Why it is too long to complete the sync

Last updated by | Keith Elmore | Apr 5, 2021 at 7:57 AM PDT

Contents

Why it is too long to complete the sync

Issues

We can start by understanding if the ratio between seconds and number of rows changed. Sync can take longer just based on the fact that had more changes to sync:

```
MonDataSync
| where TIMESTAMP >= ago(3d)
| where Detail contains 'cbb5a6be-0b54-47b5-ace9-d8395a95df3f' //Sync group id
| where ComponentAction == 'CommandAndControlService_SyncStatistics'
| extend DownloadChangesApplied = extract(<DownloadChangesApplied>(.*)&lt;/DownloadChangesApplied>',1,Detail)
| extend DownloadChangesFailed = extract(<DownloadChangesFailed>(.*)&lt;/DownloadChangesFailed>',1,Detail)
| extend DownloadChangesTotal = extract(<DownloadChangesTotal>(.*)&lt;/DownloadChangesTotal>',1,Detail)
| extend UploadChangesApplied = extract(<UploadChangesApplied>(.*)&lt;/UploadChangesApplied>',1,Detail)
| extend UploadChangesFailed = extract(<UploadChangesFailed>(.*)&lt;/UploadChangesFailed>',1,Detail)
| extend UploadChangesTotal = extract(<UploadChangesTotal>(.*)&lt;/UploadChangesTotal>',1,Detail)
| extend SyncStartTime = todatetime(extract(<SyncStartTime>(.*)&lt;/SyncStartTime>',1,Detail))
| extend SyncEndTime = todatetime(extract(<SyncEndTime>(.*)&lt;/SyncEndTime>',1,Detail))
| extend SyncDurationInSeconds = datetime_diff(&quot;Second&quot; ,SyncEndTime, SyncStartTime)
| extend Ratio = toint(DownloadChangesTotal) / toint(SyncDurationInSeconds)
| project TIMESTAMP, DownloadChangesTotal, SyncDurationInSeconds, Ratio
```

Customer Case

Issue : **The sync has been initiated on April 6th and it is still in sync (April 10th) and not getting completed.**
We need to understand why it is too long to complete the sync.

We started by having an overall idea about the sync architecture that customer has and grab the sync AppNames using a CMS query:

```
SELECT
sync_groups.sync_group_id ,
sync_groups.sync_group_name ,
sync_groups.state AS sync_group_state,
CASE sync_groups.conflict_resolution_policy WHEN 0 THEN 'Hub Win' WHEN 1 THEN 'Member Win' END AS conflict_res
sync_groups.hub_logical_server_name as HubServer,
logical_databases.logical_database_name as HubDb ,
sync_groups.application_name ,
sync_members.sync_member_id,
sync_members.sync_member_name,
sync_members.state AS member_state,
CASE sync_members.database_type WHEN 0 THEN 'AzureDB' WHEN 1 THEN 'SQLServer' END AS database_type,
CASE sync_members.sync_direction WHEN 0 THEN 'BiDirectional' WHEN 1 THEN 'ToHub' WHEN 2 THEN 'FromHub' END AS
sync_members.application_name,
sync_members.sync_member_full_dns_server_name,
sync_members.sync_member_database_name,
sync_members.sync_agent_id,
sync_members.sql_server_database_id
--,sync_groups.*
--,sync_members.*
--,sync_groups.sync_schema
FROM sync_groups
LEFT OUTER JOIN sync_members ON sync_members.sync_group_id = sync_groups .sync_group_id
LEFT OUTER JOIN logical_databases ON sync_groups.hub_logical_server_name = logical_databases.logical_server_na
INNER JOIN logical_servers on logical_servers.name = logical_databases.logical_server_name
WHERE sync_groups.hub_logical_database_id = logical_databases.logical_database_id
AND customer_subscription_id = 'Input SubscriptionId here'
--AND CAST(sync_groups.sync_schema as varchar(max)) like '%&lt;QuotedName>[[]dbo].[[]LocalProd]&lt;/QuotedName
```

Results:

sync_group_id	sync_group_name	sync_group_state	conflict_resolution_policy	HubServer	Hu
87d6bd0c-7754-475b-b475-8ccf036bf608	FeedToCloud	Ready	Hub Win	ysv9uumzlc	MI

Using the Hub AppName we can get the member states history using Kusto:

```
MonDataSync
| where AppName == "be66db5262bb" //Hub
| where Component == "DataSyncHostApplication"
```

Detail
The states of sync member DatSync (40848523-d072-49ad-9546-34965395df4f) - hubstate: 2, memberstate:4, jobid:7014735d-1ea4-4393-84e1-11303a5001f8, memebrstatelastupdated: 4/6/2018 12:45:50 PM, hubstatelastupdated: 4/6/2018 12:45:49 PM, lastsynctime:

The states of sync member DatSync (40848523-d072-49ad-9546-34965395df4f) - hubstate: 2, memberstate:16, jobid:, memebrstatelastupdated: 4/13/2018 7:34:14 AM, hubstatelastupdated: 4/6/2018 12:45:49 PM, lastsynctime:

The states of sync member DatSync (40848523-d072-49ad-9546-34965395df4f) - hubstate: 2, memberstate:4, jobid:9b32f36a-026a-451f-b6f4-a5332c445523, memebrstatelastupdated: 4/13/2018 7:37:40 AM, hubstatelastupdated: 4/13/2018 7:37:39 AM, lastsynctime:4/13/2018 7:36:25 AM

The member states mean:

4 = SyncInProgress

16 = SyncCancelled

You can get this info from DSS006 - Data Sync - States or from a sync metadata db using `SELECT * FROM [dss].[EnumType]`

From this, we can see that:

- The customer cancelled the sync at today 2018-04-13 07:34
- The customer started sync again at 2018-04-13 07:37

Focusing on the sync issue itself, using the member AppName in Kusto:

```
MonDataSync
| where AppName == "b4ddbc639f1b" //Member
| where Component != "Scheduler"
```

We can see in the Details filed that the sync is taking a long time because it's detecting a lot of conflicts (row by row comparison) (< changesConflictTillNow >13521500</ changesConflictTillNow >):
< KeyPairs >< id >SyncController_ChangeConflictedEvent</ id >< sId >b375b9d2-4596-4a69-9b87-71dc351108bc</ sId >< tracingId >14614beb-0975-4f46-b498-01a3c6d3bc62</ tracingId >< agentId >40848523-d072-49ad-9546-34965395df4f</ agentId >< agentInstanceId >00000000-0000-0000-0000-000000000000</ agentInstanceId >< syncGroupId >87d6bd0c-7754-475b-b475-8ccf036bf608</ syncGroupId >< changesConflictTillNow >13521500</ changesConflictTillNow >< /KeyPairs >

Customers can avoid a very heavy initial sync by having data in only one member, our documentation says:

"When you create a sync group, start with data in only one database. If you have data in multiple databases, SQL Data Sync (Preview) treats each row as a conflict that needs to be resolved. This conflict resolution causes the initial sync to go slowly. If you have data in multiple databases, initial sync might take between several days and several months, depending on the database size.

If the databases are in different datacenters, each row must travel between the different datacenters. This increases the cost of an initial sync."
In case the customer cannot avoid having data in both members we may have an option **if they can make sure the data is consistent (equal) between hub and member.**

This is called NoInitSync and you can see the action plan at If the destination already has data

Getting the hub AppName and checking the member states is one of the best things you can do on most of the DSS cases.

If you have any questions feel free to reach out.

Classification

Root cause Tree - DataSync/Service Issue/Uncoded

How good have you found this content?