Table of Contents

NetYCE Database Re-synchronization procedure 3
Step 1 3
Step 2 3
Step 3 3
Step 4 4
Step 5 4
Step 6a 5
Step 6b 7
Step 7 7
Step 8 8
Step 9 8
ck_dbsync.pl command 8
Skipping in-sync tables 9
Table check, defragment and repair 10
All-in-one options 11

NetYCE Database Re-synchronization procedure

Refer to the two NetYCE database servers as PRI and SEC although this assignment has nothing to do with the actual organization and hierarchy. Think of it as the local and remote NetYCE database server respectively.

Only execute this procedure if the db-sync issues could not be resolved using ‘skip errors’ and a database re-merge is required. I.e, the normal restore master archive on both servers is not possible since there is no clear master

Step 1

This procedure will use cli commands to be issued by an application manager familiar with he NetYCE setup and UNIX commands

ALL tasks have to be executed as ‘yce’ user

Do not permit user activity or API activity on any NetYCE system while executing this procedure

Step 2

Set the ignore flag for the skulker and kill it on both servers

```
$ go stop skulker
```

Check both servers are running on their preferred database.

```
$ go jobs
$ hostname -i
$ cat DSN.dat
```

If ip-addresses does not match, edit DSN.dat and update the address

This step ensures no database failovers will take place during execution

Step 3

Ensure we have sufficient workspace in /var/tmp and /var/opt/mysql The free disk space required is somewhat more than the largest *.MYD file in the databases

Check both servers:

```
$ df -h /var/opt/mysql
$ df -h /var/tmp
```

If disk space is low, see what files could be removed from the archives directory

```
$ go backup
$ ls -l
```

Or empty debugging logs

```
$ go clr dev
$ go logs
$ cp /dev/null *debug*
```

Or excessive database (bin)log files

```
$ go mysql
$ ls -l
```

If very large files like ‘mysql-bin.000nnn’ or ‘<hostname>-relay-bin.000nnn’ are found, those will be removed in step 4, other large log files can safely be deleted.

Should the mysql-bin and relay-bin files not prove to be removable, stop the database, delete ALL these files and restart it.

Step 4

On both servers ensure no table errors exist and the table fragmentation is reduced to within 5%:

```
$ ck_dbsync.pl -x -c
```

> This command MUST be executed locally on both NetYCE database servers since the '-c' command cannot perform the disk space requirement verifications remotely.

Because the '-c' option was added in later revisions of ck_dbsync.pl, use the log_maint.pl script instead if the -c option is not available:

```
$ log_maint.pl
```

Check for errors If in doubt, repeat

As with the ck_dbsync '-c' option , this command must be executed locally on both NetYCE database servers

Step 5

> Steps 5 through 8 use the 'ck_dbsync.pl' script. The command sequences for these steps are also available in all-in-one options ('-R' and '-Q' options). Familiarise yourself with the 'ck_dbsync.pl' command and its options in the paragraphs at the bottom of this page.

Reset the dbsync for both replication directions:

```
$ ck_dbsync.pl -x -F
$ ck_dbsync.pl -x -F -r
```

These commands will first repair the master-master setup between the PRI and SEC databases if it is needed.

Then the master and slave replication backlogs will be reset at both ends. This will also get rid the possibly by now very large binary log files in the var/opt/mysql directory.

Verify no large mysql-bin or relay-bin files remain after this step.

The synchronization may seem to be functional again at this stage, but since data-inconsistencies are not yet removed, the out-of-sync errors will soon reappear.

Step 6a

Steps 6a and 6b perform the actual database table comparison and merging. This could be a very slow process since all column values of all rows in all the tables need to be compared and were needed inserted or updated.

Therefore the merging can be executed in two phases. The 6a) step is faster and focuses on table mismatches where rows are missing. It ensures that the the rows (using the same keys) of a table exist in both databases. It does not verify that all rows have the same content, only the added missing rows will have that guarantee.

The second phase of this step, 6b), is significantly slower and will perform the full table field verification AND insert any missing rows as well. In situations where the time required for a full table verification is not (yet) available, the operator could choose to execute step 6a) and skip 6b). When such time restraints are no factor, 6a) is optional and could be skipped.

