*MSync Troubleshooting Guide*

Version 1.0

Table of Contents

[1 Purpose 4](#_Toc484527671)

[2 Assumptions 4](#_Toc484527672)

[3 Where is replication being used? 4](#_Toc484527673)

[4 Which tables does it cover? 4](#_Toc484527674)

[5 What is the replication delay, in minutes? 4](#_Toc484527675)

[6 Error Details 4](#_Toc484527676)

[7 Common Issues 5](#_Toc484527677)

[7.1 Access Permissions 5](#_Toc484527678)

[7.2 Initial Load 5](#_Toc484527679)

[7.3 Schema Changes 5](#_Toc484527680)

[7.4 Blank Columns 5](#_Toc484527681)

[7.4.1 Blank Columns at Publisher 5](#_Toc484527682)

[7.4.2 Blank Columns at Subscriber 6](#_Toc484527683)

[8 Initialization 6](#_Toc484527684)

[9 How do I monitor replication progress? 7](#_Toc484527685)

[10 Pausing/Resuming 9](#_Toc484527686)

[11 Is my data consistent? 9](#_Toc484527687)

[11.1 Check for Consistency using Compare 9](#_Toc484527688)

[11.2 Parameter Definitions 10](#_Toc484527689)

[11.3 Comparison Results 10](#_Toc484527690)

[11.4 Corrective Action 11](#_Toc484527691)

# Purpose

This document is a guideline for addressing common problems with MSync Replication.

# Assumptions

Many of the components referred to in this document reside in the database DBA on Server DBAJUMPBOX. For more details, please refer to the database document.

# Where is replication being used?

For a complete list of all publishers and subscribers, run:

select \* from vwRplSubscription

# Which tables does it cover?

For a complete list of all tables being replicated, please check these views in the DBA database:

* vwRplSubscription
* vwRplPublicationTable
* vwRplSubscriptionTable
* vwRplSubscriptionRoutine
* vwRplCatalogDistribution
* vwRplImportLog
* vwRplImportLogDetail

# What is the replication delay, in minutes?

To determine how many minutes behind replication is running, use this command:

exec CMS..spReplicationCheck @MinutesBehind =0, @ServerName ='%', @DatabaseName ='MSync\_Catalog', @TableName ='%'

# Error Details

To view error details in a subscription database, run this command:

Exec rpl.spR

and check the Message column for errors.

More details can be found by running:

select \* from rpl.ImportLog

select \* from rpl.ImportLogDetail

You can also review SQL Agent job history to find error details.

# Common Issues

Some of the most common issues in MSync Replication debugging are related to:

* Access permissions
* Initial load
* Schema changes
* Blank columns

## Access Permissions

To resolve access permissions problems, make sure you:

1. Add the SQL Agent service account of the subscriber server as a login in the publisher server.
2. Grant the SQL Agent Service account DBO rights on both publisher and subscriber databases.

## Initial Load

Most of the issues happen during the first load, due to the need to clean up the production tables in the subscriber database and reload all records.

To resolve initial load problems, check the following:

1. Make sure rpl.Subscription has Active=1 and Initilize=1
2. Is a SubscribedTable part of an Indexed View? If so, drop the view and recreate it after initial load.
3. Is a SubscribedTable published by Microsoft Replication? If so, drop the subscription.

## Schema Changes

Some issues may occur because MSyncReplication does not transfer Schema changes.

If a column is added or removed at the publisher it must also be added or removed at the subscriber manually.

We must also rerun publish and subscribe commands to rebuild the metadata:

exec rpl.spPublishTable 'SchemaName','TableName'  
exec rpl.spSubscribeTable 1, 'SchemaName', 'TableName'

The DBA database has utitlities to facilitate changing schema of replicated tables, for further details please refer to the DBA Database document.

## Blank Columns

Blank columns may also cause issues. If some columns are blank at the subscriber, but not at the publisher, it’s probably because they were added after the replication was configured.

