Solution Insight: SQL Server 2022 Data Analytics on Dell PowerEdge with AMD EPYC 7473X Processors and Dell ECS S3 Storage

November 2022

H19353

White Paper

Abstract

This white paper provides an insight into the benefits of a powerful, agile, and flexible infrastructure for SQL Server 2022 data analytics.

DELL Technologies

Copyright

The information in this publication is provided as is. Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Contents

Introduction

SQL Server 2022 is the newest version of SQL Server database for Microsoft. This document focuses on data analytics and data protection aspects of the features introduced in SQL Server 2022. For example, Polybase and backup and recovery of databases using S3-compatible Elastic Cloud Storage (ECS).

3rd Gen EPYC with 3D V-Cache is the new AMD EPYC processor with 3D V-Cache technology. It is the first CPU with 3D chiplet technology. This processor has three times the L3 cache than the standard 3rd Gen EPYC processors and is well suited for analytic workloads.

The goal of this white paper is to provide solution insights for data engineers, data scientists, and architects who will be running analytic workloads using SQL Server 2022 with object storage. The underlying infrastructure for this workload includes Dell PowerEdge servers with AMD EPYC 7473X processors, ECS, and PowerStore array.

Business challenge

Data virtualization has become popular among large enterprises because unstructured and semi-structured data is common and using this data is challenging. Marketing leaders across industries expect the data to be available for easy consumption to help speed up their decision-making process.

Data analytics is the process of analyzing raw, structured, and unstructured data to identify trends and answer questions. This type of analysis allows a business to interpret and communicate meaningful data patterns. There are several approaches to data analytics, including the following:

- Descriptive
- Diagnostic
- Predictive
- Prescriptive

Organizations can apply analytics to business data to describe, predict, and improve business processes and outcomes.

Organizations rely on data analytics to make better business decisions. Using the right infrastructure to run such workloads not only speeds up the analytics process, but it also provides a robust architecture to prevent any unplanned outages. Data analytics infrastructure needs to be powerful, highly available (HA), and flexible—as does any infrastructure designed for business-critical applications.

Data analytic workloads can be CPU intensive and selecting the optimal CPUs for the data analytic servers can be challenging, time consuming, and expensive. There are three common CPU components that should be considered when choosing a CPU:

- Number of cores per socket
- Frequency of the cores

- L3 cache size

Storage is another important factor for data analytic workloads. The most common storage for unstructured and semi-structured data is S3-compatible object storage. There are several object storage options available on the market, but not all options are the same. Procuring the right object storage for data analytic workloads can be complex, and time consuming.

Solution overview

Microsoft SQL Server is widely used across all industries, and these data sources are often mission critical. This means that backup and restore of SQL Server database is crucial. SQL Server 2022 backup and restore with S3-compatible object storage provides additional flexibility which can be backed up to the cloud. To use this feature, T-SQL provides the TO URL syntax for backup and FROM URL syntax for restore.

Data virtualization is a broad term used to describe an approach to data management. It allows an application to retrieve and manipulate data without requiring technical details about the data, such as how it is formatted at the source or where it is physically located. Data virtualization involves abstracting different sources through a single data access layer. There are tools and software available that organizations are adopting to integrate different types of data virtually. Data integration enables data mining and data analytics, and it is critical for predictive analytics tools that use machine learning (ML) and artificial intelligence (AI).

SQL Server 2022 PolyBase makes data virtualization possible for data scientists to use T-SQL for analytic workloads. PolyBase does this by querying data directly from other sources such as Oracle, Teradata, Hadoop cluster, and S3-compatible object storage without separately installing client connection software. It allows T-SQL queries to join the data from external sources to relational tables in an instance of SQL Server. The use of T-SQL OPENROWSET or EXTERNAL TABLE syntax delivers a powerful tool to query data in S3-compatible storage.

In this validation, AMD EPYC 7473X processors were chosen for the SQL Server 2022 database instances because running T-SQL queries for data analytics require quick response time. The AMD EPYC 7473X processors have 24-core per socket @2.8GHz with L3 cache size of 768 MB which offers enough horsepower for data analytic workloads.

Data analytic workloads require optimal CPUs for data processing. It is also important for these workloads to have a flexible and scalable S3-compatible object storage.

Dell Elastic Cloud Storage (ECS) is a software-defined, cloud-scale, object storage platform that delivers S3, Atmos, CAS, Swift, NFSv3, and HDFS storage services on a single, modern platform. It provides simple RESTful API access for storage services. Dell ECS provides significant value for organizations seeking a platform that supports rapid data growth. Dell ECS advantages and features include:

Table 1. Dell ECS features

Cloud scale	• Globally distributed object infrastructure • Exabyte+ scale without limits on storage pool, cluster, or federated environment capacity • No limits exist on the number of objects in a system, namespace, or bucket • Efficient at both small and large file workloads with no limits to object size
Flexible deployment	• Appliance deployment • Software-only deployment with support for certified or custom industry standard hardware • Multiprotocol support: Object (S3, Swift, Atmos, CAS) and File (HDFS, NFSv3) • Multiple workloads: Modern apps and traditional apps • Secondary storage for Data Domain Cloud Tier and Isilon using CloudPools • Non-disruptive upgrade paths to current generation ECS models
Enterprise grade	• Data-at-rest (D@RE) with key rotation and external key management • Encrypted inter-site communication • Reporting, policy and event-based record retention and platform hardening for SEC Rule 17a-4(f) compliance including advanced retention management such as litigation hold and min-max governance • Compliance with Defense Information System Agency (DISA) Security Technical Implementation Guide (STIG) hardening guidelines • Authentication, authorization, and access controls with Active Directory and LDAP • Integration with monitoring and alerting infrastructure (SNMP traps and SYSLOG) • Enhanced enterprise capabilities (multi-tenancy, capacity monitoring and alerting)
TCO reduction	• Global namespace • Small and large file performance • Seamless Centera migration • Fully compliant with Atmos REST • Low management overhead • Small data center footprint • High storage utilization

Solution physical architecture

To understand the architectural flexibility of SQL Server 2022 analytic workloads using T-SQL, Dell solutions engineers validated two options:

1. SQL Server 2022 on VMware virtualization
2. SQL Server 2022 on Red Hat OpenShift

To achieve this goal, the engineering team performed two tests with the following hardware components:

- Dell PowerEdge R7525 with AMD EPYC 7473X processors
- Dell Elastic Cloud Storage (ECS) EX300 cluster
- Dell PowerStore 9200T storage array
- Dell PowerSwitch S5224F-ON 1 GbE and 25 GbE
- Dell Connectrix DS6620 Fibre Channel switch

Figure 1 provides an overview of the two physical architectural options that were used in this exercise.

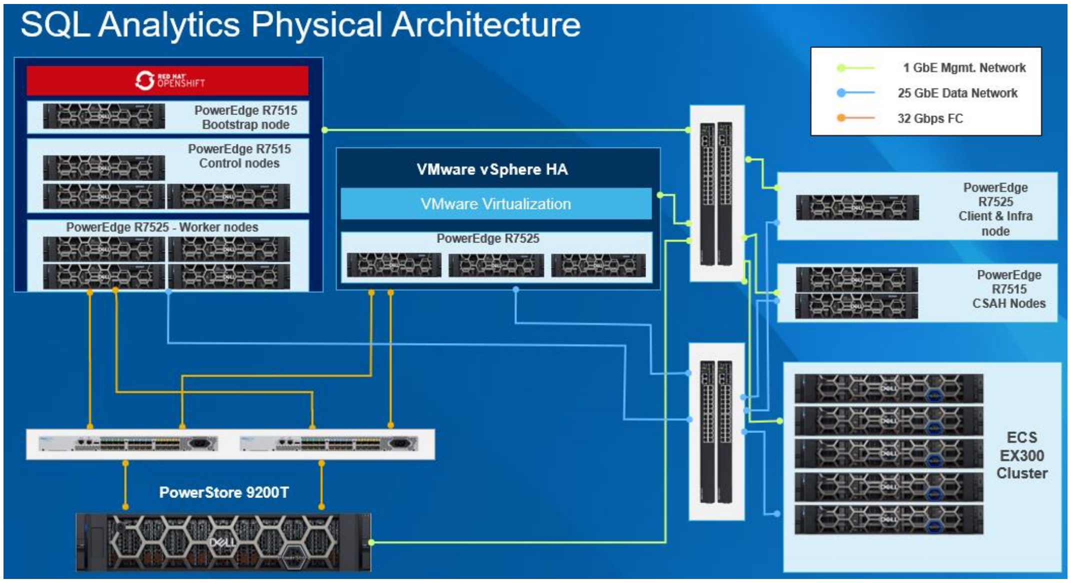
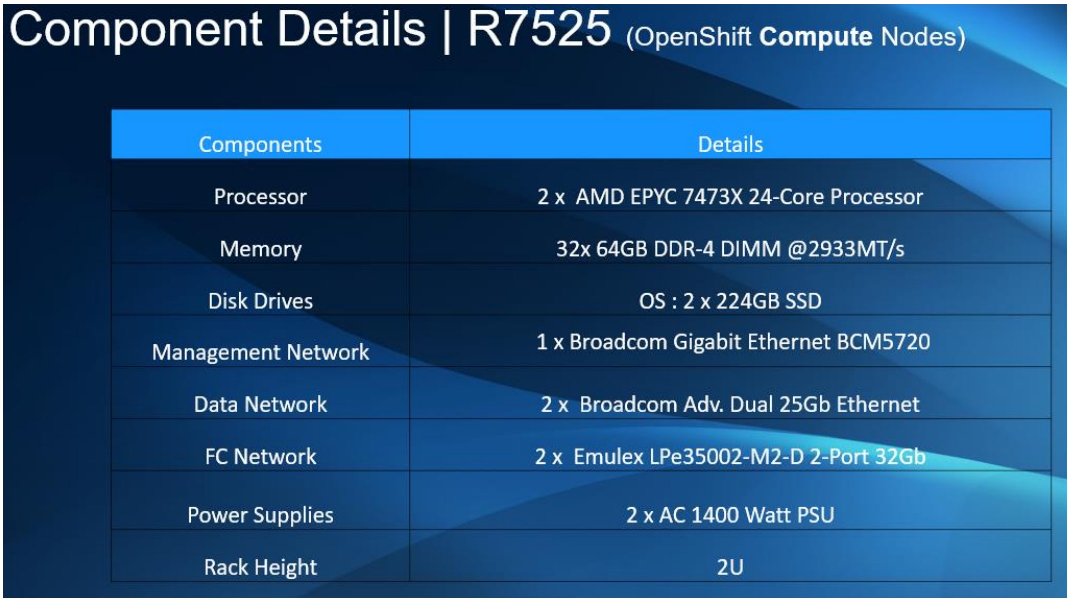

