SQL Server 2022 Database Solution with Object Storage on Dell Hardware Stack

A Dell validated design for Data Analytics and Data Protection solution with SQL Server 2022 and Dell ECS object storage

January 2023

H19462

Design Guide

Abstract

This design guide describes the architecture, components, and deployment steps for the SQL Server 2022 S3 Data Analytics solution on Dell's hardware stack. This solution validates data virtualization by connecting SQL Server instances to external object storage using S3 protocol, enhancing data protection by backing up and restoring from Dell ECS object storage, and exploring newly introduced T-SQL functions for better analysis of data.

Dell Technologies Solutions

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.

Published in the USA 01/23 Design Guide H19462.

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

Contents

Chapter 1. Introduction

This chapter presents the following topics:

SQL Server 2022 is the latest version of Microsoft's relational database engine. This document focuses on Data Analytics and Data Protection using the newly introduced features of SQL Server 2022 PolyBase for the backup and recovery of databases using S3-compatible Elastic Cloud Storage (ECS). Milan-X is the new AMD EPYC processor with 3D V-Cache technology, and it supports the analytic operations in this solution. It is the first CPU with 3D chiplet technology. This processor has three times the L3 cache of a standard Milan processor and is well suited for analytic workloads.

Business challenge

As enterprises look to reduce costs and increase data availability, technologies such as data virtualization are used to meet these organizations' data needs. Data analytics is the process of analyzing raw, structured, and unstructured data to answer business questions and identify trends.

Data virtualization allows enterprises to seek flexible and cost-effective storage options for structured, unstructured, and semi-structured data like ECS, which uses Simple Storage Services (S3) to streamline data pipelines. The PolyBase feature enables new business opportunities for Microsoft SQL Server through data virtualization. Data virtualization is sought after by large enterprises because it allows them to face the challenges of working with unstructured and semi-structured data while minimizing costs.

Data accessibility is an expectation among enterprises, and data virtualization facilitates the consumption of data to enable quick data driven decisions. Organizations are adopting data virtualization to virtually integrate different types of data for data mining, machine learning, artificial intelligence, and data analytics.

Organizations rely on data analytics to gain descriptive, predictive, prescriptive, and diagnostic based insights so that they can make meaningful changes to the enterprise's operations. Data virtualization allows for the appropriate infrastructure to be put in place to provide high availability for these business-critical data analytic services.

Storage and CPU requirements are important considerations because data analytic workloads are frequently CPU and storage intensive. This document seeks to provide insight into CPU and storage options that will meet the needs of all types of enterprises.

Solution introduction

This Dell Validated Design uses Dell's PowerEdge servers with AMD EPYC 7473X processors, Elastic Cloud Storage and PowerStore array as the underlying solution infrastructure. This design intends to provide solution insights for data engineers, data architects and data scientists who intend to run analytical workloads using Microsoft SQL Server 2022 and object storage.

This design will detail the process of establishing highly available SQL engines for Windows and Linux environments. It also examines the new data analytics and data protection features for SQL Server. This design will demonstrate the implementation of secure and accessible critical infrastructure through this solution setup.

It is vital to select the optimal CPUs for data analytic servers because analytic workloads are regularly CPU intensive. To select the appropriate CPU, it is essential to consider the quantity of cores, their frequency, and the Level 3 cache size.

T-SQL queries for data analytics require quick response times, so AMD 7473X processors were chosen for the SQL Server 2022 database instances. The AMD EPYC 7473X processors have twenty-four cores per socket at 2.8 GHz with a Level 3 cache size of 768 MB which provides enough horsepower for data analytic workloads.

It is crucial to have flexible and scalable S3-compatible object storage and the optimal CPUs for data processing for most analytic workloads.

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. The advantages and features of Dell ECS include:

Cloud Scale Storage

- Globally distributed object infrastructure
- Exabyte+ scale without any 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 object size limits

Flexible deployment

- Appliance deployment
- Software-only deployment with support for certified or custom industry standard hardware
- Multiprotocol support for Object (S3, Swift, Atmos, CAS) and File (HDFS, NFSv3) storage
- Supports workloads for both modern 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 Security

- 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 a 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)

Total Cost of Ownership reduction

- Global namespace
- High storage utilization
- Small and large file performance
- Seamless Centera migration
- Fully compliant with Atmos REST
- Low management overhead
- Small data center footprint

Microsoft SQL Server is widely used across all industries, and these data sources are commonly mission critical. The ability to backup and restore SQL Server databases is crucial. The ability of SQL Server 2022 to backup and restore to S3-compatible object storage provides additional flexibility through cloud connectivity. To use this feature, T-SQL provides the TO URL syntax for backup and FROM URL syntax for restore.

SQL Server 2022 PolyBase makes data virtualization possible for data scientists using T-SQL for analytic workloads, by querying data directly from other sources such as Oracle, Teradata, Hadoop cluster, and S3-compatible object storage without separately installing client connection software. PolyBase allows T-SQL queries to join data from external sources with relational tables in an instance of SQL Server. The T-SQL OPENROWSET and EXTERNAL TABLE syntaxes are useful for querying data in S3-compatible storage.

Data virtualization is a broad term that describes a data management approach. It allows an application to retrieve and manipulate data without requiring the data’s technical details, such as the data’s physical location and source format. Data virtualization involves abstracting various sources through a single data access layer. Organizations are adopting tools and software to integrate different types of data virtually. This data integration enables both data mining and analytics, and it is critical for predictive analytics tools for use of machine learning (ML) and artificial intelligence (AI).

Dell solutions engineers created two physical architecture setups, using VMware virtualization and Red Hat OpenShift with SQL Server 2022 to test analytic workloads using T-SQL. These architectures use Dell ECS for external data and PowerStore for the data found in the local SQL Server 2022 instance.

Audience

The scope of this Dell Validated Design paper is to offer a database solution for data engineers, data scientists, and architects. This design guide is intended for enterprises who will run analytic workloads on SQL Server 2022 with object storage and use Dell PowerEdge servers with AMD EPYC 7473X processors, Dell ECS, and a Dell PowerStore storage array as the underlying infrastructure.

Design guide introduction

Integrated solutions are created using principles that provide customers with greater value than designing their own solution, which accelerates time-to-market and reduces risks.

This design guide intends to establish a SQL Server database solution that is differentiated by its adherence to protection, scalability, and manageability principles. This solution has been validated by Dell engineers to ensure that the performance of this solution is reliable and fault tolerant.

Dell systems engineers tested Microsoft SQL Server with Dell's Elastic Cloud Storage (ECS) to ensure that the solution is scalable. This storage is highly extensible and can expand its namespaces across many storage systems. This external storage is extremely scalable, and Microsoft SQL Server can use this storage easily through the PolyBase feature and Simple Storage Services (S3) protocol.

The solution was validated with vSphere High Availability and Red Hat OpenShift deployments because manageability is critical when designing a suitable solution for large enterprises. Microsoft SQL Server can back up and restore from Dell's Elastic Cloud Storage, which helps protect essential data.

Terminology

The following table provides definitions for some key terms used in this document.

Term	Definition
Data Virtualization	Data virtualization allows applications to retrieve and manipulate data without requiring the data source's format and physical location.
C & I (Client and Infrastructure) Node	The Client and Infrastructure node is used to manage different solution components that clusters do not handle. This node provides client machines and essential technologies such as domain name services, domain controller services, NTP services, and benchmarking.
CSAH (Cluster System Admin Host) Node	The Cluster System Admin Host node is not a part of the cluster, but it is required for OpenShift cluster administration**.** The authentication tokens needed to administer an OpenShift cluster are installed on the CSAH node as part of the deployment process, whereas OpenShift CLI administration tools are deployed onto the control-plane nodes. Dell strongly discourages managing clusters through control-plane nodes.
Dell Elastic Cloud Storage (ECS)	Elastic Cloud Storage is an on-premises software-defined object storage platform that is an alternative to public cloud solutions offering scalability, flexibility, and resiliency. ECS empowers organizations to capture, store, protect, and manage unstructured data at a scale that matches the public cloud, behind an enterprise's firewall. ECS provides a flat, scale-out architecture and maintains strong global consistency, enabling organizations to lower total data ownership costs. ECS enables management of globally distributed storage infrastructure with a single global namespace, providing users access to critical business data through S3, Atmos, CAS, Swift, NFSv3, and HDFS.
Namespace	Namespaces provide a way to organize or group items in isolated storage spaces for different business purposes.

Term	Definition
Bucket	A bucket is a container for objects. Buckets can be assigned to namespaces and can have metadata associated with them.
Dell PowerStore	A storage appliance that offers all flash storage and supports the NVMe communication protocol. Dell PowerStore supports VMware vVols. PowerStore and can be scaled up and out.
vVol	Virtual disk containers, otherwise known as Virtual Volumes, are defined by VMware vVols and operate independently of underlying physical storage representation. These virtual disks eliminate pre-allocated LUNs/Volumes by serving as the primary unit of data management.
Logical Unit Number (LUN)	A LUN is defined by SCSI standards to uniquely identify an individual or collection of physical or virtual storage devices.
Container	Containers are lightweight applications decoupled from underlying host infrastructure.
Dell Container Storage Interface (CSI)	Dell Container Storage Interface allows containers to communicate with storage volumes (LUN) and abstract them into persistent volumes (PV).
High Availability	This is the ability for a system to constantly provide service despite potential failures of the underlying components.
Load Balancer	A load balancer is used to manage the traffic within a system and to ensure that requests are sent to the appropriate available nodes.
Microservice	Microservices follow a software architectural design pattern that allows for larger applications to be made up of smaller services that are kept independent as highly cohesive subcomponents. These services communicate through APIs and can be scaled using container orchestrators such as Kubernetes/OpenShift.
Virtualization	Virtualization allows a computer’s resources to be abstracted and shared by virtual machines.
Secure Socket Layer and Transport Layer Security (SSL/TSL)	SSL is a protocol for establishing secure links between networked computers. SSL is deprecated and its successor is TSL, however it is still often referred to as SSL/TSL.

Chapter 2. Solution architecture

This chapter presents the following topics:

Physical architecture

Dell solutions engineers validated two resilient architectural options for data analytics workloads:

1. SQL Server 2022 on VMware Virtualization Platform
2. SQL Server 2022 on RedHat OpenShift Containerized Platform

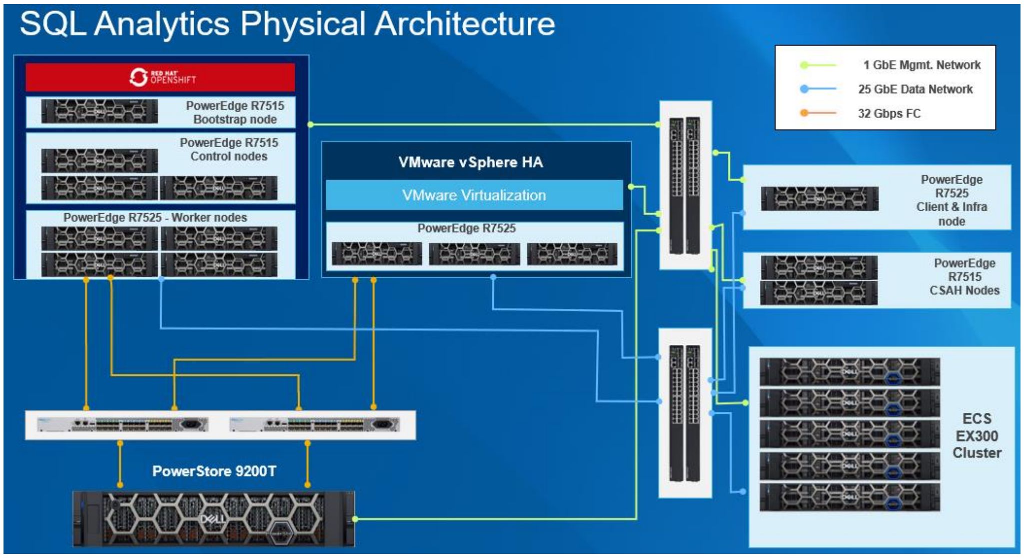

SQL Analytics Physical Architecture
RED HAT OPENSHIFT
PowerEdge R7515 Bootstrap node
PowerEdge R7515 Control nodes
PowerEdge R7525 - Worker nodes
VMware vSphere HA
VMware Virtualization
PowerEdge R7525
1 GbE Mgmt. Network
25 GbE Data Network
32 Gbps FC
PowerEdge R7525 Client & Infra node
PowerEdge R7515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

Figure 1. SQL Server 2022 database solution physical architecture.

Overview

The physical architecture design shown above demonstrates two separate database solution environments which connect with a shared Dell ECS unit and Dell PowerStore storage to process and save data.

The upper-left portion of the diagram shows a containerized environment based on a RedHat OpenShift Cluster. The central portion of the diagram illustrates an environment that uses VMware vSphere virtualization.

The clusters are connected to the Client and Infrastructure (C & I) node on the right side of the diagram and the Cluster System Admin Hosts (CSAH) nodes using Dell networking switches. Dell PowerSwitch 1 GbE and 25 GbE were used for network connectivity with the OpenShift and VMware vSphere HA clusters, and the same switches were also used to connect with a Dell ECS EX300 cluster for object storage.

The Dell PowerStore Storage is connected to both containerized and virtualized environments by Fiber Channel (FC) connectivity protocol with Dell Connectrix switches.

Server layers

Dell solutions engineers separate server layers into three groups based on their primary roles within the solution:

1. A RedHat OpenShift containerized environment consisting of eight PowerEdge R7525 rack servers.
2. A VMware vSphere based virtualized environment which consists of three PowerEdge R7525 rack servers.

3. The C & I node, along with the CSAH nodes, consisted of PowerEdge R7515 and R7525 servers.

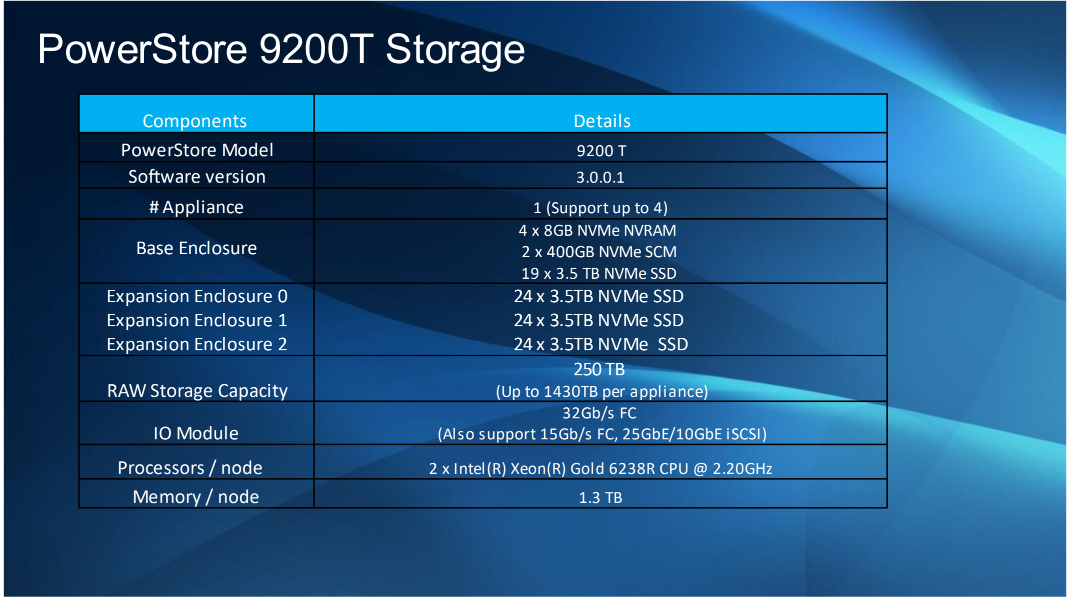
The Dell PowerEdge R7525 Rack Server is a highly adaptable rack server that delivers powerful performance and offers flexible configurations. The PowerEdge R7525 is a 2U rack server that supports up to:

- Two 2nd or 3rd Generation AMD EPYC processors, with up to sixty-four cores per processor
- Thirty-six DIMM slots supporting DDR4 RDIMM (2 TB), LRDIMM (4 TB), and bandwidth speeds of up to 3200 MT/S
- Twenty-four NVMe drives
- Three double-width 300 W GPUs, or six single-width 75 W GPUs
- 2400 W from Dual AC or DC power supply units

Storage layer

In the validated setup, two separate Dell storage units were used to process the different types of data.

- **Dell PowerStore 9200T:** Dell PowerStore achieves new levels of operational simplicity and agility, using a container-based architecture, advanced storage technologies, and intelligent automation to unlock data value. Based on a scale-out architecture and hardware-accelerated advanced data reduction, PowerStore is designed to deliver enhanced resource utilization and performance that keeps pace with application and system growth.

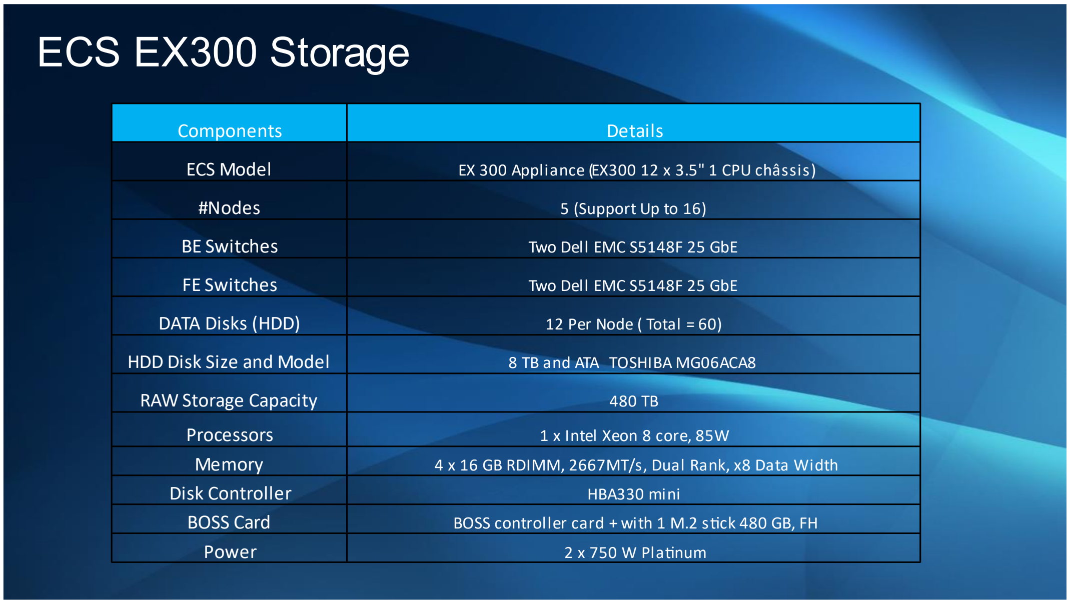


PowerStore 9200T Storage

Components	Details
PowerStore Model	9200 T
Software version	3.0.0.1
# Appliance	1 (Support up to 4)
Base Enclosure	4 x 8GB NVMe NVRAM 2 x 400GB NVMe SCM 19 x 3.5 TB NVMe SSD
Expansion Enclosure 0 Expansion Enclosure 1 Expansion Enclosure 2	24 x 3.5TB NVMe SSD 24 x 3.5TB NVMe SSD 24 x 3.5TB NVMe SSD
RAW Storage Capacity	250 TB (Up to 1430TB per appliance)
IO Module	32Gb/s FC (Also support 15Gb/s FC, 25GbE/10GbE iSCSI)
Processors / node	2 x Intel(R) Xeon(R) Gold 6238R CPU @ 2.20GHz
Memory / node	1.3 TB

Figure 2. PowerStore 9200T configuration details

- **Dell ECS EX300:** Elastic Cloud Storage (ECS) provides a complete software-defined cloud storage platform supporting the storage, manipulation, and analysis of unstructured data on commodity hardware at scale. ECS can be installed on a set of qualified commodity servers and disks or deployed as a turnkey storage appliance. ECS offers the cost advantages of commodity infrastructure with the enterprise reliability, availability, and serviceability of traditional arrays.



