Pipeline - Session 1

==========

Use case -

There is a third party service which drops a file named orders.csv in the landing folder

(ADLS Gen2)

our requirement is -

As soon as the file arrives in the landing folder

1. No duplicate order_id in orders.csv

2. Check for valid order_status

if both the conditions are true then move the file to staging folder.

else move it to discarded folder.

Tomorrow if this list of valid order_statuses change then we should be dynamically

able to incorporate the changes.

How to implement this solution

Storage -> Data Factory -> Databricks Notebook

Storage Event trigger

Pipeline - Session 2

==========

Let us create the resources

1. Storage Account - trendytechsa

create a container - sales

in the sales container we need to create 3 folders

landing, staging, discarded

2. Databricks

create a workspace with the name - trendytech-sales-databricks-ws

3. Datafactory

create a datafactory service named - trendytech-sales-df

we need to connect from datafactory to two compnonents..

1. ADLS Gen2

2. Databricks

we will try to use the key vault to store any of the passwords/secret keys.

we need to create linked service

1) storage account

2) Databricks

Pipeline - Session 3

==========

created a linked service for ADLS Gen2

created a token in databricks -

dapi2aeaee7a85299681bb0bfc72026a6edf

Key Vault Service - trendytech-sales-kvs

go to secrets -> Generate/Import

databricks-access-token

we need to create a linked service to azure key vault

1) ADLS Gen2

2) Databricks

3) Key Vault

To create a linked service to key vault we need to allow datafactory in access policies..

Pipeline - Session 4

==========

we need to create Azure SQL Database to keep list of valid status in a lookup.

database - trendytechsqldb

server - trendytechsqlserver

username - tt-sql-user

one our Database is ready..

we need to create a table..

valid_order_status

create table valid_order_status (status_name varchar(50));

insert into valid_order_status values

('ON_HOLD'),('PAYMENT_REVIEW'),('PROCESSING'),('CLOSED'),('SUSPECTED_FRAU

D'),('COMPLETE'),('PENDING'),('CANCELED'),('PENDING_PAYMENT')

sql-password is stored in key vault

1)storage

2)Databricks

3)Datafactory - 3 linked Services

4)Azure SQL DB to store valid order_status table

5)Lets create an interactive cluster in databricks..

Pipeline - Session 5

==========

Lets create an interactive cluster in databricks

storage event trigger in datafactory will trigger the pipeline

it will execute the databricks notebook..

dbutils.fs.mount(

source = 'wasbs://sales@trendytechsa.blob.core.windows.net',

mount_point = '/mnt/sales',

extra_configs={'fs.azure.account.key.trendytechsa.blob.core.windows.net':'TF
3DdEyb

ls8YRM7qoK3h0WEuEgmHbTo6+j/ASBHb5l0v8itlNB7+XUlPrfxZaJG2mREu4
U0aR3q9+

AStJmbGlg=='}

)

In our Databricks Notebook

1. we created a mountpoint

2. we wrote the spark code to read orders.csv in a dataframe and apply the
first

validation.. that order_id should not repeat.

if everything is fine we are creating an orders table.

3. we need to apply the second validation that is if the order_status is valid or
not. To

do this we need connectivity to Azure SQL DB from our databricks notebook.

dbServer = 'trendytechsqlserver'

dbPort = '1433'

dbName = 'trendytechsqldb'

dbUser = 'tt-sql-user'

dbPassword = 'sql-password'

databricksScope = 'salesprojectscope'

connectionUrl =

'jdbc:sqlserver://{}.database.windows.net:{};database={};user={};'.format(dbSer
ver,

dbPort, dbName, dbUser)

dbPassword = dbutils.secrets.get(scope = databricksScope, key=
'sql-password')

connectionProperties = {

'password': dbPassword,

'driver':'com.microsoft.sqlserver.jdbc.SQLServerDriver'

}

The database password we have stored in key vault...

can databricks access the key vault directly?