SQL Analytics Physical Architecture
RED HAT OPENSHIFT
PowerEdge R7515 Bootstrap node
PowerEdge R7515 Control nodes
PowerEdge R7525 - Worker nodes
VMware vSphere HA
VMware Virtualization
PowerEdge R7525
1 GbE Mgmt. Network
25 GbE Data Network
32 Gbps FC
PowerEdge R7525 Client & Infra node
PowerEdge R7515 CSAH Nodes
ECS EX300 Cluster
PowerStore 9200T

Figure 1. Physical architecture

AMD EPYC 7473X processors were used for the OpenShift compute nodes to deliver quick results for running analytic workloads.

Figure 2 shows the component details of the PowerEdge R7525 that was used in the OpenShift architecture for the compute nodes.



Component Details | R7525 (OpenShift **Compute** Nodes)

Components	Details
Processor	2 x AMD EPYC 7473X 24-Core Processor
Memory	32x 64GB DDR-4 DIMM @2933MT/s
Disk Drives	OS : 2 x 224GB SSD
Management Network	1 x Broadcom Gigabit Ethernet BCM5720
Data Network	2 x Broadcom Adv. Dual 25Gb Ethernet
FC Network	2 x Emulex LPe35002-M2-D 2-Port 32Gb
Power Supplies	2 x AC 1400 Watt PSU
Rack Height	2U

Figure 2. R7525 component details

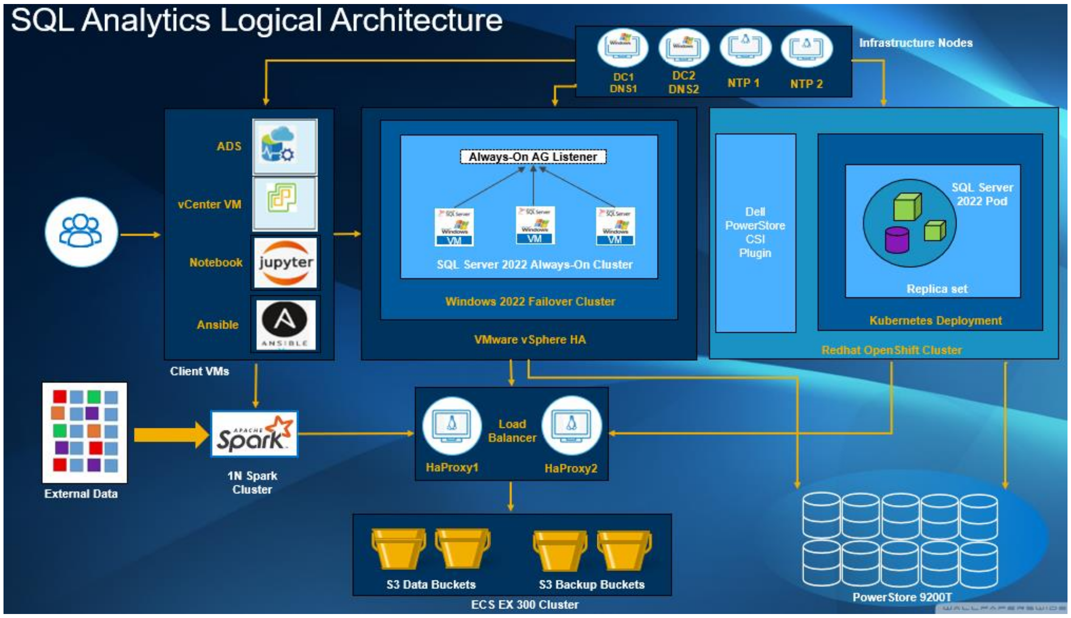
Solution logical architecture

SQL Server 2022 can be run on Windows operating system or on Linux operating system; Dell solutions engineers tested both options.

The first test included SQL Server 2022 running on Windows operating system with VMware virtual machines using vVols. The second test was to setup SQL Server 2022 on Linux container using Red Hat OpenShift as the container orchestration.

In the VMware setup, there were three SQL Server 2022 instances running with Always-On configuration. Availability Group was also configured for the three SQL instances. Windows 2022 Failover clustering and VMware vSphere HA were also configured. This setup provides the highest availability for the SQL Server 2022 instances in a virtualized environment.

In the OpenShift setup, there was one pod configured for SQL Server 2022 with persistent volume using the Dell PowerStore CSI plug-in. This setup enables the ease of management of the SQL Server 2022 pods in OpenShift.



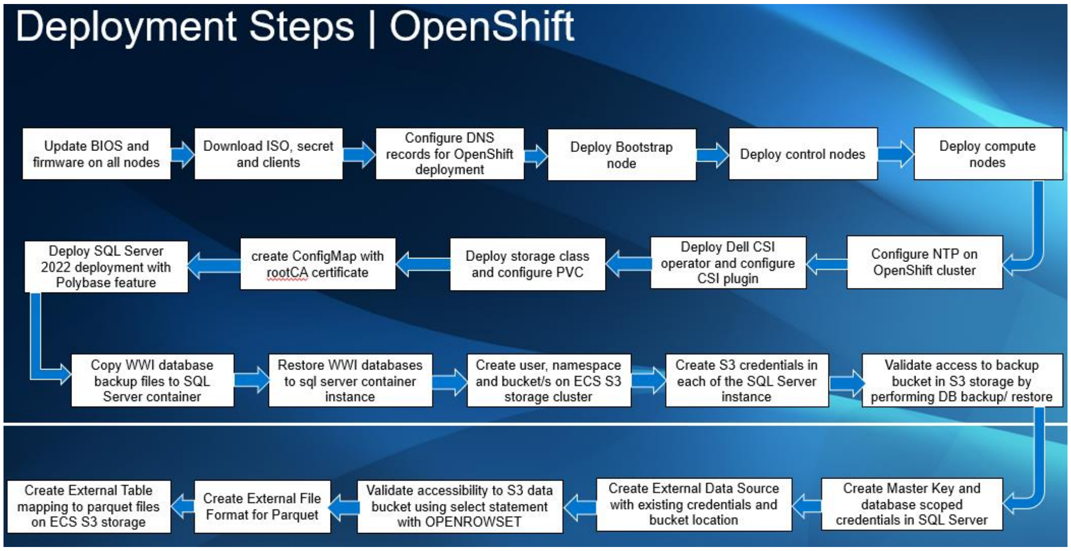
SQL Analytics Logical Architecture

Figure 3. Logical architecture overview

Deployment steps

Dell solutions engineering carefully set up and fine-tuned the data analytics process with SQL Server 2022 and Dell ECS. The figures in this section show the deployment steps with OpenShift and VMware.

Figure 4 outlines the deployment steps for the OpenShift setup.



Deployment Steps | OpenShift

Update BIOS and firmware on all nodes

Download ISO, secret and clients

Configure DNS records for OpenShift deployment

Deploy Bootstrap node

Deploy control nodes

Deploy compute nodes

Configure NTP on OpenShift cluster

Deploy Dell CSI operator and configure CSI plugin

Deploy storage class and configure PVC

create ConfigMap with rootCA certificate

Deploy SQL Server 2022 deployment with Polybase feature

Copy WWI database backup files to SQL Server container

Restore WWI databases to sql server container instance

Create user, namespace and bucket/s on ECS S3 storage cluster

Create S3 credentials in each of the SQL Server instance

Validate access to backup bucket in S3 storage by performing DB backup/ restore

Create Master Key and database scoped credentials in SQL Server

Create External Data Source with existing credentials and bucket location

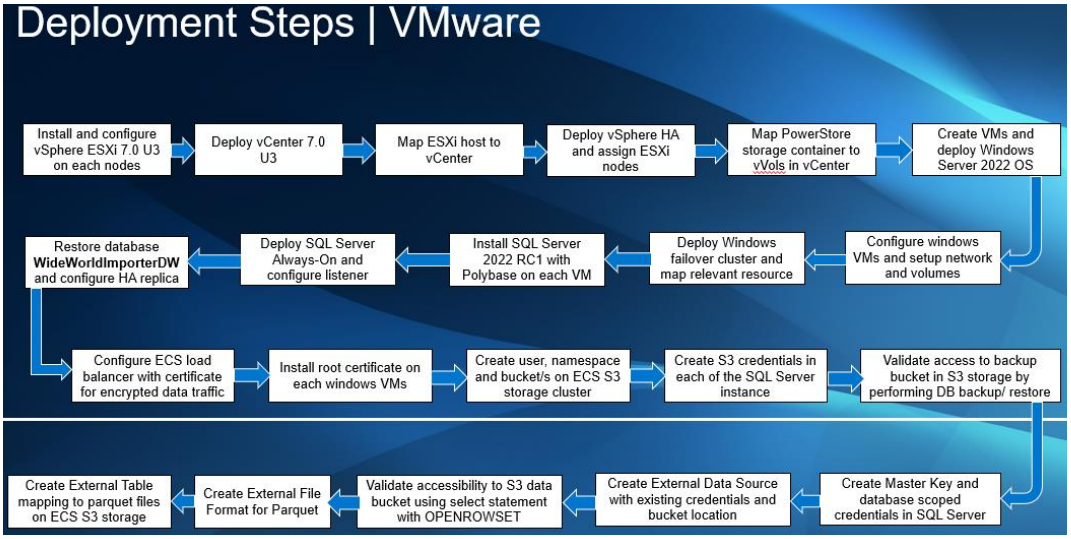
Validate accessibility to S3 data bucket using select statement with OPENROWSET

