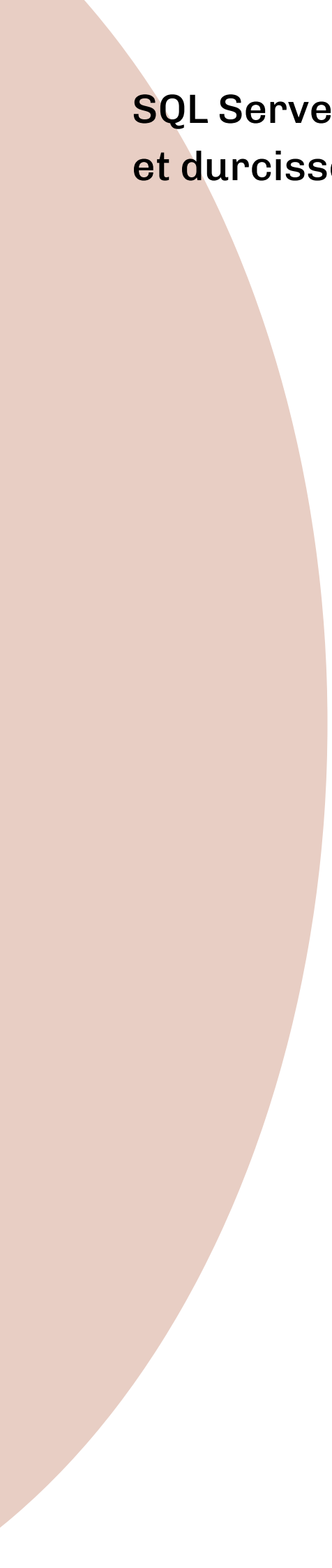




SQL Server 2022 - Conception, administration et durcissement

Table des matières

1. Installation de SQL Server 2022	6
1.1. Présentation SQL Server	6
1.2. Installation et configuration des services et connexions	7
1.3. Paramétrage de l'instance	8
1.4. Trace Flag.....	10
1.5. Bases systèmes.....	11
2. Concevoir la base de données	12
2.1. Stockage des données	12
2.2. Création d'une base.....	12
2.3. schema.....	13
3. Modélisation des données	14
3.1. Formes normales.....	14
3.2. Première forme normale	14
3.3. Seconde forme normale.....	15
3.4. Troisième forme normale.....	15
3.5. Utilisation d'un schéma normalisé	16
4. Types de données	17
4.1. Présentation des types de données	17
4.2. Champs XML.....	17
4.2.1. Utilisation des champs XML	17
4.2.2. Construction d'un schéma XML avec Visual Studio.....	18
4.2.3. Importation d'un schéma dans SQL Server.....	18
4.3. HierarchicalID.....	19
5. Tables graph	22
5.1. Présentation.....	22
5.2. Table de nœud	24
5.3. Table d'arrêt	24
5.4. Requêtes SQL.....	25
5.5. Cas d'utilisation	27
5.6. Limitations	27
6. Intégrité des données	28
6.1. Contraintes.....	28

6.2. Contraintes référentielles.....	28
6.3. Activation - désactivation de contraintes.....	28
7. Index	29
7.1. Présentation.....	29
7.2. Création d'index	31
7.3. Fragmentation d'index.....	33
7.4. Resumable index	34
8. Sauvegardes et restaurations	37
8.1. Sauvegardes et restaurations.....	37
8.2. Restauration à partir d'un cliché	38
8.3. Traitement des bases système	39
9. Sécurité et autorisations	40
9.1. Vue d'ensemble	40
9.2. Authentification sur l'instance	40
9.3. Comptes de connexion	40
9.4. Utilisateurs de base de données	41
9.5. Compte utilisateurs sans connexion.....	42
9.6. Permissions	42
9.7. Rôles.....	43
10. Taches d'administration	45
10.1. Tâches d'administration.....	45
10.2. Stratégies	45
10.3. Monitoring.....	46
10.4. Extended Event	46
11. Programmation T-SQL	48
11.1. Syntaxe de base	48
11.1.1. Directive de lot.....	48
11.1.2. Commentaires	48
11.1.3. Identificateurs.....	48
11.1.4. Variables.....	48
11.1.5. Opérateurs.....	49
11.1.6. Expressions.....	49
11.1.7. Bloc d'instruction.....	49
11.1.8. Tests.....	49
11.1.9. Boucle.....	51
11.1.10. Curseurs.....	51
11.1.11. Rupture de boucle.....	54
11.1.12. Gestion d'erreurs	56

11.2. Fonctions et procédures	58
11.2.1. Définition d'une fonction	58
11.2.2. Paramètres d'appel d'une fonction.....	59
11.2.3. Valeur de retour d'une fonction	60
11.2.4. Définition d'une procédure	61
11.2.5. EXECUTE.....	62
11.2.6. Paramètres d'appel d'une procédure	62
11.2.7. Valeur de retour d'une procédure	63
11.3. Les déclencheurs (Trigger).....	64
11.3.1. Les triggers	64
11.3.2. Trigger DML	64
11.3.3. Trigger DDL	65
12. Performances	66
12.1. Performances	66
12.2. Tables en mémoire.....	66
12.3. Query Store	67
12.3.1. Présentation Query Store	67
12.3.2. Mise en œuvre	68
12.4. Intelligent Query Processing (IQP)	70
12.4.1. Présentation.....	70
12.4.2. Parameter Sensitive Plan (PSP)	71
12.4.3. Batch Mode sur Rowstore.....	73
12.4.4. Adaptive Joins	73
12.4.5. Interleaved Execution for Multi-Statement Table-Valued Functions (MSTVF).....	73
12.4.6. Memory Feedback.....	74
12.4.7. Analyse de l'estimation automatique des cardinalités et des optimisations internes	74
12.4.8. Approximate Query Processing.....	74
12.4.9. Table Variable Deferred Compilation	75
12.4.10. Scalar UDF Inlining	75
13. Introduction durcissement	76
13.1. Confidentialité des données	76
13.2. Connexion TLS à une instance SQL Server.....	76
13.3. Chiffrement des données sur disque	76
13.4. Transparent Data Encryption	78
13.5. Ledger	78
14. Annexes	80
14.1. IFI (Instant File Initialization)	80
14.2. DBCC (Database Console Commands).....	80
14.3. Clichés instantanés	81
14.4. Partitionnement d'une table.....	82
14.5. Tables versionnées (ou tables temporelles)	83
14.6. Importation de données.....	85

14.7. Plan d'exécution	86
14.7.1. Traitement d'une requête.....	86
14.7.2. Plan d'exécution	87
14.7.3. Méthodes de parcours	88
14.7.4. Méthodes de jointure.....	89
14.8. Statistiques	90
14.9. Columnstore.....	97
14.10. Filestream et Filetable.....	98
14.10.1. Stockage des BLOB.....	98
14.10.2. Filestream	99
14.10.3. Filetable.....	100
Ressources annexes	105
Index	106
Contenus annexes	107

1. Installation de SQL Server 2022

1.1. Présentation SQL Server

Architecture générale de SQL Server

- moteur de base de données
instance qui peut gérer plusieurs bases de données
service Windows
- Agent SQL Server
permet d'effectuer des tâches d'administration
service Windows
- SQL Server Browser
tient à jour la liste des instances d'un serveur
service Windows
- SQL Server Integration Service
permet d'effectuer des opérations de transferts de données
service Windows

Éditions

- Express
- Standard
- Entreprise
- Développeur
- Web
- Compacte

Comparaison des versions : <https://learn.microsoft.com/fr-fr/sql/sql-server/editions-and-components-of-sql-server-2022?view=sql-server-ver16>

Fonctionnalités

- instance autonome
- instances autonomes avec réplication et haute disponibilité
- cluster pour haute disponibilité et éventuellement équilibrage de charge

Disponibilité

Windows (machine native ou machine virtuelle)

Cloud

Linux

Conteneurs Docker

1.2. Installation et configuration des services et connexions

SQL Server Installation Center

- Installation d'instances
- Installation des outils partagés
- Configuration de l'instance
 - emplacement des fichiers de données
 - comptes de connexion
 - démarrage des services

SQL Server Configuration Manager

console mmc de gestion des services, des connexions réseaux, ...

Version	Emplacement
SQL Server 2022	C:\Windows\SysWOW64\SQLServerManager16.msc
SQL Server 2019	C:\Windows\SysWOW64\SQLServerManager15.msc
SQL Server 2017	C:\Windows\SysWOW64\SQLServerManager14.msc
SQL Server 2016	C:\Windows\SysWOW64\SQLServerManager13.msc
SQL Server 2014 (12.x)	C:\Windows\SysWOW64\SQLServerManager12.msc
SQL Server 2012 (11.x)	C:\Windows\SysWOW64\SQLServerManager11.msc

SQL Server Management Studio

- gestion des objets d'une instance SQL Server
- écriture de scripts SQL (accès à des extraits de code et à des modules)
- outil à installer indépendamment

sqlcmd

Outil en ligne de commande pour exécuter des requêtes

```

1 1> use master;
2 2> select name from sys.databases;
3 3> go
4 Changed database context to 'master'.
5 name
6 -----
7 master
8 tempdb
9 model
10 msdb

```

powershell

utilisation de powershell et des applets spécifiques à SQL Server

installation du module sqlserver : `Install-Module -Name SqlServer -AllowClobber`

<https://www.powershellgallery.com/packages/SqlServer/22.3.0>

```

1 PS SQLSERVER:\SQL\DESKTOP-94LFLQR\DEFAULT\Databases\master\Views\dbo.spt_values> dir
2 Columns
3 ExtendedProperties
4 FullTextIndex
5 Indexes
6 ResumableIndexes
7 Statistics
8 Triggers

```

```

1 get-command -module SQLServer
2
3 CommandType      Name                                          Version  Source
4 -----
5 Alias            Decode-SqlName                             22.3.0   SqlServer
6 Alias            Encode-SqlName                             22.3.0   SqlServer
7 Function          Invoke-SqlNotebook                         22.3.0   SqlServer
8 Function          SQLSERVER:                                 22.3.0   SqlServer
9 Cmdlet            Add-RoleMember                             22.3.0   SqlServer
10 Cmdlet            Add-SqlAvailabilityDatabase                 22.3.0   SqlServer
11 Cmdlet            Add-SqlAvailabilityGroupListenerStaticIp    22.3.0   SqlServer
12 Cmdlet            Add-SqlAzureAuthenticationContext          22.3.0   SqlServer
13 Cmdlet            Add-SqlColumnEncryptionKeyValue            22.3.0   SqlServer
14 Cmdlet            Add-SqlLogin                               22.3.0   SqlServer
15 Cmdlet            Backup-ASDatabase                           22.3.0   SqlServer
16 Cmdlet            Backup-SqlDatabase                         22.3.0   SqlServer
17 Cmdlet            Complete-SqlColumnMasterKeyRotation        22.3.0   SqlServer
18 Cmdlet            ConvertFrom-EncodedSqlName                 22.3.0   SqlServer
19 Cmdlet            ConvertTo-EncodedSqlName                   22.3.0   SqlServer
20 Cmdlet            Convert-UrnToPath                           22.3.0   SqlServer
21 Cmdlet            Disable-SqlAlwaysOn                         22.3.0   SqlServer
22 Cmdlet            Enable-SqlAlwaysOn                         22.3.0   SqlServer
23 Cmdlet            Get-SqlAgent                               22.3.0   SqlServer

```

1.3. Paramétrage de l'instance

Paramétrages avec SSMS

Fenêtre propriétés de l'instance

- définition de comportements par défaut des bases des données
- définition du fonctionnement de l'instance

Propriétés du serveur - DESKTOP-94LFLQR

Sélectionner une page : Script Aide

- Général
- Mémoire
- Processeurs
- Sécurité
- Connexions
- Paramètres de base de données
- Avancé
- Autorisations

Connexion

Serveur : DESKTOP-94LFLQR

Connexion : DESKTOP-94LFLQR\admin

[Afficher les propriétés de connexion](#)

Progression

☒ L'opération de script s'est terminée

Taux de remplissage par défaut de l'index : 0

Sauvegarde et restauration

Spécifiez combien de temps SQL Server attendra une nouvelle bande.

☐ Attendre indéfiniment

☒ Essayer une fois

☐ Essayer pendant 0 minute(s)

Délai de rétention par défaut du support de sauvegarde (jours) : 0

☐ Compresser la sauvegarde ☐ Somme de contrôle de sauvegarde

Récupération

Intervalle de récupération (minutes) : 0

Emplacements de la base de données par défaut

Données : C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\

Journal : C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\

Sauvegarde : C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\

☒ Valeurs configurées ☐ Valeurs en cours d'exécution

OK Annuler

Propriétés du serveur - DESKTOP-94LFLQR

Sélectionner une page : Script Aide

- Général
- Mémoire
- Processeurs
- Sécurité
- Connexions
- Paramètres de base de données
- Avancé
- Autorisations

Connexion

Serveur : DESKTOP-94LFLQR

Connexion : DESKTOP-94LFLQR\admin

[Afficher les propriétés de connexion](#)

Autonomie	
Activer les bases de données autonomes	False
Divers	
Année de coupure à deux chiffres	2049
Autoriser les déclencheurs à activer d'autres	True
Langue de texte intégral par défaut	1033
Langue par défaut	English
Optimiser pour les charges de travail ad hoc	True
Option de mise à niveau du catalogue de texte	Reconstruire
Recherche des procédures de démarrage	False
Renforcer la priorité SQL Server	False
Seuil de rapports de processus bloqués	0
Seuil du curseur	-1
Taille de réplication de texte maximum	65536
Utiliser les fibres Windows (regroupement léger)	False
FILESTREAM	
Niveau d'accès FILESTREAM	Accès Transact-SQL activé
Nom de partage FILESTREAM	MSSQLSERVER
Parallélisme	
Attente de la requête	-1
Degré maximum de parallélisme	4
Seuil de coût pour le parallélisme	5
Verrous	0
Réseau	
Délai d'attente de la connexion distante	10
Taille du paquet réseau	4096

sp_configure

sys.sp_configure

RECONFIGURE WITH OVERRIDE

Références

sp_configure¹

RECONFIGURE²

1. <https://learn.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/sp-configure-transact-sql?view=sql-server-ver17>

```
1 EXEC sys.sp_configure N'max degree of parallelism', N'2'
2 GO
3 RECONFIGURE WITH OVERRIDE
4 GO

1 EXECUTE sp_configure 'show advanced options', '1';
2 EXEC sys.sp_configure;
```

ALTER SERVER

ALTER SERVER CONFIGURATION³

1.4. Trace Flag

Présentation

- Mécanisme permettant de modifier le comportement de SQL Server
- La modification peut être effective à trois niveaux : global, session ou requête
- Un Trace Flag peut être décrit dans la doc ou dans un KB.

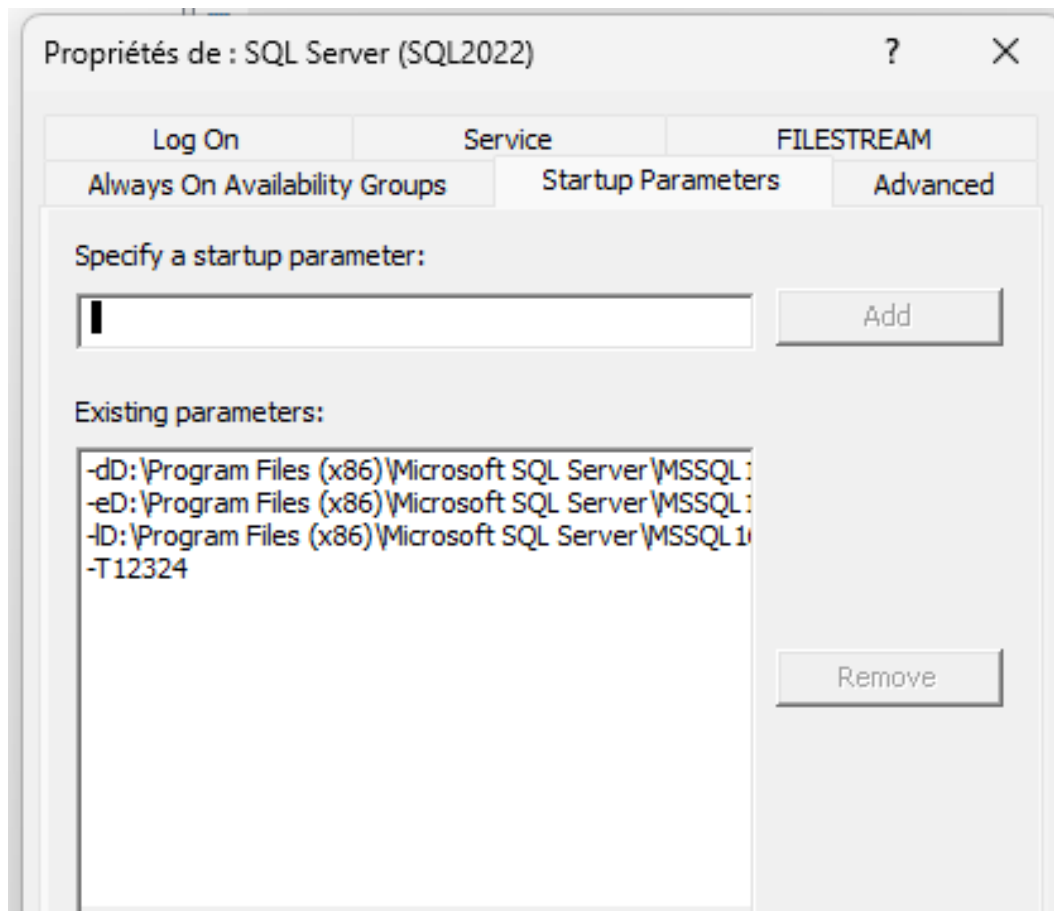
Référence

Mise en œuvre

- démarrage du serveur (portée globale)
- interactivement (portée globale ou session)
- dans une requête

Application au démarrage du serveur

ajout de l'option -T avec le numéro du Trace Flag



² <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/reconfigure-transact-sql?view=sql-server-ver17>

démarrage interactif

- commande DBCC TRACEON (xxxx, [-1]) : active le Trace Flag xxxx
paramètre optionnel -1 : active au niveau global
on peut activer plusieurs Trace Flag simultanément en spécifiant une liste de numéros séparés par des virgules
Référence⁴
- commande DBCC TRACEOFF (xxxx, [-1]) : désactive le Trace Flag xxxx
paramètre optionnel -1 : active au niveau global
on peut activer plusieurs Trace Flag simultanément en spécifiant une liste de numéros séparés par des virgules
Référence
- DBCC TRACESTATUS : informations sur les Trace Flag activés
on peut spécifier les numéros et l'option -1 si on veut des informations sur un Trace Flag spécifique
Référence

activation dans une requête

Utilisation du query hint QUERYTRACEON

Référence

1.5. Bases systèmes

master

- description de l'instance
- information sur les bases montées,
- toujours en mode de récupération simple
- à sauvegarder fréquemment

tempdb

- base pour les tables temporaires
- il n'est pas nécessaire de la sauvegarder

model

- utilisée lors de la création d'une nouvelle base
- une nouvelle base est une copie de la base model

msdb

- base utilisée par SQL Server Agent

³. <https://learn.microsoft.com/en-us/sql/t-sql/statements/alter-server-configuration-transact-sql?view=sql-server-ver17>

2. Concevoir la base de données

2.1. Stockage des données

Fichiers primaire et secondaires

- Fichier primaire : .mdf
- Fichiers secondaires : .ndf
- on peut spécifier l'emplacement que l'on veut pour le fichier .mdf et chacun des fichiers .ndf

Groupe de fichiers

- un groupe par défaut : PRIMARY
contient le fichier .mdf et autant de fichiers .ndf que l'on veut
- autant de groupes de fichiers que l'on souhaite
chaque groupe de fichier peut contenir autant de fichiers .ndf que l'on souhaite

Recommandations⁵

Fichiers journaux

Un ou plusieurs fichiers journaux de transaction : .ldf

journal de transaction :

1. Modifications écrites en mémoire
2. Modifications écrites dans le journal de transaction
3. Checkpoint écrit les données dans le fichier de données et enregistre une marque dans le journal de log

Taille et fragmentation des fichiers

⚠ Attention

- il n'y a pas de défragmentation, ni de réduction automatique de taille des fichiers de données.

C'est une opération à effectuer si nécessaire, s'il y a beaucoup d'espace perdu.

Cela peut être réalisé avec DBCC SHRINKFILE, ou DBCC SHRINKDATABASE

Commandes DBCC (cf. p.80)

- il n'y a pas besoin de défragmenter le fichier log (les écritures sont séquentielles).
suivant le mode de récupération, il faut tronquer ce fichier lors des opérations de sauvegarde

2.2. Création d'une base

Copie de la base model

une nouvelle base est une copie de la base model

On peut effectuer des modifications de la base model si on a plusieurs bases avec des structures identiques à créer

~~Penser à sauvegarder model si on y a fait des modifications~~

4. <https://learn.microsoft.com/en-us/sql/t-sql/database-console-commands/dbcc-traceon-transact-sql?view=sql-server-ver17>

Estimation de taille

- Taille de la base model
- Spécification de la taille initiale des fichiers, augmentation de taille
- Estimer les tailles des tables et des Index
- Journal de transaction

Emplacement de stockage

- Une table est créée dans un groupe de fichiers
- Intérêts :
 - Placement sur différents disques (performances) (attention : l'emplacement est défini au niveau du fichier)
 - Sauvegardes
 - Lecture seule
 - Thread en parallèle
- Fichiers logs peuvent être stockés dans n'importe quel répertoire

Création

- CREATE DATABASE
- utilisation SSMS

2.3. schema**Emplacement logique des tables : SCHEMA**

Espace de nom à l'intérieur d'une base

Nom complet d'un objet : serveur.base.schema.objet

CREATE SCHEMA

Résolution de nom

1. schéma par défaut de l'utilisateur
2. schéma dbo

Définition du schéma par défaut d'un utilisateur

ALTER USER WITH DEFAULT_SCHEMA = yyy

5. <https://learn.microsoft.com/en-us/sql/relational-databases/databases/database-files-and-filegroups?view=sql-server-ver16#recommendations>

3. Modélisation des données

3.1. Formes normales

Formes normales

Az Définition

Conception des tables visant à éviter les redondances de données stockées dans la base

Table des employés

Exemple

```
1 CREATE TABLE employe (  
2   id int(11) NOT NULL,  
3   nom varchar(30) NOT NULL DEFAULT '',  
4   ville varchar(30) NOT NULL DEFAULT '',  
5   savoir1 varchar(30) NOT NULL DEFAULT '',  
6   niv1 tinyint(4) NOT NULL,  
7   savoir2 varchar(30) NOT NULL DEFAULT '',  
8   niv2 tinyint(4) NOT NULL,  
9   PRIMARY KEY (id)  
10 ) ENGINE=InnoDB;
```

- Difficile de récupérer l'ensemble des domaines de compétence de la société : requête complexe à écrire
- Difficile d'ajouter de nouvelles compétences à un employé : il faut modifier la structure de la table et par conséquent certaines requêtes.
- Risques d'incohérence des données : les noms de compétences sont stockés directement sous forme de chaînes de caractères (SQL, sql, Sql, ...).

Gains après une mise en forme normale

Remarque

Stockage disque réduit

Diminution du risque d'erreur

Extension plus facile des possibilités d'enregistrement des informations

3.2. Première forme normale

Première forme normale

Ne pas avoir de répétition de groupe de colonnes

Suppression des colonnes en double

 Exemple

```
1 CREATE TABLE employe2 (
2   id int(11) NOT NULL,
3   nom varchar(30) NOT NULL DEFAULT '',
4   ville varchar(30) NOT NULL DEFAULT '',
5   savoir varchar(30) NOT NULL DEFAULT '',
6   niveau tinyint(4) NOT NULL
7 ) ENGINE=InnoDB;
```

On crée plusieurs enregistrements pour le même employé s'il a plusieurs compétences

3.3. Seconde forme normale

Seconde forme normale

Az Définition

Une table est en seconde forme normale si elle est en première forme normale et si tous les champs hors de la clé primaire dépendent de la clé primaire dans sa totalité.

```
1 CREATE TABLE employe (
2   id_emp int(11) NOT NULL,
3   nom varchar(30) NOT NULL DEFAULT '',
4   ville varchar(30) NOT NULL DEFAULT '',
5   PRIMARY KEY (id_emp)
6 ) ENGINE=InnoDB;
```

```
1 CREATE TABLE competence (
2   id_savoir int(11) NOT NULL,
3   savoir varchar(30) NOT NULL,
4   PRIMARY KEY (id_savoir)
5 ) ENGINE=InnoDB;
```

```
1 CREATE TABLE competence_employe (
2   id_savoir int(11) NOT NULL,
3   id_emp int(11) NOT NULL,
4   niveau tinyint(4) NOT NULL,
5   PRIMARY KEY (id_savoir,id_emp)
6 ) ENGINE=InnoDB;
```

3.4. Troisième forme normale

Troisième forme normale

Az Définition

Une table est en troisième forme normale si elle est en seconde forme normale et si tous les champs hors de la clé primaire dépendent uniquement de la clé primaire .

```
1 CREATE TABLE employe (
2   id_emp int(11) NOT NULL,
3   nom varchar(30) NOT NULL,
4   id_ville int(11) NOT NULL,
5   PRIMARY KEY (id_emp)
6 ) ENGINE=InnoDB;
```

```
1 CREATE TABLE ville (  
2   id_ville int(11) NOT NULL,  
3   ville varchar(30) NOT NULL,  
4   PRIMARY KEY (`id_ville`)  
5 ) ENGINE=InnoDB;
```

3.5. Utilisation d'un schéma normalisé

Ecriture de jointures

Il faut effectuer des jointures entre les tables

```
1 SELECT e.id_emp,e.nom,v.ville,c.savoir,ce.niveau  
2       FROM employe e INNER JOIN ville v USING(id_ville)  
3       INNER JOIN competence_employe ce USING(id_emp)  
4       INNER JOIN competence c USING(id_savoir);
```

Accès simple à des informations

sélection des compétences ou des villes

```
1 SELECT savoir FROM competence;
```

```
1 SELECT ville FROM ville;
```

4. Types de données

4.1. Présentation des types de données

Type de données

Une colonne a obligatoirement un type de données

Le type de données spécifie les valeurs qui peuvent être stockées dans la colonne et la façon dont le codage est effectué

Il y a un impact sur la taille de stockage de la table et des index, donc les performances d'entrée/sortie et l'utilisation du CPU dans les requêtes

Principaux types de données

Catégorie	Types
Valeurs numériques exactes	tinyint, smallint, int, bigint, bit, décimale, numérique, money, smallmoney
Valeurs numériques flottantes	float, real
Date et heure	date, time, datetime2, datetimeoffset, datetime, smalldatetime
Chaînes de caractères	char, varchar, text
Chaînes de caractères Unicode	nchar, nvarchar, ntext
Chaînes binaires	binary, varbinary, image
Types spécifiques	cursor, geography, geometry, hierarchyid, json, rowversion, sql_variant, table, uniqueidentifier, xml

Références

Références⁶

4.2. Champs XML

4.2.1. Utilisation des champs XML

Depuis SQL Server 2005, le type d'un champ peut être XML.

