

Une stratégie de sauvegarde automatisée avec Sybase ASE



par fadace ([Fabien Celaia](#))

Date de publication : 28.4.2008

Dernière mise à jour :

Sauvegarde et restauration à une heure précise d'une base Sybase Adaptive Server Enterprise... En théorie fort simple, ceci peut s'avérer complexe à mettre à jour. Voici un petit outil permettant de le faire.

I - Introduction.....	3
II - Mise en place.....	4
III - Utilisation.....	13

I - Introduction

La plupart des bases de données gèrent leurs transactions de manière à ce qu'il soit toujours possible de remonter une base de données dans un état consistant. Il s'agit parfois de remonter la base en l'état de la dernière sauvegarde pleine, mais il est parfois nécessaire de la remonter à une heure précise (PTR = Point in time recovery).

Le fer de lance de cette stratégie de sauvegarde adéquate est l'accès et l'utilisation de des bons fichiers, et surtout la sécurisation du système permettant de maintenir un niveau de transactions sain.

DB2-UDB et Oracle offrent la possibilité de travailler soit en log tournants (le journaux de transactions s'écrasent au fur et à mesure), soit en log non tournant (les journaux de transactions doivent être extraits / copiés sur bande afin d'être disponible qen cas de restauration).

Sybase ASE et Microsoft SQL Server gèrent un journal de transaction interne à la base (la table syslogs) qui peut se purger automatiquement ou s'extraire.

Voici un bref résumé de ces différentes méthodes :

SGBDR	Type de stratégie	Description	Récupération Point in time
Oracle	noarchivelog	Les redo logs s'écrasent au fur et à mesure de leur réutilisation. Pas de possibilité de sauvegarder la base à chaud.	Non
Oracle	archivelog	Avant de se faire écraser, les redo logs sont copiés par l'archiver dans un répertoire distinct. Ces fichiers sont ensuite sauvegardés via rman	Oui
DB2-UDB	Log cyclique	Les logs s'écrasent au fur et à mesure de leur réutilisation. Pas de possibilité de sauvegarder la base à chaud.	Non
DB2-UDB	Log non cyclique	Les logs se stockent de manière séquentielle sur le disque. Seul le nettoyage manuel ou semi-automatique permet de libérer cet espace.	Oui
Microsoft SQL Server	Full	Toutes les transactions sont stockées dans la table syslogs (le fichier .ldb). Rien n'est effacé, hormis lors du passage d'une commande backup transaction. A ce moment, les transactions mortes (déjà commitées ou déjà rollbackées) sont purgées de la table syslogs	Oui
Microsoft SQL Server	Simple	Les transactions ne sont pas sauvegardées. A chaque checkpoint, toute transaction commitée ou rollbackéed isparait du journal de transactions.	Non
Sybase ASE	trunc log on checkpoint=on	Toutes les transactions sont stockées dans la table syslogs (les segments dont le segmap est à 4). Rien n'est effacé, hormis lors du passage d'une commande dump transaction. A ce moment, les transactions mortes (déjà commitées ou déjà rollbackées, et en cas de réplication, seulement si le 2e point de troncature (Log Truncation Point) les a déjà traitées) sont purgées de la table syslogs	Non
Sybase ASE	trunc log on checkpoint=off	Les transactions ne sont pas sauvegardées. A chaque checkpoint, toute transaction commitée ou rollbackéed isparait du journal de transactions.	Oui

II - Mise en place

L'idée de cette procédure est de tracker les diverses sauvegardes faites afin que le système soit en mesure le plus aisément possible de remonter les sauvegardes utiles. Il suffirait alors de donner la date et l'heure exacte (Point in Time Recovery) de la restauration pour que le système, grâce à une table d'historisation, remonte les différentes sauvegardes nécessaires dans l'ordre adéquat.

