Esgyn Corporation

EsgynDB Multi-DataCenter Replication Guide

Published: April 2018
Edition: EsgynDB Release 2.4.0

Contents

Legal Notice

The information contained herein is subject to change without notice. This documentation is distributed on an “AS IS” basis, without warranties or conditions of any kind, either express or implied. Nothing herein should be construed as constituting an additional warranty. Esgyn Corporation shall not be liable for technical or editorial errors or omissions contained herein.

NOTICE REGARDING OPEN SOURCE SOFTWARE: Project Trafodion is licensed under the Apache License, Version 2.0 (the "License"); you may not use software from Project Trafodion except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/ licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Acknowledgements

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation. Java® is a registered trademark of Oracle and/or its affiliates. Apache®, Hadoop®, HBase®, Hive® and Trafodion® are trademarks of the Apache Software Foundation. Esgyn and EsgynDB are trademarks of Esgyn Corporation.

1. About This Document

This guide explains support in EsgynDB for replication across multiple Data Centers.

2. Intended Audience

This guide is intended for EsgynDB system administrators and users.

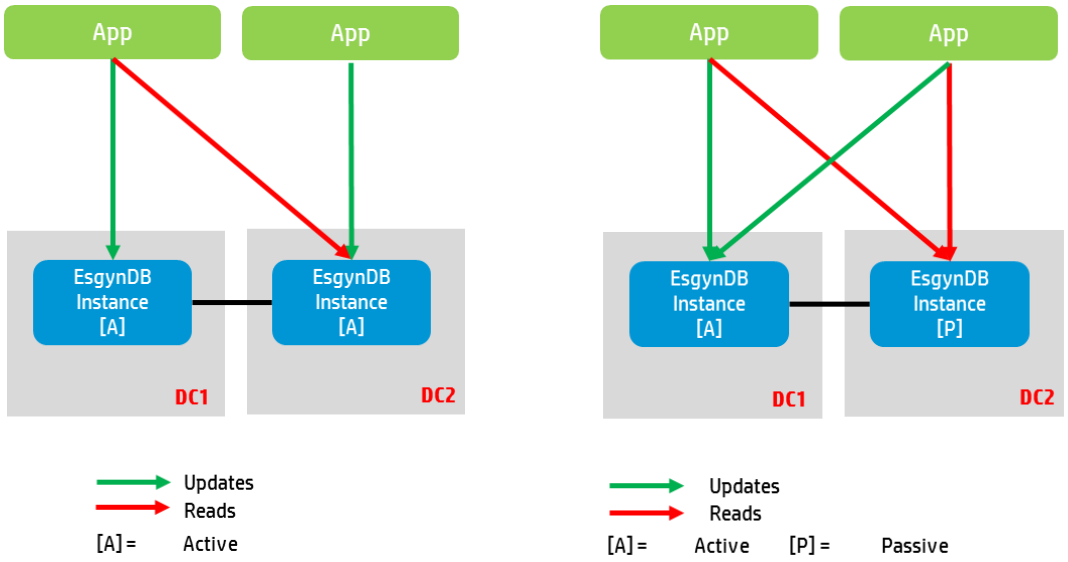
3. Overview

The EsgynDB Multi-Datacenter Replication capability enables data to be synchronously replicated across 2 data centers. HBase keeps multiple copies of data within the same Hadoop instance, allowing for it's availability despite hardware or software component failures within the instance. The EsgynDB Multi-Datacenter replication feature extends that protection across data centers, ensuring data availability in the event of planned instance outages and instance or data center failures. It can also increase write and read capacity.

Data centers may function as

- Active - Active: Applications update both instances concurrently, and read off both
- Active – Passive: Applications update one instance, and read off both

The instances participating in the replication are termed as ***peer instances***.



Replication is done at an EsgynDB table level. In the synchronous replication mode, updates in a table on one instance will automatically be replicated to the peer instance as part of the same transaction. In the no replication mode, table updates will be restricted to the local instance (default behavior).

No application changes are needed to use this functionality.

4. Synchronous Replication

When a table is tagged as synchronous, any transactional updates to that table on one instance will be automatically replicated to the other instance within the same transaction. Single row updates on synchronized tables, even if not explicitly protected within a user transaction, will be automatically made transactional by EsgynDB[1].

If the peer instance is down or unavailable, updates will continue on the single instance.

