Sybase Replication Server

Guide pratique – Astuces (version 1.0)

Légende:

```
RS>     Connexion dans Replication Server
RSSD>   Connexion dans la base RSSD
PRI>    Connexion dans la source
SEC>    Connexion dans une cible
UX>     Console Unix
```

1. Mise en veille (mode quiesced)

```
RS> suspend log transfer
        from {data_server.database | all}

RS> admin quiesce_force_rsi

RS> admin quiesce_check

Lorsque la mise en veille est réussie, le message
Replication <RS> is quiesced est affiché.
```

Pour supprimer le mode veille :

```
RS> resume log transfer
        from {data_server.database | all}
```

2. Suppression massive des exceptions dans une base RSSD Adaptive Server Enterprise

Pour cette opération, le moteur de réplication est mis en veille (mode quiesced), voir §1.

Troncature des tables rs_excepts% dans la base RSSD :

```
RSSD> truncate table rs_exceptshdr

RSSD> truncate table rs_exceptslast

RSSD> truncate table rs_exceptscmd
```

Purge de la table rs_systext :

```
RSSD> select * into #rs_systext
      from rs_systext
      where texttype != 'C'

RSSD> truncate table rs_systext

RSSD> insert into rs_systext
      select * from #rs_systext
```

Ne pas oublier de supprimer le mode veille à l'issue de la purge des tables

3. Lister et extraire des commandes SQL dans une exception

Pour lister les exceptions dans la base RSSD :

```
RSSD> rs_helpexception
```

Pour lister le détail d'une exception XactID :

```
RSSD> rs_helpexception XActID
```

Pour extraire les commandes SQL dans une exception XactID:

- Créer la procédure stockée sp_dba_dumpexception retournant les colonnes XActID, sequence et textval pour une exception XActID donnée en paramètre.
- Créer la table proxy v_dba_dumpexception ayant pour source de données la procédure sp_dba_dumexception.

```
RSSD>
create procedure sp_dba_dumpexception @xactid
int=NULL
as
select
        @xactid,
        sequence,
        textval
from rs_systext,
     rs_exceptscmd
where cmd_id = parentid
and texttype='C'
and cmd_type='L'
and (case (select convert(int, 0x0000100))
  when 65536 then convert(int,
reverse(substring(sys_trans_id, 5, 8)))
  else convert(int, substring(sys_trans_id, 5, 8))
  end
 ) = @xactid
order by src_cmd_line,sequence
go

create existing table v_dba_dumpexception (
    xactid int not null,
    sequence int not null,
    textval varchar(255) not null)
external procedure at
'loopback.<RSSD>.dbo.sp_dba_dumpexception'
go
```

- Créer une vue v_xactid interrogeant la vue v_dba_dumpexception pour l'exception XActID à extraire.

```
RSSD> create view v_xactid as
select * from v_dba_exception
where xactid=<XActID>
```

- Extraire les données de la vue v_xactid dans un fichier plat avec le binaire bcp en spécifiant un séparateur TSEP non présent dans les commandes SQL de l'exception.

```
UX> bcp <RSSD>..v_xactid out v_xactid_tmp.sql –Usa
–P<password> -S<RSSD_SERVERNAME> -t'TSEP' -c
```

- Traiter le fichier obtenu avec le programme awk ci-dessous afin de concaténer les commandes SQL réparties sur plusieurs séquences.

```
UX> cat v_xactid_tmp.sql | awk -F"TSEP" 'BEGIN
{ vTextVal="" } \
{ if ($2==1) { print vTextVal; vTextVal=$3; } else
{ vTextVal = vTextVal$3 } } \
END {  print vTextVal; }' > v_xactid.sql
```

4. Déplacer des partitions sans modifier les noms logiques

Pour cette opération, le moteur de réplication est d'abord mis en veille (mode quiesced), voir §1, puis éteint avec la commande shutdown.

```
RS> shutdown
```

Les partitions sont déplacées physiquement.

Les nouveaux chemins des partitions sont mis à jour dans la base RSSD en mettant à jour directement la table rs_diskpartitions :

```
RSSD> update rs_diskpartitions set name='<nouveau
chemin>' where logical_name='<nom logique de la
partition>'
```

Le serveur de réplication est redémarré puis le mode veille est retiré.

5. Assigner une classe d'erreurs personnalisée à une connexion (DirectConnect for MSSQL etc…)

Création de la classe d'erreurs custom_errorclass:

```
RS> create error class <custom_erroclass>
```

La nouvelle classe d'erreurs custom_errorclass hérite d'une classe d'erreurs parente pour prédéfinir les actions sur les erreurs :

```
RSSD> rs_init_erroractions <custom_erroclass>,
<parent_errorclass>

RSSD> rs_init_erroractions dcmssql_error_class,
rs_sqlserver_error_class

La classe dcmssql_error_class hérite de la classe
d'erreurs système rs_sqlserver_error_class.
```

Assignation d'une action spécifique pour un numéro d'erreur rencontrée sur une cible :

```
RS> assign action { ignore | warn | retry_log |
log | retry_stop | stop_replication }
for error_class
to data_server_error [, data_server_error ]

RS> assign action retry_stop for
dcmssql_error_class to 30291
```

Application de la classe d'erreurs personnalisée custom_errorclass à une cible :

```
RS> alter connection to <dataserver>.<database>
set error class to <custom_errorclass>
```

Après application d'une classe d'erreurs personnalisée à une cible, la connexion vers la cible est suspendue et reprise pour la prise en compte :

```
RS> suspend connection to <dataserver>.<database>
RS> resume connection to <dataserver>.<database>
```

6. Surcharge de la fonction rs_usedb avec une chaîne fonction (function string)

Création de la classe de fonctions personnalisée custom_function_class en spécifiant la classe de fonctions parente pour héritage :

```
RS> create function string class
<custom_function_class> set parent to
<parent_function_class>

RS> create function string class
rtds_function_class set parent to
rs_default_function_class
```

Surcharge de la fonction rs_usedb :

```
RS> create function string rs_usedb for
<custom_function_class>
      with overwrite output language
      'use ?rs_destination_db!sys_raw?;

         Commandes T-SQL de surcharge

      '


RS> create function string rs_usedb for
rtds_function_class
      with overwrite output language
      'use ?rs_destination_db!sys_raw?;
       set transactional messaging FULL;
      '
```

Application de la nouvelle classe de fonctions custom_function_class à la cible :

```
RS> alter connection to <dataserver>.<database>
set function string class <custom_function_class>
```

Après application d'une classe de fonctions personnalisée à une cible, la connexion vers la cible est suspendue et reprise pour la prise en compte :

```
RS> suspend connection to <dataserver>.<database>
RS> resume connection to <dataserver>.<database>
```