On crée donc une table **TrackDump** qui servira à cette historisation.

```
-- Author : Fabien Celaia
-- Date   : 27.6.1999
-- Desc.  : table trackDump, to manage the dumps made by sp__dump

if exists(select 1 from sysobjects where name = "trackDump" and type = "U")
begin
print "Dropping old trackDump table"
drop table trackDump
end
go

print "Creating new trackDump table"
create table trackDump
(
dbid int not null,
file varchar(255) not null,
type char(1) not null,
error int,
start datetime not null,
stop datetime not null
)
go
print "Creating primary key to trackDump"
create unique clustered index trackDump_unc on trackDump( start, dbid)
go
print "Granting table trackDump"
grant delete on trackDump to oper_role
go
```

Création de la procédure stockée qui permettra d'exécuter des sauvegardes et d'en garder un historique. Il sera dès lors important que toute sauvegarde passe par cette transaction afin de ne pas couper la chaîne des journaux.

```
if exists(select * from sysobjects where name = "sp__dump" and type = "P")
begin
print "Dropping old sp__dump stored procedure"
drop proc sp__dump
end
go

print "Creating new sp__dump stored procedure"
go
create proc sp__dump (@dbname varchar(30), @method char(1)=NULL, @pathtodump varchar(200)=NULL)
as
-- Author : Fabien Celaia
-- Date   : 27.6.1999
-- Syntax : sp__dump @dbname, {@method}, {@pathtodump}
-- Inputs : @dbname (mandatory) : name of the database to save
--          @method (optional)  : NULL => dump database
--          "T"                  : "T" => dump transaction log
--          @pathtodump (optional) : Null => c:\temp (Windows) or /tmp (Unix)
--          : else saved in the given directory
-- Outputs: 0 => succeed
--          -1 => DB doesn't exist
--          -2 => DB needs to be saved before it's log
--          -3 => BS is down
--          -4 => Corrupted database : try to save the last log
```

```

begin
declare @fichier varchar(255), @debut datetime, @fin datetime,
        declare @err int, @delcmd char(3), @numstripe int, @cpt int

set nocount on

if exists (select @@version where @@version like "%NT%")
/* Unix */
begin
if @pathtodump is null
select @pathtodump = "/tmp"
select @delcmd = "rm"
end
else
/* Dos */
begin
if @pathtodump is null
select @pathtodump = "C:\temp\"
select @delcmd = "del"
end

if not exists(select name from master..sysdatabases where name = @dbname)
begin
print "The database %1! does not exist", @dbname
return -1
end
else
begin
-- Creation du fichier
select @debut = getdate()
select @fichier= @pathtodump+@dbname + "_" + substring(convert(char, @debut, 112), 1,8) +

 "_" + substring(convert(char,@debut,8),1,2) + substring(convert(char,@debut,8),4,2)+ substring(convert(char,@debut,8),7,2)

-- Spécifie des stripe devices par bloc de 2Gb
select @numStripe = sum(size/512)/2048, @cpt=1
from master..sysusages
where dbid=db_id(@db_name)

while @cpt < @numStripe
begin
select @fichier = @fichier+ " stripe on " +@fichier+ convert(char(3),@cpt)
select @cpt=@cpt+1
end

-- test si base corrompue : si tel est le cas, sauvegarde du dernier log
if exists (select *
from master..sysdatabases
where name = @dbname
and status2 = status2|256)
begin
print "The database %1!is corrupt : trying to save the last transaction log.", @dbname
dump tran @dbname to @fichier with no_truncate
return (-4)
end

if @method is not null
-- si sauvegarde du journal de transaction
begin
-- test si base déjà sauvegardée
if not exists(select * from DBA..trackDump where dbid = db_id(@dbname) and type ="D")
begin
print "Save the full database before the transaction log"
return -2
end
else
begin
select @fichier = @fichier + ".tran"
dump tran @dbname to @fichier
select @err = @@error

```

```

if @err = 7205
    print "Dump failed : BS probably is down."
if @err = 0
    print "Backup succeed"
    insert into DBA..trackDump (dbid, start, stop, file, type, error)
    values (db_id(@dbname), @debut, getdate(), @fichier, "T", @err)
end
end
else
-- sinon sauvegarde de la base de donnees
begin
select @fichier = @fichier+".dmp"

-- Si la base est configuree pour utiliser dbcc checkstorage, l'executer !
dbcc checkstorage(@dbname)
select @err=@@error
if @err <> 0
    begin
    print "DBCC checkstorage failed for DB %1", @dbname
    insert into DBA..trackDump (dbid, start, stop, file, type, error)
    values (db_id(@dbname), @debut, getdate(), NULL, "C", @err)
    end

dump database @dbname to @fichier
select @err = @@error
if @err = 7205
    begin
    print "Dump failed : BS probably is down."
    return -3
    end
if @err = 0
    print "Backup succeed"
    insert into DBA..trackDump (dbid, start, stop, file, type, error)
    values (db_id(@dbname), @debut, getdate(), @fichier, "D", @@error)
end

end
return @err
end
go
    
```