Peer Instance Failure

In the event of a failure of a peer instance either due to a catastrophic instance failure or datacenter failure (see Figure 1)

- Transactional writes on synchronized tables will hang or fail. Transactional writes on tables that are not synchronized (ie local to the instance) will continue to function normally.
- Read operations on the available instance, regardless of the table synchronization attribute, will succeed.

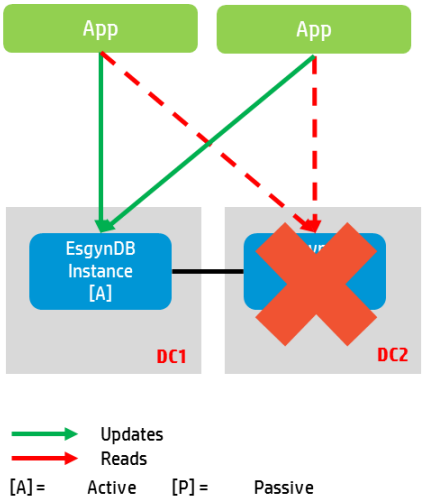


Figure 1: Catastrophic Instance or Datacenter failure

It is also possible that both peer instances are available and functioning normally, but appear to have failed from the other instance's perspective when the connectivity link between them fails (see Figure 2). The behavior will be similar to the case where a peer instance actually failed.

- Transactional writes on synchronized tables will hang or fail. Transactional writes on tables that are not synchronized (ie local to the instance) will continue to function normally ***on both peers***.
- Read operations on ***both peer instances***, regardless of the table synchronization attribute, will succeed.

[1] Single-row updates in tables that are not marked for replication are non-transactional unless they are explicitly made transactional by the user.

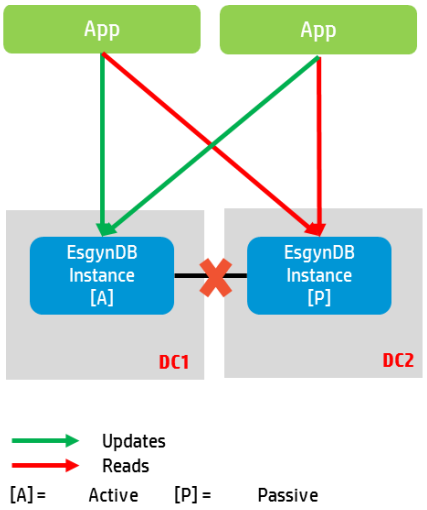


Figure 2: Inter-Datacenter Connectivity Link failure

In either scenario, operator intervention will be required to allow Transactional writes on synchronized tables of the surviving instance. Execute the following steps

1. Use the `xdc` utility to disable synchronous replication on the surviving instance
   a. If the failure is in the connectivity link between the 2 instances, and both instances are other wise operating normally, elect an instance as the surviving instance.
   b. Refer to section on Disable Replication for instructions on disabling synchronous replication.
2. Ensure applications attempting to read from or write to the failed instance are redirected to the surviving instance

5. Setup

Prerequisites

You need the following to enable Multi-Datacenter replication

- Cloudera CDH 5.4
- EsgynDB Enterprise Advanced Edition R2.0 or later
- 2 separate EsgynDB instances (peers) on local or geographically distributed clusters
  - Clusters may be asymmetric (different number of nodes)
  - Connectivity between the two instances
- Tables set up for synchronized replication
  - Table DDLs should be identical between the peers
  - Tables should have the same SALT

Configuration

A multi-datacenter replication configuration database needs to be set up by the system administrator. The database is hosted in Zookeeper. It includes information such as the cluster ID and data about the peer instance (name or IP address, Zookeeper client port, status of Hadoop, Trafodion and replication).

The `xdc` utility is used to query and publish to the configuration database.

Set up configuration database

Do the following on each peer instance as a Trafodion user.