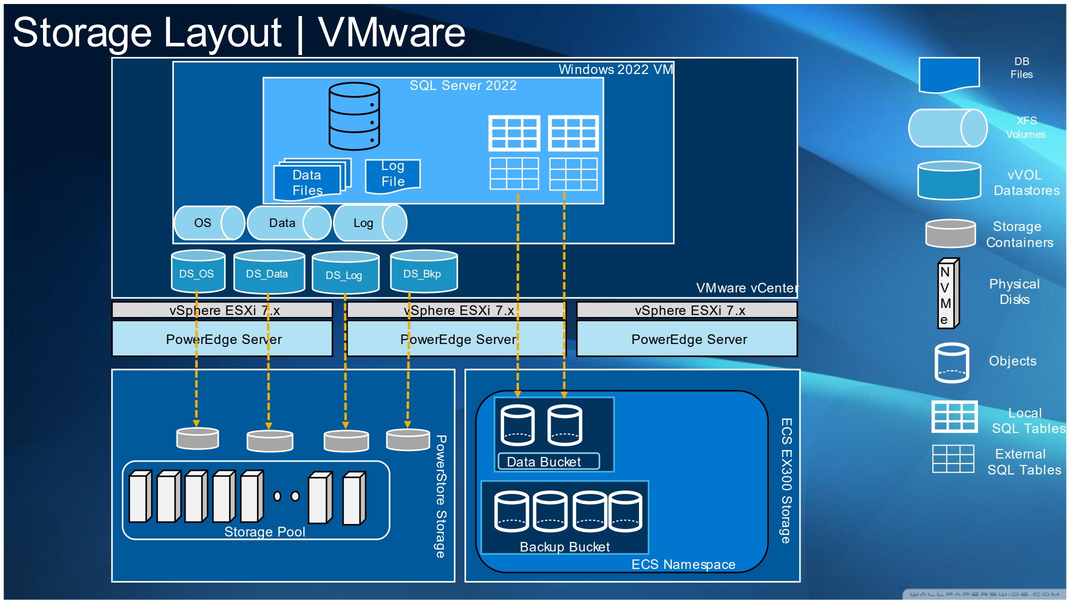
ECS EX300 Storage

Components	Details
ECS Model	EX 300 Appliance (EX300 12 x 3.5" 1 CPU châssis)
#Nodes	5 (Support Up to 16)
BE Switches	Two Dell EMC S5148F 25 GbE
FE Switches	Two Dell EMC S5148F 25 GbE
DATA Disks (HDD)	12 Per Node (Total = 60)
HDD Disk Size and Model	8 TB and ATA TOSHIBA MG06ACA8
RAW Storage Capacity	480 TB
Processors	1 x Intel Xeon 8 core, 85W
Memory	4 x 16 GB RDIMM, 2667MT/s, Dual Rank, x8 Data Width
Disk Controller	HBA330 mini
BOSS Card	BOSS controller card + with 1 M.2 stick 480 GB, FH
Power	2 x 750 W Platinum

Figure 3. Dell ECS EX300 storage cluster configuration details

Storage layout

Figure 4 represents the logical construction of a storage design. Here storage containers are carved out of a Dell PowerStore storage pool that was presented to the ESXi layer using LUNs. It was mapped to vVOLs datastore and presented as a volume within the virtual machines. Those volumes were formatted and configured as XFS volumes. The operating system and the databases data and log files are placed on the XFS volumes.



Storage Layout | VMware

Figure 4. Storage layout and mapping for Dell validated solution

Furthermore, on Dell ECS storage, different buckets for data and backups were created and virtually mapped to SQL Server 2022 external tables and backupset.

Network layer

The architecture's network layer consists of:

- Two 25 GbE network switches: Connected to two 25 GbE ports on all physical nodes and client machine to route the workload network traffic.
- Two 1 GbE network switches: Connected to two 1 GbE ports on all physical nodes and client machine to manage network traffic.
- Two 32 Gbps Fiber channel switches: Connected to two 32 Gbps ports on all physical nodes to establish a Storage Area Network with manual zoning and CSI automated zoning

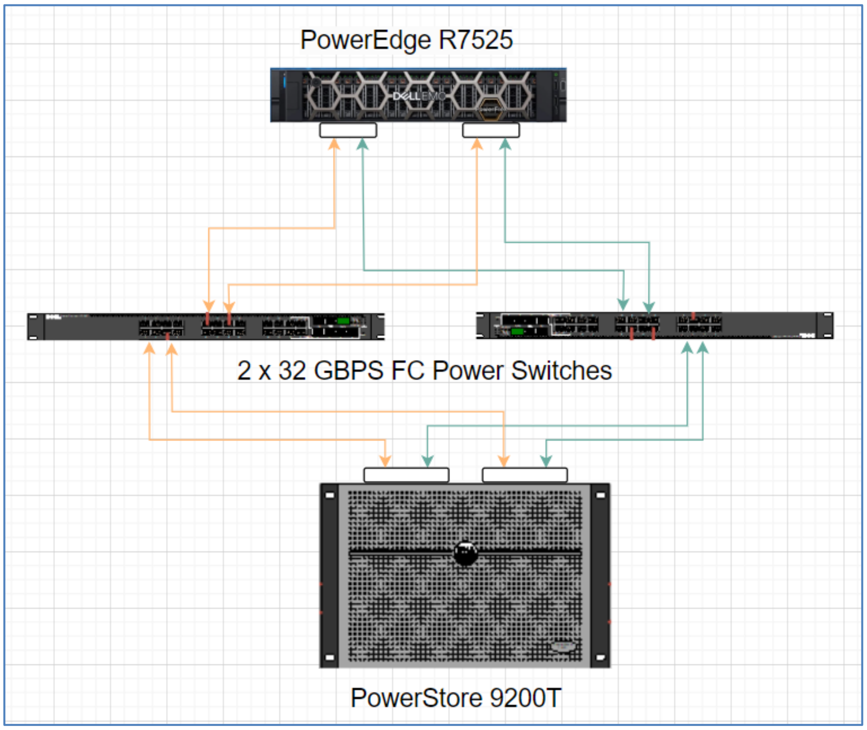


Figure 5. Fiber channel connectivity diagram for Dell PowerEdge and PowerStore

The fiber channel connectivity diagram shows an example of Fiber channel zoning for PowerEdge R7525 servers using PowerStore 9200T storage and two 32GBPS FC Power Switches. The diagram shows four 32GB front end PowerStore ports which are mapped to four 32 GB network card ports on the server ensuring that the servers have reliable storage access. Similarly, three R7525 servers are zoned with PowerStore to create a virtualized environment with four R7525 servers, the OpenShift worker nodes, to create the storage for the containerized environment.

Logical architecture

Dell solutions engineers validated SQL Server 2022 on two types of operating systems:

- Windows
- Linux

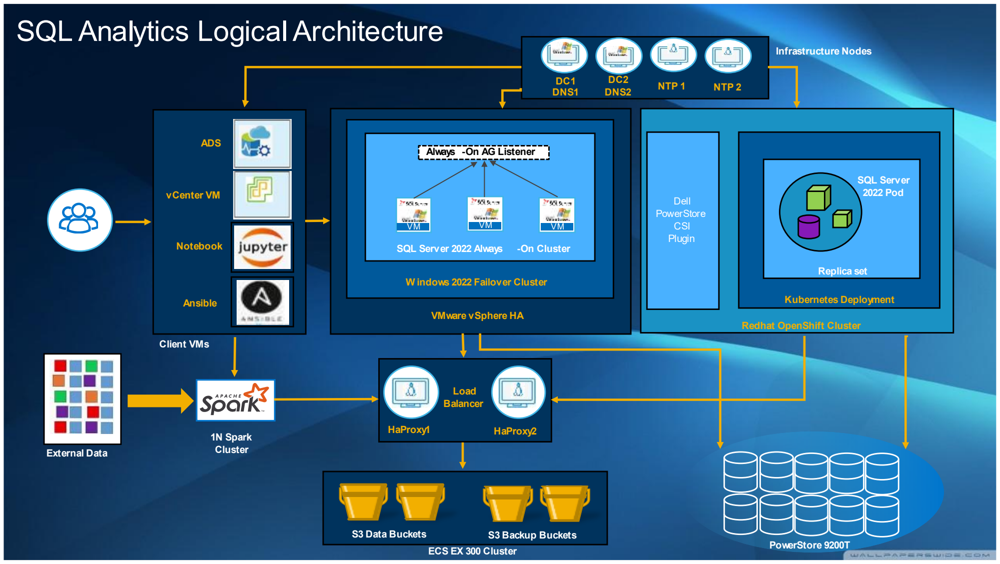


Figure 6. SQL Server 2022 database solution Logical Architecture

The first setup of SQL Server 2022 runs on a VMware High Availability cluster using Windows virtual machines and vVols. The second setup uses Red Hat OpenShift as the container orchestration tool for Red Hat Enterprise Linux SQL Server containers.

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

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

Virtualization design

In this setup, database files are stored within storage containers provided by the Dell PowerStore 9200T storage along with the local tables for the virtualized SQL instances. SQL Server 2022 also makes use of PolyBase and S3 to access and manipulate data stored in ECS bucket objects. This data can be stored in many different formats and allows both unstructured and semi-structured data to be stored alongside structured data.

In this setup, database files are stored within storage containers provided by the Dell PowerStore 9200T storage along with the local tables for the virtualized SQL instances. SQL Server 2022 also uses PolyBase and S3 to access and manipulate data stored in ECS bucket objects. This data can be stored in many different formats and allows both unstructured and semi-structured data to be stored alongside structured data.

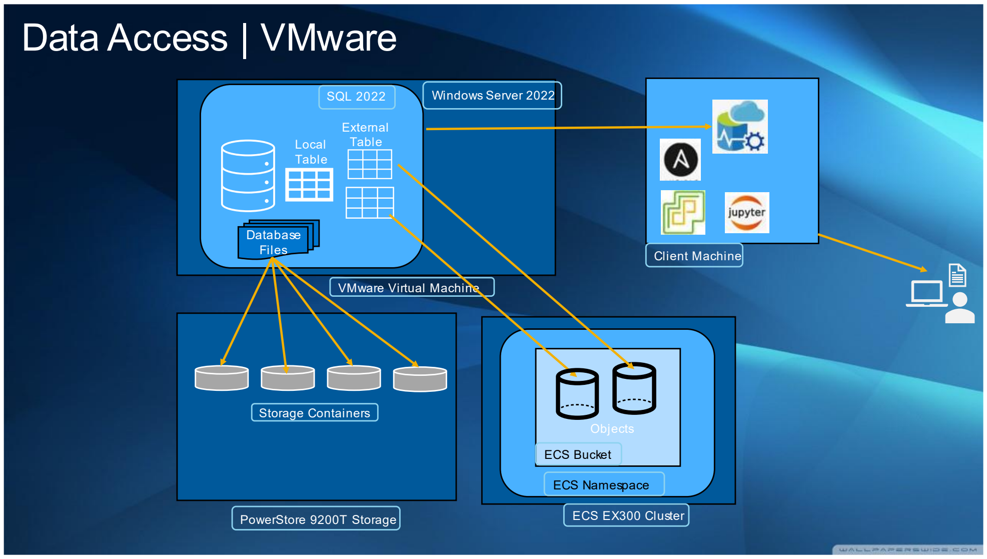


Figure 7. Virtualized environment data access pattern diagram

Container design

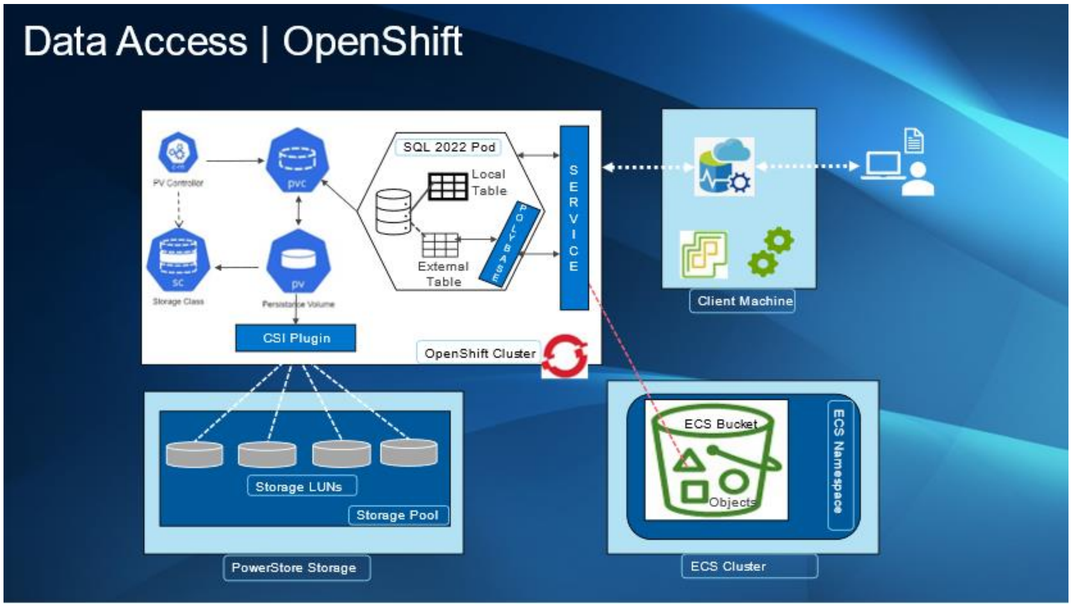
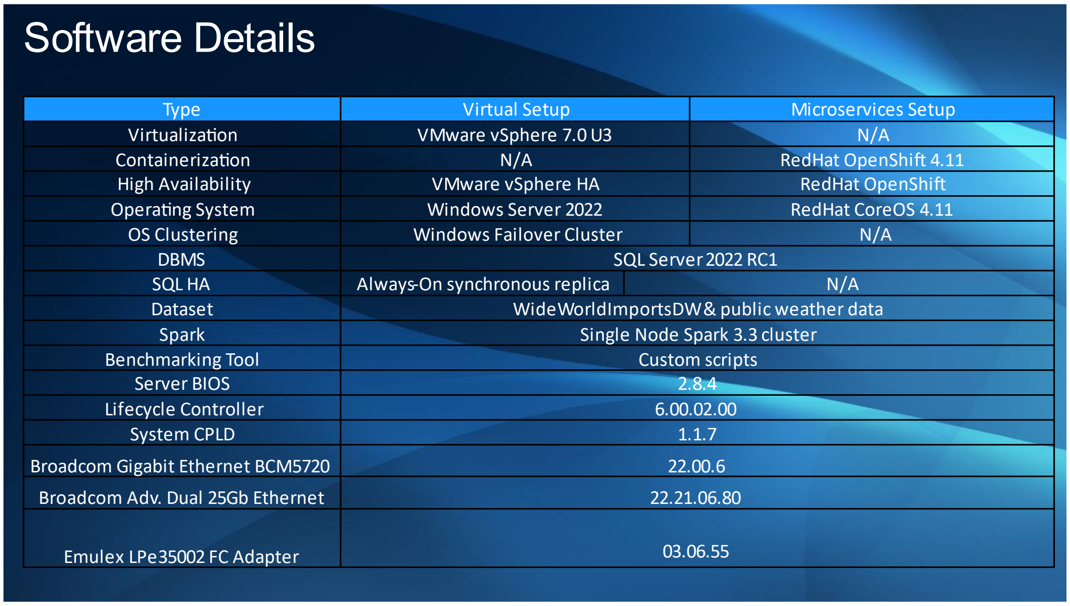


Figure 8. Data access pattern in containerized environments

The Dell CSI plugin allows for the creation of persistent volumes based on the storage's LUNs in the PowerStore storage pool. It is required that the proper storage class is established for Dell CSI. This serves as the storage for the Microsoft SQL Server 2022 container instances and is where the local tables are stored. External tables are created using data found in the ECS Cluster using PolyBase for Microsoft SQL Server and the S3 for ECS. Data is contained entirely in ECS data buckets within a namespace. This data can be accessed and modified without being stored in the Microsoft SQL Server instance.

Software



Software Details

Type	Virtual Setup	Microservices Setup
Virtualization	VMware vSphere 7.0 U3	N/A
Containerization	N/A	RedHat OpenShift 4.11
High Availability	VMware vSphere HA	RedHat OpenShift
Operating System	Windows Server 2022	RedHat CoreOS 4.11
OS Clustering	Windows Failover Cluster	N/A
DBMS	SQL Server 2022 RC1	
SQL HA	Always-On synchronous replica	N/A
Dataset	WideWorldImportsDW & public weather data	
Spark	Single Node Spark 3.3 cluster	
Benchmarking Tool	Custom scripts	
Server BIOS	2.8.4	
Lifecycle Controller	6.00.02.00	
System CPLD	1.1.7	
Broadcom Gigabit Ethernet BCM5720	22.00.6	
Broadcom Adv. Dual 25Gb Ethernet	22.21.06.80	
Emulex LPe35002 FC Adapter	03.06.55	

Figure 9. Dell validated database solution software details

General software details

Both set ups make use the same datasets and single node Spark 3.3 cluster. Custom scripts were used as the benchmarking tool in this setup. SQL Server 2022 RC1 was used as the data analytics engine in this solution. All servers were running with BIOS version 2.8.4, Lifecycle Controller version 6.00.02.00, and version 1.1.7 for the System Complex Programmable Logic Device (CPLD). The Broadcom Gigabit Ethernet BCM5720 was at version 22.00.6 and the Broadcom Advance Dual 25GB Ethernet 22.21.06.80. The Emulex LPe35002 FC Adapter is running on version 03.06.55.

Virtual setup software details

VMware vSphere 7.0 U3 serves as the basis for managing the hosts of the virtual setup which is crucial for running SQL Server on Windows Server 2022 instances. High Availability is provided by VMware’s vSphere HA and the servers are clustered using the Window’s failover cluster. SQL High Availability is provided for this setup through the Always-On synchronous replica feature.