Il n'y a pas de contrôle effectué sur les données de ce champ lorsqu'un enregistrement est créé : il peut contenir n'importe quel texte.

Comment garantir que ce champ contient bien des données XML respectant une grammaire précise ?

C'est un problème courant quand on manipule des données XML et il existe deux solutions : les DTD et les schémas XML.

Dans SQL Server, on utilise les schémas XML.

⁶ <https://learn.microsoft.com/en-us/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver16>

4.2.2. Construction d'un schéma XML avec Visual Studio

Visual Studio permet de construire un schéma à partir d'un fichier XML.

Il faut utiliser un fichier XML le plus détaillé possible, présentant les différentes constructions des éléments et des attributs afin que le schéma XML soit le plus exhaustif.

Exemple de fichier XML :

```
1 <personnes>
2 <personne>
3 <nom>Dupond</nom>
4 <naissance>1950-01-01</naissance>
5 </personne>
6 </personnes>
```

Une fois le schéma XML obtenu, il sera normalement nécessaire de le compléter et de l'améliorer pour qu'il décrive complètement la grammaire souhaitée.

Exemple de schéma XML :

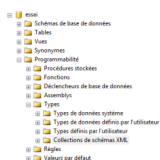
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
3 <xs:element name="personnes">
4 <xs:complexType>
5 <xs:sequence>
6 <xs:element name="personne">
7 <xs:complexType>
8 <xs:sequence>
9 <xs:element name="nom" type="xs:string" />
10 <xs:element name="naissance" type="xs:date" />
11 </xs:sequence>
12 </xs:complexType>
13 </xs:element>
14 </xs:sequence>
15 </xs:complexType>
16 </xs:element>
17 </xs:schema>
```

4.2.3. Importation d'un schéma dans SQL Server

Collection de schémas

Les schémas XML doivent être créés dans SQL Serveur à l'aide d'un type d'objets particuliers de la base : les collections de schémas.

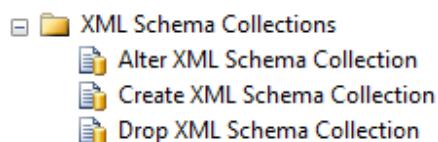
La liste des collections de schémas d'une base est accessible dans l'arborescence Programmabilité / Types / Collections de schémas XML :



Depuis cette liste, il est possible d'éditer ou de supprimer des schémas, mais pas d'en créer. Il faut utiliser une instruction SQL CREATE SCHEMA COLLECTION.

```
1 USE [essai]
2 GO
3 CREATE XML SCHEMA COLLECTION [dbo].[personne] AS N'<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"><xs:element
  name="personnes"><xs:complexType><xs:complexContent><xs:restriction base="xsd:anyType">
  <xs:sequence><xs:element name="personne"><xs:complexType><xs:complexContent>
  <xs:restriction base="xsd:anyType"><xs:sequence><xs:element name="nom" type="xsd:string"
  /><xs:element name="naissance" type="xsd:date" /></xs:sequence></xs:restriction>
  </xs:complexContent></xs:complexType></xs:element></xs:sequence></xs:restriction>
  </xsd:complexContent></xs:complexType></xs:element></xsd:schema>'
4 GO
```

Une solution simple consiste à utiliser un modèle de procédure :



(cf. Voir la création d'une collection de schéma)

4.3. HierarchicalID

HierarchicalID

Az Définition

Type de données permettant de représenter des structures hiérarchiques, avec un stockage compact et des fonctions de traitement

Catégories et sous catégories

Exemple

On veut gérer des catégories de produits.

ID	Nom	Parent_ID
1	Électronique	NULL
2	Téléphones	1
3	Smartphones	2
4	Accessoires	1
5	Chargeurs	4
6	Mode	NULL
7	Vêtements	6
8	Chaussures	6
9	Baskets	8
10	Sandales	8

```

1 DROP TABLE Categories;
2
3 CREATE TABLE Categories (
4     ID INT PRIMARY KEY,
5     Name NVARCHAR(100),
6     Hierarchy hierarchyid,
7     HierarchyLevel AS Hierarchy.GetLevel()
8 );
9
10 INSERT INTO Categories (ID, Name, Hierarchy)
11 VALUES (1, 'Électronique', hierarchyid::Parse('/1/')),
12         (2, 'Téléphones', hierarchyid::Parse('/1/1/')),
13         (3, 'Smartphones', hierarchyid::Parse('/1/1/2/')),
14         (4, 'Accessoires', hierarchyid::Parse('/1/2/')),
15         (5, 'Chargeurs', hierarchyid::Parse('/1/2/1/')),
16         (6, 'Mode', hierarchyid::Parse('/2/'));

```

```

17 INSERT INTO Categories (ID, Name, Hierarchy)
18 VALUES
19     (7, 'Vêtements', (select Hierarchy.GetDescendant(NULL,NULL) from Categories where
Name='Mode')),
20     (8, 'Chaussures', (select Hierarchy.GetDescendant(NULL,NULL) from Categories where
Name='Mode'));
21 INSERT INTO Categories (ID, Name, Hierarchy)
22 VALUES
23     (9, 'Baskets', (select Hierarchy.GetDescendant(NULL,NULL) from Categories where
ID=8)),
24     (10, 'Sandales', (select Hierarchy.GetDescendant(NULL,NULL) from Categories where
ID=8));
1 -- affichage de toute la table
2 SELECT ID, Name, Hierarchy.ToString() AS Path, HierarchyLevel
3 FROM Categories
4 ORDER BY Hierarchy;
5
6 -- affichage des sous catégories de mode
7 SELECT ID, Name, Hierarchy.ToString() AS Path, HierarchyLevel
8 FROM Categories
9 WHERE Hierarchy.IsDescendantOf((select Hierarchy from Categories where Name='Mode')) =
1;
10
11 -- affichage des sous catégories de chaussures
12 SELECT ID, Name, Hierarchy.ToString() AS Path, HierarchyLevel
13 FROM Categories
14 WHERE Hierarchy.IsDescendantOf(hierarchyid::Parse('/2/1/')) = 1;

```

Fonctions de traitement

Fonctions disponibles

Fonction	Description
GetRoot()	Identification du nœud racine de l'arborescence.
GetLevel()	Profondeur d'un nœud dans l'arborescence.
GetAncestor(niveau)	Identification du supérieur hiérarchique d'un nœud. niveau = nombre de niveaux à remonter.
GetDescendant([noeudFrere1[, noeudFrere2]])	Identification du descendant direct. S'il y a plusieurs nœud fils ,indiquer 1 ou 2 frères permet de localiser précisément l'emplacement du descendant. (utile lors de l'insertion de données dans un arbre trié)
IsDescendant(noeudATester)	Test si noeudATester est un descendant.
Parse(chaineCaractères)	Convertit une chaîne de caractères en type hierarchyId.
ToString()	Fournit une valeur textuelle de hierarchyId (inverse de Parse)
Reparent(ancienneRacine, nouvelleRacine)	Modifie le nœud racine.
Fonctions utilisables uniquement dans le CLR	
Read(lecteurBinaire)	Lecture en binaire du nœud
Write(fluxBinaire)	Ecriture de la valeur de type hierarchyId du nœud.

Références

Références⁷

⁷. <https://learn.microsoft.com/en-us/sql/relational-databases/hierarchical-data-sql-server?view=sql-server-ver16>

5. Tables graph

5.1. Présentation

Cas utilisation - Travaux scientifiques

[👁 Exemple](#)

On veut suivre des scientifiques, avec leur page Wikipedia, leur domaine d'intervention et leur sujet de recherche

```
1 -- tables relationnelles standard
2 -- utilisation de clés étrangères
3 -- et écriture d'une jointure
4 --
5 CREATE TABLE personne (
6     id_personne INT PRIMARY KEY,
7     nom VARCHAR(50),
8     prenom VARCHAR(50),
9     wikipedia VARCHAR(256)
10 );
11
12 CREATE TABLE domaine (
13     id_domaine INT PRIMARY KEY,
14     domaine VARCHAR(50)
15 );
16
17 CREATE TABLE sujet(
18     id_personne INT,
19     id_domaine INT,
20     sujet VARCHAR(256),
21     FOREIGN KEY (id_personne) REFERENCES personne(id_personne),
22     FOREIGN KEY (id_domaine) REFERENCES domaine(id_domaine)
23 );
24
25 INSERT INTO personne VALUES
26 (1, 'Curie', 'Marie', 'https://fr.wikipedia.org/wiki/Marie_Curie'),
27 (2, 'Franklin', 'Rosalind', 'https://fr.wikipedia.org/wiki/Rosalind_Franklin'),
28 (3, 'Lovelace', 'Ada', 'https://fr.wikipedia.org/wiki/Ada_Lovelace'),
29 (4, 'Leavitt', 'Henrietta', 'https://fr.wikipedia.org/wiki/Henrietta_Swan_Leavitt'),
30 (5, 'Meitner', 'Lise', 'https://fr.wikipedia.org/wiki/Lise_Meitner');
31
32 INSERT INTO domaine VALUES
33 (1, 'Physique'),
34 (2, 'Informatique'),
35 (3, 'Mathématiques'),
36 (4, 'Astronomie'),
37 (5, 'Biologie');
38
39 INSERT INTO sujet (id_personne, id_domaine, sujet) VALUES
40 (1, 1, 'Découverte de la radioactivité'),      -- Marie Curie - Physique
41 (2, 5, 'Découverte de la structure de l'ADN'),  -- Rosalind Franklin -
Biologie
42 (3, 2, 'Premier programme informatique'), -- Ada Lovelace - Informatique
43 (4, 4, 'Découverte de la relation période-luminosité des céphéides'),      --
Henrietta Leavitt - Astronomie
44 (5, 1, 'Découverte de la fission nucléaire');  -- Lise Meitner - Physique
45
46 SELECT p.nom, p.prenom, d.domaine
47 FROM personne p
48 JOIN sujet s ON p.id_personne = s.id_personne
```

```

49 JOIN domaine d ON s.id_domaine = d.id_domaine
50 WHERE d.domaine = 'Physique';
51
52
1 -- utilisation de tables graph
2 --
3 --
4 CREATE TABLE personne (
5     id_personne INT PRIMARY KEY,
6     nom VARCHAR(50),
7     prenom VARCHAR(50),
8     wikipedia VARCHAR(256)
9 ) AS NODE;
10
11 CREATE TABLE domaine (
12     id_domaine INT PRIMARY KEY,
13     domaine VARCHAR(50)
14 ) AS NODE;
15
16 CREATE TABLE sujet(
17     sujet VARCHAR(256)
18 ) AS EDGE;
19
20 INSERT INTO personne VALUES
21 (1, 'Curie', 'Marie', 'https://fr.wikipedia.org/wiki/Marie_Curie'),
22 (2, 'Franklin', 'Rosalind', 'https://fr.wikipedia.org/wiki/Rosalind_Franklin'),
23 (3, 'Lovelace', 'Ada', 'https://fr.wikipedia.org/wiki/Ada_Lovelace'),
24 (4, 'Leavitt', 'Henrietta', 'https://fr.wikipedia.org/wiki/Henrietta_Swan_Leavitt'),
25 (5, 'Meitner', 'Lise', 'https://fr.wikipedia.org/wiki/Lise_Meitner');
26
27 INSERT INTO domaine VALUES
28 (1, 'Physique'),
29 (2, 'Informatique'),
30 (3, 'Mathématiques'),
31 (4, 'Astronomie'),
32 (5, 'Biologie');
33
34 INSERT INTO sujet ( $from_id, $to_id, sujet ) VALUES
35 ((SELECT $node_id FROM personne WHERE id_personne = 1), (SELECT $node_id FROM domaine
    WHERE id_domaine = 1), 'Découverte de la radioactivité'),      -- Marie Curie -
    Physique
36 ((SELECT $node_id FROM personne WHERE id_personne = 2), (SELECT $node_id FROM domaine
    WHERE id_domaine = 5), 'Découverte de la structure de l'ADN'),      -- Rosalind
    Franklin - Biologie
37 ((SELECT $node_id FROM personne WHERE id_personne = 3), (SELECT $node_id FROM domaine
    WHERE id_domaine = 2), 'Premier programme informatique'), -- Ada Lovelace - Informatique
38 ((SELECT $node_id FROM personne WHERE id_personne = 4), (SELECT $node_id FROM domaine
    WHERE id_domaine = 4), 'Découverte de la relation période-luminosité des céphéides'),
    -- Henrietta Leavitt - Astronomie
39 ((SELECT $node_id FROM personne WHERE id_personne = 5), (SELECT $node_id FROM domaine
    WHERE id_domaine = 1), 'Découverte de la fission nucléaire');      -- Lise Meitner -
    Physique
40
41
42 SELECT p.nom, p.prenom, d.domaine
43 FROM personne p, sujet s ,domaine d
44 where MATCH(p-(s)->d)
45 and d.domaine = 'Physique'

```

Éléments majeurs

table de nœud

table d'arrête

instruction MATCH

Références⁸

Exemple graphDemo⁹

5.2. Table de nœud

Table de nœud

Az Définition

Définition de colonnes comme dans une table classique

Ajout d'une colonne \$node_id calculée automatiquement, que l'on ne peut pas modifier.

Création

Utilisation de l'option AS NODE dans l'instruction CREATE TABLE

Gestion

On peut utiliser ALTER et DROP, créer des index, ... comme pour une table classique

5.3. Table d'arrête

Table d'arrête

Az Définition

Définition de colonnes comme dans une table classique (éventuellement aucune)

Ajout de trois colonnes \$edge_id, \$from_id, \$to_id. \$edge_id est calculée automatiquement, comme \$node_id d'une table de nœud

Création

Utilisation de l'option AS EDGE dans l'instruction CREATE TABLE

Contraintes

On peut définir des contraintes spécifiant quel nœud peut être lié à quel autre nœud.

Références

```
1 CONSTRAINT constraint_name CONNECTION (clause1[, clause2...])
2 --- clause est de la forme noeud1 to noeud2

1 -- CREATE node and edge tables
2 CREATE TABLE Customer
3 (
4     ID INTEGER PRIMARY KEY
5     ,CustomerName VARCHAR(100)
6 )
7 AS NODE;
8 GO
9 CREATE TABLE Product
10 (
11     ID INTEGER PRIMARY KEY
```

⁸ <https://learn.microsoft.com/en-us/sql/relational-databases/graphs/sql-graph-architecture?view=sql-server-ver16>

```

12     ,ProductName VARCHAR(100)
13 )
14 AS NODE;
15 GO
16 CREATE TABLE bought
17 (
18     PurchaseCount INT
19     ,CONSTRAINT EC_BOUGHT CONNECTION (Customer TO Product) ON DELETE CASCADE
20 )
21 AS EDGE;

```

Attention

CONSTRAINT EC_BOUGHT CONNECTION (Supplier TO Product, Customer TO Product)

n'est pas équivalent à :

CONSTRAINT EC_BOUGHT1 CONNECTION (Supplier TO Product)

CONSTRAINT EC_BOUGHT2 CONNECTION (Customer TO Product)

La contrainte composée EC_BOUGHT est un OU : on peut insérer une relation d'un Supplier vers un Product ou une relation d'un Customer vers un Product

Les deux contraintes EC_BOUGHT1 et EC_BOUGHT2 imposent que le relation soit d'un Supplier vers un Product et d'un Customer vers un Product : on ne peut pas créer de relation dans la table !

Gestion

On peut utiliser ALTER et DROP, créer des index, ... comme pour une table classique.

Références

5.4. Requêtes SQL

Requêtes standards

On peut effectuer des requêtes standard sur les champs des tables nœud ou arrête

MATCH

L'opérateur MATCH permet d'indiquer une correspondance : noeud1-(arrete)->noeud2

```

1 -- Create person node table
2 CREATE TABLE dbo.Person (ID INTEGER PRIMARY KEY, name VARCHAR(50)) AS NODE;
3 CREATE TABLE dbo.friend (start_date DATE) AS EDGE;
4
5 -- Insert into node table
6 INSERT INTO dbo.Person VALUES (1, 'Alice');
7 INSERT INTO dbo.Person VALUES (2, 'John');
8 INSERT INTO dbo.Person VALUES (3, 'Jacob');
9
10 -- Insert into edge table
11 INSERT INTO dbo.friend VALUES ((SELECT $node_id FROM dbo.Person WHERE name = 'Alice'),
12     (SELECT $node_id FROM dbo.Person WHERE name = 'John'), '9/15/2011');
13
14 INSERT INTO dbo.friend VALUES ((SELECT $node_id FROM dbo.Person WHERE name = 'Alice'),
15     (SELECT $node_id FROM dbo.Person WHERE name = 'Jacob'), '10/15/2011');
16
17 INSERT INTO dbo.friend VALUES ((SELECT $node_id FROM dbo.Person WHERE name = 'John'),
18     (SELECT $node_id FROM dbo.Person WHERE name = 'Jacob'), '10/15/2012');
19
20 -- use MATCH in SELECT to find friends of Alice
21 SELECT Person2.name AS FriendName
22 FROM Person Person1, friend, Person Person2
23 WHERE MATCH(Person1-(friend)->Person2)
24 AND Person1.name = 'Alice';

```

9. <https://learn.microsoft.com/fr-fr/sql/relational-databases/graphs/sql-graph-sample?view=sql-server-ver16>

```

25
26 -- friend of friend of Alice
27 SELECT Person3.name AS FriendName
28 FROM Person Person1, friend, Person Person2, friend friend2, Person Person3
29 WHERE MATCH(Person1-(friend)->Person2-(friend2)->Person3)
30 AND Person1.name = 'Alice';
31
32 -- Find 2 people who are both friends with same person
33 SELECT Person1.name AS Friend1, Person2.name AS Friend2
34 FROM Person Person1, friend friend1, Person Person2,
35      friend friend2, Person Person0
36 WHERE MATCH(Person1-(friend1)->Person0<-(friend2)-Person2);
37
38 -- this pattern can also be expressed as below
39
40 SELECT Person1.name AS Friend1, Person2.name AS Friend2
41 FROM Person Person1, friend friend1, Person Person2,
42      friend friend2, Person Person0
43 WHERE MATCH(Person1-(friend1)->Person0 AND Person2-(friend2)->Person0);

```

SHORTEST-PATH

SHORTEST-PATH : Permet de trouver le chemin le plus court entre deux nœuds

FOR GRAPH : permet d'indiquer qu'on effectue une agrégation

Références¹⁰

```

1 -- base graphDemo
2 --
3 SELECT PersonName, Friends
4 FROM (
5     SELECT
6         Person1.name AS PersonName,
7         STRING_AGG(Person2.name, '->') WITHIN GROUP (GRAPH PATH) AS Friends,
8         LAST_VALUE(Person2.name) WITHIN GROUP (GRAPH PATH) AS LastNode
9     FROM
10        Person AS Person1,
11        friendOf FOR PATH AS fo,
12        Person FOR PATH AS Person2
13     WHERE MATCH(SHORTEST_PATH(Person1(-(fo)->Person2)+))
14     AND Person1.name = 'Jacob'
15 ) AS Q
16 WHERE Q.LastNode = 'Alice'

```

Fonctions graph

Références¹⁶

Fonction	Description
EDGE_ID_FROM_PARTS ¹¹	Construire un edge_id à partir de object_id et graph_id
GRAPH_ID_FROM_EDGE_ID ¹²	Extraire le graph_id d'un edge_id
GRAPH_ID_FROM_NODE_ID ¹³	Extraire le graph_id d'un node_id
NODE_ID_FROM_PARTS ¹⁴	Construire un node_id à partir d'un object_id et d'un graph_id
OBJECT_ID_FROM_EDGE_ID ¹⁵	Extraire le object_id d'un edge_id
OBJECT_ID_FROM_NODE_ID ¹⁷	Extraire le object_id d'un node_id

¹⁰ <https://learn.microsoft.com/en-us/sql/relational-databases/graphs/sql-graph-shortest-path?view=sql-server-ver15>

5.5. Cas d'utilisation

Quand utiliser des tables graph ?

- Données interconnectées et hiérarchiques avec des relations de plusieurs à plusieurs.
- HierarchylD n'est pas suffisante.
- Besoin de parcourir ou d'analyser des relations.
- Besoin de plus de relations.

5.6. Limitations

- tables temporaires, variables de type table, tables temporelles versionnées par le système et les tables optimisées pour la mémoire ne peuvent pas être des tables de nœuds ou d'arêtes.
- Support limité dans SSMS : pas de design, pas de création d'index pour \$node_id, \$from_id et \$to_id
- pas de mise à jour avec UPDATE de \$from_id et \$to_id.
Il faut insérer un nouvel enregistrement et supprimer l'ancien.
- Pas de requêtes interbases de données sur les objets de graphe.
- Pas de changement de table de nœuds en table d'arêtes et vice versa.

¹¹. <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/edge-id-from-parts-transact-sql?view=sql-server-ver16>

6. Intégrité des données

6.1. Contraintes

Types de contraintes

- unicité des valeurs d'une ou plusieurs colonnes : index unique
- valeurs valides dans une colonne : contrainte CHECK, NOT NULL
- valeurs définies dans une table : contrainte de clé étrangère
- traitement complexe : trigger

6.2. Contraintes référentielles

CONSTRAINT nomContrainte FOREIGN KEY (colonneTable) REFERENCES

AutreTable (colonneAutreTable)

```
1 ALTER TABLE [dbo].[clients] WITH CHECK ADD CONSTRAINT [FK_clients_villes] FOREIGN  
  KEY([id_ville])  
2 REFERENCES [dbo].[villes] ([id])  
3 ON DELETE CASCADE  
4 GO
```

Action	Description
NO ACTION	Génère une erreur indiquant que l'opération ne peut pas être effectuée. C'est la valeur par défaut.
CASCADE	Supprime ou modifie les enregistrements liés à la valeur supprimée ou modifiée.
SET NULL	La colonne prend la valeur NULL
SET DEFAULT	La colonne prend sa valeur par défaut.

6.3. Activation - désactivation de contraintes

Désactivation

- peut être nécessaire pour des importations massives de données
- désactivation de toutes les contraintes

```
ALTER TABLE NomDeLaTable NOCHECK CONSTRAINT ALL;
```

- désactivation d'une contrainte spécifique