> Skip to step 6b) if the fast-but-incomplete key-only compare is not desired

The ck_dbsync.pl has many options. Below is fastest method to merge only those tables with obvious row count and size differences. In this case we demonstrate the manual table-by-table approach where the operator selects the tables with an obvious row count difference. This involves using the `-t table_name` option. Without this option all tables will have their rows checked.

We need to compare each table twice. Once to compare the PRI database with the SEC database, and once to compare the SEC database to the PRI. To accomplish this reversal the -r option is used.

Execute on PRI only (the SEC gives the same results)

```
$ ck_dbsync.pl -x -s
```

Write down the table names listing issues in records or size like the Task_log table in the example below:

```
$ ck_dbsync.pl -x -s
2017-12-19 13:40:13 === Started ===
```

```
2017-12-19 13:40:13 Defaulting to -s (summary option)
2017-12-19 13:40:13 Using local PRI and remote SEC setup:
2017-12-19 13:40:13   PRImary: 172.17.10.28
2017-12-19 13:40:13   SECondary: 172.17.10.22
2017-12-19 13:40:13 Connected to databases
2017-12-19 13:40:13 Table: YCE.Action_log                |   254870
rows|    57.1 Mb|
2017-12-19 13:40:13 Table: YCE.auth_permissions          |      400
rows|     0.0 Mb|
2017-12-19 13:40:13 Table: YCE.City_codes                |     4038
rows|     0.2 Mb|
2017-12-19 13:40:13 Table: YCE.Client                    |      122
rows|     0.0 Mb|
2017-12-19 13:40:13 Table: YCE.Client_types              |      224
rows|     0.0 Mb|
2017-12-19 13:40:13 Table: YCE.Config_log                |     5983
rows|    14.4 Mb|
:::
2017-12-19 13:40:13 Table: YCE.Task_log                  |    14533
rows|   297.0 Mb|
2017-12-19 13:40:13   Rows PRI: 14533
2017-12-19 13:40:13   Rows SEC: 37266
:::
2017-12-19 13:40:13 Table: NMS.Service_status            |       13
rows|     0.0 Mb|
2017-12-19 13:40:13 Table: NCCM.Nccm_node_data           |     1178
rows|     3.8 Mb|
2017-12-19 13:40:13 Table: NCCM.Nccm_parameters          |        5
rows|     0.0 Mb|
2017-12-19 13:40:13 Table: NCCM.Nccm_selection           |       96
rows|     0.0 Mb|
2017-12-19 13:40:13 Replication status
2017-12-19 13:40:13   PRI slave IO : No
2017-12-19 13:40:13   PRI slave SQL: No
2017-12-19 13:40:13   SEC slave IO : No  Got fatal error 1236 from master
when reading data from binary log: 'Client requested master to start
replication from impossible position; the first event 'mysql-bin.000001'
at 153395, the last event read from 'mysql-bin.000001' at 4, the last
byte read from 'mysql-bin.000001' at 4.'
2017-12-19 13:40:13   SEC slave SQL: Yes
2017-12-19 13:40:13 === Completed ===
```

For each of these tables execute these two commands:

```
$ ck_dbsync.pl -x -t <table> -k
$ ck_dbsync.pl -x -t <table> -k -r
```

When done, check the summary again:

```
$ ck_dbsync.pl -x -s
```

Alternatively, if not just the row-count must be fixed, but also the presence of each row must be verified, then execute these two commands once. The number of rows may be equal, but both tables may have (an equal number) of missing rows. The commands below will verify all tables have no missing rows.

```
$ ck_dbsync.pl -x -k
$ ck_dbsync.pl -x -k -r
```

Step 6b

Once the row count of all tables are equal on the PRI and the SEC, a reasonable level of synchronization is achieved. However there could be many differences in the values of each row. To resolve these conflicts all tables need to be compared row-by-row, field-by-field.

This step (6b) is intended to do just that, but requires some time to execute. On a large database hours could be required. Therefore it is suggested to execute this step only if such time is available.