Microservice setup software details

RedHat’s OpenShift is crucial for provisioning Linux containers for SQL Server and maintaining the high availability of these services. RedHat Core OS 4.11 is used as the base operating system for the containers. SQL Server is made highly available through replica sets deployed on Kubernetes and is the ideal setup for a microservice system.

Chapter 3. Solution deployments

This chapter presents the following topics:

Overview

Deploying any database solution is as crucial as designing it. To deploy a database properly, consider:

- Utilizing the appropriate order of deployment
- Applying customary best-practices
- Connecting solution components with appropriate network considerations
- Validating that the overall deployment works as expected

This chapter describes the necessary steps to deploy the solution. The deployment steps are separate according to the technologies used within the two environments.

Initial setup

There are several components of the solution that must be set up initially.

- **iDRAC Setup**: The Integrated Dell Remote Access Controller (iDRAC) is designed to give system administrators increased productivity and to improve the availability of Dell servers. iDRAC reduces the need for physical access to Dell systems by allowing remote management and provides alerts for system issues.
- **BIOS Setup**: BIOS versions are available from the Dell Support Site. This solution uses the latest available BIOS; however, it may be necessary to select a different BIOS depending on the environment and workloads an enterprise plans on deploying.
- **Firmware Setup**: Firmware requirements will vary based on the components used within physical servers such as the Dell PowerEdge R7525. It is important to deploy the appropriate firmware and to upgrade the firmware to its latest releases.

Dell provides documentation explaining the process of setting up iDRAC and deploying the BIOS and other firmware on Dell servers. Please view the Dell EMC PowerEdge R7525 Installation and Service Manual for detailed instructions.

HAProxy LoadBalancer for Dell ECS storage

ECS is Dell’s third-generation object storage platform designed for traditional and next-generation applications providing flexible deployment, resiliency, and simplicity. ECS is a collection of software that seamlessly incorporates hardware nodes with disks and switches to provide access to object storage data.

Using a load balancer is recommended so that the load can be distributed between internal ECS nodes and ECS clusters in separate locations. ECS does not have specific load balancer requirements.

HAProxy provides a low-cost option for customers who want a reliable load balancer for ECS. Dell provides documentation for using the HAProxy LoadBalancer with Dell’s ECS.

Certificate creation and installation

End to end encrypted communication between the ECS storage and the SQL Server 2022 instance is required to perform data virtualization. A local certificate authority was created and used to generate and sign a private certificate validating the authenticity of communication between the two interfaces.

Certificate creation

Certificate Creation

1. Create the root SSL certificate and provide the password when prompted to generate the key file.

```
openssl genrsa -des3 -out certs/rootCA.key 2048
```

2. Create a server.csr.cnf file based on the following format with appropriate environment details.

```
[req]
default_bits = 2048
prompt = no
default_md = sha256
distinguished_name = dn
[dn]
C=US
ST=Durham
L=Durham
O=Dell
OU=Bizapp
emailAddress=admin@proddc.sql
CN = sqlpool.proddc.sql
```

3. Create a root CA certificate using the key generated in step one and the server.csr.cnf file. The expiration date for the certificate can be set using the *-days* parameter. This command will provide a rootCA.pem file. This .pem file can be deployed in Windows, Linux, or a container environment.

```
openssl req -x509 -new -nodes -key certs/rootCA.key -sha256
-days 1460 -out certs/rootCA.pem -config server.csr.cnf
```

Deploying the certificate in a Windows environment

Move the rootCA.pem file to a Windows host running a SQL Server instance. Open a PowerShell session as administrator and run the following command to add the provided certificate to the Windows “ROOT” certificate store.

```
certutil -addstore -f "ROOT" rootCA.pem
```

Deploying the certificate in a RHEL environment

1. The .pem file must be converted to a .crt file using the following command.

```
openssl x509 -in rootCA.pem -inform PEM -out rootCA.crt
```

2. Copy the rootCA.crt file to the folder “/etc/pki/ca-trust/source/anchors/” on Red Hat Enterprise Linux. Run the following command to install the certificate.

```
/bin/update-ca-trust
```

Mapping a certificate in a containerized environment

The process of mapping a certificate to a containerized environment is different from traditional OS environments. A ConfigMap based on the root certificate must be created and then mapped as a volume in the deployment script. This process is expanded upon in the SQL Server 2022 Deployment section.

Generating a SSL client certificate for LoadBalancer deployment

1. The private key and the CSR for the client certificate are created using the following command.

```
openssl req -new -sha256 -nodes -out certs/server.csr -newkey rsa:2048 -
keyout certs/server.key -config server.csr.cnf
```

2. Create a v3.ext file in the following format with details that suit the environment.

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation,
keyEncipherment, dataEncipherment
subjectAltName = @alt_names
[alt_names]
DNS.1 = sqlpool.proddc.sql
IP.1 = 10.230.87.43
```

3. Now, create the Client certificate using the Root CA, CSR and the v3.ext file.

```
openssl x509 -req -in certs/server.csr -CA certs/rootCA.pem -CAkey
certs/rootCA.key -CAcreateserial -out certs/server.crt -days 1460 -sha256 -
extfile v3.ext
```

4. Finally, combine the Client certificate and key for HAProxy LoadBalancer.

```
cat certs/server.key certs/server.crt > certs/combined.pem
```

Deploying a certificate on HAProxy LoadBalancer for Dell ECS

Once the certificate generation has been completed, the front-end definition in the HAProxy configuration file, “haproxy.cfg,” can be defined for HTTPS.

