

**Microsoft Fabric**  
**Business Continuity Disaster Recovery Accelerator**  
**User Guide**

## Contents

|  |    |
|--|----|
| Introduction .....   | 3  |
| Prerequisites .....  | 3  |
| Overview of DR.....  | 4  |
| Reliability in Microsoft Fabric .....  | 6  |
| Additional Considerations.....   | 8  |
| Accelerator Overview .....   | 9  |
| Scope & Limitations .....  | 11 |
| Setup and execution .....  | 12 |
| Overview.....  | 12 |
| Stage 1: Fabric deployment with DR featured enabled on capacity C1 .....     | 13 |
| Stage 2: Generate recovery metadata in a recovery workspace .....            | 13 |
| Stage 3: Create a recovery workspace and import the recovery notebooks ..... | 14 |
| Stage 4: DR execution commences .....  | 15 |
| Stage 5: Recreate workspaces .....   | 16 |
| Stage 6: Reconnect the workspaces to Git and sync items .....                | 16 |
| Stage 7: DR Scenario - recover lakehouse files and tables .....              | 17 |
| Stage 8: DR Scenario – prepare for warehouse recovery .....                  | 18 |
| Stage 9: DR Scenario – recovery of warehouse data.....                       | 18 |
| Stage 10: DR Scenario – workspace roles applied.....                         | 19 |
| Summary .....  | 20 |

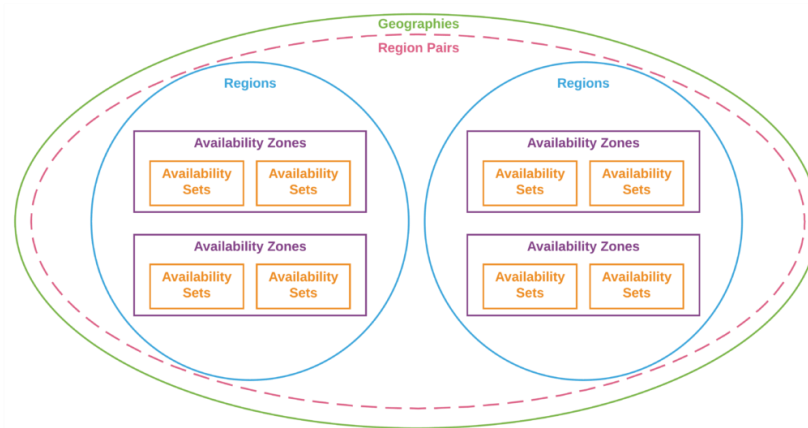
## Introduction

This document provides a high-level overview of disaster recovery topics (DR) and guidance on how to use this community-based business continuity disaster recovery (BCDR) accelerator.

Beginning with a brief introduction of the disaster recovery functionality in Microsoft Fabric, the associated set of notebooks demonstrate how to recover workspaces, Git supported items, lakehouse and warehouse data, in the event of a regional disaster.

## Prerequisites

- Required reading and baseline knowledge:
  - [BCDR concepts](#) including terms such as RPO and RTO
  - Understanding of [Azure geographies and regions](#).



- [Azure cross-region replication](#)
- Review and understand the [OneLake's standard disaster recovery features](#) and [cross-region disaster recovery](#)
- The [disaster recovery capacity setting](#) and the [steps / phases during DR](#)
- The [experience specific DR guidance for Fabric](#)
- Recommended reading and general understanding required:
  - [Fabric Pricing](#): Capacity & OneLake storage charges with/without BCDR
  - [Azure reliability documentation](#)
  - [Availability](#) (HA) vs [DR features](#) in Fabric
  - How [backups may be used in a DR scenario](#)
  - Microsoft [recommendations](#) from the Well-Architected Framework reliability pillar.
  - Understand the [availability zone support](#) for each Fabric workload per region
- Required to run this accelerator:
  - [Enable](#), and access to, Microsoft Fabric:
    - Either sign up for a [Fabric trial](#) or
    - [Purchase a Fabric capacity](#) in one of your subscriptions (F2 sufficient for demonstration purposes) or
    - Sign up for a [M365 Trial tenant with Fabric in trial mode](#) with 25 E5 licenses - not-recommended, unless truly necessary

- Decide which region will act as the primary region and which will be the target DR region. They do not have to be in the same geography but [region pairs](#) are a common choice. **Note for demonstration purposes you do not necessarily need to use multiple regions or multiple capacities** as a single region and capacity is sufficient. For multi-nationals, they could have multiple primary sites in various geos.
- Permissions to
  - create or use an existing capacity in the primary region to store recovery metadata (F2 is sufficient for demonstration purposes) with permissions to assign the capacity to a workspace (either capacity admin or [contributor permissions at capacity level](#))
  - create or use an existing capacity in the DR region
  - create two workspaces: this step is described in more detail later but ensure you have permissions to create and assign the workspaces to the chosen capacities/capacity. If you created a trial capacity above, ensure you assign the trial capacity to the workspace in the [License Info/Config section](#) of the workspace settings. Each capacity, trial or F SKU comes with a default starter (Spark) pool which can be used to run the associated notebooks.
  - permission on each workspace which you wish to recover in order to obtain necessary metadata
    - Contributor to extract git connection information
    - Admin role to extract assigned roles
  - create a lakehouse in each of the workspaces using a [non-viewer workspace role](#)
- Workspace items which you wish to recover must be stored in Git (either Azure DevOps or Github are currently supported by Fabric) i.e. the workspace must be [connected to a Git branch](#) and items [committed](#) to the repository. *Note: only workspace admins can manage connections to the Git repo.*

## Overview of DR

“DR is the process of recovering from a disaster and restoring business operations to a normal state. DR is a subset of BC, which is the process of maintaining business functions or quickly resuming them in the event of a major disruption.” <sup>[1]</sup>

Disaster recovery in cloud computing involves recovering from high-impact events like natural disasters that may affect networks and hardware across multiple regions, resulting in down time and even data loss. For BCDR, Microsoft uses a [shared responsibility model](#) in which both the cloud provider and the customer have distinct responsibilities to ensure the overall resilience and recovery of data and services.

For instance, in the context of cloud services such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), the division of responsibilities varies. Particularly for IaaS, the cloud provider typically manages the physical infrastructure, network, and hypervisor, while the customer is responsible for the operating system, applications, and data. For SaaS, the customer is primarily responsible for application/analytics code/workloads and data. This means that while the cloud provider ensures the availability and security of the underlying infrastructure, the customer must

consider their own disaster recovery strategy for their specific requirements (SLAs, RPO, RTO) and their applications, analytics and data.

Every organisation should have a DR strategy. Depending on the industry and nature of the data, some may have more stringent requirements such as financial institutions with legal and regulatory obligations. Maintaining a disaster recovery plan, on an ongoing basis, is crucial for a reliable DR strategy. The plan should be a dynamic document that is reviewed and updated regularly as the IT environment and estate changes.