The level of synchronization achieved so far is likely sufficient to prevent synchronization failures due to primary-key violations, but an equal row-count does not imply that both tables have entries for the same keys. To ensure proper synchronization, this step is required at first opportunity. And if they do, rows using the same key could have fields with different values.

The -T option is used to decide which record is the ‘latest’ should it exist on both servers but with field differences. In those cases the -T uses the record timestamps to determine, otherwise the PRI is assumed to be the valid record.

So in cases where the SEC is expected to have most of the valid records, it advisable to use the command with the reversed roles (’-T -r’) options before the regular (‘-T’) command.

Before initiating, ensure that the databases will not be halted for backup. The default backup times are 23:00 and 23:20 and are initiated by crontab (/opt/yce/bin/dbarchive.pl).

```
$ ck_dbsync.pl -x -u -T
$ ck_dbsync.pl -x -u -T -r
```

Whether this step is skipped or executed later, the remaining steps MUST be completed in either case.

Step 7

Reset some important ID's like Job-id and Task-id:

```
$ ck_dbsync.pl -x -L
$ ck_dbsync.pl -x -L -r
```

Step 8

Reset the dbsync, again on PRI only:

```
$ ck_dbsync.pl -x -F
$ ck_dbsync.pl -x -F -r
```

Step 9

Check the NetYCE front-end Admin - System' tool also reports a healthy synchronization status on both servers Test that several times because other tables might need merging too, or the initial updates cause a conflict.

In that case try the 'Skip error' button (only a few times should suffice). If that is not the case, as shown by the backlog counters in the message, then reset the sync again as in step 8.

By this time it should be safe to allow user and API operations to resume.

Also the skulkers can be restarted on each server:

```
go clr skulker
```

ck_dbsync.pl command

The latest revision of ck_dbsync has additional options to simplify, speedup and group actions together. It also has an option to check, defragment and repair corrupt tables.

The usage message:

```
Compare data between synced YCE databases
usage: /opt/yce/system/ck_dbsync.pl options
  options -x -s
          -x -c
          -x -k [-r]      [-t table_name]
          -x -u [-r] [-T] [-t table_name]
          -x -L [-r]
          -x -F [-r]
          -x -R
          -x -Q
   -x        execute: mandatory option. Just to prevent accicental use
   -s        summary: return row count and size of each table
   -c        check tables: defragment and/or repair. Only on local database
   -k        keys only. Insert missing records from PRI into SEC
   -u        update full. Insert and update all SEC records from PRI
   -T        update using timestamps. If PRI timestamp is more recent than
SEC (use with -u)
```

```
   -t table  table-name only. Use 'db.table_name' or 'table_name' format
   -r        reverse: switch PRI (default local) and SEC (remote) databases
   -L        lookup: reset Job, Task and other ID's (after finishing table
updates!)
   -F        force: reset the master and slave configuration for PRI <- SEC
   -R        repair-all: perform a sequence to full-repair using cycles
             of the various options for both databases (-F, -uT, -L, -F).
   -Q        repair-quick: perform a sequence to key-repair using cycles
             of the various options for both databases (-F, -k, -L, -F).
   -h        help: this help message

   The '-x' option is mandatory
```

The new options are '-c' for the table check and repair, the '-R' to perform the sequence for a full re-sync, and the '-Q' to perform the sequence for a quick keys-only re-sync.

The output generated while executing is also logged in the file `/var/opt/yce/logs/ck_dbsync.log`. Each invocation of the command rotates this logfile using extensions '.0' thru '.9'.

Skipping in-sync tables

A significant speedup in the overall process was achieved by determining if tables are already in-sync or not. Tables already in-sync are skipped in the comparison. The sync status is determined by calculating checksums over the data and is thereby not influenced by fragmentation or row-orders.