we have to create a secret scope in databricks

databricks -> secret scope -> key vault

#secrets/createScope

salesprojectscope

validStatusDf = spark.read.jdbc(url = connectionUrl, table = 'dbo.valid_order_status',

properties = connectionProperties )

display(validStatusDf)

Pipeline - Session 6

==========

create a datafactory pipeline..

storage account

datafactory

linked services

Azure SQL Database

KeyVault

Databricks

Storage Event Trigger

=========

List of valid order_status

==========

ON_HOLD

PAYMENT_REVIEW

PROCESSING

CLOSED

SUSPECTED_FRAUD

COMPLETE

PENDING

CANCELED

PENDING_PAYMENT

====

insert into valid_order_status values

('ON_HOLD'),('PAYMENT_REVIEW'),('PROCESSING'),('CLOSED'),('SUSPECTED_FRAU

D'),('COMPLETE'),('PENDING'),('CANCELED'),('PENDING_PAYMENT')

=====

dbutils.fs.mount(

source = 'wasbs://sales@trendytechsa.blob.core.windows.net',

mount_point = '/mnt/sales',

extra_configs={'fs.azure.account.key.trendytechsa.blob.core.windows.net':'HNbSHZ96

WTjdpHviMGwA2ctkwXnOwq28xMmKjh1XVye6RBm/1iiq9lEzTON58LTkMjxBPCR1WnS

p+ASt1slVcA==}

)

===

dbutils.notebook.exit('{"errorFlg": "true", "errorMsg":"Orderid is repeated"}')

===

connectionUrl =

'jdbc:sqlserver://{}.database.windows.net:{};database={};user={};'.format(dbSer
ver,

dbPort, dbName, dbUser)

dbPassword = dbutils.secrets.get(scope = databricksScope, key=
'sql-password')

connectionProperties = {

'password': dbPassword,

'driver':'com.microsoft.sqlserver.jdbc.SQLServerDriver'

}

Pipeline - Session 1

==========

Use case -

There is a third party service which drops a file named orders.csv in the
landing folder (ADLS

Gen2)

our requirement is -

As soon as the file arrives in the landing folder

1. No duplicate order_id in orders.csv

2. Check for valid order_status

if both the conditions are true then move the file to staging folder.

else move it to discarded folder.

Tomorrow if this list of valid order_statuses change then we should be
dynamically able to

incorporate the changes.

How to implement this solution

Storage -> Data Factory -> Databricks Notebook

Storage Event trigger

Pipeline - Session 2

==========

Let us create the resources

1. Storage Account - trendytechsa

create a container - sales

in the sales container we need to create 3 folders

landing, staging, discarded

2. Databricks

create a workspace with the name - trendytech-sales-databricks-ws

3. Datafactory

create a datafactory service named - trendytech-sales-df

we need to connect from datafactory to two compnonents..

1. ADLS Gen2

2. Databricks

we will try to use the key vault to store any of the passwords/secret keys.

we need to create linked service

1) storage account

2) Databricks

Pipeline - Session 3

==========

created a linked service for ADLS Gen2

created a token in databricks -

dapi2aeaee7a85299681bb0bfc72026a6edf

Key Vault Service - trendytech-sales-kvs

go to secrets -> Generate/Import

databricks-access-token

we need to create a linked service to azure key vault

1) ADLS Gen2

2) Databricks

3) Key Vault

To create a linked service to key vault we need to allow datafactory in access policies..

Pipeline - Session 4

==========

we need to create Azure SQL Database to keep list of valid status in a lookup.

database - trendytechsqldb

server - trendytechsqlserver

username - tt-sql-user

one our Database is ready..

we need to create a table..

valid_order_status

create table valid_order_status (status_name varchar(50));

insert into valid_order_status values

('ON_HOLD'),('PAYMENT_REVIEW'),('PROCESSING'),('CLOSED'),('SUSPECTED_FRAUD'),('

COMPLETE'),('PENDING'),('CANCELED'),('PENDING_PAYMENT')

sql-password is stored in key vault

1)storage

2)Databricks

3)Datafactory - 3 linked Services

4)Azure SQL DB to store valid order_status table

5)Lets create an interactive cluster in databricks..