Set up this cluster's cluster ID

- Execute `add_my_cluster_id` from the shell prompt to assign an id to the EsgynDB cluster where you are running on.
  Example: Assign a cluster ID 10 to my instance.
  ```
  $ add_my_cluster_id 10
  ```

- The command `xdc -list` lists the entries in the replication database (one line per cluster id). The format is:

  <cluster id>:<zookeeper quorum>:<zookeeper port>:<status>

  Example:
  ```
  $ xdc -list
   10:esgyn101,esgyn102,esgyn105:2181:tup-sup*
  ```

  Here's a breakdown of the individual elements:

10	The cluster ID
esgyn101,esgyn102,esgyn102	the zookeeper quorum of this cluster
2181	the zookeeper port
tup-sup	Trafodion and synchronization status of the cluster tup – Trafodion is up tdn – Trafodion is down sup – Synchronization is on sdn – Synchronization is off

```
xdc [<command> | <options>...]
<command>          : <   -setmyid <id>
                   :   | -getmyid
                   :   | -set <cluster info>
                   :   | -get <id>
                   :   | -list
                   :   | -delete <id>
                   :   | -push
                   :   | -pull
                   :   | -lock
                   :   | -unlock >
<options>          : [ <peer info> | -h | -v ]
<cluster info>     : < <cluster id>
                             [ <quorum info>
                             | <port info>
                             | <status info> ]
                       ... >
<cluster id>       : -id <id>
<quorum info>      : -quorum <zookeeper quorum>
<port info>        : -port <zookeeper client port>
<status info>      : -status <status>
<status>           : <Trafodion Status>:<STR Status>
<Trafodion Status>: <Trafodion Up>(tup)| <Trafodion Down> (tdn)
<STR Status>       : <STR Up>      (sup)| <STR Down>       (sdn)
<peer info>        : -peer <id>
                   :   With this option the command is executed at
                             the specified peer.
```

```
                    :    (Defaults to the local cluster)
<id>                : A number between 1 and 100 (inclusive)
-h                  : Help (this output).
-v                  : Verbose output.
```

Add peer information

- Execute the add_peer tool to add the peer instance' information. You can execute xdc –list on the peer cluster to obtain the peer's zookeeper quorum and port.

```
add_peer {-i <nn> | -q <an,.>| -p <nn> | -h}

-i <nn>
       ID of the peer (A number between 1 and 100)

-q <an, .>
       Zookeeper quorum - comma-separated list of server names/IP addresses
       Value of the property: hbase.zookeeper.quorum from peer's hbase-
       site.xml
       If this property does not exist in hbase-site.xml then the peer is most
       likely a standalone HBase server.
       In this case, simply provide the server name or the IP address of the
       server where HBase is running.

-p <nn>
       Zookeeper port number.
       Value of the property: hbase.zookeeper.property.clientPort from peer's
       hbase-site.xml
       (Defaults to 2181)

-h Help
```

Push/Pull information to/from the peers.

- Use '`xdc –push`' to push the current cluster's database entry to all its peers.
- Use '`xdc –pull`' to pull the peer cluster's database entry from all the peers.

Set up EsgynDB tables to be replicated

Determine the set of EsgynDB tables that need to be replicated to the peer instance. Use the CREATE TABLE or ALTER TABLE command attributes to set the replication mode. The operation should be repeated on the peer instance.

Note: The system administrator has to ensure that the DDL of synchronous tables on both peer instances is identical.

Indexes will use the replication attribute of the base table automatically. You cannot explicitly create an index with a replication attribute.

Replication is an expensive operation and will affect overall performance of writes to synchronous tables. Diligence should be taken to balance the data availability needs with workload performance goals.

Syntax

```
create table T(…) attributes (synchronous | no) replication;
```

```
alter table T(…) attributes (synchronous | no) replication;
```

Example

- Enable synchronous replication of table T1 between instances DC1 and DC2

DC1 Instance	DC2 Instance
Alter table T1 attributes synchronous replication;	Alter table T1 attributes synchronous replication;

- Disable replication of table T1

DC1 Instance	DC2 Instance
Alter table T1 attributes no replication;	Alter table T1 attributes no replication;

- Enable (one-way) synchronous replication of table T1 from instance DC1 to instance DC2
  Application only writes to instance DC1, reads from both instances

DC1 Instance	DC2 Instance
Alter table T1 attributes synchronous replication;	Alter table T1 attributes no replication;

Enable Replication

Use the `xdc` utility to enable synchronous replication.
Once changes are made in the xdc configuration, restart EsgynDB on the peer instances.

Examples

- Enable synchronous replication from instance DC1 to instance DC2
  Application writes to only instance DC1, reads from both instances DC1 and DC2.
  In the following set of commands, we are disabling synchronization to DC1 and enabling synchronization to DC2.