The output for the summary takes a little longer though. Sample output of the summary (-s) option:

```
$ ck_dbsync.pl -x -s
2018-01-04 10:53:05 === Started ===
2018-01-04 10:53:05 Using local PRI and remote SEC setup:
2018-01-04 10:53:05   PRImary: 172.17.10.28
2018-01-04 10:53:05   SECondary: 172.17.10.22
2018-01-04 10:53:05 Connected to databases
2018-01-04 10:53:05 Table summary
2018-01-04 10:53:06 Table: YCE.Action_log                |   251493 rows|
56.5 Mb|    sync
2018-01-04 10:53:06 Table: YCE.auth_permissions          |      400 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.City_codes                |     4038 rows|
0.2 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Client                    |      122 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Client_types              |      224 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Config_log                |     2580 rows|
9.9 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Domain                    |       32 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Graph                     |      455 rows|
```

```
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Images                      |       21 rows|
0.9 Mb|    sync
2018-01-04 10:53:06 Table: YCE.IpBKrouter                  |        0 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.IpCNrouterIN                |        1 rows|
0.0 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Ipsec_gre                   |     7302 rows|
0.5 Mb|    sync
2018-01-04 10:53:06 Table: YCE.Ipsec_map                   |     1378 rows|
0.1 Mb|    sync
2018-01-04 10:53:06 Table: YCE.IpSupernet                  |     5566 rows|
0.5 Mb|    sync
:::
2018-01-04 10:53:13 Table: YCE.Task_log                    |    33057 rows|
981.3 Mb|out-sync
:::
2018-01-04 10:53:14 Table: NCCM.Nccm_parameters            |        5 rows|
0.0 Mb|    sync
2018-01-04 10:53:14 Table: NCCM.Nccm_selection             |       96 rows|
0.0 Mb|    sync
2018-01-04 10:53:14 Replication status
2018-01-04 10:53:14   PRI slave IO : Yes
2018-01-04 10:53:14   PRI slave SQL: Yes
2018-01-04 10:53:14   SEC slave IO : Yes
2018-01-04 10:53:14   SEC slave SQL: Yes
2018-01-04 10:53:14 === Completed ===
```

Table check, defragment and repair

The '-c' option to check, defragment and repair tables will not support the reverse (-r) option. Due to the verification of free disk space and the detection of crashed repair attempts, the '-c' wil only work on the local server. In order to check both databases, the `ck_dbsync.pl -x -c` command needs to be executed on both NetYCE database servers.

The output of the '-c' option. In this case no repairs or defragmentation was required:

```
$ ck_dbsync.pl -x -c
2018-01-04 11:25:09 === Started ===
2018-01-04 11:25:09 Using local PRI and remote SEC setup:
2018-01-04 11:25:09   PRImary: 172.17.10.28
2018-01-04 11:25:09   SECondary: 172.17.10.22
2018-01-04 11:25:09 Connected to databases
2018-01-04 11:25:09 LOCAL Database table check (defragment, repair)
2018-01-04 11:25:09 Table check
2018-01-04 11:25:09 Flushing database tables to disk
2018-01-04 11:25:09 Checking fragmentation database 'YCE'...
2018-01-04 11:25:09  Table: YCE.IPv6_subnet
```

```
2018-01-04 11:25:09  Table: YCE.IPv6_prefix
2018-01-04 11:25:09  Table: YCE.IPv6_plan_client_type
2018-01-04 11:25:09  Table: YCE.IPv6_map
2018-01-04 11:25:09  Table: YCE.Ip_dhcp
2018-01-04 11:25:09  Table: YCE.Port_scan
2018-01-04 11:25:09  Table: YCE.IPv6_plans
2018-01-04 11:25:09  Table: YCE.IPv6_plan
:::
2018-01-04 11:25:12  Table: NMS.Nmmi_nodes
2018-01-04 11:25:12  Table: NMS.CmdbNodes
2018-01-04 11:25:12  Table: NMS.Dhcp_history
2018-01-04 11:25:12 Checking fragmentation database 'NCCM'...
2018-01-04 11:25:12  Table: NCCM.Nccm_parameters
2018-01-04 11:25:12  Table: NCCM.Nccm_selection
2018-01-04 11:25:12  Table: NCCM.Nccm_node_data
2018-01-04 11:25:12 Checking fragmentation database 'CMDB'...
2018-01-04 11:25:12 Database table checking complete
2018-01-04 11:25:12 === Completed ===
```

