A picture containing shape

Description automatically generated

**Using Azure Synapse Notebook and Pipeline for ETL**

**Applies to:**

Azure Data Engineering, SAP Azure Synapse Analytics

**Summary**

This article explains step by step process about how to use Azure Synapse spark notebook and pipeline to perform extraction, transformation and load(ETL)

**Author**: Prakash Kumar Sahu

**Designation**: Senior Data Analyst

**Date**: 14 October 2022

**Table of Contents**

Synapse Introduction

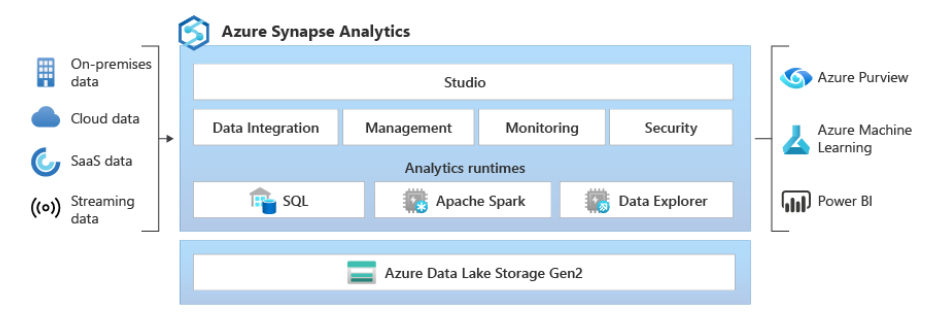
Business Scenario

Steps and Code

Integrating Synapse notebook into pipeline

**Synapse Introduction:**

Azure Synapse is an enterprise analytics service that accelerates time to insight across data warehouses and big data systems. Azure Synapse brings together the best of SQL technologies used in enterprise data warehousing, Spark technologies used for big data, Data Explorer for log and time series analytics, Pipelines for data integration and ETL/ELT, and deep integration with other Azure services such as Power BI, CosmosDB, and AzureML.



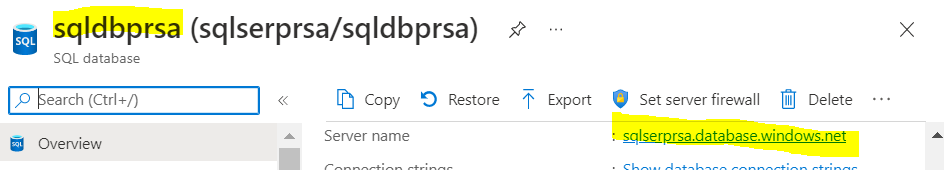
**Business Scenario:**

Source data is there in Azure Data lake Storage Gen2(ADLS Gen2). This data needs to be extracted, transformed by applying business logic and then finally needs to be loaded in Azure SQL database for reporting purpose. Instead of using different combination of Azure services like Azure Data Factory(ADF) and Azure Databricks , business want to use just single service Azure Synapse Analytics to perform ETL activity.

**Steps and Code:**

Below are step by step process with sample code snippet to perform ETL activity and to automate the ETL using Synapse Pipeline.

**Step 1**. Go to Azure SQL Database and note down Server Name and database name.



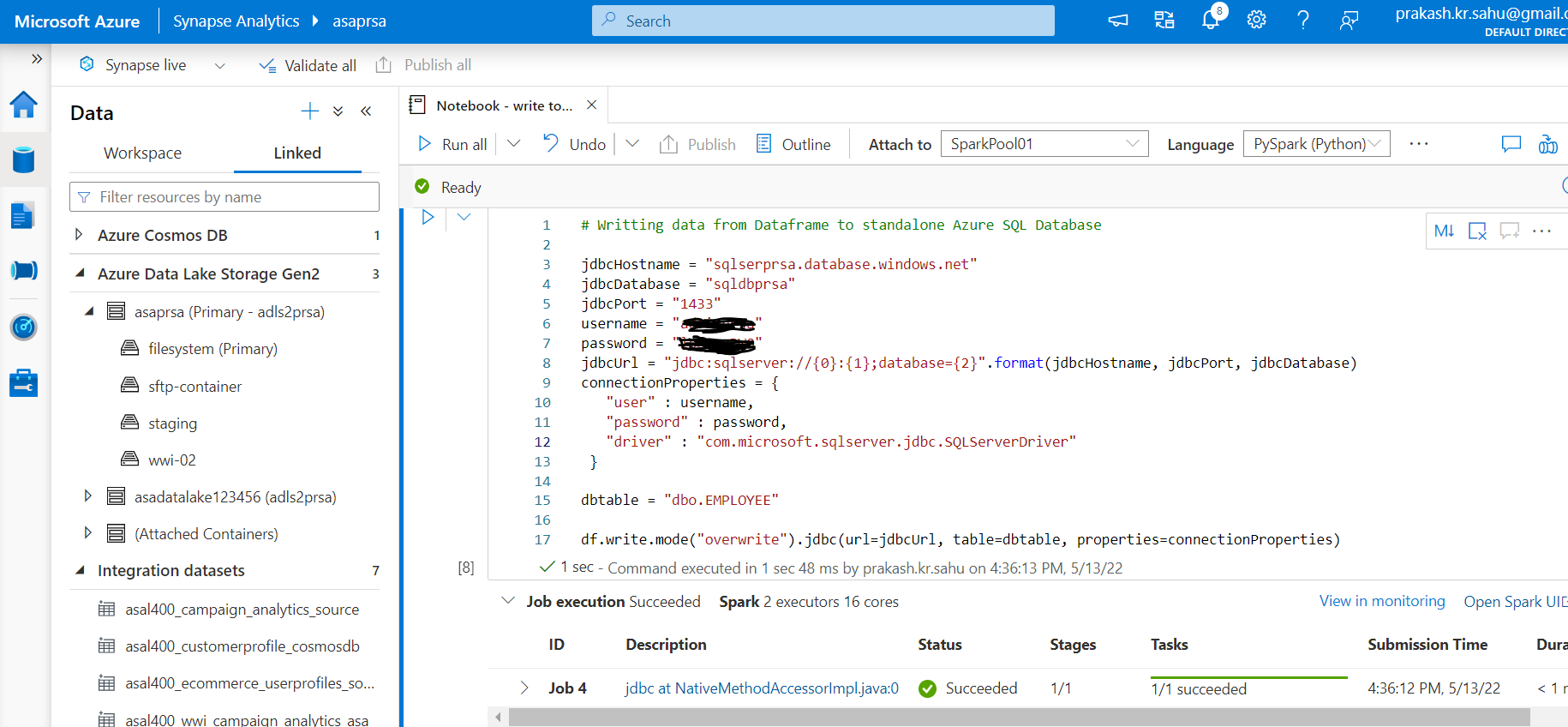
**Step 2**. Go to Connection string and then JDBC tab and note down port number.



