COHESITY

Version 1.1

July 2024

Protect SQL Server with Cohesity VDI-Based Backup

Cohesity Solution for Backup and Restore of SQL Server Databases with Microsoft Virtual Device Interface

ABSTRACT

As databases continue to grow in number and size, data centers in today's organizations need different methods and strategies to protect their databases and manage their growing data. Now, because we have integrated the Cohesity platform with VDI, you can use it with our SQL adapter to add another backup method to your toolbox; Cohesity's VDI-based backup gives you the flexibility of a SQL native backup.

Table of Contents

Figures

Complexity Forces Us to Sink or Swim

In complex data center environments, there is no easy way to manage all the backups and recoveries that comprise the foundation of their protection. Administrators often describe the task as similar to keeping many plates spinning at the same time; without an enterprise-level data management solution, one is left to either sink or swim.

Three essential factors push the demand for efficient data management:

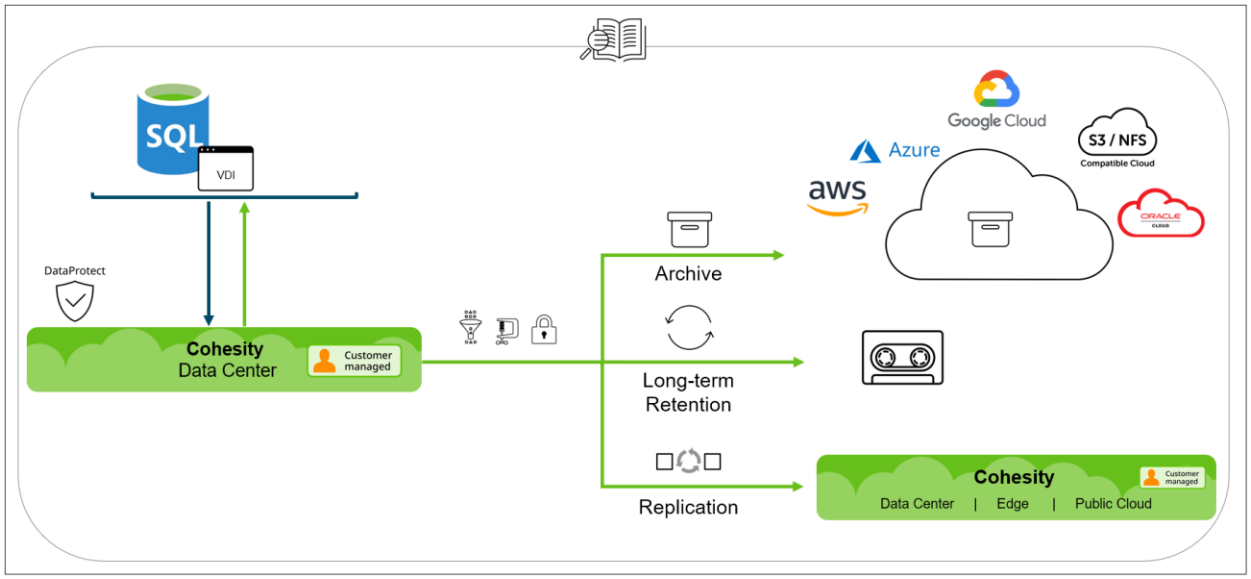
- **The growing number of databases**. The number and kind of backups increases as the demand for protection continues to grow.
- **The increasing size of databases**. Larger databases result in longer backup windows that make it harder to meet your SLAs.
- **The ever-increasing duration and cost of retention**. Managing the storage for backups becomes more difficult as company standards and government regulations increase retention requirements.

Cohesity's VDI-Based Protection for SQL Databases

Cohesity is a modern, software-defined platform for data management. Taking inspiration from its web-scale architecture and leveraging its unique distributed file system (SpanFS®), Cohesity offers high-scalability and reliability, with extensive features to help you meet your business and compliance needs, your SLAs, and your long-term retention, recovery, and preparedness requirements.

Cohesity's flexible architecture makes expansion easy, increasing operational simplicity and improving TCO. Cohesity's solution for SQL Server works on-premises, in the public cloud, and in your remote and branch offices. You choose how to back up your data, where to keep your backups, and for how long.

Figure 1: Cohesity's VDI-Based Protection for SQL Server



Features and Benefits of Cohesity Protection

Cohesity's solution for SQL Server includes many features that make your backups much more valuable, including:

- **Flexibility**. Cohesity gives you the ability to browse and search across all your snapshots, and to restore to different locations on different servers.
- **Performance to Meet Your SLAs**. Cohesity gives you the backup performance you need to protect your SQL Server databases efficiently and securely.
- **Scalability**. Cohesity protection for SQL Server is scalable from a single database to several SQL failover cluster Instances and even an entire data center with hundreds of SQL instances.

- **Compression**. Data compression significantly reduces storage usage and data transmission. Efficient storage means you have room for more backups and other important data. By default, Cohesity performs compression on all the data it stores.

  If you also enable *inline* compression, the process occurs *as* Cohesity is saving the data to storage, instead of after saving it.

- **Encryption at rest, in flight, and in the cloud**. It is vital to protect your data from unauthorized access.
  - **Data-at-Rest**. The Cohesity SpanFS® file system provides full at-rest encryption based on the strong AES-256 CBC (Cipher Block Chaining) standard.
  - **Data-in-Flight**. Cohesity can encrypt all data that is transmitted.
  - **Data-in-Cloud**. Cohesity's CloudArchive provides encryption for data stored in the cloud.

  For details, see Cohesity Security Features in the online Help.

- **Archive to cloud**. Cohesity's policy-based ability to archive to public clouds like AWS, Azure, and Google Cloud, as well as to any S3-compatible storage, makes it easy to leverage lower-cost long-term retention and protect your data from regional disasters. Cohesity makes it easy to retrieve your organization's information to different geographical locations, whenever you need to.

- **Disaster Recovery**. Protect your SQL Server universe from disaster by replicating your Cohesity backups to another location that can be ready to failover (and failback, after repairs) as soon as disaster strikes.

In addition, you can use Cohesity to protect SQL servers that are deployed with different configurations. SQL Server might be on a Windows Failover Cluster, or configured for Always On Availability Groups (AG), or SQL Server might be running on a Virtual Machine (VM). In such cases, you can use Cohesity *for other SQL Server backups* and even *create new backup strategies*. For example, you can protect the SQL Server database and the VM it is running on, or you can protect SQL databases across different data centers.

Use Cohesity VDI-Based Protection for SQL Databases

Cohesity offers a policy-based, highly scalable data protection infrastructure for your SQL Server data. With a few steps, you can set up Cohesity to meet all your data protection requirements.

Cohesity uses Microsoft SQL Server Virtual Device Interface (VDI) to perform backups of databases on any SQL Server instance that is registered with Cohesity.

The Cohesity platform supports full database server backups, differential SQL Server database backups (via Cohesity incremental backups), and SQL Server T-log backups.

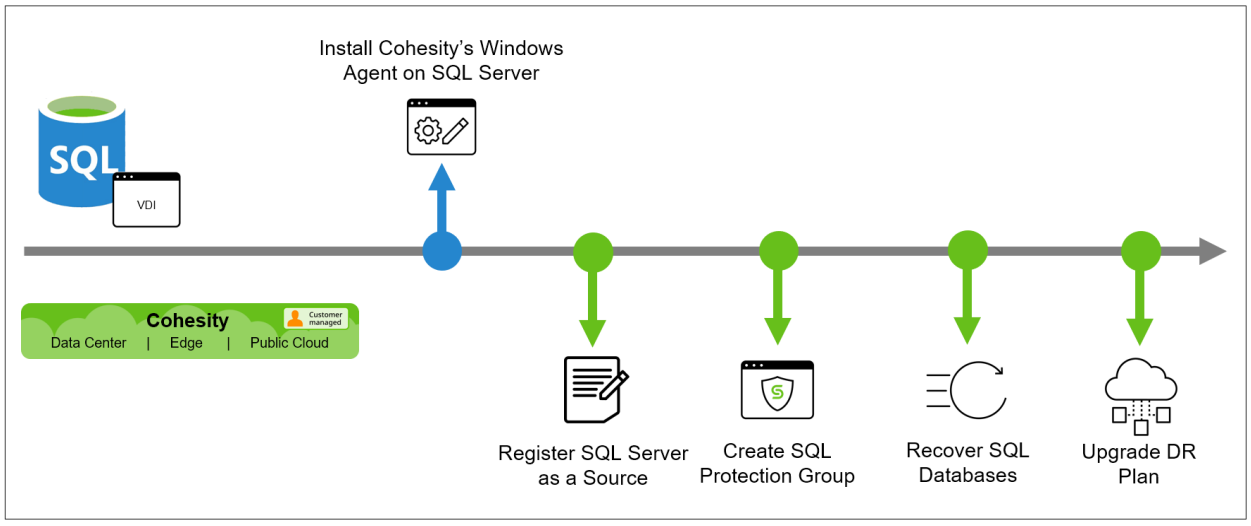
It is important to understand the process that makes a backup strategy successful. For example, it is important to have a second copy of backup data in case the original copy fails. But that is only part of the story. When you need to recover your SQL databases, it is crucial that you be able to find those backups and restore them quickly. To take full advantage of the many features of Cohesity's solution, be sure you understand each step of the implementation.

