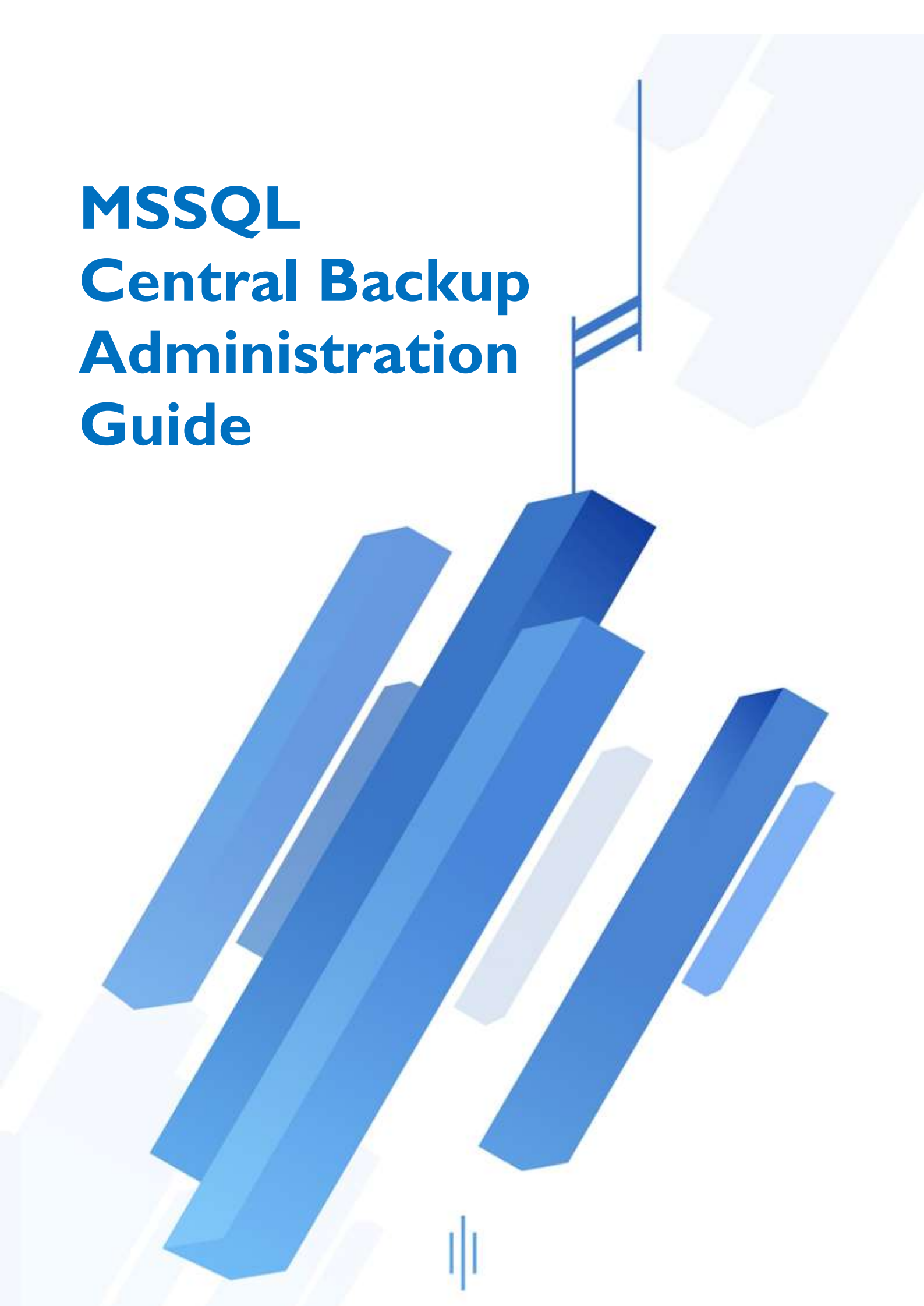


MSSQL Central Backup Administration Guide



1	INTRODUCTION	4
1.1	OBJECTIF DE LA SOLUTION DE SAUVEGARDE MSSQL CENTRALISEE	4
1.2	OUTILLAGE	4
1.3	EXIGENCES DE SECURITE	5
1.3.1	COMPTE DE SERVICE ET PRIVILEGES	5
1.3.2	FLUX RESEAU	5
1.3.3	AUTRES EXIGENCES	5
2	COMPOSANTS LOGICIELS	6
2.1	DBATOOLS	6
2.2	LOGGING	6
2.3	ROBOCOPY	6
2.4	UNE INSTANCE MSSQL (2019+)	6
3	LES SCRIPTS	7
3.1	SCRIPTS DE SAUVEGARDES	7
3.1.1	LE SCRIPT BACKUPDATABASESOneInstance.ps1	7
3.1.2	LE SCRIPT BACKUPDATABASESFromCMS.ps1	13
3.2	SCRIPT DE CENTRALISATION DES SAUVEGARDES VERS UN REPERTOIRE CENTRAL	15
3.2.1	ROBOCOPYFromCMS.ps1	15
3.3	SCRIPTS DE PURGE	17
3.3.1	PURGEFilesFromCMS.ps1	17
4	JOBS SQL ET PLANIFICATION	19
4.1	CREDENTIAL ET PROXY SQL	19
4.2	LES PLANIFICATIONS	20
4.3	LES JOBS	20
4.3.1	JOB MSSQLBACKUPSOLUTION-CMS-BACKUP-FULL-ALL	20
4.3.2	JOB MSSQLBACKUPSOLUTION-CMS-BACKUP-DIFF-ALL	23
4.3.3	JOB MSSQLBACKUPSOLUTION-CMS-BACKUP-LOG-ALL	24
4.3.4	JOB MSSQLBACKUPSOLUTION-PURGE-BACKUPCENTRAL	25
4.3.5	JOB MSSQLBACKUPSOLUTION-PURGE-REMOTE-FROMCMS	26
5	LA BASE DE TRACES ET MONITORING	27
6	LA MISE A JOUR DE LA SOLUTION	28

1 Introduction

1.1 Objectif de la solution de sauvegarde MSSQL centralisée

L'idée est d'avoir une solution de sauvegarde centralisée et unique afin de commander depuis une instance centrale les sauvegardes des différentes base de données situés sur des instances MSSQL cibles en s'appuyant sur la CMS de SQL Server qui définit la liste des instances à sauvegarder.

De cette manière les sauvegardes MSSQL sont toutes cohérentes et basées sur le même modèle.

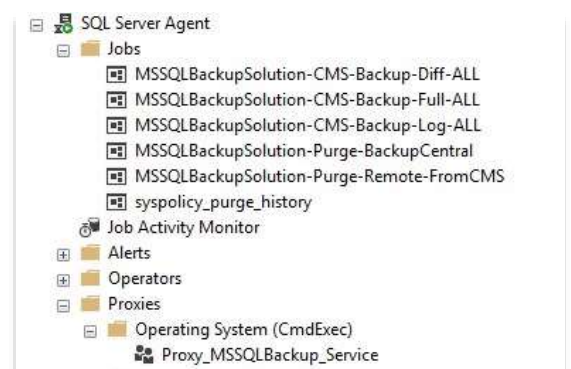
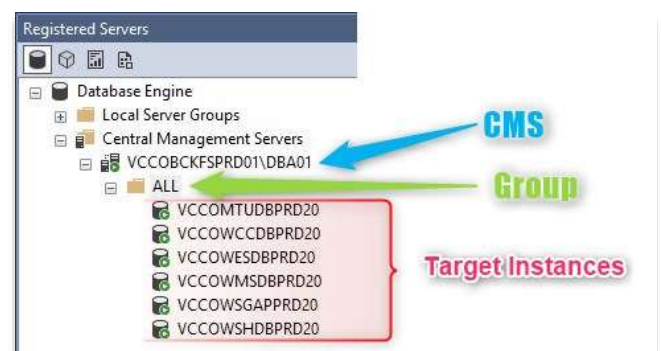
L'ajout ou le retrait d'une instance cible dans la CMS permet d'intégrer ou retirer cette instance (et toutes ces bases de données) dans le processus de sauvegardes centralisées sans avoir à déployer quoi que ce soit sur l'instance cible.

1.2 Outillage

Pour se faire, des scripts **PowerShell** avec la librairie **DBATOOLS** sont utilisés.

La **CMS**, « **Central Management Servers** » en anglais est un registre d'instances SQL Server où l'on va classer et enregistrer les instances d'une entreprise ou d'un site.

- Depuis l'instance centrale, on exécutera **les scripts de backup** pour chaque instance de la CMS embarquée dans le groupe « ALL ». Les backups seront fait localement sur chaque instance cible.
- Dans un second temps, on réalisera la « **robocopy** » des fichiers de backups locaux de chaque instance vers le serveur « Bastion » afin de centraliser les fichiers de sauvegardes.
- Puis, on lancera **des scripts de purge** afin de supprimer les fichiers de sauvegardes trop anciens.
- Afin d'ordonnancer toutes les étapes (backup, robocopy, purge) des **jobs SQL, sur l'instances CMS**, sont utilisés pour lancer les scripts powershell.
- Les résultats des étapes sont tracés dans une **table de suivi**.
- Le détails des étapes sont tracés à la fois dans des fichiers de log et dans les étapes des jobs SQL



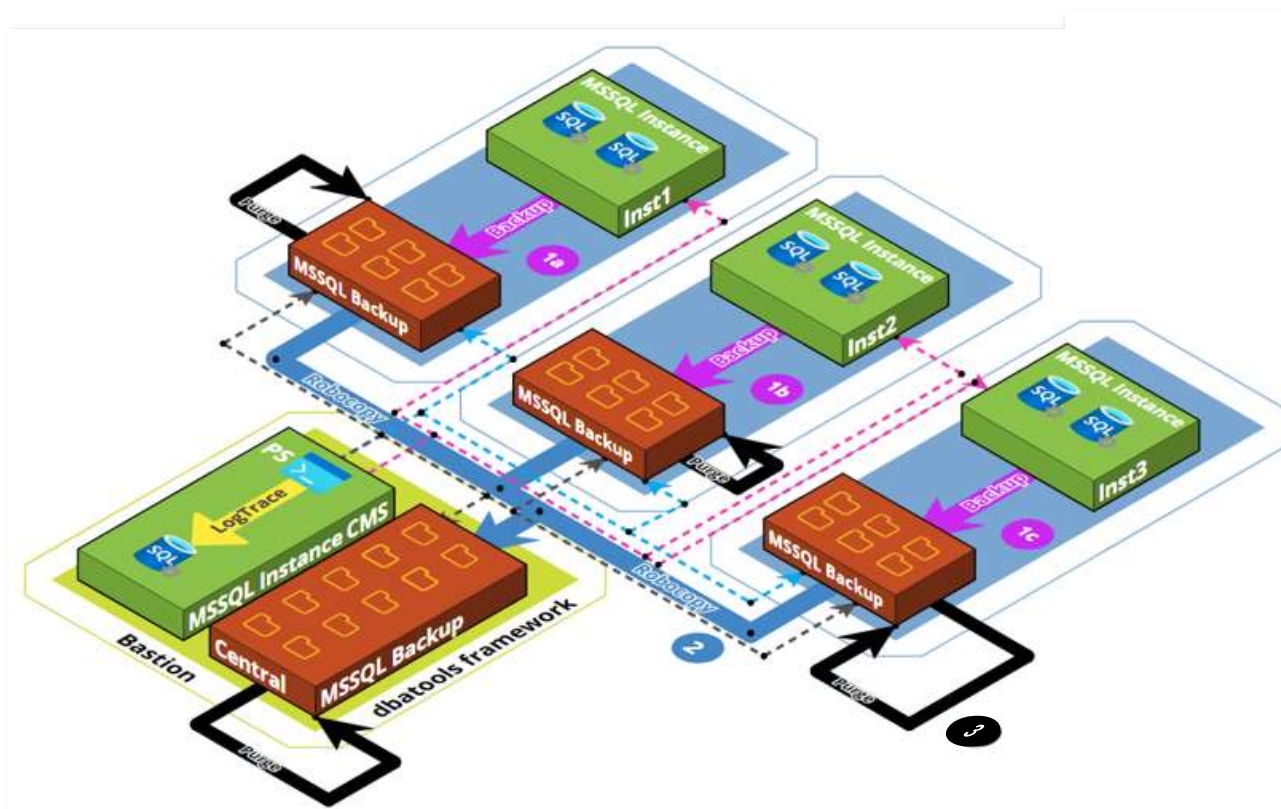
1.3 Exigences de sécurité

1.3.1 Compte de service et privilèges

Afin de pouvoir réaliser les opérations de backup sur les instances cibles, de robocopy des fichiers de backups entre les cibles et le serveur central et afin de se connecter sur l'instance hébergeant la CMS et de tracer les résultats de backups dans un table de trace il est nécessaire d'avoir :

- **Un compte de service sur l'AD qui disposera des droits nécessaires.**
- Positionner ce compte de service dans le groupe BUILTIN\Administrators de chaque machine cible ainsi que sur le serveur central CMS permet de simplifier l'installation

1.3.2 Flux réseau



Afin de permettre les différentes actions de Backup, Robocopy et Purge différents ports TCP doivent être ouverts entre le serveur central et les serveurs hébergeants les instances MSSQL cibles.