Procédure permettant de contrôler l'état des sauvegardes d'une base donnée

```

if exists(select * from sysobjects where name = "sp_dump_fault" and type ="P")
begin
print "Dropping old sp_dump_fault stored procedure"
drop proc sp_dump_fault
end
go

print "Creating new sp_dump_fault stored procedure"
go
create proc sp_dump_fault @dbname varchar(30) = Null
as
-- Author : Fabien Celaia
-- Date   : 27.6.1999
-- Syntax : sp_dump_fault @dbname
-- Inputs : @dbname (optional) : name of the database to check
-- Outputs: 0 => succeed

begin
set nocount on
if @dbname is Null
select db_name(t.dbid) DBName, t.start,t.error , m.description
from DBA..trackDump t, master..sysmessages m
where t.error <> 0
and t.error = m.error
group by dbid
order by start
    
```

```

else
select db_name(t.dbid) DBName, t.start,t.error, m.description
from DBA..trackDump t, master..sysmessages m
where t.error <> 0
and t.error = m.error
and t.dbid = db_id(@dbname)
order by start
end
go

```

Procédure permettant de nettoyer un historique de sauvegardes. Il va sans dire qu'il serait judicieux de garder tout l'historique

```

if exists(select * from sysobjects where name = "sp_dump_clean" and type ="P")
begin
print "Dropping old sp_dump_clean stored procedure"
drop proc sp_dump_clean
end
go

print "Creating new sp_dump_clean stored procedure"
go
create proc sp_dump_clean (@dbname varchar(30))
as
-- Date : 30.6.1999
-- Syntax : sp_dump_clean @dbname, @time
-- Inputs : @dbname (mandatory) : name of the database to clean
-- Outputs: 0 => succeed
--          -1 => DB doesn't exist

begin
declare @limite datetime, @fich varchar(80), @xpval int, @delcmd varchar(31)

set nocount on

if not exists(select name from master..sysdatabases where name = @dbname)
begin
print "The database %! does not exist", @dbname
return -1
end

-- test si la configuration permet d'envoyer une commande en ligne
select @xpval = cur.value
from master..sysconfigures cfg, master..syscurconfigs cur
where cfg.config= cur.config
and cfg.name like "%xp_cmdshell%"

exec sp_configure "xp_cmdshell", 0

select @limite=max(start)
from DBA..trackDump
where dbid = db_id(@dbname)
and type="D"
and error=0

declare track_cur cursor
for select file
from DBA..trackDump
where dbid=db_id(@dbname)
and start < @limite
for update

open track_cur

fetch track_cur into @fich

while @@sqlstatus != 2
begin
select @fich = @delcmd + " " + @fich
exec xp_cmdshell @fich

```

```

    fetch track_cur into @fich
    end

    close track_cur
    deallocate cursor track_cur

    select file "Deleted Files" from DBA..trackDump
    where dbid=db_id(@dbname)
    and start < @limite

    delete DBA..trackDump
    where dbid=db_id(@dbname)
    and start < @limite

    exec sp_configure "xp_cmdshell", @xpval
    end
go
    
```

Procédure permettant d'afficher l'état des sauvegardes d'une base spécifique

```

if exists(select 1 from sysobjects where name = "sp__dump_help" and type ="P")
begin
    print "Dropping old sp__dump_help stored procedure"
    drop proc sp__dump_help
end
go

print "Creating new sp__dump_help stored procedure"
go
create proc sp__dump_help (@dbname varchar(30)=null)
as
-- Date      : 8.8.2000
-- Syntax : sp__dump_help : display the list of the interessting dumps and the parameters
-- Inputs : -
-- Outputs:
begin
    declare @def varchar(255), @del varchar(30)

    if exists(select @@version where @@version like "%NT%")
        /* NT */
        select @del="del", @def="C:\temp"
    else
        /* Unix */
        select @del="rm", @def="/tmp"

    print ""
    print "Default Dump device : %! ", @def
    print "OS Command deleting a file : %! ", @del
    print ""
    print ""
    print "Available dumps"
    print "======"
    print ""
    select db_name(dbid), start Date, case
        type when "T" then "Transaction" when "C" then "Checkstorage" else "Database" end "Type", file from
    trackDump
    where error = 0
    order by 1,2
end
go
grant execute on sp__dump_help to public
go
    
```