To protect your SQL databases using Cohesity:

1. Install and deploy Cohesity's Windows Agent on your SQL server.
2. Register your SQL Server as a Cohesity source.
3. Create a Cohesity Protection Group to specify the SQL data you need to protect.
4. Recover protected SQL Server databases.
5. Upgrade your disaster recovery (DR) plan to improve your enterprise's readiness and resilience.

NOTE: For more background, see Appendix A: Terminology and Appendix B: Product Documentation.

Figure 2: Set Up SQL Server Data Protection with Cohesity



Complete these steps to protect your SQL databases. Get started by deploying Cohesity's Windows Agent next!

Deploy the Cohesity Windows Agent

To start, you need to install the Cohesity Windows Agent on your SQL Server host. The Windows agent is designed to work specifically with the Windows operating system and is compatible with Windows versions 2008R2 and above. If there are multiple SQL Server hosts, you will need to install the agent on each host you wish to protect.

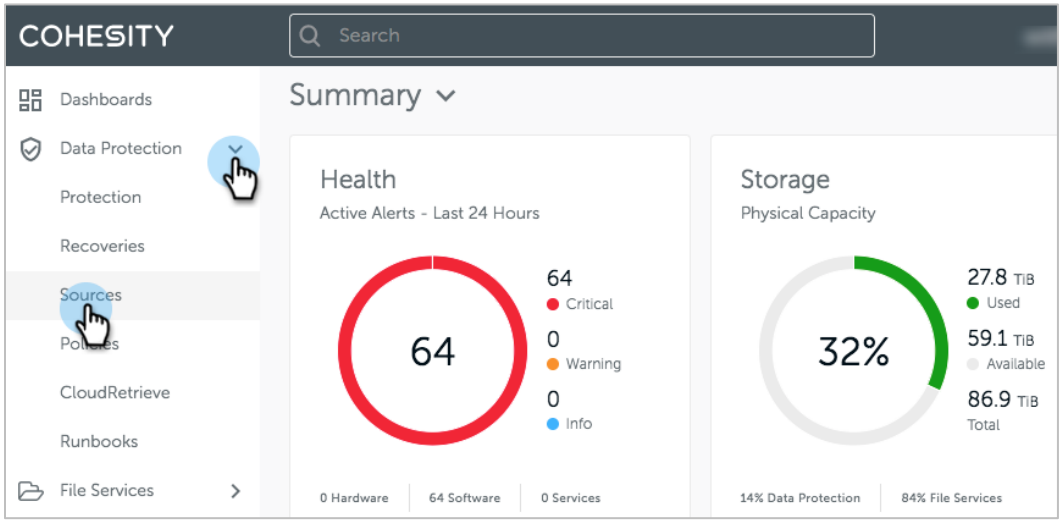
The agent is lightweight and has a small memory footprint. It carries out the tasks you define in your Cohesity Protection Groups and ties together technologies and capabilities already in Windows, like Windows VSS, and new technologies, like Cohesity Changed Block Tracker (CBT), so that you can tackle data management efficiently.

You manage the Cohesity Agent through the Sources page in Cohesity. When an upgrade becomes available for any agent you've installed, an **Upgrade Agent** button appears next to your SQL Server source. You can upgrade the agent from there.

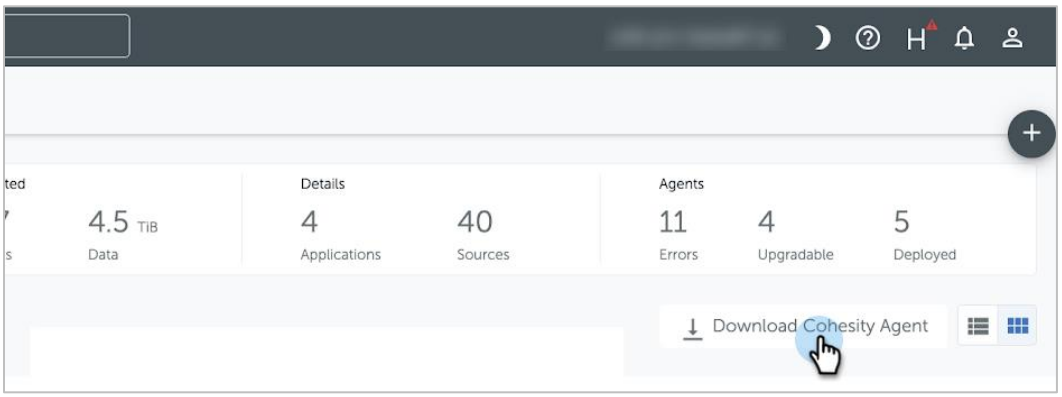
IMPORTANT: You need to install the agent on each SQL Server host you wish to protect.

To install the agent:

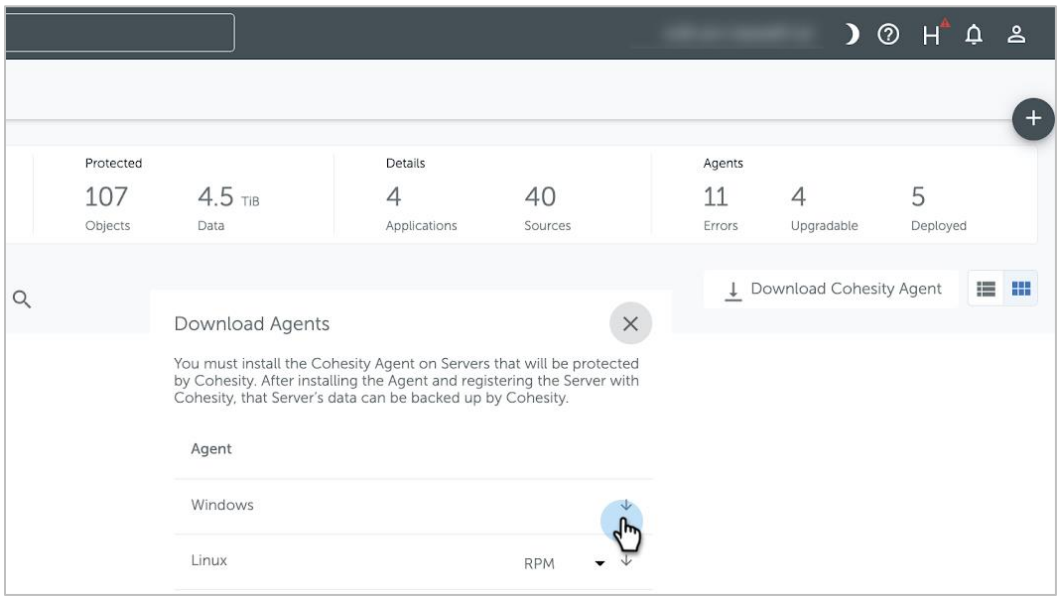
1. Log in to Cohesity and navigate to **Data Protection > Sources**.