Part of the recovery plan should detail how applications, analytics and data will failover to the chosen DR region - which may be different to the cloud provider's secondary or paired region. All supporting workload/code artifacts and automation scripts (such as this accelerator which support the failover process) should be stored in a highly available, highly resilient version control service such as Azure DevOps or Github. Data is often replicated to a similar storage engine in the DR region but may require further recovery to fully enable the DR environment as the new "primary". In some cases, customers may opt for an active-passive DR solution, balancing cost effectiveness and recovery times (RTO). In this scenario, the DR environment is regularly updated with metadata and data. In terms of analytics and Fabric, this could be workload specific metadata (e.g. pipelines, notebooks, lakehouses etc) as the source systems change in order to reduce the time needed to configure or update the recovery environment. When not in use, this Fabric environment can be "turned off" by pausing the capacity to reduce unnecessary costs. Recovering data from the paired region (BCDR feature geo-replicates the data to the paired region) may often take the longest time to recover, so if recovery-time-objectives (RTO) are to be minimised, then data may need to be recovered regularly but at higher overall costs due to both the additional compute (consumption) and storage charges required to achieve this.

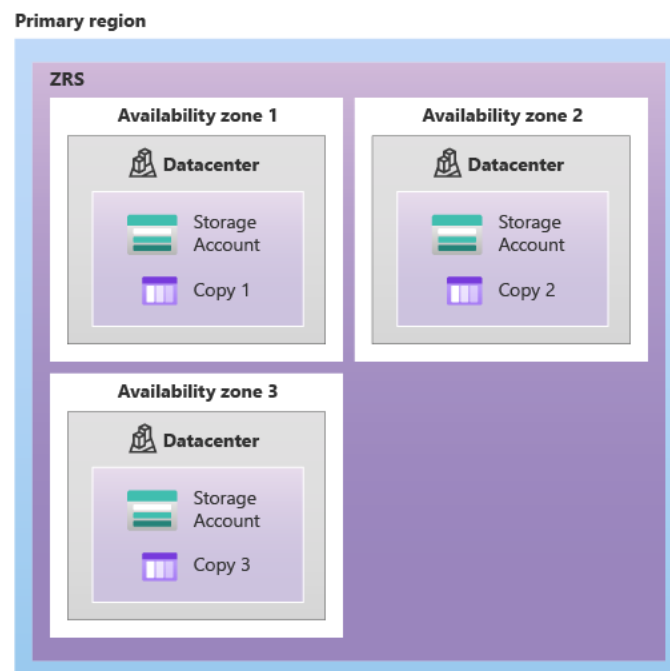
As part of the DR planning process organisations should determine which DR region to failover to. This decision may be influenced by a number of factors which include but are not limited to:

- regulations and compliance requirements which may restrict the movement of data across geographic boundaries. For instance, data sovereignty laws in certain countries mandate that data must remain within the country's borders. This means that organisations operating in such regions may not failover to a data center located in another country.
- Compute service (Fabric capacity SKUs) availability or [Fabric region availability](#). For example, the [documentation recommends](#) selecting a region outside of the primary geography to increase likelihood of capacity availability. This could be due to high demand placed on other regions in the same geo when a disaster has affected an entire region.
- [Data egress charges](#) of recovering the data when the DR region is not the same as the paired region. In the case of (cloud provider) managed replication the data is replicated to the designated paired region, and in order to avoid additional charges - especially if the volume of data is significant – organisations may decide to recover in the paired region i.e. make the DR region the same as the paired region to avoid copying the data out of the paired region. This decision needs to be considered in light of the compute service availability recommendation above which ultimately puts the DR region outside of the paired region geography.

Disaster Recovery (DR) plans should be tested regularly (DR drills) to ensure they are effective and up to date. The frequency of testing can depend on various factors, including the size of the organization, the complexity of its IT estate, and regulatory requirements. All plans involve a combination of people, process and technology, and as far as possible, the technology component should be automated to reduce error-prone manual involvement. Automation also enables the process to be repeated and tested regularly where the outputs of execution (timing), integrity (data) and validation (success or failure) can be continuously monitored (reports) for effectiveness and accuracy.

### Reliability in Microsoft Fabric

In terms of high availability, Fabric currently offers zone-redundant support for [certain workloads in certain regions](#). Subject to capacity availability, during a zone wide outage, no action would be necessary (apart from Spark jobs which need to be resubmitted) as the services will self heal and rebalance. All data in OneLake will be [zone redundant in certain regions](#) (regardless of DR setting – see below) offering 12 9's of durability (over a given year) as the data is synchronously replicated across three Azure availability zones.



OneLake also offers a soft delete feature that helps prevent accidental deletions. Any deleted data can be recovered within 7 days using code.

Disaster recovery functionality built-in to Fabric is enabled at capacity level. All OneLake data stored in workspaces assigned to these capacities is asynchronously copied to the cloud provider's secondary region, known as geo-redundancy. Note, the choice of the secondary region is determined by Azure's standard region pairings and due to the asynchronous nature of replication there is a [risk of potential data loss](#).

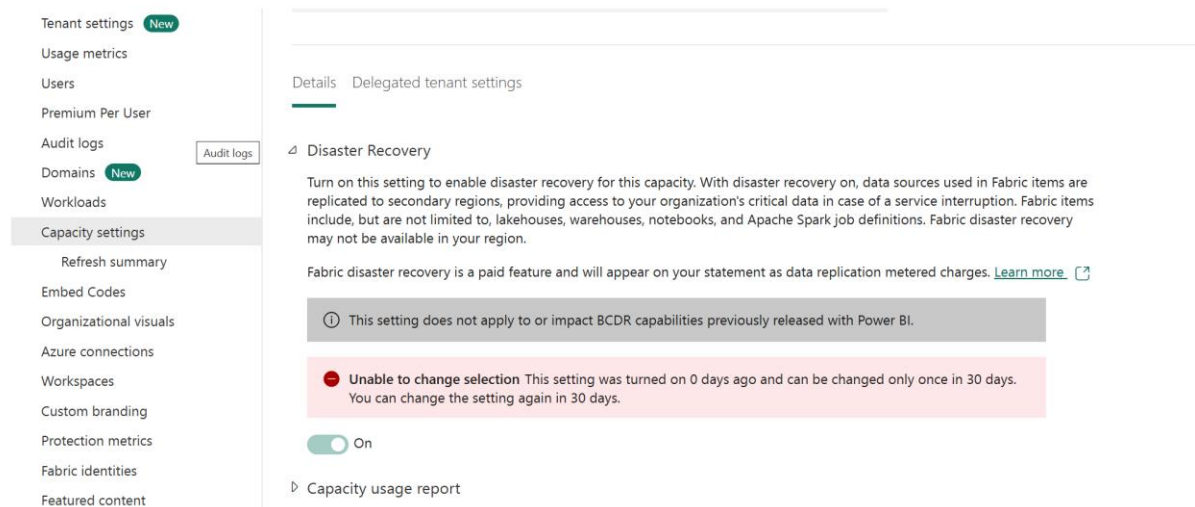
If a disaster makes the primary region unrecoverable, the OneLake service may initiate a regional failover. Once the failover completes, customers can use OneLake's APIs through the [global](#)

[endpoint](#) to access their data in the secondary region. After a failover, the new primary data center will have local redundancy only.

Failing back to the original primary may (once it has been restored) involve executing the DR plan again (in the other direction) but this is beyond the scope of this document or the accelerator.

