

How to troubleshoot SQL Database restore request issues

Last updated by | Peter Hewitt | Dec 16, 2022 at 11:17 AM PST

Contents

- [Issue](#)
- [Investigation/Analysis](#)
- [Mitigation](#)

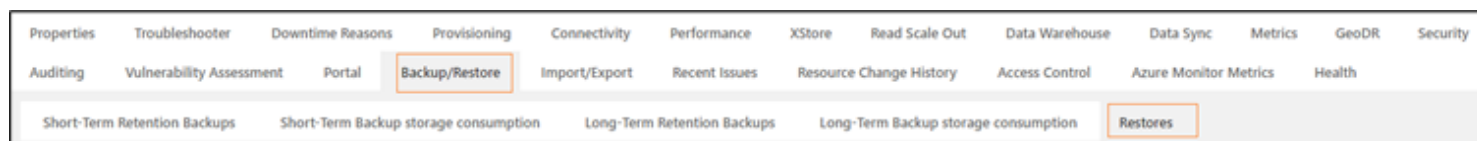
Issue

The purpose of this troubleshooting guide is not specific to any issue or customer scenario, but rather demonstrates how CSS engineers can investigate "Restore Requests" operations using our back-end telemetry tools e.g., ASC, XTS or Kusto.

Investigation/Analysis

ASC is the main tool where CSS engineers should start troubleshooting "Restore Requests" problems. The insights can help identify the issue, providing useful information such as the *restore_request_id*, the list of database restore operations and the progress of the restores.

To view the Restore requests for a specific SQL Database go to **SQL Server/Database -> Backup/Restore tab -> Restores** in ASC.



The "Restore Operations" section shows the historical restore requests in a table view with one restore request per line. It contains details that engineers can utilize in further troubleshooting steps.

Restore Operations

① All restore operations with DB as source or target

Drag a column header and drop it here to group by that column

start_time	end_time	operation_type	restore_request_id	state	SourceServer	SourceDatabase	source_edition
2022-11-20 23:00:20	2022-11-20 23:06:16	CreateRestoreReq...	4783E00E-3296-4CFF-87AE-2057E1B9F68B	Succeeded	sql-prod-uk-001	sqlldb-stg-jdx-standardbank-01	Standard

1 items per page

The "Restore Progress" tab shows the progress breakdown for a restore process.

Restore Progress

① Show latest 100 restore progress within the time frame

Drag a column header and drop it here to group by that column

originalEventTimestamp	restore_request_id	TargetServer	TargetDatabase	restore_database_progress	details
2022-11-20 23:05:14	4783e00e-3296-4cff-87ae-2057e1b9f68b	sql-prod-uk-001	sqlldb-stg-jdx-standardbank-01_AWS_26804_63804564016...	Report restore status Completed.	
2022-11-20 23:05:14	4783e00e-3296-4cff-87ae-2057e1b9f68b	sql-prod-uk-001	sqlldb-stg-jdx-standardbank-01_AWS_26804_63804564016...	Notification is sent successfully.	
2022-11-20 23:05:14	4783e00e-3296-4cff-87ae-2057e1b9f68b	sql-prod-uk-001	sqlldb-stg-jdx-standardbank-01_AWS_26804_63804564016...	Start sending notification.	
2022-11-20 23:05:14	4783e00e-3296-4cff-87ae-2057e1b9f68b	sql-prod-uk-001	sqlldb-stg-jdx-standardbank-01_AWS_26804_63804564016...	Notification is sent	

The XTS view *sterlingrestorerequests.xts* is the second main option to troubleshooting "Restore Requests" issues. From ASC, retrieve the *restore_request_id* which we will utilize later in the *sterlingrestorerequests.xts* view.

A good start point in XTS to track the restore request is using the *sterlingrestorerequests.xts* "Search string". Paste the *restore_request_id* in the search string to obtain more details.

Search string(RestoreId or SourceServer or TargetServer or Source LDB Id)

Search string

OK

The "Restore Requests" details include the information necessary to troubleshoot the issue further. Details to look out for are: *restore_id*, *status*, *source logical server name*, *logical database name*, *target server and database name*, *requested Point_in_time_restore*, *restore creation_time*, *target SLO* and *operation_detail_full*.

Restore Requests							
restore_id	state	source_logical_server_name	logical_database_name	operation_detail_full	source_logical_database_id	target_logical_server_name	target_logical_database_name
132084-c280-4c2d-9c38-b3c8b4f26823	Completed	copjo5ivt5-covetrusi6-d1-server	copjo5ivt5-covetrusi6-d1-db	The database has been restored successfully in 00:04:53.8994249.	5db38d77-d760-4c38-a53d-7ad3822a7f93	copjo5ivt5-covetrusi6-d1-server	restore-262128-copjo5ivt5-covetrusi6

In this example, the *operation_detail_full* shows that the restore database completed successfully in 4:53 minutes.

operation_detail_full
The database has been restored successfully in 00:04:53.8994249.