### Blank Columns at Publisher

To solve the problem of blank columns at the publisher, run this command:

exec rpl.spPublishTable 1,'SCHEMA\_NAME','TBL\_NAME'

### Blank Columns at Subscriber

To solve the problem of blank columns at the subscriber, run these commands:

exec rpl.spSubscribeTable 1,'SCHEMA\_NAME','TBL\_NAME'

update rpl.SubscriptionTable set Initialize=1 where tablename='TBL\_NAME'

# Initialization

Initialization will clean up tables at the subscriber and reload all records from publisher.

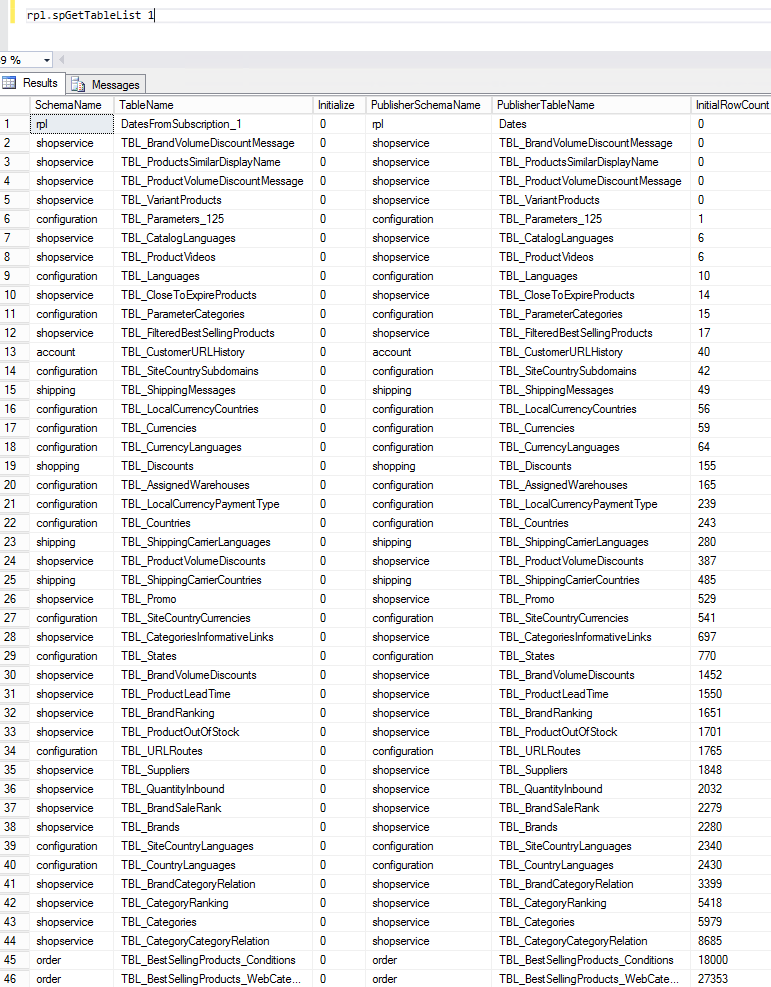
There are four ways to achieve it:

1. For the entire subscription:
   1. Update rpl.Subscription set Initialize=1 where SubscriptionId=?
   2. Or we can delete from rpl.ImportLog where Success=1
2. For one table:
   1. Update rpl.SubscriptionTable set Initialize=1 where TableId =?
   2. Or we can resubscribe the table

Please note that republishing a table will NOT mark it to be reinitialized on the next run. To prevent accidental data wipeouts all initializations must be manually set. For small tables (fewer than 100K rows) it is OK to reload, but for large tables, consider doing manual updates.