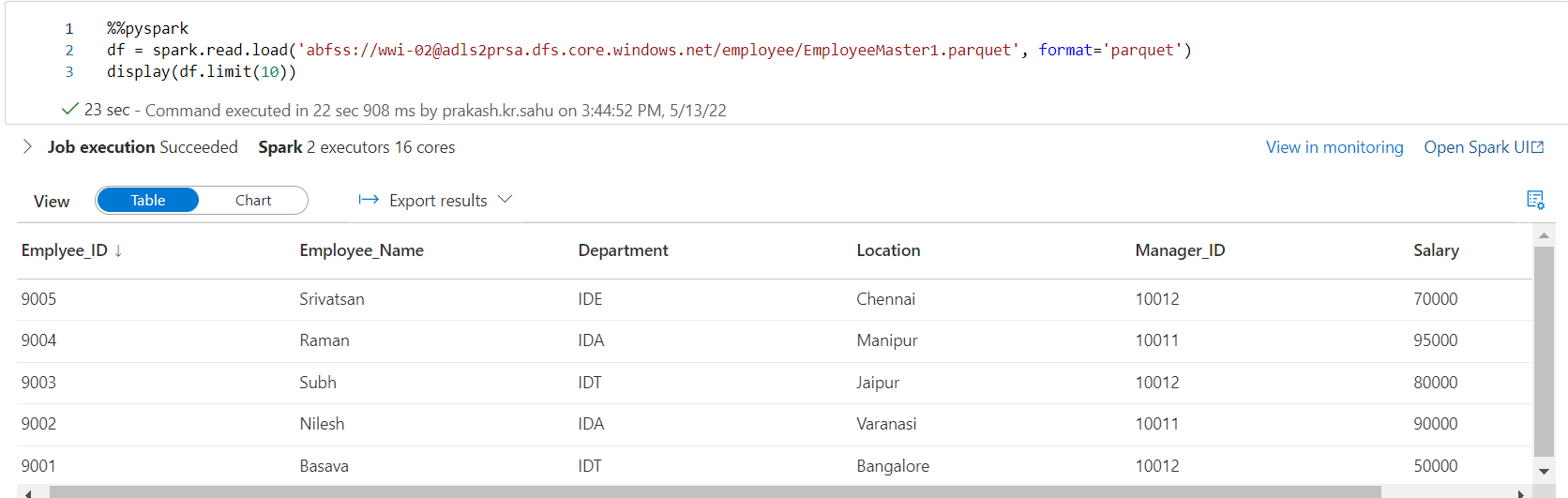
**Step 3**. Get the user id and password of Azure SQL Server. Also note down table name where you want to write data from dataframe.

**Step 4**. In Synapse Analytics Studio, create a notebook and write code like below code

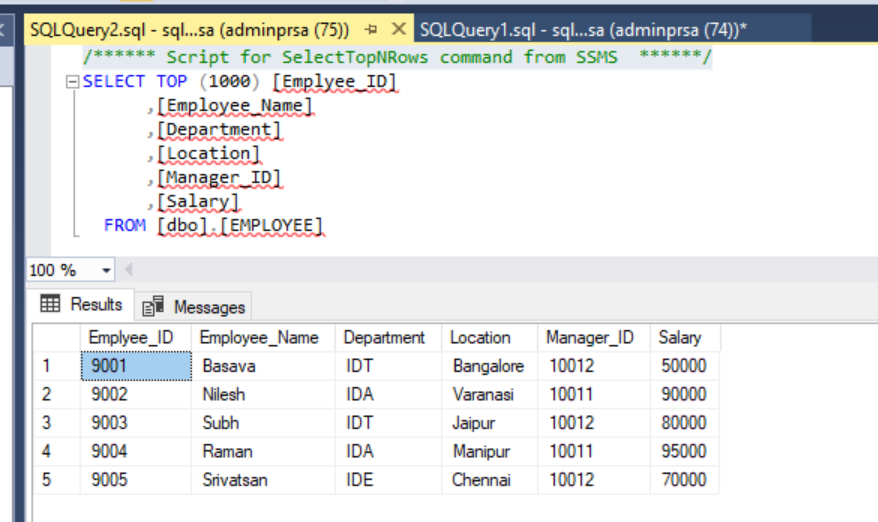
|  |
| --- |
| # Writting data from Dataframe to Azure SQL Database  jdbcHostname = "sqlserprsa.database.windows.net"  jdbcDatabase = "sqldbprsa"  jdbcPort = "1433"  username = "ZZZZZZ"  password = "XXXXXX"  jdbcUrl = "jdbc:sqlserver://{0}:{1};database={2}".format(jdbcHostname, jdbcPort, jdbcDatabase)  connectionProperties = {     "user" : username,     "password" : password,     "driver" : "com.microsoft.sqlserver.jdbc.SQLServerDriver"   }  dbtable = "dbo.EMPLOYEE"  df.write.mode("overwrite").jdbc(url=jdbcUrl, table=dbtable, properties=connectionProperties) |