Create External File Format for Parquet

Create External Table mapping to parquet files on ECS S3 storage

Figure 4. OpenShift deployment steps

Figure 5 outlines the deployment steps with VMware option.

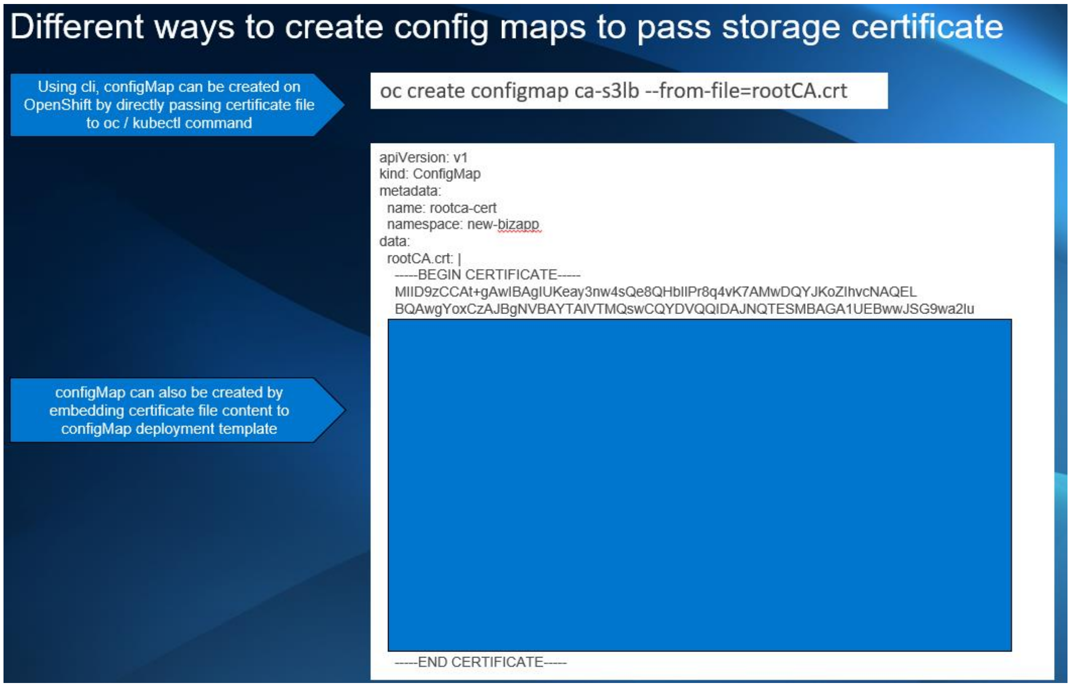


Deployment Steps | VMware

Install and configure vSphere ESXi 7.0 U3 on each nodes
Deploy vCenter 7.0 U3
Map ESXi host to vCenter
Deploy vSphere HA and assign ESXi nodes
Map PowerStore storage container to vVols in vCenter
Create VMs and deploy Windows Server 2022 OS
Configure windows VMs and setup network and volumes
Deploy Windows failover cluster and map relevant resource
Install SQL Server 2022 RC1 with Polybase on each VM
Deploy SQL Server Always-On and configure listener
Restore database **WideWorldImporterDW** and configure HA replica
Configure ECS load balancer with certificate for encrypted data traffic
Install root certificate on each windows VMs
Create user, namespace and bucket/s on ECS S3 storage cluster
Create S3 credentials in each of the SQL Server instance
Validate access to backup bucket in S3 storage by performing DB backup/ restore
Create Master Key and database scoped credentials in SQL Server
Create External Data Source with existing credentials and bucket location
Validate accessibility to S3 data bucket using select statement with OPENROWSET
Create External File Format for Parquet
Create External Table mapping to parquet files on ECS S3 storage

Figure 5. VMware deployment steps

One requirement for deploying SQL Server 2022 is to have a root certificate created for the TLS communication between the SQL Server instance and S3-compatible object storage. When running SQL Server container image on OpenShift, one way to pass the certificate is by using a ConfigMap. For more information about ConfigMap, see the Kubernetes website.



Different ways to create config maps to pass storage certificate

Using cli, configMap can be created on OpenShift by directly passing certificate file to oc / kubectl command

```
oc create configmap ca-s3lb --from-file=rootCA.crt
```

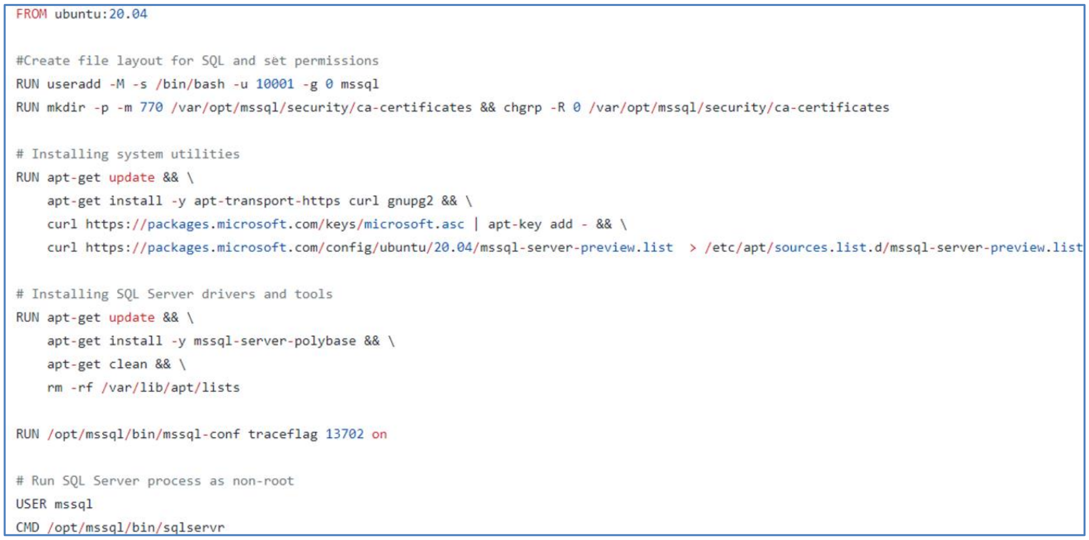
configMap can also be created by embedding certificate file content to configMap deployment template

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: rootca-cert
  namespace: new-bizapp
data:
  rootCA.crt: |
    -----BEGIN CERTIFICATE-----
    MIID9zCCAt+gAwIBAgIUKeay3nw4sQe8QHbIIPr8q4vK7AMwDQYJKoZIhvcNAQEL
    BQAwgYoxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNQTESMBAGA1UEBwwJSG9wa2lu
    [redacted]
    -----END CERTIFICATE-----
```

Figure 6. ConfigMap file

A new container image was created for SQL Server with Polybase installed because Polybase is not enabled by default in the published container image mcr.microsoft.com/mssql/server:2022-latest.

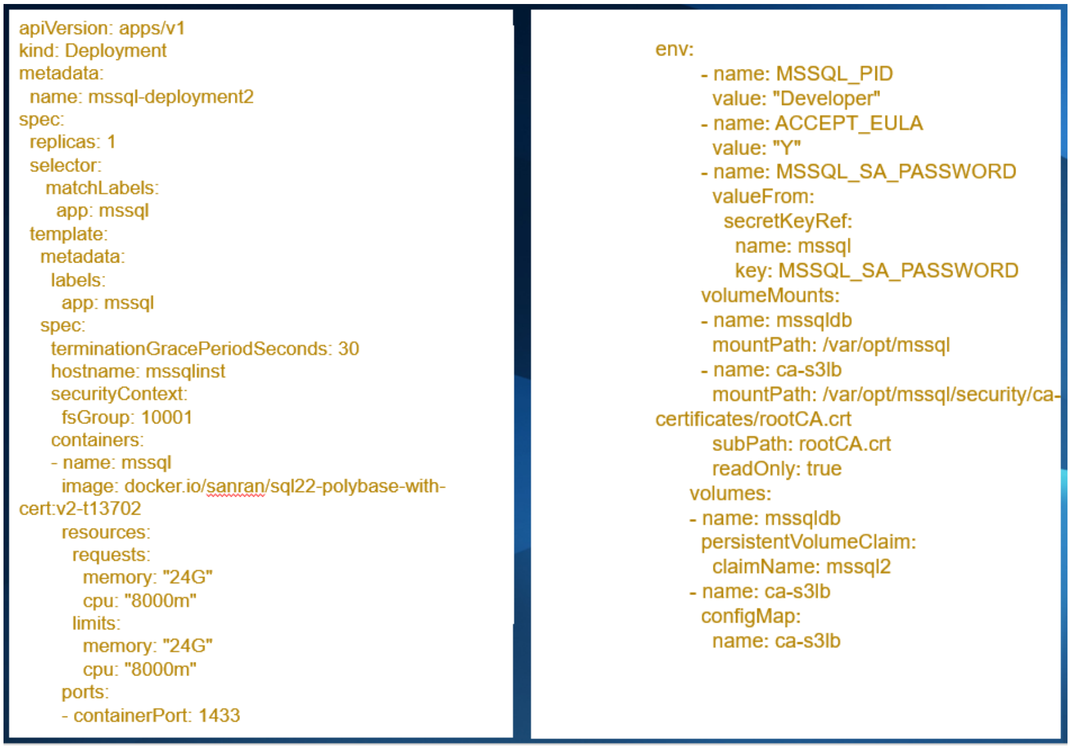
The following screenshot shows an example of the code for building a custom SQL Server 2022 container image with Polybase installed.