Etape	Ports TCP	Commentaire
Backup (1)	1433	Le port de l'instance cible
Robocopy (2)	445 (139)	Port SMB
Purge (3)	445 (139)	Port SMB

1.3.3 Autres exigences

Afin de ne pas mélanger les sauvegardes qui seraient réalisées par différents biais (plans de maintenances, job locaux de backup, système central autre) **il est impératif de ne conserver qu'un seul mécanisme de sauvegarde classique des bases** (full + diff + log).

2 Composants logiciels

La solution utilise plusieurs composants logiciels :

- Le framework powershell dbatools
- Le framework powershell logging
- L'utilitaire robocopy
- Une instance MSSQL pour les traces d'exécution en base de données et pour héberger la CMS

2.1 DBATOOLS

DBATOOLS est une librairie PowerShell réalisé par des membres de la communauté SQL Server notamment par des personnes travaillant directement chez Microsoft sur SQL Server. Cette librairie est destinée aux administrateurs de base de données afin de gérer différentes problématiques via des scripts en ligne de commandes. Cette librairie possède une très bonne documentation et elle est très intuitive, ce qui permet donc de créer des scripts pouvant être rapidement compris par un autre DBA ou des administrateurs systèmes.

Le site dbatools.io regroupe la documentation des commandes de ce framework avec de nombreux exemples : <https://dbatools.io/commands/>

L'installation de ce composant est fait pendant l'installation de la solution MSSQLBACKUP

Ce composant sert à

- lister les bases à sauvegarder
- faire les sauvegardes
- pousser les résultats dans la table de trace

2.2 LOGGING

LOGGING est une librairie PowerShell qui permet de tracer l'exécution des scripts proprement dans des fichiers de journalisation

2.3 ROBOCOPY

ROBOCOPY est un utilitaire Windows qui permet de transférer les fichiers de manière robuste. Il sera utilisé pour rapatrier les fichiers de backups générer sur chaque instance cible vers un répertoire central sur la machine pilotant les backups

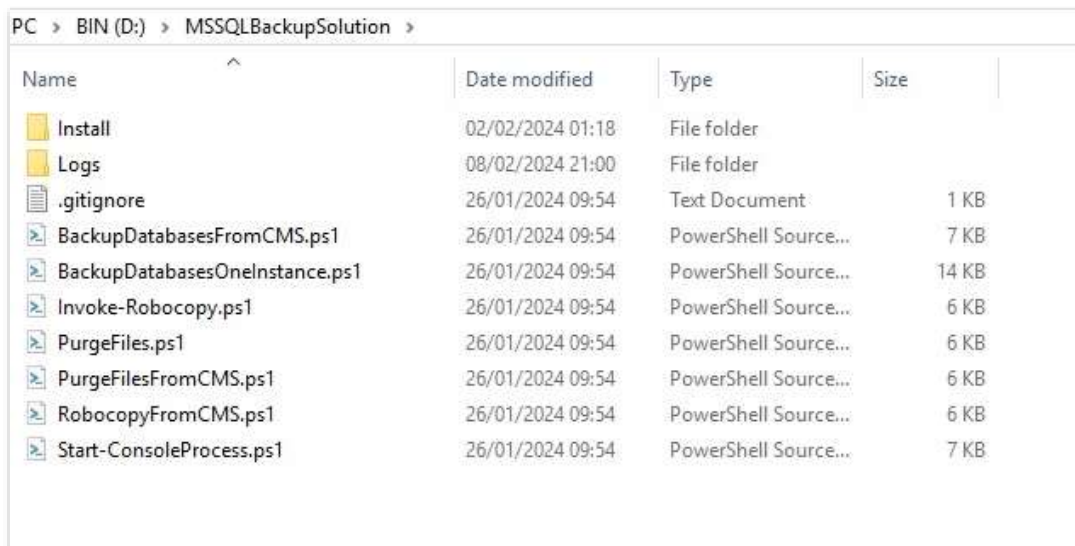
2.4 Une instance MSSQL (2019+)

Une base de données de trace permet de centraliser les résultats d'exécution des sauvegardes et ainsi de disposer du source unique qui permettra le restituer l'états des sauvegardes réalisées et de détecter des problèmes.

NB : Une édition disposant du composant Job Agent est nécessaire.

3 Les scripts

Il y a différents scripts nécessaires pour la solution. Par défaut les scripts sont positionnés sur **D:\MSSQLBackupSolution** et les logs dans le sous répertoires Logs



Name	Date modified	Type	Size
Install	02/02/2024 01:18	File folder	
Logs	08/02/2024 21:00	File folder	
.gitignore	26/01/2024 09:54	Text Document	1 KB
BackupDatabasesFromCMS.ps1	26/01/2024 09:54	PowerShell Source...	7 KB
BackupDatabasesOneInstance.ps1	26/01/2024 09:54	PowerShell Source...	14 KB
Invoke-Robocopy.ps1	26/01/2024 09:54	PowerShell Source...	6 KB
PurgeFiles.ps1	26/01/2024 09:54	PowerShell Source...	6 KB
PurgeFilesFromCMS.ps1	26/01/2024 09:54	PowerShell Source...	6 KB
RobocopyFromCMS.ps1	26/01/2024 09:54	PowerShell Source...	6 KB
Start-ConsoleProcess.ps1	26/01/2024 09:54	PowerShell Source...	7 KB

3.1 Scripts de sauvegardes

Il y a deux scripts en jeu pour la sauvegarde :

- **BackupDatabasesOneInstance.ps1** qui permet de réaliser les backups de toutes les databases d'une seule instance
- **BackupDatabasesFromCMS.ps1** qui permet d'appeler le script précédent pour toutes les instances d'un groupe passé en paramètre

3.1.1 Le script BackupDatabasesOneInstance.ps1

BackupDatabasesOneInstance.ps1 permet de sauvegarder toutes les bases de données d'une instance. Ce script prends les paramètres suivant

3.1.1.1 Paramètres



```
param
(
    [Parameter(Mandatory)] [string] $SqlInstance,
    [Parameter(Mandatory)] [ValidateSet('Full','Diff','Log')] [string] $BackupType,
    [Parameter(Mandatory)] [string] $BackupDirectory,
    [Parameter()] [Int16] $FileCount = 1,
    [Parameter()] [Int16] $FullBackupInterval = 15,
    [Parameter()] [string] $LogLevel = "INFO",
    [Parameter()] [string] $LogDirectory,
    [Parameter()] [string] $LogSQLInstance = "localhost\ORA01",
    [Parameter()] [string] $LogDatabase = "MSSQLBackupSolutionDB"
)
```

- **SqlInstance** : L'instance cible que l'on souhaite sauvegarder.
- **BackupType** : Le type de backup que l'on veut faire : Full, Diff ou Log.
- **BackupDirectory** : Le répertoire sur l'instance cible où seront entreposés les fichiers de backup.
- **FileCount** : Le nombre de fichiers de backup pour une base que l'on souhaite avoir afin de répartir les données pour gagner en performance. Par défaut, un seul fichier est générer par backup.

- **FullBackupInterval** : Le nombre de jours d'écart avec la dernière backup full que l'on tolère, si la condition n'est pas respectée, on réalise une backup full. Cela concerne uniquement lorsque l'on réalise une backup de type Diff ou Log car c'est type de sauvegardes s'appuient sur la dernière backup Full, donc si cette dernière est inexistante ou trop vieille, on prend soin d'en réaliser une nouvelle.
- **LogLevel** : Les niveaux de log des scripts sont actuellement définis dans les scripts directement avec un niveau INFO
- **LogDirectory** : Le répertoire où seront stocké les fichiers de log du script pour voir si l'exécution du script s'est bien déroulée. Garder un œil sur le bon déroulement est très important d'où l'intérêt de mettre en place du log afin de rapidement diagnostiquer un problème.
- **LogInstance** : Instance où l'on va enregistrer le résultat de l'exécution du script.
- **LogDatabase** : Base sur laquelle, on va enregistrer dans des tables, le bon déroulement du script, cela permet d'avoir en plus des fichiers de log, une autre trace de ce qui s'est passé. Le fait de mettre les informations de l'exécution du script dans des tables va permettre de réaliser des analyses en créant par exemple des tableaux de bord.