**Step 5**. Check the source data

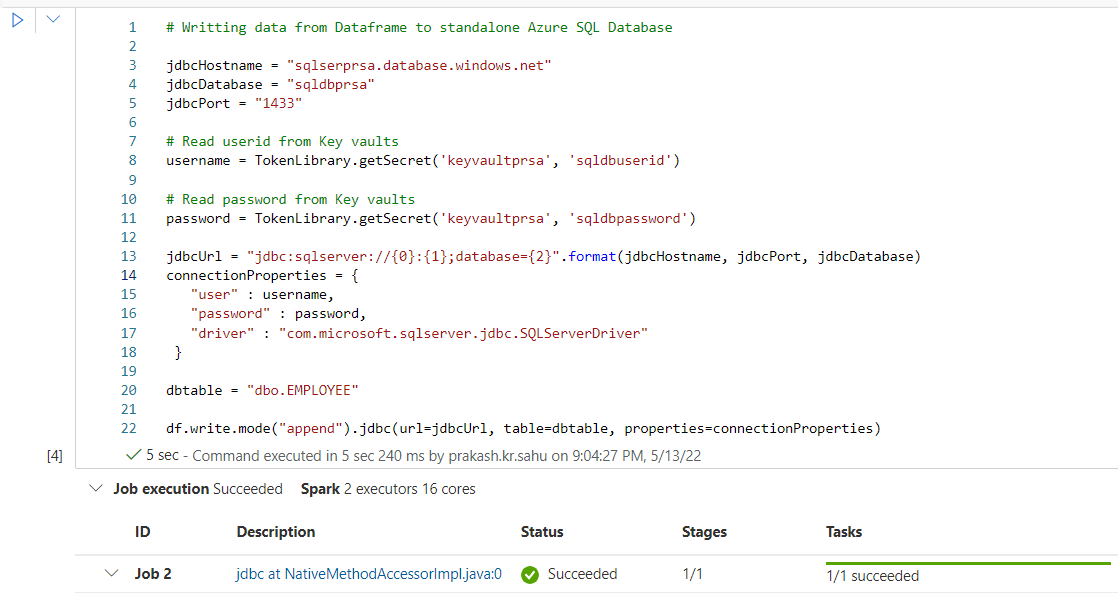


**Step 6**. Validate the data in SQL database table



**Step 7**. We can also use Azure Key Vaults to store credentials of Azure SQL server and then in Synapse spark notebook we can use TokenLibrary.getSecret method to use credentials. Code will look like as below

|  |
| --- |
| # Writting data from Dataframe to standalone Azure SQL Database  jdbcHostname = "sqlserprsa.database.windows.net"  jdbcDatabase = "sqldbprsa"  jdbcPort = "1433"  # Read userid from Key vaults  username = TokenLibrary.getSecret('keyvaultprsa', 'sqldbuserid')  # Read password from Key vaults  password = TokenLibrary.getSecret('keyvaultprsa', 'sqldbpassword')  jdbcUrl = "jdbc:sqlserver://{0}:{1};database={2}".format(jdbcHostname, jdbcPort, jdbcDatabase)  connectionProperties = {     "user" : username,     "password" : password,     "driver" : "com.microsoft.sqlserver.jdbc.SQLServerDriver"   }  dbtable = "dbo.EMPLOYEE"  df.write.mode("append").jdbc(url=jdbcUrl, table=dbtable, properties=connectionProperties) |