```
ALTER TABLE NomDeLaTable NOCHECK CONSTRAINT NomDeLaContrainte;
```

Activation

- ALTER TABLE NomDeLaTable [WITH CHECK] CHECK CONSTRAINT NomDeLaContrainte;
- ALTER TABLE NomDeLaTable [WITH CHECK] CHECK CONSTRAINT ALL;

WITH CHECK : active la vérification des lignes existantes

¹² <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/graph-id-from-edge-id-transact-sql?view=sql-server-ver16>

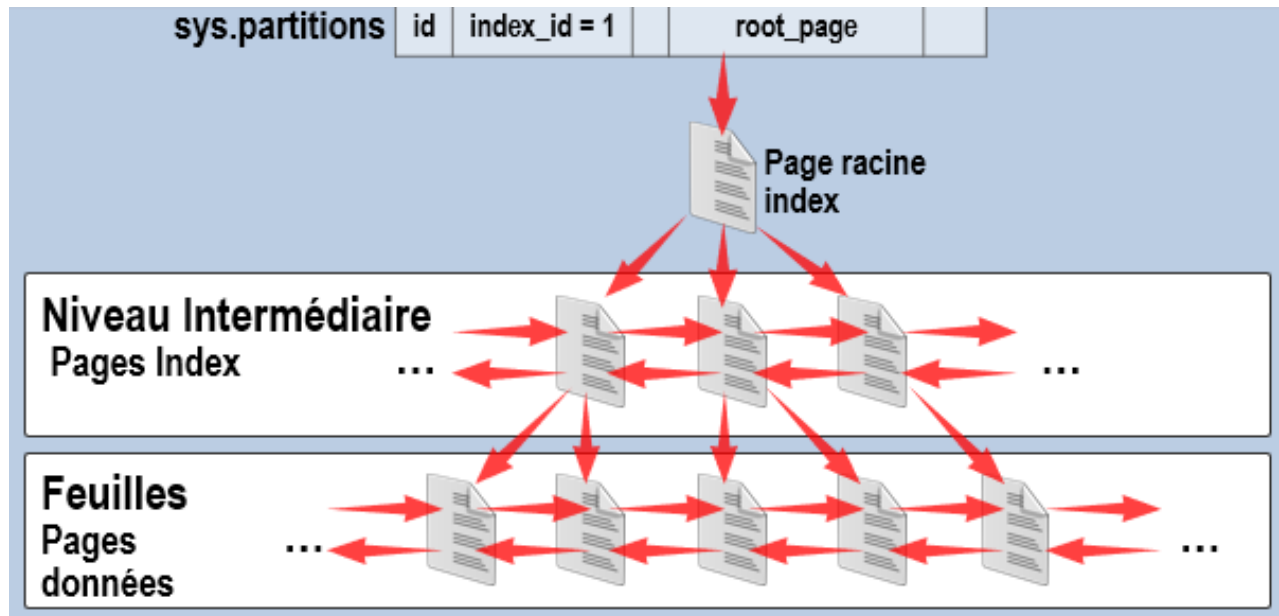
7. Index

7.1. Présentation

Index Cluster

Un index cluster par table

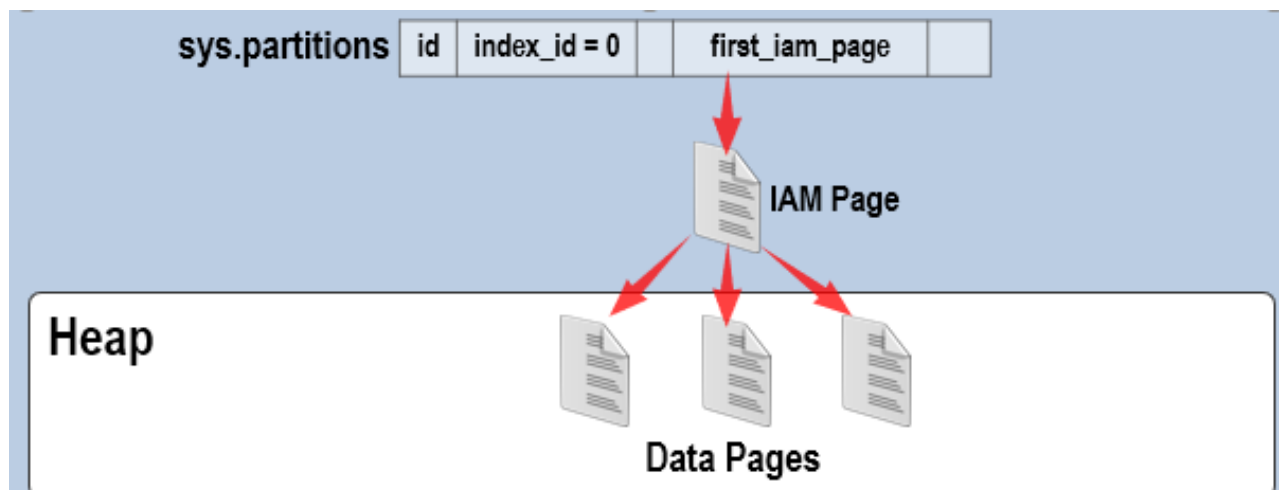
Arbre B-tree stocke les pages de données dans l'ordre de la clé d'index



Heap

Une table sans index cluster

Pages stockées sans ordre particulier

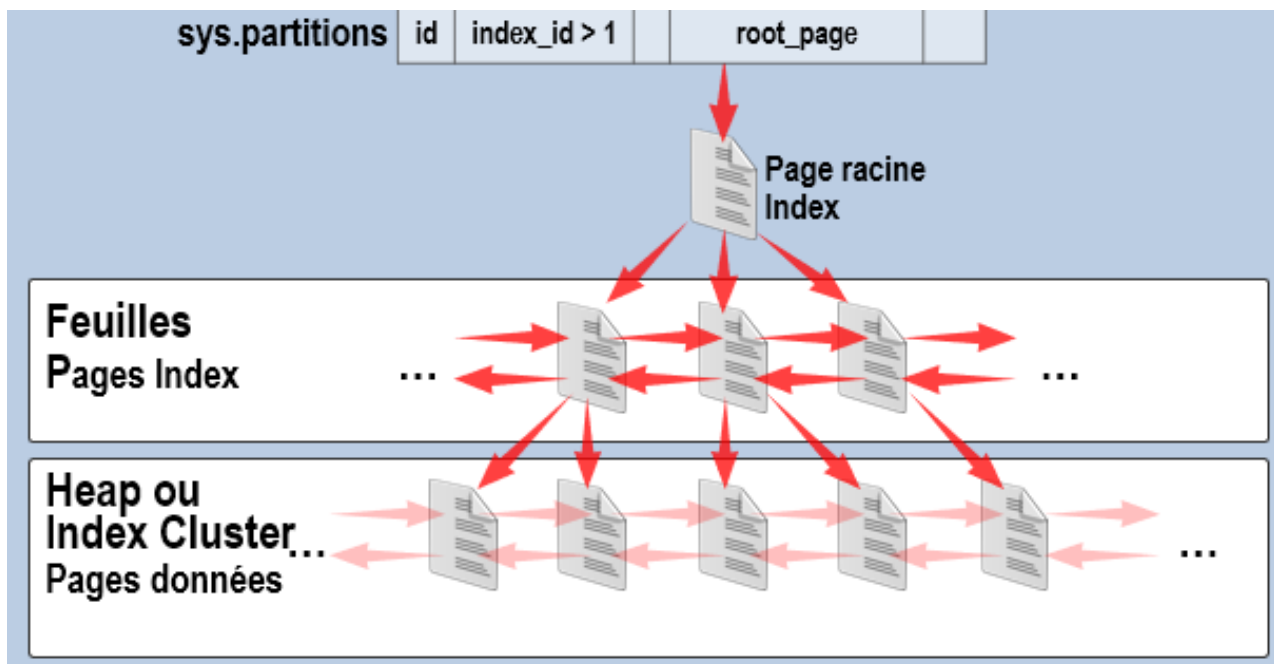


13. <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/graph-id-from-node-id-transact-sql?view=sql-server-ver16>

Index Non Cluster

Arbre B-tree référence un heap ou un index cluster

Au maximum 249 index non cluster par table



Index sur les colonnes de données de type xml

Représentation B-tree des nœuds des valeurs xml

Amélioration des performances car le moteur de requête ne doit pas transformer les données XML à chaque requête

Type d'index	Description
Primaire	représentation cluster B-tree des nœuds dans une colonne xml
Secondaire	Path Requêtes par path et valeur
	Property Requêtes par path
	Value Requêtes par path imprécis

¹⁴. <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/node-id-from-parts-transact-sql?view=sql-server-ver16>

7.2. Création d'index

Présentation

```
1 CREATE [ UNIQUE ] [ CLUSTERED | NONCLUSTERED ] INDEX index_name ON { table | view } ( column
  [ ASC | DESC ] [ ,...n ] ) INCLUDE ( column [ ,...n ] ) [WITH option [ ,...n ] ] [ON
  {partition_scheme (column) | filegroup | "default" } ]
```

Option WITH	But
ALLOW_ROW_LOCKS	Active/désactive les verrous de ligne sur l'index
ALLOW_PAGE_LOCKS	Active/désactive les verrous de page sur l'index
ONLINE	Active/désactive l'accès à l'index pendant la création
FILLFACTOR	Contrôle l'espace libre dans les pages feuilles
PAD_INDEX	Contrôle l'espace libre dans les pages non feuilles

Index unique

Assure que les valeurs sont uniques dans la clé

Index multi-colonnes

Index composite

Comporte jusqu'à 16 colonnes et 900 octets dans une clé

```
1 CREATE NONCLUSTERED INDEX K_Contact_LastName_FirstName ON Person.Contact ( LastName ASC,
  FirstName ASC)
```

Colonnes incluses

on peut indiquer des colonnes non clés incluses dans l'index

```
1 CREATE NONCLUSTERED INDEX AK_Employee_LoginID ON HumanResources.Employee ( LoginID ASC)
  INCLUDE ( ContactID, NationalIDNumber)
```

filtre : on peut indiquer des conditions de filtrage pour n'indexer que certains enregistrements (amélioration de performances sur certaines requêtes si l'index n'est pas très discriminant)

Emplacement de stockage

On peut spécifier le groupe de fichier dans lequel stocker l'index.

Par exemple, on peut améliorer les performances en plaçant un index non cluster dans un autre groupe de fichier que celui contenant les données.

Création d'index sur des colonnes calculées

Création possible si :

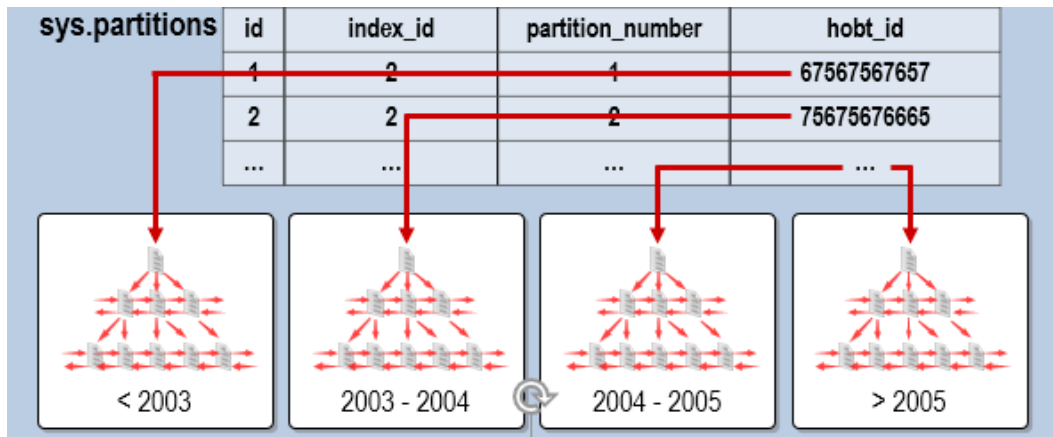
- Expression est déterministe
- Option ANSI_NULLS connection-level à ON
- Option NUMERIC_ROUNDABORT à OFF
- Colonnes ne sont pas des types text, ntext ou image

Index partitionné

Index est partitionné horizontalement par plage semblable à une table partitionnée

Attention à l'alignement avec les tables sous-jacentes et l'inclusion de clé partitionnée dans les clés d'index (pour les index uniques)

15. <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/object-id-from-edge-id-transact-sql?view=sql-server-ver16>



Options d'espace libre dans les index

- La présence d'espace libre améliore les performances des mise à jour des index
- FILLFACTOR définit le pourcentage de remplissage pour les feuilles au moment de la création de l'index
80 -> 80 % de l'espace est rempli et 20% reste libre pour des ajouts de valeur d'index
l'espace libre est réparti régulièrement dans la feuille
référence¹⁸
- Recommandations :
 - FILLFACTOR faible pour des applications *OLTP* (cf. p.107)
beaucoup d'espace libre pour faciliter les modifications de données
attention : peut ralentir les lectures car il y a plus de pages à lire
 - FILLFACTOR élevé pour des applications *OLAP*
pas besoin d'espace libre car pas de modification des données
- PAD_INDEX permet de spécifier si FILLFACTOR est utilisé pour les pages intermédiaires (par défaut OFF)

```
1 CREATE UNIQUE NONCLUSTERED INDEX [AK_Employee_LoginID] ON [HumanResources].[Employee] (
  [LoginID] ASC) WITH ( FILLFACTOR = 65, PAD_INDEX = ON)
```

Méthodes pour obtenir des informations

- SQL Server Management Studio
 - Explorateur d'objet
 - Fenêtre des propriétés des index
 - Rapports
- Procédures stockées système
 - sp_help
 - sp_helpindex
- Vues du catalogue
- Fonctions système

```
1 -- Information sur les index jamais utilisés
2 SELECT
3     object_name(i.object_id) AS table_name,
4     i.name AS index_name,
5     i.index_id,
6     i.type_desc,
7     us.user_seeks,
```

¹⁶ <https://learn.microsoft.com/en-us/sql/t-sql/functions/graph-functions-transact-sql?view=sql-server-ver16>

```

8      us.user_scans,
9      us.user_lookups,
10     us.user_updates
11 FROM sys.indexes i
12 LEFT JOIN sys.dm_db_index_usage_stats us
13     ON i.object_id = us.object_id AND i.index_id = us.index_id AND database_id = DB_ID()
14 WHERE i.type_desc <> 'HEAP' -- ignore les tables sans cluster index
15     AND (user_seeks IS NULL OR user_seeks = 0)
16     AND (user_scans IS NULL OR user_scans = 0)
17     AND (user_lookups IS NULL OR user_lookups = 0)
18     AND lower(i.name) like lower('I%')
19 ORDER BY user_updates DESC;
20
21

```

1	table_name	index_name	index_id	type_desc
2	user_seeks	user_scans	user_lookups	user_updates
3	PersonPhone	IX_PersonPhone_PhoneNumber	2	NONCLUSTERED
4	vProductAndDescription	IX_vProductAndDescription	1	CLUSTERED
5	vStateProvinceCountryRegion	IX_vStateProvinceCountryRegion	1	CLUSTERED
6	fulltext_index_docidstatus_946 i1		1	CLUSTERED
7	fulltext_index_docidstatus_946 i2		2	NONCLUSTERED
8	fulltext_index_docidstatus_946 i3		3	NONCLUSTERED
9	fulltext_docidfilter_946102411 i1		1	CLUSTERED
10	fulltext_indexeddocid_94610241 i1		1	CLUSTERED
11	fulltext_index_docidstatus_192 i1		1	CLUSTERED
12	fulltext_index_docidstatus_192 i2		2	NONCLUSTERED
13	fulltext_index_docidstatus_192 i3		3	NONCLUSTERED
14	fulltext_docidfilter_192558189 i1		1	CLUSTERED
15	fulltext_indexeddocid_19255818 i1		1	CLUSTERED
16	ShoppingCartItem	IX_ShoppingCartItem_ShoppingCa	2	NONCLUSTERED
17	fulltext_index_docidmap_141358 i1		1	CLUSTERED

7.3. Fragmentation d'index

Fragmentation d'index

SQL Server réorganise les pages d'index quand les données sont modifiées et scinde les pages d'index

Types de fragmentation

Interne – Les pages ne sont pas pleines

Externe – Les pages ne sont pas dans un ordre logique

17. <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/object-id-from-node-id-transact-sql?view=sql-server-ver16>

Détection de la fragmentation

SQL Server Management Studio – Fenêtre propriété des index

Fonction système – sys.dm_db_index_physical_stats

```
1 SELECT
2     OBJECT_SCHEMA_NAME(ips.object_id) AS schema_name,
3     OBJECT_NAME(ips.object_id) AS object_name,
4     i.name AS index_name,
5     i.type_desc AS index_type,
6     ips.avg_fragmentation_in_percent,
7     ips.page_count
8 FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'LIMITED') ips
9 INNER JOIN sys.indexes i ON ips.object_id = i.object_id AND ips.index_id = i.index_id
10 WHERE ips.avg_fragmentation_in_percent > 10 AND ips.page_count > 1000
11 ORDER BY ips.avg_fragmentation_in_percent DESC;
```

Options pour défragmenter les index

fragmentation <= 30% : Reorganize

```
1 ALTER INDEX AK_Product_Name ON Production.Product REORGANIZE
```

fragmentation > 30% : Rebuild

```
1 ALTER INDEX AK_Product_Name ON Production.Product REBUILD
```

7.4. Resumable index

Principe

Possibilité de suspendre et reprendre les opérations d'index de longue durée (CREATE, ALTER, DROP) sans perdre le travail déjà effectué.

Introduits dans SQL Server 2017

Sans Resumable Index

- Les opérations d'index longues devaient être relancées depuis le début si interrompues
- Risque de blocage prolongé des ressources
- Difficultés de maintenance dans les fenêtres limitées

Avec Resumable Index

- Reprise là où l'opération s'est arrêtée
- Contrôle granulaire des opérations
- Maintenance flexible dans le temps

Référence¹⁹

```
1 CREATE INDEX IX_Example ON TableName(Column1, Column2)
2 WITH (
3     RESUMABLE = ON,                -- Active le mode resumable
4     MAX_DURATION = 120 MINUTES,    -- Durée maximale avant pause automatique
5     ONLINE = ON,                  -- Compatible avec les opérations en ligne
6     MAXDOP = 4                    -- Degré de parallélisme
7 )
```

¹⁸ <https://learn.microsoft.com/en-us/sql/relational-databases/indexes/specify-fill-factor-for-an-index?view=sql-server-ver16>

SQL Server 2022

REORGANIZE, REBUILD, DROP resumables

```

1 ALTER INDEX ALL ON Sales.Orders REBUILD
2 WITH (
3     RESUMABLE = ON,
4     MAX_DURATION = 180 MINUTES,
5     ONLINE = ON,
6     MAXDOP = 0
7 )

```

Manipulations

Pause Manuelle

Reprise d'Opération

Annulation d'Opération

```

1 -- Suspendre une opération resumable
2 ALTER INDEX IX_Orders_CustomerDate ON Orders PAUSE
3
4 -- Reprendre une opération suspendue
5 ALTER INDEX IX_Orders_CustomerDate ON Orders RESUME WITH (MAX_DURATION = 60 MINUTES)
6
7 -- Reprendre avec nouvelles options
8 ALTER INDEX IX_Orders_CustomerDate ON Orders RESUME WITH ( MAX_DURATION = 120 MINUTES, MAXDOP
  = 8 )
9
10 -- Annuler complètement l'opération
11 ALTER INDEX IX_Orders_CustomerDate ON Orders ABORT

```

Surveillance

vue sys.index_resumable_operations

```

1 -- Requête de monitoring
2 SELECT
3     -- Informations générales
4     DB_NAME() AS [Base],
5     OBJECT_SCHEMA_NAME(iro.object_id) + '.' +
6     OBJECT_NAME(iro.object_id) AS [Table],
7     ISNULL(iro.name, 'HEAP (index_id=0)') AS [Index],
8
9     -- État de l'opération
10    iro.state_desc AS [État],
11    iro.percent_complete AS [Progression %],
12
13    -- Temps
14    iro.start_time AS [Début],
15    CASE
16        WHEN iro.state_desc = 'Paused' THEN iro.last_pause_time
17        ELSE NULL
18    END AS [Dernière pause],
19    iro.total_execution_time AS [Durée totale (min)],
20
21    -- Estimation
22    CASE
23        WHEN iro.percent_complete > 0
24        THEN DATEADD(MINUTE,
25            (iro.total_execution_time * (100.0 / iro.percent_complete)) -
26            iro.total_execution_time,
27            GETDATE())
28        ELSE NULL

```

19. <https://learn.microsoft.com/en-us/sql/relational-databases/indexes/guidelines-for-online-index-operations?view=sql-server-ver16#resumable-index-considerations>

```
29     END AS [Fin estimée],
30
31     -- Ressources
32     FORMAT(iro.page_count * 8.0 / 1024, 'N0') AS [Taille estimée (MB)]
33
34 FROM sys.index_resumable_operations iro
35 ORDER BY iro.start_time DESC
```

8. Sauvegardes et restaurations

8.1. Sauvegardes et restaurations

Les différents types de sauvegardes.

Type	Détails
Complète	Fichiers de données et une partie des log
Journal	Modifications enregistrées dans le log
Fin du journal	Partie active des log
Différentielle complète	Modifications depuis la dernière
Fichier/Groupe	Fichiers ou groupes spécifiques

Mode de récupération et influence sur la récupération des données.

- Simple : Sauvegardes complètes ou différentielles, pas de log
- Complète : Sauvegardes avec log
- Journalisé en bloc : Moins d'espace du fichier log

Stratégies de sauvegarde.

- sauvegarde complète
Petite base, peu ou pas de changement
Penser à vider le log
- sauvegarde base et fichier log
Base souvent modifiée
Temps de sauvegarde complète long
- sauvegarde différentielle
Base modifiée souvent
Utilisée pour minimiser le temps de sauvegarde
Journal de log à part
- sauvegarde fichiers ou groupes de fichier
Base de grande taille, sauvegarde complète trop longue
Sauvegarde de log à part
Peut être complexe à restaurer ...

Opérateurs de sauvegarde

Rôles :

- sysadmin
- db_owner
- db_backupoperator

Supports

- Bande
- Disque
- Notion de support logique
- Attention à la sauvegarde sur plusieurs supports

Réalisation de la sauvegarde

BACKUP DATABASE TO

BACKUP LOG

BACKUP DATABASE ... WITH DIFFERENTIAL

BACKUP DATABASE FILE | FILEGROUP

BACKUP ... MIRROR TO : écriture de la sauvegarde sur 2 supports

CHECKSUM

RESTORE VERIFONLY

Restauration

- Principe :
 - Copie des données
 - Journal des log rejoué
- Restauration d'une base :
 - RESTORE
 - MOVETO
 - REPLACE
- Restauration d'un journal :
 - RESTORE LOG
 - NORECOVERY / RECOVERY
- Restauration fichier/groupe :
 - RESTORE FILE
 - RESTORE LOG

8.2. Restauration à partir d'un cliché

Rappel sur un cliché

- État de la base à un instant donné, en lecture seule
- Même serveur que la base source

cliché (cf. p.81)

Restauration

deux façons de faire :

- montage du cliché et Instructions SQL classiques (UPDATE, INSERT)
- RESTORE DATABASE FROM DATABASE_SNAPSHOT =

8.3. Traitement des bases système

Sauvegardes des bases

- master : après chaque création / modification de base
- msdb : après chaque création de job, alertes, ...
- model : après chaque modification

Restauration

Restauration à partir d'un backup

Sinon : création à partir de SQL Studio ou script

Rattacher les bases non endommagées plutôt que restauration

Restauration de master

- SQL Server accessible
 1. SQL Server mono-utilisateur (sqlservr.exe -c -m)
powershell en mode administrateur :
& 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Binn\sqlservr.exe' -sMSSQLSERVER -c -m
 2. restauration de master depuis une sauvegarde
RESTORE DATABASE master FROM DISK = 'C:\Backup\master.bak' with replace
 3. redémarrer SQL Server
- SQL inaccessible
 1. setup.exe /QUIET /ACTION=REBUILDDATABASE /INSTANCENAME=MSSQLSERVER
SQLSYSADMINACCOUNTS="Administrateur" /SQLCOLLATION=French_CI_AS
 2. restauration « SQL Server accesssibel »

9. Sécurité et autorisations

9.1. Vue d'ensemble

principaux éléments

compte de connexion : identifiant permettant de se connecter à l'instance SQL Server

utilisateur de base de données : compte permettant d'interagir avec une base de données

autorisations : droits permettant d'effectuer des opérations sur l'instance, sur une base, sur des objets de la base

rôles : entités disposant de certaines autorisations

9.2. Authentification sur l'instance

- Windows
- mode mixte

9.3. Comptes de connexion

2 types de connexions

- Comptes Windows : utilisation d'un compte individuel ou appartenance à un groupe.
Comptes ou groupes locaux ou Active Directory
- Comptes SQL Server : compte et mot de passe stockés dans SQL Server (base master)

CREATE LOGIN

Paramètres d'une connexion

- base par défaut
- activation/désactivation
- credential

Authentification vers d'autres ressources Windows

CREATE CREDENTIAL

informations sur les credential : sys.credentials

Rôles fixes du serveur

Rôle	Détails
sysadmin	Tous les droits
dbcreator	Création et modification de bases
diskadmin	Gestion des disques
serveradmin	Configuration du serveur
securityadmin	Gestion des comptes
processadmin	Gestion des process

Rôle	Détails
bulkadmin	BULK INSERT
setupadmin	Gestion réplication et serveurs liés

Attribution et informations sur les rôles fixes de serveur

sp_helpsrvrole

sp_helpsrvrolemember

Sources d'information sur les connexions

sys.server_principals	Liste des connexions définies sur le serveur.
sys.sql_logins	Liste des connexions définies avec le mode de sécurité SQL Server.
sys.server_permissions	Liste des autorisations accordées aux connexions directement sur le serveur.
sys.server_role_member	Liste des rôles de serveur et des connexions qui en bénéficient.
sys.system_components_surface_area_configuration	Retourne la liste des objets système susceptibles d'être visibles ou non en fonction de la configuration effectuée.

9.4. Utilisateurs de base de données

Compte d'interaction avec une base de données

utilisé pour interagir avec une base de données

correspondance d'un compte de connexion avec des utilisateurs de base de données

CREATE USER

liste des utilisateurs d'une base : sys.database_principals

liste des utilisateurs connectés : sp_who