Procédure permettant de restaurer une base à une date et une heure précise

```

if exists(select 1 from sysobjects where name = "sp__dump_recover" and type ="P")
begin
    print "Dropping old sp__dump_recover stored procedure"
    
```

```

drop proc sp__dump_recover
end
go

print "Creating new sp__dump_recover stored procedure"
go

create proc sp__dump_recover (@dbname varchar(30), @time datetime=NULL)
as
-- Author : Fabien Celaia
-- Date : 27.6.1999
-- Syntax : sp__dump_recover @dbname, @time
-- Inputs : @dbname (mandatory) : name of the database to reload
--          @time (optional) : recover till the given time (default = current datetime)
-- Outputs: 0 => succeed
--          -1 => Corrupted database
--          -2 => No valide database dumps exist
--          -3 => BS is down
--          -4 => Database in use

begin
declare @file varchar(255), @err int

set nocount on

if db_name() = @dbname
begin
print "You must exit of the database %1! before loading it. Please, use master."
return (-4)
end

if @time is null
select @time = getdate()

declare dump_cur cursor for
select file
from DBA..trackDump
where stop < @time
and error = 0
and dbid = db_id(@dbname)
and start > ( select max(start)
from DBA..trackDump
where error = 0
and dbid = db_id(@dbname)
and type ="D")
order by start

select @err= count(uid)
from master..sysprocesses
where dbid = db_id(@dbname)
and suid <>0

if @err > 0
begin
select @file= convert(varchar(8), @err)
print "%1! users are currently using %2!. Kill them before executing the load !",@file, @dbname
select u.uid UserID, l.name Login, u.name UserName
from master..sysprocesses p, sysusers u, master..syslogins l
where p.dbid = db_id(@dbname)
and p.suid <>0
and p.suid = l.suid
and u.suid = l.suid
return -4
end

/*cas de la base corrompue*/
if exists (select *
from master..sysdatabases
where name = @dbname
and status2 = status2|256)
begin

```

```

print "The database %1!
is corrupt : check the errorlog to make it stable the problem before uploading !", @dbname
return (-1)
end
else
begin
select @file = file
from DBA..trackDump
where error = 0
and dbid = db_id(@dbname)
and type = "D"
and start = (select max(start)
from DBA..trackDump
where error = 0
and dbid = db_id(@dbname)
and type = "D")
if @file is null
begin
select @file = convert(char(8), @time, 4)+ " at " +convert(char(8), @time, 8)
print "The database %1! has no valid load before %2!", @dbname, @file
return (-2)
end
else
begin
load database @dbname from @file
select @err = @@error
if @err = 7205
begin
print "Load failed : BS probably is down."
return -3
end
if @err = 0
print "Load succeed"

open dump_cur
fetch dump_cur into @file

while @@sqlstatus <> 2
begin
load tran @dbname from @file
select @err = @@error
if @err = 7205
begin
print "Load failed : BS probably is down."
return -3
end
if @err = 0
print "Load succeed"
fetch dump_cur into @file
end
deallocate cursor dump_cur
end
online database @dbname
end
return @err
end
end
go

print "Granting sp_dump... stored procedures"
grant execute on sp__dump_recover to oper_role
grant execute on sp__dump to oper_role
grant execute on sp__dump_clean to oper_role
go
    
```

Procédure permettant de tuer les sessions présentes sur une base spécifique, ceci afin de pouvoir effectuer une restauration en mode utilisateur unique.

```

print "Creating new sp__kill stored procedure"
go
    
```

```

create proc sp__kill (@dbname varchar(30))
as
begin
    set nocount on

    declare @snum varchar(5)
    declare ckill cursor for select convert(char(5),spid) from master..sysprocesses where
    dbid=db_id(@dbname)

    open ckill
    fetch ckill into @snum
    while @@sqlstatus <> 2
    begin
        print "Killing process %1!",@snum
        exec ("kill " + @snum)
        fetch ckill into @snum
    end
end
go
grant execute on sp__kill to oper_role , sa_role, sso_role
go
    
```