```
frontend https-in
   bind *:443 ssl crt /etc/haproxy/combined.pem
   reqadd X-Forwarded-Proto:\ https
 # Define the hostnames
   acl host_s3 hdr(host) -i -m dom sqlpool.proddc.sql
   acl host_s3_ip hdr(host) -i -m dom 10.230.87.43
 # Route to backend
   use_backend s3_backend if host_s3
   use_backend s3_backend if host_s3_ip
```

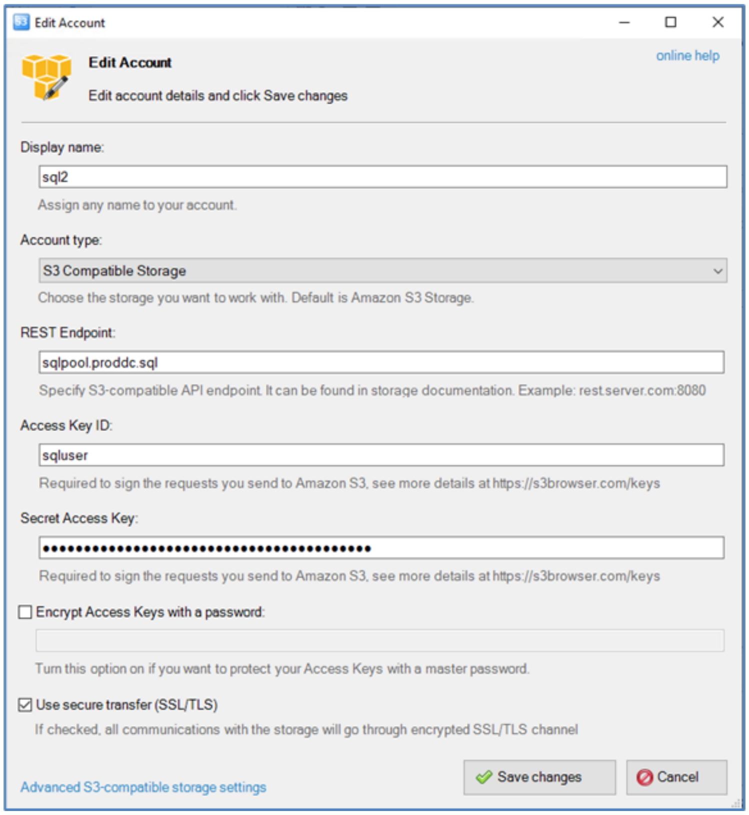
In this example, SSL is terminated at the HAProxy load balancer and thus certificates do not need to be created for the ECS nodes and the non-SSL ports of the ECS nodes will be used as defined in the HTTP backend section of the haproxy.cfg configuration file.

Once the changes in “haproxy.cfg” are processed, verify the configuration file, and restart the HAProxy service to activate the certificate and load balancing directives defined for “https-in”.

Validating SSL communication

There are multiple ways to validate encrypted communication through HTTPS to the Dell ECS storage cluster. In the use case section, SQL Server is used to access objects present on ECS storage with S3. However, to quickly validate that the communication is working, use the free “S3 Browser” application.

The Access Key ID and Secret Access Key that are obtained from Dell ECS storage along with the rest endpoint for the HAProxy LoadBalancer are used for this communication. Make sure to select “Use secure transfer (SSL/TLS)”.



Edit Account

Edit Account

Edit account details and click Save changes

online help

Display name:

sql2

Assign any name to your account.

Account type:

S3 Compatible Storage

Choose the storage you want to work with. Default is Amazon S3 Storage.

REST Endpoint:

sqlpool.proddc.sql

Specify S3-compatible API endpoint. It can be found in storage documentation. Example: rest.server.com:8080

Access Key ID:

sqluser

Required to sign the requests you send to Amazon S3, see more details at https://s3browser.com/keys

Secret Access Key:

••

Required to sign the requests you send to Amazon S3, see more details at https://s3browser.com/keys

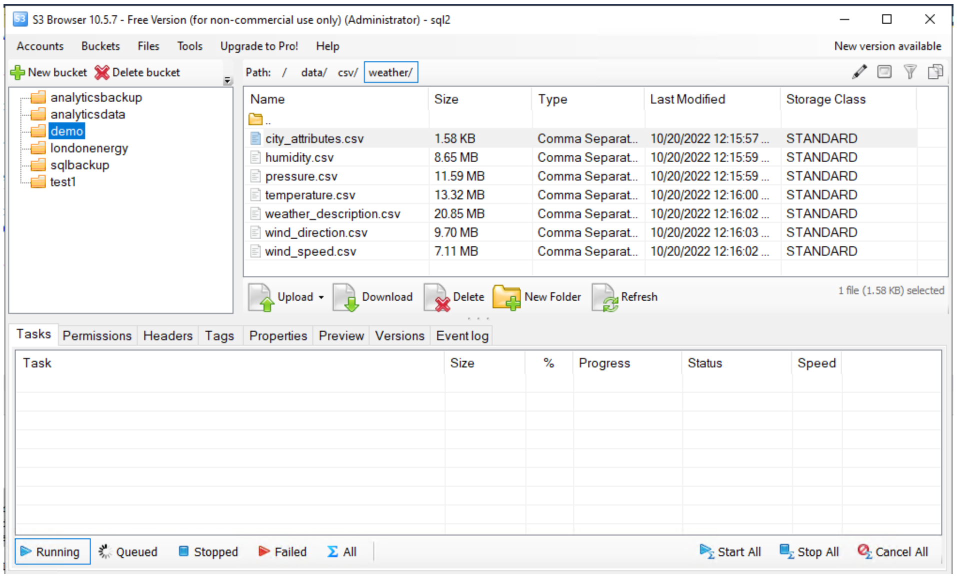
□ Encrypt Access Keys with a password:

Turn this option on if you want to protect your Access Keys with a master password.

☑ Use secure transfer (SSL/TLS)

If checked, all communications with the storage will go through encrypted SSL/TLS channel

Figure 10. Connection settings for S3 Browser



Path: / data/ csv/ weather/

Name	Size	Type	Last Modified	Storage Class
..				
city_attributes.csv	1.58 KB	Comma Separat...	10/20/2022 12:15:57 ...	STANDARD
humidity.csv	8.65 MB	Comma Separat...	10/20/2022 12:15:59 ...	STANDARD
pressure.csv	11.59 MB	Comma Separat...	10/20/2022 12:15:59 ...	STANDARD
temperature.csv	13.32 MB	Comma Separat...	10/20/2022 12:16:00 ...	STANDARD
weather_description.csv	20.85 MB	Comma Separat...	10/20/2022 12:16:02 ...	STANDARD
wind_direction.csv	9.70 MB	Comma Separat...	10/20/2022 12:16:03 ...	STANDARD
wind_speed.csv	7.11 MB	Comma Separat...	10/20/2022 12:16:02 ...	STANDARD

1 file (1.58 KB) selected

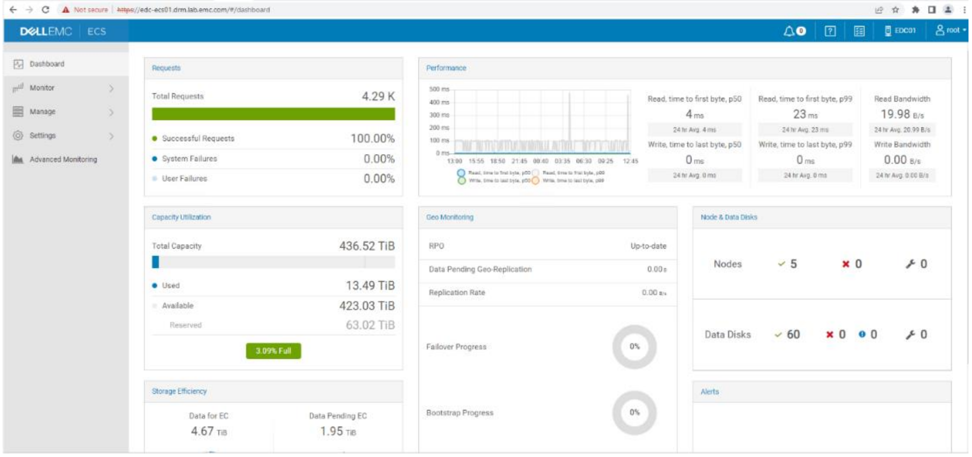
Figure 11. Browsing Dell ECS storage objects using S3 browser

Connect to ECS storage and browse through the buckets and objects within the applicable namespaces once an account has been created.

Setup access to Dell ECS storage cluster

ECS dashboard

After logging into the ECS Cluster, the dashboard page displays general cluster information including details about capacity, performance, CPU and memory usage details, nodes health, requests and more. This is a great tool for multiple administrators to remotely manage enterprise storage.



Total Requests 4.29 K
Successful Requests 100.00%
System Failures 0.00%
User Failures 0.00%

Read, time to first byte, p50 4 ms (24 hr Avg. 4 ms)
Read, time to first byte, p99 23 ms (24 hr Avg. 23 ms)
Read Bandwidth 19.98 B/s (24 hr Avg. 20.99 B/s)
Write, time to last byte, p50 0 ms (24 hr Avg. 0 ms)
Write, time to last byte, p99 0 ms (24 hr Avg. 0 ms)
Write Bandwidth 0.00 B/s (24 hr Avg. 0.00 B/s)

Total Capacity 436.52 TiB
Used 13.49 TiB
Available 423.03 TiB
Reserved 63.02 TiB
3.09% Full

RPO Up-to-date
Data Pending Geo-Replication 0.00 s
Replication Rate 0.00 B/s
Failover Progress 0%
Bootstrap Progress 0%

Nodes ✓ 5 ✗ 0 0
Data Disks ✓ 60 ✗ 0 0 0

Data for EC 4.67 TiB
Data Pending EC 1.95 TiB

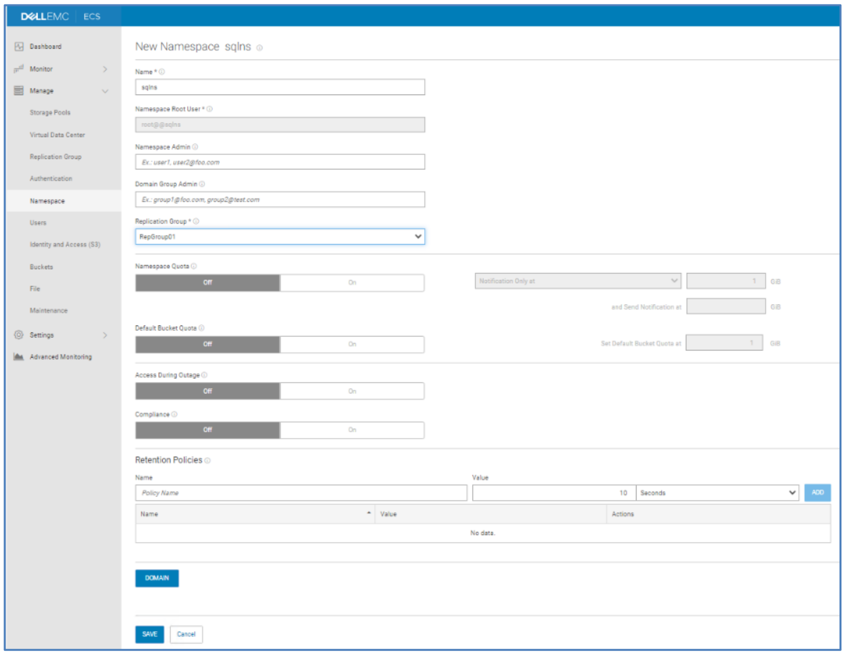
Figure 12. Dell ECS storage cluster dashboard

Namespaces in Dell ECS

Namespaces allow for the segregation of storage space which reduces coupling, and they enable ECS's multi-tenancy feature. Unlike storage pools and replication groups, many namespaces can be created. However, a single namespace is appropriate for some environments. There are several beneficial use cases for namespaces:

- Providing unique namespaces for distinct business units.
- Separating data for use with specific applications.

- Creating unique namespaces for each reporting boundary. Buckets can also be reported on.
- Providing a unique namespace to each subscriber for a service. Dell ECS Test Drive is configured with a unique namespace being created for each user.
- As a workaround, namespaces can be used to allow the targeting of specific replication groups for legacy applications. Some legacy applications cannot access a specific storage pool so it may be necessary for these applications to use buckets that access storage pools using specific replication groups.



New Namespace sqlns

Name * sqlns

Namespace Root User * root@@sqlns

Namespace Admin Ex.: user1, user2@foo.com

Domain Group Admin Ex.: group1@foo.com, group2@test.com

Replication Group * RepGroup01

Namespace Quota: Off On — Notification Only at 1 GiB — and Send Notification at GiB

Default Bucket Quota: Off On — Set Default Bucket Quota at 1 GiB

Access During Outage: Off On

Compliance: Off On

Retention Policies

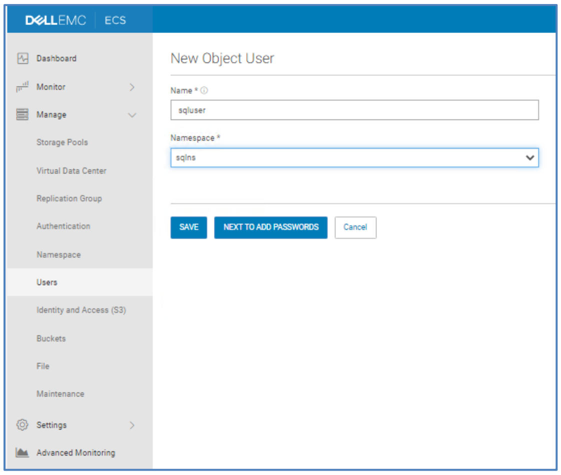
Name: Policy Name — Value: 10 Seconds

Name	Value	Actions
No data.		

Figure 13. Dell ECS storage cluster namespace creation dialogue

ECS User Creation

The following screenshots show steps to create a user in Dell ECS storage. Users can access the objects present in ECS storage.

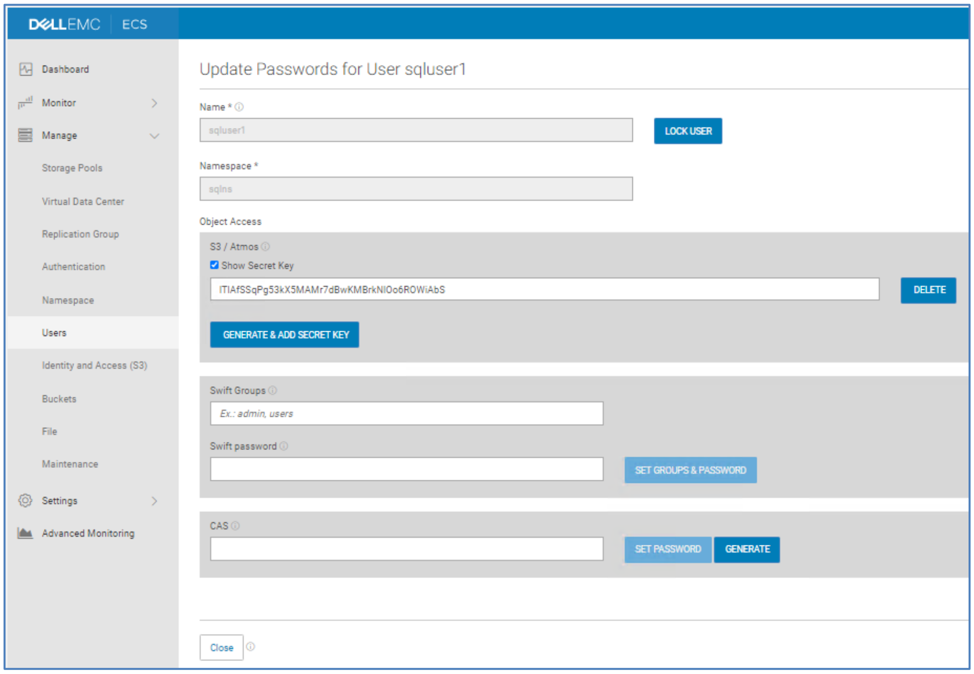


New Object User

Name * sqluser

Namespace * sqlns

Figure 14. Dell ECS storage cluster namespace creation dialogue



Update Passwords for User sqluser1

Name * sqluser1

Namespace * sqlns

Object Access

S3 / Atmos

☑ Show Secret Key

ITIAfSSqPg53kX5MAMr7dBwKMBrkNIOo6ROWiAbS

Swift Groups: Ex.: admin, users

Swift password

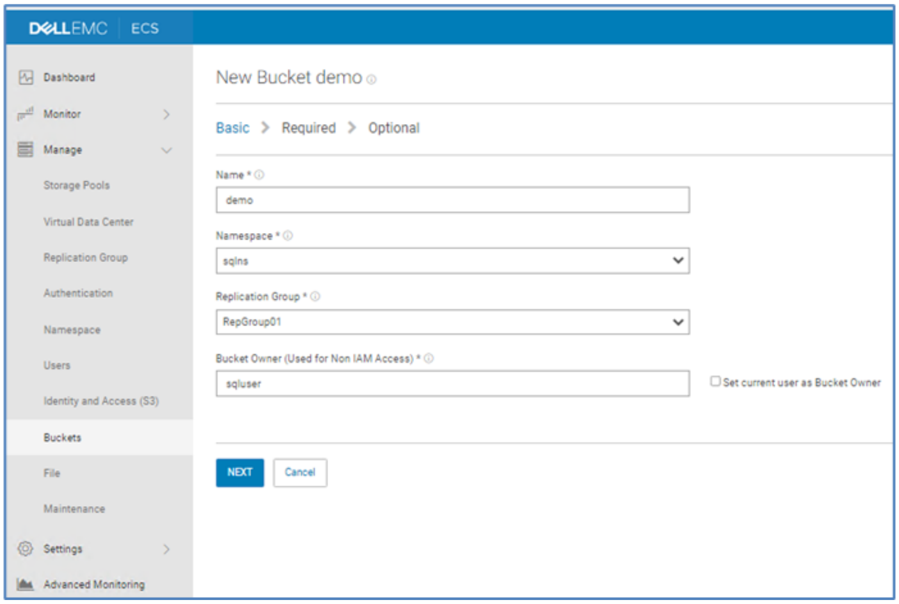
CAS

Figure 15. Update a user’s access token and password for the storage cluster

ECS bucket creation

Buckets contain objects created in a namespace and are sometimes considered a logical container for sub-tenants. Each namespace is created within a Replication Group (RG). The term ‘buckets’ has been adopted by ECS because that is what S3 calls containers. Buckets are global resources in ECS that can span multiple sites.

Bucket creation involves assigning it to a namespace and a Replication Group. The bucket level is where the ownership and file or CAS access is enabled.



New Bucket demo

Name * demo

Namespace * sqlns

Replication Group * RepGroup01

Bucket Owner (Used for Non IAM Access) * sqluser

☐ Set current user as Bucket Owner

Figure 16. Dell ECS Bucket creation dialogue

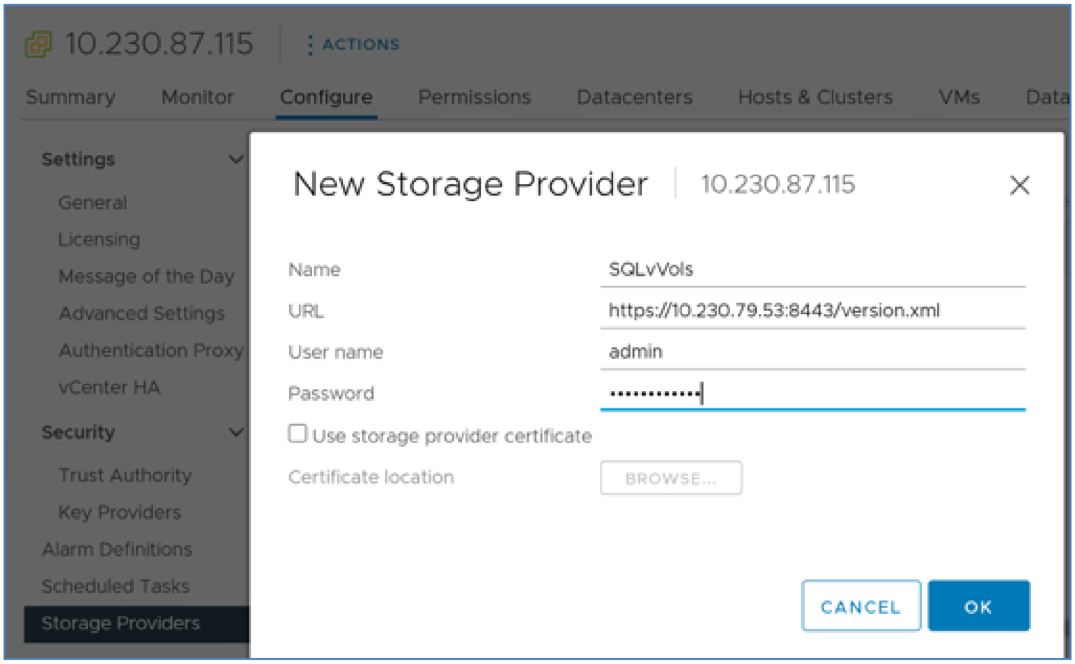
Configuring vVols on Dell PowerStore Storage

Overview

vVols are a technology from VMware that offers a better way of provisioning, managing, and accessing virtual disks. The access and management of vVols is enabled through a software component called the vStorage APIs for Storage Awareness (VASA) provider, also known as a vendor provider. VASA providers are developed by storage vendors such as Dell. The vendor provider enables the creation of protocol end points which act as access points for hosts and storage systems. vVols offer multiple benefits such as:

- Storage policy-based management (SPBM) which allows the creation and application of policies based on storage tiers.
- Improved datastore management.
- Enhanced fine-grained storage operations and services.

For more information about the benefit of vVols, visit vSphere Virtual Volumes (vVols) and vVols Getting Started Guide.



New Storage Provider | 10.230.87.115

Name: SQLvVols

URL: https://10.230.79.53:8443/version.xml

User name: admin

Password: ••••••••••••

□ Use storage provider certificate

Certificate location: BROWSE...

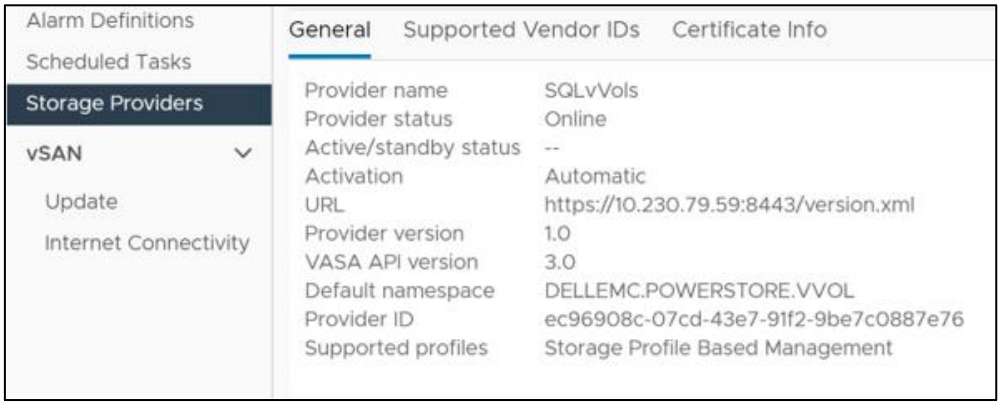
Figure 17. Creating a new storage provider for PowerStore in VMware vCenter

Creating a new storage provider for PowerStore in VMware vCenter

To use vVols, the VASA provider must be registered with vCenter. The storage capabilities are exported and presented to VMware based virtual infrastructure through the protocol end points enabled by VASA.

Storage Provider Registration Procedure

1. Login to vSphere client.
2. Go to vCenter Server in the vSphere Web Client navigator.
3. Click the Configure tab and click **Storage Providers**.
4. Click the **+ Add** icon to register a storage provider.
5. Enter the connection details for the new storage provider, including the name, URL, and credentials.
6. Click **OK** to complete the registration.



Provider name	SQLvVols
Provider status	Online
Active/standby status	--
Activation	Automatic
URL	https://10.230.79.59:8443/version.xml
Provider version	1.0
VASA API version	3.0
Default namespace	DELLEMC.POWERSTORE.VVOL
Provider ID	ec96908c-07cd-43e7-91f2-9be7c0887e76
Supported profiles	Storage Profile Based Management

Figure 18. Storage provider registration summary

7. The storage provider VASA has been successfully registered with the vCenter as shown in the above screenshot.
8. Once the VASA is registered, create a vVol based datastore to use vVol based services.

Virtual Datastore Creation Procedure

1. Log in to the vSphere Web Client.
2. Select the host in the vSphere inventory.
3. Right-click the host and select the storage menu.
4. Click **New Datastore**.

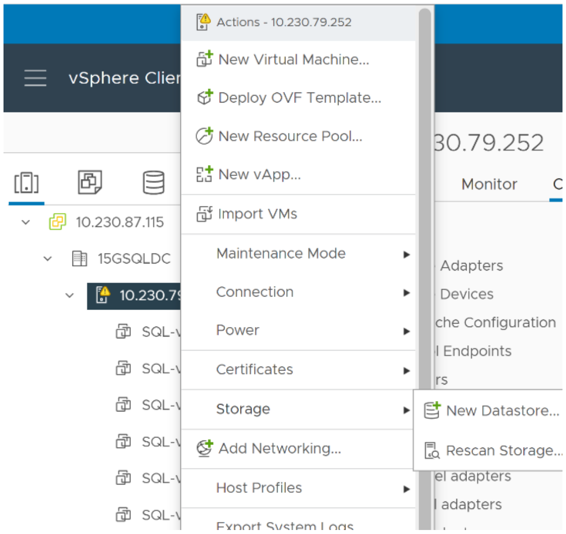


Figure 19. Creating a VMware vSphere datastore

Creating a VMware vSphere datastore

1. Enter a unique datastore name.

2. Select vVol as the virtual datastore type.

3. Select the appropriate storage container referring to the previously registered VASA. This backing container will be used to host the virtual volumes.

4. Click next to review the selections and click Finish.

5. At the end of the procedure, the newly created datastore can be viewed on the host.
6. To access the datastore on additional hosts, the datastore must be explicitly mounted on the required hosts based on system requirements.
7. Once the datastore is present on the VMware vCenter, virtual disks can be carved out from the datastore and mapped to a given virtual machine.

Virtual environment setup

Dell solutions engineers deployed a virtual environment in accordance with the previously described architecture. For this setup, the following technologies from VMware were used:

- VMware vSphere
- VMware vCenter

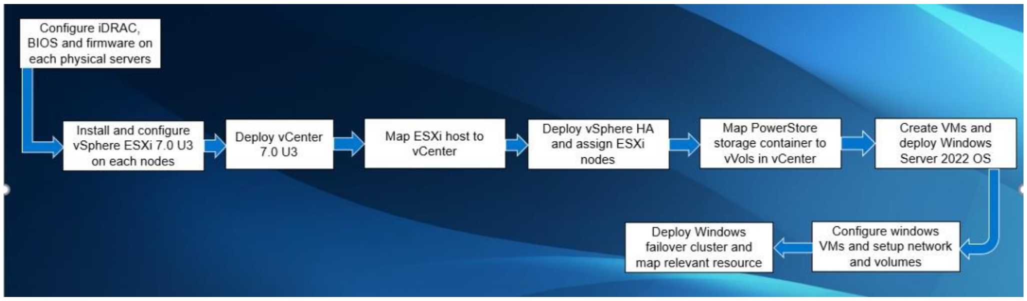

Configure iDRAC, BIOS and firmware on each physical servers
Install and configure vSphere ESXi 7.0 U3 on each nodes
Deploy vCenter 7.0 U3
Map ESXi host to vCenter
Deploy vSphere HA and assign ESXi nodes
Map PowerStore storage container to vVols in vCenter
Create VMs and deploy Windows Server 2022 OS
Configure windows VMs and setup network and volumes
Deploy Windows failover cluster and map relevant resource

Figure 20. VMware environment deployment summary

VMware vSphere

Deploying VMware vSphere 7.x is ubiquitous among system administrators, and there exists many documents detailing the deployment of vSphere. This solution followed the installation guide for VMware vSphere ESXi 7.x on Dell EMC PowerEdge Servers.

VMware vCenter

A preexisting VMware vCenter was used with version 7.0U3 for the setup and validation process. However, Dell engineers have provided a step-by-step guide for deploying VMware vCenter 7.x on Dell servers.

Installing vCenter

1. On a Windows machine or VM, locate the VMware-VCSA installer image.
2. Mount the image, go to the vcsa-ui-installer folder, and double-click *win32*.
3. Double-click *installer.exe* and select *Install*. Accept the terms of the license agreement.
4. Leave the default (vCenter Server with an Embedded Platform Services Controller) selected and click *Next*.
5. Enter the FQDN (Fully Qualified Domain Name) or IP address of the device that will host vCenter.
6. Provide the server's credentials.
7. Configure the environment's network.
8. Select *Finish* when the settings are as wanted.
9. The deployment will complete and go to the introduction page for vCenter.
10. Select time synchronization mode on the *Appliance Configuration* page. Select the SSH access settings. The solution used has a local Network Time Protocol (NTP) server with SSH enabled.

11. Select *Create a New SSO Domain*. Choose and confirm the password. Provide an SSO domain name and SSO site name.
12. Click *Next* on the *CEIP* page. Then click *Finish* on the *Ready to Complete* page.
13. Close the window once installation is complete.
14. Using the vSphere web client, log in to the vCenter server using the credentials that were previously provided.

Creating the data center

1. After logging into VMware vCenter®, go to *Hosts and Clusters*.
2. Select the primary site management vCenter.
3. Right-click the vCenter object and select *New Datacenter*.
4. Enter a name for the new data center.

Adding hosts

1. Right-click the data center and choose *Add Host*.
2. Enter the FQDN or IP address of the PowerEdge R7525 host.
3. Enter the root credentials for the server.
4. Accept the server's certificate.
5. Proceed after reviewing the server details.
6. Assign the license.
7. Disable Lockdown mode.
8. Click *Finish* to complete the process of adding a host.
9. Repeat steps 1 through 8 to add additional hosts to the data center.

vSphere HA Cluster

An empty cluster needs to be created to enable the cluster for vSphere HA. After planning the resources and cluster's networking architecture, use the vSphere Web Client to add hosts to the cluster and specify the vSphere HA settings. Follow the VMware documentation for detailed instructions for configuring the vSphere HA cluster.

Create a VMware vSphere template for Windows Server

The following article from Microsoft details the process of creating a VMware vSphere virtual machine (VM) template for Windows Server.

Create VM from VMware Template

Deploying a virtual machine from a template will create a virtual machine identical to the template. The new virtual machine has the virtual hardware, installed software, and other properties that are configured for the template. Please go to Deploying a Virtual Machine from a Template in the vSphere Web Client, for detailed instructions.

Create Windows Server 2022 Failover Cluster

The process of creating a Windows server failover cluster is documented by Microsoft.

Install and configure SQL Server Always-On Availability croup

The Always-On Availability Groups feature is a high-availability and disaster-recovery solution that provides an enterprise-level alternative to database mirroring. Introduced in SQL Server 2012 (11.x), this feature maximizes the availability of a set of user databases for an enterprise. An availability group supports a failover environment for a discrete set of user databases, known as availability databases, that fail over together. An availability group supports a set of read-write primary databases and one to eight sets of corresponding secondary databases. Optionally, secondary databases can be made available for read-only access and/or some backup operations.

An availability group fails over at the level of an availability replica. Database issues such as datafile loss, database deletion, and transaction log corruption don’t cause failovers. Microsoft documents their SQL Server 2022 Always-On availability group at this site.

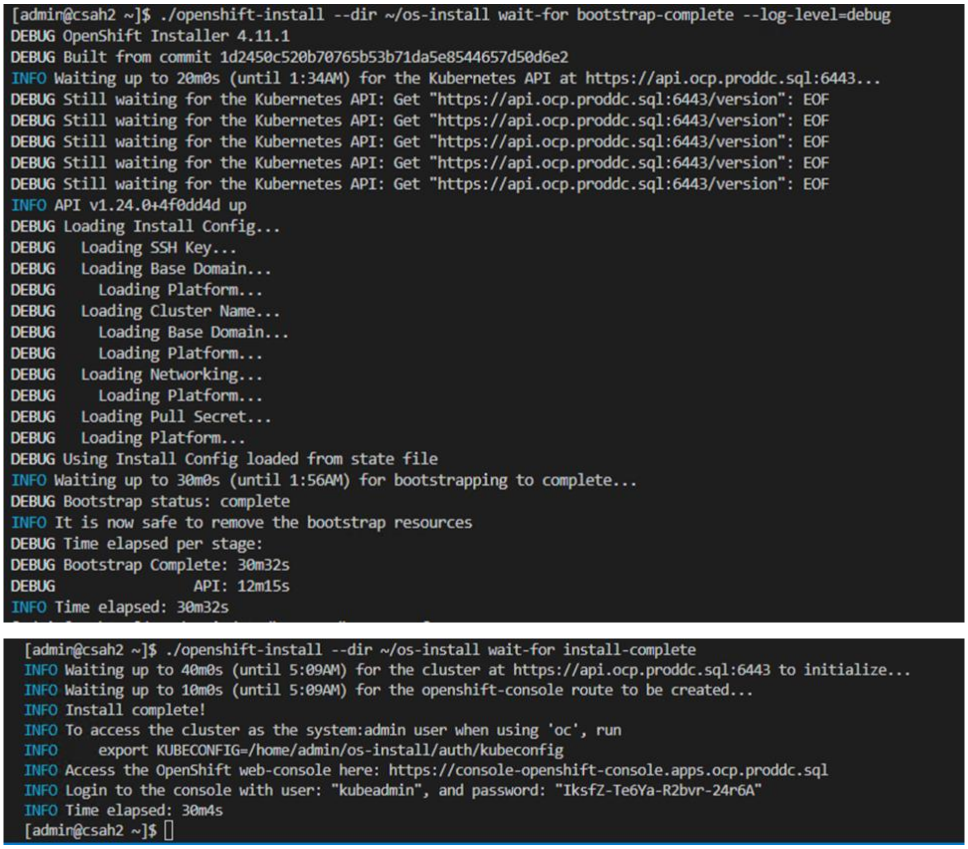
Containerized environment set up

The architecture details a separate environment using RedHat OpenShift 4.11 to host containerized applications and refers to it as the containerized environment.

Environment setup

RedHat Openshift

RedHat provides step by step guidance for the installation and configuration of OpenShift on bare-metal servers.

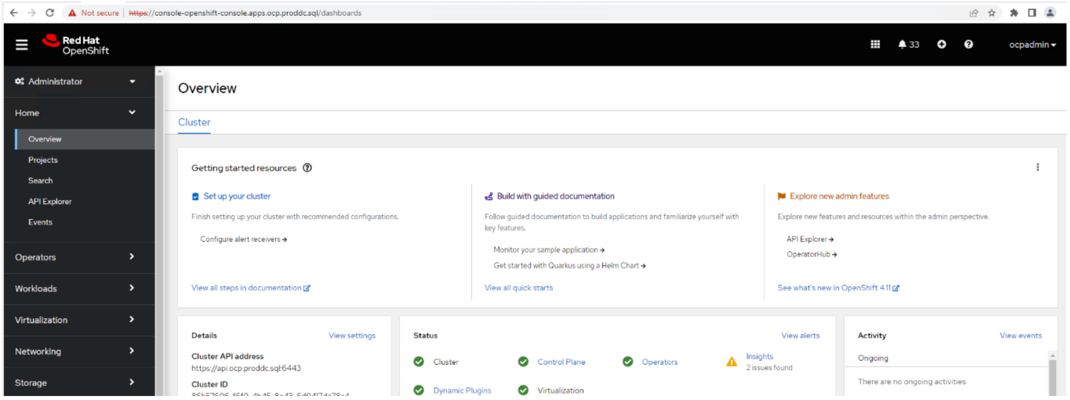