```

1 SELECT s.name as "Connexion", p.name as "Utilisateur" FROM sys.database_principals p
2 INNER JOIN sys.server_principals s ON s.sid=p.sid;

1 DECLARE cLesBases cursor for select name
2 from sys.databases
3 where name not in('master','model','tempdb','msdb');
4
5 DECLARE @nomBase sysname; DECLARE @rqt nvarchar(500) BEGIN
6 OPEN cLesBases;
7 FETCH cLesBases INTO @nomBase;
8 WHILE (@@FETCH_STATUS=0) BEGIN
9 set @rqt='SELECT '''+@nomBase+''' as Base,'
10 set @rqt=@rqt+'s.name as Connexion, p.name as Utilisateur ' set @rqt=@rqt+'FROM
   master.sys.server_principals p '
11 set @rqt=@rqt+'INNER JOIN '+@nomBase+'.sys.database_principals s ' set @rqt=@rqt+'ON
   s.sid=p.sid'
12 exec sp_executesql @rqt
13 FETCH cLesBases INTO @nomBase; END;
14 CLOSE cLesBases;
15 DEALLOCATE cLesBases

```

```
16 END;  
17
```

propriétés

schéma par défaut

comptes de connexion correspondant

Utilisateurs prédéfinis

- dbo
Par défaut dans toutes les bases
Membres du rôle sysdamin et compte sa
Ne peut pas être détruit
- guest user
Désactivé par défaut
Utilisé pour l'accès à une base sans user associé à la connexion

9.5. Compte utilisateurs sans connexion

Rendre une base de données indépendante du serveur

Az Définition

Compte utilisateur permettant de se connecter sur la base sans association avec un compte de connexion

Prérequis

Instance SQL Server en mode contained database authentication

```
1 sp_configure 'contained database authentication',1  
2 reconfigure
```

option CONTAINMENT de la base de données avec la valeur PARTIAL

```
1 ALTER DATABASE [GESCOM] SET CONTAINMENT = PARTIAL WITH NO_WAIT;
```

Remarque : il faut obligatoirement indiquer la base de données lors de la connexion avec ce type de compte.

9.6. Permissions

Principes

GRANT / REVOKE

Notion d'héritage

DENY

droits sur la base

CREATE

DROP

ALTER

droits sur les objets de la base

SELECT

UPDATE

INSERT

DELETE

EXECUTE

droits sur l'instance

CREATE ANY DATABASE

SHUTDOWN

...

9.7. Rôles

Rôles de serveur

CREATE SERVER ROLE

Rôle de base de données

- rôles prédéfinis : db_accessadmin, db_backupoperato, ...
- rôles utilisateur

CREATE ROLE

Rôles de base prédéfinis

Rôle	Description
db_owner	Ensemble de droits équivalents à ceux du propriétaire de la base.
db_accessadmin	Permet d'ajouter ou de supprimer des utilisateurs dans la base de données.
db_datareader	Permet de consulter (SELECT) le contenu de toutes les tables de la base de données.
db_datawriter	Permet d'ajouter (INSERT), modifier (UPDATE) ou supprimer (DELETE) des données dans toutes les tables utilisateur de la base de données.
db_ddladmin	Permet d'ajouter (CREATE), modifier (ALTER) ou supprimer (DELETE) des objets de la base de données.
db_securityadmin	Permet de gérer les rôles, les membres des rôles, et les autorisations sur les instructions et les objets de la base de données.
db_backupoperator	Permet d'effectuer une sauvegarde de la base de données.
db_denydatareader	Interdit la visualisation des données de la base.
db_denydatawriter	Interdit la modification des données contenues dans la base.

Membres d'un rôle

1 ALTER ROLE nomRoleServer ADD MEMBER nomUtilisateur;

2 ALTER ROLE nomRoleServer DROP MEMBER nomUtilisateur ;

Informations sur les rôles

```
1 select utilisateurs.name, roles.name
2 from sys.database_role_members as drm join sys.database_principals as utilisateurs
3 on drm.member_principal_id=utilisateurs.principal_id join sys.database_principals as roles
4 on drm.role_principal_id=roles.principal_id where roles.type='R';
5
```

Rôle d'application

Application associée à un rôle

Contexte de sécurité n'existe que quand l'application tourne

CREATE APPLICATION ROLE

Activation avec sp_setapprole

```
1 sp_setapprole 'rôle', [Encrypt N] 'mot-passe' [, 'style_cryptage']
```

10. Taches d'administration

10.1. Tâches d'administration

L'outil de plan de maintenance

Utilisation de plans de maintenance prédéfinis

(cf. *plans maintenance.png*) (cf. p.105)

Agent SQL Server.

- planification de taches
- travaux
- alertes
- envoi de notification
- opérateurs

10.2. Stratégies

Problème

S'assurer que tous les paramétrages voulus sont bien définis pour une nouvelle base est coûteux

Vérifier que le paramétrage d'une base en cours d'utilisation n'a pas été modifié est coûteux

Solution proposées par SQL Server : Stratégies, facettes et conditions

Stratégie

Az Définition

Ensemble de règles de validation de certaines propriétés d'un objet.

Exemple : est ce que le propriétaire de l'objet est dbo ?

Une stratégie est définie à partir d'une facette ou d'une condition.

Elle peut être évaluée automatiquement ou à la demande.

Les objets sur lesquels elle s'applique peuvent être définis par des conditions

Facette

Az Définition

Modèle définissant un ensemble de règles pour vérifier la conformité d'un objet

Exemple : la facette table permet de contrôler le paramétrage d'une table de la base de données.

Conditions

Az Définition

Règles définissant des valeurs de propriétés

Elles sont construites à partir d'une facette

10.3. Monitoring

Outil Monitor

Suivi en tant réel du fonctionnement de l'instance

vues système

Vue ou fonction	Information
sys.dm_exec_requests	Requêtes en cours
sys.dm_exec_sql_text(handle)	Texte SQL de la requête
sys.dm_exec_query_plan(handle)	Plan d'exécution
sys.dm_exec_sessions	Infos sur les connexions
sys.dm_tran_session_transactions	Sessions avec transactions
sys.dm_tran_active_transactions	Transactions actives

```

1 -- affichage des transactions en cours
2 select st.* , at.*
3 FROM sys.dm_tran_active_transactions at
4 JOIN sys.dm_tran_session_transactions st ON at.transaction_id = st.transaction_id
5 WHERE at.transaction_state != 4 -- 4 = Committed

```

10.4. Extended Event

Extended Events

- visualisation en temps réel des XEvent avec les sessions standards
- création de sessions spécifiques, visualisables en live ou stockées

Création de sessions spécifiques

création avec CREATE EVENT SESSION

spécification d'un fichier d'enregistrement

```

1 CREATE EVENT SESSION [suivi_login] ON SERVER
2 ADD EVENT sqlserver.login(
3     ACTION(sqlserver.client_hostname,sqlserver.database_name,sqlserver.username)),
4 ADD EVENT sqlserver.logout(
5     ACTION(sqlserver.client_hostname,sqlserver.database_name,sqlserver.username))
6 ADD TARGET package0.event_file(SET filename=N'C:\sessions\suivi_login.xel')
7 WITH (MAX_MEMORY=4096 KB,EVENT_RETENTION_MODE=ALLOW_SINGLE_EVENT_LOSS,MAX_DISPATCH_LATENCY=30
8     SECONDS,MAX_EVENT_SIZE=0 KB,MEMORY_PARTITION_MODE=NONE,TRACK_CAUSALITY=OFF,STARTUP_STATE=ON)
8 GO

```

Analyse

Analyse interactive depuis SSMS

Analyse par T-SQL

```

1 SELECT
2     event_data.value('(event/@name)[1]', 'varchar(100)') AS event_name,
3     event_data.value('(event/@timestamp)[1]', 'datetime2') AS [timestamp],
4     event_data.value('(event/data[@name="statement"]/value)[1]', 'nvarchar(max)') AS
statement_text,
5     event_data.value('(event/action[@name="sql_text"]/value)[1]', 'nvarchar(max)') AS
sql_text,
6     event_data.value('(event/action[@name="client_app_name"]/value)[1]', 'nvarchar(max)') AS
application_name,
7     event_data.value('(event/action[@name="username"]/value)[1]', 'nvarchar(max)') AS
username

```

```

8 FROM (
9     SELECT CAST(event_data AS XML) AS event_data
10    FROM sys.fn_xe_file_target_read_file(
11        'C:\backups\masession*.xel', -- wildcard possible
12        NULL, NULL, NULL)
13 ) AS x;

```

event_name	timestamp	statement_text
sql_text	application_name	username
login	2025-05-13 15:52:37.8890000	NULL
NULL	NULL	win10\admin
login	2025-05-13 15:52:38.2180000	NULL
NULL	NULL	win10\admin
login	2025-05-13 15:52:38.2330000	NULL
NULL	NULL	win10\admin
login	2025-05-13 15:52:38.3010000	NULL
NULL	NULL	win10\admin
login	2025-05-13 15:52:38.3300000	NULL
NULL	NULL	win10\admin

Information sur les deadlocks

[👁 Exemple](#)

utilisation des événements étendus

```

1 -- Activation du trace flag pour capturer les deadlocks
2 DBCC TRACEON(1222, -1);
3
4 -- Événements étendus pour analyser les deadlocks
5 CREATE EVENT SESSION deadlock_monitor ON SERVER
6 ADD EVENT sqlserver.xml_deadlock_report
7 ADD TARGET package0.event_file(SET filename='C:\temp\deadlocks.xel');

```

11. Programmation T-SQL

11.1. Syntaxe de base

11.1.1. Directive de lot

Instruction GO

Indique la fin d'un lot d'instructions

C'est une instruction reconnue par sqlcmd, osql ou SSMS

Ce n'est pas une commande SQL

11.1.2. Commentaires

2 méthodes

– commentaire sur la ligne en cours

/* */ commentaires sur plusieurs lignes

11.1.3. Identificateurs

Identificateurs standards

Le premier caractère doit être alphabétique

Les autres caractères peuvent être des lettres, des chiffres ou des symboles

Les identificateurs commençant par un symbole ont un usage spécial

Identificateurs délimités

À utiliser lorsque les noms contiennent des espaces ou des mots réservés

Délimités par crochets ([]) ou des guillemets (" ")

11.1.4. Variables

Définition

```
1 declare @fid int
2 set @fid=3
3 select title from film where film_id=@fid
```

Définies par l'utilisateur avec l'instruction DECLARE @nom type

Portée locale au script

Unicité de définition

 Attention

On ne peut définir une variable qu'une fois par lot ou procédure

```
1 declare @n int
2
3 declare @n int
4 Msg 134, Niveau 15, État 1, Ligne 3
5 The variable name '@n' has already been declared. Variable names must be unique within a
   query batch or stored procedure.

1 declare @n int
2 GO
3 declare @n int
```

Affectation

Attribuées au moyen de l'instruction SET ou SELECT @

```
1 declare @titre varchar(256)
2 select @titre=title from film where film_id=@fid
3 select @titre as titre
```

11.1.5. Opérateurs

Types d'opérateurs

Arithmétique

Comparaison

Concaténation de chaînes

Logique

Niveaux de priorité des opérateurs

11.1.6. Expressions

Expression

Combinaison de symboles et d'opérateurs

Évaluation en une valeur scalaire individuelle

Type de données du résultat fonction des éléments dans l'expression

```
1 declare @tva int
2 set @tva=20
3
4 select amount*(1-@tva/100.) as montantHT from payment
```

11.1.7. Bloc d'instruction

BEGIN ... END

Regroupement d'instructions, utilisé dans des constructions IF ou WHILE

peut être imbriqué

Références

Documentation²⁰

11.1.8. Tests

a) IF

IF, ELSE

IF expression_booléenn

instruction ou bloc d'instruction

ELSE

instruction ou bloc d'instruction

```
1 use sakila
2 declare @nb int
3
4 select @nb=count(*) from film where title like 'W%'
5
6 if @nb > 10
7 begin
```

²⁰. <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/begin-end-transact-sql?view=sql-server-ver-16>

```

8  select title from film where title like 'W%'
9 end
10 else
11  select 'moins de 10 film' as title

```

Imbrications

Les blocs suivant un IF ou un ELSE peuvent contenir des IF

Références

Documentation²¹

b) CASE

CASE

CASE expression

WHEN val1 THEN expression1

[WHEN val2 THEN expression2]

[WHEN val3 THEN expression3]

[ELSE expression4]

END

```

1 SELECT
2     first_name,
3     last_name,
4     rental_date,
5     CASE
6         WHEN return_date IS NULL THEN 'Pas encore rendu'
7         WHEN return_date <= DATEADD(DAY, 7, rental_date) THEN 'Rendu dans les délais'
8         ELSE 'En retard'
9     END AS etat_emprunt
10 FROM rental
11 JOIN customer ON rental.customer_id = customer.customer_id

```

Syntaxe alternative de CASE

Remarque

CASE

WHEN expression booléenne THEN instructions

[WHEN expression booléenne THEN instructions]

[WHEN expression booléenne THEN instructions]

[ELSE Instructions]

END

Imbrications

Les instructions CASE peuvent être imbriquées, jusqu'à 10 niveaux

Références

Documentation²²

21. <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/if-else-transact-sql?view=sql-server-ver16>

11.1.9. Boucle

WHILE

WHILE expression_booléenn

instruction ou bloc d'instruction

```
1 use sakila
2 declare @nb int
3
4 select @nb=count(*) from film where title like 'W%'
5
6 while @nb > 0
7 begin
8   select @nb nombre
9   set @nb=@nb-1
10 end
```

Imbrications

Les blocs suivant un WHILE peuvent contenir des WHILE

Références

Documentation

11.1.10. Curseurs

a) Présentation des curseurs

Curseur

Az Définition

type d'objet particulier d'une base de données qui permet d'accéder aux lignes renvoyées par une requête.

Cycle de vie

définition avec DECLARE CURSOR

ouverture avec OPEN (chargement des données)

récupération de résultat avec FETCH (dans une boucle WHILE)

mise à jour avec UPDATE ... WHERE CURRENT OF

suppression avec DELETE ... WHERE CURRENT OF

fermeture avec CLOSE

libération des ressources avec DEALLOCATE

b) Définition d'un curseur

DECLARE CURSOR (syntaxe ISO)

```
1 DECLARE cursor_name [ INSENSITIVE ] [ SCROLL ] CURSOR
2 FOR select_statement
3 [ FOR { READ_ONLY | UPDATE [ OF column_name [ , ...n ] ] } ]
4 [ ; ]
```

INSENSITIVE : copie des données dans une table tampon. Pas de modification possible des données dans le curseur, ni de visualisation des modifications d'autres transactions

SCROLL : tous les déplacements sont possibles

FOR READ ONLY : pas de modification possible du contenu du curseur

²² <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/case-transact-sql?view=sql-server-ver16>

FOR UPDATE : autorise la mise à jour du curseur ou de certaines colonnes seulement

DECLARE CURSOR (syntaxe Transact SQL)

```

1 DECLARE cursor_name CURSOR [ LOCAL | GLOBAL ]
2 [ FORWARD_ONLY | SCROLL ]
3 [ STATIC | KEYSET | DYNAMIC | FAST_FORWARD ]
4 [ READ_ONLY | SCROLL_LOCKS | OPTIMISTIC ]
5 [ TYPE_WARNING ]
6 FOR select_statement
7 [ FOR UPDATE [ OF column_name [ , ...n ] ] ]
8 [ ; ]
9

```

LOCAL : le curseur est défini uniquement dans l'entité dans laquelle il est déclaré.

GLOBAL : le curseur est défini dans la session en cours.

Par défaut : option de base de données **default to local cursor**.

FORWARD_ONLY : les déplacements ne sont possibles que vers l'avant

SCROLL : tous les déplacements sont possibles

STATIC : copie des données dans une table tampon. Pas de visualisation des modifications d'autres transactions

DYNAMIC : visualisation de toutes les modifications apportées aux données

KEYSET : intermédiaire entre STATIC et DYNAMIC : visualisation des modifications d'autres transactions, mais pas les nouveaux enregistrements

FAST_FORWARD : curseur FORWARD_ONLY et READ ONLY, aux performances optimisées.

READ ONLY : pas de modification possible du contenu du curseur

SCROLL_LOCKS : verrouillage des lignes lues. La réussite des modifications éventuelles est garantie.

OPTIMISTIC : pas de verrouillage des lignes lues. Une modification peut échouer (contrôle de modification sur une colonne timestamp ou checksum de l'enregistrement)

TYPE_WARNING : spécifie qu'un avertissement est émis en cas de conversion du type de curseur demandé en un autre type par le moteur SQL

FOR UPDATE : autorise la mise à jour du curseur ou de certaines colonnes seulement

Informations sur les curseurs

Procédures stockées système	Description
sp_cursor_list ²³	Renvoie une liste des curseurs actuellement visibles par la connexion et leurs attributs.
sp_describe_cursor	Décrit les attributs d'un curseur, par exemple s'il s'agit d'un curseur vers l'avant uniquement ou de défilement.
sp_describe_cursor_columns	Décrit les attributs des colonnes contenues dans le jeu de résultats du curseur.
sp_describe_cursor_tables	Décrit les tables de base auxquelles accède le curseur.

²³ <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-stored-procedures/sp-cursor-list-transact-sql?view=sql-server-ver16>

c) Utilisation d'un curseur

FETCH

lecture de la ligne courante et déplacement vers une nouvelle ligne

FETCH

[[NEXT | PRIOR | FIRST | LAST

| ABSOLUTE { n | @nvar }

| RELATIVE { n | @nvar }

]

FROM

]

{ { [GLOBAL] cursor_name } | @cursor_variable_name }

[INTO @variable_name [,...n]]

Affichage des titres et durée des films

Exemple

```

1 use sakila
2
3 DECLARE @FilmTitle NVARCHAR(255);
4 DECLARE @FilmLength INT;
5
6 DECLARE FilmCursor CURSOR FOR
7 SELECT title, length
8 FROM film;
9
10 OPEN FilmCursor;
11
12 FETCH NEXT FROM FilmCursor INTO @FilmTitle, @FilmLength;
13 WHILE @@FETCH_STATUS = 0
14 BEGIN
15     PRINT 'Titre: ' + @FilmTitle + ', Durée: ' + CAST(@FilmLength AS NVARCHAR(10));
16     FETCH NEXT FROM FilmCursor INTO @FilmTitle, @FilmLength;
17 END;
18
19 CLOSE FilmCursor;
20 DEALLOCATE FilmCursor;
21
  
```

@@FETCH_STATUS

Remarque

Information sur la dernière opération FETCH effectuée

Valeur retournée	Description
0	L'instruction FETCH a réussi.
-1	L'instruction FETCH a échoué ou la ligne se situait au-delà du jeu de résultats.
-2	La ligne recherchée est manquante.
-9	Le curseur n'effectue pas d'opération de récupération (fetch).

Remarque : si plusieurs curseurs sont utilisés, il faut appeler @@FETCH_STATUS immédiatement après le FETCH.

Déplacement dans le curseur

Utilisation de l'attribut scroll dans le open ou la déclaration du curseur

Utilisation dans FETCH pour indiquer le déplacement (NEXT par défaut)

Déplacement	Description
NEXT	Déplacement d'une ligne
PRIOR	Retour en arrière d'une ligne
FIRST	Première ligne
LAST	Dernière ligne
ABSOLUTE nombre	Positionnement à la ligne nombre. Si nombre est négatif, le décompte est effectué en partant de la fin.
RELATIVE nombre	Déplacement relatif par rapport à la ligne courante. Si nombre est négatif, il y a retour en arrière dans les enregistrements.

clause current_of

modification d'enregistrement courant du curseur

```

1 declare c1 cursor for
2   select id, nom from Clients
3   for update;
4 declare @id int;
5 declare @nom nchar(20);
6
7 open c1;
8 fetch next from c1 into @id,@nom;
9
10 update t1 set nom='modifie' where current of c1;
11
12 close c1;
13 deallocate c1;
```

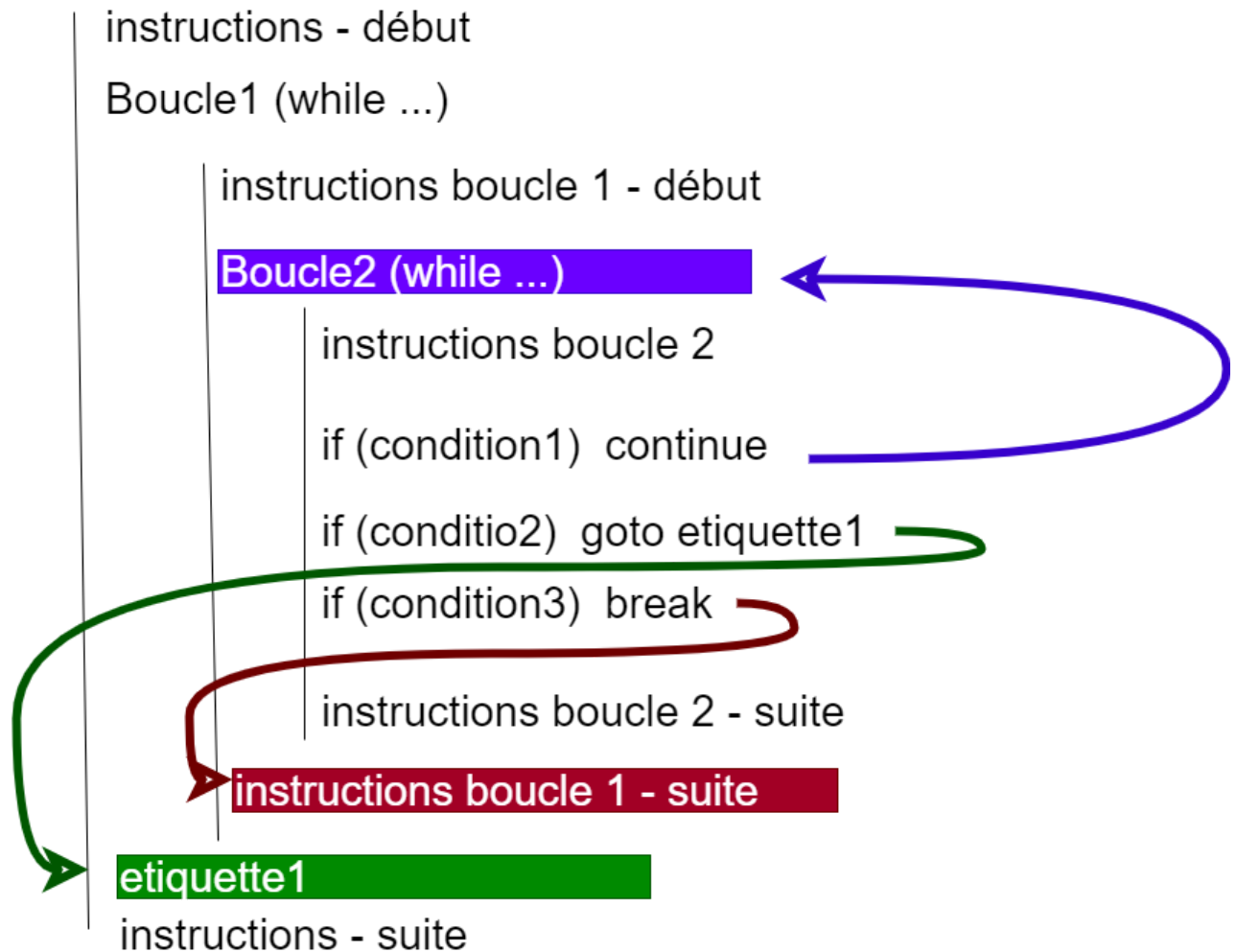
11.1.11. Rupture de boucle

BREAK

BREAK permet d'interrompre la boucle en cours et de poursuivre l'exécution à l'instruction suivant la boucle

Documentation²⁵

²⁴ <https://learn.microsoft.com/fr-fr/sql/t-sql/functions/fetch-status-transact-sql?view=sql-server-ver16#remarks>



CONTINUE

CONTINUE permet d'interrompre la boucle en cours et de reprendre l'exécution d'une nouvelle itération de la boucle

Documentation²⁶

Interruptions du traitement au niveau le plus interne

⚠ Attention

BREAK et CONTINUE s'appliquent à la boucle la plus interne.

Si deux boucles sont imbriquées, on ne peut pas interrompre la boucle externe avec BREAK et CONTINUE.

GOTO

GOTO permet d'interrompre le traitement en séquence et de le poursuivre à l'étiquette utilisée dans le GOTO

Documentation



Bonne pratique

Utiliser GOTO uniquement pour quitter plusieurs boucles imbriquées.

25. <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/break-transact-sql?view=sql-server-ver16>

11.1.12. Gestion d'erreurs

Exception

BEGIN TRY

instruction ou bloc d'instruction

END TRY

BEGIN CATCH

instruction ou bloc d'instruction

END CATCH

Fonction	Description
ERROR_NUMBER	Numéro de l'erreur
ERROR_SEVERITY	Sévérité
ERROR_STATE	Numéro d'état de l'erreur
ERROR_PROCEDURE	Nom de l'objet où l'erreur s'est produite
ERROR_LINE	Numéro de ligne où s'est produit l'erreur
ERROR_MESSAGE	Message d'erreur

Fonctions d'information disponibles