However, customers can complain that the restore request is takes much longer time, such as many hours to finish. For example, for a restore that took ~14 hours, how do we track down where the total of ~14 hours for the SQL Database was spent restoring? Is there any exception reported? What is on the SQL Database side? What is the SQL Database SLO?

From the XTS view we can retrieve additional helpful information, if not already in ASC, and use these details to run Kusto queries to investigate further. The *operation_request_id* is what we want to obtain from the XTS view before moving to run further Kusto queries. In the same XTS view, we can capture the MonManagement *operation_request_id* equivalent to the *restore_request_id* in **MonManagement – MO to RestoreId**.

MonManagement - MO to RestoreId		
operation_request_id	restore_request_id	
3CD02217-70C7-4795-8652-4DFCFF724895	C12D564C-C2BB-402D-9D36-B3E9B4F36825	

MonRestoreEvents | Sql ErrorLog in last hour | Restore Throughput | Customer Details | MonManagement - MO to RestoreId

In this XTS view, we can also capture details from Kusto tables such as **MonRestoreEvents**, **Restore Throughput**

The customer reported the restore request started at 8:51 PM UTC on 10/17/2022. From the **MonManagement – MO** progress tab in XTS view we notice there is a gap between 10/17/2022 8:52 PM to 10/18/2022 11:17 AM with no events or records:

10/17/2022 8:52:08 PM	experience_create_cached_region_traits_complete
10/17/2022 8:52:08 PM	fabric_service_creation
10/18/2022 11:17:52 AM	management_workflow_notify_restore_progress_begin
10/18/2022 11:17:52 AM	fsm_starting_request

In cases where the customer didn't share the "Restore Request", we can find this information by looking into ASC, or by querying the Kusto **MonManagement** table, searching for the target server and database name in *operation_parameters* column and reviewing the *operation_parameters* XML output. From here we obtain the *operation_request_id* (*request_id*) and the *restore_request_id* details.

```
MonManagement
| where operation_parameters contains "copjo5ivt5-covetrusi6-d1-server" and operation_parameters contains "res"
| where operation_type contains "Restore"
| project TIMESTAMP, request_id, operation_type, event, operation_category, operation_parameters
```

An example of the XML output from the *operation_parameters* column.

```
<?xml version="1.0"?>
<InputParameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-inst
  <SubscriptionId>1cb17b61-2ce7-45ad-af57-285295e9ed76</SubscriptionId>
  <RestoreId>c12d564c-c2bb-402d-9d36-b3e9b4f36825</RestoreId>
  <SourceLogicalServerName>copjo5ivt5-covetrusi6-d1-server</SourceLogicalServerName>
  <SourceLogicalDatabaseName>copjo5ivt5-covetrusi6-d1-db</SourceLogicalDatabaseName>
  <TargetLogicalServerName>copjo5ivt5-covetrusi6-d1-server</TargetLogicalServerName>
  <TargetLogicalDatabaseName>restore-262128-copjo5ivt5-covetrusi6-d1-db</TargetLogicalDatabaseName>
  <TargetEdition>Standard</TargetEdition>
  <TargetServiceLevelObjectiveName>S2</TargetServiceLevelObjectiveName>
  <TargetLogicalDatabaseResourceTags>{"DT_TAGGING":"copjo5ivt5-covetrusi6-d1"}</TargetLogicalDatabaseResourceT
  <PointInTime>2022-10-17T17:48:00Z</PointInTime>
  <SourceDatabaseDroppedTime xsi:nil="true" />
  <ExtensionData xsi:nil="true" />
  <IsValidatingTargetDbName>true</IsValidatingTargetDbName>
  <SourceLogicalDatabaseId>5db361f7-d7dd-4a56-a53d-7ac03022ef83</SourceLogicalDatabaseId>
  <SubmitResourceHydration>false</SubmitResourceHydration>
  <SkipValidation>false</SkipValidation>
  <ReadScaleEnabled xsi:nil="true" />
  <ReadScaleUnits xsi:nil="true" />
  <IsAbTest>false</IsAbTest>
  <AllowedReplicaIdForAllPageServers xsi:nil="true" />
  <ZoneResilient xsi:nil="true" />
</InputParameters>
```

The output helps retrieve the *operation_request_id* that can be used to look into the **MonManagementException** table in Kusto for verbose and exception messages.

In addition, the *restore_id* can help in the XTS view to get more details about the restore operation breakdown by looking in **MonRestoreRequests**

Note: MonRestoreEvents and MonRestoreRequests only show information when the Restore Operation started and completed. Neither tables show operation details prior to the Restore operation started. That's why we need to check **MonManagementOperation**, **MonManagementException** for more details. So be careful in some customer case scenarios looking into MonRestoreEvents and/or MonRestoreRequests as it's not enough to track where the issue is or where the delay is coming from. The following Kusto query searches and returns information related to the restore_id "c12d564c-c2bb-402d-9d36-b3e9b4f36825" from **MonRestoreRequests**

```
MonRestoreRequests
| where restore_request_id contains "c12d564c-c2bb-402d-9d36-b3e9b4f36825"
| project TIMESTAMP, operation_details
```

TIMESTAMP	operation_details
2022-10-18 11:24:04.7669184	The database has been restored successfully in 00:04:53.8994249.