2. Click **Download Cohesity Agent**.

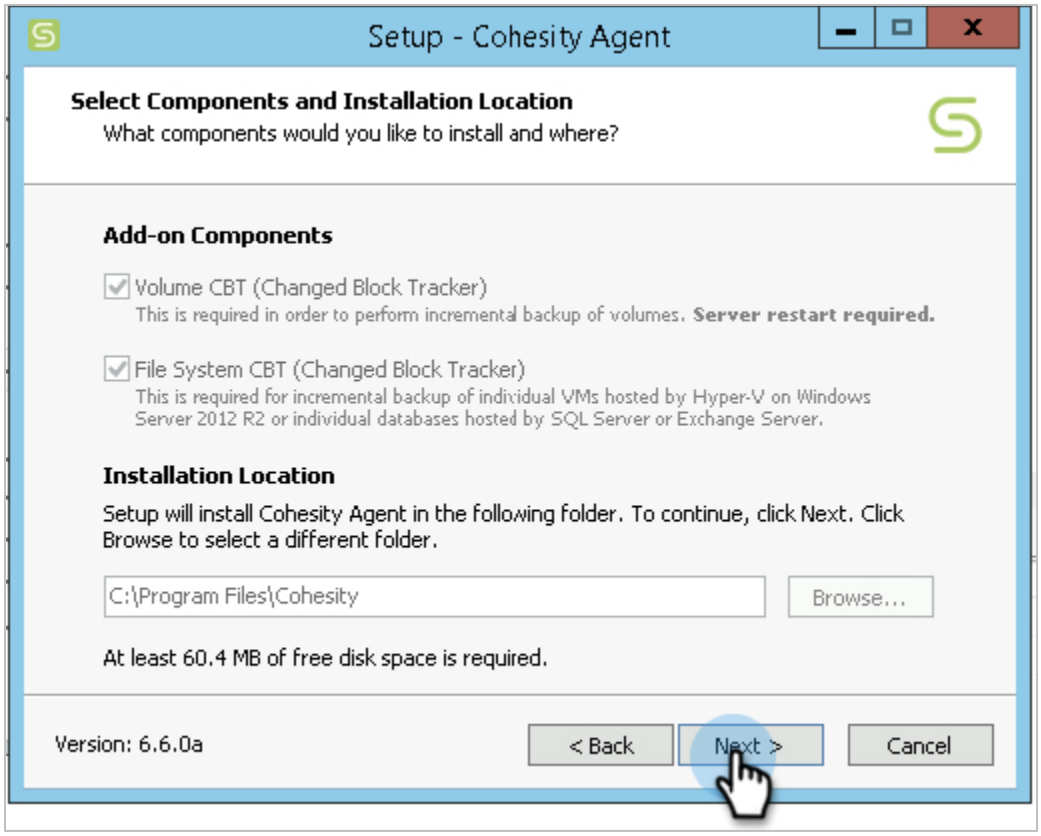


3. Click the **Download** (↓) button for **Windows**.



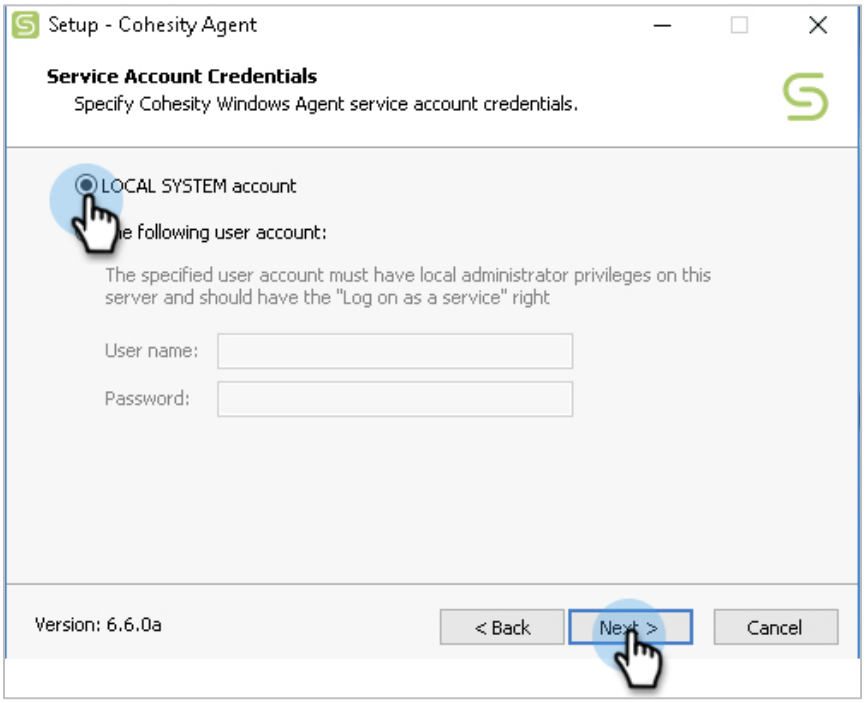
4. Download or copy the agent to all SQL Server hosts you want to protect. On each SQL Server host, install the agent and run the installer.

5. When you launch the Cohesity Windows Agent installer, under **Add-on Components**, ensure that **Volume CBT** (the default) and **File System CBT** are selected and click **Next**.



> **TIP**: The Cohesity volume snapshot captures all the data on the SQL Server host. That means it captures all the changes on the volume *and* changes to other, non-SQL files. To keep your SQL backups running efficiently, keep the volume that SQL Server is using clear of lower-priority and unrelated files.

6. Select the type of service account to use for control over your systems. The **LOCAL SYSTEM account** gives you direct control, but you can also enter an Active Directory domain admin user account.



> **TIP:** If you are unsure which account to use, don't let that slow you down — choose the **LOCAL SYSTEM account**. This account already has most of the required permissions. You can change the agent service account later if you change your mind.

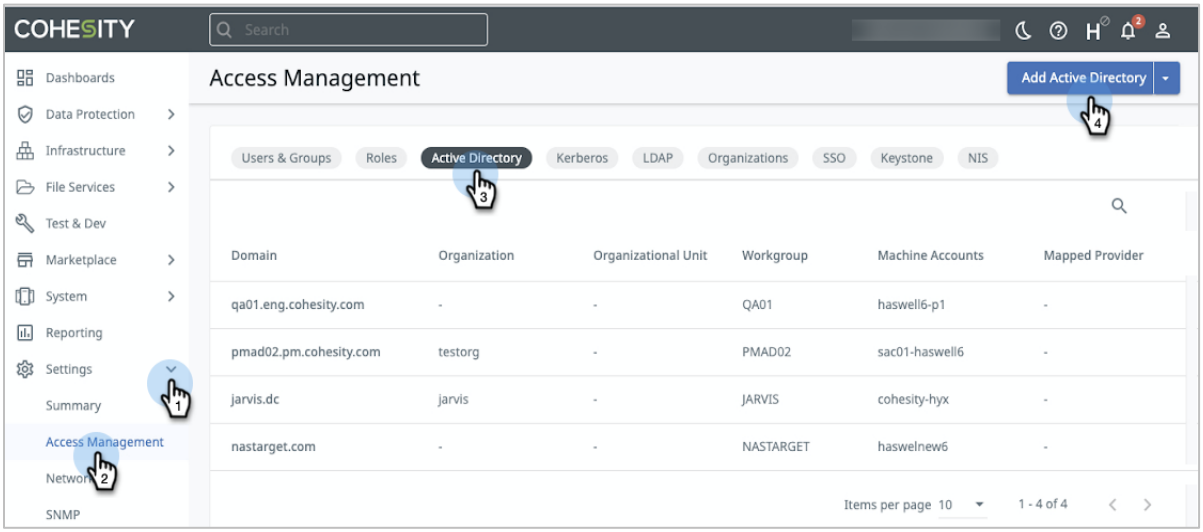
Your SQL Server host is now ready to be registered as a Cohesity source in the next chapter.

Register SQL Server in Cohesity

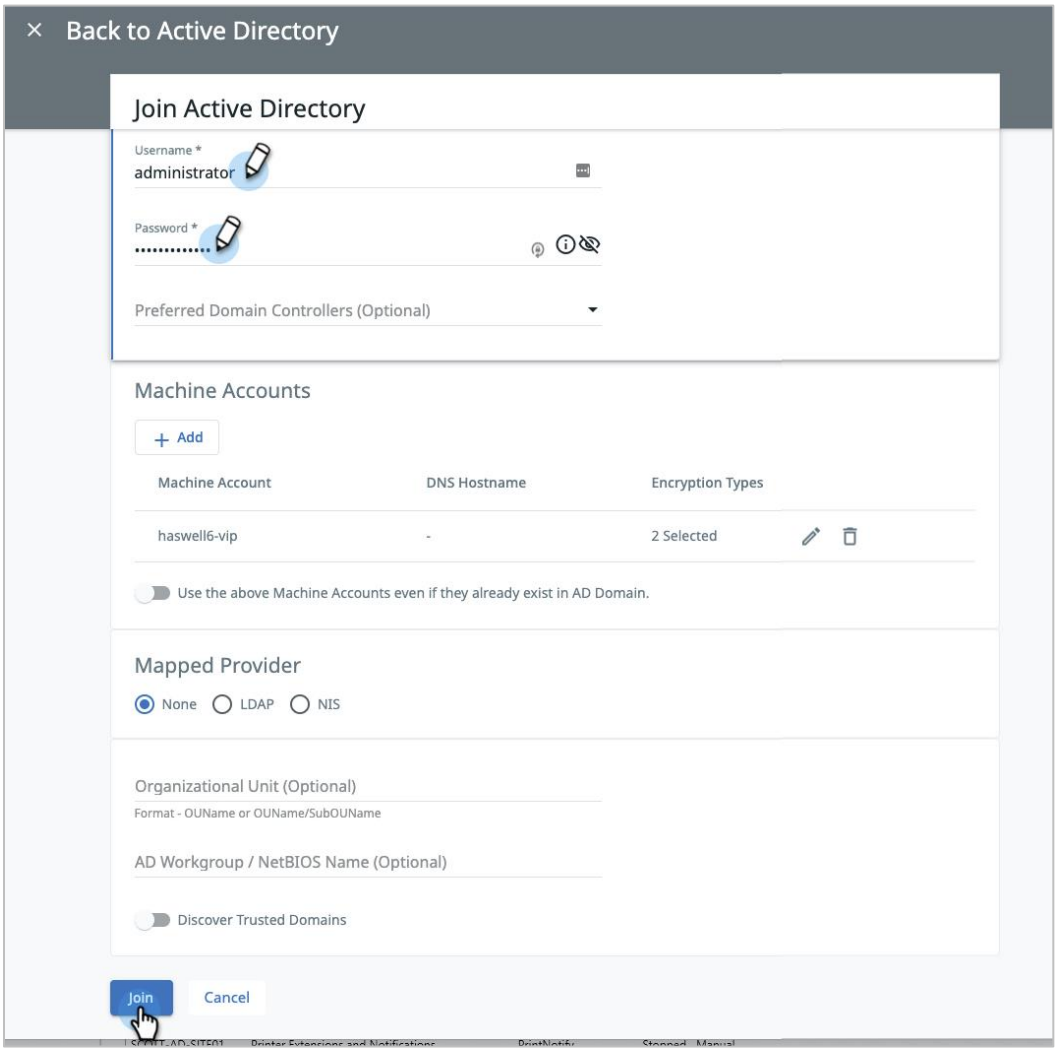
To protect your SQL Server databases with Cohesity and take advantage of its many features, you need to register it as a Cohesity source. Once it's registered in Cohesity, you will be able to add it to a Protection Group and configure the settings for your environment.

