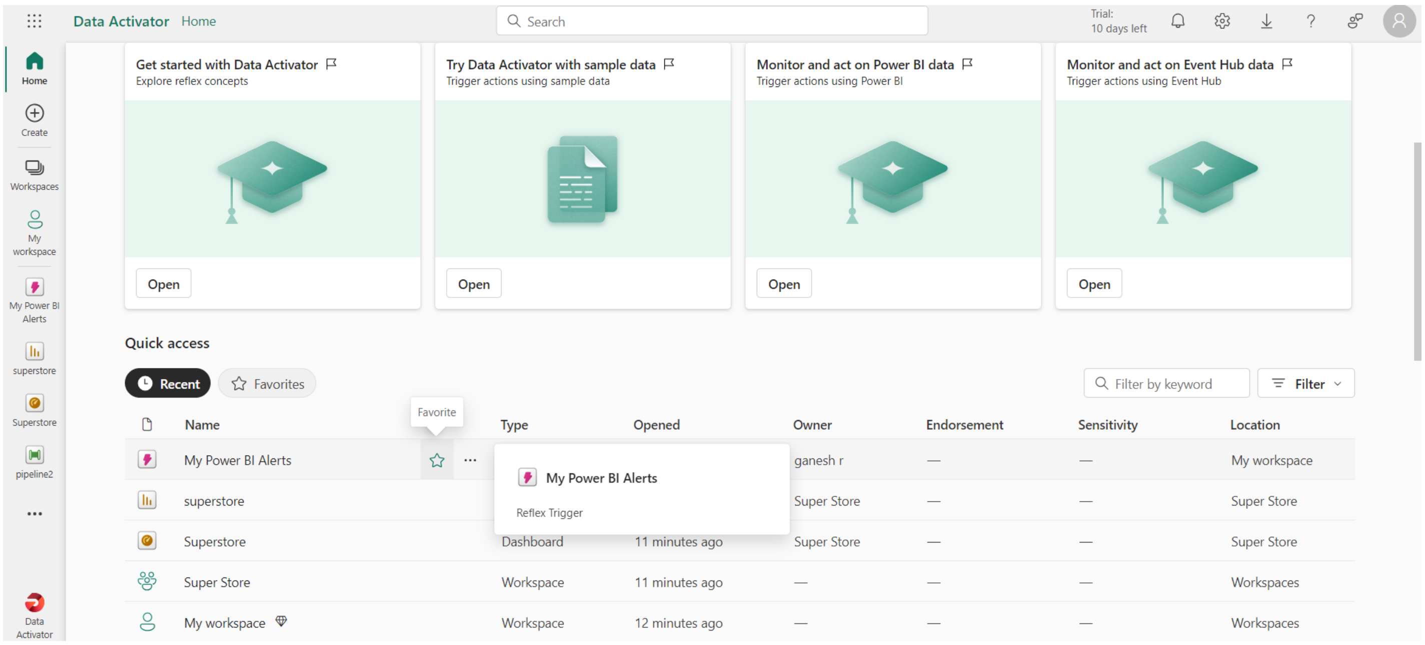
Headline *: City : Sum of Sales becomes greater than 175851.341

Optional message: The trigger condition has been met for your Power BI report

Additional information: City, Sum of Sales

Limit of 5.

Lets go data activator component


Data Activator Home

Get started with Data Activator	Try Data Activator with sample data	Monitor and act on Power BI data	Monitor and act on Event Hub data
Explore reflex concepts	Trigger actions using sample data	Trigger actions using Power BI	Trigger actions using Event Hub

Quick access

Name	Type	Opened	Owner	Endorsement	Sensitivity	Location
My Power BI Alerts	Reflex Trigger		ganesh r	—	—	My workspace
superstore			Super Store	—	—	Super Store
Superstore	Dashboard	11 minutes ago	Super Store	—	—	Super Store
Super Store	Workspace	11 minutes ago	—	—	—	Workspaces
My workspace	Workspace	12 minutes ago	—	—	—	Workspaces

You can update, change, alter, stop and edit it itself

Now we can test it.

Lets goto data engineering component--->open news ingestion pipeline.

Run the pipeline--->click run option--->in the pipeline run(value is now movies)--->click ok

The pipeline has finished end-to-end successfully.

And now check the power BI reports.

It has been updated.

Since the positive sentiment is greater than zero alerts.

So, for that data activator--->the value of the positive sentiment is still Zero.

This is not updated the reason is in the bottom left it is in the design mode.