```

1 BEGIN TRY
2     BEGIN TRANSACTION;
3
4     UPDATE inventory
5     SET store_id = 2
6     WHERE inventory_id = 1000;
7
8     INSERT INTO rental (rental_date, inventory_id, customer_id, staff_id)
9     VALUES (GETDATE(), 100, 5, 100);
10
11    COMMIT TRANSACTION;
12 END TRY
13 BEGIN CATCH
14     ROLLBACK TRANSACTION;
15
16     SELECT
17         ERROR_NUMBER() AS ErrorNumber,
18         ERROR_SEVERITY() AS ErrorSeverity,
19         ERROR_STATE() AS ErrorState,
20         ERROR_PROCEDURE() AS ErrorProcedure,
21         ERROR_LINE() AS ErrorLine,
22         ERROR_MESSAGE() AS ErrorMessage;
23 END CATCH;
24

```

26. <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/while-transact-sql?view=sql-server-ver16>

ErrorNumber	ErrorSeverity	ErrorState	ErrorProcedure	ErrorLine	ErrorMessage
547	16	0	NULL	8	The INSERT statement conflicted with the FOREIGN KEY constraint "fk_rental_staff". The conflict occurred in database "sakila", table "dbo.staff", column 'staff_id'.

Deux possibilités pour lever une exception

THROW [{ error_number | @local_variable } , { message | @local_variable } , { state | @local_variable }]

ou

RAISERROR ({ msg_id | msg_str | @local_variable } { , severity , state } [, argument [, ...n]]) [WITH option [, ...n]]

RAISERROR (instruction)	Instruction THROW
Si un <i>msg_id</i> est passé, RAISERROR l'ID doit être défini dans <code>sys.messages</code> .	Le <i>paramètre error_number</i> n'a pas besoin d'être défini dans <code>sys.messages</code> .
Le <i>paramètre msg_str</i> peut contenir des mises en forme de type printf	Le <i>paramètre de message</i> n'accepte pas des mises en forme de type printf
Le paramètre <i>severity</i> spécifie la gravité de l'exception.	Il n'existe aucun <i>paramètre de gravité</i> . Lorsque HROW est utilisée, la gravité est toujours définie sur 16. Si THROW est utilisée pour réactiver une exception existante, la gravité est définie sur le niveau de gravité de cette exception.
N'honore pas SET XACT_ABORT ²⁷ .	Les transactions sont restaurées si SET XACT_ABORT ²⁸ est ON.

Différence RAISERROR et THROW

utilisation

 Remarque

Une exception ne peut pas être générée dans certains types de code : fonction, contraintes check, colonnes calculées persistantes, ...

Références

Documentation²⁹

²⁷. <https://learn.microsoft.com/fr-fr/sql/t-sql/statements/set-xact-abort-transact-sql?view=sql-server-ver16>

11.2. Fonctions et procédures

11.2.1. Définition d'une fonction

Fonction

Az Définition

- regroupement d'instructions
- nom
- paramètres (optionnels)
- valeur de retour

```
1 CREATE [ OR ALTER ] FUNCTION [ schema_name. ] function_name
2 ( [ { @parameter_name [ AS ] [ type_schema_name. ] parameter_data_type [ NULL ]
3 [ = default ] [ READONLY ] }
4 [ , ...n ]
5 ]
6 )
7 RETURNS return_data_type
8 [ WITH <function_option> [ , ...n ] ]
9 [ AS ]
10 BEGIN
11     function_body
12     RETURN value
13 END
14 [ ; ]
```

Nom d'un client

Exemple

```
1 use sakila
2 go
3
4 CREATE OR ALTER FUNCTION nom(@id INT)
5 RETURNS VARCHAR(30)
6 AS
7 BEGIN
8     DECLARE @result VARCHAR(30);
9     SELECT @result = first_name + ' ' + last_name FROM customer WHERE customer_id = @id;
10    RETURN @result;
11 END;
12
13 GO
14
15 SELECT dbo.nom(1);
```

Utilisation

Appel comme n'importe quelle fonction SQL standard avec un SELECT

CREATE OR ALTER

Remarque

- permet de remplacer la définition de la fonction

28. <https://learn.microsoft.com/fr-fr/sql/t-sql/statements/set-xact-abort-transact-sql?view=sql-server-ver16>

Options

```

1 <function_option> ::=
2 {
3     [ ENCRYPTION ]
4     | [ SCHEMABINDING ]
5     | [ RETURNS NULL ON NULL INPUT | CALLED ON NULL INPUT ]
6     | [ EXECUTE_AS_Clause ]
7     | [ INLINE = { ON | OFF } ]
8 }

```

ENCRYPTION : la définition de la fonction ne peut pas être lue

SCHEMABINDING : la définition de la fonction est liée aux objets utilisés, qui ne peuvent être modifiés qu'en supprimant la fonction.

RETURNS NULL ON NULL INPUT : la fonction n'est pas appelée et renvoie NULL si un paramètre est NULL

CALLED ON NULL INPUT : (Valeur par défaut) la fonction est appelée si un paramètre est NULL. Elle doit traiter ce cas d'appel.

EXECUTE_AS_Clause : la fonction est appelée avec la prérogative indiquée (CALLER | SELF | OWNER | 'user_name'). Référence³⁰

INLINE = { ON | OFF } : rend une fonction scalaire INLINE. Référence

11.2.2. Paramètres d'appel d'une fonction

Définition

- nom
- type de données
- valeur par défaut

Utilisation

Utilisation comme des variables locales.

Les valeurs sont définies lors de l'appel dans l'ordre de déclaration

```

1 CREATE OR ALTER FUNCTION get_nom_store
2     (@prenom VARCHAR(10),
3     @nom VARCHAR(10),
4     @store_id INT = 1)
5 RETURNS TABLE
6 AS
7 RETURN
8 (
9     SELECT first_name AS prenom, last_name AS nom
10    FROM customer
11   WHERE first_name LIKE @prenom
12         AND last_name LIKE @nom
13         AND store_id = @store_id
14 );
15 GO
16 select prenom,nom from dbo.get_nom_store('M%','M%',1)

```

valeur par défaut

Il faut utiliser le mot clé DEFAULT lors de l'appel.

```

1 select prenom,nom from dbo.get_nom_store('M%','M%', DEFAULT)

```

²⁹ <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/try-catch-transact-sql?view=sql-server-ver16>

11.2.3. Valeur de retour d'une fonction

Valeur de retour

Les résultats de la dernière instruction sont renvoyés par la fonction ou par le paramètre de l'instruction RETURN

Le type des résultats est fourni par la clause RETURNS typedonnées

Le retour peut être une valeur scalaire ou une table.

```

1 CREATE OR ALTER FUNCTION dbo.nom
2   (@id INT)
3 RETURNS VARCHAR(30)
4 AS
5 BEGIN
6   DECLARE @result VARCHAR(30);
7   SELECT @result = first_name + ' ' + last_name FROM customer WHERE customer_id = @id;
8   RETURN @result;
9 END;
10 go
11
12 select dbo.nom(1);

```

Valeur de retour de type table

RETURNS TABLE indique que la fonction renvoie une table

Il faut utiliser une instruction SELECT dans le return

```

1 CREATE OR ALTER FUNCTION get_nom_store
2   (@prenom VARCHAR(10),
3    @nom VARCHAR(10),
4    @store_id INT = 1)
5 RETURNS TABLE
6 AS
7 RETURN
8 (
9   SELECT first_name AS prenom, last_name AS nom
10  FROM customer
11  WHERE first_name LIKE @prenom
12        AND last_name LIKE @nom
13        AND store_id = @store_id
14 );
15 GO
16 select prenom,nom from dbo.get_nom_store('M%','M%',1)

```

Valeur de retour de type @variable table

RETURNS @var1 TABLE indique que la fonction renvoie la valeur de @var1 (de type table)

```

1 CREATE OR ALTER FUNCTION get_nom_store_table
2   (
3     @prenom VARCHAR(10),
4     @nom VARCHAR(10),
5     @store_id INT = 1
6   )
7 RETURNS @val TABLE
8   (
9     prenom VARCHAR(10),
10    nom VARCHAR(10)
11  )
12 AS
13 BEGIN
14   INSERT INTO @val (prenom, nom)
15   SELECT first_name, last_name

```

³⁰ <https://learn.microsoft.com/en-us/sql/t-sql/statements/execute-as-clause-transact-sql?view=sql-server-ver16&tabs=sqlserver>

```

16      FROM customer
17      WHERE first_name LIKE @prenom
18            AND last_name LIKE @nom
19            AND store_id = @store_id;
20
21      RETURN;
22  END;
23  GO
24
25  SELECT prenom, nom FROM dbo.get_nom_store_table('M%', 'M%', 1);

```

Informations sur les fonctions

Vue système	Description
sys.sql_modules ³¹	Affiche des informations sur les fonctions définies par l'utilisateur
sys.parameters ³²	Affiche des informations sur les paramètres définis dans les fonctions définies par l'utilisateur.
sys.sql_expression_dependencies ³³	Affiche les objets sous-jacents référencés par une fonction.

Références

Documentation³⁴

11.2.4. Définition d'une procédure

Procédure

Az Définition

- regroupement d'instructions semblable à une fonction
- nom
- paramètres (optionnels), en entrée ou en sortie
- valeur de retour est un entier indiquant si la procédure s'est bien déroulée

```

1 CREATE [OR ALTER] PROCEDURE procedure_name
2 (
3     { @parameter_name [AS] [type_schema_name.]parameter_data_type [= default] [OUTPUT] }
4     [,...n]
5 )
6 [WITH {RECOMPILE | ENCRYPTION | EXECUTE AS Clause}]
7 AS
8 BEGIN
9     -- Corps de la procédure
10    -- Instructions T-SQL
11 END
12

```

³¹. <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-catalog-views/sys-sql-modules-transact-sql?view=sql-server-ver16>

Consultation d'un nom de client

 Exemple

```

1  CREATE OR ALTER PROCEDURE proc_customer_info
2      @customer_id INT,
3      @full_name VARCHAR(50) OUTPUT
4  AS
5  BEGIN
6      SELECT @full_name = first_name + ' ' + last_name
7      FROM customer
8      WHERE customer_id = @customer_id;
9  END;
10 GO
11
12 DECLARE @full_name VARCHAR(50);
13 EXEC proc_customer_info 1, @full_name OUTPUT;
14 SELECT @full_name AS 'Nom';

```

Options

RECOMPILE : Force la recompilation de la procédure à chaque exécution.

ENCRYPTION : la définition de la fonction ne peut pas être lue

EXECUTE_AS_Clause : la fonction est appelée avec la prérogative indiquée (CALLER | SELF | OWNER | 'user_name'). Référence³⁵

11.2.5. EXECUTE

EXEC ou EXECUTE

Exécute une procédure stockée.

EXEC AS / REVERT

EXEC AS [connexion1] : change le contexte d'exécution avec les prérogatives de la connexion [connexion1]

REVERT : revient à la connexion précédant le EXEC AS

SQL dynamique

Exécute une instruction SQL définie dans une chaîne

```
1 Execute ('use sakila;select title from film where film_id=1')
```

11.2.6. Paramètres d'appel d'une procédure

Définition

- nom
- type de données
- direction (OUTPUT)
- valeur par défaut

³². <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-catalog-views/sys-parameters-transact-sql?view=sql-server-ver16>

Utilisation

Utilisation comme des variables locales.

Les valeurs sont définies lors de l'appel dans l'ordre de déclaration

```

1 CREATE OR ALTER PROCEDURE search_customers
2   @first_name VARCHAR(50) = NULL,
3   @last_name  VARCHAR(50) = NULL,
4   @store_id   INT = 1
5 AS
6 BEGIN
7   SELECT first_name, last_name
8   FROM customer
9   WHERE (first_name LIKE @first_name OR @first_name IS NULL)
10      AND (last_name LIKE @last_name OR @last_name IS NULL)
11      AND store_id = @store_id;
12   RETURN 0
13 END;
14 GO
15
16 DECLARE @etat int;
17 EXEC @etat=search_customers @first_name = 'M%', @last_name = 'M%';
18 PRINT 'Indicateur : ' + CONVERT(VARCHAR(30), @etat);
  
```

valeur par défaut

 Remarque

Il ne faut pas utiliser le mot clé DEFAULT lors de l'appel, contrairement à une fonction

11.2.7. Valeur de retour d'une procédure

Valeur de retour

L'instruction RETURN permet de retourner un entier pour indiquer le déroulement de la procédure (0 : ok, <>0 : erreur)

Les paramètres peuvent avoir un attribut OUTPUT

Informations sur les procédures

Vue système	Description
sys.sql_modules ³⁶	Affiche des informations sur les procédures définies par l'utilisateur
sys.parameters ³⁷	Affiche des informations sur les paramètres définis dans les procédures définies par l'utilisateur.
sys.sql_expression_dependencies sys.dm_sql_referenced_entities sys.dm_sql_referencing_entities	Affiche les objets sous-jacents référencés par une procédure.

Références

Documentation

³³. <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-catalog-views/sys-sql-expression-dependencies-transact-sql?view=sql-server-ver16>

11.3. Les déclencheurs (Trigger)

11.3.1. Les triggers

Intérêt

Réalisations d'action à l'aide d'un langage procédural

Contraintes difficiles ou impossibles à exprimer avec les contraintes SQL

S'appliquent lors de modifications de données ou lors de modification de la base

11.3.2. Trigger DML

Trigger DML

Trigger sur les modifications de données

```
1 CREATE [ OR ALTER ] TRIGGER [ schema_name . ]trigger_name
2 ON { table | view }
3 [ WITH <dml_trigger_option> [ ,...n ] ]
4 { FOR | AFTER | INSTEAD OF }
5 { [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }
6 AS sql_statement
7
8 <dml_trigger_option> ::=
9     [ ENCRYPTION ]
10    [ EXECUTE AS Clause ]
```

Options

ENCRYPTION : la définition de la fonction ne peut pas être lue

EXECUTE_AS_Clause : la fonction est appelée avec la prérogative indiquée (CALLER | SELF | OWNER | 'user_name'). Référence

Spécification de l'opération

INSERT

UPDATE

DELETE

Type de déclenchement

AFTER : l'opération est réalisée et le trigger est exécuté ensuite

INSTEAD OF : le trigger est exécuté à la place des modifications

Pseudo tables contenant les lignes ajoutées ou supprimées

Table	Événement	Description
INSERTED	INSERT ou UPDATE	Nouvelles lignes de la table
DELETED	UPDATE ou DELETE	Anciennes lignes de la table

Copie des enregistrements supprimés de la table actor dans actor_old

 Exemple

```
1 CREATE TRIGGER trg_CopyDeletedRows
2 ON actor
3 FOR DELETE
4 AS
5 BEGIN
6     INSERT INTO actor_old (actor_id, first_name, last_name, last_update)
```

34. <https://learn.microsoft.com/en-us/sql/t-sql/statements/create-function-transact-sql?view=sql-server-ver16>

```

7  SELECT actor_id, first_name, last_name, last_update
8  FROM deleted;
9 END;

```

11.3.3. Trigger DDL

Trigger DDL

Trigger sur les modifications de structure de la base

```

1 CREATE [ OR ALTER ] TRIGGER trigger_name
2 ON { ALL SERVER | DATABASE }
3 [ WITH <ddl_trigger_option> [ ,...n ] ]
4 { FOR | AFTER } { event_type | event_group } [ ,...n ]
5 AS { sql_statement [ ; ] [ ,...n ] | EXTERNAL NAME < method specifier > [ ; ] }
6
7 <ddl_trigger_option> ::=
8   [ ENCRYPTION ]
9   [ EXECUTE AS Clause ]

```

Envoi d'un mail lorsque la structure d'une base est modifiée

[Exemple](#)

```

1 IF EXISTS (
2   SELECT *
3   FROM sys.triggers
4   WHERE name = 'trg_structure_change'
5 )
6 BEGIN
7   DROP TRIGGER trg_structure_change;
8 END;
9
10 CREATE TRIGGER trg_structure_change
11 ON DATABASE
12 FOR DDL_DATABASE_LEVEL_EVENTS
13 AS
14 BEGIN
15   DECLARE @subject NVARCHAR(100) = 'Structure Change in Sakila Database';
16   DECLARE @body NVARCHAR(MAX) = 'The structure of the Sakila database has been
17     modified. Please review the changes.';
17   DECLARE @mail_recipients NVARCHAR(100) = 'admin@acme.com';
18
19   EXEC msdb.dbo.sp_send_dbmail
20     @profile_name = 'YourMailProfile',
21     @recipients = @mail_recipients,
22     @subject = @subject,
23     @body = @body;
24 END;

```

³⁵ <https://learn.microsoft.com/en-us/sql/t-sql/statements/execute-as-clause-transact-sql?view=sql-server-ver16&tabs=sqlserver>

12. Performances

12.1. Performances

Database Tuning Advisor

Tuning d'une requête

Tuning de l'instance

Column Store Index.

mécanisme pour accéder rapidement à des tables de grande taille pour des requêtes analytiques

Columnstore (cf. p.97)

Partitionnement.

exécution sur des disques différents (cf. p.82)

12.2. Tables en mémoire

Caractéristiques

- Tables entièrement stockées en mémoire : accès et opérations très rapides
- Peuvent être persistantes ou non
- Les transactions dans les tables persistantes sont enregistrées dans le journal des transactions
- Index en mémoire
- Pas de verrouillage de page : réduction de la contention
- Modèle de transaction optimiste pour réduire les conflits de verrouillage et améliorer les performances
- Possibilité de compiler les procédures stockées en code natif pour améliorer les performances

Création

- MEMORY_OPTIMIZED
ON -> table en mémoire
- DURABILITY
SCHEMA_AND_DATA : table persistante
SCHEMA_ONLY : table non persistante

```
1 CREATE TABLE MaTable
2 (
3     ID INT PRIMARY KEY NONCLUSTERED,
4     Nom VARCHAR(50),
5     Prenom VARCHAR(50)
6 ) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA);
```

Cas d'utilisation

- applications OLTP à haute fréquence de transactions
- caches de données
- tables de session (ASP.NET)

³⁶. <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-catalog-views/sys-sql-modules-transact-sql?view=sql-server-ver16>

Limitations

- pas d'index cluster
- pas de déclencheurs AFTER
- pas de contraintes CHECK
- pas de colonnes calculées persistantes.

Documentations

Références

Mise en œuvre pratique sur SQL Server 2022

- bug -> option de démarrage : -T12324
- ajout d'un filegroup MEMORY_OPTIMIZED_DATA
- compatibility level >= 130
- MEMORY_OPTIMIZED_ELEVATE_TO_SNAPSHOT=ON
- script

Procédures stockées optimisées

- ne peut utiliser que des tables en mémoire

```
1 CREATE OR ALTER PROCEDURE [dbo].[pm_nom]
2   @id INT = 1,
3   @nomcomplet VARCHAR(30) OUTPUT
4 WITH NATIVE_COMPILATION, SCHEMABINDING
5 AS
6 BEGIN ATOMIC WITH (TRANSACTION ISOLATION LEVEL = SNAPSHOT, LANGUAGE = N'us_english');
7   SELECT @nomcomplet = prenom + ' ' + nom FROM dbo.mclients WHERE id = @id;
8 END;
```

12.3. Query Store

12.3.1. Présentation Query Store

Présentation

Enregistre des informations sur

- plan d'exécution des instructions DML
- statistiques sur les exécutions
- statistiques sur les temps d'attente

Disponibilités

- activation sur une base (pas sur l'instance)
- disponible depuis SQL Server 2016
- activé par défaut en mode READ WRITE sur SQL Server 2022 pour les nouvelles bases

Références

Références

Bonnes pratiques

³⁷ <https://learn.microsoft.com/fr-fr/sql/relational-databases/system-catalog-views/sys-parameters-transact-sql?view=sql-server-ver16>

12.3.2. Mise en œuvre

Activation

Activation sur une base avec SSMS ou Transact SQL

```
1 ALTER DATABASE <database_name>
2 SET QUERY_STORE = ON (OPERATION_MODE = READ_WRITE);
3
4 ALTER DATABASE <database_name>
5 SET QUERY_STORE = ON ( WAIT_STATS_CAPTURE_MODE = ON );
```

Interrogation

3 magasins, accessibles au travers de vues :

- plan store : information sur l'exécution des plans.
- runtime stats store : statistiques sur les exécutions.
- wait stats store : statistiques sur les attentes.

```
1 SELECT Txt.query_text_id, Txt.query_sql_text, Pln.plan_id, Qry.*, RtSt.*
2 FROM sys.query_store_plan AS Pln
3 INNER JOIN sys.query_store_query AS Qry
4     ON Pln.query_id = Qry.query_id
5 INNER JOIN sys.query_store_query_text AS Txt
6     ON Qry.query_text_id = Txt.query_text_id
7 INNER JOIN sys.query_store_runtime_stats RtSt
8     ON Pln.plan_id = RtSt.plan_id;
```

Vues³⁸

Vue	Description
sys.database_query_store_options	Fournit des informations sur les options de stockage des requêtes de la base de données.
sys.query_context_settings	Contient les paramètres de contexte de requête utilisés lors de l'exécution des requêtes.
sys.query_store_plan	Stocke les plans d'exécution des requêtes dans le Query Store.
sys.query_store_query	Contient des informations sur les requêtes stockées dans le Query Store.
sys.query_store_query_text	Stocke le texte des requêtes exécutées dans le Query Store.
sys.query_store_runtime_stats	Fournit des statistiques d'exécution en temps réel pour les requêtes stockées dans le Query Store.
sys.query_store_wait_stats	Stocke les statistiques d'attente pour les requêtes exécutées dans le Query Store.
sys.query_store_runtime_stats_interval	Fournit des informations sur les intervalles de temps utilisés pour les statistiques d'exécution en temps réel.
sys.database_query_store_internal_state	Contient des informations sur l'état interne du Query Store de la base de données.

³⁸. <https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/query-store-catalog-view-s-transact-sql?view=sql-server-ver16>

Procédures stockées³⁹

Procédure stockée	Description
sp_query_store_flush_db	Vide les données du Query Store de la base de données en mémoire vers le disque. Cela peut être utile pour s'assurer que toutes les données sont enregistrées avant une opération de maintenance ou un redémarrage du serveur.
sp_query_store_reset_exec_stats	Réinitialise les statistiques d'exécution pour une requête spécifique dans le Query Store. Cela peut être utilisé pour effacer les statistiques historiques et recommencer la collecte de nouvelles statistiques.
sp_query_store_force_plan	Force l'utilisation d'un plan d'exécution spécifique pour une requête donnée. Cela peut être utile pour optimiser les performances en utilisant un plan d'exécution connu pour être efficace.
sp_query_store_unforce_plan	Annule l'application forcée d'un plan d'exécution pour une requête donnée. Cela permet au moteur de base de données de choisir automatiquement le meilleur plan d'exécution.
sp_query_store_remove_plan	Supprime un plan d'exécution spécifique du Query Store. Cela peut être utilisé pour nettoyer les plans d'exécution obsolètes ou inutilisés.
sp_query_store_remove_query	Supprime une requête spécifique et tous ses plans d'exécution associés du Query Store. Cela peut être utilisé pour nettoyer les requêtes obsolètes ou inutilisées.
sp_query_store_clear_message_queues	Efface les files d'attente de messages du Query Store. Cela peut être utilisé pour résoudre les problèmes de file d'attente ou pour réinitialiser l'état des messages.
sp_query_store_consistency_check	Effectue une vérification de la cohérence des données du Query Store. Cela peut être utilisé pour s'assurer que les données du Query Store sont intègres et cohérentes.

Utilisation du Query Store

interrogation de query store

- dernières requêtes exécutées
- nombre d'exécution
- requêtes avec plusieurs plans
- ...

spécification d'un plan à exécuter : sp_query_store_force_plan

Surveillance⁴⁰

Optimisation

³⁹. <https://learn.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/query-store-stored-procedures-transact-sql?view=sql-server-ver16>

```

1 SELECT Txt.query_text_id, Txt.query_sql_text, Pln.plan_id, Qry.*, RtSt.*
2 FROM sys.query_store_plan AS Pln
3 INNER JOIN sys.query_store_query AS Qry
4     ON Pln.query_id = Qry.query_id
5 INNER JOIN sys.query_store_query_text AS Txt
6     ON Qry.query_text_id = Txt.query_text_id
7 INNER JOIN sys.query_store_runtime_stats RtSt
8     ON Pln.plan_id = RtSt.plan_id;

```

Automatic Tuning

Activation de Query Store permet d'activer le *Tuning Automatique* (cf. p.108)

Query Store Hints

Possibilité de spécifier des hints sans modifier le code de l'application

Références⁴¹

```

1 EXEC sys.sp_query_store_set_hints
2     @query_id = 123,
3     @query_hints = N'OPTION(RECOMPILE)';

```

12.4. Intelligent Query Processing (IQP)

12.4.1. Présentation

Intelligent Query Processing (IQP)

Az Définition

ensemble de fonctionnalités dans SQL Server qui permettent d'optimiser automatiquement les performances des requêtes sans modification du code.

Objectif

Optimiser les performances des requêtes sans intervention manuelle

(il suffit d'augmenter le niveau d'une base de données)