To register SQL Server as a Source in Cohesity:

1. Before you can register SQL Server as a source, you need to join AD to your Cohesity cluster. Log in to Cohesity, navigate to **Settings > Access Management > Active Directory**, and click **Add Active Directory**.

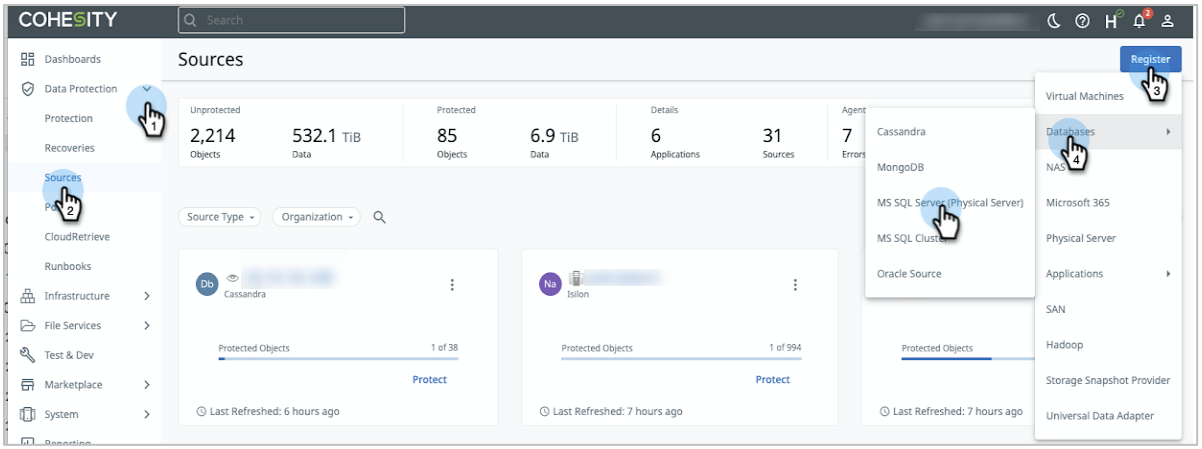


2. In the **Join Active Directory** form, enter the AD administrator **Username** and **Password** and click **Join**.



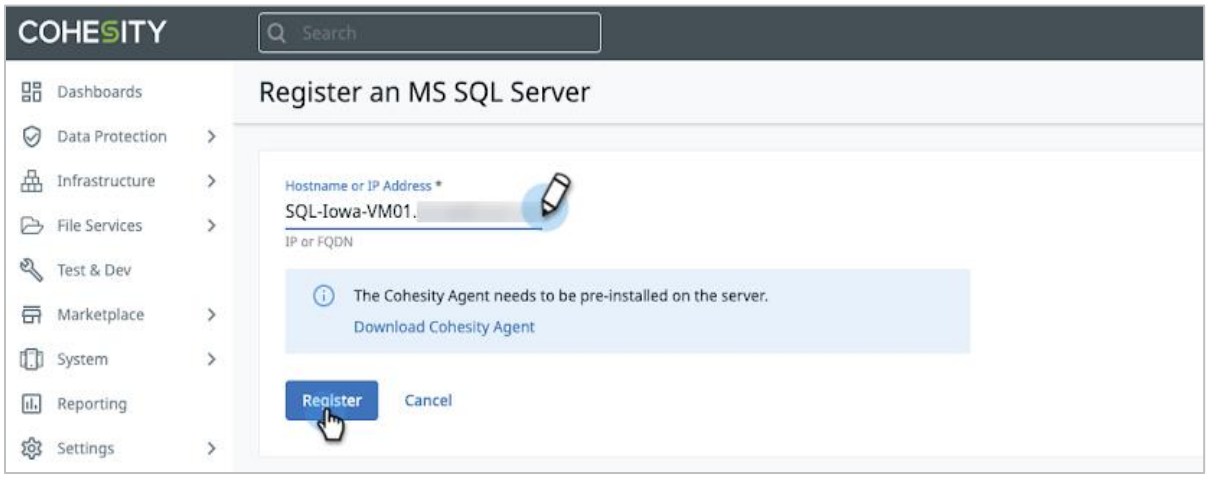
For details on the other options here, see Join the Cluster to an AD Domain in the online Help.

3. Navigate to **Data Protection** > **Sources**. Then click **Register** and select **Databases** > **MS SQL Server (Physical Server)**.

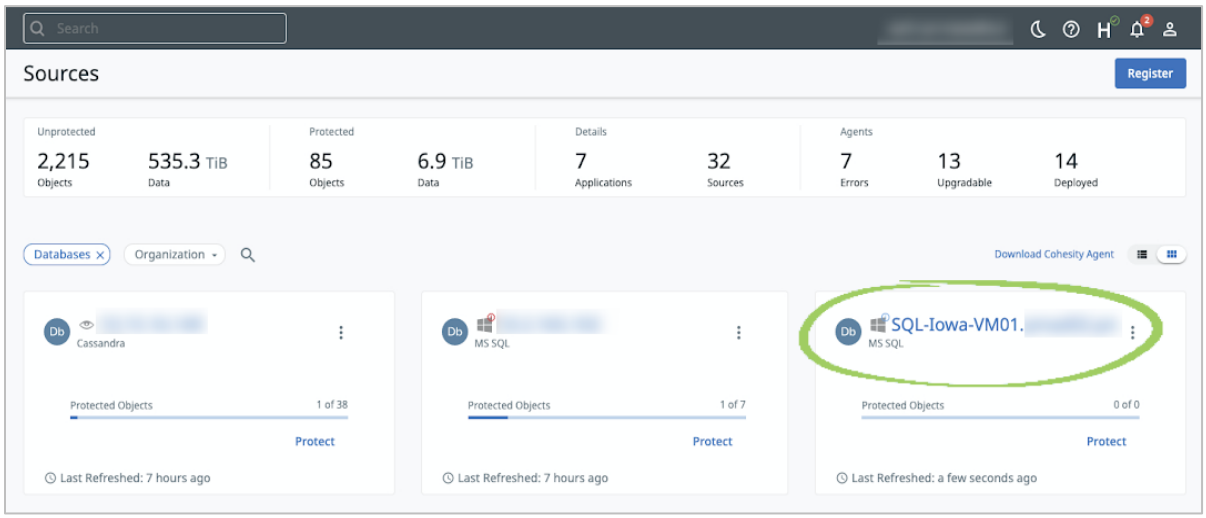


NOTE: When you register SQL Server, Cohesity registers both the host and the application. Cohesity sees the host *and* the SQL Server application so that it has full application awareness.

4. In the **Register an MS SQL Server** form, enter the SQL Server FQDN or IP address and click **Register**.



5. The **Sources** page now includes your SQL Server (in our example, **SQL-Iowa-VM01**), available for immediate protection.



To protect your newly registered SQL source, you'll create a Cohesity Protection Group for it in the next chapter.

Create a Protection Group

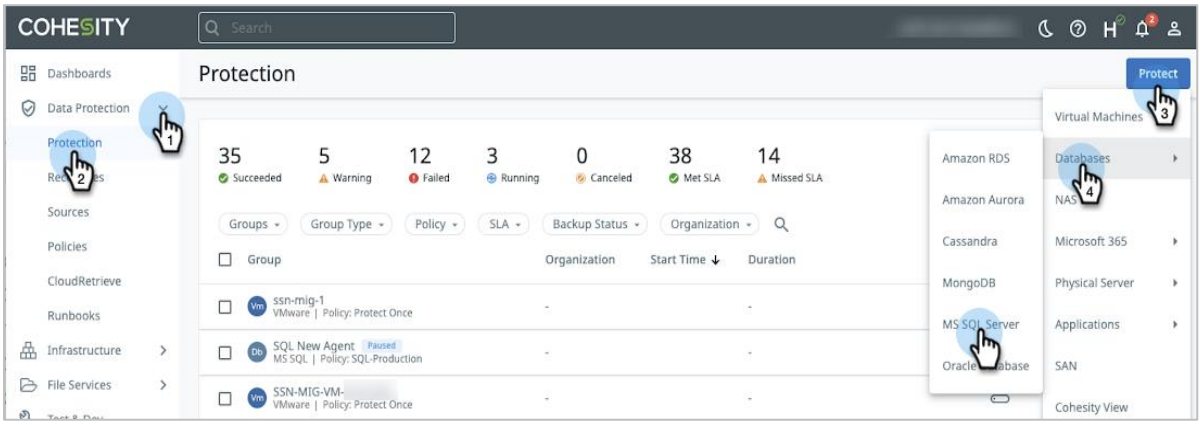
Automation is the only way to stay ahead of the demand for backups and data management. In Cohesity, Protection Groups combine operational requirements (which objects to protect, indexing, alerts, exclusions, inclusions, etc.) with the business requirements that are defined in a Protection Policy (scheduling, retention, etc.). Multiple Protection Groups can use the same Protection Policy, but each Group can have only one Policy. For more, see About Policies and Protection Groups in the online Help.