```
FROM ubuntu:20.04

#Create file layout for SQL and set permissions
RUN useradd -M -s /bin/bash -u 10001 -g 0 mssql
RUN mkdir -p -m 770 /var/opt/mssql/security/ca-certificates && chgrp -R 0 /var/opt/mssql/security/ca-certificates

# Installing system utilities
RUN apt-get update && \
    apt-get install -y apt-transport-https curl gnupg2 && \
    curl https://packages.microsoft.com/keys/microsoft.asc | apt-key add - && \
    curl https://packages.microsoft.com/config/ubuntu/20.04/mssql-server-preview.list  > /etc/apt/sources.list.d/mssql-server-preview.list

# Installing SQL Server drivers and tools
RUN apt-get update && \
    apt-get install -y mssql-server-polybase && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists

RUN /opt/mssql/bin/mssql-conf traceflag 13702 on

# Run SQL Server process as non-root
USER mssql
CMD /opt/mssql/bin/sqlservr
```

Figure 7. Sample code for building SQL Server 2022 container image

Figure 8 provides an example of a YAML file that can be used to deploy the custom SQL Server container image with persistent volume on PowerStore and Dell ECS storage certificate using ConfigMap.



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mssql-deployment2
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mssql
  template:
    metadata:
      labels:
        app: mssql
    spec:
      terminationGracePeriodSeconds: 30
      hostname: mssqlinst
      securityContext:
        fsGroup: 10001
      containers:
      - name: mssql
        image: docker.io/sanran/sql22-polybase-with-
cert:v2-t13702
        resources:
          requests:
            memory: "24G"
            cpu: "8000m"
          limits:
            memory: "24G"
            cpu: "8000m"
        ports:
        - containerPort: 1433
```

```
        env:
            - name: MSSQL_PID
              value: "Developer"
            - name: ACCEPT_EULA
              value: "Y"
            - name: MSSQL_SA_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mssql
                  key: MSSQL_SA_PASSWORD
            volumeMounts:
            - name: mssqldb
              mountPath: /var/opt/mssql
            - name: ca-s3lb
              mountPath: /var/opt/mssql/security/ca-
        certificates/rootCA.crt
              subPath: rootCA.crt
              readOnly: true
           volumes:
           - name: mssqldb
            persistentVolumeClaim:
              claimName: mssql2
           - name: ca-s3lb
            configMap:
              name: ca-s3lb
```

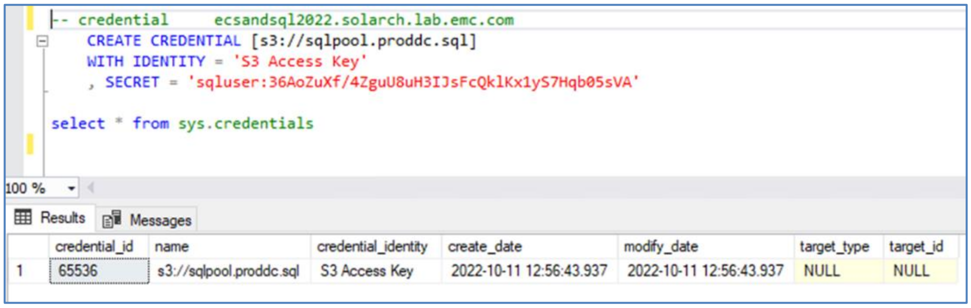
Figure 8. Sample YAML file

Backup and restore use case

SQL Server 2022 can run backup and restore operations using T-SQL. This feature allows database administrators (DBAs) to send backups of SQL databases to S3-compatible storage in addition to the traditional backup method. With the scalability of Dell ECS object storage, DBAs do not have to worry about running out of storage capacity.

A Dell ECS credential is required to backup SQL Server databases to Dell ECS on the SQL Server instance. This credential permits the S3 connection between the SQL Server instance and the object storage. After creating the credential, run the backup operation using T-SQL command with the TO URL syntax.

Create credential inside SQL Server instance for connection to S3 storage with storage URL and user credentials.

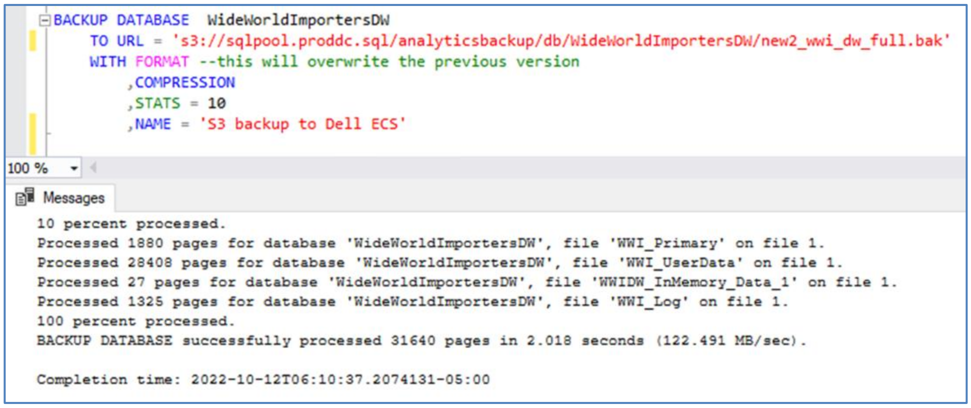


```
-- credential      ecsandsql2022.solarch.lab.emc.com
    CREATE CREDENTIAL [s3://sqlpool.proddc.sql]
    WITH IDENTITY = 'S3 Access Key'
    , SECRET = 'sqluser:36AoZuXf/4ZguU8uH3IJsFcQklKx1yS7Hqb05sVA'

select * from sys.credentials
```

	credential_id	name	credential_identity	create_date	modify_date	target_type	target_id
1	65536	s3://sqlpool.proddc.sql	S3 Access Key	2022-10-11 12:56:43.937	2022-10-11 12:56:43.937	NULL	NULL

Figure 9. Create ECS storage credential at SQL Server instance

Perform a backup of a database to ECS storage.



```
BACKUP DATABASE  WideWorldImportersDW
    TO URL = 's3://sqlpool.proddc.sql/analyticsbackup/db/WideWorldImportersDW/new2_wwi_dw_full.bak'
    WITH FORMAT --this will overwrite the previous version
        ,COMPRESSION
        ,STATS = 10
        ,NAME = 'S3 backup to Dell ECS'
```

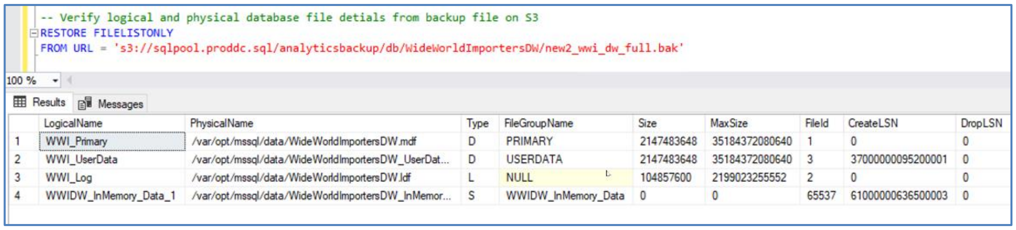
```
10 percent processed.
Processed 1880 pages for database 'WideWorldImportersDW', file 'WWI_Primary' on file 1.
Processed 28408 pages for database 'WideWorldImportersDW', file 'WWI_UserData' on file 1.
Processed 27 pages for database 'WideWorldImportersDW', file 'WWIDW_InMemory_Data_1' on file 1.
Processed 1325 pages for database 'WideWorldImportersDW', file 'WWI_Log' on file 1.
100 percent processed.
BACKUP DATABASE successfully processed 31640 pages in 2.018 seconds (122.491 MB/sec).

Completion time: 2022-10-12T06:10:37.2074131-05:00
```

Figure 10. Perform backup operation

For a restore operation, use the FROM URL syntax. This syntax can also be used to verify whether the backup files are available on the object storage before doing the actual restoration. To verify, run the “RESTORE FILELISTONLY FROM URL = ‘s3://path/filename’” command.

To perform a restore operation, run the RESTORE DATABASE <DBname> FROM URL = ‘s3://path/filename’ script with other operations.