```

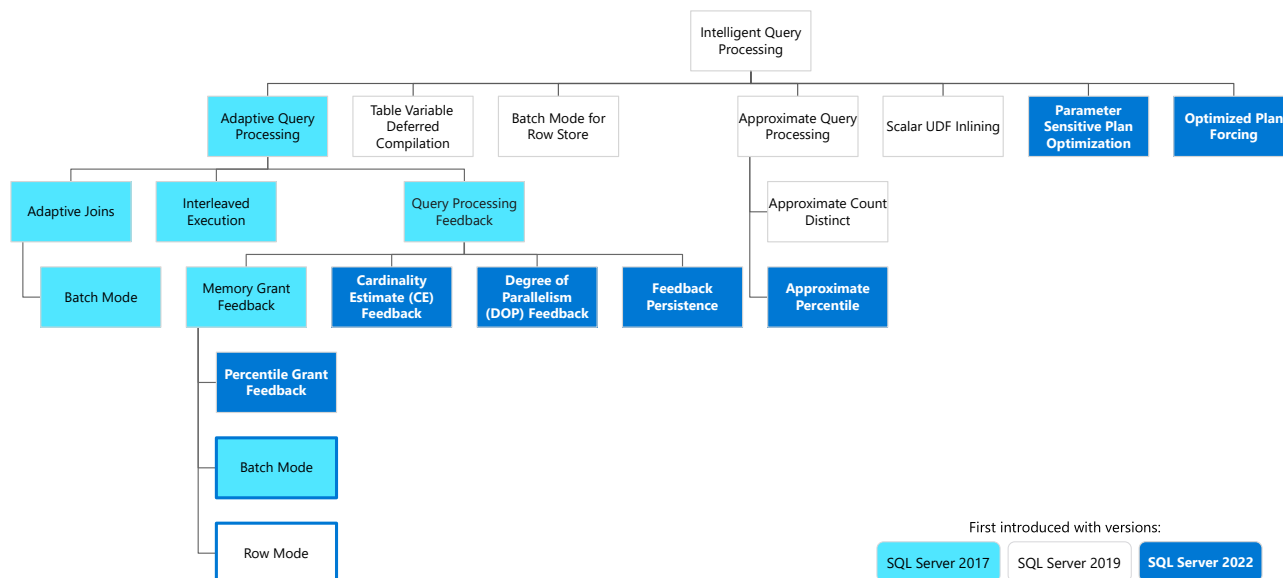
1 ALTER DATABASE [WideWorldImportersDW] SET COMPATIBILITY_LEVEL = 160;

```

⁴⁰. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/monitoring-performance-by-using-the-query-store?view=sql-server-ver17>

Historique

Introduit à partir de SQL Server 2017, amélioré dans SQL Server 2019 et versions ultérieures.



Documentation

Références⁴²

Détails

12.4.2. Parameter Sensitive Plan (PSP)

Parameter Sensitive Plan

Az Définition

Création automatique de plusieurs plans d'exécution actifs en cache pour une seule instruction paramétrée.

Plans d'exécution adaptés à différentes tailles de données selon les valeurs de paramètres fournies à l'exécution.

Architecture

- Dispatcher Plan (Plan répartiteur)
 - plan contenant l'expression dispatcher mis en cache pour la requête originale.
 - collection des prédicats sélectionnés avec des détails supplémentaires comme les valeurs de limites hautes et basses utilisées pour diviser les valeurs de paramètres en différentes plages.
- Dispatcher Expression
 - Évalue la cardinalité des prédicats basée sur les valeurs de paramètres à l'exécution
 - route l'exécution vers différentes variantes de requêtes.
- Query Variant (Variante de requête)
 - requêtes enfants créées par le dispatcher plan pour traiter une plage de données s

41. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/query-store-hints?view=sql-server-ver17>

- Predicate Cardinality Range

plage de cardinalité évaluée par le Dispatcher Plan en fonction des valeurs de paramètres
typiquement trois plages de cardinalité basse, moyenne et haute

Fonctionnement

- Phase de compilation initiale

calcul des histogrammes de statistiques de colonnes pour identifier les distributions non uniformes

évaluation des prédicats paramétrés les plus à risque (au maximum 3 pour ne pas saturer les ressources)

- Phase d'exécution

Évaluation de l'expression dispatcher pour l'ensemble donné de paramètres afin de calculer la plage de cardinalité

Associe les plages à des variantes de requêtes spécifiques et compile/exécute les variantes

Limitations

- uniquement avec les prédicats d'égalité
- nombre de variantes de plans uniques par dispatcher stockées dans le plan cache est limité pour éviter le gonflement du cache. Le seuil interne n'est pas documenté.
- Si le parameter sniffing est désactivé par Trace Flag 4136, la configuration `PARAMETER_SNIFFING` au niveau base de données, ou le hint `USE HINT('DISABLE_PARAMETER_SNIFFING')`, PSP est désactivé.

Surveillance

Events

- `parameter_sensitive_plan_optimization_skipped_reason` : Raisons pour lesquelles PSP a été ignoré
- `parameter_sensitive_plan_optimization` : Quand une requête utilise PSP
- `query_with_parameter_sensitivity` : Détails sur les prédicats intéressants trouvés

vues

- `sys.query_store_query_variant`
- `sys.query_store_plan`

```
1 -- Nouvelle vue Query Store pour les variantes
2 SELECT * FROM sys.query_store_query_variant;
3
4 -- Vérifier les plans avec PSP
5 SELECT
6     query_id,
7     plan_id,
8     is_forced_plan,
9     query_plan
10 FROM sys.query_store_plan
11 WHERE query_plan LIKE '%PLAN PER VALUE%';
```

42. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing?view=sql-server-ver16>

12.4.3. Batch Mode sur Rowstore

Présentation

- Introduit la capacité de *traitement en batch* (cf. p.108) sur les tables rowstore (non-columnstore).
- Améliore considérablement les performances sur les tables traditionnelles sans index columnstore.

Avantage

Traitement des requêtes plus rapide même sur des tables classiques rowstore.

Documentation

Références⁴³

12.4.4. Adaptive Joins

Présentation

- Permet au plan d'exécution de la requête de s'adapter dynamiquement en fonction de la taille des données.
- SQL Server choisit entre Nested Loops et Hash Joins à l'exécution.

Remarque : la requête doit s'exécuter en *mode batch* (cf. p.108)

Avantage

Amélioration de la performance sur des données avec des volumes variables

Documentation

références⁴⁴

12.4.5. Interleaved Execution for Multi-Statement Table-Valued Functions (MSTVF)

Présentation

- Améliore le plan d'exécution pour les fonctions MSTVF, qui auparavant avaient des estimations de cardinalité incorrectes.
- Maintenant, le plan est ajusté après l'exécution partielle.

```
1 CREATE FUNCTION dbo.GetSales(@ProductID INT)
2 RETURNS @Sales TABLE (SalesOrderID INT)
3 AS
4 BEGIN
5     INSERT INTO @Sales
6     SELECT SalesOrderID
7     FROM Sales.SalesOrderDetail
8     WHERE ProductID = @ProductID;
9     RETURN;
10 END;
```

Avantages

Amélioration des estimations pour mieux optimiser le plan de requête.

Documentation

Références

⁴³. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing-details?view=sql-server-ver16#batch-mode-on-rowstore>

12.4.6. Memory Feedback

Memory Grant Feedback

Utilisation des exécutions précédentes pour optimiser automatiquement l'allocation mémoire des requêtes futures.

Trois types de feedback sont disponibles :

- Row Mode Memory Grant Feedback : Ajuste les allocations mémoire pour les opérations en mode ligne
- Batch Mode Memory Grant Feedback : Optimise les allocations pour les opérations en mode batch
- Percentile-based Memory Grant Feedback : Utilise des statistiques percentiles pour des allocations plus précises

Fonctionnement

- Ajuste dynamiquement les allocations de mémoire pour une requête après chaque exécution.
- Si la mémoire allouée était insuffisante ou excessive, elle est ajustée pour les exécutions futures.

12.4.7. Analyse de l'estimation automatique des cardinalités et des optimisations internes

Cardinality Estimation Feedback (CEF)

détecte les requêtes où l'estimation du nombre de lignes et le nombre de ligne sont très différents
applique un autre plan (si disponible) pour les exécutions suivantes

Référence⁴⁵

Degree of Parallelism (DOP)

ajuste automatiquement le niveau de parallélisme des requêtes en fonction de de l'historique

Référence

Optimized Plan Forcing (OPF)

Lorsqu'un plan est forcé via Query Store, SQL Server réapplique les règles d'optimisation pour ré-optimiser le plan forcé.

Cela permet de conserver la stabilité du plan tout en s'assurant qu'il reste adapté aux données actuelles.

Résultat : moins de régressions de performance liées à des plans forcés obsolètes.

Référence

12.4.8. Approximate Query Processing

Présentation

Permet des calculs approximatifs pour des agrégats comme `COUNT DISTINCT`, améliorant les performances tout en maintenant une précision acceptable.

```
1 SELECT APPROX_COUNT_DISTINCT(CustomerID)
2 FROM Sales.SalesOrderHeader;
```

Avantages

Plus rapide pour de grands ensembles de données tout en conservant une précision acceptable.

Documentation

Références

44. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/joins?view=sql-server-ver16#adaptive>

12.4.9. Table Variable Deferred Compilation

Présentation

- Dans les versions précédentes, les variables de table avaient des estimations de cardinalité par défaut.
- Désormais, SQL Server attend d'avoir des informations concrètes sur les données pour optimiser le plan.

```
1 DECLARE @Sales TABLE (ProductID INT, Quantity INT);
2 INSERT INTO @Sales
3 SELECT ProductID, OrderQty
4 FROM Sales.SalesOrderDetail;
5 SELECT * FROM @Sales;
```

Avantages

Optimisation des plans de requêtes utilisant des variables de table

Documentation

Références⁴⁶

12.4.10. Scalar UDF Inlining

Présentation

- Inline les fonctions scalaires pour éliminer le surcoût de leur exécution comme des "boîtes noires".
- SQL Server transforme les UDF scalaires en requêtes équivalentes intégrées dans le plan de requête principal.

Avantages

Améliore les performances des fonctions scalaires en réduisant les aller-retours.

Documentation

Références

⁴⁵. <https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing-cardinality-estimation-feedback?view=sql-server-ver17>

13. Introduction durcissement

13.1. Confidentialité des données

Risques sur les données

- accès aux données transitant sur le réseau
- accès aux données sur disque
- accès aux données sur les sauvegardes

13.2. Connexion TLS à une instance SQL Server

Etapes

- installer un certificat sur la machine Windows
- installer le certificat dans l'instance avec SQL Configuration Manager

Référence⁴⁷

Utilisation d'une clé autosignée

 Exemple

1. Depuis un powershell en mode administrateur

```
1 $cert = New-SelfSignedCertificate -DnsName "DESKTOP-94LFLQR" -CertStoreLocation  
"Cert:\LocalMachine\My" -KeyUsage KeyEncipherment, DataEncipherment -KeySpec KeyExchange
```

1. Lancer certlm.msc

2. Aller dans Personnel, certificat

3. Exporter le certificat avec la clé privée

1. Lancer SQL Server Configuration Manager

2. Sélectionner services Réseaux

3. Aller dans l'onglet propriétés et importer le certificat

variante (si SQL Server Configuration Manager plante)

Dans certlm.msc

Clic droit > All Tasks > Manage Private Keys.

Ajouter le compte de service SQL Server (NT SERVICE\MSSQLSERVER) avec Read.

Dans SQL Server Configuration Manager, service Réseaux, Sélectionner le certificat

13.3. Chiffrement des données sur disque

Transparent data encryption (TDE)

Chiffrement transparent des données sur le disque

Permet de se prémunir d'une fuite de données si quelqu'un peut accéder aux fichiers de données

Peu d'impact sur la taille

Disponible dans les éditions Enterprise, Standard et Web depuis 2022

⁴⁶ <https://learn.microsoft.com/en-us/sql/relational-databases/performance/intelligent-query-processing-details?view=sql-server-ver16#table-variable-deferred-compilation>

Référence⁴⁸**Mise en œuvre**

- création d'une clé principale de chiffrement dans master
- création d'un certificat dans master
- création d'une clé de chiffrement dans la base à chiffrer
- activation du chiffrement de la base

```

1 USE master;
2 GO
3 CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'MotComplicue';
4 GO
5 CREATE CERTIFICATE TDECertificate
6 WITH SUBJECT = 'TDE Certificate';
7 GO
8
9 USE demo_TDE;
10 GO
11 CREATE DATABASE ENCRYPTION KEY
12 WITH ALGORITHM = AES_256
13 ENCRYPTION BY SERVER CERTIFICATE TDECertificate;
14 GO
15
16 ALTER DATABASE demo_TDE
17 SET ENCRYPTION ON;
18 GO

```

chiffrement automatique de tempdb Remarque

Si une base met en œuvre le chiffrement, tempdb l'est automatiquement chiffrée.

Vérification de l'état du chiffrement

```

1 SELECT
2     db.name,
3     db.is_encrypted,
4     dm.encrypted_state,
5     dm.percent_complete,
6     dm.key_algorithm,
7     dm.key_length
8 FROM
9     sys.databases db
10    LEFT JOIN sys.dm_database_encryption_keys dm
11    ON db.database_id = dm.database_id;

```

clé principale et certificat de master Attention

- il ne faut pas supprimer la clé principale ni le certificat dans master
- il faut les sauvegarder (Backup certificate)

vérification : si pvt_key_last_backup_date est NULL, le certificat n'a pas été sauvegardé

⁴⁸. https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/configure-sql-server-encryption?utm_source=pocket_shared&view=sql-server-ver16

```

1 use master
2 go
3 SELECT pvt_key_last_backup_date,
4         Db_name(dek.database_id) AS encrypteddatabase,
5         c.name AS Certificate_Name
6 FROM sys.certificates AS c
7     INNER JOIN sys.dm_database_encryption_keys AS dek
8         ON c.thumbprint = dek.encryptor_thumbprint;

1 pvt_key_last_backup_date encrypteddatabase
2 Certificate_Name
2 -----
-----
-----
3 NULL demo_TDE

1 BACKUP CERTIFICATE TDECertificate TO FILE = 'C:\backup\certificat_TDE.pfx'
2 WITH
3     FORMAT = 'PFX',
4     PRIVATE KEY
5     (
6     ENCRYPTION BY PASSWORD = 'MotComplicue',
7     ALGORITHM = 'AES_256'
8     )

```

13.4. Transparent Data Encryption

Rendre les données inaccessibles à l'administrateur

- chiffrement des données dans l'application, avant l'envoi
- connexion spécifique

13.5. Ledger

Tables ledger (depuis SQL Server 2022)

- Suivi des modifications effectuées sur des données
- table versionnée sous jacente

Tables versionnées (cf. p.83)

Création

Utilisation de l'assistant

```

1 IF OBJECT_ID('dbo.demo_ledger', 'U') IS NOT NULL
2     DROP TABLE dbo.demo_ledger_table
3 GO
4
5 --Create updatable ledger table.
6 CREATE TABLE [dbo].[demo_ledger_table]
7 (
8     --The table must contain at least one user-defined column.
9     id int,
10    nom nchar(20),
11
12    --Ledger tables require GENERATED ALWAYS columns, which can be explicitly defined or
13    auto-generated during table creation
14    ledger_start_transaction_id BIGINT GENERATED ALWAYS AS TRANSACTION_ID START,
15    ledger_end_transaction_id BIGINT GENERATED ALWAYS AS TRANSACTION_ID END,

```

48. <https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption?view=sql-server-ver16>

```

15 ledger_start_sequence_number BIGINT GENERATED ALWAYS AS SEQUENCE_NUMBER START,
16 ledger_end_sequence_number BIGINT GENERATED ALWAYS AS SEQUENCE_NUMBER END
17 )
18 WITH
19 (
20     --Set SYSTEM_VERSIONING to ON
21     SYSTEM_VERSIONING = ON
22     (
23         --Option to specify the name of the ledger history table. If not supplied, the
24         history table will be created with the
25         --default name format: [<schema_name>].
26         [MSSQL_LedgerHistoryFor_<ledger_table_objectid>]
27         HISTORY_TABLE = [dbo].[demo_ledger_history]
28     ),
29     --Set LEDGER = ON. This is optional if this is a ledger database
30     LEDGER = ON
31     (
32         --Option to specify the name of the ledger view. If not supplied, the view will be
33         created with the
34         --default name format: [<schema_name>].[<ledger_table_name>_Ledger]
35         LEDGER_VIEW = [dbo].[demo_ledger_view]
36     )
37     (
38         --Options to specify ledger view default column names
39         TRANSACTION_ID_COLUMN_NAME = ledger_transaction_id,
40         SEQUENCE_NUMBER_COLUMN_NAME = ledger_sequence_number,
41         OPERATION_TYPE_COLUMN_NAME = ledger_operation_type,
42         OPERATION_TYPE_DESC_COLUMN_NAME = ledger_operation_type_desc
43     )
44 )
45 )

```

Références

Documentation⁴⁹

⁴⁹. <https://learn.microsoft.com/en-us/sql/relational-databases/security/ledger/ledger-overview?view=sql-server-ver16>

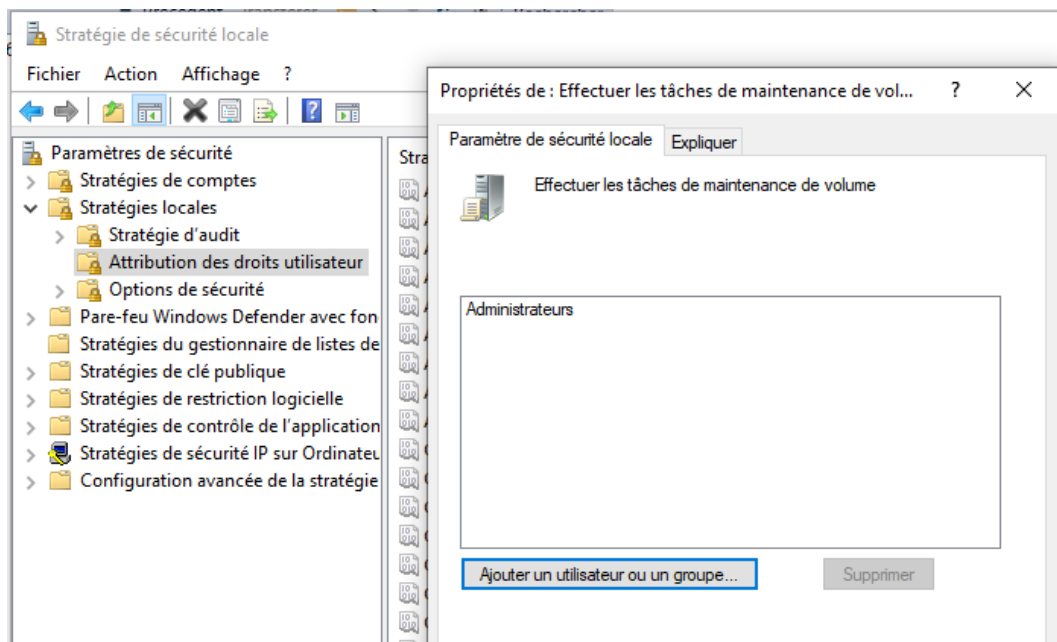
14. Annexes

14.1. IFI (Instant File Initialization)

IFI (Instant File Initialization)

- Par défaut, SQL Server met à zéro les fichiers qu'il crée ou étend (mdf, ndf, ldf).
- Si IFI est activé, la mise à zéro n'est pas effectuée et SQL Server écrit directement dans l'espace alloué, donc accélération des créations et expansions de fichiers.
- L'activation se fait automatiquement si le compte de service dispose du privilège SE_MANAGE_VOLUME_NAME.

Référence⁵⁰



14.2. DBCC (Database Console Commands)

DBCC

Az Définition

ensemble de commandes utilitaires dans SQL Server permettant de vérifier, maintenir et dépanner les bases de données.

Syntaxe générale : DBCC nom_commande [(paramètres)] [WITH options]

Référence⁵¹

Commandes de vérification

DBCC CHECKDB - Vérifie l'intégrité complète d'une base de données

DBCC CHECKTABLE - Vérifie l'intégrité d'une table spécifique

DBCC CHECKALLOC - Vérifie la cohérence des structures d'allocation

```
1 DBCC CHECKDB( 'AdventureWorks2022' )
```

⁵⁰ <https://learn.microsoft.com/en-us/sql/relational-databases/databases/database-instant-file-initialization?view=sql-server-ver16>

```

1 DBCC results for 'AdventureWorks2022'.
2 Service Broker Msg 9675, State 1: Message Types analyzed: 14.
3 Service Broker Msg 9676, State 1: Service Contracts analyzed: 6.
4 Service Broker Msg 9667, State 1: Services analyzed: 3.
5 Service Broker Msg 9668, State 1: Service Queues analyzed: 3.
6 Service Broker Msg 9669, State 1: Conversation Endpoints analyzed: 0.
7 Service Broker Msg 9674, State 1: Conversation Groups analyzed: 0.
8 Service Broker Msg 9670, State 1: Remote Service Bindings analyzed: 0.
9 Service Broker Msg 9605, State 1: Conversation Priorities analyzed: 0.
10 DBCC results for 'sys.sysrscscols'.
11 There are 2387 rows in 27 pages for object "sys.sysrscscols".
12 DBCC results for 'sys.sysrowsets'.
13 There are 387 rows in 5 pages for object "sys.sysrowsets".
14
15 ....
16
17 There are 19972 rows in 3808 pages for object "Person.Person".
18 DBCC results for 'sys.filetable_updates_2105058535'.
19 There are 0 rows in 0 pages for object "sys.filetable_updates_2105058535".
20 DBCC results for 'Sales.SalesReason'.
21 There are 10 rows in 1 pages for object "Sales.SalesReason".
22 CHECKDB found 0 allocation errors and 0 consistency errors in database 'AdventureWorks2022'.
23 DBCC execution completed. If DBCC printed error messages, contact your system administrator.

```

Commandes de maintenance

DBCC SHRINKDATABASE - Réduit la taille d'une base de données

DBCC SHRINKFILE - Réduit la taille d'un fichier spécifique

```
1 DBCC SHRINKDATABASE('AdventureWorks2022', 10)
```

```
1 Cannot shrink log file 2 (AdventureWorks2022_log) because requested size (9104KB) is larger
  than the start of the last logical log file.
```

2 DbId	FileId	CurrentSize	MinimumSize	UsedPages	EstimatedPages
3 -----	-----	-----	-----	-----	-----
4 9	1	28688	1024	25808	25808
5 9	2	3072	1024	3072	1024

```
6
7 (2 lignes affectées)
8
```

```
9 DBCC execution completed. If DBCC printed error messages, contact your system administrator.
```

Commandes d'information

DBCC SHOW_STATISTICS - Affiche les statistiques d'un index

DBCC OPENTRAN - Montre les transactions ouvertes

```
1 DBCC OPENTRAN
```

Commandes de nettoyage

DBCC FREEPROCCACHE - Vide le cache des plans d'exécution

DBCC DROPCLEANBUFFERS - Vide le cache des données

14.3. Clichés instantanés

Présentation

Vue d'une base à un instant donné

Lecture seule

Lié à la base initiale

Utile pour des tests ou des analyses

⁵¹. <https://learn.microsoft.com/en-us/sql/t-sql/database-console-commands/dbcc-transact-sql?view=sql-server-ver17>

Fonctionnement

- Création immédiate, sans copie de données
- Copie des données dans la base cliché lorsqu'il y a une modification des données dans la base initiale

Création

Création uniquement par instruction SQL

```
1 CREATE DATABASE [northwind_1912] ON
2 ( NAME = N'Northwind',
3 FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL.2\MSSQL\Data\northwnd_1912.mdf' )
4 AS SNAPSHOT OF [northwind]
5
```

Nombre de fichiers

⚠ Attention

Il faut indiquer autant de fichiers au moment de la création qu'il y a de fichiers dans la définition, sans prendre en compte les filegroup

```
1 USE [master]
2 GO
3
4
5 CREATE DATABASE [annuaire_17h]
6 ON
7 ( NAME = N'annuaire', FILENAME = N'C:\donnee1\annuaire.mdf' ),
8 ( NAME = N'annuaire1', FILENAME = N'C:\donnee1\annuaire1.ndf' ),
9 ( NAME = N'annuaire2', FILENAME = N'C:\donnee1\annuaire2.ndf' ),
10 ( NAME = N'annuaire3', FILENAME = N'C:\donnee1\annuaire3.ndf' ),
11 ( NAME = N'annuaire4', FILENAME = N'C:\donnee1\annuaire4.ndf' )
12 AS snapshot of annuaire
13 GO
```

Utilisation

- accès dans « Database Snapshot »
- *restauration* (cf. p.38)

14.4. Partitionnement d'une table

Présentation

Répartition des données sur plusieurs fichiers

Intérêts :

- répartition des données sur plusieurs disques (gestion de taille, parallélisation)
- amélioration des requêtes avec un filtre ou une jointure sur la clé de partition

Mise en œuvre

- Définir de groupes de fichier pour stocker les données
- Définir une fonction permettant de spécifier où ranger une ligne

```
1 REATE PARTITION FUNCTION FonctionPlage (datetime)
2 AS RANGE LEFT FOR VALUES ('2019-01-01 00:00', '2022-01-01 00:00');
3
4 CREATE PARTITION SCHEME MonSchemaPartition
5 AS PARTITION FonctionPlage
6 TO (fg1, fg2, fg3);
7
```