Pipeline - Session 5

==========

Lets create an interactive cluster in databricks

storage event trigger in datafactory will trigger the pipeline

it will execute the databricks notebook..

dbutils.fs.mount(

source = 'wasbs://sales@trendytechsa.blob.core.windows.net',

mount_point = '/mnt/sales',

extra_configs={'fs.azure.account.key.trendytechsa.blob.core.windows.net':'TF3DdEybls8YRM7q

oK3h0WEuEgmHbTo6+j/ASBHb5l0v8itlNB7+XUlPrfxZaJG2mREu4U0aR3q9+AStJmbGlg=='}

)

In our Databricks Notebook

1. we created a mountpoint

2. we wrote the spark code to read orders.csv in a dataframe and apply the first validation.. that

order_id should not repeat.

if everything is fine we are creating an orders table.

3. we need to apply the second validation that is if the order_status is valid or not. To do this we

need connectivity to Azure SQL DB from our databricks notebook.

dbServer = 'trendytechsqlserver'

dbPort = '1433'

dbName = 'trendytechsqldb'

dbUser = 'tt-sql-user'

dbPassword = 'sql-password'

databricksScope = 'salesprojectscope'

connectionUrl =

'jdbc:sqlserver://{}.database.windows.net:{};database={};user={};'.format(dbSer
ver, dbPort,

dbName, dbUser)

dbPassword = dbutils.secrets.get(scope = databricksScope, key=
'sql-password')

connectionProperties = {

'password': dbPassword,

'driver':'com.microsoft.sqlserver.jdbc.SQLServerDriver'

}

The database password we have stored in key vault...

can databricks access the key vault directly?

we have to create a secret scope in databricks

databricks -> secret scope -> key vault

#secrets/createScope

salesprojectscope

validStatusDf = spark.read.jdbc(url = connectionUrl, table =
'dbo.valid_order_status', properties

= connectionProperties )

display(validStatusDf)

Pipeline - Session 6

==========

create a datafactory pipeline..

storage account

datafactory

linked services

Azure SQL Database

KeyVault

Databricks

Storage Event Trigger

=========

List of valid order_status

==========

ON_HOLD

PAYMENT_REVIEW

PROCESSING

CLOSED

SUSPECTED_FRAUD

COMPLETE

PENDING

CANCELED

PENDING_PAYMENT

====

insert into valid_order_status values

('ON_HOLD'),('PAYMENT_REVIEW'),('PROCESSING'),('CLOSED'),('SUSPECTED_FRAUD'),('

COMPLETE'),('PENDING'),('CANCELED'),('PENDING_PAYMENT')

=====

dbutils.fs.mount(

source = 'wasbs://sales@trendytechsa.blob.core.windows.net',

mount_point = '/mnt/sales',

extra_configs={'fs.azure.account.key.trendytechsa.blob.core.windows.net':'HN
bSHZ96WTjdpHvi

MGwA2ctkwXnOwq28xMmKjh1XVye6RBm/1iiq9lEzTON58LTkMjxBPCR1Wn
Sp+ASt1slVcA=='}

)

===

dbutils.notebook.exit('{"errorFlg": "true", "errorMsg":"Orderid is repeated"}')

===

connectionUrl =

'jdbc:sqlserver://{}.database.windows.net:{};database={};user={};'.format(dbSer
ver, dbPort,

dbName, dbUser)

dbPassword = dbutils.secrets.get(scope = databricksScope, key=
'sql-password')

connectionProperties = {

'password': dbPassword,

'driver':'com.microsoft.sqlserver.jdbc.SQLServerDriver'

}

Pipeline - Session 7 (A Quick Recap & Recreating the resources)

==========
===

Third party service that drops a file named "orders.csv" in azure datalake
storage (landing

folder)

we need to perform 2 checks...

1. No duplicate order_id

2. check for valid order_status

if the validation pass, we need to move the file to staging folder else move it to discarded folder.