Figure 11 shows an example for performing the verification process before the restore operation.

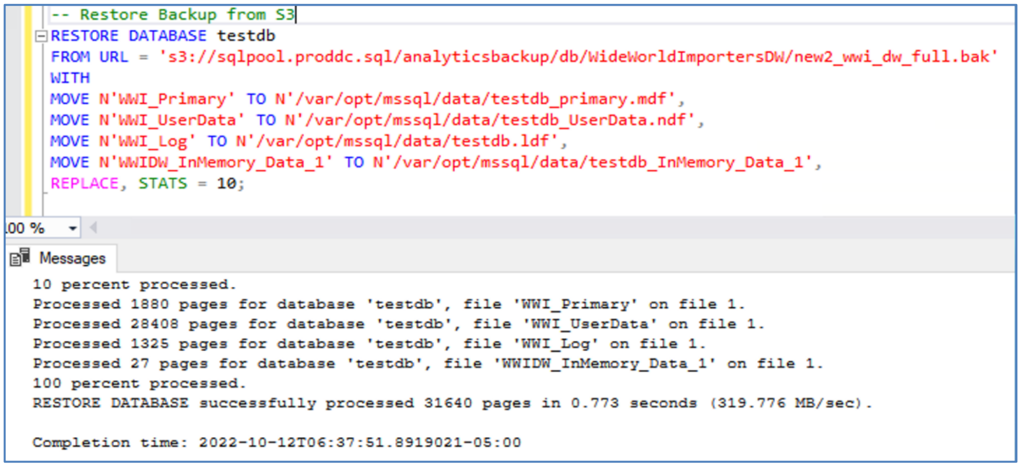


```
-- Verify logical and physical database file detials from backup file on S3
RESTORE FILELISTONLY
FROM URL = 's3://sqlpool.proddc.sql/analyticsbackup/db/WideWorldImportersDW/new2_wwi_dw_full.bak'
```

	LogicalName	PhysicalName	Type	FileGroupName	Size	MaxSize	FileId	CreateLSN	DropLSN
1	WWI_Primary	/var/opt/mssql/data/WideWorldImportersDW.mdf	D	PRIMARY	2147483648	35184372080640	1	0	0
2	WWI_UserData	/var/opt/mssql/data/WideWorldImportersDW_UserDat...	D	USERDATA	2147483648	35184372080640	3	37000000095200001	0
3	WWI_Log	/var/opt/mssql/data/WideWorldImportersDW.ldf	L	NULL	104857600	2199023255552	2	0	0
4	WWIDW_InMemory_Data_1	/var/opt/mssql/data/WideWorldImportersDW_InMemor...	S	WWIDW_InMemory_Data	0	0	65537	61000000636500003	0

Figure 11. Backup file verification

Perform SQL Server database restore where FROM URL points to storage location for backup file. All other options should not be changed.



```
-- Restore Backup from S3
RESTORE DATABASE testdb
FROM URL = 's3://sqlpool.proddc.sql/analyticsbackup/db/WideWorldImportersDW/new2_wwi_dw_full.bak'
WITH
MOVE N'WWI_Primary' TO N'/var/opt/mssql/data/testdb_primary.mdf',
MOVE N'WWI_UserData' TO N'/var/opt/mssql/data/testdb_UserData.ndf',
MOVE N'WWI_Log' TO N'/var/opt/mssql/data/testdb.ldf',
MOVE N'WWIDW_InMemory_Data_1' TO N'/var/opt/mssql/data/testdb_InMemory_Data_1',
REPLACE, STATS = 10;
```

```
10 percent processed.
Processed 1880 pages for database 'testdb', file 'WWI_Primary' on file 1.
Processed 28408 pages for database 'testdb', file 'WWI_UserData' on file 1.
Processed 1325 pages for database 'testdb', file 'WWI_Log' on file 1.
Processed 27 pages for database 'testdb', file 'WWIDW_InMemory_Data_1' on file 1.
100 percent processed.
RESTORE DATABASE successfully processed 31640 pages in 0.773 seconds (319.776 MB/sec).

Completion time: 2022-10-12T06:37:51.8919021-05:00
```

Figure 12. Restore database from ECS Storage

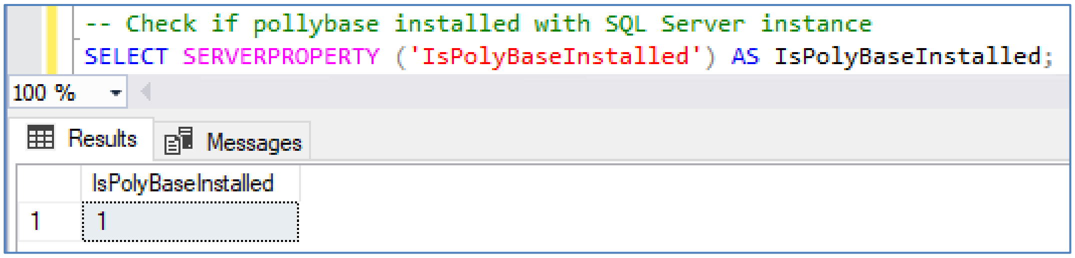
More information about backup and restore using T-SQL, see these documents from Microsoft.

Data virtualization use case

Data virtualization allows for retrieval and manipulation of data without knowing where the data is stored or how it is formatted. This concept integrates data from disparate sources without copying or moving the data, giving data scientists a single virtual layer that spans multiple formats and physical locations.

SQL Server 2022 PolyBase makes data virtualization possible by enabling a SQL Server instance to query data with T-SQL directly from SQL Server or other sources. For this feature to work properly, PolyBase must be installed and enabled on the SQL Server instance.

Figure 13 provides an example of how to verify whether PolyBase is installed and enabled.

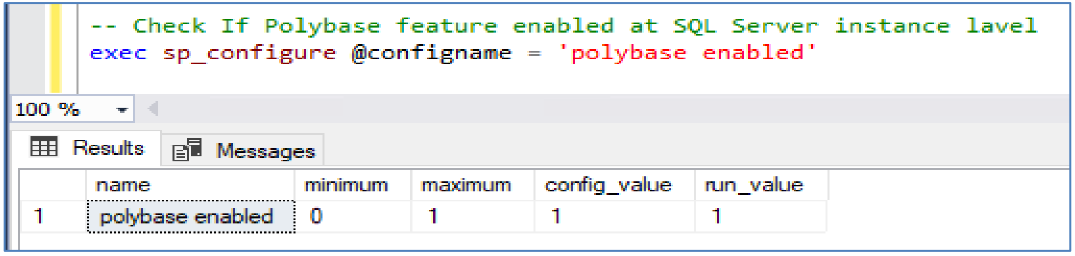


```
-- Check if pollybase installed with SQL Server instance
SELECT SERVERPROPERTY ('IsPolyBaseInstalled') AS IsPolyBaseInstalled;
```

	IsPolyBaseInstalled
1	1

Figure 13. Verify if Polybase feature is enabled at SQL Server instance level

Enable Polybase feature at SQL Server instance level if it is not already enabled.

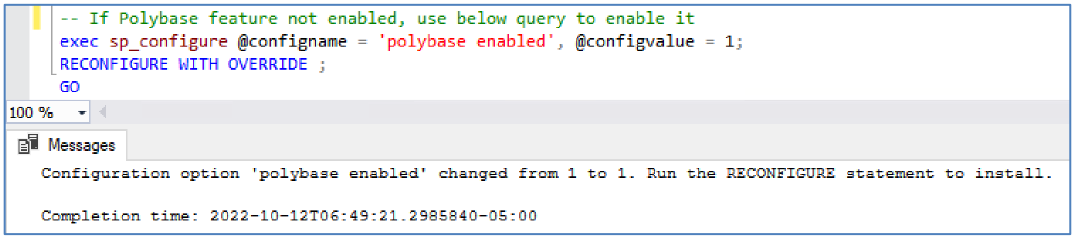


```
-- Check If Polybase feature enabled at SQL Server instance lavel
exec sp_configure @configname = 'polybase enabled'
```

	name	minimum	maximum	config_value	run_value
1	polybase enabled	0	1	1	1

Figure 14. Enable Polybase feature

Verify that PolyBase was enabled successfully.



```
-- If Polybase feature not enabled, use below query to enable it
exec sp_configure @configname = 'polybase enabled', @configvalue = 1;
RECONFIGURE WITH OVERRIDE ;
GO
```

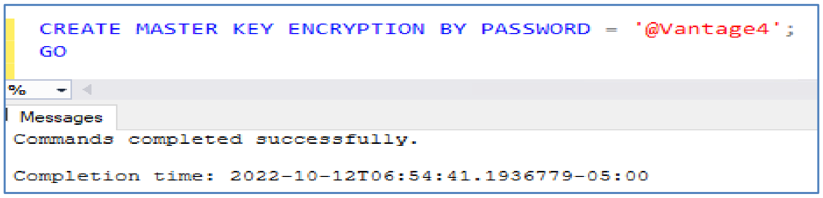
```
Configuration option 'polybase enabled' changed from 1 to 1. Run the RECONFIGURE statement to install.

Completion time: 2022-10-12T06:49:21.2985840-05:00
```

Figure 15. PolyBase verification

An external data source should be created after PolyBase has been installed and enabled. In this exercise, the external source was created on a Dell ECS object storage. An encryption key is required for the communication between the SQL Server instance and the external data source.

The following figure shows an example of how to create an encryption key and verify the communication between SQL Server and the Dell ECS.