3.1.1.2 Exemple de lancement manuel

```

C:\Program Files\Microsoft SQL Server\150\Tools\Binn\sqlcmd -i "D:\VETP\01\SSQLBackupSolution\BackupDatabaseOneInstance.ps1" -s localhost -b backupType Diff -b backupDirectory "D:\SSQL\BACKUPS" -f FileCount 1 -l logDirectory "D:\ssqldataling"
logSQLInstance localhost -logDatabase MSSQLBackupSolutionDB

Directory: D:\scripts

Name                LastWriteTime         Length Name
-----
09/02/2024    12:48                4      logs

Directory: D:\scripts\logs

Name                LastWriteTime         Length Name
-----
09/02/2024    12:48                4      localhost

[2024-02-09 12:48:29+01] [INFO] Log File : D:\scripts\logs\localhost\MSSQLBackupSolution_localhost_Diff_2024-02-09_120829.log
[2024-02-09 12:48:29+01] [INFO] Parameter SQLInstance : localhost
[2024-02-09 12:48:29+01] [INFO] Parameter BackupType : Diff
[2024-02-09 12:48:29+01] [INFO] Parameter BackupDirectory : D:\SSQL\BACKUPS
[2024-02-09 12:48:29+01] [INFO] Parameter FileCount : 1
[2024-02-09 12:48:29+01] [INFO] Backup Extension : diff
[2024-02-09 12:48:29+01] [INFO] Backup Diff of localhost - Database : msdb : Successful in 00:00:00 and 264,17 KB
[2024-02-09 12:48:30+01] [INFO] Backup Diff of localhost - Database : DBA01 : Successful in 00:00:00 and 10,22 KB
[2024-02-09 12:48:30+01] [INFO] Backup Diff of localhost - Database : disleapord00 : Successful in 00:00:00 and 780,18 KB
[2024-02-09 12:48:30+01] [INFO] Backup Diff of localhost - Database : disleapord01 : Successful in 00:00:00 and 54,99 KB
[2024-02-09 12:48:31+01] [INFO] Backup Diff of localhost - Database : AdventureWorks2008R3 : Successful in 00:00:01 and 39,50 MB
[2024-02-09 12:48:31+01] [INFO] Backup Diff of localhost - Database : AdventureWorks2008R3 : Successful in 00:00:00 and 125,95 KB
[2024-02-09 12:48:32+01] [INFO] Backup Diff of localhost - Database : AdventureWorks2008R3 : Successful in 00:00:01 and 52,52 KB
[2024-02-09 12:48:32+01] [INFO] Backup Diff of localhost - Database : AdventureWorks2008R3 : Successful in 00:00:00 and 183,38 KB
[2024-02-09 12:48:32+01] [INFO] Backup Diff of localhost - Database : DBASWAMP : Successful in 00:00:00 and 1,77 KB
[2024-02-09 12:48:33+01] [INFO] Backup Diff of localhost - Database : DBASWAMP : Successful in 00:00:00 and 47,88 KB
[2024-02-09 12:48:33+01] [INFO] Backup Diff of localhost - Database : DBATools : Successful in 00:00:00 and 4,02 KB
[2024-02-09 12:48:34+01] [INFO] Backup Diff of localhost - Database : DBATools : Successful in 00:00:00 and 16,58 KB
[2024-02-09 12:48:34+01] [INFO] Backup Diff of localhost - Database : FASTExportData : Successful in 00:00:03 and 148,36 KB
[2024-02-09 12:48:37+01] [INFO] Backup Diff of localhost - Database : FASTExportData : Successful in 00:00:04 and 77,84 KB
[2024-02-09 12:48:37+01] [INFO] Backup Diff of localhost - Database : migratorspress : Successful in 00:00:00 and 437,94 KB
[2024-02-09 12:48:38+01] [INFO] Backup Diff of localhost - Database : migratorspress : Successful in 00:00:00 and 71,09 KB
[2024-02-09 12:48:39+01] [INFO] Backup Diff of localhost - Database : OpenDataLocal : Successful in 00:00:01 and 107,45 KB
[2024-02-09 12:48:40+01] [INFO] Backup Diff of localhost - Database : OpenDataLocal : Successful in 00:00:01 and 60,20 KB
[2024-02-09 12:48:40+01] [INFO] Backup Diff of localhost - Database : SQLMeshutils : Successful in 00:00:00 and 870,05 KB
[2024-02-09 12:48:40+01] [INFO] Backup Diff of localhost - Database : SQLMeshutils : Successful in 00:00:00 and 40,46 KB
[2024-02-09 12:48:40+01] [INFO] Backup Diff of localhost - Database : SPORT_TDF : Successful in 00:00:00 and 3,41 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : SPORT_TDF : Successful in 00:00:00 and 59,45 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : tpch18_collation_bin2 : Successful in 00:00:22 and 9,44 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : tpch18_collation_bin2 : Successful in 00:00:00 and 127,97 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : TSQLCode smells : Successful in 00:00:04 and 740,65 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : TSQLCode smells : Successful in 00:00:00 and 49,62 KB
[2024-02-09 12:48:41+01] [INFO] Backup Diff of localhost - Database : TSQL2012 : Successful in 00:00:00 and 481,80 KB

```


3.1.1.3 Détail du code

Dans la section du code ci-dessous, on réalise pour chaque type de backup, l'extension du fichier de backup qui va être utilisé. De plus, on réalise l'inventaire des bases que seront sauvegardées en vérifiant que le statut « RecoveryModel » est à Full ou bulk logged ce qui signifie que les journaux de la base en question doivent être sauvegardée à l'inverse des bases en mode de récupération **Simple**. Pour finir, on exclut certaine base système (tempdb, model).

```
switch ( $BackupType )
{
    "Full"
    {
        $BackupExtension="bak"
        $Databases = Get-DbaDatabase -SqlInstance $SqlInstance -ExcludeDatabase "tempdb","model"
        -EnableException
        -WarningVariable WarningVariable | Where-Object (( $_.IsUpdateable) -and ( $_.Status -like "Normal*" ))
    }
    "Diff"
    {
        $BackupExtension="diff"
        $Databases = Get-DbaDatabase -SqlInstance $SqlInstance -ExcludeDatabase "tempdb","model","master"
        -EnableException
        -WarningVariable WarningVariable | Where-Object (( $_.IsUpdateable) -and ( $_.Status -like "Normal*" ))
    }
    "Log"
    {
        $BackupExtension="trn"
        $Databases = Get-DbaDatabase -SqlInstance $SqlInstance
        -ExcludeDatabase "tempdb","model" -EnableException
        -WarningVariable WarningVariable | Where-Object [ ( $_.IsUpdateable) -and ( $_.Status -like "Normal*" ) -and ( $_.RecoveryModel -ne
```

Par la suite, on effectue une boucle Foreach afin de sauvegarder chaque base présente dans la liste, en vérifiant dans le cas d'une sauvegarde de type Différentiel ou Log qu'une sauvegarde Full est bien existante ou date de moins de jours que le paramètre **FullBackupInterval**.

On exécute la sauvegarde pour chaque base avec la ligne de commande de DBATOOLS ci-dessous :

```
$SilentRes=Backup-DbaDatabase -SqlInstance $SqlInstance -Database $DatabaseName -Type $BackupType
-CompressBackup -Checksum -Verify -FileCount $FileCount -Path "${BackupDirectory}\servername\instancename\dbname\backuptype"
-FilePath "servername_dbname_backuptype_timestamp.${BackupExtension}" -TimeStampFormat "yyyyMMdd_H#mm"
-ReplaceInName -CreateFolder -WarningVariable WarningVariable -OutVariable BackupResults -EnableException
```

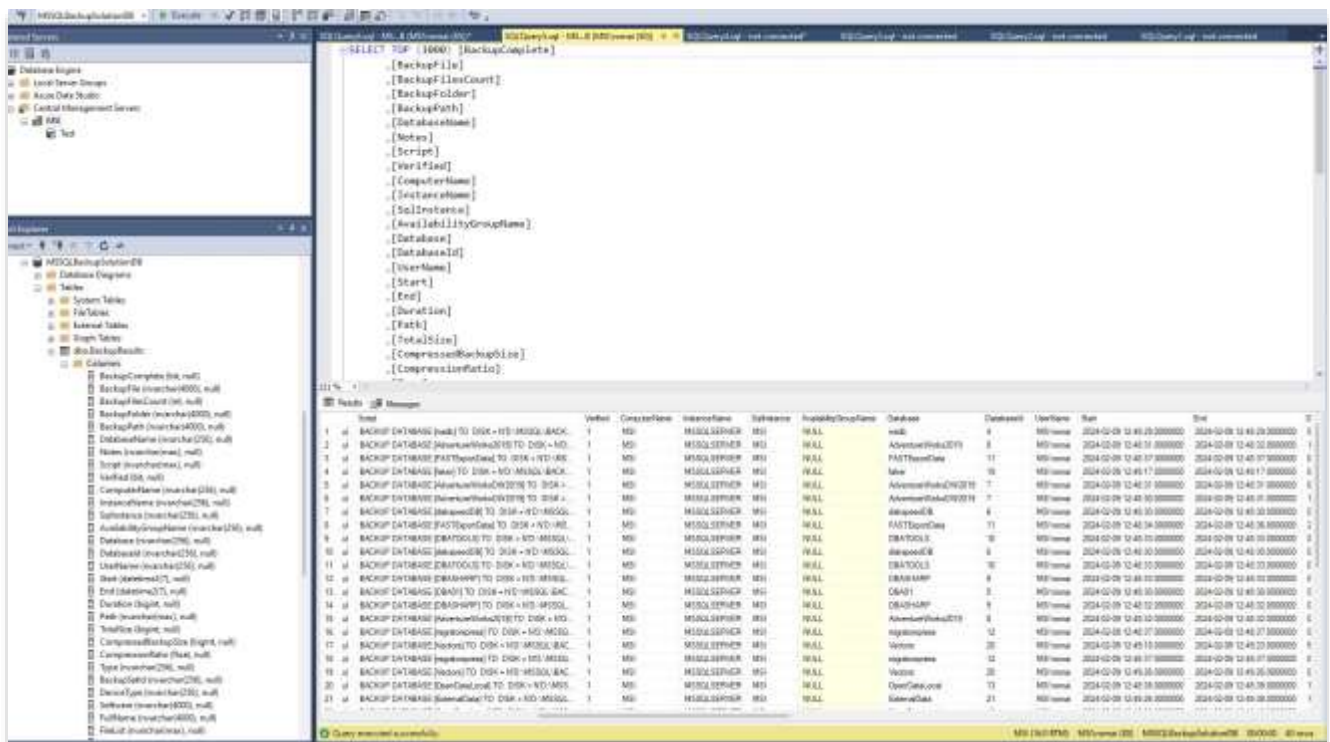
On notera la présence des switch -Checksum et -Verify qui permettent de contrôler respectivement que les blocs de données écrits sont conformes (-Checksum) et qu'à la relecture du fichier de backup celui-ci est bien lisible et valide (-Verify)

On utilise le paramètre « -OutVariable » qui permet de récupérer un tableau contenant toutes les informations de l'exécution du backup pour la base en question, dans une variable, « BackupResults » dans notre cas. Il ne reste plus qu'à insérer ce tableau, dans une table de log afin d'avoir une trace des résultats du Backup sur SQL Server.

```
Write-DbaDbTableData -SqlInstance $logSqlInstance -Database $logDatabase -InputObject $BackupResults
-Table dbo.BackupResults -AutoCreateTable -UseDynamicStringLength
```

3.1.1.4 Traces en base de données :

On peut ensuite requêter la table sur SSMS afin d'obtenir des informations plus précises sur le déroulement des sauvegardes.



JobName	BackupType	BackupTime	BackupSize
BACKUP DATABASE (msdb) TO DISK = 'D:\SQL\Backup\msdb.bak'	Full	2023-12-15 14:41:36	594.45 KB
BACKUP DATABASE (AdventureWorks2017) TO DISK = 'D:\SQL\Backup\AdventureWorks2017.bak'	Full	2023-12-15 14:41:36	199.97 KB
BACKUP DATABASE (DWConfiguration) TO DISK = 'D:\SQL\Backup\DWConfiguration.bak'	Full	2023-12-15 14:41:36	67.89 KB
BACKUP DATABASE (DWQueue) TO DISK = 'D:\SQL\Backup\DWQueue.bak'	Full	2023-12-15 14:41:36	130.12 KB
BACKUP DATABASE (FASTExportData) TO DISK = 'D:\SQL\Backup\FASTExportData.bak'	Full	2023-12-15 14:41:36	63.02 KB
BACKUP DATABASE (tpch10) TO DISK = 'D:\SQL\Backup\tpch10.bak'	Full	2023-12-15 14:41:36	4.21 GB
BACKUP DATABASE (DBA01_Bis) TO DISK = 'D:\SQL\Backup\DBA01_Bis.bak'	Full	2023-12-15 14:41:36	402.00 KB
BACKUP DATABASE (Test) TO DISK = 'D:\SQL\Backup\Test.bak'	Full	2023-12-15 14:41:36	173.82 KB
BACKUP DATABASE (WideWorldImporters) TO DISK = 'D:\SQL\Backup\WideWorldImporters.bak'	Full	2023-12-15 14:41:36	16.55 MB
BACKUP DATABASE (AdventureWorks2017Backup) TO DISK = 'D:\SQL\Backup\AdventureWorks2017Backup.bak'	Full	2023-12-15 14:41:36	95.30 KB
BACKUP DATABASE (TestEncryption) TO DISK = 'D:\SQL\Backup\TestEncryption.bak'	Full	2023-12-15 14:41:36	140.99 KB
BACKUP DATABASE (DuplicateWideWorldImporters) TO DISK = 'D:\SQL\Backup\DuplicateWideWorldImporters.bak'	Full	2023-12-15 14:41:36	16.55 MB
BACKUP DATABASE (DBA01) TO DISK = 'D:\SQL\Backup\DBA01.bak'	Full	2023-12-15 14:41:36	131.69 KB
BACKUP DATABASE (AdventureWorks2017Backup) TO DISK = 'D:\SQL\Backup\AdventureWorks2017Backup.bak'	Full	2023-12-15 14:41:36	95.30 KB
BACKUP DATABASE (MyProductDB) TO DISK = 'D:\SQL\Backup\MyProductDB.bak'	Full	2023-12-15 14:41:36	141.24 KB
BACKUP DATABASE (MSSQLShrinkSolutionDB) TO DISK = 'D:\SQL\Backup\MSSQLShrinkSolutionDB.bak'	Full	2023-12-15 14:41:36	273.97 KB
BACKUP DATABASE (DBA01) TO DISK = 'D:\SQL\Backup\DBA01.bak'	Full	2023-12-15 14:41:36	1.03 MB
BACKUP DATABASE (Test1) TO DISK = 'D:\SQL\Backup\Test1.bak'	Full	2023-12-15 14:41:36	97.85 KB
BACKUP DATABASE (Test2) TO DISK = 'D:\SQL\Backup\Test2.bak'	Full	2023-12-15 14:41:36	602.62 KB
BACKUP DATABASE (Test2) TO DISK = 'D:\SQL\Backup\Test2.bak'	Full	2023-12-15 14:41:36	68.94 KB
BACKUP DATABASE (Test2) TO DISK = 'D:\SQL\Backup\Test2.bak'	Full	2023-12-15 14:41:36	68.94 KB