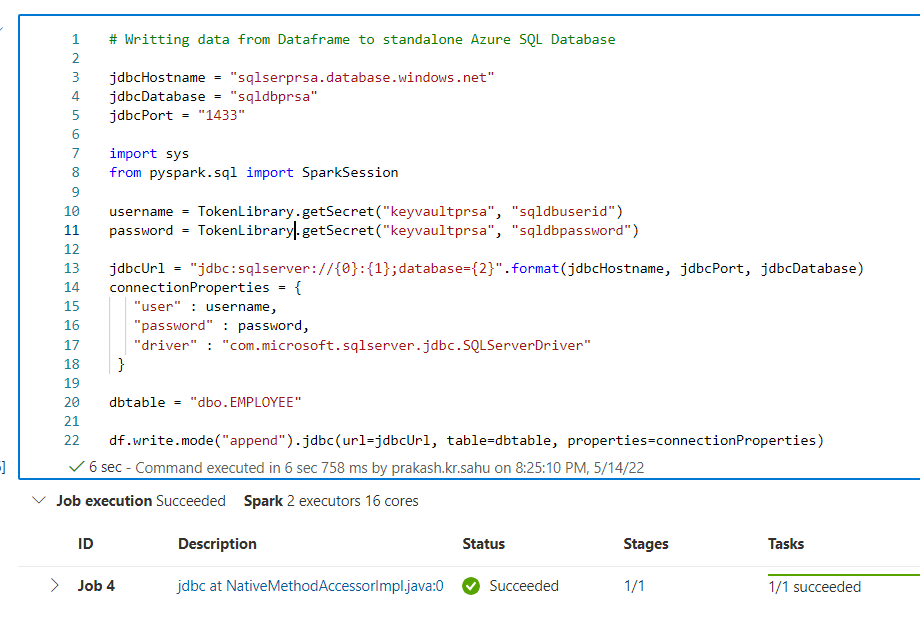


**Integrating Synapse notebook into pipeline:**

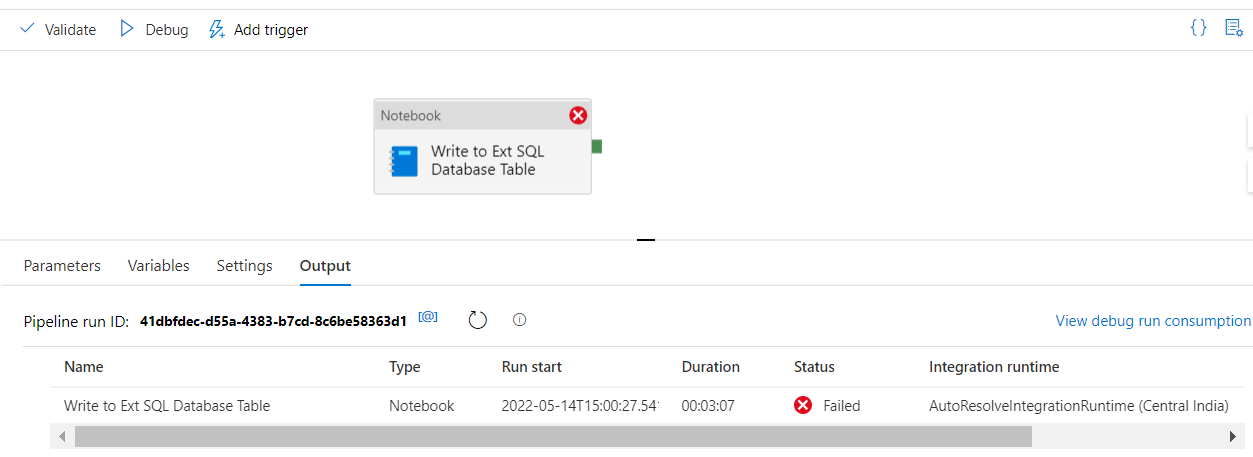
**Introduction**: Above notebook work fine when we run it independently. But if we integrate above notebook in pipeline and run the pipeline to perform same task of writing data in Azure SQL database then we get error.

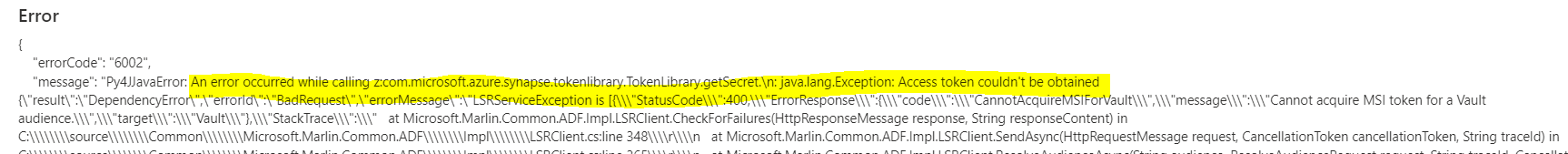
This is known Product issue. **Solution for this issue that we need to create Linked Service for Key Vault** and the use in in code in pyspark notebook and **second thing is that we need to configure security in key vault** so that it can include azure synapse and grant necessary authorization to synapse to access key vault secrets.