Automate your SQL database backups by building a Protection Group and assigning the SQL host you registered earlier and applying the Protection Policy that meets your business requirements.

NOTE: Because SQL Server configurations can be implemented across multiple hosts, or can be part of a Windows cluster, it's important that you identify all the SQL Server hosts so that they can be included in your Protection Group.

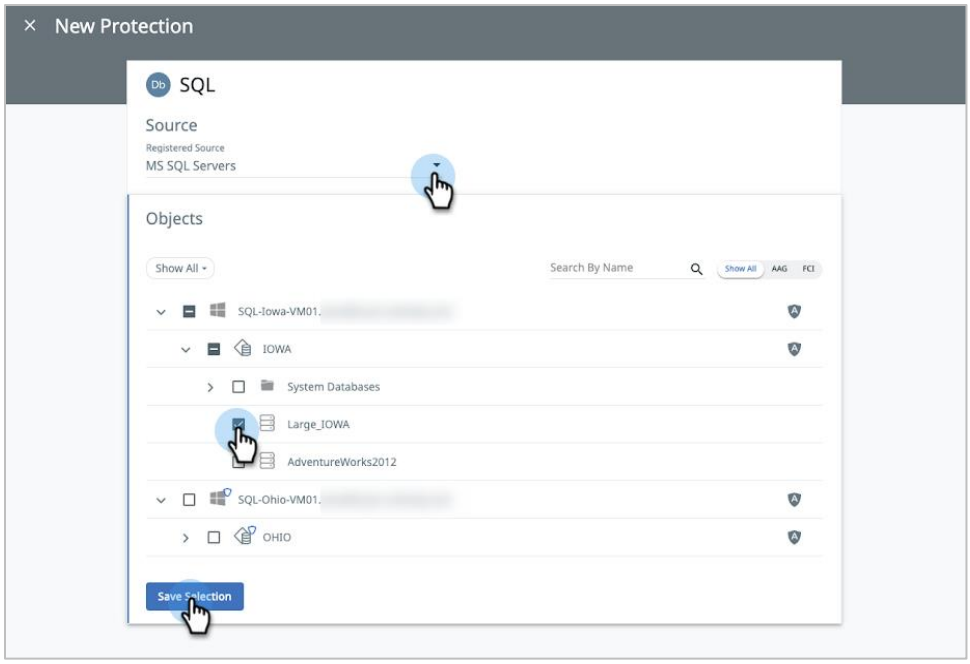
To create a Protection Group:

1. Log in to Cohesity and navigate to **Data Protection** > **Protection**. Then click **Protect** and select **Databases** > **MS SQL Server**.

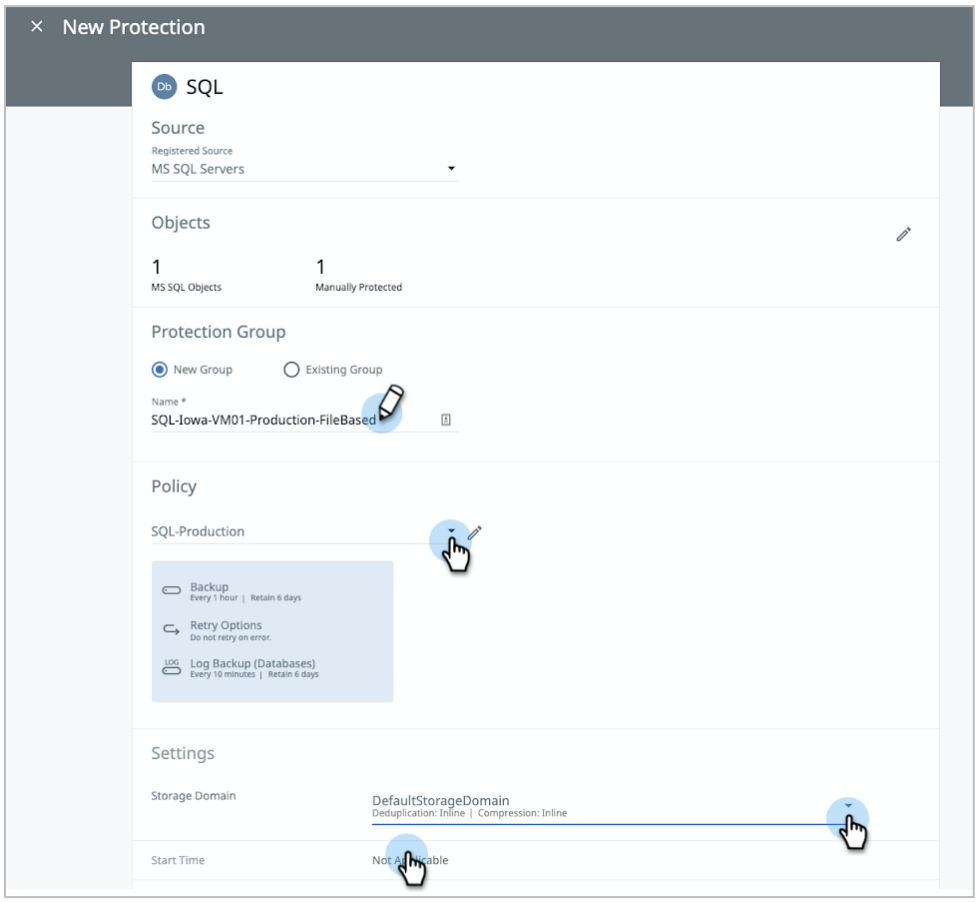


TIP: You can add or remove more SQL sources to the Protection Group later if you like. In this way, you can build onto the Protection Group to manage all your SQL servers.

2. In the **New Protection** form, under **Source**, select the SQL host(s) you registered earlier, make your selections, and click **Save Selection**.



3. In the same form, enter a Protection Group **Name** and select the appropriate **Policy**. Under **Settings**, select the **Storage Domain** and set the **Start Time**.



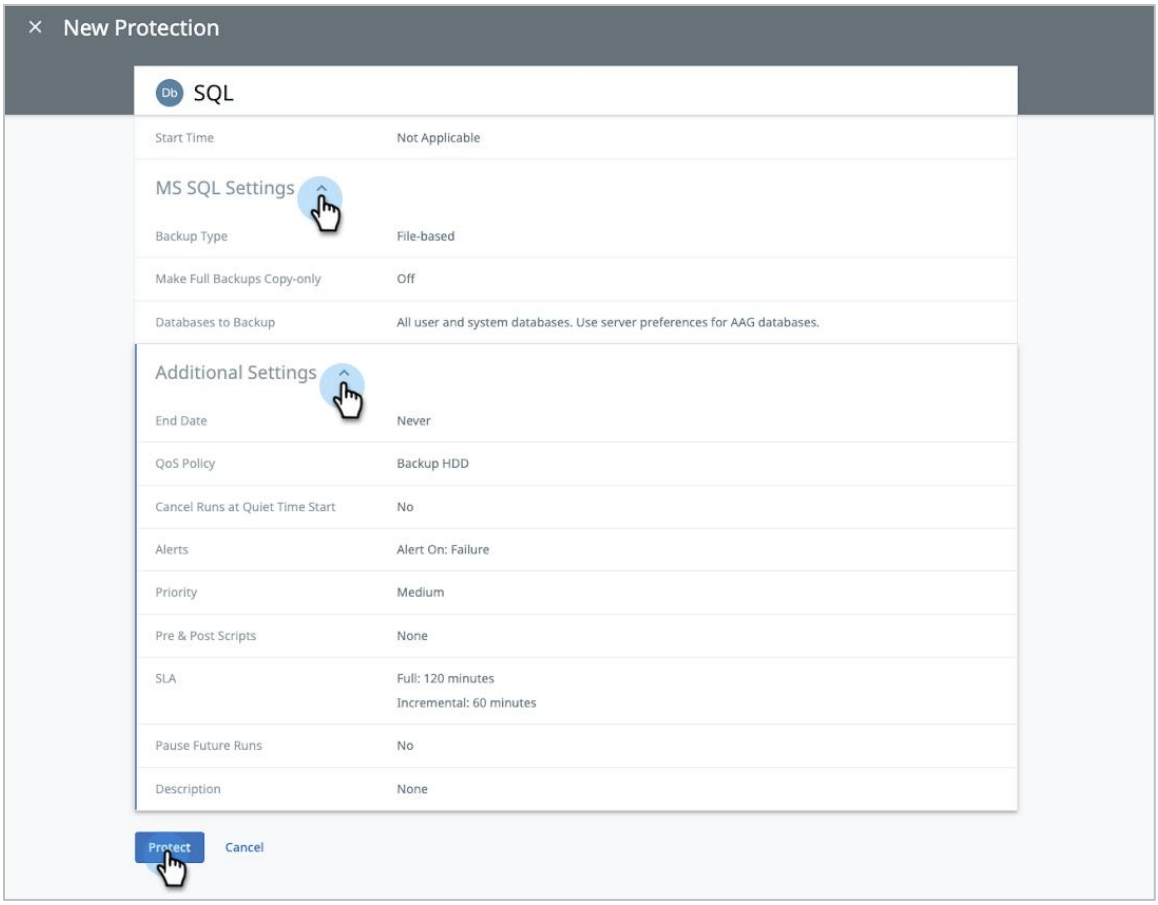
TIPS:

- Give your Protection Group a descriptive name that identifies the kind of data being protected and how it is managed. This will help you identify and manage your SQL backups as your environment grows. Use descriptors like: production (PROD), critical, infrastructure (INFRA), financial, sales, primary, secondary, and employees (EMP). For example:

Production_Sales	DataCenter_Dallas_Production
Critical_Infrastructure	Production_ReplicatedTo_DRsite
Archive_LongRetention	Development_User_Data

- Once you set the Policy for a Protection Group, all the sources assigned to that Protection Group will be conveniently managed the same way.
- To create a custom Protection Policy to meet specific scheduling, retry options, log backup, replication, and archiving needs, learn how to Create or Edit a Standard Policy in the online Help.
- For maximum space savings and security, choose a Storage Domain with compression, deduplication, and encryption enabled. For details, see Create or Edit Storage Domains in the online Help.

4. In the same form, configure any **MS SQL Settings** and **Additional Settings** that you need to and then click **Protect**. For details on the **Additional Settings**, see Create a VDI-based MS SQL Protection Group in the online Help.



Your new SQL Protection Group is now active and running, and appears on the **Protection** page. For more on optimizing your protection, see Cohesity MS SQL Best Practices in the online Help.

Now that you have created a Protection Group for your SQL Server databases, you can add other SQL sources, or change the Protection Policy and settings. In this way, all your SQL sources in this Protection Group will be managed the same way. For example, to replicate all the backups to an off-site target, simply add replication to the Policy that is assigned to this Protection Group.

TIP: In larger environments, build two or three different Protection Groups with different configurations to manage several SQL Server hosts. This helps keep your data management simpler even as your environment grows.

Retention for VDI Backups

The aim of the SQL DBA is to have a combination of backups so that the database can be *restored* to any point in time. A good combination of backups consists of having a full backup, differential (in Cohesity, *incremental*) backups, and log backups.

For example, all SQL database restores must begin with a full database backup, secondly, a differential can then be applied to the full, and finally, log backups can be applied in sequence to complete the database restore.

When the backups are applied during the database restore process, you are sequentially adding the captured changes to the database: FULL+DIFF+Log1+Log2+Log3 = Restored Database.

IMPORTANT: All Microsoft VDI backups, their differentials, and their logs are dependent on a full backup to perform a database restore. Microsoft requires that in order to restore a SQL Database, you must start with a full backup, then its transaction logs can be applied. This means your backup retention policy must keep a full backup along with its log backups in order to successfully restore a database.

Simply put, a SQL VDI-based database restore requires a full backup to seed the database, then a differential and/or log backups are applied to the specified point in time.

We recommend retaining two sets of full backups with their differential and log backups.

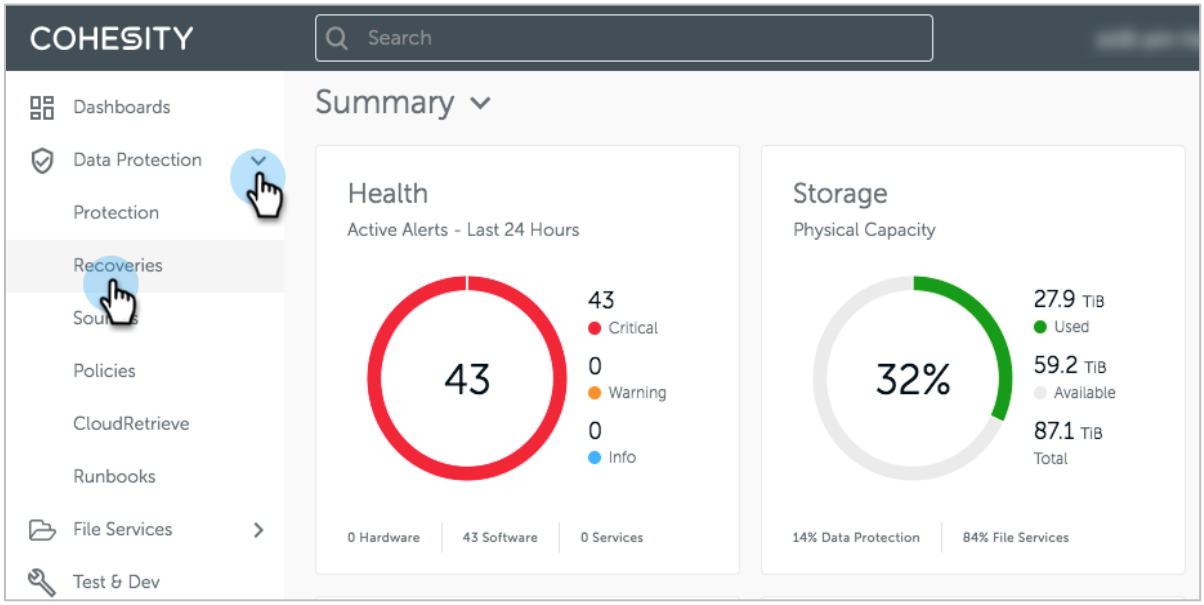
TIP: A good backup plan always includes a combination of full, differential, and log backups.

Recover SQL Database

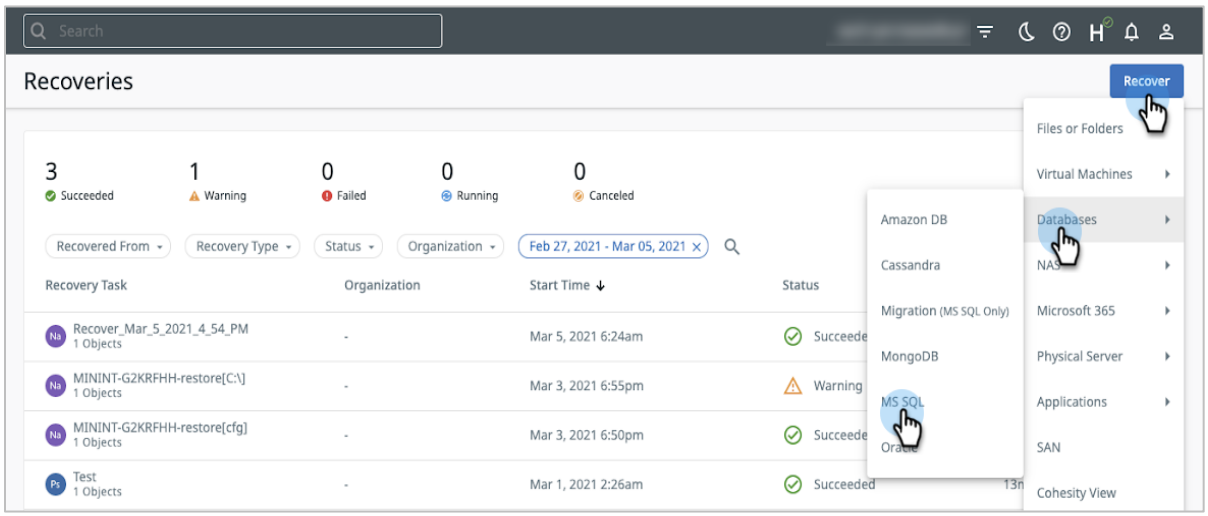
For the DBA, restoring the database starts with a restore of the full backup and then a differential and/or hundreds of individual log files. This is a slow process and very time-consuming. In the same way a SQL DBA manually performs the restore sequence, Cohesity knows to look for the full backup, the differential (if any) in between the full and log backups, and then the logs necessary to walk the restore forward to a specific point in time. Cohesity makes the SQL database restore process easy.