```
[admin@csah2 ~]$ ./openshift-install --dir ~/os-install wait-for bootstrap-complete --log-level=debug
DEBUG OpenShift Installer 4.11.1
DEBUG Built from commit 1d2450c520b70765b53b71da5e8544657d50d6e2
INFO Waiting up to 20m0s (until 1:34AM) for the Kubernetes API at https://api.ocp.proddc.sql:6443...
DEBUG Still waiting for the Kubernetes API: Get "https://api.ocp.proddc.sql:6443/version": EOF
DEBUG Still waiting for the Kubernetes API: Get "https://api.ocp.proddc.sql:6443/version": EOF
DEBUG Still waiting for the Kubernetes API: Get "https://api.ocp.proddc.sql:6443/version": EOF
DEBUG Still waiting for the Kubernetes API: Get "https://api.ocp.proddc.sql:6443/version": EOF
DEBUG Still waiting for the Kubernetes API: Get "https://api.ocp.proddc.sql:6443/version": EOF
INFO API v1.24.0+4f0dd4d up
DEBUG Loading Install Config...
DEBUG   Loading SSH Key...
DEBUG   Loading Base Domain...
DEBUG     Loading Platform...
DEBUG   Loading Cluster Name...
DEBUG     Loading Base Domain...
DEBUG     Loading Platform...
DEBUG   Loading Networking...
DEBUG     Loading Platform...
DEBUG   Loading Pull Secret...
DEBUG   Loading Platform...
DEBUG Using Install Config loaded from state file
INFO Waiting up to 30m0s (until 1:56AM) for bootstrapping to complete...
DEBUG Bootstrap status: complete
INFO It is now safe to remove the bootstrap resources
DEBUG Time elapsed per stage:
DEBUG Bootstrap Complete: 30m32s
DEBUG                API: 12m15s
INFO Time elapsed: 30m32s
```

```
[admin@csah2 ~]$ ./openshift-install --dir ~/os-install wait-for install-complete
INFO Waiting up to 40m0s (until 5:09AM) for the cluster at https://api.ocp.proddc.sql:6443 to initialize...
INFO Waiting up to 10m0s (until 5:09AM) for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
INFO     export KUBECONFIG=/home/admin/os-install/auth/kubeconfig
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.ocp.proddc.sql
INFO Login to the console with user: "kubeadmin", and password: "IksfZ-Te6Ya-R2bvr-24r6A"
INFO Time elapsed: 30m4s
[admin@csah2 ~]$ 
```

Figure 21. RedHat OpenShift deployment summary

With the cluster deployed, it is possible to view cluster information and manipulate the cluster’s settings through RedHat OpenShift web console.



Getting started resources

Set up your cluster

Finish setting up your cluster with recommended configurations.

Configure alert receivers →

View all steps in documentation

Build with guided documentation

Follow guided documentation to build applications and familiarize yourself with key features.

Monitor your sample application →

Get started with Quarkus using a Helm Chart →

View all quick starts

Explore new admin features

Explore new features and resources within the admin perspective.

API Explorer →

OperatorHub →

See what's new in OpenShift 4.11

Details — View settings

Cluster API address
https://api.ocp.proddc.sql:6443

Cluster ID

Status — View alerts

Cluster ✓ · Control Plane ✓ · Operators ✓ · Insights: 2 issues found · Dynamic Plugins ✓ · Virtualization ✓

Activity — View events

Ongoing

There are no ongoing activities.

Figure 22. RedHat OpenShift cluster dashboard

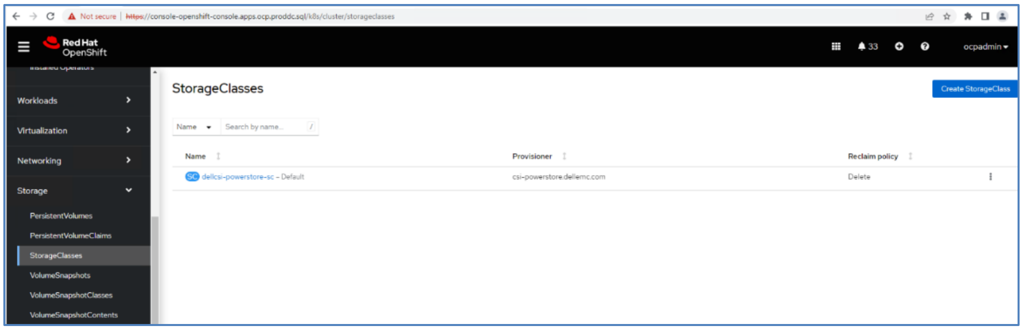
Dell CSI Operator for PowerStore

With the OpenShift cluster up and running, the next step is to deploy the Dell CSI Operator so that the containerized environment can access persistent storage.

The Dell CSI Operator is a Kubernetes Operator, which is used to install and manage the CSI Drivers provided by Dell for various storage platforms. This operator is available as a community operator for upstream Kubernetes deployable through OperatorHub.io. It is also available as a certified operator for OpenShift clusters and can be deployed using the OpenShift Container Platform. Both these methods of installation use the OLM (Operator Lifecycle Manager). The operator can also be deployed manually.

The process of installing the Dell CSI operator for Dell PowerStore is captured in detail on the GitHub page for Installing CSI Driver for PowerStore via Operator.

After the successful deployment of the CSI Operator/Plugin for PowerStore on the OpenShift Cluster, a storage class with the name “dellemc-powerstore-sc” is configured within the OpenShift cluster. The name can be changed during the deployment process.



StorageClasses

Name	Provisioner	Reclaim policy
SC dellcsi-powerstore-sc – Default	csi-powerstore.dellemc.com	Delete

Figure 23. StorageClass in an OpenShift Cluster

The storage class will be used to create persistent volumes (PV) and persistent volume claims (PVC) which can be mapped to containers and pods to provide persistent storage to the application. Following is a reference for the storage class in OpenShift cluster.

SQL Server 2022 deployment

Deploying SQL Server 2022 in an OpenShift cluster is simple. The SQL Server 2022 image was modified to include PolyBase for the testing of potential use cases. PolyBase is not available by default and is used in this solution to enable data virtualization.

PolyBase Container Images

PolyBase is not currently installed and enabled by default in the published container image **mcr.microsoft.com/mssql/server:2022-latest,** so Dell engineers created a custom image with PolyBase installed. The following script is used to create a custom container image for SQL Server 2022.

```
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
```

Use the following command to build the docker container image:

```
docker build -t sql-custom-2022:latest
```

Change the image artifacts using the tag command.

```
docker tag sql-custom-2022:latest docker.io/sanran/sql-custom 2022:latest
```

Then push the newly created container image to docker’s public container registry.

```
docker push sanran/sql-custom-2022:latest
```

In this command, **sanran** refers to a username on the dockerhub.io public container registry. This container image can be pulled to run SQL Server 2022 with PolyBase pre-installed on any setup. More details about using SQL Server with PolyBase can be found in the Use Case section.

Mapping storage certificates with OpenShift

The certificate must be mapped so that the Kubernetes cluster can communicate with the Elastic Cloud Storage deployment. This can be done using the Kubernetes ConfigMap, which maps the certificate to a SQL container.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: rootca-cert
  namespace: demo
data:
  rootCA.crt: |
    -----BEGIN CERTIFICATE-----
    MIID9zCCAt+gAwIBAgIUKeay3nw4sQe8QHbIlPr8q4vK7AMwDQYJKoZIhvcNAQEL
    BQAwgYoxCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJNQTESMBAGA1UEBwwJSG9
wa2lu
    dG9uMQ0wCwYDVQQKDAREZWxsMQ8wDQYDVQQLDAZCaXpBcHAxGDAWBgNV
BAMMD0hB
    .
    ----  certificate content -----
    .
    4/XHKa9DXk8g9pBvSpF7HC1DJRee7ZnJ6p4Vme9LyXjmA+OBzaovOU4i54iKwvkl
    qXXlevx+E0KLy1QbhL9n49F2zi0SZRsjJ5+A5+gCz9CKhLL9b7VvsYqg/Ok40ZT8
    to7ahH31PTtnrzM=
    -----END CERTIFICATE-----
```

There is another way to deploy a ConfigMap within an OpenShift environment. When creating a ConfigMap this way, rootCA.crt is the certificate file that will be used.

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

Persistent volume claim creation

A PersistentVolumeClaim (PVC) is a request for storage by a user. It is like a Pod, as Pods consume node resources and PVCs consume PV resources. Pods can request specific resource amounts (CPU and Memory). Claims can request specific amounts of storage and can be mounted with ReadWriteOnce, ReadOnlyMany or ReadWriteMany access modes.

To provide persistent storage to a container or pod, a persistent volume claim is created using the Dell CSI storage class. The following script provides an example of how to create a PVC that will be used by the SQL Server pods. This pvc yaml script creates a 500 GB volume in ReadWriteOnce access mode.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: new-sql-pvc-31445
  namespace: demo
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 500Gi
  storageClassName: dellcsi-powerstore-sc
```

SQL Server 2022 secret creation

A Secret is an object that contains a small amount of sensitive data like a password, token, or key which might otherwise be put in a Pod specification or container image.

Secrets can be created independently of the Pods that use them, reducing the risk of the secrets data being exposed during the workflow of creating, viewing, and editing Pods.

To provide the SA password within the deployment file for the containers, a secret was created in OpenShift’s cluster environment using a base 64 encryption for the password.

```
apiVersion: v1
kind: Secret
metadata:
  name: mssql-secret
  namespace: demo
data:
  MSSQL_SA_PASSWORD: QFZhbnRhZ2U0
type: Opaque
```

SQL server 2022 deployment

The deployment section provides a declarative update pattern for pods and replica sets. A preferred state is described, and the deployment controller changes the actual state to match the preferred state at a controlled rate. This allows high availability to be maintained, and it is possible to specify when resources should be destroyed, allowing cluster updates to occur without any loss of service in some cases.

This yaml script is used to deploy SQL Server 2022 on a Kubernetes cluster.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: new-sql-deployment
  namespace: demo
spec:
  replicas: 1
  selector:
     matchLabels:
       app: mssql-31445
  template:
    metadata:
      labels:
        app: mssql-31445
    spec:
      terminationGracePeriodSeconds: 30
      hostname: mssqlinst
      securityContext:
        fsGroup: 10001
      containers:
      - name: mssql
```

```
        image: docker.io/sanran/sql-custom-2022:latest
        resources:
          requests:
            memory: "128G"
            cpu: "12000m"
          limits:
            memory: "128G"
            cpu: "12000m"
        ports:
        - containerPort: 31445
        env:
        - name: MSSQL_PID
          value: "Developer"
        - name: ACCEPT_EULA
          value: "Y"
        - name: MSSQL_SA_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mssql-secret
              key: MSSQL_SA_PASSWORD
        volumeMounts:
        - name: mssqldb
          mountPath: /var/opt/mssql
        - name: rootca-cert
          mountPath: /var/opt/mssql/security/ca-certificates/rootCA.crt
          subPath: rootCA.crt
          readOnly: true
      volumes:
      - name: mssqldb
        persistentVolumeClaim:
          claimName: new-sql-pvc-31445
      - name: rootca-cert
        ConfigMap:
          name: rootca-cert
```

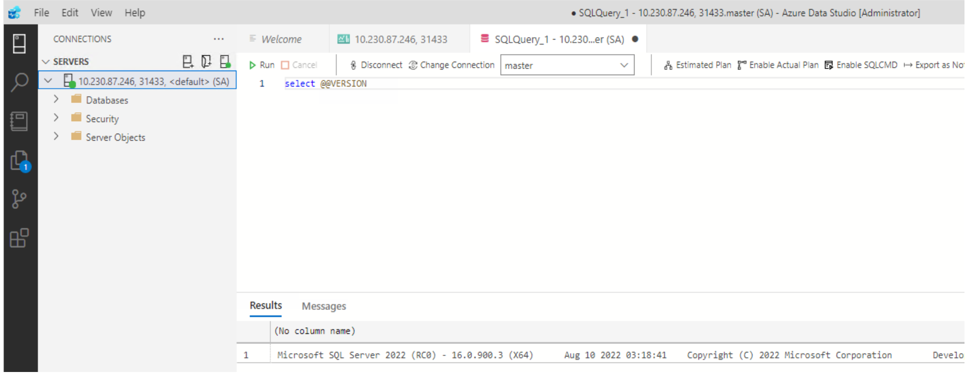
Service creation

Services are an abstract way to expose an application running on a set of Pods as a network service. With OpenShift and Kubernetes there is no need to modify the application to use an unfamiliar service discovery mechanism. Pods are given IP addresses by Kubernetes. Pods can be load balanced and can share a single DNS.

A Service is an abstraction which defines a logical set of Pods and their access policy. The set of Pods targeted by a Service is usually determined by a selector (in this case it is “mssql-31445”). Specific ports can be specified for container port mapping.

```
kind: Service
apiVersion: v1
metadata:
  name: mssql-service-31445
  namespace: demo
spec:
  type: NodePort
  ports:
    - port: 1433
      targetPort: 1433
      nodePort: 31445
  selector:
    app: mssql-31445
```

There are several types of services that can be created such as ClusterIP, NodePort, LoadBalancer, and ExternalName. This example creates a service of type **NodePort**.



```
select @@VERSION
```

```
   (No column name)
1  Microsoft SQL Server 2022 (RC0) - 16.0.900.3 (X64)    Aug 10 2022 03:18:41    Copyright (C) 2022 Microsoft Corporation    Develo
```

More details on the service and service types can be found on the Kubernetes website.

Figure 24. SQL Server 2022 connection using Azure Data Studio

Once the SQL Server pods are running successfully, these SQL Server pods can be connected to just like any SQL Server instance. The above screenshot shows a connection to a SQL Server 2022 instance running in an OpenShift cluster environment.

Many different tools can be used to connect to a SQL Server instance that is running in a pod. SQL Server Management Studio, PowerShell, Visual Studio, third-party SQL monitoring or development tools and other platforms will connect to the container instance as intended.

Chapter 4. Use cases

This chapter presents the following topics:

Introduction

SQL Server 2022 introduces features that enable easier access to external data on object storage. The BACKUP and RESTORE functions work directly with object storage. PolyBase enables easy access to ECS data through OPENROWSET and CETAS, allowing data engineers to use data from different storage locations. The new T-SQL functions offer helpful use cases for data analytics which generate value from object storage data. Automation is beneficial for deploying containers that use S3 protocol for data virtualization. Automation can create SQL Server 2022 engines and provision access to the appropriate data, which makes establishing data virtualization environments simple and reproducible.