Procédure de seuil standard permettant de lancer automatiquement une sauvegarde du journal de transaction au moment où un seuil critique est atteint.

```

use sybssystemprocs
go
if exists(select name from sysobjects where name = "sp_thresholdaction" and type = "P")
begin
    declare @old varchar(30), @debut datetime
    select @debut = getdate()
    select @old = "sp_threshold_" + substring(convert(char, @debut, 112), 1,8) +
        "_" + substring(convert(char,@debut,8),1,2) + substring(convert(char,@debut,8),4,2)

    print "Renaming old sp_thresholdaction to %1!", @old
    exec sp_rename sp_thresholdaction, @old
end
go

print "Creating new sp_thresholdaction stored procedure"
go

create procedure sp_thresholdaction
@dbname varchar(30),
@segmentname varchar(30),
@space_left int,
@status int
as
begin
    declare @taille_avant decimal(7,2),
        @taille_apres decimal(7,2),
        @err int,
        @ficname varchar(128),
        @repname varchar(128),
        @dumpname varchar(255),
        @datet datetime

    /* Tronque les decimales superflus */
    set arithabort numeric_truncation off

    /* journal de transactions (debut) */
    if @segmentname = ( select name from syssegments where segment = 2 )
    begin

        /* Recupere l'espace occupe dans le journal avant la sauvegarde */
        select @taille_avant = (reserved_pgs(id, doampg) + reserved_pgs(id, ioampg))/1024.0
        from sysindexes
        where name = "syslogs"

        /* Effectue la sauvegarde du journal + purge */
    
```

```
exec DBA..sp__dump @dbname, "T"

/* Controle d'erreur */
select @err = @@error

/* Si Erreur sur dump tran to file */
if @err != 0
begin
    print "Impossible de sauvegarder le journal de la base
        '%1!'. (Error Nr '%2!')", @dbname, @err
    return
end

/* Recupere l'espace occupe dans le journal apres la sauvegarde */
select @taille_apres = (reserved_pgs(id, doampg) + reserved_pgs(id, ioampg))/1024.0
from sysindexes
where name = "syslogs"

end
else
begin
    print "BE CAREFULL: THE DATABASE %1! IS FILLING UP !!!", @dbname
    print "Available : %1! (2Kb)", @space_left
end
end
go
print "Granting sp_thresholdaction"
grant execute on sp_thresholdaction to public
go
```

III - Utilisation

Batch Windows permettant de démarrer les sauvegardes de base de données.

```
rem Auteur   : Fabien Celaia
rem Date    : 16.8.00
rem Description : sauvegardes de bases

set CHEMIN=C:\temp

@echo off
echo use DBA > %CHEMIN%\bck_db.sql
echo go >> %CHEMIN%\bck_db.sql
echo exec sp__dump VotreBaseUtilisateur >> %CHEMIN%\bck_db.sql
echo exec sp__dump master >> %CHEMIN%\bck_db.sql
echo go >> %CHEMIN%\bck_db.sql
%sybase%\%sybase_ocs%\bin\isql -Ubatch -P***** -i %CHEMIN%\bck_db.sql -
S%DSQUERY% >> %CHEMIN%\dump_db.log
del %CHEMIN%\bck_db.sql
```

Equivalence sous Unix

```
#!/bin/ksh
# Auteur   : Fabien Celaia
# Date    : 16.8.00
# Description : sauvegardes de bases

CHEMIN=/usr/sybase/log

$SYBASE/$SYBASE_OCS/bin/isql -Ubatch -P***** -i $CHEMIN/bck_db.sql -S $DSQUERY -o $CHEMIN/dump_db.log
-d DBA << EOF
exec sp__dump VotreBaseUtilisateur
exec sp__dump master
echo go
EOF
rm $CHEMIN/bck_db.sql
```

Même appel pour les sauvegardes des journaux de transaction (sous Windows)

```
rem Auteur   : Fabien Celaia
rem Date    : 16.8.00
rem Description : sauvegardes chronique du journal de transactions

@echo off
echo use DBA > c:\temp\bck_tran.sql
echo go >> c:\temp\bck_tran.sql
echo exec sp__dump VotreBaseUtilisateur, 'T' >> c:\temp\bck_tran.sql
echo go >> c:\temp\bck_tran.sql
%sybase%\bin\isql -Ubatch -P***** -i c:\temp\bck_tran.sql -S%DSQUERY% >> g:\sybase\syblog\dump_db.log
del c:\temp\bck_tran.sql
```