**Step 1**. Below code work fine independently.

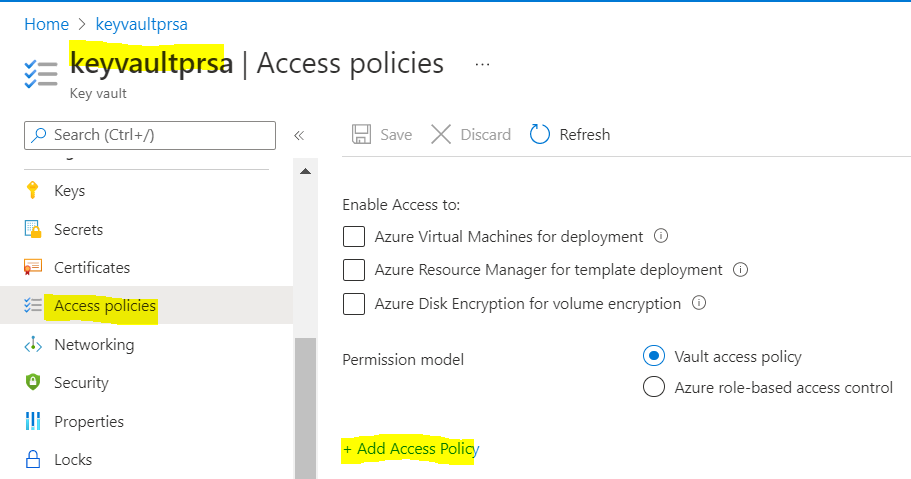


|  |
| --- |
| # Writting data from Dataframe to standalone Azure SQL Database  jdbcHostname = "sqlserprsa.database.windows.net"  jdbcDatabase = "sqldbprsa"  jdbcPort = "1433"  import sys  from pyspark.sql import SparkSession  username = TokenLibrary.getSecret("keyvaultprsa", "sqldbuserid")  password = TokenLibrary.getSecret("keyvaultprsa", "sqldbpassword")  jdbcUrl = "jdbc:sqlserver://{0}:{1};database={2}".format(jdbcHostname, jdbcPort, jdbcDatabase)  connectionProperties = {     "user" : username,     "password" : password,     "driver" : "com.microsoft.sqlserver.jdbc.SQLServerDriver"   }  dbtable = "dbo.EMPLOYEE"  df.write.mode("append").jdbc(url=jdbcUrl, table=dbtable, properties=connectionProperties) |
|  |

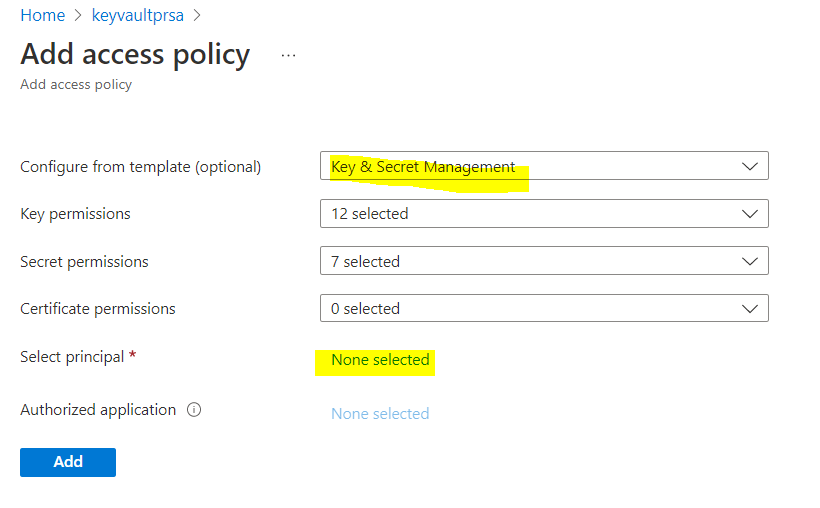
**Step 2**. We integrate above notebook code and integrate in pipeline. When we run this pipeline, it fails with some error as shown below.



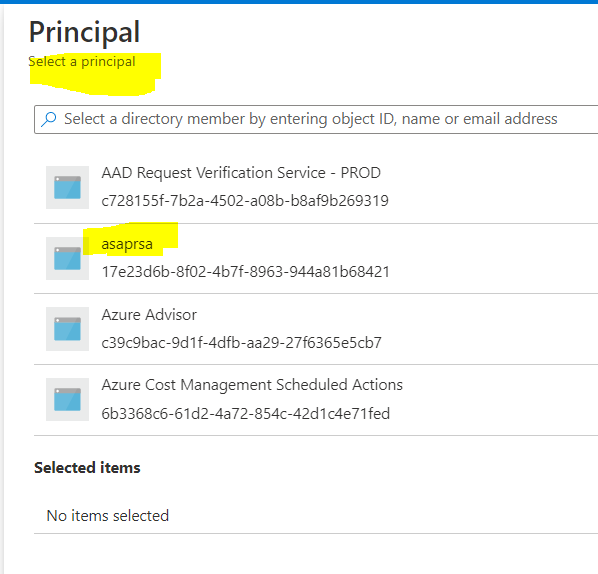
**Step 3**. To fix above issue, go to Azure Key vault and in Access Policies blade, click on Add access policy.



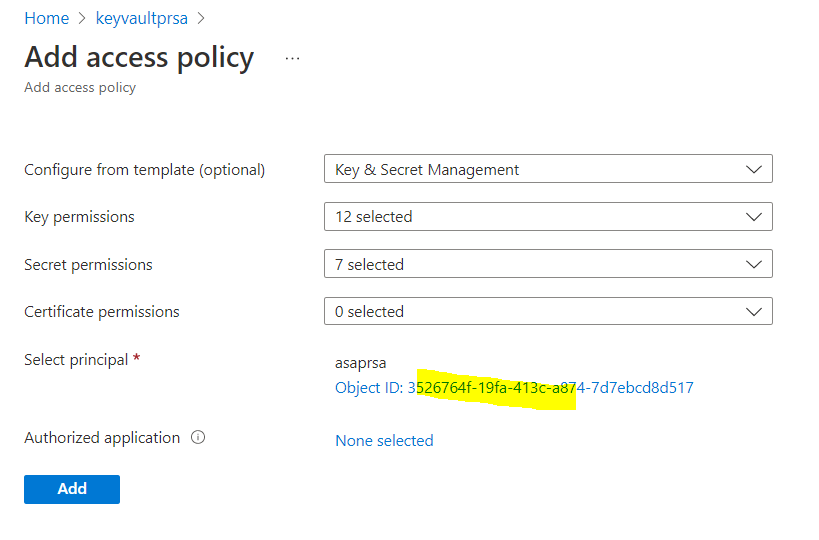
**Step 4**. Select template as below and click on “None Selected” link for select Principal.