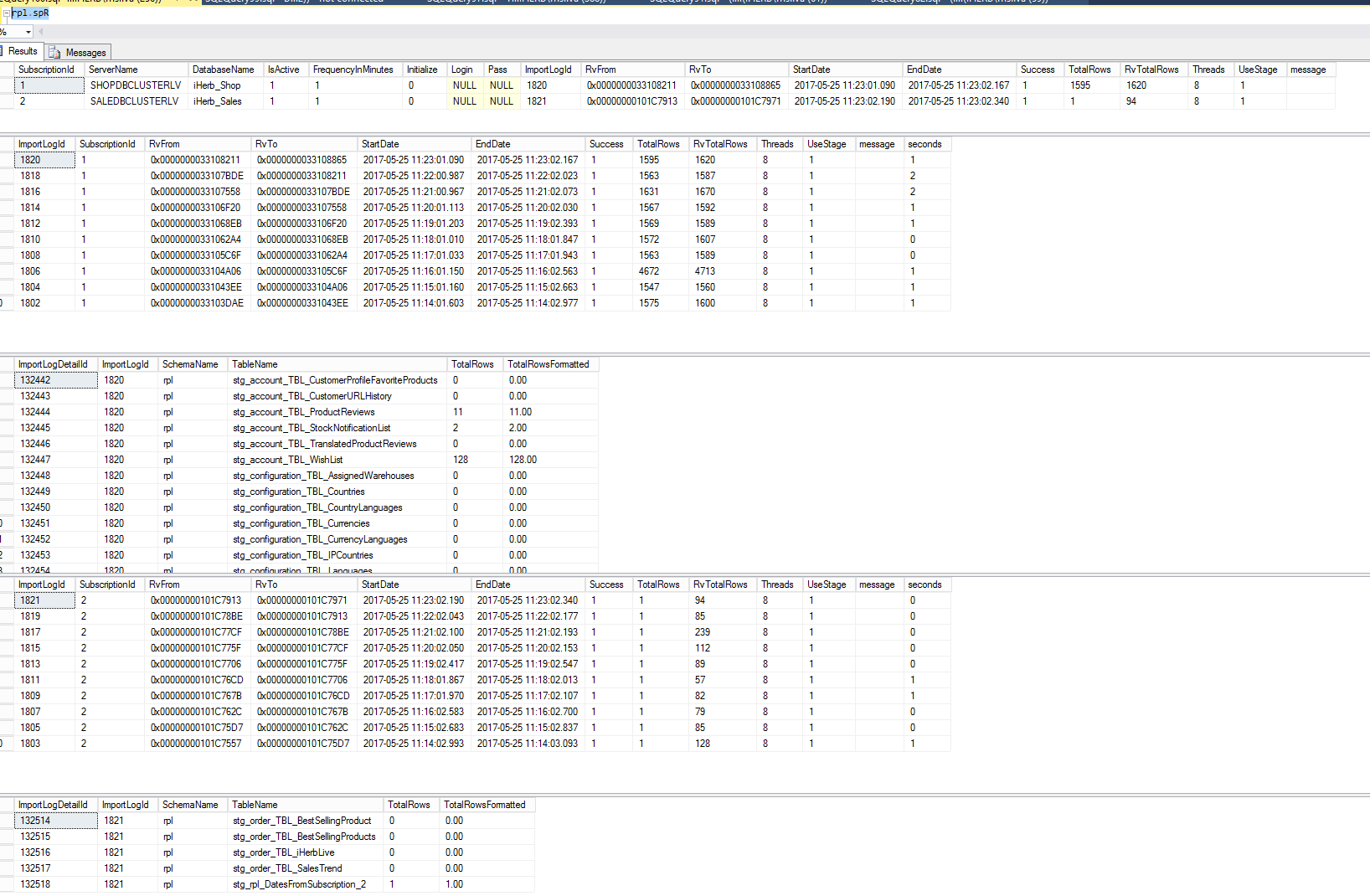
# How do I monitor replication progress?

The proc **rpl.spGetTableList** 1 will tell you which tables will be processed:



Note that they are sorted in ascending order by size, so the smaller tables are done earlier.

During a load the proc **rpl.spR** will indicate the progress of the import:



The first result set shows the active subscriptions, along with the last execution. **Note:** A null EndDate indicates the load is in progress.

For each subscription it will return two more result sets:

1. This history of the 10 last executions.
2. Which tables have been worked by the last (or current) execution, along with the rowcounts. If the load is initial you will see the production table names. If the load is incremental you will see the Stage tables.