3.1.1.5 Log fichier :

Nous allons voir ci-dessous à quoi ressemble un fichier de log pour un backup d'une instance. Grâce à Visual Studio Code on arrive donc bien à visualiser que tout c'est bien passé avec le code couleur vert, et en rouge si une erreur s'est produite. De plus, on remarque bien les différents paramètres avec lesquels le script a été exécuté. Puis, on peut observer pour chaque base de données le temps nécessaire effectuer la sauvegardes et la taille du fichiers qui a été généré.

```
[2023-12-15 14:41:36+01] [INFO] Log File : C:\Stage\PowerShell\BackupCentral\Logs\DESKTOP_I71QL69\MSSQLBackupSolution_DESKTOP_I71QL69
[2023-12-15 14:41:36+01] [INFO] Parameter SQLInstance : DESKTOP-I71QL69
[2023-12-15 14:41:36+01] [INFO] Parameter BackupType : Diff
[2023-12-15 14:41:36+01] [INFO] Parameter BackupDirectory : C:\Stage\PowerShell\BackupCentral\Backup
[2023-12-15 14:41:36+01] [INFO] Parameter FileCount : 1
[2023-12-15 14:41:36+01] [INFO] Backup Extension : diff
[2023-12-15 14:41:45+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : msdb : Successful in 00:00:00 and 594.45 KB
[2023-12-15 14:41:49+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DWDiagnositics : Successful in 00:00:00 and 199.97 KB
[2023-12-15 14:41:53+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DWConfiguration : Successful in 00:00:00 and 67.89 KB
[2023-12-15 14:41:58+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DWQueue : Successful in 00:00:00 and 130.12 KB
[2023-12-15 14:42:03+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : FASTExportData : Successful in 00:00:00 and 63.02 KB
[2023-12-15 14:42:53+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : tpch10 : Successful in 00:00:13 and 4.21 GB
[2023-12-15 14:42:57+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DBA01_Bis : Successful in 00:00:00 and 402.00 KB
[2023-12-15 14:43:01+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : Test : Successful in 00:00:00 and 173.82 KB
[2023-12-15 14:43:07+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : WideWorldImporters : Successful in 00:00:00 and 16.55 MB
[2023-12-15 14:43:11+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : AdventureWorks2017 : Successful in 00:00:00 and 95.30 KB
[2023-12-15 14:43:17+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : TestEncryption : Successful in 00:00:00 and 140.99 KB
[2023-12-15 14:43:22+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DuplicateWideWorldImporters : Successful in 00:00:00 and 16.55 MB
[2023-12-15 14:43:26+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DBA01 : Successful in 00:00:00 and 131.69 KB
[2023-12-15 14:43:30+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : AdventureWorks2017Backup : Successful in 00:00:00 and 95.30 KB
[2023-12-15 14:43:35+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : MyProductDB : Successful in 00:00:00 and 141.24 KB
[2023-12-15 14:43:39+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : MSSQLShrinkSolutionDB : Successful in 00:00:00 and 273.97 KB
[2023-12-15 14:43:43+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : DBA01 : Successful in 00:00:00 and 1.03 MB
[2023-12-15 14:43:47+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : Test1 : Successful in 00:00:00 and 97.85 KB
[2023-12-15 14:43:51+01] [INFO] Backup Full of DESKTOP-I71QL69 - Database : Test2 : Successful in 00:00:00 and 602.62 KB
[2023-12-15 14:43:56+01] [INFO] Backup Diff of DESKTOP-I71QL69 - Database : Test2 : Successful in 00:00:00 and 68.94 KB
[2023-12-15 14:43:57+01] [INFO] Backup Diff of databases on DESKTOP-I71QL69 : Successful in 140.4748029 seconds
```

3.1.1.6 Arborescence des fichiers de backups :

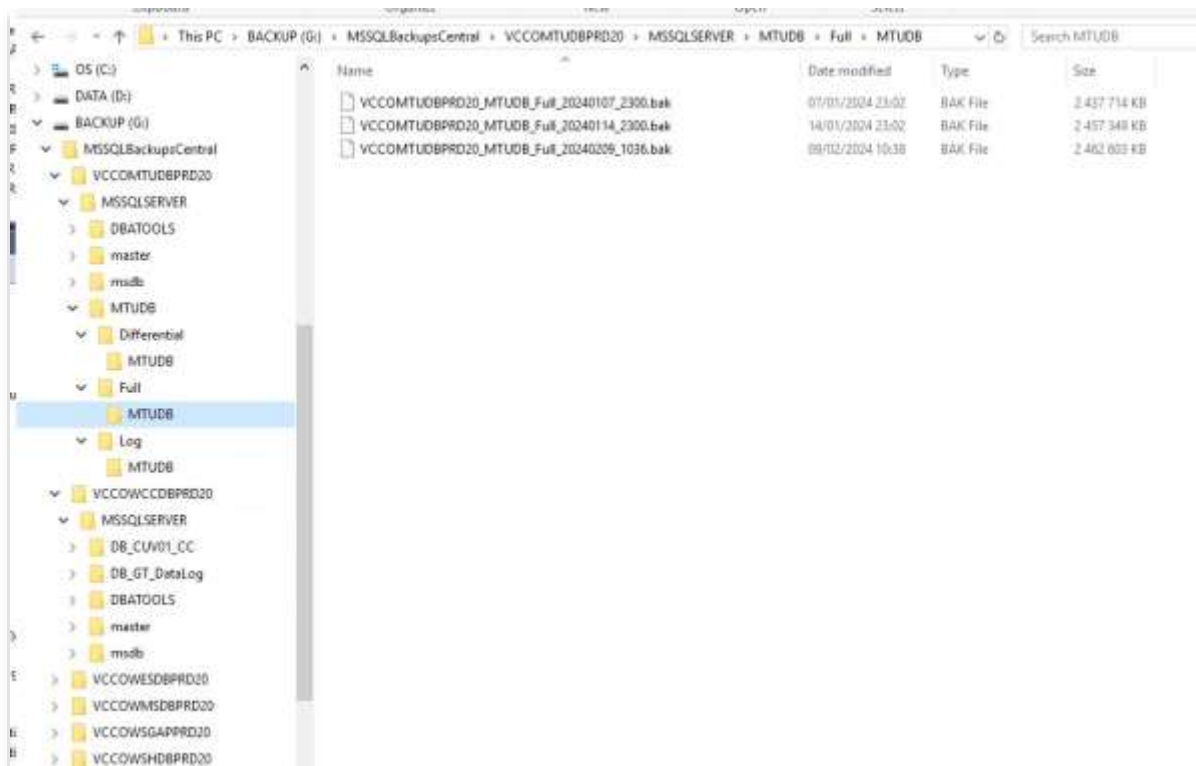
Pour chaque instances on obtient les différents dossiers normalisés pour chacune des bases de données que l'on a sauvegardées. Il y a une racine commune fixe puis une arborescence qui suit la norme suivante :

```
G:\BACKUPDB\  
  <SERVERNAME>\  
    <INSTANCENAME>\  
      <DATABASENAME>\  
        <BACKUPTYPE>\  
          <DATABASENAME>\Files
```

NB : Si l'instance n'est pas nommée, <INSTANCENAME> prend la valeur « MSSQLSERVER »

Sur le serveur CMS centralisé les arborescences se retrouvent donc également de la même manière mais sur une racine différente :

```
G:\MSSQLBackupsCentral\  
  <SERVERNAME>\  
    <INSTANCENAME>\  
      <DATABASENAME>\  
        <BACKUPTYPE>\  
          <DATABASENAME>\Files
```




3.1.1.7 Nommage des fichiers de backups

<INSTANCENAME>_<DATABASENAME>_<BACKUPTYPE>_<YYYYMMDD>_<HHMM>.<Suffix>

Suffix :

- Full ➔ .bak
- Differential ➔ .diff
- Logs ➔ .trn

Exemple :

Name	Date modified	Type	Size
 DESKTOP-I7IQL69_Test2_Full_20231215_1443.bak	12/15/2023 2:43 PM	BAK File	620 KB

3.1.2 Le script BackupDatabasesFromCMS.ps1

BackupDatabasesFromCMS.ps1 permet de récupérer toutes les instances du groupe de la CMS que l'on souhaite sauvegarder et exécute pour chacune d'entre elle, le script de backup **BackupDatabasesOneInstance.ps1** en propageant les autres paramètres (BackupType, BackupDirectory....)

NB : C'est ce script qui sera lancé dans les jobs SQL.