Also, in **MonRestoreEvents** Kusto table it returns only events when the Restore operation started at 10/18/2022 11:17:33 AM.

```
MonRestoreEvents
| where restore_request_id contains "c12d564c-c2bb-402d-9d36-b3e9b4f36825"
| project TIMESTAMP, event, exception_message
| order by TIMESTAMP desc
| take 10
```



TIMESTAMP	event	exception_message
2022-10-18 11:17:33.0527132	restoring_database_progress	
2022-10-18 11:21:37.6800920	restoring_database_end	
2022-10-18 11:21:36.4925681	sql_connection_query_result	
2022-10-18 11:21:36.4925681	restoring_database_progress_prediction	
2022-10-18 11:21:36.4925681	restoring_database_progress	
2022-10-18 11:21:36.4925681	restoring_database_progress	
2022-10-18 11:21:36.4925681	restoring_database_progress	
2022-10-18 11:21:36.4925681	restoring_database_progress	
2022-10-18 11:21:36.4925681	restoring_database_progress	
2022-10-18 11:21:36.4925681	restoring_database_progress	

We still have not been able to capture the error or exception from the **MonManagement**, **MonRestoreRequests** or **MonRestoreEvents** Kusto tables.

To do this, we need to check in **MonManagementException** Kusto table and search on the *operation_request_id* = "3CD02217-70C7-4795-8652-4DFCFF724895".

```
MonManagementExceptions
| where request_id == '3CD02217-70C7-4795-8652-4DFCFF724895'
| project TIMESTAMP, event, message
```

TIMESTAMP	event	message
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [LogicalDatabaseSecurityPolicyStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [LogicalDatabaseSecurityPolicyStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [LogicalDatabaseSecurityPolicyStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [LogicalDatabaseSecurityPolicyStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.

TIMESTAMP	event	message
2022-10-18 11:22:50.7038080	exception	Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:22:03.8256383	exception	Finite State Machine [SubscriptionFeatureStateMachine] with key [[1cb17b61-2ce7-45ad-af57-285295e9ed76],[Microsoft.Sql], [SubscriptionFeatureSterlingRestoreProgressReporting]] doesn't exist in the backing store.
2022-10-18 11:22:03.8256383	exception	Finite State Machine [SubscriptionFeatureStateMachine] with key [[1cb17b61-2ce7-45ad-af57-285295e9ed76],[Microsoft.Sql], [SubscriptionFeatureSterlingRestoreProgressReporting]] doesn't exist in the backing store.
2022-10-18 11:21:40.3721543	exception	One or more errors occurred.
2022-10-18 11:21:40.3721543	exception	Exception of type 'Microsoft.Xdb.Common.Service.ResolveInstanceTransientException' was encountered. Exception message does not meet compliance 'ObjectName' so it was logged to PiiManagementExceptions with error identifier 'acaf303f-6406-4a1d-9134-7605bdafcf24'.
2022-10-18 11:21:40.3721543	exception	Finite State Machine [FabricPropertyStateMachine] with key [[tr32445.eastus1-a.worker.database.windows.net ], [fabric:/Worker.ISO/eec86b97e724/SQL.UserDb/e4429c35-3ac7-4d2d-831f-c9fb93ef90d1],[IsInCreate]] doesn't exist in the backing store.
2022-10-18 11:21:40.3721543	exception	Finite State Machine [VldbMigrationRequestStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.
2022-10-18 11:21:40.3721543	exception	Finite State Machine [FabricPropertyStateMachine] with key [[tr32445.eastus1-a.worker.database.windows.net ], [fabric:/Worker.ISO/eec86b97e724/SQL.UserDb/e4429c35-3ac7-4d2d-831f-c9fb93ef90d1],[IsInCreate]] doesn't exist in the backing store.
2022-10-18 11:21:40.3721543	exception	Finite State Machine [VldbMigrationRequestStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.

Repeated exception of type (FiniteStateMachineDoesNotExistException). Finite State Machine [VeryLargeDatabaseStateMachine] with key [[copjo5ivt5-covetrusi6-d1-server],[5c51f9ee-7456-45b3-a64b-eec88fb4fd69]] doesn't exist in the backing store.

Mitigation

Recommendation is not to restore to a sub-core SLO. For example, for this issue with the long running restore that took ~14 hours the customer was restoring to a sub-core S2 SLO tier. The advice is restore to a higher SLO tier, minimum S3 or ideally higher still.

How good have you found this content?

