WIKIPÉDIA

Microsoft SQL Server

Microsoft SQL Server est un système de gestion de base de données (SGBD) en langage SQL incorporant entre autres un SGBDR (SGBD relationnel ») développé et commercialisé par la société Microsoft. Il fonctionne sous les OS Windows et Linux (depuis mars 2016), mais il est possible de le lancer sur Mac OS via Docker, car il en existe une version en téléchargement sur le site de Microsoft[1].

SQL Server

Informations

Développé par	Microsoft
Première version	24 avril 1989
Dernière version	2017 (2 octobre 2017)
Écrit en	C++, C et C#
Système d'exploitation	Linux, Microsoft Windows Server (d) et Microsoft Windows
Type	Système de gestion de base de données relationnelle (en)
Licence	Licence propriétaire et EULA
Site web	www.microsoft.com/sql-server (https://www.microsoft.com/sql-server/)

Sommaire

- Fichiers
- Sécurité
  - Comptes de service
  - Comptes de connexion
  - Utilisateur SQL
  - Rôles
  - Schéma SQL
  - Chiffrement

Objets de la base de données

- Tables
- Vues
- Index
  - Indexation relationnelle
  - Indexation verticale
  - Indexation textuelle et sémantique
  - Indexation des objets
- Procédures stockées
- Déclencheurs (trigger)
- Fonctions (UDF : User Defined Function)
- Aspects objet

Outils

- SQL Server Management Studio (SSMS)
- SQL Server Business Intelligence Development Studio / SQL Server Data Tools
- Profiler (générateur de profil)

Services et moteurs

- SQL Server
- SQL Server Agent
- Full Text Search
- Distributed Transaction Coordinator
- Notification Services
- Décisionnel
- Cartographie
- Integration Services (SSIS : SQL Server Integration Services)
- Analysis Services
- Reporting Services

Éditions

- Datacenter Edition
- Enterprise Edition
- Developer Edition
- Standard Edition
- Workgroup Edition
- Web Edition
- Personal Edition
- Windows Internal Database / SQL Server Embedded
- MSDE / Express Edition
- Compact Edition

Licences

Outils complémentaires intégrant un serveur SQL server

- BizTalk
- Sharepoint
- Dynamics
- TFS (Team Foundation Server)
- SCOM (System Center Operations Manager)

Historique

Bien qu'il ait été initialement codéveloppé par Sybase et Microsoft, Ashton-Tate a également été associé à sa première version, sortie en 1989. Cette version est sortie sur les plates-formes Unix et OS/2. Depuis, Microsoft a porté ce système de base de données sous Windows.

Lors de sa création, Sybase SQL Server hérite des principes du moteur Ingres développé à l'origine par l'université de Berkeley.

En 1994, le partenariat entre les deux sociétés ayant été rompu, Microsoft a sorti la version 6.0 puis 6.5 seul, sur la plate-forme Windows NT.

- Microsoft a continué de commercialiser le moteur de base de données sous le nom de SQL Server
- Tandis que Sybase, pour éviter toute confusion, a renommé Sybase SQL Server en Sybase Adaptive Server Enterprise.

Microsoft SQL Server fait désormais partie de la stratégie technique de Microsoft en matière de base de données. Le moteur MSDE, qui est la base de SQL Server, doit à terme remplacer le moteur Jet (celui qui gère les bases Access) dans les applications telles que Exchange et Active Directory.

La version 2005 de SQL Server est sortie le 3 novembre 2005 en même temps que Visual Studio 2005. La prise en charge de Windows Vista et de Windows Server 2008 n'a été ajoutée qu'à partir du Service Pack 2 (SP2). Actuellement le Service Pack 3 est disponible. L'optimiseur a été entièrement refait, tout comme le moteur relationnel.

La version 2008 de SQL Server (nom de code Katmaï) est sortie en août 2008. La version mineure 2008 R2 est sortie en 2010.

La version 2012 de SQL Server est sortie en avril 2012.

La version 2014 de SQL Server est sortie le 1er avril 2014 avec un moteur « in memory » complémentaire au moteur relationnel.

La version 2016 de SQL Server est sortie le 1er juin 2016.

Lors du développement de la version 2005, le projet était à l'époque l'un des plus grands projets informatiques du monde. Plus de 1 600 développeurs ont participé au codage du noyau et l'on estime que plus de 10 000 autres personnes y ont travaillé de près ou de loin (Interactions homme-machine, documentation, traduction...). [réf. nécessaire]

Particularités

SQL Server se distingue de la concurrence par une grande richesse ne nécessitant aucune option payante supplémentaire dans la limite de la version choisie.