3.1.2.1 Paramètres

```
param
(
    [Parameter(Mandatory)] [ValidateSet('Full','Diff','Log')] [string] $BackupType,
    [Parameter(Mandatory)] [string] $BackupDirectory,
    [Parameter(Mandatory)] [string] $CMSSqlInstance,
    [Parameter(Mandatory)] [string] $ExecDirectory,
    [Parameter()] [int16] $FileCount = 1,
    [Parameter()] [int16] $FullBackupInterval = 15,
    [Parameter()] [string] $Group = "All",
    [Parameter()] [int16] $Timeout = 3600,
    [Parameter()] [string] $LogLevel = "INFO",
    [Parameter()] [string] $LogDirectory,
    [Parameter()] [string] $LogSqlInstance = "localhost\DBA01",
    [Parameter()] [string] $LogDatabase = "MSSQLBackupSolution06"
)
```

- **CMSSqlInstance** : L'instance qui héberge la CMS.
- **Group** : Le nom du groupe racine de la CMS qui permettra de lister les instances à sauvegarder. Le parcours est récursif. Les instances qui seraient dans d'autres sous-groupes seront bien prises en compte.
- **BackupType** : Le type de backup que l'on veut faire : Full, Diff ou Log.
- **BackupDirectory** : Le répertoire sur l'instance cible où seront entreposés les fichiers de backup.
- **FileCount** : Le nombre de fichiers de backup pour une base que l'on souhaite avoir afin de répartir les données pour gagner en performance. Par défaut, un seul fichier est généré par backup.
- **FullBackupInterval** : Le nombre de jours d'écart avec la dernière backup full que l'on tolère, si la condition n'est pas respectée, on réalise une backup full. Cela concerne uniquement lorsque l'on réalise une backup de type Diff ou Log car ce type de sauvegardes s'appuie sur la dernière backup Full, donc si cette dernière est inexistante ou trop vieille, on prend soin d'en réaliser une nouvelle.
- **LogDirectory** : Le répertoire où seront stockés les fichiers de log du script pour voir si l'exécution du script s'est bien déroulée.
- **LogInstance** : Instance où l'on va enregistrer l'exécution du script, dans notre cas, l'instance sera le « Bastion ».
- **LogDatabase** : Base sur laquelle, on va enregistrer dans des tables, le bon déroulement du script, cela permet d'avoir en plus des fichiers de log, une autre trace de ce qui s'est passé.
- **LogLevel** : Les niveaux de log des scripts sont actuellement définis dans les scripts directement avec un niveau INFO

3.1.2.2 Détail du code

Récupération des instances de la CMS :

```
#####
## BACKUP PREPARATION : Get Instances in CMS to run BackupDatabasesOneInstance.ps1 script
#####

try {
    $InstancesName=Get-DbaRegServer -SqlInstance $CMS$SqlInstance -Group $Group -EnableException -WarningVariable warningVariable | Select-Object -Unique Name
    $CMSBackupStartTimeStamp = Get-Date -UFormat "YY-MM-dd HH:MM:SS"

    Write-Log -Level DEBUG -Message "Starting backup of ${Group} group of the CMS ${CMS$SqlInstance} at ${CMSBackupStartTimeStamp}"
    $FailedInstance=@()
}
catch {
    Write-Log -Level ERROR -Message "Impossible to connect to CMS instance : ${SqlInstance}"
    $WarningVariable.Message | ForEach-Object{
        Write-Log -Level ERROR -Message $_
    }
}

Exit 1
}
```

Exécution du script BackupDatabasesOneInstance.ps1 pour chaque instance récupérée avec propagation des paramètres (BackupType, FileCount, Logdirectory, LogInstance...) :

```
#####
## BACKUP : run BackupDatabasesOneInstance.ps1 script for Each Instance found
#####
$InstancesName | ForEach-Object {
    $InstanceName=$_ .Name
    Write-Log -Level DEBUG -Message "Starting backup of instance ${InstanceName}"
    cd $ExecDirectory
    .\BackupDatabasesOneInstance.ps1 -SqlInstance $InstanceName -BackupType $BackupType -FileCount $FileCount
    -FullBackupInterval $FullBackupInterval -BackupDirectory $BackupDirectory -LogDirectory $LogDirectory -LogSQLInstance $LogSQLInstance
    -LogDatabase $LogDatabase
    $RC=$LASTEXITCODE
    if($RC -ne 0){
        $FailedInstance+= $InstanceName
    }
}
}
```


3.2 Script de centralisation des sauvegardes vers un répertoire central

Ce qu'il faut bien comprendre c'est que l'on pilote, depuis le bastion central, l'exécution des scripts de sauvegardes, en revanche, les fichiers de sauvegardes créés sont présent uniquement sur les serveurs des instances cibles.

Une fois les scripts de sauvegarde terminés, il faut rapatrier les fichiers de sauvegarde présent sur chaque instance vers le serveur central. Pour cela nous utilisons des scripts et modules powershell qui feront appel à l'utilitaire **robocopy** et remapperont les codes erreurs pour une meilleure gestion des erreurs.

3.2.1 RobocopyFromCMS.ps1

RobocopyFromCMS.ps1, va chercher lui aussi dans la CMS toutes les instances pour lesquelles on souhaite centraliser les fichiers de backup. Puis pour chaque instance, on copie les fichiers de sauvegardes vers le Bastion via une commande **Invoke-Robocopy** qui permet de remonter une éventuelle erreur à l'appelant.

C'est ce script qui est appelé dans les jobs MSSQL

3.2.1.1 Paramètres

```
param
(
    [Parameter(Mandatory)] [string] $CentralBackupDirectory,
    [Parameter(Mandatory)] [string] $RemoteBackupDirectory,
    [Parameter(Mandatory)] [string] $CMSSqlInstance,
    [Parameter(Mandatory)] [string] $ExecDirectory,
    [Parameter()] [string] $Group = "All",
    [Parameter()] [Int16] $Timeout = 3600,
    [Parameter()] [string] $LogLevel = "INFO",
    [Parameter()] [string] $LogDirectory
)
```

- **CentralBackupDirectory** : Le répertoire où vont être centralisés tous les fichiers de sauvegarde.
- **RemoteBackupDirectory** : Le répertoire racine où sont positionnés les fichiers de sauvegarde sur l'instance distante.
- **CMSSqlInstance** : L'instance qui héberge la CMS.
- **ExecDirectory** : Le répertoire où se situe le script.
- **Group** : Le nom du groupe de la CMS où l'on souhaite copier les fichiers de sauvegarde.
- **Timeout** : Timeout de robocopy
- **LogDirectory** : Le répertoire où seront stockés les fichiers de log pour voir si l'exécution du script s'est bien déroulée.
- **LogLevel** : Les niveaux de log des scripts sont actuellement définis dans les scripts directement avec un niveau INFO

3.2.1.2 Détails du code

Récupération les instances avec les fichiers de sauvegardes à copié.

```
#####  
## ROBOCOPY PREPARATION : Get Instances in CMS to run Invoke-Robocopy for each Instance  
#####  
$ServersName=Get-DbRegServer -SqlInstance $CMSSqlInstance -Group $Group | select -Unique ServerName
```

La difficulté avec le Robocopy est le remapping des codes erreurs :

```
#####  
## Robocopy : run Invoke-Robocopy for each Instance found  
#####  
#####  
$ServersName | ForEach-Object {  
    $ComputerName=$_.ServerName  
  
    $SourceRemoteDir = "\\${ComputerName}\${RemoteBackupDirectory}"  
    Write-Log -Level INFO -Message "Starting Robocopy of Computer ${ComputerName} from ${SourceRemoteDir} to ${CentralBackupDirectory}"  
    Invoke-Robocopy -Source $SourceRemoteDir -Destination $CentralBackupDirectory -ArgumentList @"/e", "/np", "/ndl", "/mt:1" -Verbose  
    $RC=$LASTEXITCODE  
  
    if($RC -gt 4){  
        $FailedRobocopy+=$ComputerName  
    }  
}  
#####  
## Log and EXITCODE  
#####  
$CMSRobocopyEndTimestamp = Get-Date -Format "EY-Md-Md Hr:Qn:Ks"  
$span= New-TimeSpan -Start $CMSRobocopyStartTimestamp -End $CMSRobocopyEndTimestamp  
  
$CMSRobocopyDuration=$span.TotalSeconds  
  
if($FailedRobocopy.count -eq 0){  
    $NbFailedRobocopy=0  
    Write-Log -Level INFO -Message "Robocopy of ${Group} group of the CMS ${CMSsqlInstance} finish successfully in ${CMSRobocopyDuration} seconds".  
}  
else{  
    $NbFailedRobocopy=$FailedRobocopy.count  
    $FailedRobocopy | ForEach-Object {  
        $NameOfRobocopyFailed+=$_ + "  
    }  
    Write-Log -Level ERROR -Message "Robocopy of ${Group} group of the CMS ${CMSsqlInstance} finish with errors in ${CMSRobocopyDuration} seconds. $[NbFail  
}  
  
Wait-Logging  
$ExitCode=0  
if ($NbFailedRobocopy -gt 0)  
{  
    $ExitCode=1  
}  
Exit $ExitCode
```

Récupération d'une erreur Robocopy (RC > 4)

On peut ainsi remonter les erreurs réelles de Robocopy et pas les warnings voir les copies sans problèmes mais qui remonte un RC à 1 au lieu de 0.

3.3 Scripts de purge

Dans un troisième temps, une fois la sauvegarde des instances effectuée et les fichiers copiés sur le « Bastion », il faut **purger** des fichiers de sauvegardes et des fichiers de logs trop anciens afin de ne pas saturer l'espace de stockage de chaque instance local et de l'instance centrale.

Si on le souhaite on peut déterminer un **délai de rétention** différents suivant le type de backup (full, diff ou log). Par exemple, garder les sauvegardes Complètes/Full 30 jours, les sauvegardes Différentielles /Diff 2 semaines et 1 semaine pour les logs.

On utilise pour cela « **PurgeFilesFromCMS.ps1** » qui purge les fichiers d'une extension précise pour une instance avec une valeur adaptée pour le paramètre délai de rétention : \$HoursDelay.

3.3.1 PurgeFilesFromCMS.ps1

C'est ce script qui est appelé dans les jobs MSSQL

3.3.1.1 Paramètres

```
param
(
    [Parameter(Mandatory)] [ValidateSet('Full','Diff','Log','ShellLog')] [string] $FileType,
    [Parameter(Mandatory)] [string] $RemoteBackupDirectory,
    [Parameter(Mandatory)] [string] $CMSSqlInstance,
    [Parameter(Mandatory)] [string] $ExecDirectory,
    [Parameter()] [string] $Group = "All",
    [Parameter()] [int] $HoursDelay=168, #24*7
    [Parameter()] [string] $LogLevel = "INFO",
    [Parameter()] [string] $LogDirectory
)
```

Le délai de rétention

- **FileType** : Le type de fichier que l'on souhaite purger donc soit Full, Diff, Log pour les fichiers de backup et ShellLog pour les fichiers de logs générés par les scripts PowerShell.
- **RemoteBackupDirectory** : Le répertoire qui contient les fichiers de sauvegarde ou de logs. Ce répertoire permettra de construire un chemin UNC avec le nom de la machine cible. Exemple « G\$\BACKUPDB »
- **CMSSqlInstance** : L'instance qui héberge la CMS.
- **Group** : Le nom du groupe de la CMS que l'on souhaite purger.
- **HoursDelay** : **Délai de rétention des backups en heures**
- **LogLevel** : Niveau de log (INFO par défaut)
- **LogDirectory** : Le répertoire où seront stockés les fichiers de log pour voir si l'exécution du script s'est bien déroulée.

3.3.1.2 Détails du code

On récupère dans la variable **ServersName**. toutes les instances faisant parti du groupe de la CMS passé en paramètre

```
#####  
## Purge PREPARATION : Get Instances in CMS to run PurgeFiles.ps1 for each Instance  
#####  
$ServersName=Get-DbRegServer -SqlInstance $CMSsqlInstance -Group $Group | select -Unique ServerName
```

La chaine de caractère **SourceRemoteDir** permet de construire et pointer sur le répertoire distant de chaque instance qui contient les fichiers à purger en concaténant le nom de la machine avec le paramètre \$RemoteBackupDirectory.

