Azure End-To-End Data Engineering Project – AdventureWorks Database

# Create the container/image for the express sql server in docker. Set it up with the password (Toasteroven314). It will be named 'sql\_server2022express' in docker. Do this in the terminal in docker.

# Other info for the SQL server instance: Server: (localhost,1433), Authentication: SQL Login, Username: SA, Password: Toasteroven314

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=Toasteroven314' -e "MSSQL\_PID=Express" -p 1433:1433 --name sql\_server2022express -d mcr.microsoft.com/mssql/server:2022-latest

# This is to first make the backup directory in my SQL server container. Then to copy the file from my local computer in my Data Projects folder over to the SQL server container in docker. Note the file name has LT in it (AdventureWorksLT2022.bak).

docker exec -it sqlserver2022express mkdir -p /var/opt/mssql/backup

docker cp "/Users/milanmarkovic/Data Projects/SQL Server/AdventureWorksLT2022.bak" sqlserver2022express:/var/opt/mssql/backup/

# In a .sql file (in the Azure data studio connection I made to the SQL Server docker, I hit ‘new query’), I created a login for the SQL Server database:

CREATE LOGIN milanmarkovic WITH PASSWORD = 'Toasteroven314'

create user milanmarkovic for login milanmarkovic

# This login will go into Key Vault on Azure, so that I can connect my SQL Server database to Azure Data Factory.

SHIR = Self Hosted Integration Runtime

# For the linked service I made to connect Azure Data Factory to my ‘on prem’ SQL Server database: Azure Data Factory connects to the SQL Server instance I have running on a docker container, whose connection I am tunneling to a public ip using ngrok (I will have to change Server name in the linked service configuration on Data Factory every time I create a new tunnel (8hr limit with ngrok free plan) because tunnel will have different forwarding address every 8 hours). So it's an abstracted 'on prem' database. All this because I have a Mac and am using SQL Server. Postgre would have probably been easier.

# Example Server name (notice comma has to be there): 8.tcp.ngrok.io,18908

Part 1 Completed – I have made a pipeline that does the Extract part of ETL now. The pipeline access the on prem SQL Server database using a username and password, gets back a list of table names, and copies the data off those tables into an Azure data lake (into bronze specifically). This is the ingestion process.

A screenshot of a computer

Description automatically generated

Part 2 Completed – Now I have done the transformations from bronze -> silver -> gold in databricks. I did not use Azure databricks premium, however, because I did not want to pay or use free 14 day trial. I instead worked around and accessed data lake using SAS tokens in my personal databricks community edition.

I got the SAS tokens in my storage account (myadventureworkssg), containers, Security + networking, Shared access signature. I made one that lasts until 2099 and used that one. It’s in the databricks notebook when I need it.

So really, I never used the Azure Databricks workspace I made in my resource group for this project, which would have needed me to sign up for databricks premium, which I didn’t want to. So I didn’t setup the automatic Azure Databricks steps in the pipeline workflow either. Instead I’ll have to run the pipeline, and then run my databricks community edition notebooks. The way I set it up, my databricks community edition takes files from the bronze layer, transforms them, puts them into silver layer, transforms them further, puts them into gold layer. All across 2 notebooks in my personal databricks community edition workspace.

But, I’ll really only need to run these notebooks now when there is new data to put into gold storage. I only did this as a demo, so I won’t need to rerun these notebooks, it won’t do anything, the data is already in gold storage in Azure data lake. But I can if I want I guess, it’s free (but I don’t think it would be free if I wasn’t on DB community edition).

Onto Synapse, I created a serverless SQL database (I think an Azure SQL database specifically now) called gold\_db. I used a script the tutorial provided that will create a view for all the tables. Then I created a link service to connect this serverless SQL database, to let me use the stored procedure from the synapse pipeline.

… Then follow some video steps…

Then at the end of Synapse part, this pipeline only needs to be run if there are schema changes. I think that means if you simply add or delete some rows, you don’t have to run this synapse pipeline again. This is because the view that you created for the synapse pipeline can adjust to new rows being added or deleted. But the view cannot account for schema changes to the table (eg, new column added or column types changed) (the only time the pipeline would have to change is if how you view the data changes, and that only happens if the schema changes). I think.

Next step would be to connect data to Power BI (or Tableau in my case).