ADLS Gen2 -> DataFactory ->Databricks Notebook

Creation of resources

==========

1. Storage account - trendytechsa

container - sales

three folders - landing, staging, discarded

2. Databricks workspace - trendytech-sales-databricks-ws

databricksToken - dapia5f242de66f66d3bc6eab25c24dce416

databricksScope

3. Datafactory

linked service

=> Storage account

=> Databricks

=> Keyvault - trendytech-sales-kvv

Pipeline - Session 8

==========

Problem statement - right now our solution caters only to orders.csv and thats why we have

hardcoded it. what if the problem says it can be any file which is in landing folder.

how to solve this...

trigger should dynamically read the filename -> Pipeline -> databricks notebook

orders.csv

[orders,csv]

Pipeline - Session 9

==========

TriggerBody -> Pipeline -> Databricks notebook

quota

we can create a interactive cluster...

Pipeline - Session 10

==========

we made the filename dynamic

but the problem that we were facing is...

the code for mounting works for the first time.. and from second time we need to remove it.

dbutils.fs.mount(

source = 'wasbs://sales@trendytechsa.blob.core.windows.net',

mount_point = '/mnt/sales',

extra_configs={'fs.azure.account.key.trendytechsa.blob.core.windows.net':'XLmj2JG5z9eWHXc

PmVrZBct+mIgmQ6McCLV1RaNm7ux/owLrRAitesAPt4J2czlx5a4mYRt2sUHh+ASt8S0wGQ=='

}

)

storage-account-key

Pipeline - Session 11

==========

1. dynamic filename

2. for mounting we have made generic code

3. storage account key is secured

orders.csv (orders data)

order_items (Amazon S3 in Json format)

customers (will be published by an agency in Azure SQL DB)

order_items

Amazon S3 -> ADLS gen2

bucket name : trendytech-sales

folder : order-items

filename : order_items.json

Pipeline - Session 12

==========

Amazon S3 bucket

AWSAccessKeyId=AKIA3IB4DE2JBVSKT3MI

AWSSecretKey=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

awsAccessKeyId

awsSecretKey

go to datafactory and create a linked service for AWS S3

as soon as orders file arrive in landing folder of ADLS GEN2

we need to get order items file from amazon S3 and bring to ADLS Gen2

Executing the databricks notebook

Customers dataset...

Azure SQL DB

Orders - landing folder (ADLS Gen 2)

Order_item - Amazon S3 bucket

Customer - Azure SQL DB

Pipeline - Session 13

==========

background activity

customers table in Azure SQL DB

customer.csv

datafactory..

1. adls gen2 - available

2. azure sql db - we have to create

```
CREATE TABLE customers (
customer_id INT NOT NULL,
customer_fname VARCHAR(45) NOT NULL,
customer_lname VARCHAR(45) NOT NULL,
customer_email VARCHAR(45) NOT NULL,
customer_password VARCHAR(45) NOT NULL,
customer_street VARCHAR(255) NOT NULL,
customer_city VARCHAR(45) NOT NULL,
customer_state VARCHAR(45) NOT NULL,
customer_zipcode VARCHAR(45) NOT NULL,
PRIMARY KEY (customer_id)
);
```

orders (landing folder in adls gen2)

order item (amazon s3 bucket)

customers (azure sql database)

as soon as the orders file arrive in landing folder..

get order_items from amazon s3 to adls gen2 in folder order_items

Databricks notebook...

validation code...

Pipeline - Session 14

==========

orders (landing folder in adls gen2)

order item (amazon s3 bucket)

customers (azure sql database)

we want to do some validations..

how many orders are placed by each customer and how much amount is spent by each

customer..

orders.csv (orders view in spark)

===

order_items (we need to create spark dataframe from the csv file)

customers (spark dataframe directly through jdbc)

amazon s3 (order_items)

storage account (orders file)

azure sql db (customers)

linked service

key vault

Databricks

interactive vs job cluster

DataFactory

storage event trigger

the code generic

read filename dynamically

pushed the result to sql db for reporting...