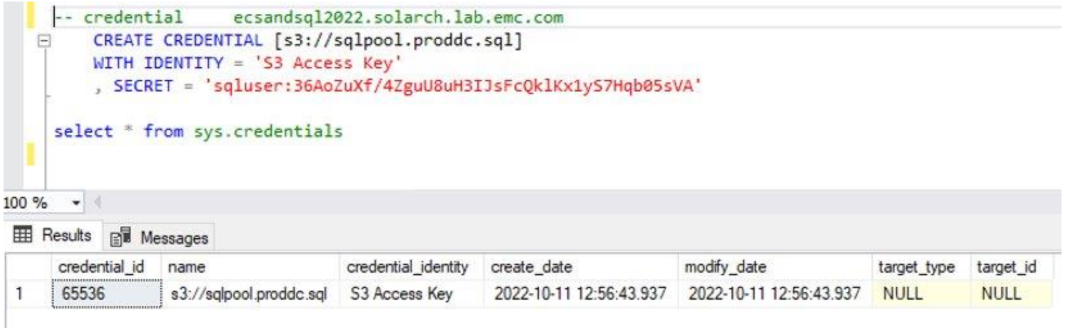
Backup and restore use case

T-SQL has been used to run backup and restore operations long before Microsoft SQL Server 2022. The newly introduced features allow BACKUP and RESTORE to have their locations set to external object storage which increases the reliability of backups by storing them across multiple fault domains while still maintaining accessibility. Database Administrators no longer need to be concerned about running out of storage capacity either, because Dell's Elastic Cloud Storage can be scaled to meet their storage needs. Dell's ECS is secure, and it requires the appropriate credentials to perform backup or restore operations for Microsoft SQL Server 2022 through the S3 protocol. This use case was validated by simulating how a backup and restore procedure would work using the S3 protocol and Dell ECS.

Credentials setup

The credentials used for connecting with S3 compatible storage require an appropriate storage URL and user credentials. Credentials that are created can be observed in the sys.credentials table. The following is an example of the process of creating credentials:

```
CREATE CREDENTIAL [s3://<ECS Bucket URL>]
WITH IDENTITY = ‘S3 Access Key’,
SECRET = ‘sqluser:<Username>/<Password>’;
```



```
-- 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
```

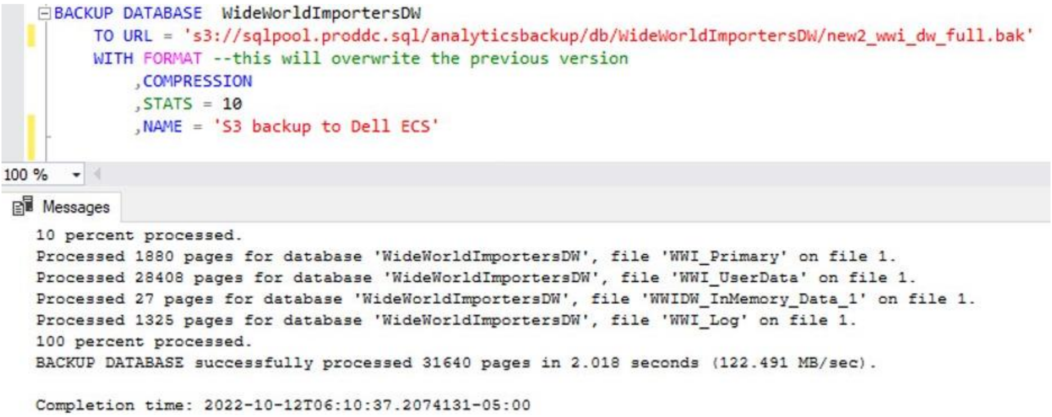
	credential_id	name	credential_identity	create_date	modify_date	target_type	target_id
1	65536	s3://sqlpool.proddc.sql	S3 Access Key	2022-10-11 12:56:43.937	2022-10-11 12:56:43.937	NULL	NULL

Backing up a Database to S3 Compatible Storage

Microsoft SQL Server 2022 allows users to directly store their backups in external object storage supported by the S3 protocol. A single query is all that is needed to back up the database to Dell's Elastic Cloud Storage once the credentials have been confirmed. The query looks as follows:

```
BACKUP DATABASE <DatabaseName>
   TO URL = ‘s3://<ECS Bucket URL>/<FolderName>/<Filename>
   WITH FORMAT, COMPRESSION, STATS = 10,
                 NAME = ‘S3 backup to Dell ECS’;
```

The following snippet shows the process of backing up a database into Dell’s Elastic Cloud Storage using S3 and SQL:



```
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
```

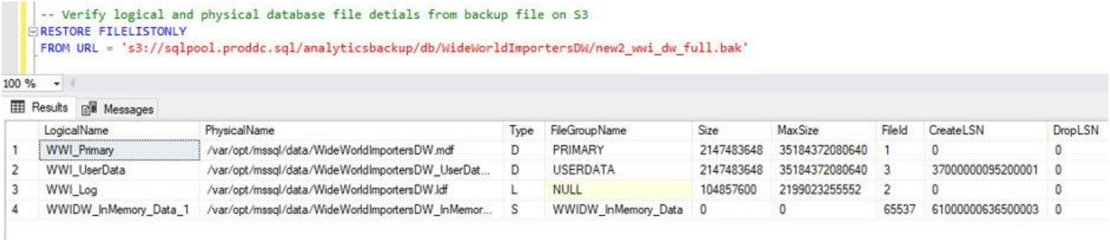
Restoring a Database from S3 compatible storage

It is also possible to restore a database directly from a backup file located in external S3 compatible storage without copying the data locally. A single query can specify the location of the database file to restore and the destination for the restored database.

The RESTORE FILELISTONLY command can be used to verify that the backup files are available in object storage and the database admin can verify the sanctity of the backup file along with its metadata. This query can be used as follows:

```
RESTORE FILELISTONLY
FROM URL = ‘s3://<ECS Bucket URL>/<FolderName>/<Filename>;
```

The following image shows the results of running this query in Microsoft SQL Server 2022:



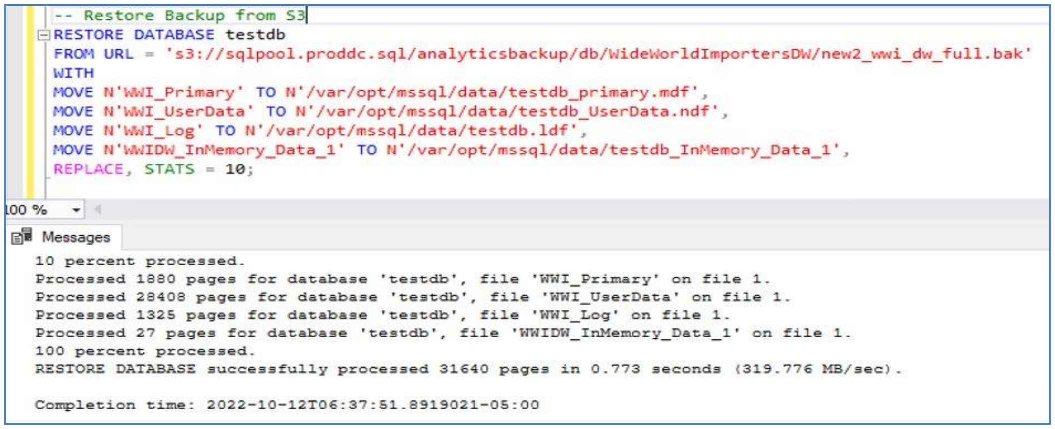
```
-- Verify logical and physical database file detials from backup file on S3
RESTORE FILELISTONLY
FROM URL = 's3://sqlpool.proddc.sql/analyticsbackup/db/WideWorldImportersDW/new2_wwi_dw_full.bak'
```

	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

Use the following query to restore the database from a file located in object storage:

```
RESTORE DATABASE <DatabaseName>
FROM URL = ‘s3://<ECS Bucket URL>/<FolderName>/<Filename>
WITH
MOVE N’<LogicalName1>’ TO N’/var/opt/mssql/data/<Destination FileName1>,
MOVE N’<LogicalName2>’ TO N’/var/opt/mssql/data/<Destination FileName2>,
MOVE N’<LogicalName3>’ TO N’/var/opt/mssql/data/<Destination FileName3>,
REPLACE, STATS = 10;
```

Database restoration with a backup file in ECS is demonstrated below:



```
-- 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
```

Data virtualization use case

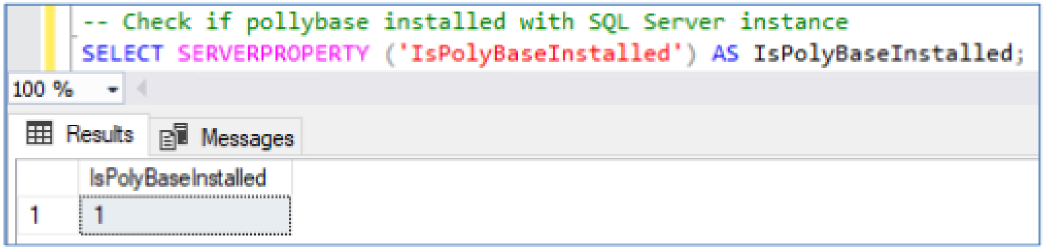
Data virtualization does not require information about the data's format and storage location which revolutionizes data retrieval and manipulation. This allows data integration to occur without needing to copy or move data that is stored in a separate location. A single virtual layer can span multiple storage formats and physical locations.

Accessing external data

Microsoft has added object storage access to the already existing PolyBase features in SQL Server 2022. This allows the use of data virtualization to take full advantage of the benefits of object storage. This new introduction allows for a SQL server instance to directly query data from SQL server and outside data sources. It even permits T-SQL to combine local and external data. This feature is not enabled by default, and it must be installed and enabled on the server instance to be used.

PolyBase

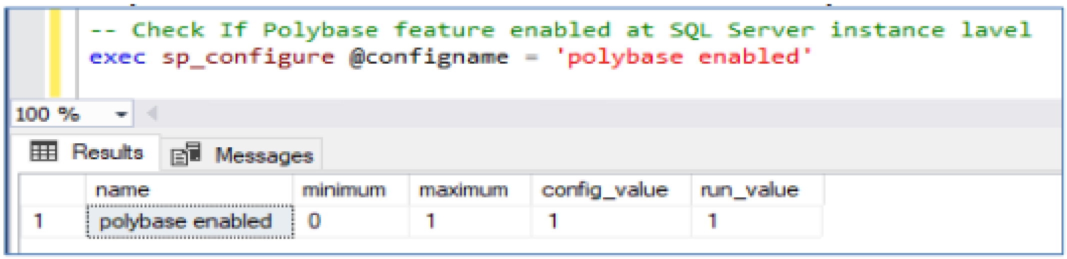
PolyBase's installation status is checked with the following query:



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

	IsPolyBaseInstalled
1	1

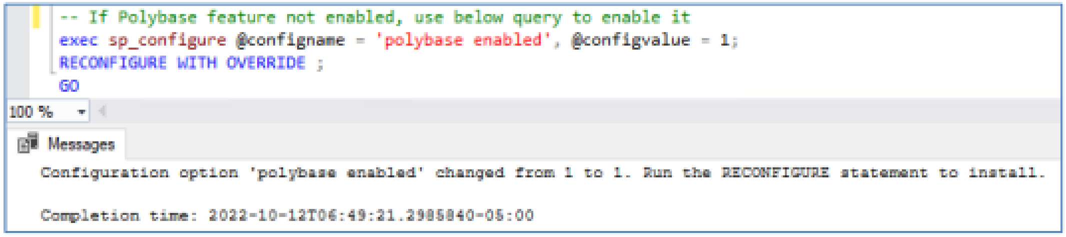
The following stored procedure is run to enable PolyBase:



```
-- 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

Run the following stored procedure to verify that PolyBase is enabled:



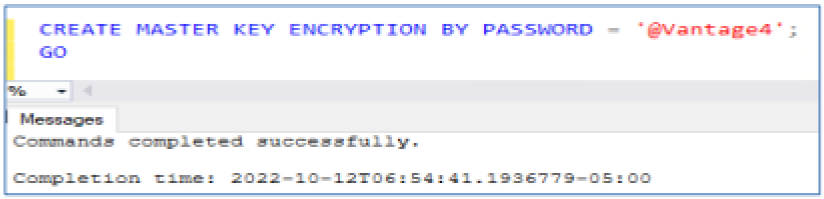
```
-- If Polybase feature not enabled, use below query to enable it
exec sp_configure @configname = 'polybase enabled', @configvalue = 1;
RECONFIGURE WITH OVERRIDE ;
GO
```

```
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
```

Encryption and communication setup

To access an external data source, an encryption key is used to verify that communication between the external data source and the SQL Server instance is secure. Dell Elastic Cloud Storage is used as the external data source in this example.

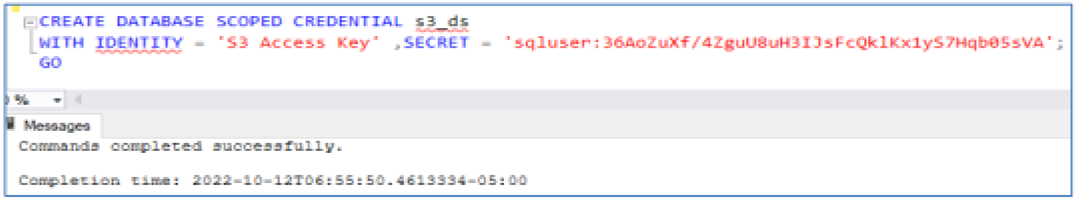
1. Create the encryption key in the SQL Server instance.



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

```
Commands completed successfully.

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

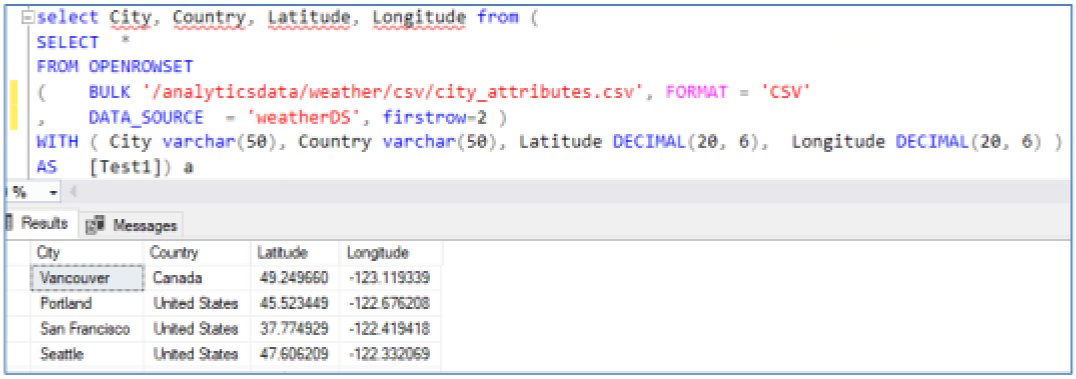
2. Create database scope credentials within the database.



```
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
```

3. Establish a connection with the external data source by pointing to the S3 storage URL and passing the database scoped credentials.



```
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

4. Use OPENROWSET to validate that the data located in ECS is reachable.

Creating external tables through the use of select queries

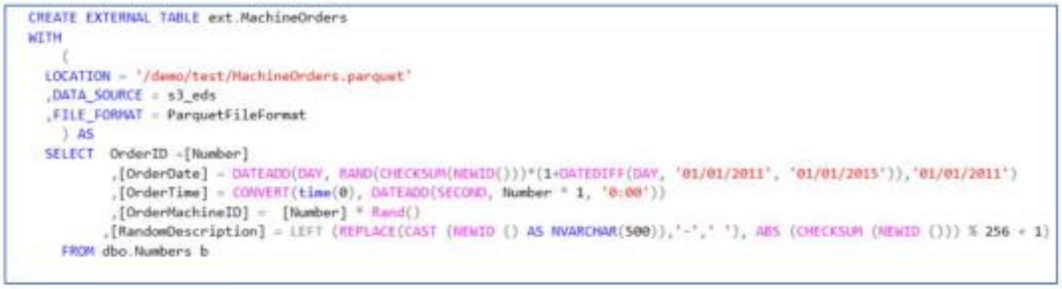
CREATE EXTERNAL TABLE AS SELECT (CETAS) allows for the creation of external tables using data that is not present on a local Microsoft SQL Server 2022 instance. Data virtualization is valuable for enterprises engaging with large volumes of data as a part of their daily operations and CETAS makes this process easy. Data virtualization allows all types of data to be accessed from one place which hides the underlying complexity of heterogeneous data and gives data specialists direct data access. Data virtualization requires no extra infrastructure for data and allows new applications to be integrated with existing infrastructure which eliminates wasteful data silos and reduces costs.

Data virtualization with parquet files

Parquet is an open-source datafile format created by Apache that is designed for efficient data storage and retrieval. The data in a Parquet file is stored in columnar format and uses efficient data compression and encoding schemes. Parquet files provide the enhanced performance needed for handling complex data at scale. These files were designed by Apache to be a common interchange format for both interactive and batch workloads.

Traditional database engines such as SQL Server store data in row-based comma separated files, however in SQL Server 2022, T-SQL queries enable the conversion of csv files to Parquet using CREATE EXTERNAL TABLE AS SELECT (CETAS) with the OPENROWSET syntax. Using these features, it is possible to join relational data in SQL with efficiently stored non-relational data located in object storages such as Dell’s Elastic Cloud Storage. It is possible to use CETAS queries to create external datasets in Parquet file format without ever landing the data within the SQL server instance. The following is an example of creating an external dataset with CETAS.

It is also possible to convert table data to Parquet through CETAS. The following is an example this command:



```
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
```

```
CREATE CREDENTIAL [s3://<ECS Bucket URL>]
WITH IDENTITY = ‘S3 Access Key’,
SECRET = ‘sqluser:<Username>/<Password>’;
```

CSV data can be converted to Parquet using CETAS and OPENROWSET together. Here is a command that accomplishes this:

```
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
      ) AS A;
```

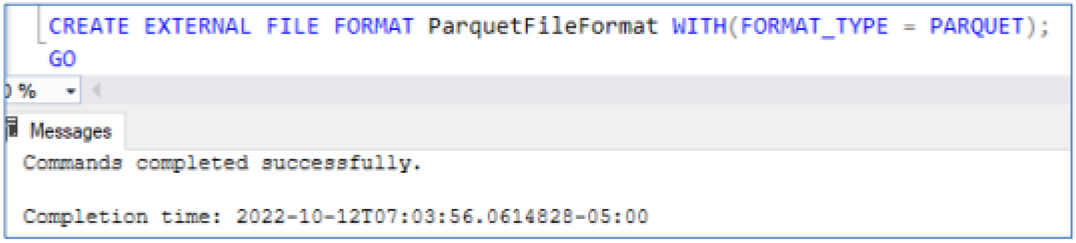
PySpark can also be used to convert the data from CSV to Parquet as shown below:

```
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")
```

These different methods of converting data to and from Parquet files serve different purposes. The OPENROWSET use case works well for small batches of data that can quickly run and for scenarios where it is not necessary to have immediate access to the database. Large workloads can take many hours to run and would use the resources required for regular database operations to convert data to a different format. In these cases, it is more practical to rely on a platform such as PySpark to handle the data conversion without utilizing resources meant for analytical processes.

Accessing external data

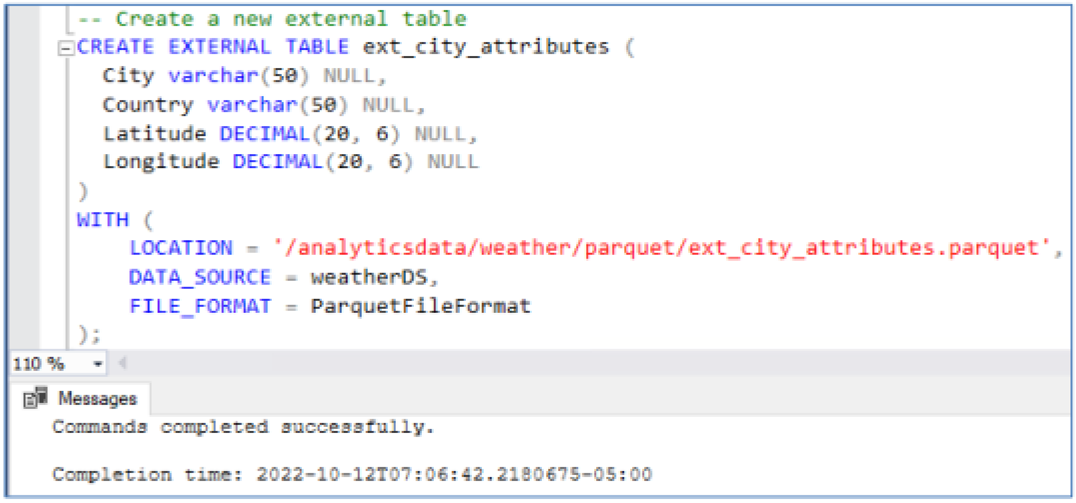
External tables simplify working with data outside of SQL Server, using PolyBase to access the external data. When creating external tables, information about the file format is required along with the data source, and location of the files. An external file format must be specified in SQL Server:



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

```
Commands completed successfully.