To recover SQL Server databases:

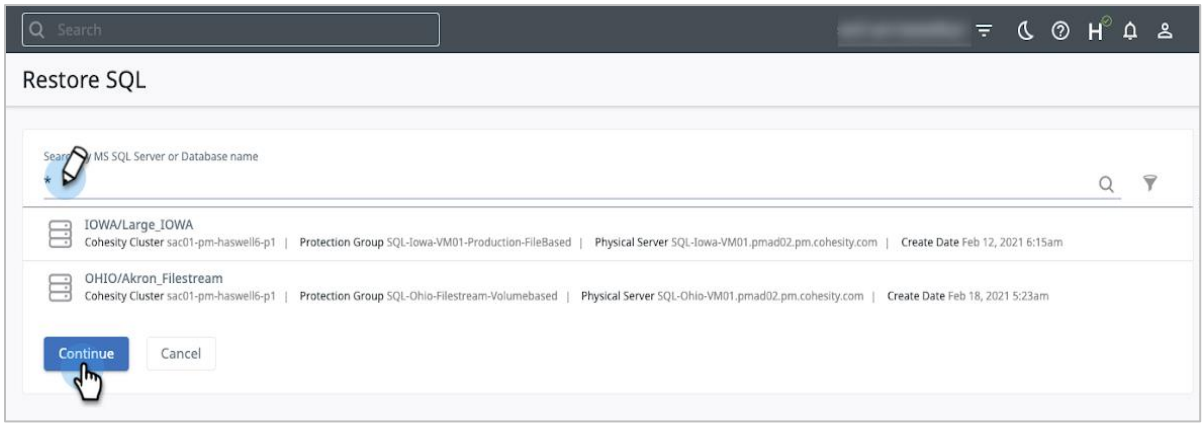
1. Log in to Cohesity and navigate to **Data Protection > Recoveries**.



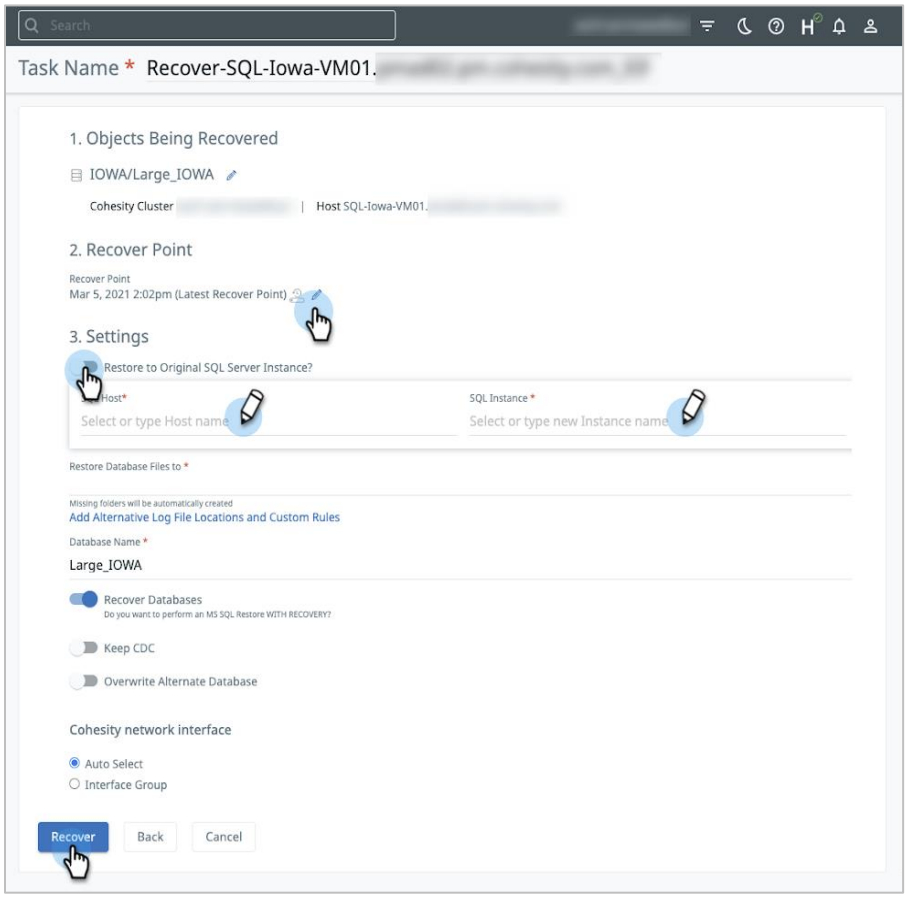
2. On the **Recoveries** page, click the **Protect** and select **Databases > MS SQL.**



3. In the **Restore SQL** form, enter a search term to locate your database backup. When you do, click **Continue**.



4. In the **Task Name** form, if you need a different backup (point in time), click the **Recover Point** to select an earlier backup. Under **Settings**, select the **Restore to Original Server Instance** and enter the **SQL Host** and **SQL Server Instance** names where you need to restore the backup. If you need a different backup (point in time), click the backup listed under **Recover Point** to select an earlier backup. Click **Recover**.



For more information on the other SQL DB recovery options, see Restore MS SQL Databases in the online Help.

Your recovery task launches and appears on the **Recoveries** page.

Upgrade Your Disaster Recovery Preparedness

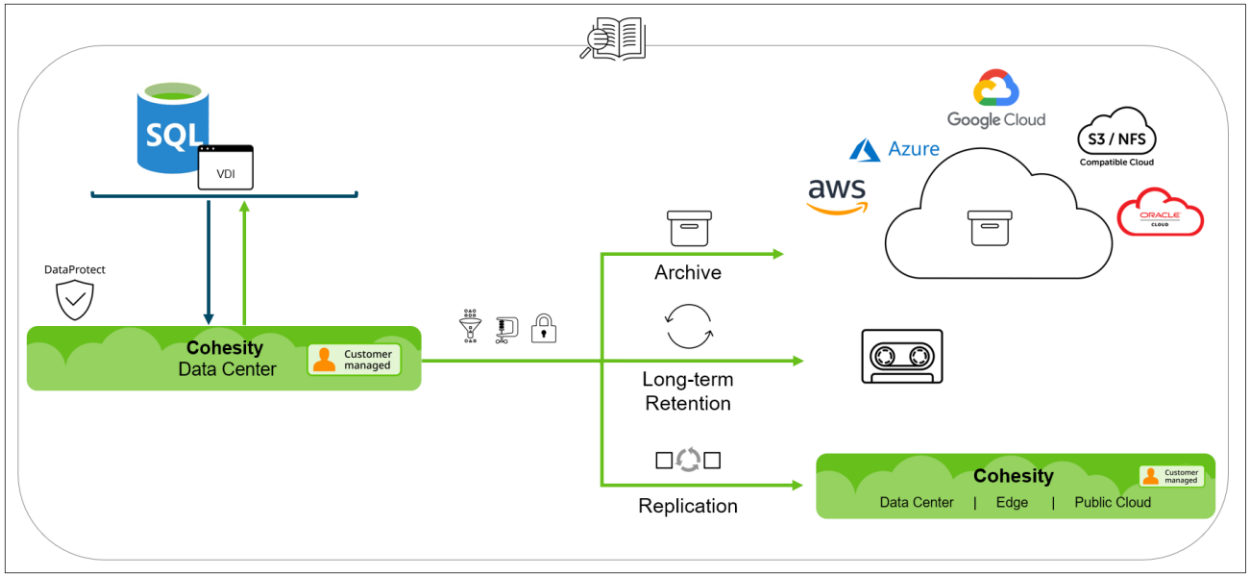
Disaster Recovery (DR) and business continuity are closely related plans designed to proactively protect a business's infrastructure and data. Taking SQL backups is only one part of protecting your business; you must also protect the data from corruption and catastrophic disaster. You can achieve this by keeping a series of backups, replicating those backups off-site, and archiving them under a long-term retention plan.

Cohesity gives you the foundation to build a DR plan to protect your business:

- **Capture and Store**. Protect your SQL databases from loss and corruption with regularly scheduled backups.
- **Geo-Redundancy**. Replicate your SQL backups to an off-site location to protect from catastrophic loss and disaster.
- **Cost-Effective Archival**. Archive your SQL backups to the cloud and store them on lower-cost storage tiers for long-term retention.

Use a Protection Group to schedule regular SQL backups and assign a Protection Policy to include archiving and replicating those backups for long-term retention and disaster recovery.

Figure 3: SQL Backups in Cohesity are Available to Replicate and Archive



Take Local Snapshots

Protect your SQL backups over time by maintaining a series of *local* Cohesity snapshots.

Use Cohesity Protection Groups to schedule and automate SQL backup management.