1. Create Encryption key with password in the desired user database.

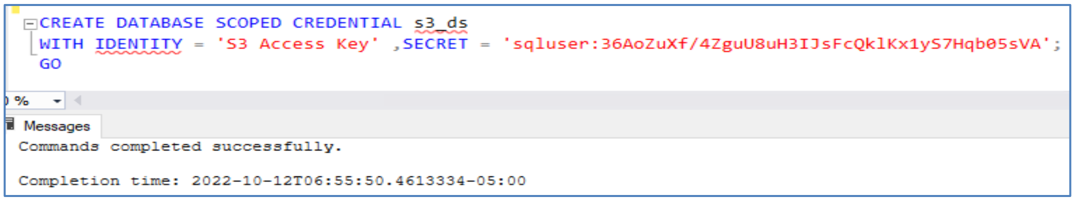


```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = '@Vantage4';
GO
```

```
Commands completed successfully.

Completion time: 2022-10-12T06:54:41.1936779-05:00
```

Figure 16. Create encryption key

2. Create database scope credentials within desired user database.

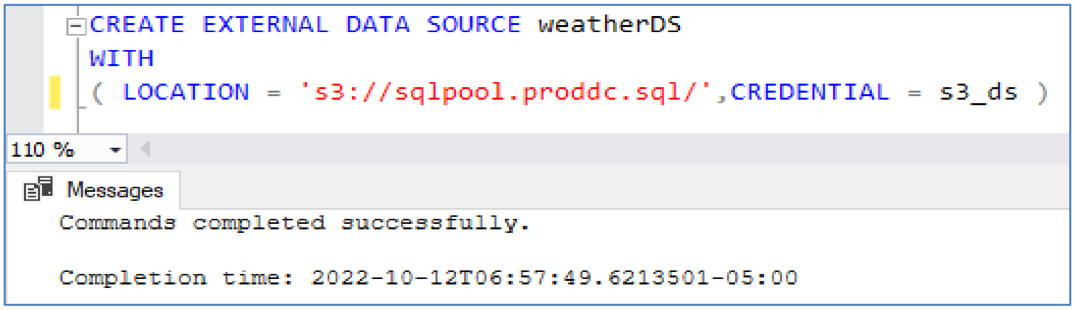


```
CREATE DATABASE SCOPED CREDENTIAL s3_ds
WITH IDENTITY = 'S3 Access Key' ,SECRET = 'sqluser:36AoZuXf/4ZguU8uH3IJsFcQklKx1yS7Hqb05sVA';
GO
```

```
Commands completed successfully.

Completion time: 2022-10-12T06:55:50.4613334-05:00
```

Figure 17. Create database scope credentials

3. Create external data source by pointing to S3 storage URL and database scoped credentials.

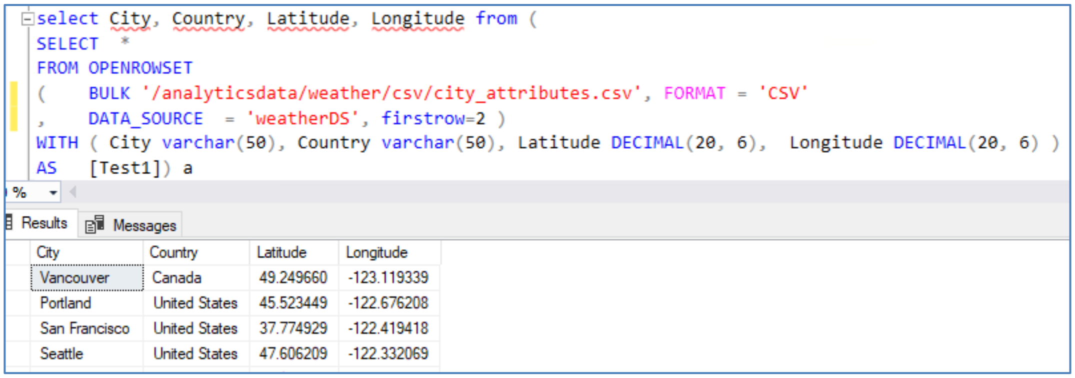


```
CREATE EXTERNAL DATA SOURCE weatherDS
WITH
( LOCATION = 's3://sqlpool.proddc.sql/',CREDENTIAL = s3_ds )
```

```
Commands completed successfully.

Completion time: 2022-10-12T06:57:49.6213501-05:00
```

Figure 18. Create external data source

4. Validate reachability to data present in S3 storage using OPENROWSET.



```
select City, Country, Latitude, Longitude from (
SELECT  *
FROM OPENROWSET
(     BULK '/analyticsdata/weather/csv/city_attributes.csv', FORMAT = 'CSV'
,     DATA_SOURCE  = 'weatherDS', firstrow=2 )
WITH ( City varchar(50), Country varchar(50), Latitude DECIMAL(20, 6),  Longitude DECIMAL(20, 6) )
AS   [Test1]) a
```

City	Country	Latitude	Longitude
Vancouver	Canada	49.249660	-123.119339
Portland	United States	45.523449	-122.676208
San Francisco	United States	37.774929	-122.419418
Seattle	United States	47.606209	-122.332069

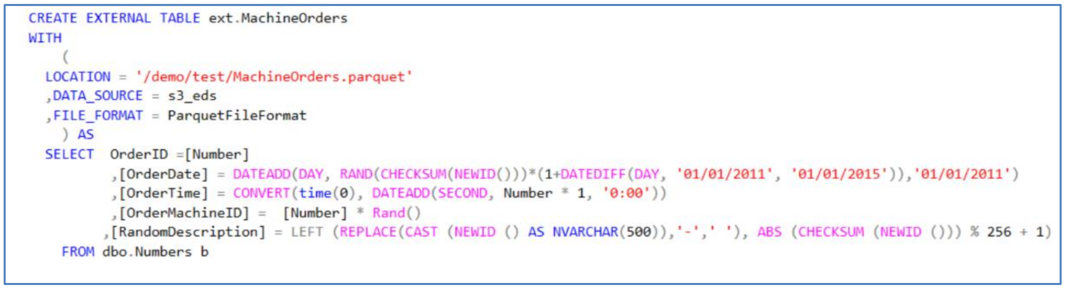
Figure 19. Configure and validate external data source

Parquet file

Parquet file is an Apache open-source column-oriented datafile format that is designed for efficient data storage and retrieval. It provides efficient data compression and encoding schemes with enhanced performance to handle complex data in bulk. Parquet is designed to be a common interchange format for both batch and interactive workloads.

SQL Server 2022 T-SQL enables the conversion of a .csv file into Parquet file format by using Create External Table as Select (CETAS) with OPENROWSET syntax. This is a powerful option to join relational data in SQL and non-relational data on object storage, such as Dell ECS.

CETAS can also be used to create external datasets directly, without ever landing within SQL, directly to a parquet file format.

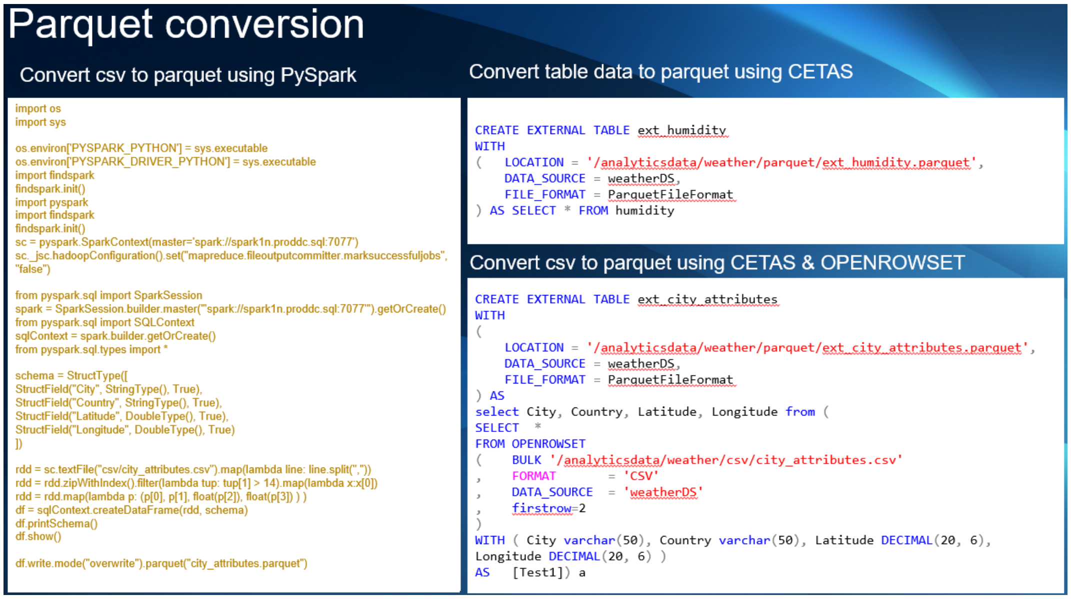


```
CREATE EXTERNAL TABLE ext.MachineOrders
WITH
    (
  LOCATION = '/demo/test/MachineOrders.parquet'
  ,DATA_SOURCE = s3_eds
  ,FILE_FORMAT = ParquetFileFormat
    ) AS
  SELECT  OrderID =[Number]
          ,[OrderDate] = DATEADD(DAY, RAND(CHECKSUM(NEWID()))*(1+DATEDIFF(DAY, '01/01/2011', '01/01/2015')),'01/01/2011')
          ,[OrderTime] = CONVERT(time(0), DATEADD(SECOND, Number * 1, '0:00'))
          ,[OrderMachineID] =  [Number] * Rand()
         ,[RandomDescription] = LEFT (REPLACE(CAST (NEWID () AS NVARCHAR(500)),'-',' '), ABS (CHECKSUM (NEWID ())) % 256 + 1)
    FROM dbo.Numbers b
```

Figure 20. Using CETAS to create external datasets

Users can also use other analytics data processing engines like Apache Spark or .csv to convert files into Parquet format.