DC1 Instance	DC2 Instance
xdc –set –status sdn	*xdc –set –status sup*
xdc –push	*xdc –push*

All the commands could be entered from either instance (DC1 or DC2). E.g., the following set is executed on DC1 (let's assume that the cluster id of DC1 is 1 and that of DC2 is 2):

Command on DC1 Instance	Comment
xdc –set –status sdn	Disables synchronization to DC1
xdc –id 2 –set –status sup	Enable synchronization to DC2
xdc –push	Pushes information to the peers
xdc –pull	Pulls information from peers

- Enable synchronous replication between instances DC1 and DC2
  Application writes to both instances DC1 and DC2, reads from both instances DC1 and DC2

DC1 Instance	DC2 Instance
xdc –set –status sup	*xdc –set –status sup*
xdc –push	*xdc –push*

Disable Replication

Use the xdc utility to disable synchronous replication.
Once changes are made in the xdc configuration, restart EsgynDB on the peer instances.

Examples

- Disable synchronous replication from instance DC1 to instance DC2
  Application writes to only instance DC2, reads from both instances DC1 and DC2

DC1 Instance	DC2 Instance
xdc -set –status sup	*xdc -set –status sdn*
xdc –id 2 –set status sdn	*xdc –id 1 –set –status sup*
xdc -push	*xdc -push*

- Disable synchronous replication between instances DC1 and DC2

DC1 Instance	DC2 Instance
xdc –set –status sdn	*xdc –set –status sdn*
xdc –push	*xdc –push*

6. Operational Impacts

The Multi-Datacenter support feature may have an impact on some operational procedures.

Update Statistics

The UPDATE STATISTICS operation is independent of the replication attribute of the table. An operation on a table with the synchronous attribute will only act on the local instance.

Backup Restore Utility

Backups of tables with the synchronous attribute will be no different from backups of regular tables. A restore of the synchronous table will retain the attribute.

EXPLAIN Function

The EXPLAIN function output will display the replication attribute for the table or index.

```
>>explain insert into t values (1,1);

------------------------------------------------------------ PLAN SUMMARY
```

```
MODULE_NAME ............. DYNAMICALLY COMPILED
…

TRAFODION_INSERT ========================  SEQ_NO 2        NO CHILDREN
TABLE_NAME ............. TRAFODION.SEABASE.TI
…
  iud_type ............. index_trafodion_insert TRAFODION.SEABASE.TI
  replication .......... synchronous
  new_rec_expr ......... ("B@" assign TRAFODION.SEABASE.T.B),
                           (A assign TRAFODION.SEABASE.T.A)


TRAFODION_INSERT ========================  SEQ_NO 1        NO CHILDREN
TABLE_NAME ............. TRAFODION.SEABASE.T
…
  iud_type ............. trafodion_insert TRAFODION.SEABASE.T
  replication .......... synchronous
  new_rec_expr ......... (A assign %(1)), (B assign %(1))

--- SQL operation complete.
```

7. Recovery from Failure

Most software failures in a Multi Datacenter configuration are handled transparently. There are scenarios where manual intervention will be needed.

Active – Passive configuration

- Application writes to synchronous tables exclusively on one peer (say DC1)
- Application reads off of synchronous tables on either peer

DC1 (Active)	DC2 (Passive)	Impact & Recovery
EsgynDB DTM process failure	No failure	Impact: Current queries on the node may be affected. New queries can execute normally. Recovery: No intervention needed. EsgynDB will automatically restart the process.
EsgynDB Monitor process failure	No failure	Impact: Queries executing on the node may be affected. Recovery: If the physical node is healthy, use the Node Reintegration procedure to reintegrate the node into the EsgynDB instance.
HBase Region Server failure	No failure	Impact: Queries writing to affected regions on the impacted Region Server may fail. EsgynDB will internally retry many failures before responding back to the application. Recovery: Use the Cloudera Manager or Ambari tools to restart the downed Region Server.

HBase/Hadoop failure	No failure	Impact: Queries on the active peer instance will fail. Recovery: Use the xdc utility to declare DC1 down. This will initiate the failover processing. DC2 should now be elected as the surviving instance. Redirect applications to DC2 to continue further processing.
No failure	EsgynDB DTM process failure	Impact: Synchronized updates will be unaffected. Recovery: No intervention needed. EsgynDB will automatically restart the DTM process.
No failure	EsgynDB Monitor process failure	Impact: Queries executing on the node may be affected. Recovery: If the physical node is healthy, use the Node Reintegration procedure to reintegrate the node into the EsgynDB instance.
No failure	HBase Region Server failure	Impact: Synchronized updates on the affected regions will fail until HBase has migrated the regions to other Region Servers (initiated automatically by HBase). Recovery: Use the Cloudera Manager or Ambari tools to restart the downed Region Server.
No failure	HBase/Hadoop failure	Impact: Synchronized updates will fail. Recovery: Manual intervention needed. Use the xdc utility to declare DC2 down. DC1 will now continue processing as a standalone system.