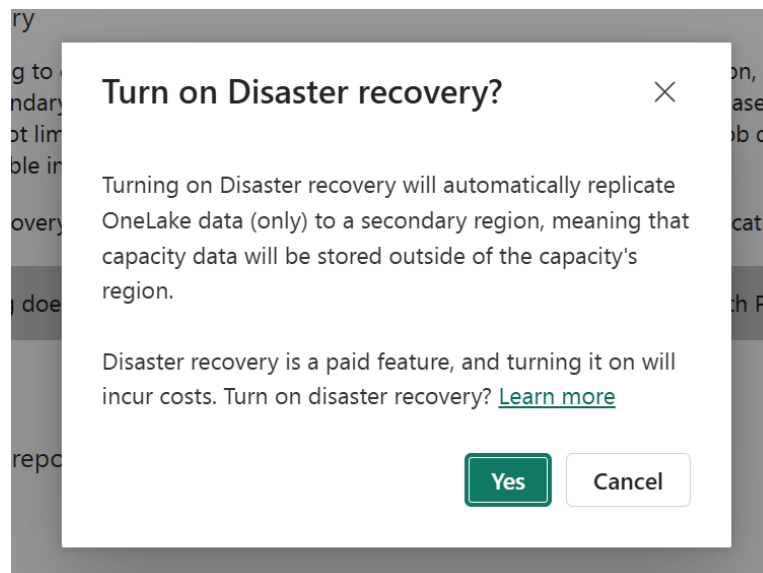
To enable Fabric DR functionality for a specific capacity navigate to the Capacity Settings menu in the Admin Portal.

#### Admin portal



The screenshot shows the Admin Portal interface. On the left is a navigation menu with items like Tenant settings, Usage metrics, Users, Premium Per User, Audit logs, Domains, Workloads, Capacity settings (highlighted), Refresh summary, Embed Codes, Organizational visuals, Azure connections, Workspaces, Custom branding, Protection metrics, Fabric identities, and Featured content. The main content area is titled 'Details Delegated tenant settings' and shows the 'Disaster Recovery' section. It contains a description of the feature, a note that it's a paid feature, and a toggle switch that is currently 'On'. There are also informational messages: one stating the setting doesn't apply to BCDR capabilities, and another stating it was turned on 0 days ago and can only be changed once in 30 days.

Note: The DR setting can only be change once in 30 days.



Enabling DR incurs additional OneLake [storage charges](#) and [CU consumption](#) and will take up 72 hours to start replication.



Disaster recovery is turned on  
Data replication will start in up to  
72 hours.



### Additional Considerations

An analytics platform is seldom static and is continuously undergoing changes as new data is ingested and curated, new models and reports are developed. Customers are therefore responsible for testing and validating the results of the process on a regular (scheduled) basis, continuously monitoring the results and adapting where necessary to satisfy their requirements and provide confidence to the business that their DR solution supports their business continuity objectives.

DR obviously extends far beyond the analytics platform to the source systems (applications and databases) and services on which the analytics platform depends for an ongoing supply of data. Customers often overlook the importance of upstream sources, forgetting that a Disaster Recovery (DR) plan is only effective if it comprehensively covers both the operational and analytical estate.

The [Microsoft documentation](#) describes that in order to recover lakehouse and warehouse data in a DR scenario, the customer should copy the data using the ABFS path (pointing to the original lakehouse / warehouse) to a new lakehouse/warehouse in a new workspace which has been assigned to a new capacity in the DR region. Whilst the lakehouse recovery is relatively straightforward - copying data from the previous OneLake endpoint to the Tables and Files section of the new corresponding lakehouse - the warehouse recovery process is more complicated. It requires creating a temporary lakehouse so that warehouse data can be copied from the OneLake endpoint into this lakehouse, so that insert into statements can be run to copy the data into warehouse tables. Unfortunately, this solution is not possible to automate therefore this accelerator uses a different approach using pipelines which is discussed in more detail in stage 8 and 9 later.

Refencing the documentation again, the scenario described in the documentation refers to recovering one workspace (manually) however customers often have hundreds if not thousands of workspaces, so repeating this process manually would be tedious and error prone if conducted workspace by workspace. To this end, **one of the original goals of this accelerator is to demonstrate a minimum set of steps (APIs) required to automate the recovery lakehouse and warehouse data across multiple workspaces, with stated limitations above. It should by no means be regarded as a final DR solution, and requires the customer to further enhance and adapt to suit their environment and their requirements.**



## Accelerator Overview

The current version of this accelerator provides a baseline automated process which recovers certain Git supported Fabric items (such as pipelines, notebooks, lakehouses, warehouses etc) and associated data in another region. It is designed to demonstrate to customers the very minimum set of steps which can be automated (using the rest APIs or SDKs) to facilitate recovery.

### Note:

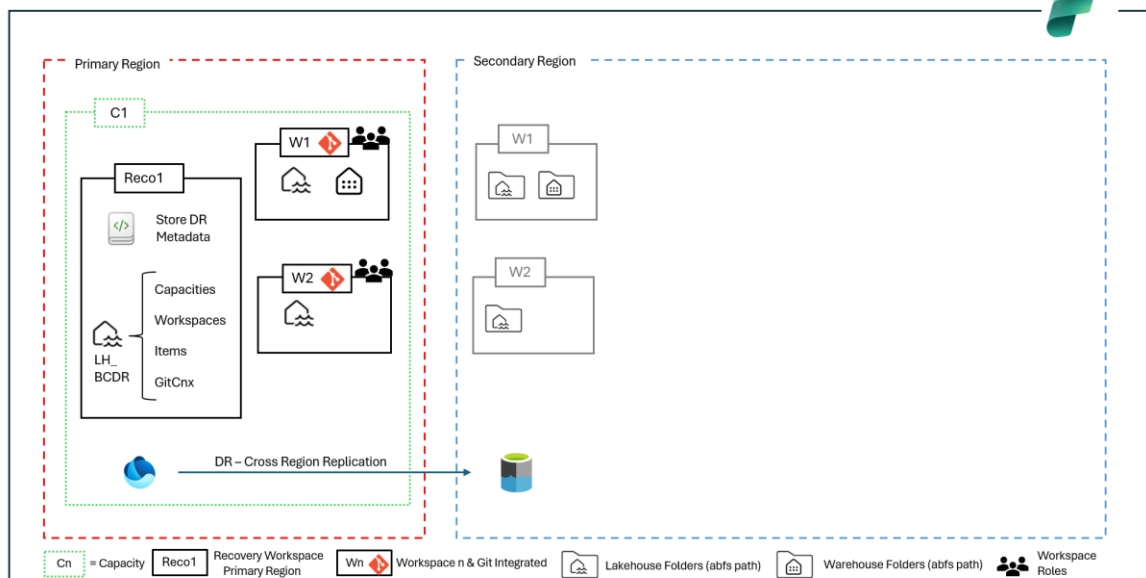
This accelerator has not yet been tested running as service principal or managed identity.

The associated notebooks will be run using the user's credentials therefore will need to be run with by a user with sufficient (elevated) permissions to each workspace you wish to recover. Please see pre-requisites for more information.