Pour finir, on exécute le script « PurgeFiles.ps1 » avec les bons paramètres.

```
#####  
## Purge : run Purgefiles.ps1 Each Instance found  
#####  
$FailedPurge=@()  
$ServersName | ForEach-Object {  
    $ComputerName=$_.ServerName  
  
    $SourceRemoteDir = "\\${ComputerName}\${RemoteBackupDirectory}"  
    Write-log -Level INFO -Message "Starting Purging ${FileType} files on Computer ${ComputerName} from ${SourceRemoteDir} older than ${HoursDelay} hours"  
    cd $ExecDirectory  
    .\PurgeFiles.ps1 -FileType $FileType -RootDirectory $SourceRemoteDir -LogDirectory $LogDirectory -HoursDelay $HoursDelay  
    $RC=$LASTEXITCODE  
  
    if($RC -gt 0){  
        $FailedPurge+= $ComputerName  
        Write-log -Level ERROR -Message "Failed Purging ${FileType} files on Computer ${ComputerName} from ${SourceRemoteDir} older than ${HoursDelay} hours"  
    }  
}
```

3.3.1.3 Fichiers de trace d'exécution :

Les fichiers de traces pour les purges sont préfixé

```
MicrosoftBackupSolution_PurgeFromCMS_2024-02-09.log  Purge_Diff_2024-02-09.log  Purge_Log_2024-02-09.log  
D:\MicrosoftBackupSolution> type Purge_Diff_2024-02-09.log  
322 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
323 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
324 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
325 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
326 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
327 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
328 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
329 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
330 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\VCCMCCDBPRD28_usdb_Differential  
331 [2024-02-09 10:15:47+01] [INFO] | Purge of .diff files in \\VCCMCCDBPRD28\US\BACKUPDB : Successful purge of 18 files in 0.3312892 seconds  
332 [2024-02-09 10:15:47+01] [INFO] | Starting Purging Diff files on Computer VCCMCCDBPRD28 from \\VCCMCCDBPRD28\US\BACKUPDB older than 100 hours  
333 [2024-02-09 10:15:47+01] [INFO] | Log File : D:\MicrosoftBackupSolution\Logs\Purge_Diff_2024-02-09.log  
334 [2024-02-09 10:15:47+01] [INFO] | Parameter FileType : Diff  
335 [2024-02-09 10:15:47+01] [INFO] | Parameter RootDirectory : \\VCCMCCDBPRD28\US\BACKUPDB  
336 [2024-02-09 10:15:47+01] [INFO] | Parameter HoursDelay : 100  
337 [2024-02-09 10:15:47+01] [INFO] | File Extension : .diff  
338 [2024-02-09 10:15:47+01] [INFO] | Found 24 files to delete  
339 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
340 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
341 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
342 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
343 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
344 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
345 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
346 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
347 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
348 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
349 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
350 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
351 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
352 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
353 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
354 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
355 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
356 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
357 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
358 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
359 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
360 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
361 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
362 [2024-02-09 10:15:47+01] [INFO] | Starting the delete of \\VCCMCCDBPRD28\US\BACKUPDB\VCCMCCDBPRD28\SSQLSERVER\usdb\Diffrential\usdb\usdbtools\VCCMCCDBPRD28_usdbtools  
363 [2024-02-09 10:15:47+01] [INFO] | Purge of .diff files in \\VCCMCCDBPRD28\US\BACKUPDB : Successful purge of 24 files in 0.888814 seconds  
364
```


4 Jobs SQL et planification

La solution est déployée avec plusieurs jobs permettant de planifier les sauvegardes FULL, DIFF et LOG ainsi que les purges sur les serveurs distants et sur le serveur central.



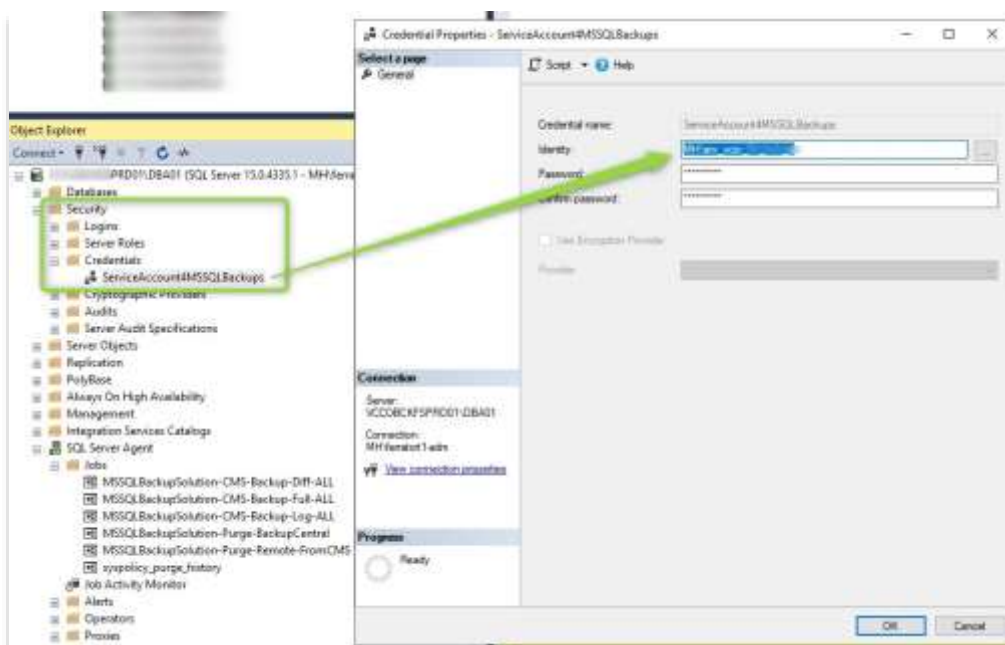
Afin d'obtenir une conservation des journaux de l'agent SQL plus longue il est recommandé d'augmenter la profondeur d'historique de l'agent

Pour cela lancer la commande SQL suivante avec un utilisateur sysadmin

```
EXEC msdb.dbo.sp_set_sqlagent_properties @jobhistory_max_rows=999999,  
@jobhistory_max_rows_per_job=10000  
GO
```

4.1 Credential et Proxy SQL

Chaque job est composé d'étapes. Chaque étape s'occupe de lancer un script powershell. Pour que la remontée des erreurs soit correctement effectuée il faut que les étapes soient de type CmdExec et qu'un Proxy soit défini. Le proxy s'appuie sur un credential (un objet credential dans l'instance MSSQL Centrale). Ce credential nommé **ServiceAccount4MSSQLBackups** est créé à l'installation mais, si nécessaire, il est possible de le modifier.



4.2 Les planifications

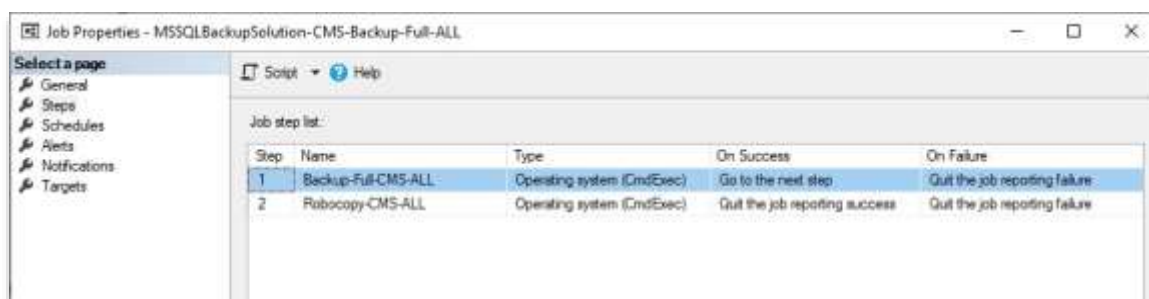
Lors de l'installation des planifications par défaut sont créées.

Job	Fréquence	Planification
Backup Full	Hebdomadaire	Dimanche 23h
Backup Diff	Quotidien	Quotidien 23h (sauf dimanche)
Backup Log	Horaire	Toutes les 30 minutes
Purge Cibles Distantes	Quotidien	Quotidien 21h
Purge Central	Quotidien	Quotidien 22h

NB : La solution étant centralisée et séquentielle, les sauvegardes auront lieu au fil de l'eau et les temps de démarrage sur les serveurs distants ne sera pas identique entre les différents lancements. Cette solution a le mérite de limiter l'impact sur les disques centraux en évitant un pic de charge trop important si tous les serveurs avaient fait leur backups exactement au même moment.

4.3 Les jobs

4.3.1 Job MSSQLBackupSolution-CMS-Backup-Full-ALL



Les jobs de backups sont tous composés de 2 étapes : La partie Backup à proprement parlé, puis la partie Robocopy qui se déclenche dans la foulée.

4.3.1.1 L'étape de backup

Cette étape appelle le script `D:\MSSQLBackupSolution\BackupDatabasesFromCMS.ps1` avec le paramètre `BackupType = Full`




```
powershell.exe -nopprofile -executionpolicy bypass -File
"D:\MSSQLBackupSolution\BackupDatabasesFromCMS.ps1" -CMSSqlInstance "localhost\DBA01"
-Group "ALL" -BackupType "Full" -BackupDirectory "G:\BACKUPDB" -LogDirectory
"D:\MSSQLBackupSolution\Logs" -ExecDirectory "D:\MSSQLBackupSolution"
```

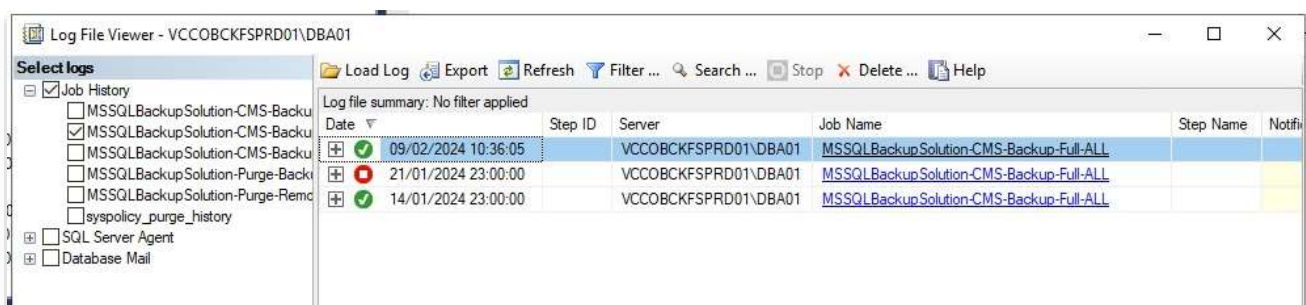
4.3.1.2 L'étape de robocopy

L'étape suivante lance le script D:\MSSQLBackupSolution\RobocopyFromCMS.ps1 qui permet de rapatrier tous les fichiers qui n'ont pas pu être déjà copier sur le central



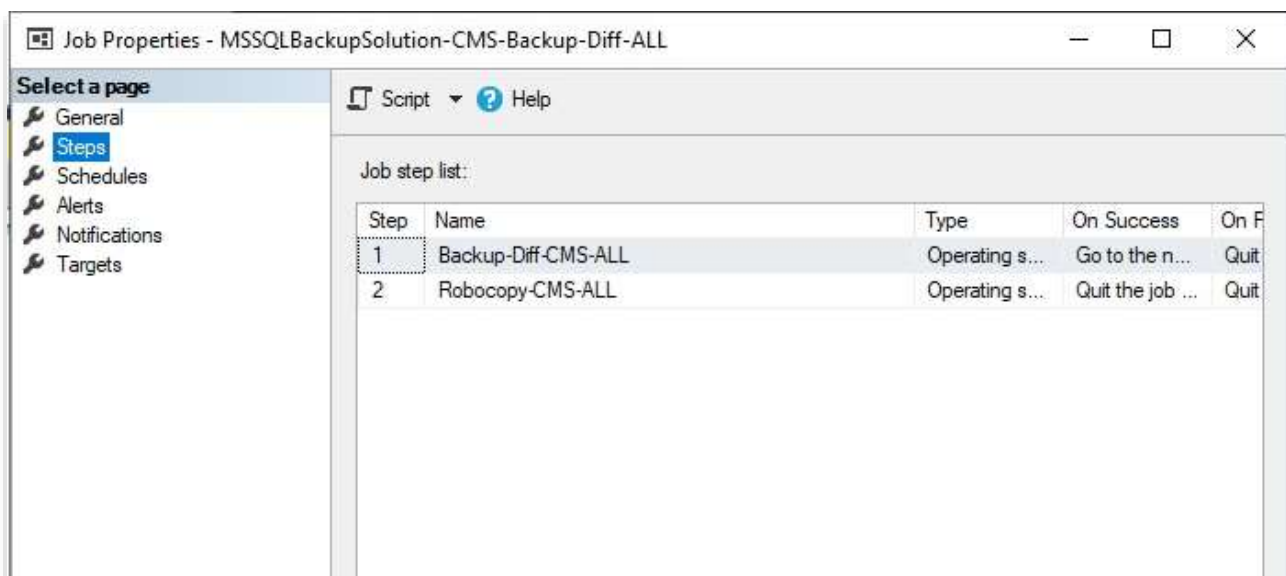
4.3.1.3 Log de l'agent

Grace au capacité de l'agent SQL il est possible de suivre les logs des traitements directement via l'historique des jobs