The bulk insert usually takes most of the runtime, but these subsequent steps may also take awhile:

The Merge phase, or spl.spMerge for each table, when the load is NOT initial. For an idea you may run:

select \* from sys.sysprocesses p (nolock)

cross apply sys.dm\_exec\_sql\_text(p.sql\_handle) as tx

where text like '%rpl.spMerge%' and spid <> @@SPID

Index rebuilds, in case of an initial load. For an idea you may run:

select \* from sys.indexes where is\_disabled=1

# Pausing/Resuming

Some operations would require us to pause the replication agents such as troubleshooting problematic tables, adding servers, or rearranging tables between subscriptions. There are many ways to disable this process. Here are some of those options.

Example: Catalog

For Catalog, for example, disable the SQL Agent’s job:

MSyncReplicate - Catalog -  Central on server DistDB

This will stop all Catalog replication jobs on all servers.

For more granular control, when you want to disable some subscriptions but not others, you may update the replication tables directly, such as with the following commands:

To disable/enable an entire subscription:

Update rpl.Subscription set IsActive = 0/1 where SubscriptionId=?

To disable/enable one table:

Update rpl.SubscriptionTable set IsActive = 0/1 where TableId=?

# Is my data consistent?

## Check for Consistency Using Compare

To confirm that your data is consistent, follow these steps:

1. Review the checkpoint table on subscriber

select \* from [rpl].[DatesFromSubscription\_1] order by 1 desc

This table gets populated with the contents from rp.Dates in the publisher database, and is a per minute indicator of how far behind the subscription is.

1. Compare all tables:

The following routine will compare every table/row/column between publisher and subscriber.

Before you can use this you must configure a linked server from subscriber to publisher with the same name in the subscription.

exec rpl.spSubscriptionCompare @subscriptionid=1, @debug=1, @exec=0, @MaxRows = 100000, @Days=30

To alleviate resource utilization by comparing large tables, the proc uses the parameters @MaxRows and @Days as in the command in point 2, above. If a table has fewer rows than @MaxRows, then it compares everything. Otherwise, it will check to see whether it has a column named DateLastUpdated or DateCreated. If it does, it filters only the most recent dates, based on those columns. If a table does not have one of those date columns, and has more records that @MaxRows, then it will be skipped altogether.

## Parameter Definitions

|  |  |
| --- | --- |
| Parameter | Description |
| @subscriptionid=1 | Subscription ID |
| @debug=1 | This tells the proc to print each query. |
| @exec=0 | This tells the proc to not execute the query. (It helps to run this in a new window.) |
| @MaxRows = 100000 | This is the threshold in number of rows for tables that do not have DateLastChanged or DateCreated columns. |
| @Days=30 | If a table has more than @MaxRows, and it has a DateLastChanged or DateCreated column, then it limits the date to the most recent number of days. |
| @RunAtPublisher | If your network topology does not allow creating a link from Subscriber to Publisher, like our catalog servers, bug you can create a linked server from Publisher to Subscriber, then you may run the proc with parameters @RunAtPublisher=1, @Exec=0, @Debug=1. Notice you still execute the proc at the Subscriber, because it knows all the tables and columns that it subscribes to, but now the proc will only print the commands, instead of executing them. You should copy the commands and run at the publisher. To prevent double hop authentication issues you should RDP directly into the publisher server. For Catalogs run at DISTDB. |

## Comparison Results

The results of this proc may include messages such as:

* If rows match it will return something like:

“Table purchasing.TBL\_VendorLeadTimes matched 1,224.00 rows!”

* If the table has too many rows and no date column it will return something like:

“Table taxonomy.TBL\_ProductAttributes skipped because it has no DateLastChanged and number of rows exceeds threshold (156,626.00)”

* If a difference is found:
  + The first column has the table name
  + The second column has:
    - Missing in Publisher
    - Missing in Subscriber
    - ColumnName does not match
  + The other columns come in pairs with publisher and subscriber values.

Differences are inevitable. This is because changes are propagated on a schedule. Pay attention to differences that do not correct themselves with new imports.

Compare small tables first, so we can get results back more quickly. If a table has too many rows and no date column, then it will be skipped altogether.

## Corrective Action

To correct the differences, we can issue dummy updates at the publisher database. For instance, “update currency set rate = rate.” This does not change anything, but triggers the replication to pick up those records and repropagate them.