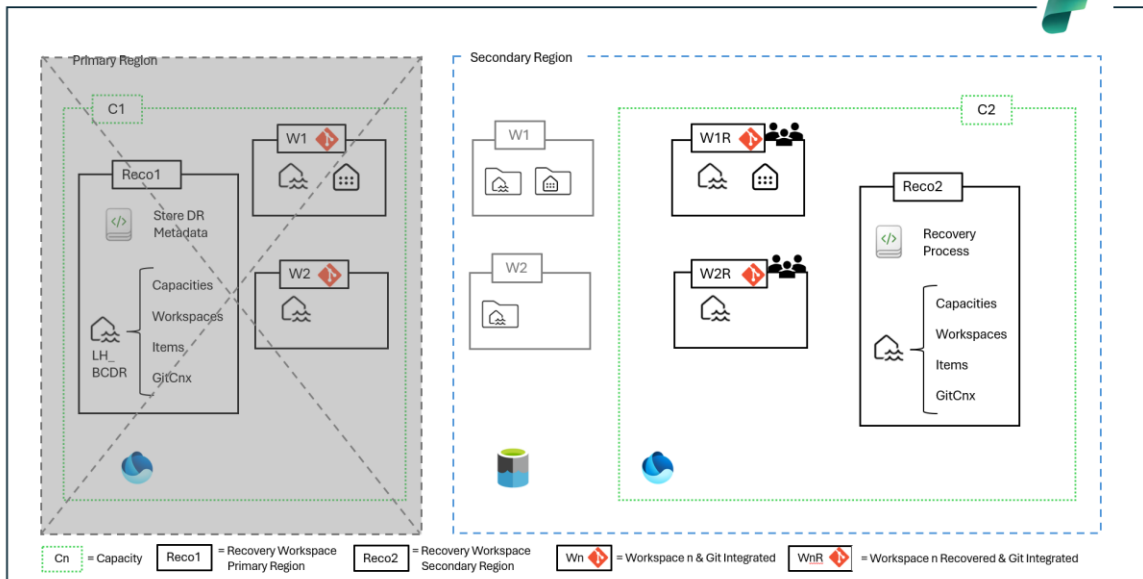
In real-world scenarios either the tenant admin can be used, or service principal with required permissions or sets of workspace admins.

The two images below depict the environment state, prior and post running the recovery solution. *For simplicity, the secondary and DR regions are depicted in the same region however, in practice, these are not necessarily in the same region, depending on considerations mentioned above and customer preference.*

Before: DR scenario with DR setting enabled + recovery metadata



After fail-over and recovery – Fabric items and data recovered. Workspace roles applied



The accelerator consists of three notebooks, one which is run in the primary region (01 – Run in Primary) and another which is run in the DR/secondary region (02 – Run in DR). The third notebook (workspaceutils) contains utility functions utilised in both notebooks.

#### 01 – Run in Primary.ipynb

Gathers metadata about the current primary environment and stores this as tables in OneLake. This metadata forms part of the replicated DR data and will be recovered in the DR environment in subsequent stages.

#### 02 – Run in DR.ipynb

Performs the recovery steps such as recreating workspaces, connecting them to Git, syncing and restoring items (lakehouses, warehouses, pipelines etc) and applying workspaces roles.

#### Workspaceutils.ipynb

A utility notebook with many useful supporting functions such as extracting and storing metadata, copying data, and more, which is included (using [%run](#)) in the other two notebooks to reduce the amount of code in each main notebook and for code re-use.

## Scope & Limitations

This document and accelerator does not cover or utilise Power BI multi-geo features and focuses primarily on Fabric (non-Power BI) items and OneLake data.

Recommendations on Fabric capacity size required to perform the recovery is beyond the scope of this document. Various capacity sizes should be tested by the customer and may be dependent on the volume of data to recover and the RTO. Particularly, the recovery of warehouse data with this accelerator use pipelines (which can run concurrently) and could consume significant CUs if there are many warehouses to recover and many large tables.

This accelerator should be considered as a “minimum viable accelerator” which provides the basis of a DR recovery process (notebooks) run in Fabric Spark. The primary focus of this accelerator is to demonstrate the recovery of lakehouse and warehouse data, but additional functionality may be added over time.

This solution relies on both [Git Integration](#) and OneLake DR functionality to perform a recovery of workspaces, Git supported Fabric items, lakehouse and warehouse data and workspace roles.

Limitations are set out below and are either due to a lack of supported (documented) APIs, Git Integration support or features not yet implemented in the accelerator.

### Limitations:

- Capacities are not created as part of the process - coming soon.
- The recovery notebook (02 – Run in DR) only parameterises one capacity used for all new workspaces in the DR region. Support for multiple capacities and workspace assignment is coming soon.
- Warehouses containing views using lakehouse tables fail due to a known Git support issue whereby the warehouse schema is not recreated. This causes the copy pipeline to fail – as a workaround redeploy the schema (using SQL Projects or scripts) prior to running the copy pipeline.
- Default semantic models are not recovered (not yet supported by Git)
- Reports based on default semantic models will not work in the DR region due to the limitation above.
- Reports are not rebound to the datasets in the DR region - coming soon.
- Connections strings for any connections or gateways are not updated in the DR region.
- Data gateways (both vNet and On-prem) are not recreated in the DR region.
- Any Fabric items not yet [supported by Git Integration](#)

- Note there a known issue when Git tries to deploy the data warehouse schema if there are objects which depend on lakehouse items e.g. a view which queries a table from the lakehouse will cause the Git update to fail when it tries to recreate the schema in the target warehouse. The recommendation for the purposes of demonstration is not include such objects in your warehouses when syncing to Git and if customers face this issue today they will need to add a step in the pipeline which recreates the schema via a script prior to copy data activity.
- Shortcuts are not recovered - APIs are not available yet.
- Lakehouse schemas are not supported yet.
- Item level permissions (direct shares) are not supported yet.

## Setup and execution

### Overview

Prior to running this accelerator, the customer will need one or more (new or existing) Fabric capacities. For simplicity and demonstration purposes, the same capacity can be used to run both primary and DR notebooks. Alternatively, a capacity is required in the primary region and another in the DR region of choice.

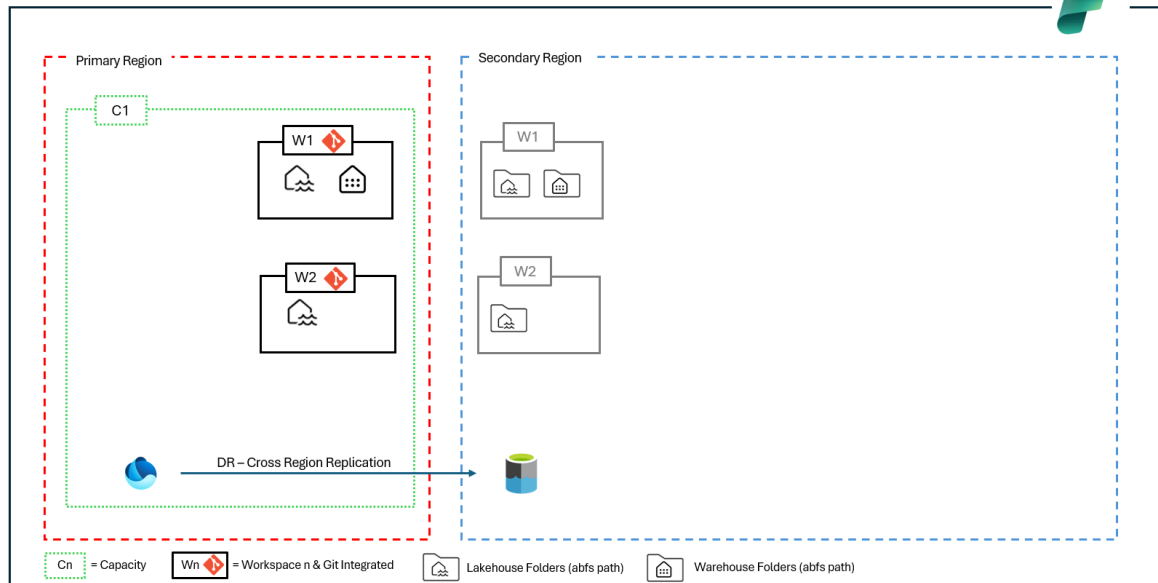
The following diagrams attempt to visually represent the stages throughout the process, which excluding setup, often equates to the execution of associated cells in the notebooks.