When including the '-r' option, the action is denied:

```
$ ck_dbsync.pl -x -c -r
2018-01-04 11:26:46 === Started ===
2018-01-04 11:26:46 Using reversed remote PRI and local SEC setup:
2018-01-04 11:26:46   PRImary: 172.17.10.22
2018-01-04 11:26:46   SECondary: 172.17.10.28
2018-01-04 11:26:46 Connected to databases
2018-01-04 11:26:46 LOCAL Database table check (defragment, repair)
2018-01-04 11:26:46 Table check
2018-01-04 11:26:46   Cannot defragment/repair remote tables
2018-01-04 11:26:46 === Completed ===
```

All-in-one options

The '-R' option combines the majority of steps involved with a full resynchronisation. Using the '-R' option is equivalent to ck_dbsync invocations in the following sequence:

```
-x -s
-x -F
-x -F -r
-x -u -T
-x -u -T -r
-x -L
-x -L -r
-x -F
-x -F -r
-x -s
```

> Please note that the '-c' option is NOT included. Since the table repairs can only be executed

> locally, these steps should be performed BEFORE using the '-R' or '-Q' sequences.

Likewise, the '-Q' option will execute in succession all steps needed to perform key-only comparisons. This is less accurate than the full re-sync, but it is the fastest way to get synchronisation up and running again. The '-R' can then be performed at a later, less busy moment.

Using the '-R' option is equivalent to ck_dbsync invocations in the following sequence:

```
  -x -s
  -x -F
  -x -F -r
  -x -k
  -x -k -r
  -x -L
  -x -L -r
  -x -F
  -x -F -r
  -x -s
```