```

8 CREATE TABLE essai (
9 [id] [int] NOT NULL,
10 [date_achat] [datetime] )
11 ON monSchemaPartition ( date_achat );
12
13 insert into essai values (1,'2000-15-08 00:00');
14 insert into essai values (1,'2021-15-08 00:00');
15 insert into essai values (1,'2025-15-08 00:00');
16
17 select * from essai where date_achat between '2000-01-01 00:00' and '2018-01-01 00:00';

```

Assistant dans SSMS

 Remarque

SSMS propose un assistant pour partitionner une table existante

```

1 USE [AdventureWorks2022]
2 GO
3 BEGIN TRANSACTION
4 CREATE PARTITION FUNCTION [Fpart](datetime) AS RANGE LEFT FOR VALUES (N'2012-01-
    01T00:00:00', N'2013-01-01T00:00:00')
5 CREATE PARTITION SCHEME [SCHEMA_FPART] AS PARTITION [Fpart] TO ([FG1], [FG2], [FG3])
6
7 ALTER TABLE [Sales].[SalesOrderDetail] DROP CONSTRAINT
    [PK_SalesOrderDetail_SalesOrderID_SalesOrderDetailID] WITH ( ONLINE = OFF )
8 ALTER TABLE [Sales].[SalesOrderDetail] ADD CONSTRAINT
    [PK_SalesOrderDetail_SalesOrderID_SalesOrderDetailID] PRIMARY KEY NONCLUSTERED
9 (
10 [SalesOrderID] ASC,
11 [SalesOrderDetailID] ASC
12 )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF,
    IGNORE_DUP_KEY = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
    OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
13
14
15 CREATE CLUSTERED INDEX [ClusteredIndex_on_SCHEMA_FPART_638803348246786621] ON [Sales].
    [SalesOrderDetail]
16 (
17 [ModifiedDate]
18 )WITH (SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, ONLINE = OFF) ON [SCHEMA_FPART]
    ([ModifiedDate])
19 DROP INDEX [ClusteredIndex_on_SCHEMA_FPART_638803348246786621] ON [Sales].
    [SalesOrderDetail]
20 COMMIT TRANSACTION

```

14.5. Tables versionnées (ou tables temporelles)

Présentation

- deux colonnes supplémentaires : date de début de validité, date de fin de validité, de type DATETIME2
- une table historique créée automatiquement
- quand un enregistrement est modifié (UPDATE), la ligne initiale est placée dans la table historique, SQL Server met à jour la date de fin de validité et crée une nouvelle ligne dans la table versionnée
- quand une nouvelle ligne est créée (INSERT ou UPDATE), SQL Server met à jour automatiquement la date de début de validité et assigne la valeur '9999-12-31' à la date de fin de validité.
- quand un enregistrement est supprimé, la ligne initiale est placée dans la table historique et SQL Server met à jour la date de fin de validité

Référence⁵²

```

1 USE demo_TDE
2 GO
3
4 BEGIN
5     --If table is system-versioned, SYSTEM_VERSIONING must be set to OFF first
6     IF ((SELECT temporal_type FROM sys.tables WHERE object_id =
OBJECT_ID('dbo.table_versionnee', 'U')) = 2)
7         BEGIN
8             ALTER TABLE [dbo].[table_versionnee] SET (SYSTEM_VERSIONING = OFF)
9         END
10    DROP TABLE IF EXISTS [dbo].[table_versionnee]
11 END
12 GO
13
14 --Create system-versioned temporal table. It must have primary key and two datetime2 columns
that are part of SYSTEM_TIME period definition
15 CREATE TABLE [dbo].[table_versionnee]
16 (
17     id int NOT NULL,
18     nom char(50) NULL,
19
20     --Period columns and PERIOD FOR SYSTEM_TIME definition
21     SysStartTime datetime2(7) GENERATED ALWAYS AS ROW START NOT NULL ,
22     SysEndTime datetime2(7) GENERATED ALWAYS AS ROW END NOT NULL ,
23     PERIOD FOR SYSTEM_TIME(SysStartTime,SysEndTime),
24
25     --Primary key definition
26     CONSTRAINT PK_sampletable PRIMARY KEY (id)
27 )
28 WITH
29 (
30     --Set SYSTEM_VERSIONING to ON and provide reference to HISTORY_TABLE.
31     SYSTEM_VERSIONING = ON
32     (
33         --If HISTORY_TABLE does not exists, default table will be created.
34         HISTORY_TABLE = [dbo].[table_versionnee_history],
35         --Specifies whether data consistency check will be performed across current and
history tables (default is ON)
36         DATA_CONSISTENCY_CHECK = ON
37     )
38 )
39 GO
40

```

Requêtes

```
1 select * from table_versionnee;
```

1 id	nom	SysStartTime	SysEndTime
2 -----	-----	-----	-----
3 1	Marie Curie	2025-04-09 13:51:54.3640661	9999-12-31 23:59:59.9999999
4 2	Irène Jolliot Curie	2025-04-09 13:52:31.1297110	9999-12-31 23:59:59.9999999
5 3	Albert Einstein	2025-04-09 13:55:59.0205350	9999-12-31 23:59:59.9999999

```
1 select * from table_versionnee_history;
```

1 id	nom	SysStartTime	SysEndTime
------	-----	--------------	------------

2 -----	-----	-----	-----
3 -----			

```
4 (0 lignes affectées)
```

52. https://learn.microsoft.com/en-us/sql/relational-databases/tables/temporal-tables?utm_source=pocket_shared&view=sql-server-ver16

```

1 delete from table_versionnee where id=1;
2 update table_versionnee set nom='Professeur Albert Einstein' where id=3;
3 select * from table_versionnee;
4 select * from table_versionnee_history;

```

1 id	nom	SysStartTime	SysEndTime
2	-----	-----	-----
3 2	Irène Jolliot Curie	2025-04-09 13:52:31.1297110	9999-12-31 23:59:59.9999999
4 3	Professeur Albert Einstein	2025-04-09 14:09:47.4841882	9999-12-31 23:59:59.9999999

5
6 (2 lignes affectées)

8 id	nom	SysStartTime	SysEndTime
9	-----	-----	-----
10 1	Marie Curie	2025-04-09 13:51:54.3640661	2025-04-09 14:09:47.4841882
11 3	Albert Einstein	2025-04-09 13:55:59.0205350	2025-04-09 14:09:47.4841882

12
13 (2 lignes affectées)

clause FOR SYSTEM TIME

clause pour interroger la table

(cf. *clause_for_system_time.png*) (cf. p.105)

```

1 select * from table_versionnee
2 for system_time
3     BETWEEN '2025-04-09 00:00:00.0000000' AND '2025-04-09 13:55:00.0000000';

```

1 id	nom	SysStartTime	SysEndTime
2	-----	-----	-----
3 2	Irène Jolliot Curie	2025-04-09 13:52:31.1297110	9999-12-31 23:59:59.9999999
4 1	Marie Curie	2025-04-09 13:51:54.3640661	2025-04-09 14:09:47.4841882

Taille d'une table avec historique

⚠ Attention

- Une table avec historique peut grossir indéfiniment ...
- Il faut prévoir un nettoyage
- le paramètre RETENTION_PERIOD permet de définir un nettoyage automatique

14.6. Importation de données

Insertion et importation de données.

Méthode	Description	Importation	Exportation
Utilitaire bcp ⁵³	Utilitaire en ligne de commande (Bcp.exe) qui exporte et importe en bloc des données et génère des fichiers de format.	Oui	Oui

⁵³. <https://docs.microsoft.com/fr-fr/sql/relational-databases/import-export/import-and-export-bulk-data-by-using-the-bcp-utility-sql-server?view=sql-server-ver15>

Méthode	Description	Importation	Exportation
Instruction BULK INSERT ⁵⁴	Instruction Transact-SQL qui importe des données directement d'un fichier de données dans une table de base de données ou une vue non partitionnée.	Oui	Non
instruction INSERT ... instruction SELECT * FROM OPENROWSET(BULK...) ⁵⁵	Instruction Transact-SQL qui utilise le fournisseur d'ensembles de lignes OPENROWSET pour importer en bloc des données dans une table SQL Server en spécifiant la fonction OPENROWSET(BULK...) afin de sélectionner des données dans une instruction INSERT.	Oui	Non
Assistant Importation et Exportation SQL Server ⁵⁶	L'Assistant crée des packages simples qui importent et exportent des données dans plusieurs formats de données courants (bases de données, feuilles de calcul, fichiers texte, etc.).	Oui	Oui

<https://docs.microsoft.com/fr-fr/sql/relational-databases/import-export/bulk-import-and-export-of-data-sql-server?view=sql-server-ver15>

14.7. Plan d'exécution

14.7.1. Traitement d'une requête

Analyse

- Parsing
- Analyse Syntaxique

Binding et Résolution des Objets

- Vérification de l'existence des objets (tables, colonnes)
- Résolution des permissions
- Création de l'arbre de requête interne

Optimisation

- Choix du plan d'exécution optimal : Cost-Based Optimizer (CBO)
- Évaluation des statistiques
- Estimation des coûts (CPU, I/O, mémoire)

Compilation et Mise en Cache

- Génération du plan d'exécution
- Stockage dans le plan cache
- Réutilisation pour les requêtes similaires

⁵⁴. <https://docs.microsoft.com/fr-fr/sql/relational-databases/import-export/import-bulk-data-by-using-bulk-insert-or-openrowset-bulk-sql-server?view=sql-server-ver15>

Exécution

- Allocation des ressources (mémoire, workers)
- Exécution selon le plan choisi
- Gestion des locks et de la concurrence

14.7.2. Plan d'exécution

Visualisation des plans d'exécution estimés

paramètre SHOWPLAN_ALL

```
1 use AdventureWorks2022;
2 GO
3 SET SHOWPLAN_ALL ON;
4 GO
5 select jobTitle, loginID from HumanResources.Employee;
6 GO
7 SET SHOWPLAN_ALL OFF;
8 GO
```

1 StmtText	StmtId	NodeId	Parent	PhysicalOp
LogicalOp	Argument	RowSize	DefinedValues	
EstimateRows	EstimateIO	EstimateCPU	AvgRowSize	TotalSubtreeCost
Warnings	Type	Parallel	EstimateExecutions	OutputList
2				
3	select jobTitle, loginID from	1	0	NULL
NULL	1	NULL	0,008045444	NULL
290	NULL	0	NULL	NULL
NULL	SELECT	0	NULL	NULL
4	--Clustered Index Scan(OBJE 1	2	1	Clustered Index Scan
Clustered Index Scan	OBJECT:([AdventureWorks2022].[[AdventureWorks2022].[HumanRes			
290	0,007569444	0,000476	319	0,008045444
[HumanRes NULL	PLAN_ROW	0	1	[AdventureWorks2022].
5				

Visualisation des plans d'exécution réels

SET STATISTICS IO ON;

SET STATISTICS TIME ON;

```
1 USE AdventureWorks2022;
2 GO
3
4 SET STATISTICS IO ON;
5 SET STATISTICS TIME ON;
6 select jobTitle, loginID from HumanResources.Employee;
7 GO
```

```
1 Table 'Employee'. Scan count 1, logical reads 9, physical reads 1, page server reads 0, read-
  ahead reads 7, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob
  page server reads 0, lob read-ahead reads 0, lob page server read-ahead reads 0.
2
3 SQL Server Execution Times:
4   CPU time = 16 ms,  elapsed time = 51 ms.
5
```

55. <https://docs.microsoft.com/fr-fr/sql/relational-databases/import-export/import-bulk-data-by-using-bulk-insert-or-openrowset-bulk-sql-server?view=sql-server-ver15>

14.7.3. Méthodes de parcours

Table Scan

Lecture de tous les enregistrements d'une table heap (table sans index cluster) pour trouver les enregistrements appropriés.

Utilisé quand une table n'a aucun index, et le moteur de base de données doit scanner toute la table pour récupérer les enregistrements.

Caractéristiques :

- Parcours séquentiel de toutes les pages de données d'une table heap
- Généralement considéré comme mauvais, mais pas toujours - si vous avez un très petit jeu de données, SQL Server peut choisir un table scan plutôt qu'un index seek

Bonnes pratiques :

- Créer un index cluster approprié ou des index non-cluster pour éviter les table scans sur les grandes tables.

Clustered Index Scan

Parcours de toutes les données d'une table avec index cluster (table physiquement triée).

Les opérations "Scan" lisent l'intégralité de l'index ou de la table.

Caractéristiques :

- Lecture séquentielle de toutes les pages de l'index cluster
- Plus efficace qu'un table scan car les données sont physiquement organisées
- Utilisé quand une grande partie des enregistrements doivent être retournés

Bonnes pratiques :

- Optimiser les clauses WHERE pour améliorer la sélectivité et permettre des index seeks.

Index Scan (Non-Clustered)

Lecture complète d'un index non-cluster.

Une opération scan lit l'intégralité de l'index, contrairement à un seek qui utilise l'arbre B ou une adresse physique pour accéder à une partie spécifique.

Caractéristiques :

- Parcours de toutes les pages de l'index non-cluster
- Peut nécessiter des lookups supplémentaires si toutes les colonnes requises ne sont pas dans l'index

Index Seek

Opération qui utilise l'arbre B ou une adresse physique (RID) pour accéder à une partie spécifique de l'index ou de la table. Un index seek ne scanne pas l'intégralité de l'index, mais navigue dans la structure B-tree pour trouver un ou plusieurs enregistrements rapidement.

Caractéristiques :

- Généralement plus rapide car il navigue directement vers les lignes pertinentes dans l'index
- Navigation efficace dans l'arbre B-tree
- Utilisé quand peu d'enregistrements sont ciblés

Bonnes pratiques :

- Indexer les colonnes utilisées dans les clauses WHERE avec une bonne sélectivité.

⁵⁶ <https://docs.microsoft.com/fr-fr/sql/integration-services/import-export-data/import-and-export-data-with-sql-server-import-and-export-wizard?view=sql-server-ver15>

RID Lookup

Un RID Lookup trouve un enregistrement unique dans une table heap à partir de l'identificateur de ligne unique.

Utilisé après un index seek sur un index non-cluster pour récupérer les colonnes non incluses dans l'index

RID (Row Identifier)

Az Définition

Adresse physique permettant de localiser précisément un enregistrement.

Key Lookup

Similaire au RID Lookup mais sur une table avec index cluster.

Opération de recherche d'enregistrements supplémentaires dans l'index cluster après un index seek sur un index non-cluster.

Bonnes pratiques :

- Inclure les champs supplémentaires nécessaires à la requête dans l'index pour éviter les key lookups.

14.7.4. Méthodes de jointure

Nested Loops Join (Boucles imbriquées)

La table externe (outer input - table du haut) est parcourue et chaque ligne est comparée à chaque ligne de la table interne (inner input - table du bas).

Pour chaque ligne de la table externe, SQL Server effectue une recherche dans la table interne.

Variantes

- Index Nested Loops Join : Utilise un index sur la table interne pour accélérer la recherche
- Naive Nested Loops Join : Parcourt entièrement la table interne pour chaque ligne de la table externe

Méthode est particulièrement efficace quand la table externe est petite et que la recherche dans la table interne est rapide, ce qui est souvent obtenu en indexant la colonne de jointure de la table interne.

Bonnes pratiques

- Indexer les colonnes de jointure de la table interne
- Une jointure Nested Loops avec index fonctionne mieux qu'une merge join ou hash join quand un petit ensemble de lignes est impliqué
- Efficace pour les requêtes avec des prédicats sélectifs

Hash Match Join (Jointure par hachage)

Création d'une table de hachage puis recherche des enregistrements.

La table de hachage est créée en mémoire.

La plus petite table (ou le résultat avec le moins de lignes) est utilisée pour construire la table de hachage.

La table la plus grande est parcourue et chaque ligne est utilisée pour sonder la table de hachage

Hash Match est utilisé par l'optimiseur quand aucun index utile n'est disponible, quand une table est substantiellement plus petite que l'autre, quand les tables ne sont pas triées sur les colonnes de jointure.

Avantages : Efficace pour joindre de grandes tables non indexées

Inconvénients : Consomme de la mémoire pour la table de hachage, peut déborder sur le disque si la mémoire est insuffisante

Merge Join (Jointure par fusion)

Chaque table doit être triée sur les attributs de jointure avant le début de la jointure.

Les deux tables sont ensuite parcourues en parallèle, et la jointure fournit les enregistrements qui correspondent.

Si les deux entrées de jointure ne sont pas petites mais sont triées sur leur colonne de jointure (par exemple, si elles ont été obtenues en parcourant des index triés), une merge join est l'opération de jointure la plus rapide.

Conditions d'utilisation optimales

- Les deux tables sont déjà triées sur les colonnes de jointure
- Présence d'index triés sur les colonnes de jointure
- Tables de tailles relativement importantes

Avantages

- Parcourt chaque table une seule fois
- Performant quand les données sont déjà triées
- Prévisible en termes de performance

Adaptive Join (Jointure adaptative) à partir de SQL Server 17

Adaptive join permet à SQL Server de choisir automatiquement un algorithme physique à la volée entre hash join et nested loops join. Cette fonctionnalité fait partie des améliorations de traitement intelligent des requêtes.

14.8. Statistiques

Statistiques

Az Définition

Métadonnées qui décrivent la distribution des données dans les colonnes d'une table.

Utilisées par l'optimiseur de requêtes pour choisir le meilleur plan d'exécution

Contiennent des informations sur :

- Nombre de lignes distinctes
- Histogramme des valeurs
- Densité des données
- Date de dernière mise à jour

Impact sur les performances

- Statistiques à jour = Plans d'exécution optimaux
- Statistiques obsolètes = Performances dégradées
- Essentielles pour les jointures et les prédicats WHERE

Des statistiques obsolètes = Plans d'exécution inadaptés = Performances dégradées

Types de Statistiques

Statistiques Automatiques

- Créées automatiquement sur les clés primaires et index
- Auto-crées sur colonnes utilisées dans les clauses WHERE
- Mise à jour automatique par défaut (configurable)
- Nommage automatique par SQL Server

Statistiques Manuelles

- Créées explicitement
- Possibilité de créer des statistiques multi-colonnes
- Contrôle total sur la mise à jour
- Utiles pour des cas spécifiques

Visualisation des statistiques

vue sys.stats

```
1 SELECT name, auto_created, user_created, no_recompute, has_filter
2 FROM sys.stats WHERE object_id = OBJECT_ID('MaTable');
```

Gestion des statistiques

CREATE STATISTICS

DROP STATISTICS

```
1 -- Créer des statistiques manuelles sur une colonne
2 CREATE STATISTICS Stats_Colonne1 ON MaTable (Colonne1);
3
4 -- Statistiques multi-colonnes (très utile pour les requêtes complexes)
5 CREATE STATISTICS Stats_Multi ON MaTable (Col1, Col2, Col3);
6
7 -- Statistiques filtrées (SQL Server 2008+)
8 CREATE STATISTICS Stats_Filtered ON MaTable (DateCommande)
9 WHERE DateCommande >= '2024-01-01';
10
11 -- Supprimer des statistiques
12 DROP STATISTICS MaTable.Stats_Colonne1;
```

Mise à jour des statistiques

UPDATE STATISTICS⁵⁷

sp_updatestats⁵⁸

```
1 -- Mise à jour complète de toutes les statistiques d'une table
2 UPDATE STATISTICS MaTable;
3
4 -- Mise à jour avec échantillonnage spécifique
5 UPDATE STATISTICS MaTable WITH SAMPLE 50 PERCENT;
6
7 -- Mise à jour complète (scan complet)
8 UPDATE STATISTICS MaTable WITH FULLSCAN;
9
10 -- Mise à jour d'une statistique spécifique
11 UPDATE STATISTICS MaTable Stats_Colonne1 WITH FULLSCAN;
12
13 -- Mise à jour de toutes les tables
14 EXEC sp_updatestats;
```

⁵⁷. <https://learn.microsoft.com/en-us/sql/t-sql/statements/update-statistics-transact-sql?view=sql-server-ver17>

Analyse des statistiques

DBCC SHOW_STATISTICS

Résultat en 3 parties :

- Header : informations générales
- Density Vector : densité des colonnes
- Histogram : distribution des valeurs

script

(Vérifiez régulièrement modification_counter et last_updated)

```

1 -- Afficher le détail complet des statistiques
2 DBCC SHOW_STATISTICS('MaTable', 'IX_MaTable_Colonne1');

1 -- Informations détaillées sur toutes les statistiques
2 SELECT
3     OBJECT_NAME(s.object_id) AS TableName,
4     s.name AS StatName,
5     c.name AS ColumnName,
6     s.auto_created,
7     s.user_created,
8     sp.last_updated,
9     sp.rows,
10    sp.rows_sampled,
11    sp.steps AS HistogramSteps,
12    sp.modification_counter,
13    CASE
14        WHEN sp.rows < 500 THEN 500
15        ELSE 500 + (sp.rows * 0.2)
16    END AS UpdateThreshold
17 FROM sys.stats s
18 JOIN sys.stats_columns sc ON s.stats_id = sc.stats_id AND s.object_id = sc.object_id
19 JOIN sys.columns c ON sc.column_id = c.column_id AND sc.object_id = c.object_id
20 CROSS APPLY sys.dm_db_stats_properties(s.object_id, s.stats_id) sp
21 WHERE s.object_id = OBJECT_ID('MaTable')
22 ORDER BY sp.last_updated ASC;

```

Configuration Automatique

Options de base de données

- AUTO_CREATE_STATISTICS
- AUTO_UPDATE_STATISTICS
- AUTO_UPDATE_STATISTICS_ASYNC (recommandée pour éviter les blocages)

```

1 -- Activer la création automatique des statistiques
2 ALTER DATABASE MaDB SET AUTO_CREATE_STATISTICS ON;
3
4 -- Activer la mise à jour automatique des statistiques
5 ALTER DATABASE MaDB SET AUTO_UPDATE_STATISTICS ON;
6
7 -- Mise à jour asynchrone (RECOMMANDÉ pour éviter les blocages)
8 ALTER DATABASE MaDB SET AUTO_UPDATE_STATISTICS_ASYNC ON;
9
10 -- Vérifier la configuration actuelle
11 SELECT
12     name,
13     is_auto_create_stats_on,
14     is_auto_update_stats_on,
15     is_auto_update_stats_async_on

```

58. <https://learn.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/sp-updatestats-transact-sql?view=sql-server-ver17>

```
16 FROM sys.databases
17 WHERE name = 'MaDB';
```

Seuils de Mise à Jour

Règles par Défaut

Taille de Table	Seuil de Mise à Jour
< 500 lignes	500 modifications
≥ 500 lignes	500 + 20% des lignes

Le seuil de 20% peut être trop élevé pour des tables à plusieurs millions de lignes

SQL Server 2016+ : Seuils Dynamiques Améliorés

- Seuils plus intelligents pour les grandes tables
- Réduction du seuil pour tables > 25,000 lignes
- Amélioration significative des performances

Script de vérification des seuils

```
1 -- Vérifier quelles statistiques nécessitent une mise à jour
2 SELECT
3     OBJECT_NAME(s.object_id) AS TableName,
4     s.name AS StatName,
5     sp.modification_counter,
6     sp.rows AS TotalRows,
7     CASE
8         WHEN sp.rows < 500 THEN 500
9         ELSE 500 + (sp.rows * 0.2)
10    END AS UpdateThreshold,
11    CASE
12        WHEN sp.modification_counter >
13            (CASE WHEN sp.rows < 500 THEN 500
14              ELSE 500 + (sp.rows * 0.2) END)
15        THEN 'NEEDS UPDATE'
16        ELSE 'OK'
17    END AS Status,
18    sp.last_updated
19 FROM sys.stats s
20 CROSS APPLY sys.dm_db_stats_properties(s.object_id, s.stats_id) sp
21 WHERE s.object_id > 100
22 AND sp.modification_counter > 0
23 ORDER BY sp.modification_counter DESC;
```

Stratégies d'Optimisation

Phase d'Identification

Signaux d'alarme

- Requêtes soudainement lentes
- Plans d'exécution avec estimations incorrectes
- Scans de table au lieu de seeks d'index
- Jointures avec algorithmes inadaptés

Scripts d'identification (à ajuster en fonction des cas)

```
1 -- Requêtes avec estimations très différentes des valeurs réelles
2 SELECT
3     query_plan,
4     execution_count,
5     total_worker_time,
6     total_elapsed_time
```

```

7 FROM sys.dm_exec_query_stats
8 CROSS APPLY sys.dm_exec_query_plan(plan_handle)
9 WHERE total_worker_time > 1000000; -- à ajuster

```

Actions Correctives

1. Créer des statistiques manuelles

– Pour colonnes critiques sans index

```
CREATE STATISTICS Stats_StatusColonne ON MaTable (Status);
```

2. Mise à jour avec FULLSCAN

– Pour colonnes avec distribution très inégale

```
UPDATE STATISTICS MaTable Stats_DateColonne WITH FULLSCAN;
```

3. Statistiques multi-colonnes

– Pour requêtes avec prédicats multiples

```
CREATE STATISTICS Stats_MultiCols ON MaTable (ClientID, DateCommande, Statut);
```



Scripts de Monitoring

Scripts de monitoring

Maintenance des statistiques