*Note: the naming conventions in the diagrams are for guidance only and do not need to be followed exactly as they are parameterised in the notebooks, but good naming conventions are always a best practice, and will make it easier to identify the item's purpose in future.*

Stage 1 below requires nothing more than enabling the DR switch for C1 (or chosen capacity for the primary region and any other capacities assigned to workspaces you wish to recover)

## Stage 1: Fabric deployment with DR featured enabled on capacity C1

### Stage 1: Fabric deployment with DR enabled



The image above depicts a simple fabric environment with a single capacity which has the DR feature enabled to ensure OneLake data is automatically geo-replicated to the secondary region. In this hypothetical scenario the customer has two workspaces -workspace (W1) with a lakehouse and a warehouse and another workspace (W2) with only a lakehouse.

*Note: For demonstration purposes configure a similar environment and for simplicity use the get sample data options built into lakehouse and warehouse to populate them with data.*

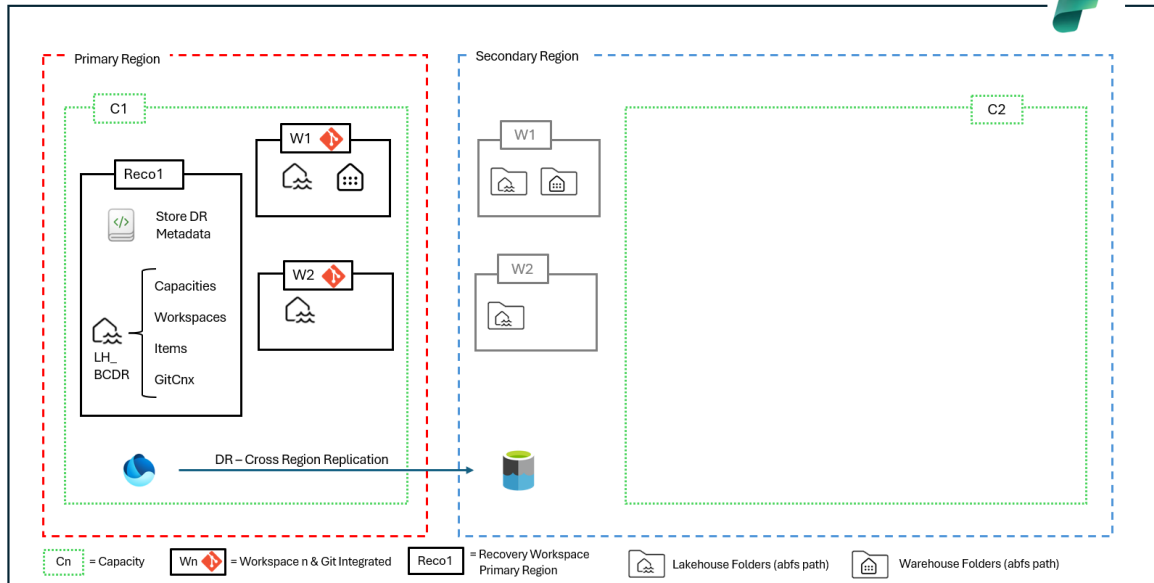
The workspaces to be recovered must be [connected to Git](#). For demonstration purposes use Azure DevOps in the same tenant as your Fabric environment, and connect each workspace to a different branch either in the same project or different projects. In order to recover these items and their associated data in the event of a disaster there is certain metadata or information about the Fabric environment required beforehand (gathered in stage 2 below), and therefore it is crucial that prior to any disaster scenario, this information is stored in a workspace where the capacity has the DR feature enabled. If not enabled, this metadata must be backed-up in another region. *The latter scenario is not supported by this accelerator.* This metadata contains information about all the workspaces, capacities, Git connections, items and more. Other metadata such as reports, models etc will be useful for other DR activities in the future.

## Stage 2: Generate recovery metadata in a recovery workspace

Create a new workspace such as Reco1 and import notebooks 01 – Run in Primary and workspaceutils. Open 01 – Run in Primary (ensure capacity C1 is running) and add a new lakehouse called LH\_BCDR (or appropriately named lakehouse used for storing recovery metadata) ensuring that it is the default lakehouse for the notebook. Running all cells in the notebook successfully will extract the required recovery metadata, saving the output in lakehouse tables.

At this point, run queries against the tables to familiarise yourself with their contents.

## Stage 2: Store recovery metadata in a recovery workspace & prepare DR capacity in the DR region

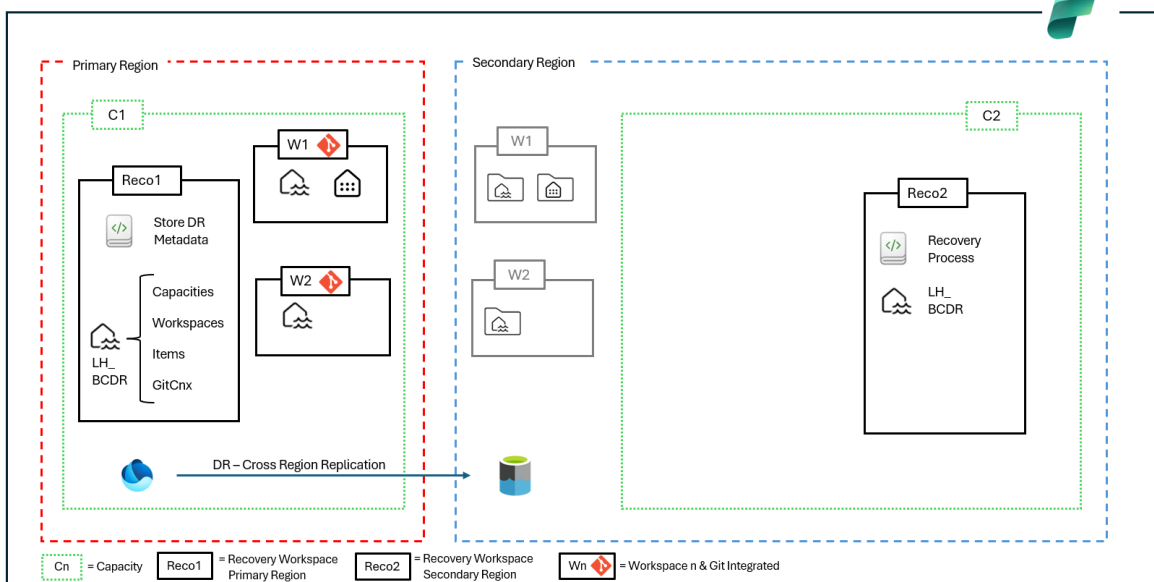


*Note: In real-world scenarios, it is recommended that notebook 01 – Run in Primary is scheduled to run on a regular basis to keep this crucial metadata up to date as the environment changes. This will reduce the RPO in terms of environmental metadata, not the data which is constantly being (asynchronously) geo-replicated so will already have an extremely small RPO.*

Finally, ensure at least one capacity (C2) is created in the DR region which will be used to run the recovery process. Whilst not in use, this DR capacity can be paused to reduce unnecessary costs.