A sample session summary of the '-R' usage:

```
$ ck_dbsync.pl -x -R
2018-01-04 11:40:21 === Started ===
2018-01-04 11:40:21 Using local PRI and remote SEC setup:
2018-01-04 11:40:21   PRImary: 172.17.10.28
2018-01-04 11:40:21   SECondary: 172.17.10.22
2018-01-04 11:40:21 Connected to databases
2018-01-04 11:40:21 Doing full repair using a sequence of options
2018-01-04 11:40:21 Table status
2018-01-04 11:40:22 Table: YCE.Action_log                |   251493 rows|
56.5 Mb|    sync
2018-01-04 11:40:22 Table: YCE.auth_permissions          |      400 rows|
0.0 Mb|    sync
2018-01-04 11:40:22 Table: YCE.City_codes                |     4038 rows|
0.2 Mb|    sync
2018-01-04 11:40:22 Table: YCE.Client                    |      122 rows|
0.0 Mb|    sync
::
2018-01-04 11:40:29 Table: YCE.Task_log                  |    33066 rows|
981.3 Mb|out-sync
::
2018-01-04 11:40:29 Table: NCCM.Nccm_selection           |       96 rows|
0.0 Mb|    sync
2018-01-04 11:40:29 Reset dbsync PRI <- SEC
2018-01-04 11:40:29   PRI stop slave
2018-01-04 11:40:30   SEC reset master
2018-01-04 11:40:32   PRI reset slave
2018-01-04 11:40:33   PRI start slave
2018-01-04 11:40:34 Reset dbsync SEC <- PRI
```

```
2018-01-04 11:40:34   PRI stop slave
2018-01-04 11:40:35   SEC reset master
2018-01-04 11:40:36   PRI reset slave
2018-01-04 11:40:37   PRI start slave
2018-01-04 11:40:38 Sync table records PRI -> SEC
2018-01-04 11:40:38 Table: YCE.Action_log                 |   251493 rows|
56.5 Mb|    sync
::
2018-01-04 11:40:45 Table: YCE.Task_log                   |    33066 rows|
981.3 Mb|out-sync
2018-01-04 11:40:45   keys: Task_id
2018-01-04 11:40:45   page: 1 / 34
2018-01-04 11:40:46   page: 2 / 34
2018-01-04 11:40:47   page: 3 / 34
2018-01-04 11:40:49   page: 4 / 34
::
2018-01-04 11:42:06   page: 34 / 34
2018-01-04 11:42:06   Record mismatch: Task_id = [0103_0017]
2018-01-04 11:42:06     Task_type [Request: '1' action: '' ] != [Request:
'1' action: 'done' ]
2018-01-04 11:42:06     Task_request [<task response=""> ] != [<task
response="completed"> ]
2018-01-04 11:42:06   Skipping update, PRI timestamp is older than SEC
2018-01-04 11:42:06 Table: YCE.Template                   |     1134 rows|
0.1 Mb|    sync
::
2018-01-04 11:42:07 Table: NCCM.Nccm_selection            |       96 rows|
0.0 Mb|    sync
2018-01-04 11:42:07 Sync table records SEC -> PRI
2018-01-04 11:42:08 Table: YCE.Action_log                 |   251493 rows|
56.5 Mb|    sync
::
2018-01-04 11:42:14 Table: YCE.Task_log                   |    33066 rows|
981.3 Mb|out-sync
2018-01-04 11:42:14   keys: Task_id
2018-01-04 11:42:14   page: 1 / 34
2018-01-04 11:42:16   page: 2 / 34
2018-01-04 11:42:17   page: 3 / 34
2018-01-04 11:42:18   page: 4 / 34
::
2018-01-04 11:43:31   page: 34 / 34
2018-01-04 11:43:31   Record mismatch: Task_id = [0103_0017]
2018-01-04 11:43:31     Task_type [Request: '1' action: 'done' ] !=
[Request: '1' action: '' ]
2018-01-04 11:43:31     Task_request [<task response="completed"> ] !=
[<task response=""> ]
2018-01-04 11:43:31     Updating record in SEC
::
2018-01-04 11:43:32 Table: NCCM.Nccm_selection            |       96 rows|
0.0 Mb|    sync
2018-01-04 11:43:32 Reset PRI Job-, Task- and other ID's
```

```
2018-01-04 11:43:32   Task ID: 40
2018-01-04 11:43:32   Job ID: 1
2018-01-04 11:43:32   Service key: 28893
2018-01-04 11:43:32   Topo ID: 34251
2018-01-04 11:43:32   Crypt update: mirage
2018-01-04 11:43:32   Action_log pos: mirage, yceseven
2018-01-04 11:43:32   Config_log pos: mirage, yceseven
2018-01-04 11:43:32   Task_log pos: mirage, yceseven
2018-01-04 11:43:32 Reset SEC Job-, Task- and other ID's
2018-01-04 11:43:32   Task ID: 40
2018-01-04 11:43:32   Job ID: 1
2018-01-04 11:43:32   Service key: 28893
2018-01-04 11:43:32   Topo ID: 34251
2018-01-04 11:43:32   Crypt update: mirage
2018-01-04 11:43:32   Action_log pos: mirage, yceseven
2018-01-04 11:43:32   Config_log pos: mirage, yceseven
2018-01-04 11:43:32   Task_log pos: mirage, yceseven
2018-01-04 11:43:32 Reset dbsync PRI <- SEC
2018-01-04 11:43:32   PRI stop slave
2018-01-04 11:43:33   SEC reset master
2018-01-04 11:43:34   PRI reset slave
2018-01-04 11:43:35   PRI start slave
2018-01-04 11:43:36 Reset dbsync SEC <- PRI
2018-01-04 11:43:36   PRI stop slave
2018-01-04 11:43:37   SEC reset master
2018-01-04 11:43:38   PRI reset slave
2018-01-04 11:43:39   PRI start slave
2018-01-04 11:43:40 Table status
2018-01-04 11:43:41 Table: YCE.Action_log                |   251493 rows|
56.5 Mb|    sync
::
2018-01-04 11:43:47 Table: YCE.Task_log                  |    33066 rows|
981.3 Mb|    sync
::
2018-01-04 11:43:48 Table: NCCM.Nccm_selection           |       96 rows|
0.0 Mb|    sync
2018-01-04 11:43:48 Sync status
2018-01-04 11:43:48 Replication status
2018-01-04 11:43:48   PRI slave IO : Yes
2018-01-04 11:43:48   PRI slave SQL: Yes
2018-01-04 11:43:48   SEC slave IO : Yes
2018-01-04 11:43:48   SEC slave SQL: Yes
2018-01-04 11:43:48 === Completed ===
```