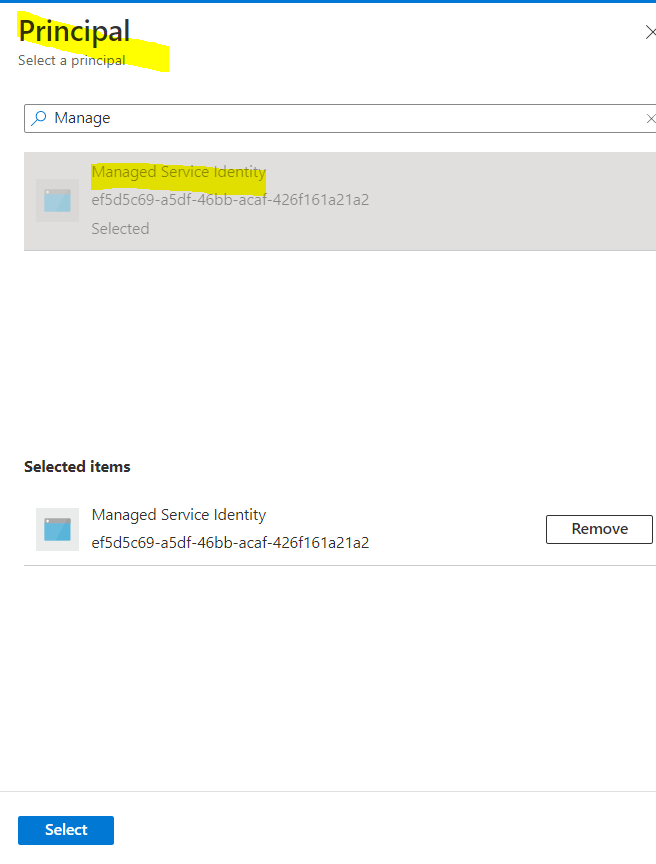
**Step 5**. Select Synapse workspace that we are using.



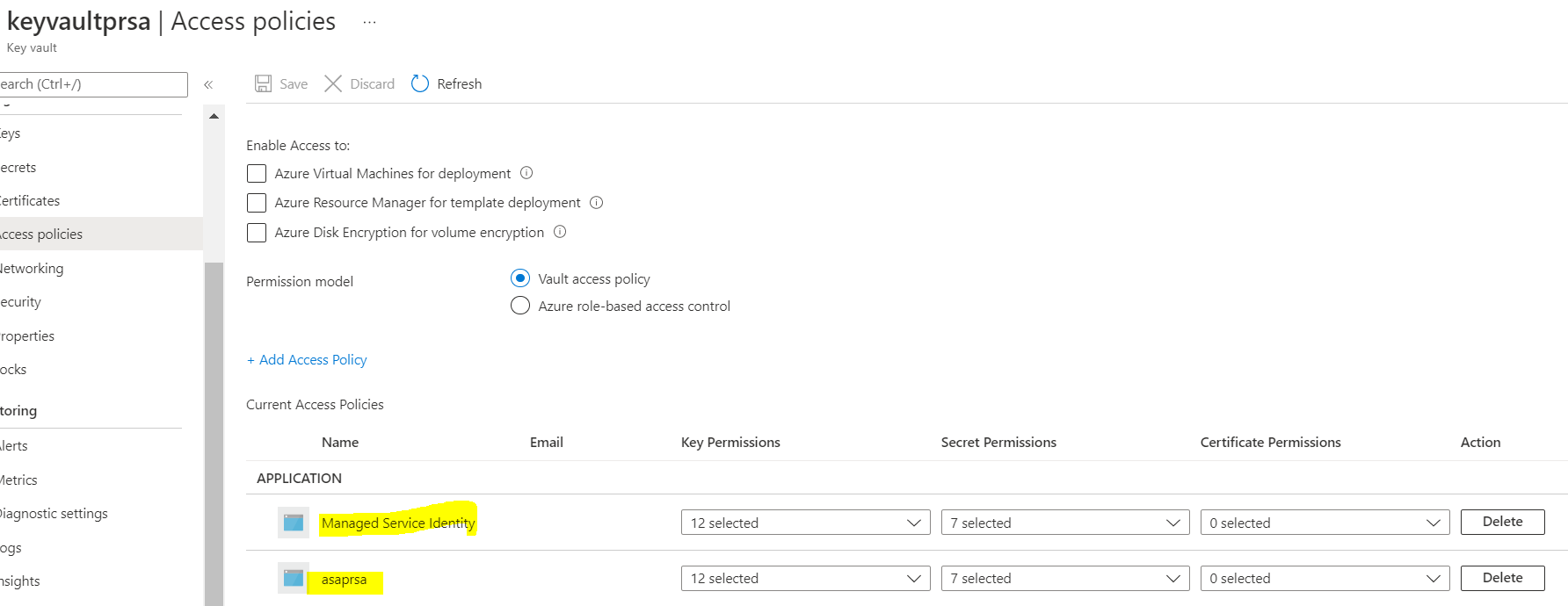
**Step 6**.**a** Click on Add to include it.