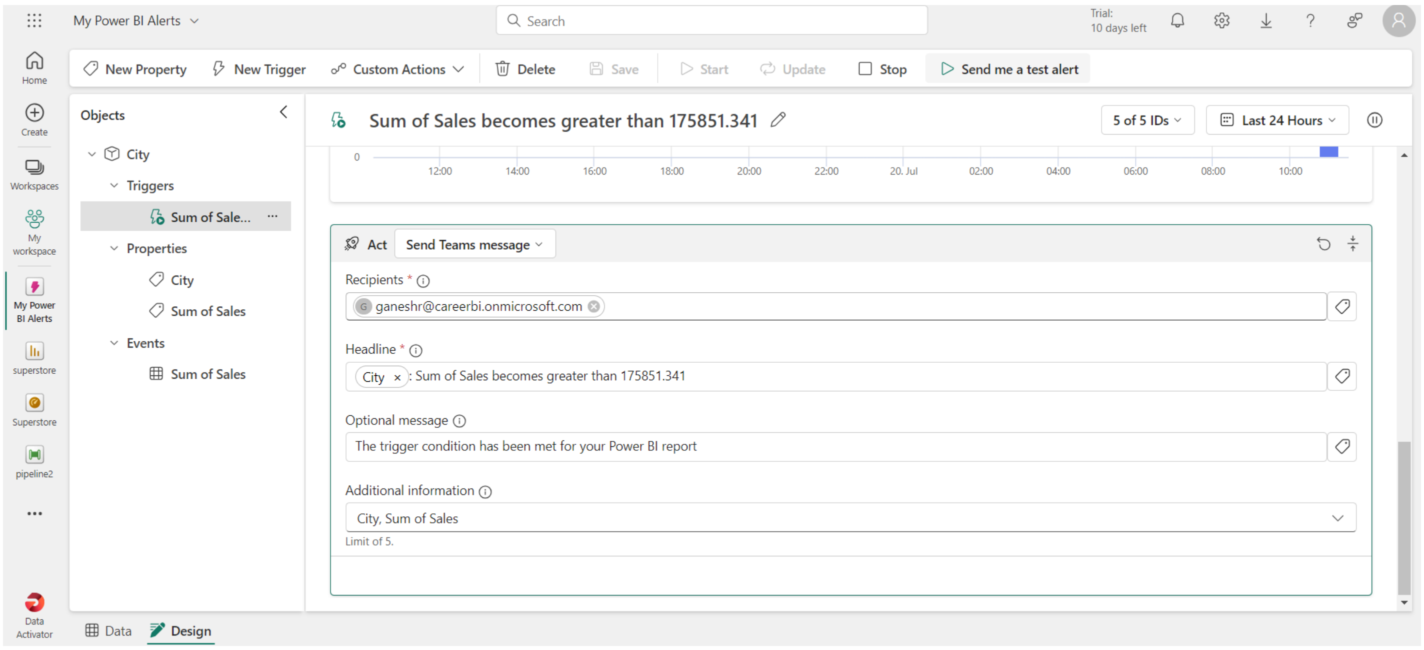
Let's goto--->data mode.

In the right side i symbol click on it you see refresh rate is every 1 hour.

If you streaming data alerts will compute in real time.

Otherwise, the refresh rate is every one hour.

Send me a test alert.