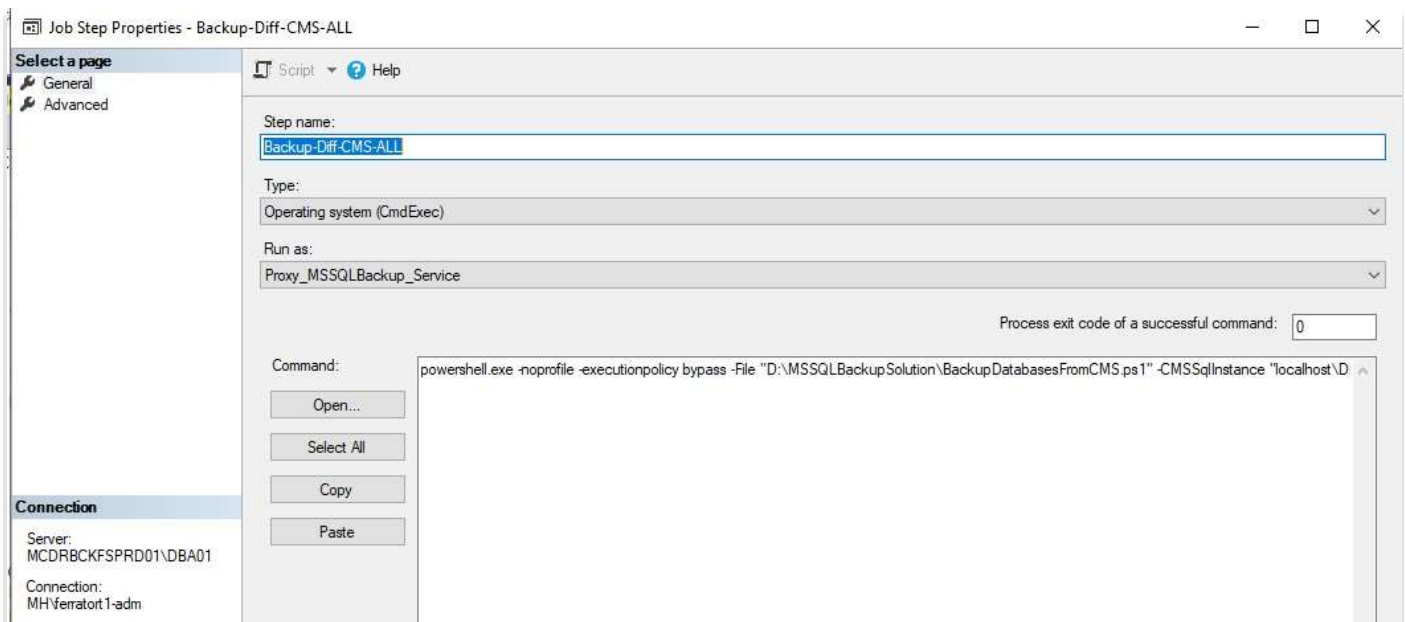
Le détail des étapes permet d'en savoir plus :

4.3.2 Job MSSQLBackupSolution-CMS-Backup-Diff-ALL

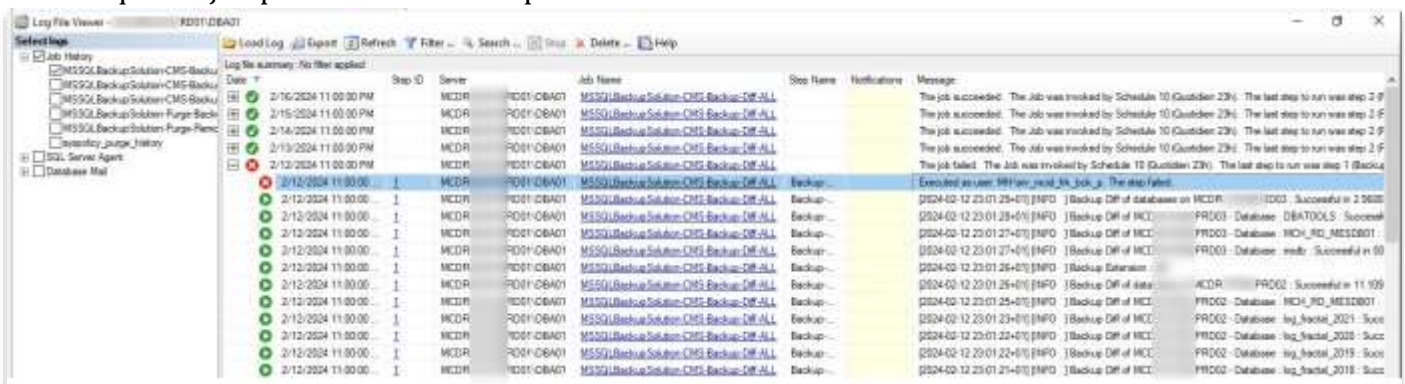


2 étapes : Backup, puis Robocopy.

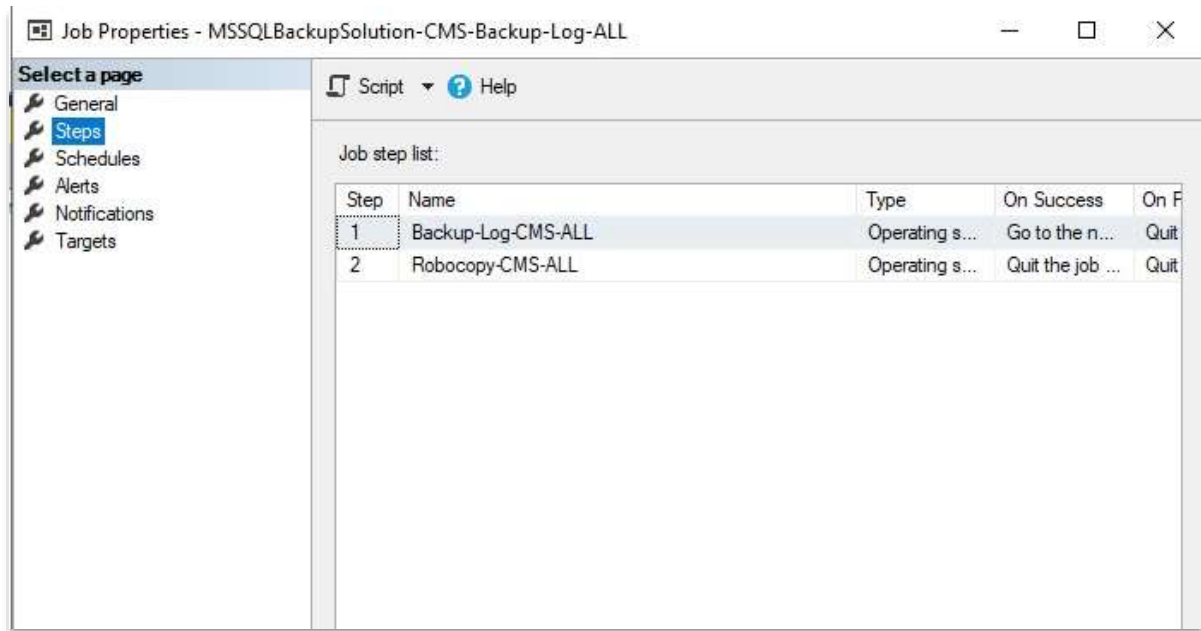
La seule difference est le schédule (quotidien) et le step de backup qui est un step de cmdExec avec une légère différence sur le type de backup (Diff et non Full)



L'historique du job permet d'en savoir plus :