Replicate Backups Off-Site

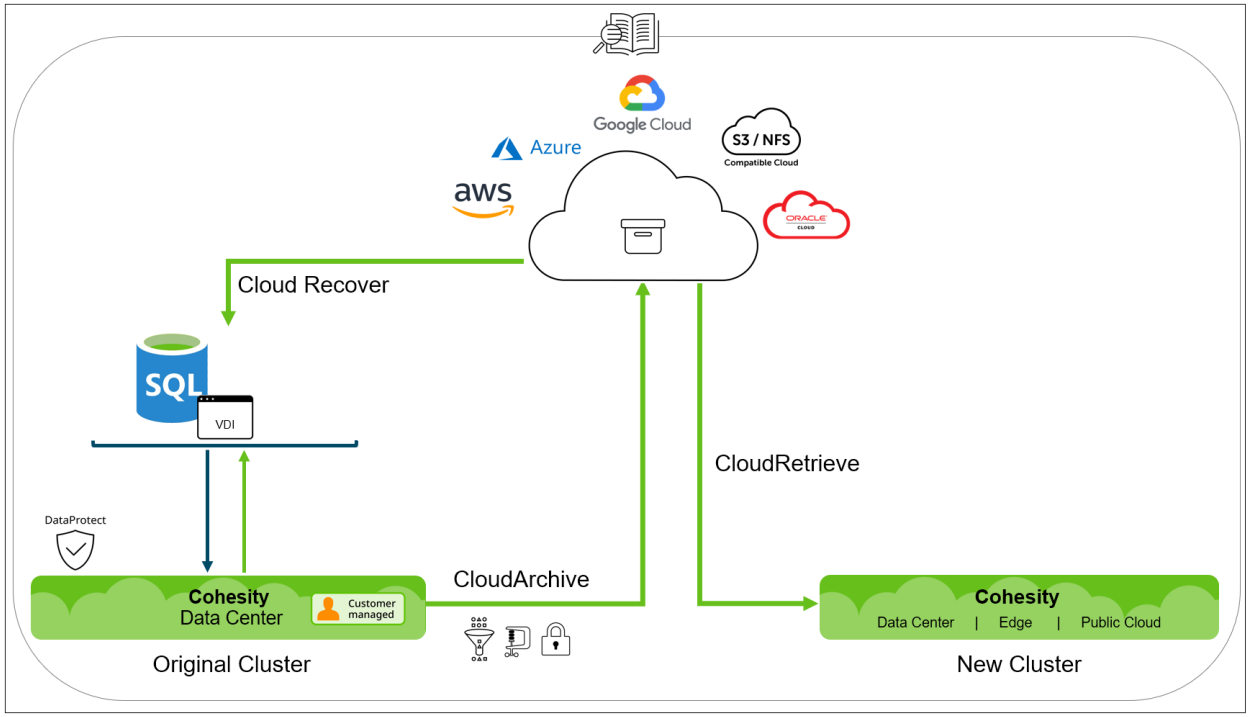
Protect your entire set of SQL backups from catastrophic loss by replicating your SQL backups to an off-site location. Choose a Protection Policy for your Protection Group that automatically copies the SQL backups to a second, off-site Cohesity cluster. By default, deduplication and compression are enabled for replication, and Cohesity sends *only the changed data* over the network, producing a significant reduction in network traffic and cost.

Archive Backups to the Cloud

Archive your SQL backups to the cloud as a way to address long-term data retention requirements and simultaneously lower the cost of storage. Cohesity provides a policy-based method to archive to public clouds (AWS, Azure, and GCP), as well as to any S3-compatible storage.

With Cohesity CloudArchive, Cloud Recover, and CloudRetrieve, your SQL backups are available for recovery to their original Cohesity cluster or onto a different Cohesity cluster, for geo-redundancy and disaster recovery.

Figure 4: Cohesity CloudArchive, Cloud Recover, and CloudRetrieve Provide Disaster Recovery



Best Practices for Cohesity VDI-Based SQL Server Protection

Configuring the right Cohesity settings dramatically improves the performance of your backups, and the efficiency of your storage and archives. Manage your backups by choosing the optimal settings for deduplication, compression, and encryption.

- **Use inline deduplication**. Deduplication (enabled by default) prevents duplicate blocks of repeated data from being stored, dramatically reducing your storage consumption.

  With *inline deduplication*, the process occurs as Cohesity is saving the blocks, instead of waiting until *after* Cohesity has written the data to its Storage Domain. We recommend that you use deduplication and wherever possible, enable inline deduplication.

- **Use inline compression**. Compressing your data significantly reduces the space needed to store your backups and frees up space for more backups and other important data.

  With inline compression, the process occurs as Cohesity is saving the blocks, instead of waiting until *after* Cohesity has written the data to its Storage Domain. Both compression and inline compression are enabled by default, and we recommend that you take advantage of them. For details, see Create or Edit Storage Domains in the online Help.

- **Use encryption**. When a platform governs access to data across the systems in your environment, it is crucial to protect the data it manages. We recommend that you enable encryption — at rest, in flight, and in the cloud — for all your SQL Server backups. For more, see Cohesity Security Features in the online Help.

- **Keep multiple snapshots to guard against corruption**. We recommend you maintain five to seven local snapshots of your backups.

- When you capture and store your backups like this, you protect your data from corruption over time. By taking and maintaining several snapshots, you are in position to recover data from its state *prior* to being corrupted. Snapshots are efficient because they capture just the changed blocks of data, and then use deduplication and compression.

  We recommend protecting all your SQL Servers with regularly scheduled backups, and then in turn moving some of those backups off-site and archiving them under a long-term retention plan for disaster recovery.

- **Validate the backups**. We recommend a periodic restore of a SQL Server database in a test environment from its backup. This is an important step in the overall backup strategy because it tests, verifies, and validates the integrity of the backup. Restore a sample from the snapshot set to a non-production server and evaluate it. In addition to confirming that the right data is backed up properly, this practice also ensures that you have already validated your method of restoring objects *before* a critical event necessitates it.

- **Verify logging and auditing operations**. It is very important to log and audit changes on a SQL Server. Cohesity logs its recovery process. In Cohesity, navigate to **Data Protection > Recoveries** and click into your SQL database recovery task to view detailed logs.

- **Shelter in the cloud**. We recommend archiving to the cloud for low-cost, long-term storage and protection from regional disasters.

  Cohesity provides a policy-based method to archive to public clouds (AWS, Azure, Google Cloud Platform) or any S3-compatible storage. This makes it easy to change policies, meet regulatory requirements, and retrieve your data to different geographical locations.

- **Use replication to defend against site disaster loss**. Protect your entire set of SQL Server backups from catastrophic loss by replicating them to a different geographical site. Cohesity can automatically replicate the SQL database backups stored in the Cohesity cluster to a second, off-site Cohesity cluster.

  Cohesity replication always performs source-side deduplication and compression first and sends only the changed data over the network for cost-effective disaster recovery. As such, Cohesity replication is an essential part of every disaster recovery (DR) plan.

- **Be prepared *before* disaster hits with a DR plan**. One of the best things you can do to protect your SQL Server is to include it when you create your DR plans.

Appendix A: Terminology

There are several concepts and terms that are important to understand as you learn how to take advantage of all of Cohesity's features for SQL database protection.

- **Protection Group**. A collection of objects from your registered sources that share a recurring backup schedule of Protection Runs. Use a Protection Group to identify which SQL databases to protect. When you create a Protection Group, you associate it with a Cohesity Protection Policy.
- **Protection Policy**. A reusable collection of settings that define how and when objects are backed up, replicated, and archived.
- **Cohesity Replication**. Replication automatically makes copies of snapshots captured by Protection Runs on one Cohesity cluster and puts the copies on a second Cohesity cluster.

There are three types of database backups that can be taken with SQL VDI: *full*, *differential*, and *log*. Having a combination of these backup types on hand will protect your SQL database.

- **Full Backup**. The full backup is a complete backup of a database. The full backup contains all the data in a database and can be used to do a complete restore of the database to the point in time that the full backup completed.
- **Differential Backup**. Used only in conjunction with a full backup, a differential backup specifies that the backup file should consist only of changes in the database since the last full backup. A differential backup typically takes up far less space than a full backup. Note, however, that a differential backup is not independent and must be based on the latest full backup of the data. That means there must be a full backup as a base, then the differential can be applied. Optionally, you can then also use log backups to bring it to the appropriate point in time.

  NOTE: In Cohesity, a VDI-based *differential* backup is referred to as an *incremental* backup.

- **Log Backup.** The transaction log backup is a sequential set of backup files. They comprise a record of all the transactions that have been performed against the database since the transaction log was last backed up. With transaction log backups, you can recover the database to a specific point in time.

  Each log backup captures that part of the transaction log that was active when the backup was created, and it includes all transactions that were not backed up in a previous log backup. An uninterrupted sequence of log backups contains the complete (*unbroken*) log chain of the database.

Appendix B: Product Documentation

Review our SQL Server product documentation for in-depth details:

- MS SQL Requirements
- Cohesity MS SQL Best Practices
- Key Concepts
- Protect MS SQL (VDI-based)

Your Feedback

Was this document helpful? Send us your feedback!

About the Authors

Scott Lorenz is a SQL Solutions Engineer at Cohesity. In his role, Scott focuses on business-critical applications, MS SQL Server databases, cloud storage, and enterprise data protection. Scott has over 26 years' experience as an enterprise DBA.

Other essential contributors included:

- Bart Abicht, Senior Technology Writer and Editor

Document Version History

VERSION	DATE	DOCUMENT HISTORY
1.0	April 2021	Original document
1.1	July 2024	Republished

ABOUT COHESITY

Cohesity radically simplifies data management. We make it easy to protect, manage, and derive value from data -- across the data center, edge, and cloud. We offer a full suite of services consolidated on one multicloud data platform: backup and recovery, disaster recovery, file and object services, dev/test, and data compliance, security, and analytics -- reducing complexity and eliminating mass data fragmentation. Cohesity can be delivered as a service, self-managed, or provided by a Cohesity-powered partner.

Visit our website and blog, follow us on Twitter and LinkedIn and like us on Facebook.