Sum of Sales becomes greater than 175851.341

Act: Send Teams message

Recipients *: ganeshr@careerbi.onmicrosoft.com

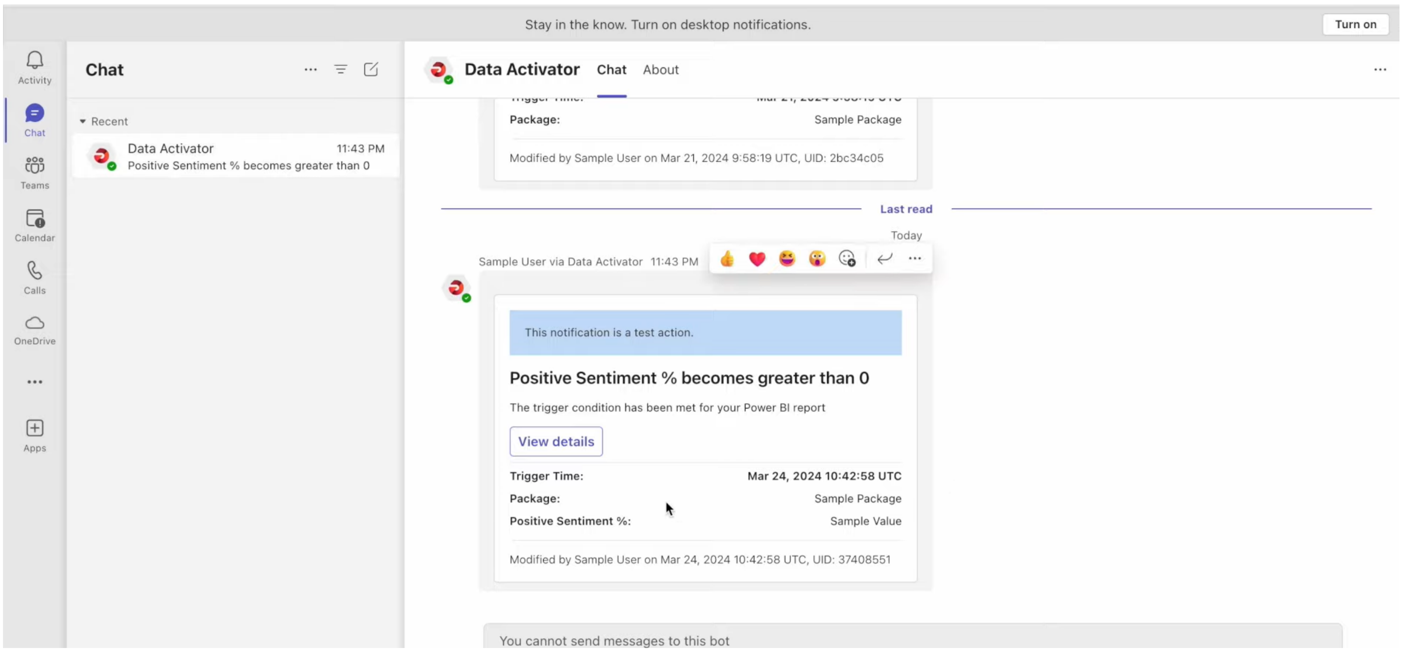
Headline *: City : Sum of Sales becomes greater than 175851.341

Optional message: The trigger condition has been met for your Power BI report

Additional information: City, Sum of Sales

Limit of 5.

The alert has been sent.



Data Activator 11:43 PM
Positive Sentiment % becomes greater than 0

Package: Sample Package

Modified by Sample User on Mar 21, 2024 9:58:19 UTC, UID: 2bc34c05

Last read

Today

Sample User via Data Activator 11:43 PM

This notification is a test action.

Positive Sentiment % becomes greater than 0

The trigger condition has been met for your Power BI report

View details

Trigger Time: Mar 24, 2024 10:42:58 UTC

Package: Sample Package

Positive Sentiment %: Sample Value

Modified by Sample User on Mar 24, 2024 10:42:58 UTC, UID: 37408551

You cannot send messages to this bot

we have completed the section by configuring alerts using data activator and also we have done the complete end to end testing so basically we have completed this end to end project success fully so we have covered all the components in this architecture starting from data injection where we created pipeline in data Factory tree which connects to your Bing API and inject all the news articles as a raw Json structure to the Laos database and after doing this we use the synapse data engineering component to read the ingested raw Json file and process it to a clean and structure Delta table and load that into the same Lakers database and once that is done we use the synapse data science component to read this clean Delta table and the sentiment analysis is performed further by using a description column which contains information about the news articles so basically we use this information and predicted the sentiment of the news using a pre-trained synapse machine learning model and the data is stored as a Delta table in the Leos database this sentiment predictor table is our final table which we are using in the power bi to create reports so in this report we have explored the auto create report option and after that we created a new page which is our main news dashboard based on our requirements in this dashboard we have configured in a way that every time when you open this report only the latest news that are published in the last 24 hours will be displayed in that functionality we have to face some challenges and we have successfully fixed that too also we have created a pipeline in data Factory to orchestrate all the task end to end and have scheduled the pipeline to run every single day exactly at in the morning finally we have used the data activator in the section and configure the alerts for the visuals we created the alert configuration is then finally tested by doing an end pipeline testing by ingesting the new set of data and then saw an example of data activator test alert message that is received in Microsoft teams so this is the end to end flow of this project and we have seen all of these in detail and have completed the project successfully I believe this end to end project will be really useful for everyone Microsoft fabric is a very important tool everyone should know about it if you want to become an Azure data engineer then Microsoft fabric is an important tool to learn so please try to implement this project end to end and only when you actually implement it you will learn the complete functionality of the tool and get a clear understanding of how the tool can be used in the real-time projects I believe everyone watching this video will try and implement it end to end it took me a lot of time and effort to make this entire project I hope you all liked it also I have plans to make more project videos and for that your support is really is really important to me so please support by giving a like And subscribe to my YouTube channel so that's it for today see you in another great video Until Then cheers bye.