## Stage 3: Create a recovery workspace and import the recovery notebooks

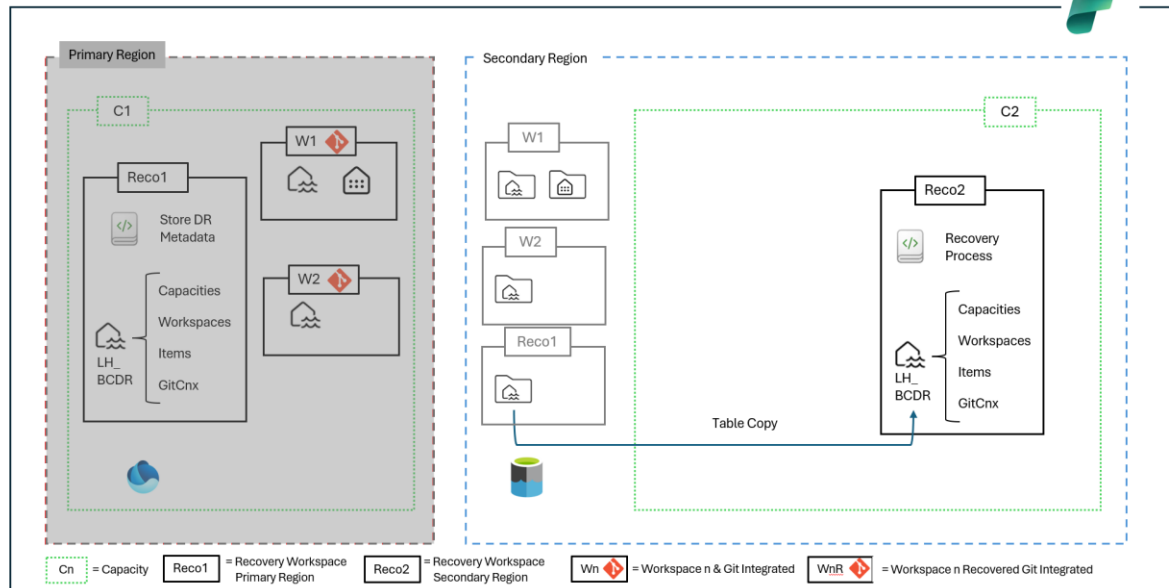
### Stage 3: Create a recovery workspace and import the recovery notebooks mounted to a new lakehouse



Create a new workspace called Reco2 and import notebooks 02 - Run in DR and workspaceutils. Create a new lakehouse as the default lakehouse called LH\_BCDR.

## Stage 4: DR execution commences

Stage 4: DR Scenario - run the initial cells to recover the primary metadata tables



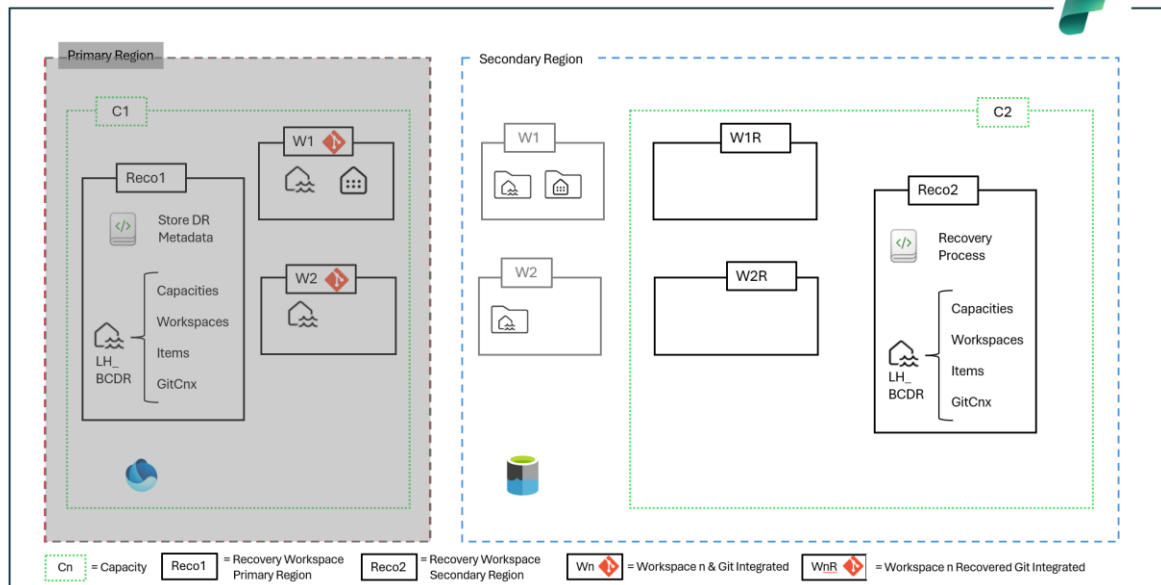
In this stage we assume a disaster has occurred and it is time to recover the primary environment in the DR region. The capacity or capacities in the primary region are no longer running, and in the worst case scenario, all Fabric workloads and data is no longer accessible through the Fabric UI.

The recovery process is started by setting the parameters in the notebook 02 – Run in DR and recovering the crucial recovery metadata which is copied from the DR storage location (using abfs paths) and saved in the recovery lakehouse (C2.Reco2.LH\_BCDR) as tables. With the recovery metadata available the subsequent steps in the notebook will recover workspaces and all Git supported items and associated lakehouse and warehouse data.