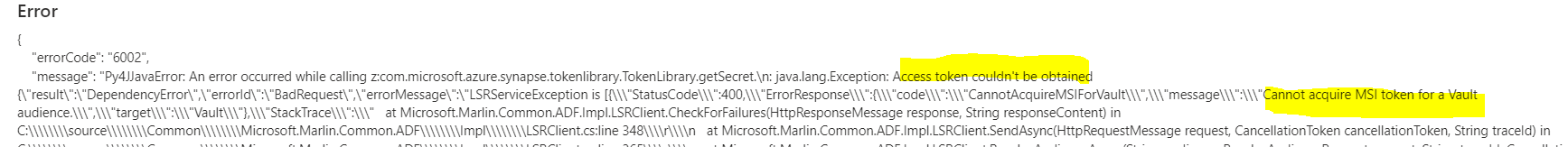
**Step 6.b** Similarly add Managed Service Identity(MSI).



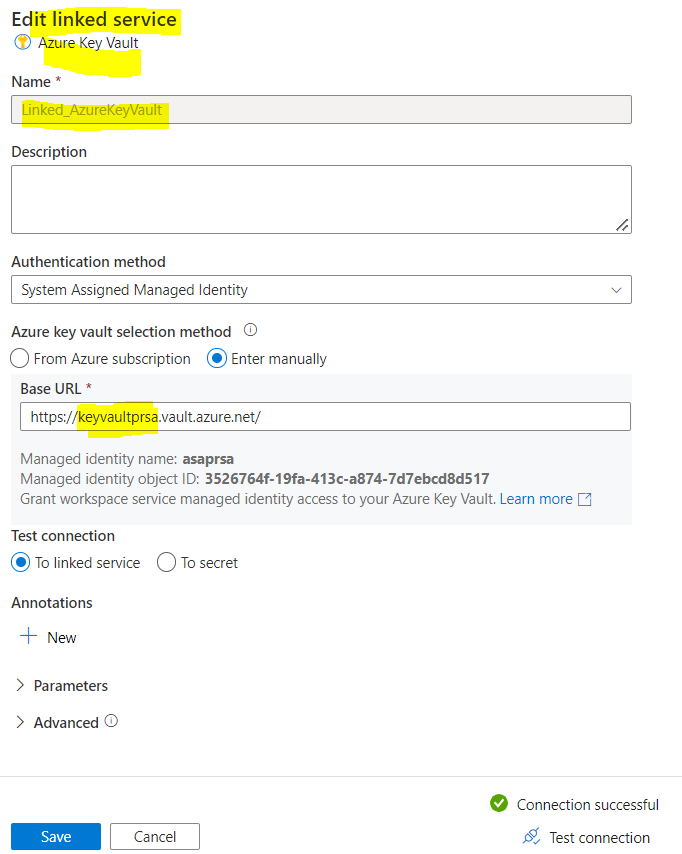
**Step 7**. Screen window will look like below. Click on Save. This way we have fixed the security issue.



**Step 8**. Again, run the pipeline. This time pipeline will fail again with below error.



**Step 9**. To fix above issue, create Linked service for key vault as shown below.



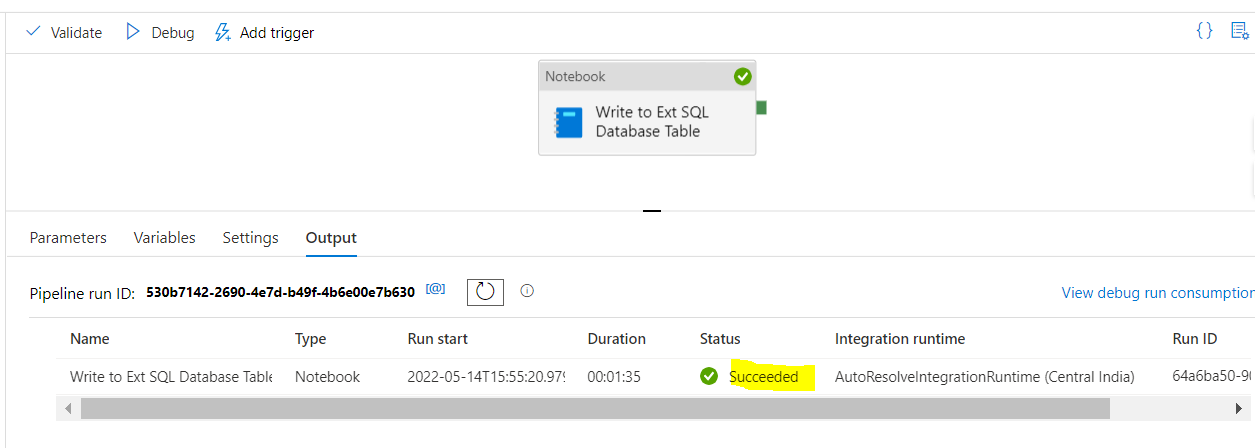
Step 10. Go to notebook code and include Linked service name for key vault in code as shown below.