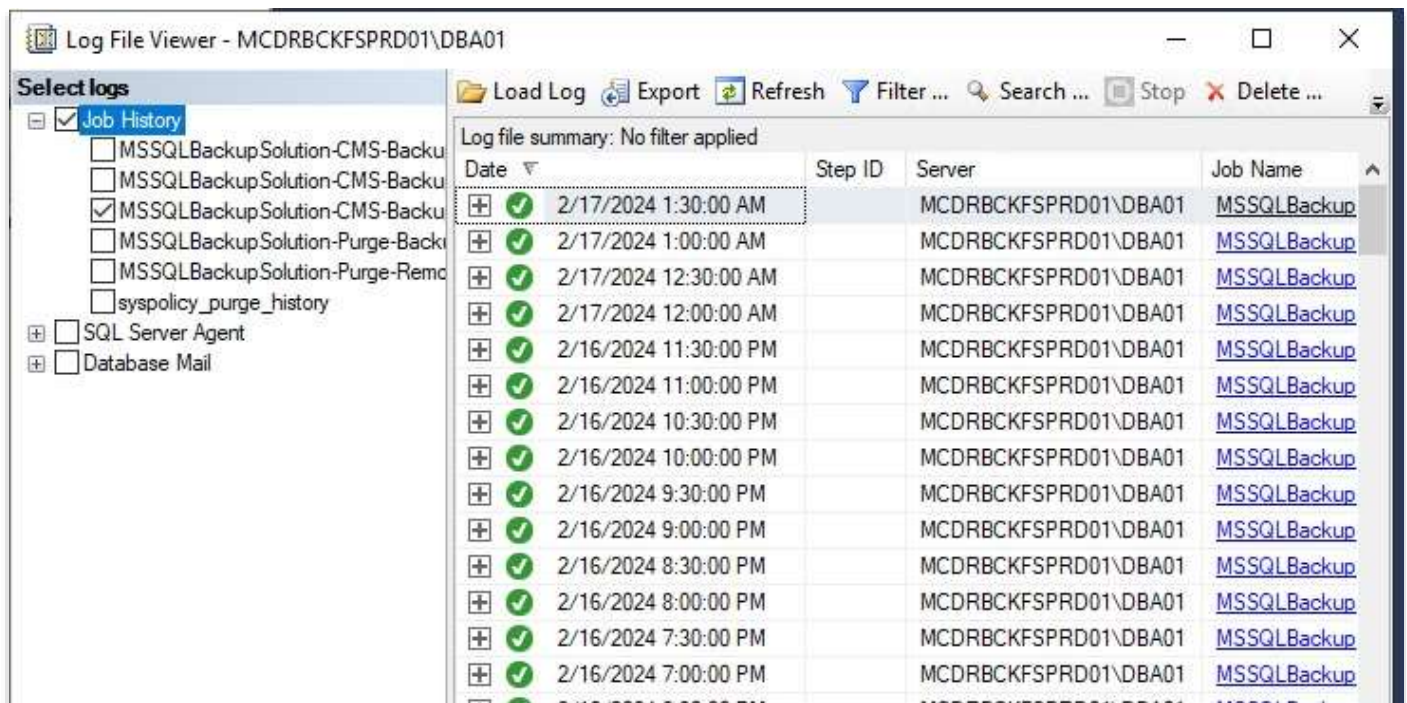


4.3.3 Job MSSQLBackupSolution-CMS-Backup-Log-ALL



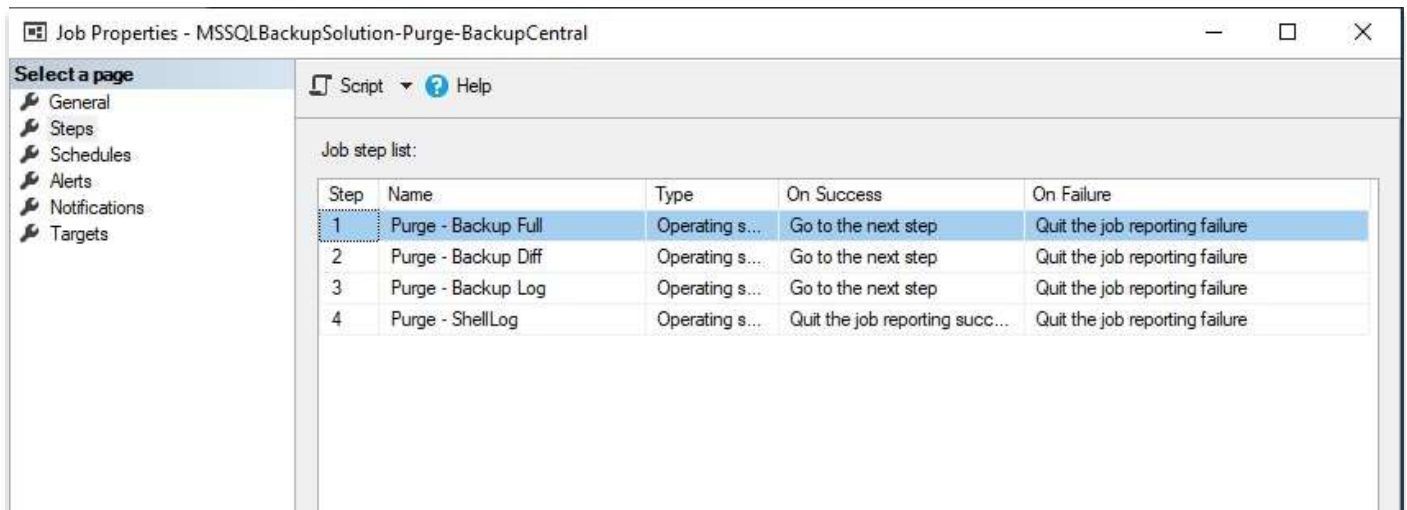
The backup log step of the job log use a backup Type “log” and is run every 30 minutes. It will backup logs data only for the databases with a recovery model to COMPLETE.

If no full backup exists for a database (because the database is brand new in the environment) then a full backup is proceeded first for this database.



4.3.4 Job MSSQLBackupSolution-Purge-BackupCentral

Le job de purge central est composé de 4 étapes chacune charger de nettoyer les différents types de backups ainsi que les fichiers de traces de l'outils lui-même

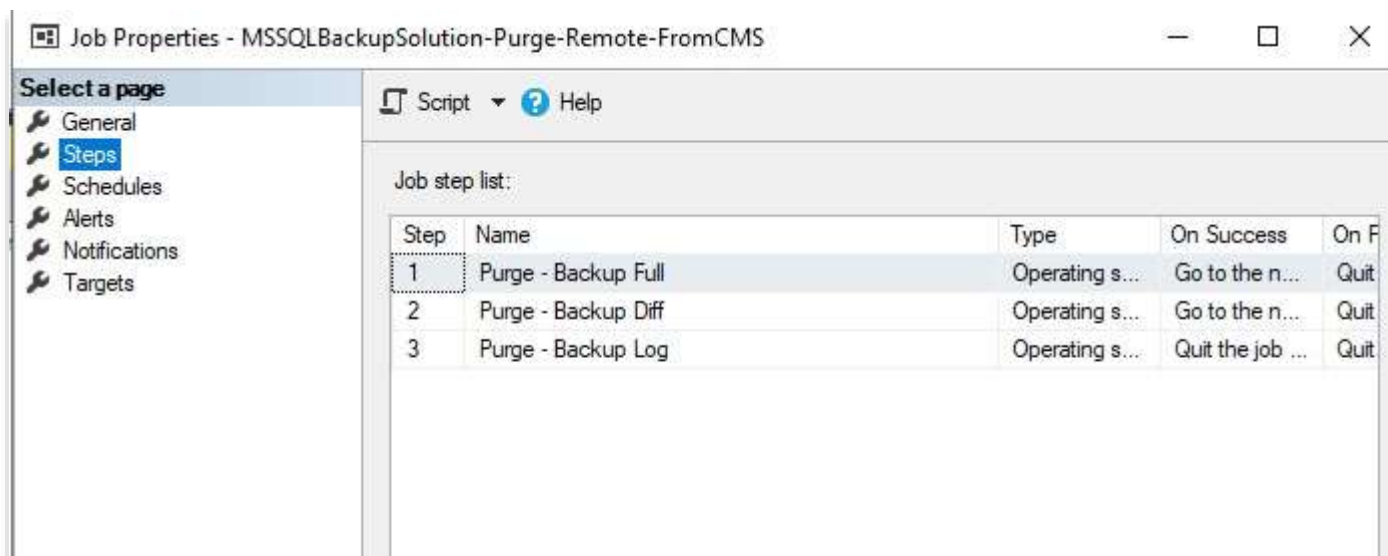


Exemple pour l'étape de purge des backups full

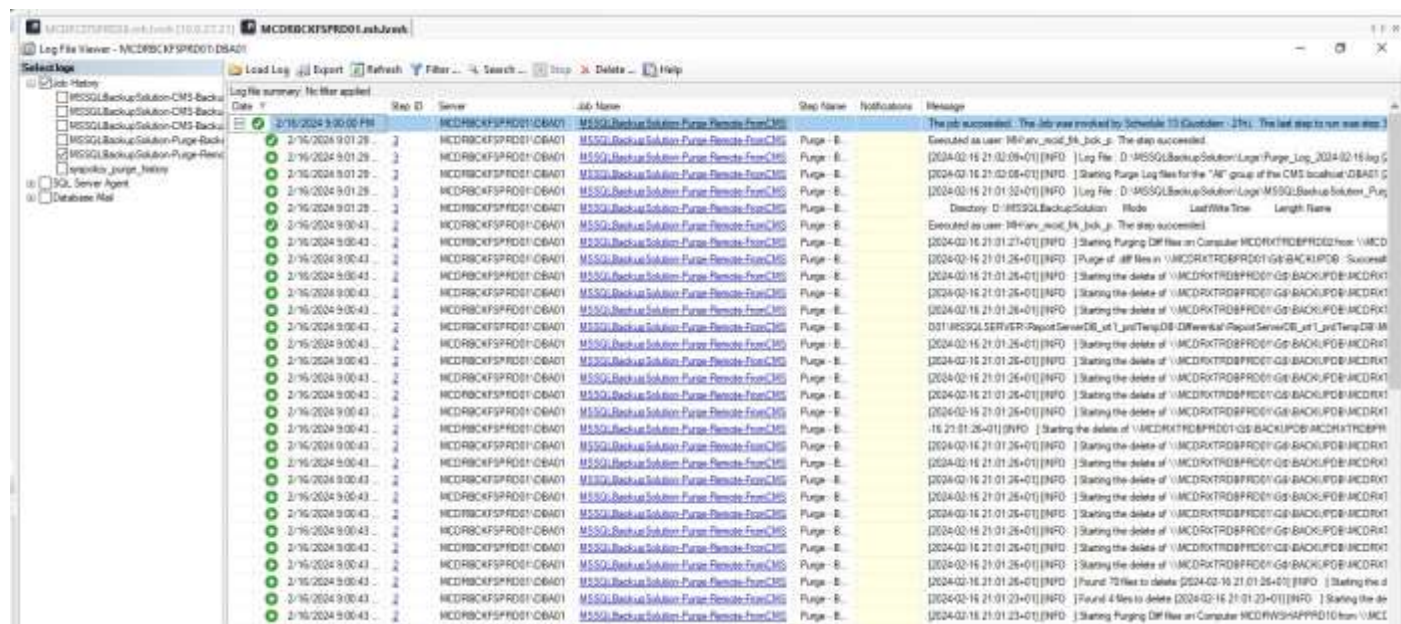
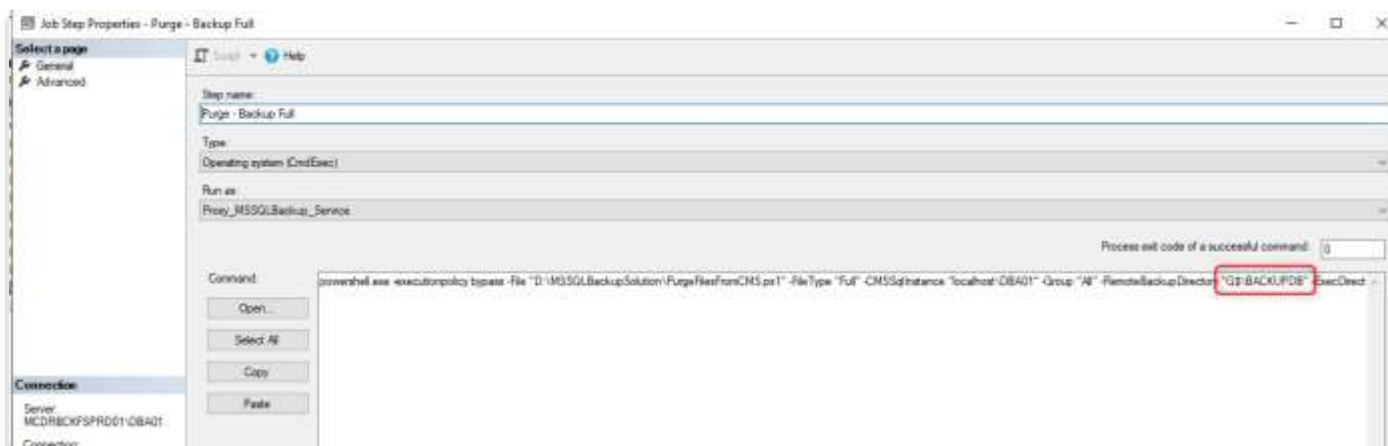


4.3.5 Job MSSQLBackupSolution-Purge-Remote-FromCMS

Ce job est en charge de lancer le nettoyage distant des fichiers, il est lancer quotidiennement



Exemple pour les backups full



5 La base de traces et monitoring

La base de trace est située sur l'instance installée sur le serveur central. Cette base contient une table permettant de tracer et de remonter les succès ou les échecs des étapes des jobs : backup, robocopy et purge.

Toutefois il est possible que d'autres problèmes viennent entraver le déroulement correct des traitements de sauvegarde.

Afin de s'assurer que les bases de données sont bien sécurisées correctement il faut mettre en place un monitoring des sauvegardes. Ce monitoring peut en effet chercher à détecter si l'ancienneté des sauvegardes (full, diff ou log) correspond bien aux exigences en se basant sur les données de la table de trace des sauvegardes centralisées.

Une autre technique, encore plus sûre serait d'utiliser la CMS et des scripts dédiés pour lister les dernières sauvegardes valides par type (full, diff, log) des toutes les databases à partir des tables systèmes MSSQL.

6 La mise à jour de la solution

La solution étant basé sur un ensemble de script powershell, la mise à jour peut se faire de deux façons :

- Si la plateforme est connecté à internet et que la politique de sécurité l'autorise il est possible de faire un git pull de la branche release pour mettre à jour les scripts.
- Si la plateforme n'est pas connecté à internet ou si la politique de sécurité interdit la connexion sur github.com, il est possible de récupérer la dernière release packagées sur le github (.zip) et de dézipper les fichiers dans un répertoire temporaire. Il est conseillé de faire un backup des scripts powershell sur la plateforme CMS avant de les écraser par les scripts de la nouvelle version.