*Note: During demonstration and testing this DR scenario, the abfs path will still resolve to the primary (abfs) OneLake storage location.*

## Stage 5: Recreate workspaces

### Stage 5: DR Scenario – recreate workspaces

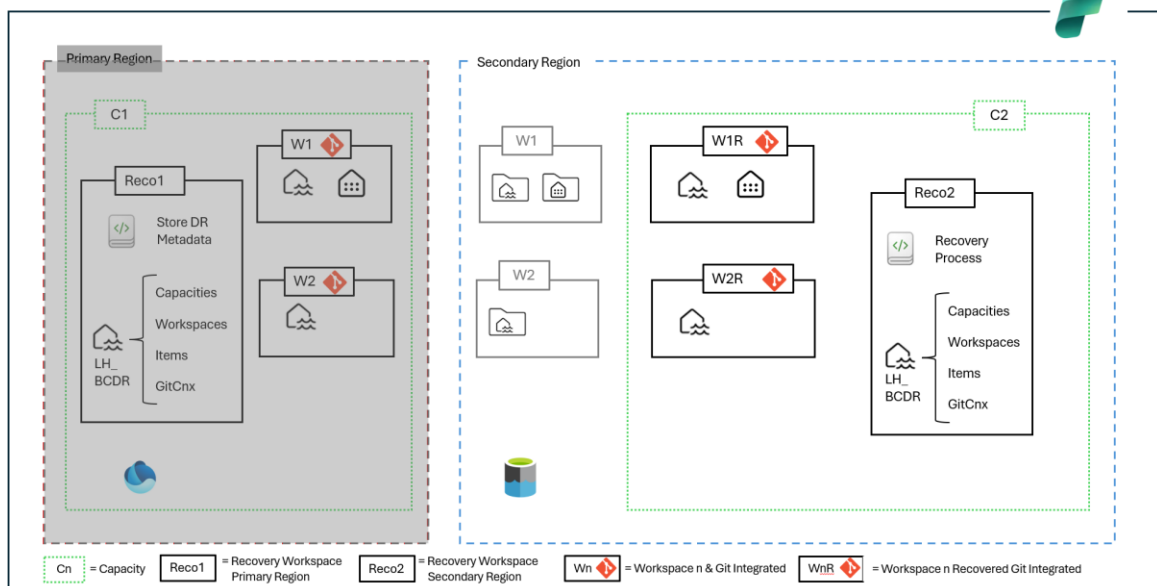


The first step is to recreate the workspaces with a given suffix in order to avoid name clashes with the original workspaces, but each workspace will be somewhat of a mirror image or a clone of the corresponding workspace in primary.

## Stage 6: Reconnect the workspaces to Git and sync items

Now that the workspaces have been “cloned” in our DR environment, we use the Git connections recovery metadata (lakehouse table) to determine which repository & branch each workspace should be assigned to. Once connected the notebook cell will perform a Git sync and restore all the Fabric items which are Git supported and were committed to the respective branch when the primary environment was operational.

### Stage 6: DR Scenario - reconnect the workspaces to Git and sync items



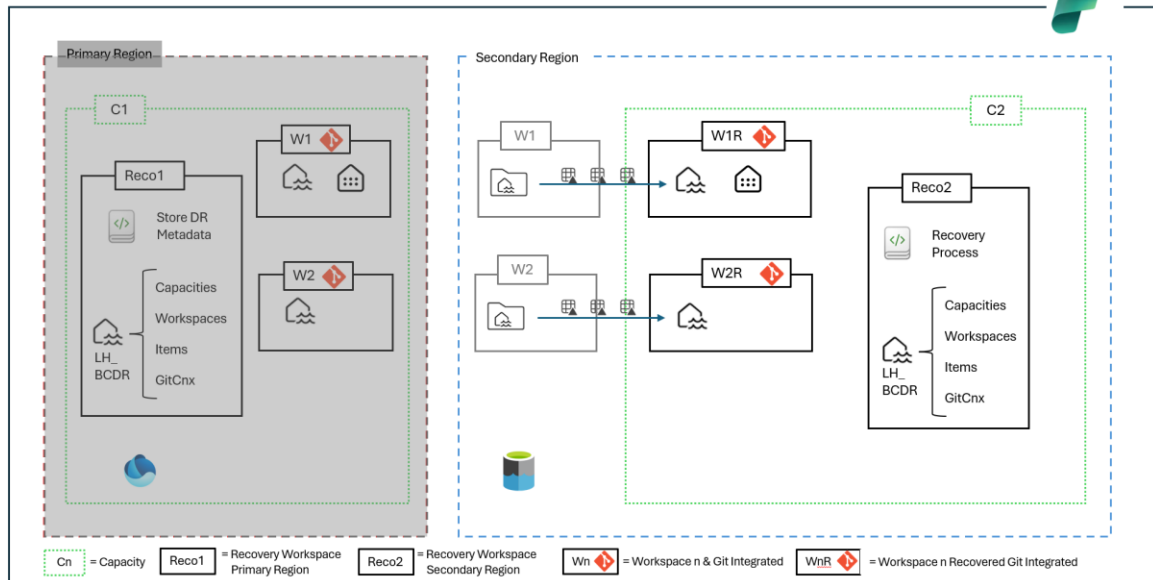


At this point the lakehouses and warehouses (and other items) have been recreated but contain no data. Lakehouse have no files or tables, and the [warehouse schema will be recreated](#) but the tables contain no data.

### Stage 7: DR Scenario - recover lakehouse files and tables

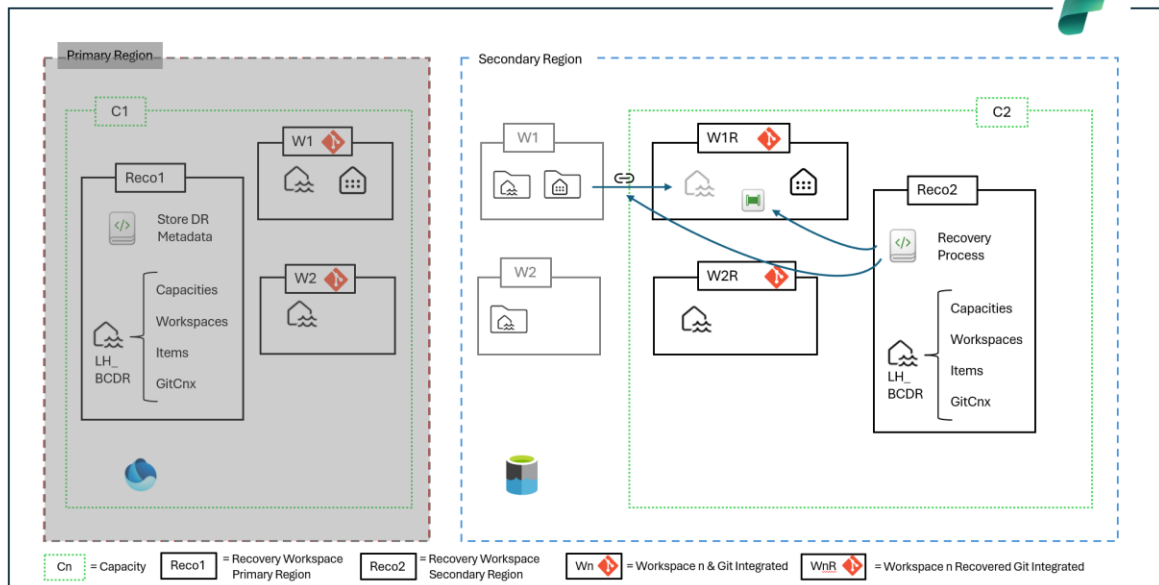
In stage 7, lakehouse data is recovered by copying the lakehouse data from the storage (abfs) endpoint to the corresponding (new) target lakehouse in the DR region.