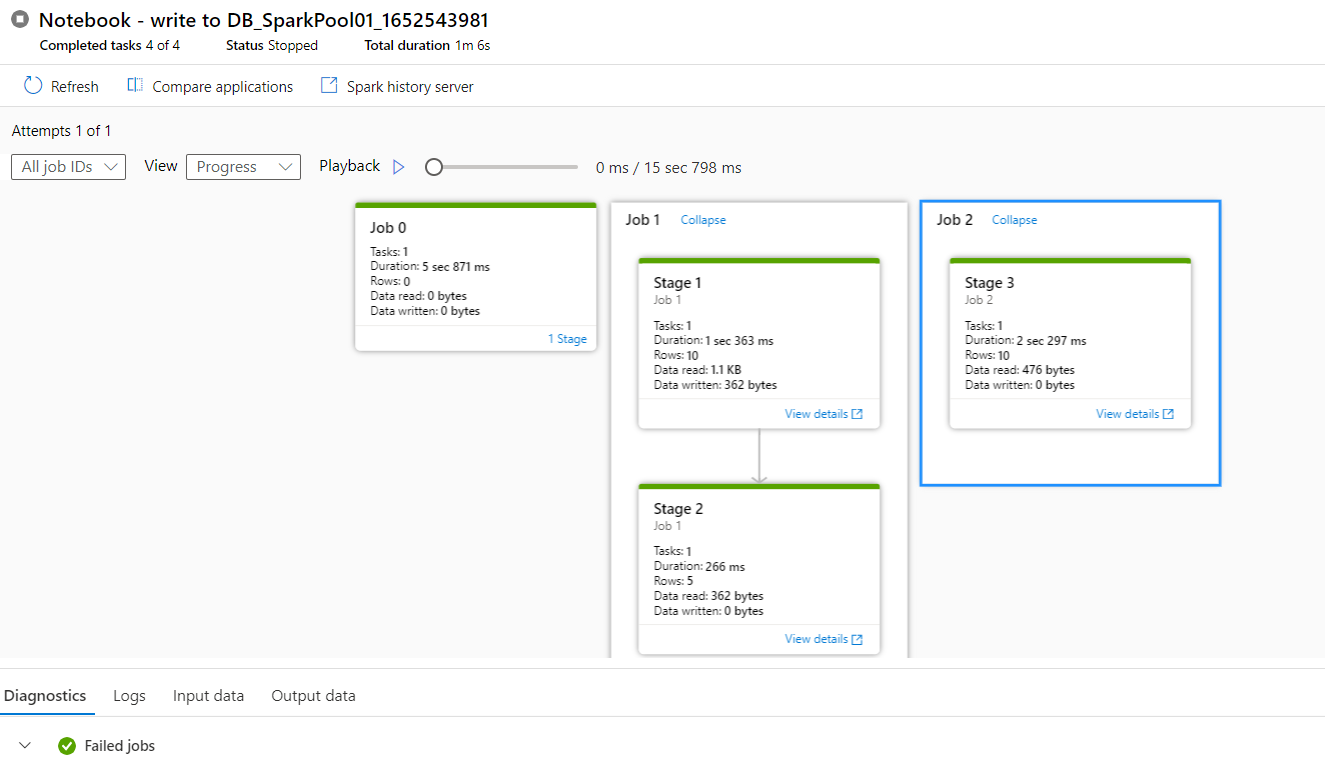
username = TokenLibrary.getSecret("keyvaultprsa", "sqldbuserid", "Linked\_AzureKeyVault")

password = TokenLibrary.getSecret("keyvaultprsa", "sqldbpassword", "Linked\_AzureKeyVault")



**Step 11**. Run the pipeline again. This time it should be successful.



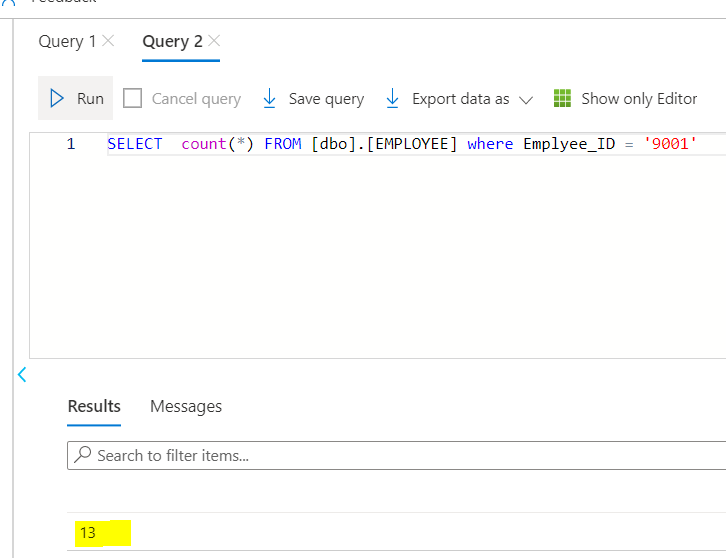


**Step 12**. Validate records in database table.

Before count



After count



Note: below notebook can also be used as alternative to above code,

|  |
| --- |
| # Writting data from Dataframe to standalone Azure SQL Database  jdbcHostname = "sqlserprsa.database.windows.net"  jdbcDatabase = "sqldbprsa"  jdbcPort = "1433"  import sys  from pyspark.sql import SparkSession  sc = SparkSession.builder.getOrCreate()  token\_library = sc.\_jvm.com.microsoft.azure.synapse.tokenlibrary.TokenLibrary  # Read userid from Key vaults  username = token\_library.getSecret("keyvaultprsa", "sqldbuserid", "Linked\_AzureKeyVault")  # Read password from Key vaults  password = token\_library.getSecret("keyvaultprsa", "sqldbpassword", "Linked\_AzureKeyVault")  jdbcUrl = "jdbc:sqlserver://{0}:{1};database={2}".format(jdbcHostname, jdbcPort, jdbcDatabase)  connectionProperties = {     "user" : username,     "password" : password,     "driver" : "com.microsoft.sqlserver.jdbc.SQLServerDriver"   }  dbtable = "dbo.EMPLOYEE"  df.write.mode("append").jdbc(url=jdbcUrl, table=dbtable, properties=connectionProperties) |