Active – Active configuration

- Applications write to synchronous tables on both peers
- Applications read off of synchronous tables on either peer

DC1 (Active)	DC2 (Active)	Impact & Recovery
EsgynDB DTM process failure	No failure	Impact: Synchronized updates may be impacted for queries with transactions originating in the same node as the failed DTM process on DC1. Queries originating on DC2 are unaffected. Recovery: No intervention needed. EsgynDB will automatically restart the process.

EsgynDB Monitor process failure	No failure	Impact: Queries originating on the node may be affected. Queries executing on the node, but originated elsewhere, will be fine. Recovery: If the physical node is healthy, use the Node Reintegration procedure to reintegrate the node into the EsgynDB instance.
HBase Region Server failure	No failure	Impact: Queries writing to affected regions will fail and will need to be retried. Recovery: Use the Cloudera Manager or Ambari tools to restart the downed Region Server.
HBase/Hadoop failure	No failure	Impact: Queries on the active peer instance will fail. Recovery: Manual intervention is needed. Use xdc utility to declare DC1 down and elect DC2 as the sole instance. Redirect applications to DC2.

Getting back to Original working state

Once a failed instance has been repaired and before it is ready to be integrated into the replica set, the synchronous tables on the repaired instance have to be synchronized. This is a manual operation that will require the surviving instance to be taken offline. Execute the following steps

1. Stop the workload on the surviving instance, and take it offline gracefully. This will allow all currently executing transactions to complete.
2. Delete data in the existing synchronous tables of the newly repaired instance using the SQL PURGEDATA command or by DROPping and recreating the tables.
   A PURGEDATA operation will delete data from the indexes of the table, if any.
3. Shut down EsgynDB on both the newly repaired and surviving instances.
4. Use the `cleanat` tool to remove existing EsgynDB transaction log files (TLOGs) on the repaired and surviving instances.
5. Restart EsgynDB on both systems.
6. Use the HBase `CopyTable` command to copy synchronous tables from the surviving instance (Source) to the new instance (Destination)

Task	Action
Replicate synchronous table DDLs from the source to the destination instance	Note: Ignore this task if you have already set up the tables **Source instance** *Invoke `showddl` to output and save away a copy of the synchronous table DDL. If the table has indexes, they will be displayed too.* `sqlci> showddl TRAFODION.JAVABENCH.ED_TABLE_20;`

	Destination instance *Copy `showddl` output from source cluster* *Use `sqlci` or `trafci` to create the table(s)*

Collect information needed for CopyTable	**Destination instance**
	Get the Zookeeper client port
	`$ hbase org.apache.hadoop.hbase.util.HBaseConfTool hbase.zookeeper.property.clientPort`
	The default port number is 2181.
	Get the HBase root zookeeper node
	`$ hbase org.apache.hadoop.hbase.util.HBaseConfTool zookeeper.znode.parent`
	The value will typically be `/hbase`
	Get the Zookeeper quorum
	`$ hbcheck`

Replicate data from synchronous table(s) on source to destination instance	**Source instance**
	Switch to hdfs user
	`$ su –c /bin/bash hdfs`
	Start the CopyTable
	`$ hbase org.apache.hadoop.hbase.mapreduce.CopyTable --peer.adr=<zookeeper-quorum>:<zookeeper-client-port>:<hbase-root-znode-path> <synchronous-table-name>`
	Example:
	`$ hbase org.apache.hadoop.hbase.mapreduce.CopyTable --peer.adr=nap009.esgyn.local,nap008.esgyn.local,nap007.esgyn.local:2181:/hbase TRAFODION.JAVABENCH.ED_TABLE_20`
	Verify data has been replicated
	Note: If you have indexes on the table(s), there are two options
	1. Use CopyTable to copy the indexes
	2. Drop and recreate the index on the table on the destination instance

7. Re-enable synchronous replication mode on both instances
8. Move the instances online
9. Restart the workload