```

1 -- Mise à jour intelligente des statistiques
2 DECLARE @TableName NVARCHAR(128)
3 DECLARE @SQL NVARCHAR(MAX)
4 DECLARE TableCursor CURSOR FOR
5     SELECT DISTINCT OBJECT_NAME(s.object_id)
6     FROM sys.stats s
7     CROSS APPLY sys.dm_db_stats_properties(s.object_id, s.stats_id) sp
8     WHERE sp.modification_counter >
9         CASE WHEN sp.rows < 500 THEN 500
10             ELSE 500 + (sp.rows * 0.2) END
11
12 OPEN TableCursor
13 FETCH NEXT FROM TableCursor INTO @TableName
14
15 WHILE @@FETCH_STATUS = 0
16 BEGIN
17     SET @SQL = 'UPDATE STATISTICS ' + QUOTENAME(@TableName) + ' WITH SAMPLE 30 PERCENT'
18     EXEC sp_executesql @SQL
19     PRINT 'Statistiques mises à jour pour : ' + @TableName
20
21     FETCH NEXT FROM TableCursor INTO @TableName
22 END
23
24 CLOSE TableCursor
25 DEALLOCATE TableCursor

```

Monitoring des performances

```

1 -- Statistiques avec le plus de modifications non traitées
2 SELECT TOP 20
3     OBJECT_NAME(s.object_id) AS TableName,
4     s.name AS StatName,
5     sp.last_updated,
6     sp.modification_counter,
7     sp.rows,
8     CAST(sp.modification_counter * 100.0 / sp.rows AS DECIMAL(10,2)) AS
    ModificationPercentage
9 FROM sys.stats s
10 CROSS APPLY sys.dm_db_stats_properties(s.object_id, s.stats_id) sp
11 WHERE sp.rows > 0
12 ORDER BY sp.modification_counter DESC;

```



Bonne pratique

Automatisation

- **Activez** AUTO_CREATE_STATISTICS et AUTO_UPDATE_STATISTICS
- Utilisez AUTO_UPDATE_STATISTICS_ASYNC pour éviter les blocages
- Planifiez des mises à jour régulières via SQL Agent

Échantillonnage

- FULLSCAN pour colonnes critiques et tables < 1M lignes
- SAMPLE 30-50% pour tables moyennes
- SAMPLE 10-20% pour très grandes tables

Ciblage

- Créez des statistiques multi-colonnes pour requêtes complexes
- Utilisez des statistiques filtrées pour données partitionnées
- Concentrez-vous sur les colonnes de jointure et WHERE

Monitoring

- Surveillez modification_counter régulièrement
- Vérifiez la qualité des estimations dans les plans d'exécution
- Testez l'impact après mise à jour

Maintenance

- script à lancer aux heures creuses : UPDATE STATISTICS MaTable WITH SAMPLE 50 PERCENT;
- Après des opérations bulk (INSERT/UPDATE/DELETE massifs), mettez toujours à jour manuellement les statistiques
BULK INSERT MaTable FROM 'data.csv';
UPDATE STATISTICS MaTable WITH FULLSCAN;
- Evitez NORECOMPUTE sauf cas très spécifiques - cela désactive les mises à jour automatiques

Table avec beaucoup d'INSERT

Exemple

Contexte : Table de logs avec 100k+ insertions par jour

Problème : Statistiques obsolètes causent des scans complets

Solution : Mise à jour manuelle

```
1 BULK INSERT LogTable FROM 'daily_logs.csv';
2 UPDATE STATISTICS LogTable WITH FULLSCAN;
3
```

Colonne avec distribution très inégale

Exemple

Contexte : Colonne Status avec 95% = 'Actif', 5% = autres valeurs

Problème : Mauvaises estimations pour les 5%

Solution : Statistiques filtrées

```

1 CREATE STATISTICS Stats_StatusInactif
2 ON Clients (Status)
3 WHERE Status != 'Actif';
4
5 -- Mise à jour avec scan complet pour précision maximale
6 UPDATE STATISTICS Clients Stats_StatusInactif WITH FULLSCAN;

```

Requêtes multi-colonnes fréquentes

[👁 Exemple](#)

Contexte : Requêtes fréquentes sur (ClientID, DateCommande, Region)

Problème : Estimations incorrectes pour combinaisons de colonnes

Solution : Statistiques multi-colonnes

```

1 CREATE STATISTICS Stats_ClientDateRegion
2 ON Commandes (ClientID, DateCommande, Region);
3
4 -- Vérification de l'amélioration
5 SET STATISTICS IO ON;
6 SELECT * FROM Commandes
7 WHERE ClientID = 123
8    AND DateCommande >= '2024-01-01'
9    AND Region = 'Europe'

```

Table partitionnée

[👁 Exemple](#)

Contexte : Table partitionnée par mois

Problème : Statistiques globales pas assez précises

Solution : Statistiques filtrées par partition

```

1 CREATE STATISTICS Stats_Janvier2024
2 ON VentesPartitionnees (Montant)
3 WHERE DateVente >= '2024-01-01' AND DateVente < '2024-02-01';
4
5 CREATE STATISTICS Stats_Fevrier2024
6 ON VentesPartitionnees (Montant)
7 WHERE DateVente >= '2024-02-01' AND DateVente < '2024-03-01';

```

Checklist de Vérification



Bonne pratique

Configuration de Base

- AUTO_CREATE_STATISTICS activé
- AUTO_UPDATE_STATISTICS activé
- AUTO_UPDATE_STATISTICS_ASYNC activé
- Seuils de mise à jour compris

Monitoring Régulier

- Script de vérification modification_counter programmé
- Alertes sur statistiques obsolètes configurées
- Plans d'exécution vérifiés régulièrement
- Performances des requêtes critiques suivies

Maintenance Proactive

- Jobs de mise à jour programmés
- Statistiques manuelles identifiées et créées
- Maintenance post-bulk operations documentée
- Tests de performance avant/après mise à jour

Points Clés



Bonne pratique

Éléments majeurs

- Les statistiques sont cruciales pour l'optimiseur de requêtes
- Automatisation recommandée mais surveillance nécessaire
- Mise à jour proactive après opérations massives
- Statistiques multi-colonnes pour requêtes complexes
- Monitoring régulier du modification_counter

Actions Immédiates

- Auditez vos statistiques existantes
- Implémentez un monitoring automatisé
- Testez l'impact sur vos requêtes critiques
- Établissez une stratégie de maintenance
- Documentez vos processus

Bénéfices Attendus

- Amélioration des temps de réponse
- Réduction de la charge CPU
- Stabilité des performances
- Prévisibilité des plans d'exécution

Interrogations

```

1 -- Vue d'ensemble des statistiques de la base
2 SELECT COUNT(*) AS TotalStats, SUM(CASE WHEN auto_created = 1 THEN
3   1 ELSE 0 END) AS AutoCreated,
4   SUM(CASE WHEN user_created = 1 THEN 1 ELSE 0 END) AS UserCreated FROM sys.stats WHERE
   object_id > 100;
5 -- Statistiques jamais mises à jour
6 SELECT OBJECT_NAME(object_id) AS TableName, name AS StatName, DATEDIFF(day, last_updated,
   GETDATE()) AS DaysOld
7   FROM sys.stats s CROSS APPLY sys.dm_db_stats_properties(s.object_id, s.stats_id) sp
8   WHERE last_updated IS NULL OR last_updated < DATEADD(month, -1, GETDATE());
9
```

14.9. Columnstore

Stockage des données par colonne

- Compression exceptionnelle : Les données similaires étant regroupées, les algorithmes de compression sont beaucoup plus efficaces

Réduction de l'espace de stockage de 70 à 90% par rapport au rowstore traditionnel.

- Performances analytiques : Lecture uniquement des colonnes nécessaires à la requête
Gains de performance de 10 à 100 fois pour les requêtes analytiques typiques grâce à la compression et à l'élimination des segments non pertinents.
- Traitement vectorisé : Opérations sur des blocs de données plutôt que ligne par ligne
Parallélisation optimale : Exploitation maximale des processeurs multi-cœurs pour les opérations analytiques.

Clustered Columnstore Index (CCI)

L'index clustered columnstore remplace complètement la structure de la table.

Toutes les données sont stockées au format en colonne

ROWSTORE (Traditionnel)					
ID	Nom	Age	Ville	Salaire	
1	Alice	25	Paris	45000	
2	Bob	30	Lyon	50000	
3	Claire	28	Paris	48000	

Stockage physique par LIGNES					
1	Alice	25	Paris	45000	
2	Bob	30	Lyon	50000	
3	Claire	28	Paris	48000	

COLUMNSTORE (Analytique)					
ID	Nom	Age	Ville	Salaire	
1	Alice	25	Paris	45000	
2	Bob	30	Lyon	50000	
3	Claire	28	Paris	48000	

Stockage physique par COLONNES					
ID	Nom	Age	Ville	Salaire	
1	Alice	25	Paris	45000	
2	Bob	30	Lyon	50000	
3	Claire	28	Paris	48000	

Nonclustered Columnstore Index (NCCI)

Coexiste avec l'index clustered existant, permettant des requêtes analytiques sur une table principalement utilisée pour l'OLTP.

Limitations

- Pas de données xml, sql_variant, hierarchalID
- pas de triggers sur les tables avec CCI

14.10. Filestream et Filetable

14.10.1. Stockage des BLOB

Stockages disponible pour les objets de grande taille

- dans une table de la base (type varbinary(max))
problème : contrainte de taille du système de fichier
- dans un Filestream
- dans une FileTable

Accès aux objets du système de fichier

Filestream : stockage des BLOB en tant que fichiers

FileTable : stockage d'une arborescence dans un Filestream

Quand utiliser filestream ?

- Les objets stockés ont, en moyenne, une taille supérieurs à 1 MB.
- L'accès à la lecture rapide est important.
- Applications qui ont besoin d'accéder directement aux élément stockés.

14.10.2. Filestream

Mise en œuvre

- activation de filestream sur l'instance
(configuration manager ou propriété de l'instance)
- création d'un groupe de fichier de type filestream
- création d'une colonne de type varbinary(max) avec l'attribut filestream

référence⁵⁹

groupe de fichier de type filestream

```
1 CREATE DATABASE Archive
2 ON
3 PRIMARY ( NAME = Arch1,
4 FILENAME = 'C:\data\archdat1.mdf'),
5 FILEGROUP FileStreamGroup1 CONTAINS FILESTREAM ( NAME = Arch3,
6 FILENAME = 'C:\data\filestream1')
7 LOG ON ( NAME = Archlog1,
8 FILENAME = 'C:\data\archlog1.ldf')
9 GO
```

table avec une colonne stockée dans le filestream

```
1 CREATE TABLE Archive.dbo.Records
2 (
3 [Id] [uniqueidentifier] ROWGUIDCOL NOT NULL UNIQUE,
4 [SerialNumber] INTEGER UNIQUE,
5 [Chart] VARBINARY(MAX) FILESTREAM NULL
6 );
7 GO
```

SSMS

 Remarque

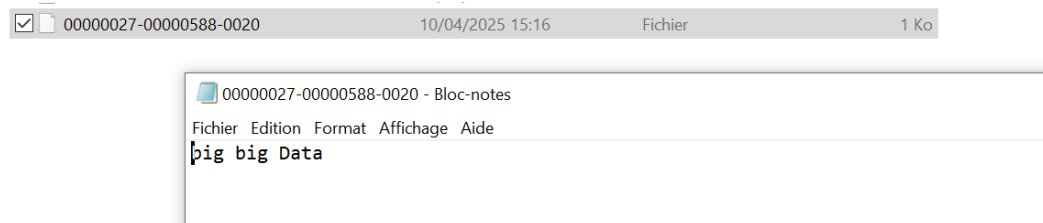
On ne peut pas voir l'attribut filestream d'une colonne

Utilisation avec T-SQL

```
1 INSERT INTO Archive.dbo.Records
2 VALUES (NEWID(), 1, NULL);
3 INSERT INTO Archive.dbo.Records
4 VALUES (NEWID(), 2,
5 CAST ('' AS VARBINARY(MAX)));
6 INSERT INTO Archive.dbo.Records
7 VALUES (NEWID(), 3,
8 CAST ('big big Data' AS VARBINARY(MAX)));
9 GO
10
11 select * from Archive.dbo.Records;
```

1 Id	SerialNumber	Chart
2 -----	-----	-----
3 79FE0C01-9767-49CC-82FC-39C71F3669D6	1	NULL
4 B6F393B2-0DD8-49A9-8F9C-87A5DA57B4B7	2	0x
5 557E5120-8F0D-4A86-8ACA-4EE76774D08B	3	0x626967206269672044617461

⁵⁹. <https://learn.microsoft.com/en-us/sql/relational-databases/blob/filestream-sql-server?view=sql-server-ver16>



Utilisation avec l'API Win32

On peut accéder aux fichiers avec l'API Win32

Référence⁶⁰

Modification ou suppression des fichiers

⚠ Attention

Ne pas supprimer ou modifier directement des fichiers stockés dans un filestream !

14.10.3. Filetable

Structure prédéfinie

Le schema d'une filetable est fixe : Filetable schema⁶¹

Nom d'attribut de fichier	type	Taille	Default	Description	Accessibilité du système de fichiers
path_locator	hierarchyid	variable	hierarchyid qui identifie la position de cet élément.	Position de ce nœud dans le FileNamespace hiérarchique. Clé primaire de la table.	Peut être créée et modifiée en définissant les valeurs de chemin d'accès Windows.
stream_id	[uniqueidentifieur] rowguidcol		Valeur retournée par la fonction NEWID() .	ID unique pour les données FILESTREAM.	Non applicable.
file_stream	varbinary(max) flux de fichier	variable	NULL	Contient les données FILESTREAM.	Non applicable.

⁶⁰. <https://learn.microsoft.com/en-us/sql/relational-databases/blob/filestream-sql-server?view=sql-server-ver16#file-system-streaming-access>

⁶¹. <https://learn.microsoft.com/en-us/sql/relational-databases/blob/filetable-schema?view=sql-server-ver16>

Nom d'attribut de fichier	type	Taille	Default	Description	Accessibilité du système de fichiers
file_type	nvarchar(255)	variable	NULL. Une opération de création ou de changement de nom dans le système de fichiers remplit la valeur d'extension du fichier à partir du nom.	Représente le type du fichier. Cette colonne peut être utilisée comme TYPE COLUMN quand vous créez un index de recherche en texte intégral. file_type est une colonne calculée persistante.	Calculé automatiquement. Ne peut pas être définie.
Nom	nvarchar(255)	variable	Valeur GUID.	Nom du fichier ou du répertoire.	Peut être créé ou modifié à l'aide des API Windows.
parent_path_locator	hierarchyid	variable	hierarchyid qui identifie le répertoire qui contient cet élément.	hierarchyid du répertoire conteneur. parent_path_locator est une colonne calculée persistante.	Calculé automatiquement. Ne peut pas être définie.
cached_file_size	bigint			Taille des données FILESTREAM, en octets. cached_file_size est une colonne calculée persistante.	Bien que la taille du fichier mis en cache soit automatiquement mise à jour, elle peut être mal synchronisée dans des circonstances exceptionnelles. Pour calculer la taille exacte, utilisez la fonction DATALENGTH() .
creation_time	datetime2(4) Non Null	8 octets	Heure actuelle.	Date et heure de création du fichier.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.

Nom d'attribut de fichier	type	Taille	Default	Description	Accessibilité du système de fichiers
last_write_time	datetime2(4) Non Null	8 octets	Heure actuelle.	Date et heure de dernière mise à jour du fichier.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
last_access_time	datetime2(4) Non Null	8 octets	Heure actuelle.	Date et heure du dernier accès au fichier.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
is_directory	bit Non Null	1 octet	FAUX	Indique si la ligne représente un répertoire. Cette valeur est calculée automatiquement et ne peut pas être définie.	Calculé automatiquement. Ne peut pas être définie.
is_offline	bit Non Null	1 octet	FAUX	Attribut de fichier hors connexion.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
is_hidden	bit Non Null	1 octet	FAUX	Attribut de fichier masqué.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
is_readonly	bit Non Null	1 octet	FAUX	Attribut de fichier en lecture seule.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
is_archive	bit Non Null	1 octet	FAUX	Attribut Archive.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.

Nom d'attribut de fichier	type	Taille	Default	Description	Accessibilité du système de fichiers
is_system	bit Non Null	1 octet	FAUX	Attribut de fichier système.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.
is_temporary	bit Non Null	1 octet	FAUX	Attribut de fichier temporaire.	Calculé automatiquement. Peut également être défini à l'aide d'API Windows.

Activation dans la base de données

Propriétés

- activer filetable
- définir le répertoire de stockage

Référence⁶²

Création d'une table

```

1 USE Archive
2 GO
3
4 IF OBJECT_ID('dbo.demo_filetable', 'U') IS NOT NULL
5     DROP TABLE dbo.demo_filetable
6 GO
7
8 CREATE TABLE dbo.demo_filetable AS FILETABLE
9 WITH
10 (
11     FILETABLE_DIRECTORY = 'demo_filetable',
12     FILETABLE_COLLATE_FILENAME = database_default
13 )
14 GO
15

```

⁶². <https://learn.microsoft.com/en-us/sql/relational-databases/blob/enable-the-prerequisites-for-filetable?view=sql-server-ver16>

Utilisation

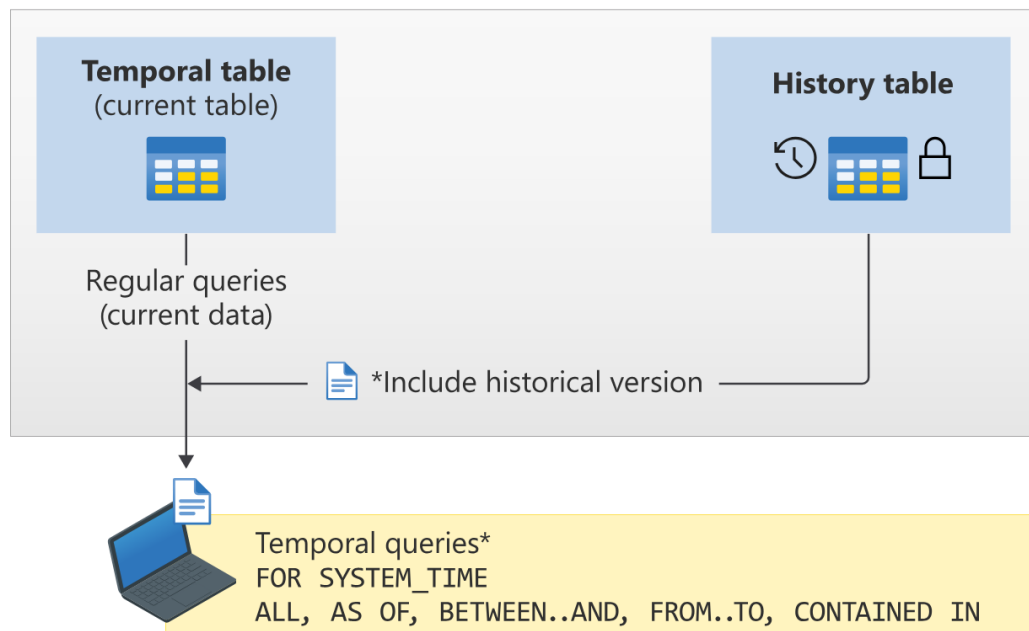
- Copie des fichiers dans le répertoire et mise à jour du FileTable
Référence⁶³
- Utilisation de T-SQL
Référence⁶⁴
- Utilisation de l'API

⁶³. <https://learn.microsoft.com/en-us/sql/relational-databases/blob/load-files-into-filetables?view=sql-server-ver16&tabs=bcp>

Ressources annexes

Sélectionnez une ou plusieurs tâches de maintenance :

- ☒ Vérifier l'intégrité de la base de données
- ☐ Compacter la base de données
- ☐ Réorganiser l'index
- ☐ Reconstruire l'index
- ☐ Mettre à jour les statistiques
- ☐ Nettoyer l'historique
- ☐ Exécuter le travail de SQL Server Agent
- ☐ Sauvegarder la base de données (complète)
- ☐ Sauvegarder la base de données (différentielle)
- ☐ Sauvegarder la base de données (journal des transactions)
- ☐ Tâche de nettoyage de maintenance



⁶⁴. <https://learn.microsoft.com/en-us/sql/relational-databases/blob/access-filetables-with-transact-sql?view=sql-server-ver16>

Index

BEGIN	49
Bitmap Heap scan	88
Bitmap Index scan	88
bitmap scan	88
BREAK.....	54
contained database.....	42
CONTINUE.....	54
CREATE FUNCTION	58
CREATE OR ALTER FUNCTION	58
CREATE OR ALTER PROCEDURE.....	61
CREATE PROCEDURE	61
DECLARE	48
ELSE.....	49
END	49
EXEC	62
filestream	99
GO	48
GOTO	54
Hash join.....	89
IF	49
index only scan	88
index scan	88
Nested loop join.....	89
sequential scan.....	88
SET.....	48
Sort Merge.....	89
WHILE	51

Contenus annexes

1. OLTP - OLAP

OLTP (Online Transaction Processing)

Az Définition

Le traitement des transactions en ligne (Online Transaction Processing, OLTP) est un type de traitement des données qui consiste à exécuter un certain nombre de transactions survenant simultanément, par exemple, des opérations bancaires en ligne, des achats, la saisie de commandes ou l'envoi de messages texte.

OLAP (Online Analytical Processing)

Az Définition

le traitement analytique en ligne (Online Analytical Processing, OLAP) est un type de traitement des données orienté vers l'analyse sur-le-champ d'informations selon plusieurs axes, dans le but d'obtenir des rapports de synthèse pour effectuer des analyses.

Comparaison OLTP - OLAP

Système OLTP	Systèmes OLAP
Permet l'exécution en temps réel d'un nombre élevé de transactions de base de données par de nombreux utilisateurs	Interroge généralement de nombreux enregistrements (voire tous) dans une base de données à des fins d'analyse
Nécessite un temps de réponse extrêmement court	Exige des temps de réponse largement plus lents que ceux requis par OLTP
Modifie fréquemment de petites quantités de données et nécessite généralement un équilibre entre opération de lecture et d'écriture	Ne modifie pas du tout les données ; les workloads consomment généralement beaucoup de ressources en lecture
Utilise des données indexées pour améliorer les temps de réponse	Stocke les données en colonnes pour faciliter l'accès à un grand nombre d'enregistrements
Requiert des sauvegardes fréquentes ou simultanées de la base de données	Nécessite des sauvegardes de base de données beaucoup moins fréquentes
Nécessite relativement peu d'espace de stockage	Requiert généralement d'importants espaces de stockage pour les grands volumes de données historiques
Exécute généralement des requêtes simples impliquant un ou plusieurs enregistrements	Exécute des requêtes complexes impliquant un grand nombre d'enregistrements

2. Tuning automatique

Tuning automatique

Az Définition

Application automatique d'un meilleur plan en cas de régression

Remarque Il faut que le Query Store soit activé en READ/WRITE sur la base

```
1 -- Activer FORCE_LAST_GOOD_PLAN
2 ALTER DATABASE [AdventureWorks2022]
3 SET AUTOMATIC_TUNING (FORCE_LAST_GOOD_PLAN = ON);
4
5 -- Désactiver
6 ALTER DATABASE [AdventureWorks2022]
7 SET AUTOMATIC_TUNING (FORCE_LAST_GOOD_PLAN = OFF);
8
9 -- Vérifier l'état actuel
10 SELECT name, desired_state_desc, actual_state_desc, reason_desc
11 FROM sys.database_automatic_tuning_options;
```

Documentation

Références⁶⁵

Azure database

Remarque

possibilité de créer et supprimer automatiquement des index

3. Mode batch

Mode batch

Az Définition

Principe de fonctionnement

- regroupement des données en lots (900 à 1000 lignes)
- exécution des opérations sur l'ensemble du lot en une seule fois.

Avantages

- Amélioration des performance
moins d'appels de fonctions.
optimisation de l'utilisation du CPU.
opérations arithmétiques et de comparaison effectuées sur plusieurs valeurs simultanément.
- Meilleure utilisation de la mémoire cache :
Traitement des blocs de données contigus => réduction des accès mémoires
- Optimisations vectorielles
utilisation des instructions SIMD (Single Instruction, Multiple Data) pour traiter plusieurs valeurs en parallèle.

⁶⁵ <https://learn.microsoft.com/en-us/sql/relational-databases/automatic-tuning/automatic-tuning?view=sql-server-ver17>

Utilisations

- Columnstore Index : Principalement avec les index columnstore (clustered ou non-clustered)
- Tables importantes : Généralement pour les tables contenant plus de 130 000 lignes
- Opérations analytiques : Agrégations, jointures, tris sur de gros volumes de données
- Mode de compatibilité : à partir de SQL Server 2019

Opérateurs compatibles

- *Hash Join et Nested Loop Join* (cf. p.89)
- Agrégations (SUM, COUNT, AVG, etc.)
- Filtres et prédicats
- Opérateurs de tri

Limitations

- Petites tables : Le surcoût de mise en place peut dépasser les bénéfices
- Opérations transactionnelles : Plus adapté aux charges analytiques qu'OLTP
- Types de données complexes : XML, HierarchicalID, ... pas supportés

Références

Microsoft⁶⁶

⁶⁶. <https://learn.microsoft.com/en-us/sql/relational-databases/query-processing-architecture-guide?view=sql-server-ver17>