Import or Export Stuck - Sterling

Last updated by | Simon Redman | Feb 6, 2023 at 9:01 AM PST

Contents

Issue

The customer reports that when importing or exporting a database the operation's progress is not progressing, in some cases it stays at 1% for an extended period of time, sometimes many hours or longer.

Investigation/Analysis

Analysing MomManagement with the following

MonManagement
| where request_id =~ "71D4CAC5-CBEC-4685-956E-FC4B94798BC1"
| where operation_type contains "ExportDatabase" or operation_type contains 'Import'
| project TIMESTAMP , operation_type, event, elapsed_time, request_id , logical_database_name

There will be rows showing Pending status

TIMESTAMP	operation_type	event	elapsed_time	requ
2021-02-01 13:01:41.3245580	ExportDatabase	management_operation_pending		71D CBE 956I FC4I
2021-02-01 13:02:41.9355588	ExportDatabase	management_operation_pending		71D CBE 956I FC4I
2021-02-01 13:03:43.0151670	ExportDatabase	management_operation_pending		71D CBE 956I FC4I
2021-02-02 01:04:17.6354536	ExportDatabase	management_operation_pending		71D CBE 956I FC4I
2021-02-02 01:05:17.8401425	ExportDatabase	management_operation_released		71D CBE 956I FC4I
2021-02-02 01:08:25.8449725	ExportDatabase	management_operation_success	12:07:44.4440000	71D CBE 956I FC4I

The key moment in the trace above is the switch from management_operation_pending to management_operation_released this appears to be the point where the Import/Export starts processing and after this point there will be entries in the MonImportExport table.

When the status is at 1% prior to the management_operation_released the MonImportExport table is empty.

To check the queue length for a region you can use the following query

```
let binWindowHours = 2;
let secondsPerHour = 60 * 60;
let baseData = materialize(MonImportExport
| distinct MachineName
// Count the machines which have reported doing work recently. This will _under_ count if the region is lightl
// and some machines have not serviced jobs within the telemetry retention window.
| summarize num4CWorkers = countif(MachineName startswith "DB4C"), num16CWorkers = countif(MachineName startsw
);
let num4CWorkers = toscalar(baseData | project num4CWorkers);
let num16CWorkers = toscalar(baseData | project num16CWorkers);
MonManagement
| where TIMESTAMP > ago(7d)
| where event in ("management_workflow_create_importexport_request_start")
| join kind=inner (MonImportExport // Determine the time the job started based on the first traces from the TR
        | summarize arg_min(originalEventTimestamp, originalEventTimestamp), arg_max(originalEventTimestamp, o
        | extend request_id = toupper(request_id), ProcessingTime = (originalEventTimestamp2 - originalEventTi
        | project request_id, NodeRole, originalEventTimestamp1, ProcessingTime
    ) on request_id
| extend QueueDurationSeconds = (originalEventTimestamp1 - originalEventTimestamp)
| extend RequiredHardware = NodeRole1
| summarize avg(QueueDurationSeconds), sum(ProcessingTime), count() by bin(originalEventTimestamp1, binWindowH
| extend capacity = iif(RequiredHardware == "DB4C", binWindowHours * secondsPerHour * num4CWorkers, binWindowH
| render timechart with (ytitle="Average time (Seconds)", xtitle="Time (UTC)")
```

Interpreting the graph

The graph should have six series per region in the Kusto cluster, one for each hardware type (DB4C and DB16C), and one for total count, total processing time, and one for queue time.

The thing which indicates a capacity problem would be if the queue time never goes down to zero in any given 24-hr interval and is instead growing infinitely. If you see that the queue time is simply high, this does not necessarilly indicate a problem (since there is no SLA on Import/Export requests).

Check private links

If the customer is using the ImportExport service with private links, there are two pitfalls which may cause a job to appear stuck.

1. The customer forgot to approve the private links
2. The customer's resource groups which contain the private link(s) contains one or more resource locks

You can check whether a particular request uses private links by checking for the parameters on the request, like:

```
MonManagementOperations
| where request_id =~ "303427DD-281F-4D60-97E4-CA0A8529D478" // Put customer's request ID here
| where event =~ "management_operation_start"
| where operation_type =~ "ExportDatabase"
| extend PrivateLinkSqlServerResourceId = extract("<PrivateLinkSqlServerResourceId>([^<]*)</.*", 1, operation_
| extend PrivateLinkStorageResourceId = extract("<PrivateLinkStorageResourceId>([^<]*)</.*", 1, operation_para
| project originalEventTimestamp, request_id, PrivateLinkSqlServerResourceId, PrivateLinkStorageResourceId, op
```

If one of PrivateLinkSqlServerResourceId or PrivateLinkStorageResourceId exists, then this request is using private link.

Verify private links approved

If the customer has forgotten to approve the private endpoint connections, the ImportExport operation will not start.

Help the customer follow the public-facing blog post to verify private links are approved: https://techcommunity.microsoft.com/t5/azure-sql-blog/import-export-using-private-link-now-in-preview/ba-p/3256192 ⤤

In the customer's portal, search for "Private Endpoints". You should find the Private Endpoint Center. Click the Pending connections button on the left pane, and you should see any pending private endpoint connections. Verify that any endpoints which contain "ImportExport" in the name are approved.

Verify resource locks

Resource group locks will prevent the private endpoint from being cleaned up, which will prevent an operation from finishing or being cancel-able.

In the customer's portal, navigate to the resource groups (as specified in the PrivateLinkSqlServerResourceId or PrivateLinkStorageResourceId found above). Verify there are no locks in the resource group. Delete locks will prevent the operation from finishing, read-only locks will prevent being able to approve the private links.

Ask the customer to remove any locks from the resource group. If the ImportExport request was stuck finishing or cancelling, it should automatically clean up in around 30-60 seconds.

Mitigation

In most cases the import will start on its own, but how long it will take is uncertain. It is possible that the service has stopped processing requests so an ICM may be required.

The alternative is to cancel the operation and try again.

This uses the Stop-AzSQLDatabaseActivity powershell command, an example of how to use it is [here] (https://techcommunity.microsoft.com/t5/azure-database-support-blog/how-to-cancel-azure-sql-database-import-or-export-operation/ba-p/1935001 ⤤)

Root Cause Classification

Cases resolved by this TSG should be coded to the following root cause:
Root Cause: Azure SQL v3\Import/Export\Import issues\Import failed Root Cause: Azure SQL v3\Import/Export\Import issues\Export failed

How good have you found this content?

😐 ☹️