Figure 21 shows two examples for data conversion into Parquet format: using PySpark and using CETAS with OPENROWSET.



Parquet conversion

Convert csv to parquet using PySpark

```
import os
import sys

os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable
import findspark
findspark.init()
import pyspark
import findspark
findspark.init()
sc = pyspark.SparkContext(master='spark://spark1n.proddc.sql:7077')
sc._jsc.hadoopConfiguration().set("mapreduce.fileoutputcommitter.marksuccessfuljobs",
"false")

from pyspark.sql import SparkSession
spark = SparkSession.builder.master('"spark://spark1n.proddc.sql:7077"').getOrCreate()
from pyspark.sql import SQLContext
sqlContext = spark.builder.getOrCreate()
from pyspark.sql.types import *

schema = StructType([
StructField("City", StringType(), True),
StructField("Country", StringType(), True),
StructField("Latitude", DoubleType(), True),
StructField("Longitude", DoubleType(), True)
])

rdd = sc.textFile("csv/city_attributes.csv").map(lambda line: line.split(","))
rdd = rdd.zipWithIndex().filter(lambda tup: tup[1] > 14).map(lambda x:x[0])
rdd = rdd.map(lambda p: (p[0], p[1], float(p[2]), float(p[3]) ) )
df = sqlContext.createDataFrame(rdd, schema)
df.printSchema()
df.show()

df.write.mode("overwrite").parquet("city_attributes.parquet")
```

Convert table data to parquet using CETAS

```
CREATE EXTERNAL TABLE ext_humidity
WITH
(   LOCATION = '/analyticsdata/weather/parquet/ext_humidity.parquet',
    DATA_SOURCE = weatherDS,
    FILE_FORMAT = ParquetFileFormat
) AS SELECT * FROM humidity
```

Convert csv to parquet using CETAS & OPENROWSET

```
CREATE EXTERNAL TABLE ext_city_attributes
WITH
(
    LOCATION = '/analyticsdata/weather/parquet/ext_city_attributes.parquet',
    DATA_SOURCE = weatherDS,
    FILE_FORMAT = ParquetFileFormat
) AS
select City, Country, Latitude, Longitude from (
SELECT  *
FROM OPENROWSET
(    BULK '/analyticsdata/weather/csv/city_attributes.csv'
,    FORMAT        = 'CSV'
,    DATA_SOURCE   = 'weatherDS'
,    firstrow=2
)
WITH ( City varchar(50), Country varchar(50), Latitude DECIMAL(20, 6),
Longitude DECIMAL(20, 6) )
AS   [Test1]) a
```

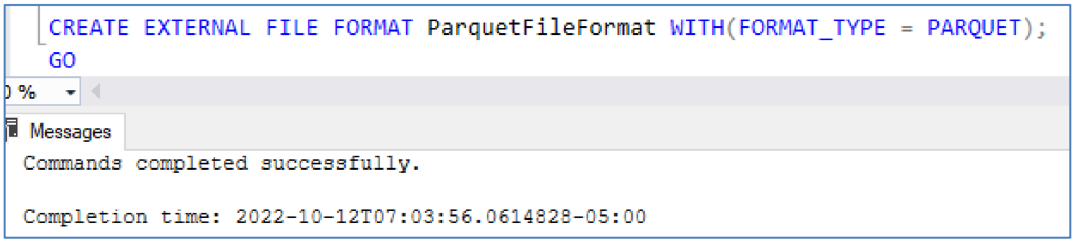
Figure 21. Parquet conversion methods

Working with data outside of SQL Server could be simplified using SQL Server external table. External table uses PolyBase to access data stored externally to SQL Server, in our case it would be ECS object storage.

Following configuration has to be created before creating the external table:

- An external file format
- An external data source and
- Location of the external files

The following screenshots show how to create the external file format for parquet files.

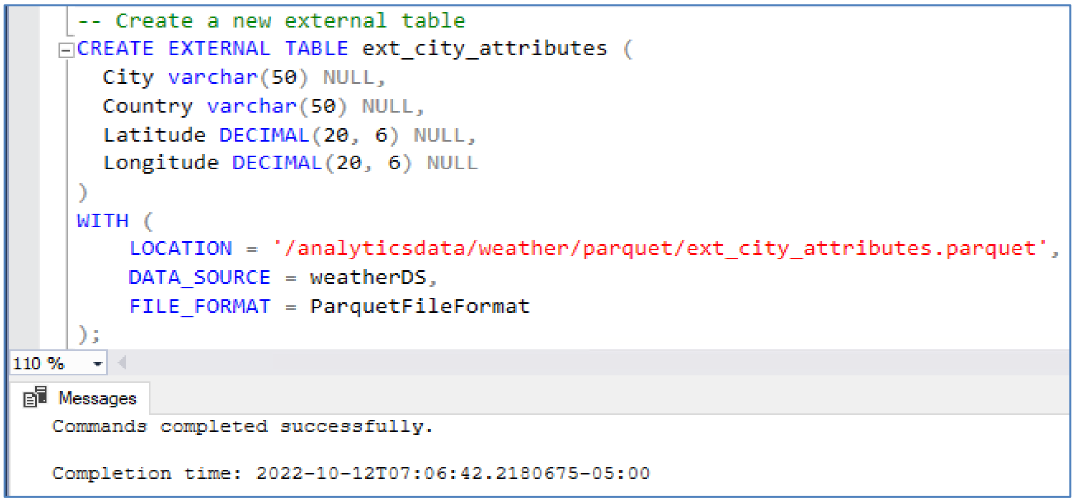


```
CREATE EXTERNAL FILE FORMAT ParquetFileFormat WITH(FORMAT_TYPE = PARQUET);
GO
```

```
Commands completed successfully.

Completion time: 2022-10-12T07:03:56.0614828-05:00
```

Figure 22. Create external table

Create external table pointing to Parquet files on S3 storage by providing file location, data source, and file format.

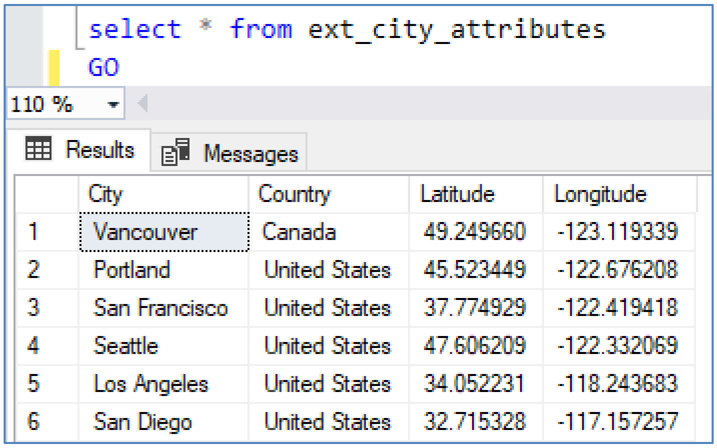


```
-- Create a new external table
CREATE EXTERNAL TABLE ext_city_attributes (
  City varchar(50) NULL,
  Country varchar(50) NULL,
  Latitude DECIMAL(20, 6) NULL,
  Longitude DECIMAL(20, 6) NULL
)
WITH (
    LOCATION = '/analyticsdata/weather/parquet/ext_city_attributes.parquet',
    DATA_SOURCE = weatherDS,
    FILE_FORMAT = ParquetFileFormat
);
```

```
Commands completed successfully.

Completion time: 2022-10-12T07:06:42.2180675-05:00
```

Figure 23. Add file location, data source, and file format to parquet file

Select data from external table like any other tables in SQL Server database.



```
select * from ext_city_attributes
GO
```

	City	Country	Latitude	Longitude
1	Vancouver	Canada	49.249660	-123.119339
2	Portland	United States	45.523449	-122.676208
3	San Francisco	United States	37.774929	-122.419418
4	Seattle	United States	47.606209	-122.332069
5	Los Angeles	United States	34.052231	-118.243683
6	San Diego	United States	32.715328	-117.157257