Stage 7: DR Scenario - recover lakehouse files and tables



## Stage 8: DR Scenario – prepare for warehouse recovery

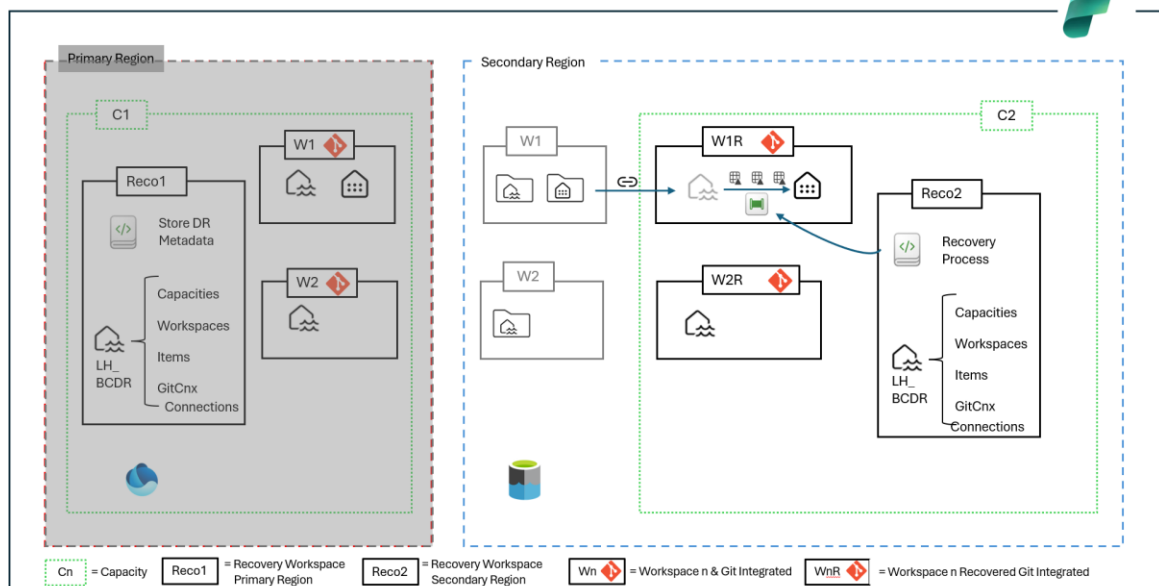
### Stage 8: DR Scenario - setup warehouse recovery – create shortcuts and inject copy pipeline



In stage 9 the warehouse data will be recovered but before this can be done, an interim lakehouse is required per warehouse to make the data accessible via the SQL endpoint of this lakehouse. Shortcuts are programmatically created in this interim lakehouse to expose schemas and tables (in the format of schema\_table) from the original warehouse. A parameterized pipeline is injected into each workspace per warehouse (with the name of [pipeline prefix]\_[workspace name]\_[warehouse name]) that will copy table data from these shortcuts into the target warehouse in the DR region.

## Stage 9: DR Scenario – recovery of warehouse data

### Stage 9: DR Scenario - recover the warehouse data



The injected pipeline(s) is/are invoked to copy the data into their respective target tables. The call is asynchronous meaning the notebook cell will successfully complete having triggered the pipelines to start however the pipelines can take a while to complete. Navigate to the Monitoring page to verify

whether the pipelines have completed successfully. Use the recovery pipeline prefix to filter the results and select column options to add other useful columns such as duration.

**Monitor**

View and track the status of the activities across all the workspaces for which you have permissions within Microsoft Fabric.

[Refresh](#)  [Filter](#) [Column Options](#)

[Clear all](#) To apply filters, select the values from the Filter dropdown menu.

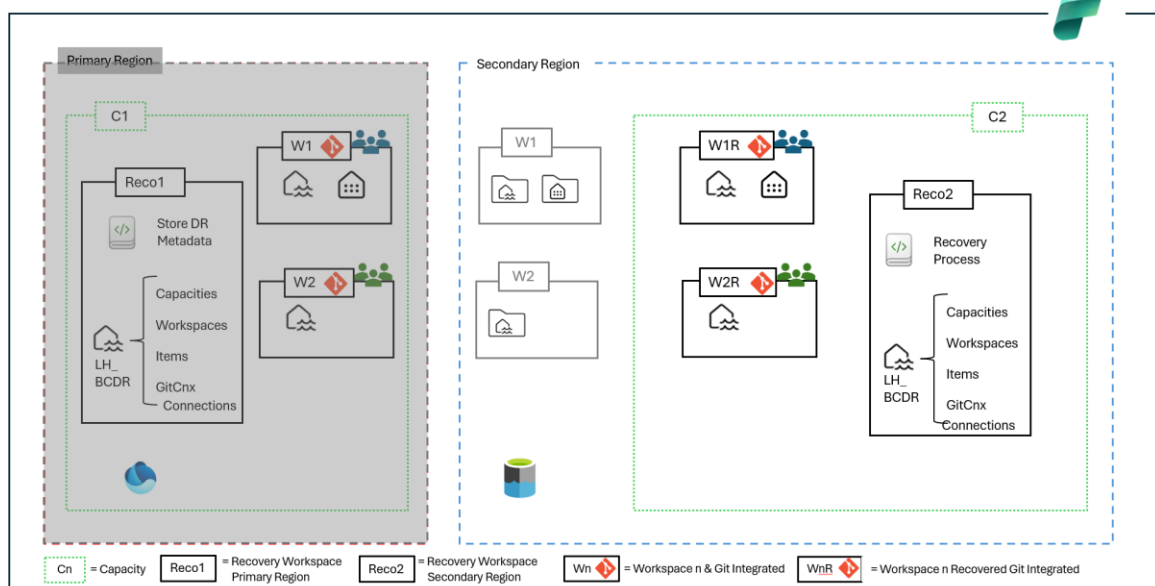
| Activity name                                | Status    | Item type     | Start time       | Location          | Duration |
|--|-----------|---------------|------------------|-------------------|----------|
| plRecover_Warehouse_Workspace1_for_BCDR_W... | Succeeded | Data pipeline | 7:41 AM, 8/27/24 | Workspace1_for... | 27m 14s  |
| plRecover_Warehouse_DEV_Gold                 | Succeeded | Data pipeline | 7:41 AM, 8/27/24 | DEV_SECONDARY     | 9m 15s   |

[Show more](#)

*Note: Depending how many warehouses are being recovered, the pipeline run could be deferred (manually run or scheduled in a staggered manner) based on how much capacity has been assigned to C2 (or the associated capacity with the warehouse workspace), to avoid reaching capacity limits and being throttled.*

## Stage 10: DR Scenario – workspace roles applied

### Stage 10: DR Scenario – workspace roles applied



In stage 10 the workspace role assignments are restored against the workspace “clones”.

### Future stages

- Currently, item shares are not recovered as there are no supported public APIs to retrieve and set item shares.
- Shortcuts will be saved and recreated when the supported APIs are published.
- Reports will be rebound to the new datasets and dataset connection strings will be updated.
- In the future delegated tenant settings will also be recovered.

## Summary

This document provided a general overview of business continuity and disaster recovery fundamentals in terms of cloud-based solutions and more specifically the reliability features in Microsoft Fabric which could support BCDR requirements. The remainder of the document provided an overview of the BCDR accelerator notebooks, their setup, content and execution guidance, including scope and limitations.

After running the accelerator, you should have successfully demonstrated the recovery of critical Fabric elements such as workspaces and assigned roles, lakehouses, warehouses and various Git supported code artifacts (pipelines, notebooks, spark job definitions etc). Whilst reports and non-default semantic models would also have been recreated in the new workspaces the reports may be unusable until they are rebound to their new respective semantic models (datasets).