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

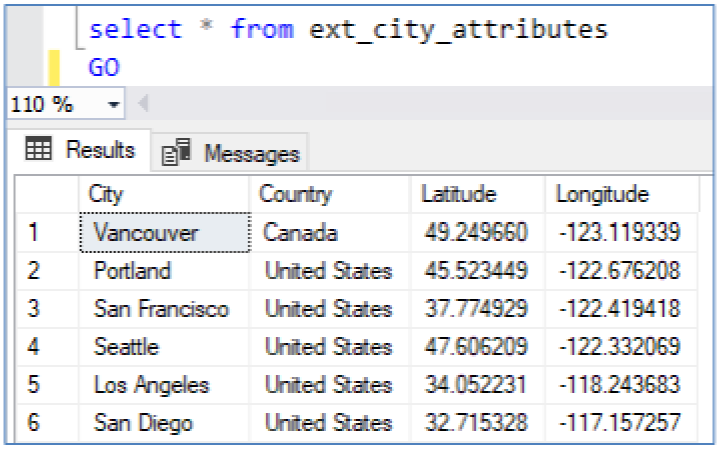
Users can create an external table from the data on S3 compatible storage by providing the file location, the data source and file format as shown in the following screenshots.



```
-- 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
```

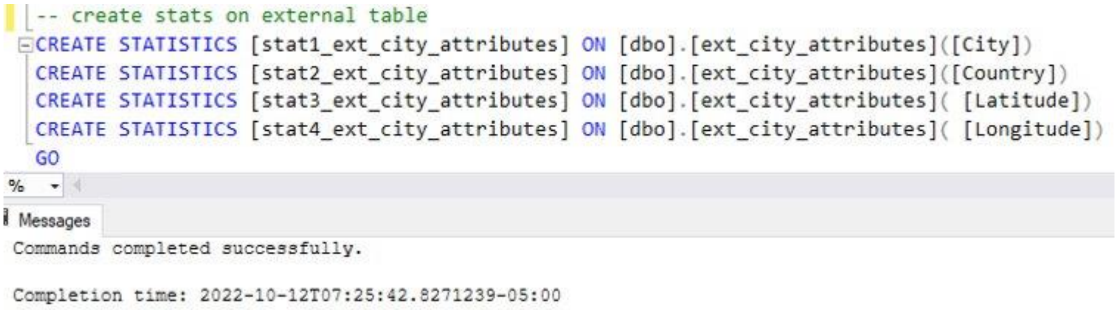
Once the external table has been created, data can be queried.



```
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

It is not currently possible to have multicolumn statistics for external data. It is possible to create statistics for singular columns of external tables which can improve performance.



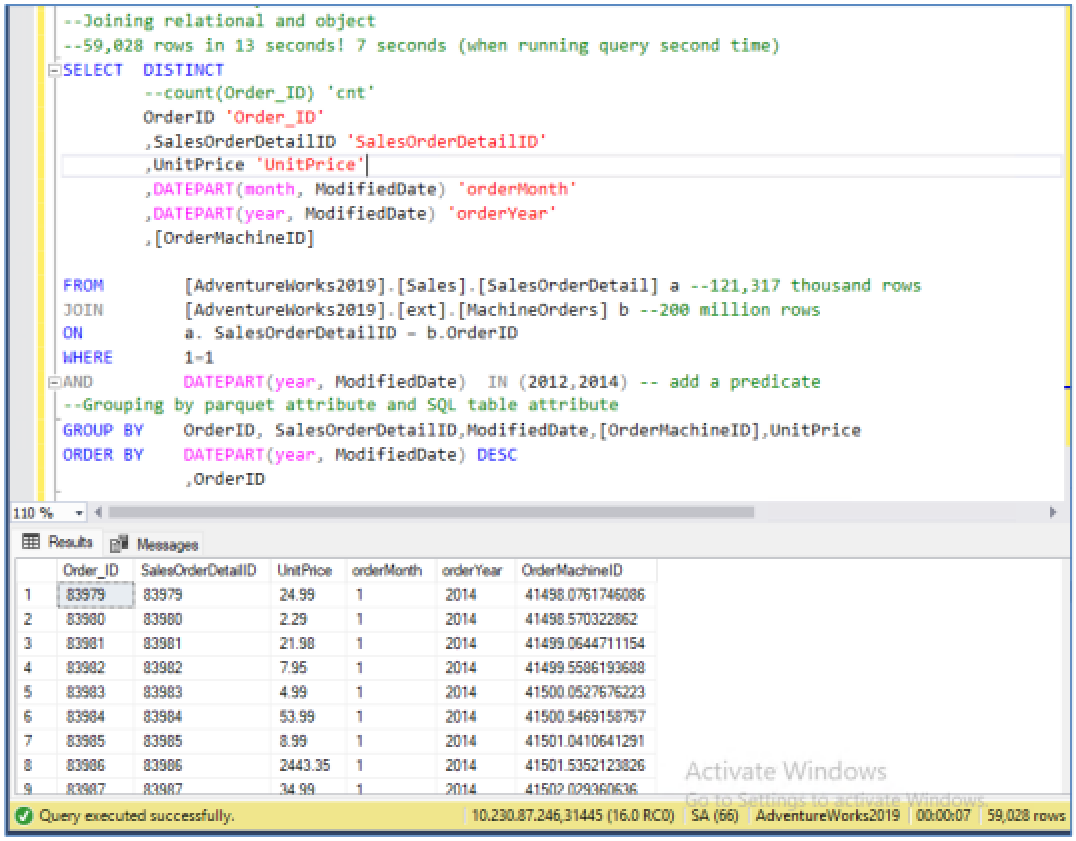
```
-- create stats on external table
CREATE STATISTICS [stat1_ext_city_attributes] ON [dbo].[ext_city_attributes]([City])
CREATE STATISTICS [stat2_ext_city_attributes] ON [dbo].[ext_city_attributes]([Country])
CREATE STATISTICS [stat3_ext_city_attributes] ON [dbo].[ext_city_attributes]( [Latitude])
CREATE STATISTICS [stat4_ext_city_attributes] ON [dbo].[ext_city_attributes]( [Longitude])
GO
```

```
Commands completed successfully.

Completion time: 2022-10-12T07:25:42.8271239-05:00
```

Creating statistics is costly and it is not always practical or necessary to have statistics for all a table’s columns. It is more useful to simply have statistics for fields such as City and Country which would likely need to be accessed more frequently than the other table attributes. This is akin to the rational used for creating indexes. This allows for a more effective use of system resources. It is important to consider cardinality when creating statistics for databases as depending on the cardinality the statistic may provide no benefits for queries that attempt to use it. This is especially true with large volumes of externally stored data where statistics must be manually generated by the user.

Data virtualization testing details

The Dell solutions engineering team conducted tests and performed research to determine the feasibility of reading hundreds of millions of records from ECS. This is was an extremely important consideration when evaluating the practical use of external data with Microsoft SQL Server 2022 for data analytics. To validate this, the relational data present in the SalesOrderDetail table from Microsoft’s AdventureWorks database was joined with an external table. The SalesOrderDetail table had 121,000 records and the external table backed up by Parquet files on Dell ECS contained 200 million records. The following is an example of such a query that demonstrates the join completing within seconds.

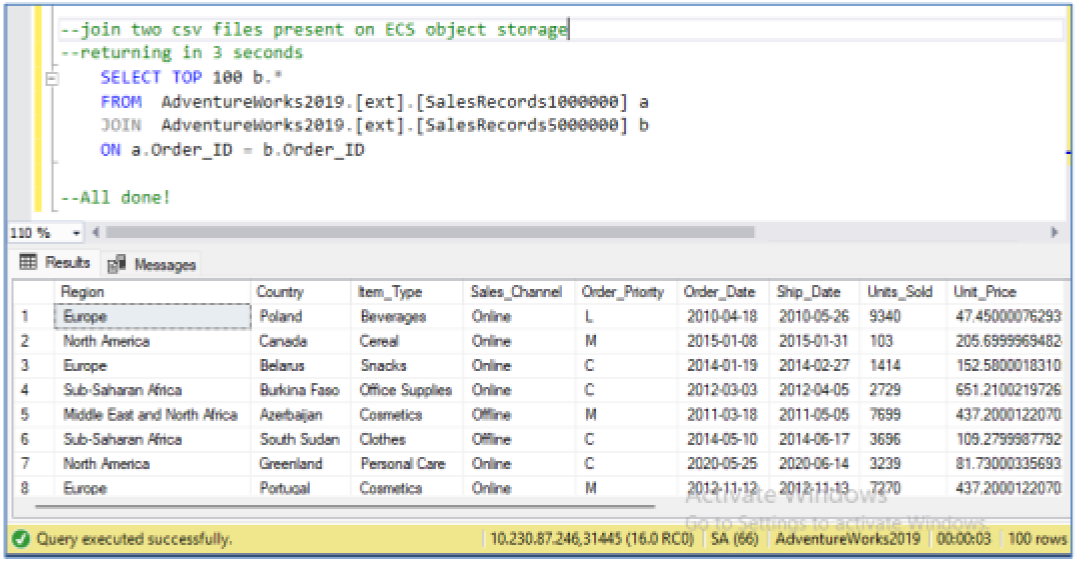


```
--Joining relational and object
--59,028 rows in 13 seconds! 7 seconds (when running query second time)
SELECT  DISTINCT
        --count(Order_ID) 'cnt'
        OrderID 'Order_ID'
        ,SalesOrderDetailID 'SalesOrderDetailID'
        ,UnitPrice 'UnitPrice'
        ,DATEPART(month, ModifiedDate)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

Dell solutions engineers also used Microsoft SQL Server to query data that resided externally on ECS. Two external tables backed up in CSV files were joined. These tables contained one and five million records.



```
--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

This allows for semi-structured and unstructured data to be kept externally from SQL Server in a more affordable object storage such as Dell ECS without hindering performance. Accessing data this way creates countless opportunities for performing ETL/ELT workflows and offloading the ETL/ELT process altogether.

Data tiering use case

Data tiering is another beneficial solution architecture that is offered by Dell. Transferring larger read-only fact tables out of the RDBMS system and into S3 compatible ECS can decrease the total duration and size of SQL Server backups. The SQL datafile storage footprint is also reduced by storing data externally in Dell Elastic Cloud Storage. By converting table data to Parquet format additional storage efficiencies are achieved. A simple T-SQL CETAS statement can be used to export data from the RDBMS to Parquet file format.

A Parquet file is a columnar storage format that performs exceptionally well when used as an external table for a SQL Server database. This usually results in reduced process time for aggregation queries compared to the row-oriented storage SQL Server traditionally uses.

New T-SQL functions use cases

Overview

Microsoft SQL Server 2022 has introduced several new Transact-SQL (T-SQL) queries that can be used to gather additional analytical information. The functions tested by Dell’s software engineers in queries with internal and external data include GREATEST, LEAST, FIRST_VALUE, LAST_VALUE, STRING_SPLIT, GENERATE_SERIES, and DATE_BUCKET. These queries, which were added to Microsoft SQL Server 2022 allow for additional information to be gathered from data stored in a Dell Elastic Cloud Storage data lake. These queries were tested on data stored in Parquet files to simulate a realistic scenario. The data used for these queries comes from the US Weather dataset, the US AQI dataset, and Microsoft’s Wide World Importers dataset.

DATE_BUCKET Function

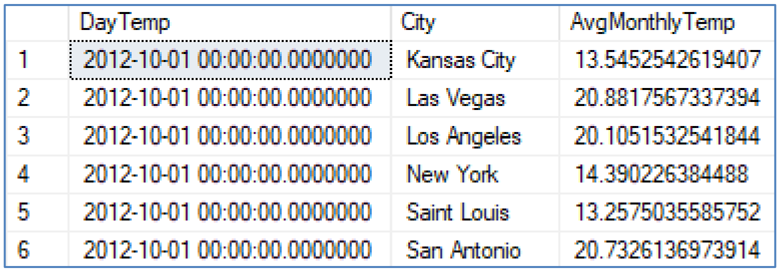
Function overview

Date buckets allow for data to be grouped according to ranges of dates which is especially useful. This query allows companies to analyze monthly profitability patterns. The DATE_BUCKET function supports parameters that allow for different increments of time to be selected, including hourly, monthly, and yearly. It is possible to allow for periods of several weeks or years to fall into the same bucket by modify the unit of time’s scalar. The start and end dates of the range can be specified, allowing for business data to be precisely analyzed.

Example Monthly DATE_BUCKET Query

```
SELECT DATE_BUCKET(MONTH, 1, DateOf) AS DayTemp, City,
        AVG(Temperature) AS AvgMonthlyTemp
FROM Temperature_View
GROUP BY DATE_BUCKET(MONTH, 1, DateOf), City
ORDER BY DayTemp;
```

Results



	DayTemp	City	AvgMonthlyTemp
1	2012-10-01 00:00:00.0000000	Kansas City	13.5452542619407
2	2012-10-01 00:00:00.0000000	Las Vegas	20.8817567337394
3	2012-10-01 00:00:00.0000000	Los Angeles	20.1051532541844
4	2012-10-01 00:00:00.0000000	New York	14.390226384488
5	2012-10-01 00:00:00.0000000	Saint Louis	13.2575035585752
6	2012-10-01 00:00:00.0000000	San Antonio	20.7326136973914

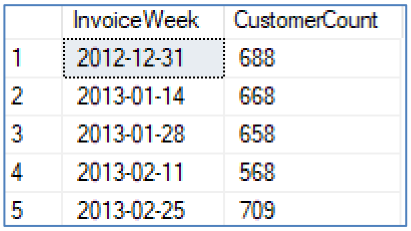
Query description

This query returns the average temperature for each city every month beginning on the first day of the month. It groups all the values that occur within the range of a month and then these groups can be aggregate.

Example Biweekly DATE_BUCKET Query

```
SELECT DATE_BUCKET(WEEK, 2, InvoiceDate) AS InvoiceWeek,
            COUNT(CustomerId) AS CustomerCount
FROM WideWorldImporters.Sales.Invoices
GROUP BY DATE_BUCKET(WEEK, 2, InvoiceDate)
ORDER BY InvoiceWeek;
```

Results



	InvoiceWeek	CustomerCount
1	2012-12-31	688
2	2013-01-14	668
3	2013-01-28	658
4	2013-02-11	568
5	2013-02-25	709

Query description

This query returns the biweekly counts of customers for each invoice. It demonstrates how custom time periods can be specified by users to acquire the preferred information.

GREATEST and LEAST Functions

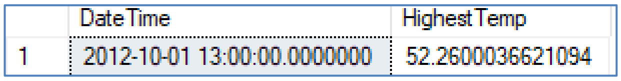
Function Overview

The MIN and MAX functions do not always permit the retrieval of data in the preferred format. The GREATEST and LEAST T-SQL functions introduced for SQL 2022 allow for the retrieval of the maximum or minimum value of a row, respectively. This is useful for data analysts who could previously only find these values for columns. These functions can be used in SQL stored procedures and functions to select specific parameters.

Example GREATEST Query

```
SELECT DateTime, 32 + 9/5 * (
            GREATEST([San_Antonio], [San_Diego], [San_Francisco], [Kansas_City], [Saint_Louis],
                        [Las_Vegas], [Los_Angeles], [New_York], Seattle)-273.15) AS HighestTemp
FROM Temperature
WHERE DateTime = '2012-10-01 13:00:00:000';
```

Results



	DateTime	HighestTemp
1	2012-10-01 13:00:00.0000000	52.2600036621094

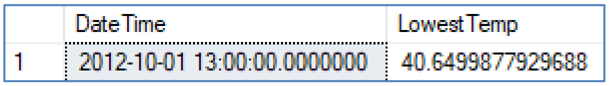
Query description

This query selects the maximum temperature value from a list of columns for the row entry at a specified DateTime which is useful for perform comparisons between the values in a row. The relevant result can then be added as a new column in the resulting table.

Example LEAST Query

```
SELECT DateTime, 32 + 9/5 * (
        LEAST([San_Antonio], [San_Diego], [San_Francisco], [Kansas_City], [Saint_Louis],
                [Las_Vegas], [Los_Angeles], [New_York], Seattle)-273.15) AS LowestTemp
FROM Temperature
WHERE DateTime = '2012-10-01 13:00:00:000';
```

Results



	DateTime	LowestTemp
1	2012-10-01 13:00:00.0000000	40.6499877929688

Query description

This query selects the minimum temperature from a list of columns for each row entry at a specified DateTime.

GENERATE_SERIES Function

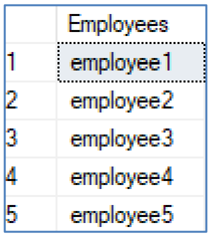
Function overview

GENERATE_SERIES is a function that allows for the creation of a series of numeric values. This is helpful for querying specific values or creating new data through joins. It can be used to create completely new data through string concatenation or mathematical operations. It is a powerful function for transforming data and has many use cases.

New String Creation with GENERATE_SERIES

```
WITH Series AS (
  SELECT value AS SeriesId FROM GENERATE_SERIES(1,1000)
)
SELECT TOP 5 'employee'+(CONVERT (varchar (10), SeriesId)) AS Employees
FROM Series;
```

Results



	Employees
1	employee1
2	employee2
3	employee3
4	employee4
5	employee5

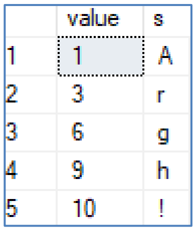
Query description

This example creates a series using the GENERATE_SERIES function and then creates a new table which is populated with numbered employee strings. It demonstrates one of the ways that the GENERATE_SERIES function can create new data.

Finding the First Occurrence of a Repeated Character Sequence Using GENERATE_SERIES