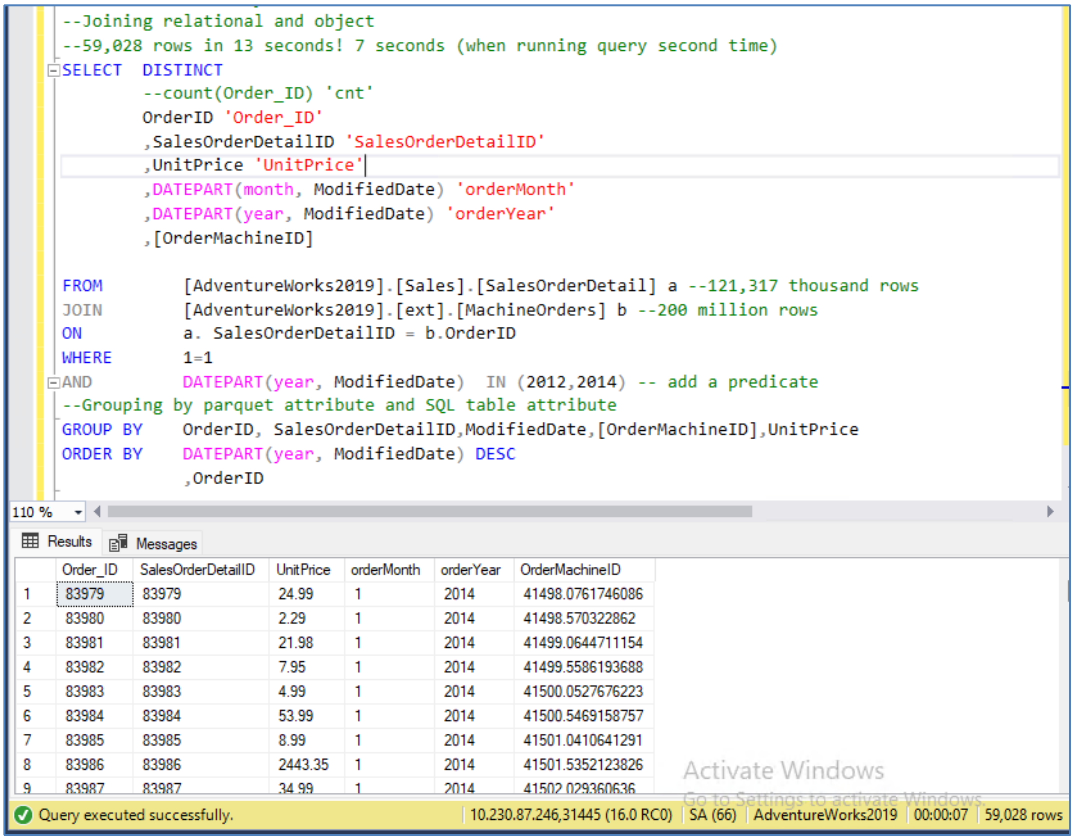
Figure 24. Select data for parquet file

Relevant studies

The Dell solutions engineering team conducted additional tests and studies to understand the feasibility of reading hundreds of millions of records present on ECS object storage.

These studies involved joined relational data present in the SalesOrderDetail table in AdventureWorks database with around 121 thousand records along with external table ext.MachineOrders backed by Parquet files on ECS with 200 million records.

Figure 25 provides an example of joining the relational table and external table with millions of records that can return results within seconds.



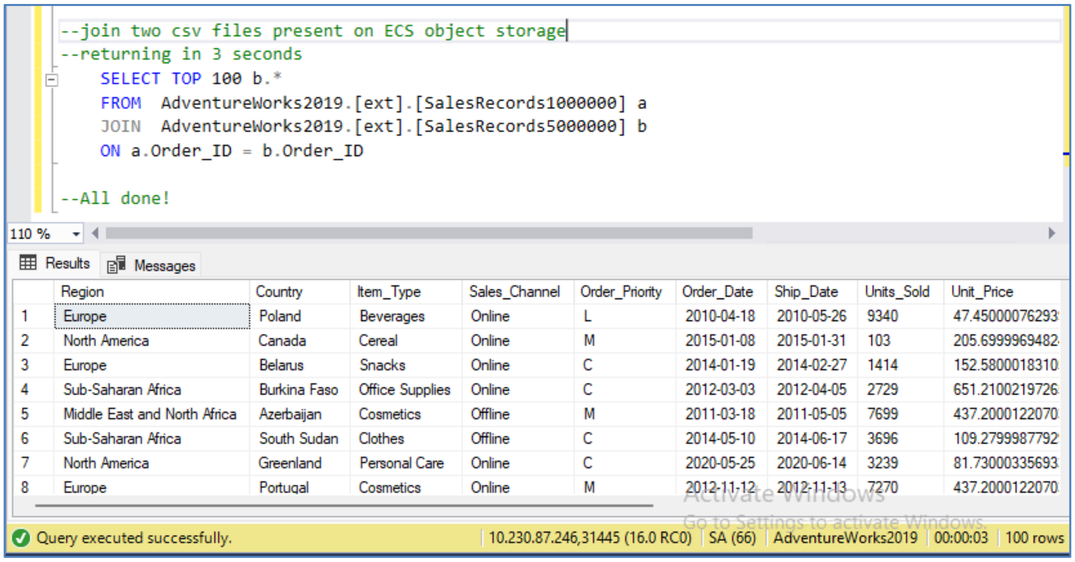
```
--Joining relational and object
--59,028 rows in 13 seconds! 7 seconds (when running query second time)
SELECT  DISTINCT
        --count(Order_ID) 'cnt'
        OrderID 'Order_ID'
        ,SalesOrderDetailID 'SalesOrderDetailID'
        ,UnitPrice 'UnitPrice'
        ,DATEPART(month, ModifiedDate) 'orderMonth'
        ,DATEPART(year, ModifiedDate) 'orderYear'
        ,[OrderMachineID]

FROM        [AdventureWorks2019].[Sales].[SalesOrderDetail] a --121,317 thousand rows
JOIN        [AdventureWorks2019].[ext].[MachineOrders] b --200 million rows
ON          a. SalesOrderDetailID = b.OrderID
WHERE       1=1
AND         DATEPART(year, ModifiedDate)  IN (2012,2014) -- add a predicate
--Grouping by parquet attribute and SQL table attribute
GROUP BY    OrderID, SalesOrderDetailID,ModifiedDate,[OrderMachineID],UnitPrice
ORDER BY    DATEPART(year, ModifiedDate) DESC
            ,OrderID
```

	Order_ID	SalesOrderDetailID	UnitPrice	orderMonth	orderYear	OrderMachineID
1	83979	83979	24.99	1	2014	41498.0761746086
2	83980	83980	2.29	1	2014	41498.570322862
3	83981	83981	21.98	1	2014	41499.0644711154
4	83982	83982	7.95	1	2014	41499.5586193688
5	83983	83983	4.99	1	2014	41500.0527676223
6	83984	83984	53.99	1	2014	41500.5469158757
7	83985	83985	8.99	1	2014	41501.0410641291
8	83986	83986	2443.35	1	2014	41501.5352123826
9	83987	83987	34.99	1	2014	41502.029360636

Query executed successfully. | 10.230.87.246,31445 (16.0 RC0) | SA (66) | AdventureWorks2019 | 00:00:07 | 59,028 rows

Figure 25. Relational table and external table

In a separate test, the Dell solutions engineers used SQL Server as query and data hub only where all the data is external to SQL Server and present on ECS object storage. To do this, they joined two external tables backed by .csv file formats with one and five million records, respectively.



```
--join two csv files present on ECS object storage
--returning in 3 seconds
    SELECT TOP 100 b.*
    FROM  AdventureWorks2019.[ext].[SalesRecords1000000] a
    JOIN  AdventureWorks2019.[ext].[SalesRecords5000000] b
    ON a.Order_ID = b.Order_ID

--All done!
```

	Region	Country	Item_Type	Sales_Channel	Order_Priority	Order_Date	Ship_Date	Units_Sold	Unit_Price
1	Europe	Poland	Beverages	Online	L	2010-04-18	2010-05-26	9340	47.45000076293
2	North America	Canada	Cereal	Online	M	2015-01-08	2015-01-31	103	205.6999969482
3	Europe	Belarus	Snacks	Online	C	2014-01-19	2014-02-27	1414	152.5800018310
4	Sub-Saharan Africa	Burkina Faso	Office Supplies	Online	C	2012-03-03	2012-04-05	2729	651.2100219726
5	Middle East and North Africa	Azerbaijan	Cosmetics	Offline	M	2011-03-18	2011-05-05	7699	437.2000122070
6	Sub-Saharan Africa	South Sudan	Clothes	Offline	C	2014-05-10	2014-06-17	3696	109.2799987792
7	North America	Greenland	Personal Care	Online	C	2020-05-25	2020-06-14	3239	81.73000335693
8	Europe	Portugal	Cosmetics	Online	M	2012-11-12	2012-11-13	7270	437.2000122070

Query executed successfully. | 10.230.87.246,31445 (16.0 RC0) | SA (66) | AdventureWorks2019 | 00:00:03 | 100 rows

Figure 26. Two external tables backed by .csv file formats

This new way of accessing semi-structured and unstructured data residing on object storage like Dell ECS storage, which is external to SQL Server. This access allows various use cases and new opportunities of performing ETL/ELT activities or offload ETL/ELT process altogether.

Conclusion

Both large and small organizations can benefit from the new features introduced in SQL Server 2022. These features include backup, restore, object storage, and the use of T-SQL for data conversion.

Embracing the powerful AMD EPYC 7473X processors and Dell ECS object storage give organizations a competitive edge in their analytic workloads. This combination of products not only delivers deeper insights quicker for business leaders, but it also provides a platform that is powerful, robust, highly available, flexible, and scalable.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email.

Author: Tom Dau, Sanjeev Ranjan, Robert Sonders

Contributors: Ava English, Stephen McMaster

Note: For links to additional documentation for this solution, see the Dell Technologies Solutions Info Hub for SQL Server.

References

Dell Technologies documentation

The following Dell Technologies documentation provides additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- Dell Elastic Cloud Storage (ECS)
- Dell ECS and Microsoft SQL 2022 S3 Object Storage

AMD documentation

The following AMD documentation provides additional information about AMD EPYC 7473X Processors.

- AMD EPYC 7473X Processor

Microsoft documentation

The following Oracle documentation provides additional and relevant information:

- SQL Server backup and restore with S3-compatible object storage
- SQL Server 2022 CETAS