```
DECLARE @S VARCHAR(8000) = 'Aarrrgggh!';
SELECT value, S
FROM (
    SELECT value=1, S=LEFT (@S, 1)
    UNION ALL
    SELECT value,
    CASE
    WHEN SUBSTRING(@S, value - 1, 1) <> SUBSTRING(@S, value, 1)
      THEN SUBSTRING(@S, value, 1)
      END
    FROM GENERATE_SERIES(1, 100)
    WHERE value BETWEEN 2 AND LEN(@S)
    ) AS A
WHERE S IS NOT NULL;
```

Results



	value	s
1	1	A
2	3	r
3	6	g
4	9	h
5	10	!

Query Description

GENERATE_SERIES can be used to provide additional information regarding the indexing of values. In this case, the string’s repeat character sequences are associated with their ordinal position and the first occurrence of the character in a sequence of identical characters has its index retained to condense string information.

STRING_SPLIT Function

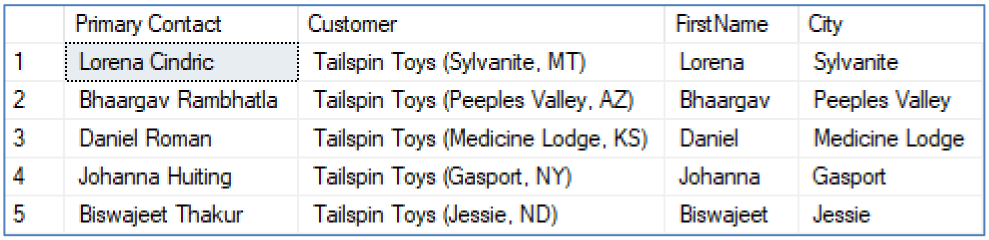
Function overview

The STRING_SPLIT function can split a string into substrings by using a delimiter. The function creates a column containing the substring values and another column with their ordinal position if the ordinal option is enabled. This can be used to analyze frequent words within a column of strings. It can also be used to select and clean substrings to them in an understandable format. For example, each string in a row might contain an employee’s name and badge number. A string split could be used for all the strings in a column and only retrieve the names of employees to display in a view to a user who should not be privileged to see both the employees’ badge numbers and names by using the ordinal option.

Substring Extraction Example with STRING_SPLIT

```
SELECT D.[Primary Contact], D.Customer, D.FirstName, value City
FROM (
  SELECT C2.[Primary Contact], C2.Customer, C2.FirstName, value C2v, ordinal C2o
  FROM (
      SELECT B.[Primary Contact], B.Customer, B.FirstName, value Bv, ordinal Bo
      FROM (
        SELECT *
        FROM (
          SELECT value AS FirstName, C.[Primary Contact], C.Customer
          FROM Customer C
          CROSS APPLY STRING_SPLIT(C.[Primary Contact], ' ',1)
          WHERE ordinal = 1
        ) AS NS
        WHERE Customer <> 'Unknown' AND Customer NOT LIKE '%Head Office%'
        ) B CROSS APPLY STRING_SPLIT(B.Customer, '(', 1) WHERE ordinal = 2 ) B2
    CROSS APPLY STRING_SPLIT(Bv, ')', 1)
    WHERE ordinal = 1) D CROSS APPLY STRING_SPLIT(C2v, ',', 1)
WHERE ordinal = 1;
```

Results



	Primary Contact	Customer	FirstName	City
1	Lorena Cindric	Tailspin Toys (Sylvanite, MT)	Lorena	Sylvanite
2	Bhaargav Rambhatla	Tailspin Toys (Peeples Valley, AZ)	Bhaargav	Peeples Valley
3	Daniel Roman	Tailspin Toys (Medicine Lodge, KS)	Daniel	Medicine Lodge
4	Johanna Huiting	Tailspin Toys (Gasport, NY)	Johanna	Gasport
5	Biswajeet Thakur	Tailspin Toys (Jessie, ND)	Biswajeet	Jessie

Query description

The city name of each value stored in the customer column is added to the row in this example query. The function splits strings and uses the ordinal property to extract the required substring. This allows data transformation entirely in Microsoft SQL Server.

FIRST_VALUE and LAST_VALUE Functions

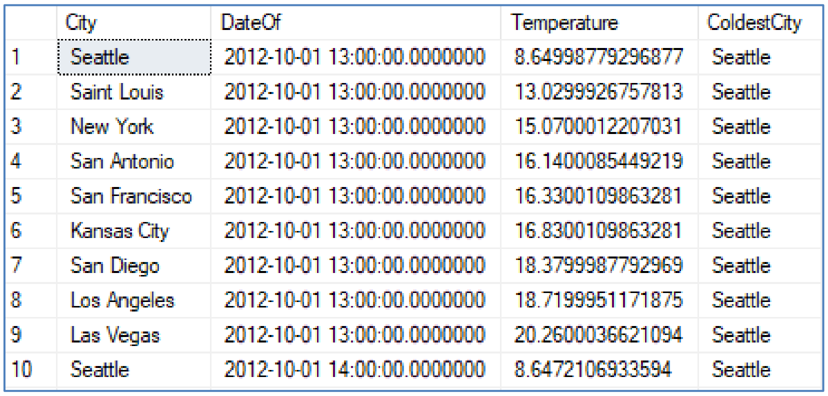
Function overview

The FIRST_VALUE and LAST_VALUE functions are nondeterministic because the order by function is nondeterministic, which changes the first and last values depending on the conditions chosen when there are multiple equal values. These functions are helpful for rapidly determining the first and last values that appear in a table or partition. This can be used to find the item with the minimum price and populate its name alongside all the other items in that table’s partition which is an additional functionality not previously available to the MIN and MAX functions without performing a self-join. Partitions can provide great value because they allow for more data insights to be easily visualized by a single query.

Example FIRST_VALUE Query

```
SELECT City, DateOf, Temperature,
        FIRST_VALUE(City) OVER (PARTITION BY DateOf ORDER BY Temperature ASC) AS ColdestCity
FROM Temperature_View
ORDER BY DateOf;
```

Results



	City	DateOf	Temperature	ColdestCity
1	Seattle	2012-10-01 13:00:00.0000000	8.64998779296877	Seattle
2	Saint Louis	2012-10-01 13:00:00.0000000	13.0299926757813	Seattle
3	New York	2012-10-01 13:00:00.0000000	15.0700012207031	Seattle
4	San Antonio	2012-10-01 13:00:00.0000000	16.1400085449219	Seattle
5	San Francisco	2012-10-01 13:00:00.0000000	16.3300109863281	Seattle
6	Kansas City	2012-10-01 13:00:00.0000000	16.8300109863281	Seattle
7	San Diego	2012-10-01 13:00:00.0000000	18.3799987792969	Seattle
8	Los Angeles	2012-10-01 13:00:00.0000000	18.7199951171875	Seattle
9	Las Vegas	2012-10-01 13:00:00.0000000	20.2600036621094	Seattle
10	Seattle	2012-10-01 14:00:00.0000000	8.6472106933594	Seattle

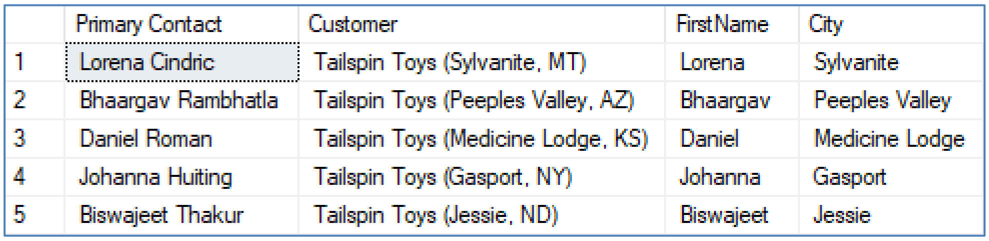
Query description

The FIRST_VALUE function selects the first value in each partition. In this case, the partitions are based on the DateOf values. The first ordered column partition value can be selected and added as a new column without the need for a self-join.

Example LAST_VALUE Query

```
SELECT City, DateOf, Temperature, LAST_VALUE(City)
        OVER (
        PARTITION BY DateOf
        ORDER BY Temperature ASC
        ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS HottestCity
FROM Temperature_View
ORDER BY DateOf;
```

Results



	Primary Contact	Customer	FirstName	City
1	Lorena Cindric	Tailspin Toys (Sylvanite, MT)	Lorena	Sylvanite
2	Bhaargav Rambhatla	Tailspin Toys (Peeples Valley, AZ)	Bhaargav	Peeples Valley
3	Daniel Roman	Tailspin Toys (Medicine Lodge, KS)	Daniel	Medicine Lodge
4	Johanna Huiting	Tailspin Toys (Gasport, NY)	Johanna	Gasport
5	Biswajeet Thakur	Tailspin Toys (Jessie, ND)	Biswajeet	Jessie

Query description

LAST_VALUE is less intuitive than the FIRST VALUE function, and extra care must be taken to implement it correctly. The simplest use case, however, is to combine FIRST_VALUE and use descending order to emulate the LAST_VALUE function. Thanks to ordering the two functions' roles are highly interchangeable.

The reason that the LAST_VALUE function is less intuitive is framing. A frame refers to a set of rows for the window which is usually smaller than a partition. The default frame contains the rows between the current row and the first row. For example, with row 5, the window holds the values of rows 1 through 5. FIRST_VALUE includes the first row by default and so this concept does not interfere with retrieving the desired results.

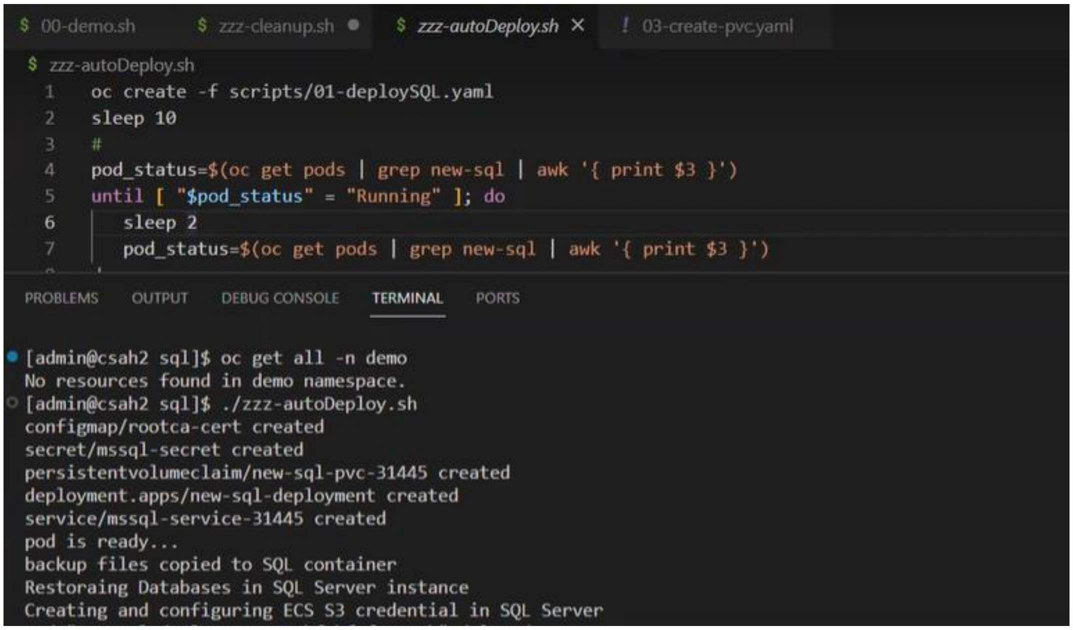
The window only goes up to the current row for the default LAST_VALUE frame. To circumvent this, specify the following frame, ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING. This forces the window to begin with the current row and end at the last row of the partition and provides the expected query results.

Deployment automation use case

To create an SQL deployment for OpenShift cluster, the Dell engineering team developed an automated script to ensure that the data virtualization use cases were easily replicable and repeatable. This script uses a single .yaml file to deploy the SQL Server 2022 instances and perform necessary steps for the data virtualization use case. Encrypted communication between SQL Server and ECS requires prior setup of appropriate certificates.

Deploying SQL Server with S3 Connectivity Through an Automated Script

This script creates the ConfigMap, the secret for logging in to SQL Server, establishes the persistent volume claim and deploys the pods with services exposed. The automated script also is used to automatically restore the sample databases, configure PolyBase, setup S3 connectivity, and run use case validation tests. This is an extremely useful scenario for quickly bringing new database engines online without worrying about the physical location of the data stores.



```
oc create -f scripts/01-deploySQL.yaml
sleep 10
#
pod_status=$(oc get pods | grep new-sql | awk '{ print $3 }')
until [ "$pod_status" = "Running" ]; do
    sleep 2
    pod_status=$(oc get pods | grep new-sql | awk '{ print $3 }')
```

```
[admin@csah2 sql]$ oc get all -n demo
No resources found in demo namespace.
[admin@csah2 sql]$ ./zzz-autoDeploy.sh
configmap/rootca-cert created
secret/mssql-secret created
persistentvolumeclaim/new-sql-pvc-31445 created
deployment.apps/new-sql-deployment created
service/mssql-service-31445 created
pod is ready...
backup files copied to SQL container
Restoraing Databases in SQL Server instance
Creating and configuring ECS S3 credential in SQL Server
```

The script quickly configures and creates all the required components for the analytics as a service scenario. An additional script has been provided to clean up the environment when it is no longer needed. These automation files are available on GitHub. Visit Microsoft’s GitHub for the WorldWideImporters and WorldWideImportersDW backup files. The appendix contains further details about the individual scripts created for automation.

Chapter 5. Summary and conclusion

This chapter presents the following topics:

Summary

This document discusses the key insights gathered by Dell solutions engineers during their setup of the S3 Data Analytics solution. This document also explains process of creating a solution for using ECS for enterprises to seamlessly create their own setups that use data virtualization. By combining S3 compatible object storage with Microsoft SQL Server, it is possible to enhance data protection and reduce storage costs. In this solution, PolyBase was used while connecting to external storage with S3 protocol to perform queries with external data including the backup and restore operations for cloud storage. The ability to connect and query data present on external object storage was also validated using OPENROWSET and SQL Server external tables. This offers an example of data virtualization where users can directly query unstructured or semi-structured data not located internally in SQL server.

This setup was validated for both Linux containers running in an OpenShift cluster and on Virtualized Windows Machines in a VMware High Availability cluster providing deeper insights into the practicality of such setups.

The recently introduced T-SQL functions such as GREATEST, LEAST, DATE_BUCKET, GENERATE_SERIES, STRING_SPLIT, FIRST_VALUE AND LAST_VALUE had their potential analytics use cases tested. The ability to backup and restore from ECS was successfully validated and potential use cases were demonstrated. The deployment process for the containerized setup was automated to streamline container deployment and the analytics-as-a-service approach for an on-premises environment was realized.

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. Data virtualization enables more opportunities for data engineers to improve and reduce the cost of data analytics operations.

Embracing the powerful AMD EPYC 7473X processors and Dell ECS object storage will give organizations' analytic workloads a competitive edge. This combination of products quickly delivers deeper insights for business leaders and provides a platform that is powerful, robust, highly available, flexible, and scalable.

Request for 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. We appreciate the time taken to provide feedback and it allows for us to improve the solutions that we provide.

Author: Sanjeev Ranjan, Caden Weiner

Contributors: Robert Sonders

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

Chapter 6. References

This chapter presents the following topics:

VMware Documentation

The following VMware documentation provides additional and relevant information:

- Create a vSphere HA Cluster
- VMware vSphere ESXi 7.x on Dell EMC PowerEdge Servers Installation Instructions and Important Information Guide
- What is VMware vSphere Virtual Volumes? | vVols
- vVols Getting Started Guide
- Deploy a Virtual Machine from a Template in the vSphere Web Client

Microsoft Documentation

The following Microsoft documentation provides additional and relevant information:

- Create a failover cluster | Microsoft Learn
- Create a VMware vSphere template for Windows Server 2019
- Tutorial: Configure a SQL Server Always On availability group
- DATE_BUCKET T-SQL function documentation
- GREATEST and LEAST T-SQL function documentation
- GENERATE_SERIES T-SQL function documentation
- STRING_SPLIT T-SQL function documentation
- FIRST_VALUE T-SQL documentation
- LAST_VALUE T-SQL documentation
- OVER Clause (Transact-SQL) - SQL Server | Microsoft Learn
- BACKUP and RESTORE (Transact-SQL) - SQL Server | Microsoft Learn

RedHat Documentation

The following OpenShift documentation provides additional and relevant information:

- Installing a user-provisioned cluster on bare metal - Installing on bare metal | Installing | OpenShift Container Platform 4.11

Kubernetes Documentation

The following Kubernetes documentation provides additional and relevant information:

- Concepts | Kubernetes

AMD Documentation

The following AMD documentation provides additional and relevant information:

- AMD 3D V-Cache™ Technology | AMD

Appendix A Automation Scripts

This appendix presents the following topics:

Automation scripts

Deployment

This script creates a Kubernetes deployment running Microsoft SQL Server 2022 instances. It will handle all the required setup such as creating persistent volume claims, secrets for login credentials, mapping config certificates, creating the deployment and exposing the deployment through a service. This yaml file is discussed in detail within the containerized environment setup section of the paper.

S3-Analytics-Automation/01-deploySQL.yaml at main · caden-at-dell/S3-Analytics-Automation · GitHub

Restore Database

These queries automatically restore the WideWorldImporters and WideWorldImportersDW databases into a Microsoft SQL Server instance. These files are restored from the PowerStore Server volume into the appropriate .mdf, .ndf and .ldf files.

S3-Analytics-Automation/01-restoredb.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub

Setup Connectivity

This automation handles the process of assigning a credential and access key to the database for S3. This file also contains functions that will configure PolyBase and create database scoped credentials that will be used for Dell ECS S3.

S3-Analytics-Automation/02-setup_s3_connectivity.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub

Setup PolyBase

This script establishes Dell ECS as the external data source and creates the Parquet file format which will allow for Parquet data to be imported and converted to CSV. It also reconfigures PolyBase so that it allows the PolyBase export feature to be activated.

S3-Analytics-Automation/03-setupPolyBase.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub

Converting CSV to Parquet

This script creates external tables using the data available from S3 compatible storage. It imports the data stored in Parquet format as CSV data values. It makes use of the CETAS functionality discussed in the data virtualization use case section of the paper.

S3-Analytics-Automation/04-convertCsvToParquetUsingCETAS.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub

Setup views

This script contains SQL functions that set up the views that are used to demonstrate the new T-SQL functions and the process of joining data from different sources. This uses the CROSS APPLY function to reformat the data into a format appropriate for the queries that are used to demonstrate the data virtualization functionality.

S3-Analytics-Automation/05-setupViews.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub

Cleanup external Tables

This script allows the external tables created during the testing phase to be dropped.

S3-Analytics-Automation/z-cleanup-external-tables.sql at main · caden-at-dell/S3-Analytics-Automation · GitHub