Multi-base, multi-schéma

Par rapport à ses concurrents que sont Oracle, MySQL ou PostgreSQL, SQL Server se distingue par le fait que c'est un SGBDR originellement multibase et multischéma. Il est possible de faire des requêtes nativement interbases. Par exemple la requête suivante lie deux tables de deux bases de données différentes :

```
SELECT *
FROM   BASE_A.dbo.TABLE1 AS T1
       INNER JOIN BASE_B.dbo.TABLE2 AS T2
             ON T1.ID = T2.ID;
```

L'optimiseur étant capable de faire un plan de requête parfaitement optimisé même si la requête consulte des données de plusieurs bases...

Bien que PostGreSQL soit multibase et multischéma, cette possibilité d'interrogation simultanée n'est pas native et il faut passer par le truchement de "dblink" qui interdit les jointures et donc toute possibilité d'optimisation... Oracle avec sa version 12 tente d'intégrer ce même concept de multibase (appelé multi-tenant) mais souffre du même problème que PostGreSQL. MySQL est mono schéma, multibase.

Gestion des schémas SQL

La souplesse de la gestion des schémas SQL est telle qu'il est possible de transférer un objet d'un schéma à l'autre par le simple biais d'une commande `ALTER SCHEMA`.

Les propriétaires sont distincts des schémas et il est possible de transférer la propriété d'une base, d'un schéma ou d'un objet d'un utilisateur SQL à l'autre, par le biais de la commande `ALTER AUTHORIZATION`.

Parallélisme

SQL Server fonctionne nativement de manière parallèle. Dès qu'une requête est estimée dépasser le seuil du coût à partir duquel un plan de requête peut être parallélisé, SQL Server récrit le plan en utilisant des algorithmes multi-threadés et si le nouveau plan s'avère moins couteux, la substitution a lieu de manière automatique. Cette fonctionnalité existe dans toutes les éditions et n'est pas un module payant à rajouter en sus comme c'est le cas d'Oracle.

Les opérations de lecture et d'écriture physique bénéficient aussi du parallélisme systématiquement du fait que les opérations d'IO sont effectuées directement par SQL Server et non à travers la couche système comme c'est le cas de PostGreSQL ou MySQL.

Compression

Dans la version Enterprise, le moteur est capable de compresser les données au niveau ligne ou page dans les tables comme dans les index. La compression des sauvegardes est disponible à partir de l'édition standard depuis la version 2008 R2.

Sauvegarde à chaud

Depuis SQL Server 7 (1998), SQL Server permet de sauvegarder "base ouverte", sans interrompre le service des données et sans perturber l'activité des utilisateurs. Cette sauvegarde peut s'effectuer de manière globale, par groupe de fichiers, par fichiers ou par le biais du journal de transaction.

Unique... sauf NULL !

Dans SQL Server, les contraintes d'unicité prennent en compte le marqueur NULL comme étant une valeur et par ce fait interdisent donc la présence de plusieurs NULL dans les colonnes d'une telle contrainte. Ceci n'est pas conforme à la norme SQL puisque le NULL étant une absence de valeur, l'unicité ne se pose pas. Il existe cependant un moyen simple de contourner ce problème, qui consiste à créer un index unique filtré qui concernera les données valuées.

Exemple, soit la table :

```
CREATE TABLE T_EMPLOYE_EMP
(EMP_ID             INT PRIMARY KEY,
 EMP_NOM            VARCHAR(32) NOT NULL,
 EMP_MATRICULE      CHAR(8));
```

Comme le matricule doit être unique pour les valeurs connues, la création d'un index unique filtré sur les valeurs exprimées de la colonne matricule, permet de résoudre le problème :

```
CREATE UNIQUE INDEX X_EMP_MATRICULE
   ON T_EMPLOYE_EMP(EMP_MATRICULE)
   WHERE EMP_MATRICULE IS NOT NULL;
```

Déclencheurs ré-entrants (table "mutante")

Dans les déclencheurs SQL Server (triggers) il est possible de mettre à jour la table cible de l'événement à l'origine de l'exécution du déclencheur, contrairement à de nombreuses bases qui interdisent cette possibilité (erreur de "table mutante" dans Oracle). Il faut se rappeler que c'est Sybase qui, à l'origine a inventé le concept de trigger en 1986, mettant ainsi pour la première fois du code exécutable dans une base de données relationnelle.

Par exemple un déclencheur `UPDATE` sur une table peut modifier les données en instance de modification :

```
CREATE TRIGGER E_U_PERSONNE
ON T_PERSONNE_PRS
FOR UPDATE
AS
   IF UPDATE(PRS_NOM)
      UPDATE T_PERSONNE_PRS
      SET    PRS_NOM = UPPER(PRS_NOM)
      WHERE  PRS_ID IN (SELECT PRS_ID
                        FROM   inserted);
```

Dans l'exemple ci-avant, les noms des personnes nouvellement insérées ou dont le nom est modifié est remis en majuscule.

La réentrance pouvant conduire à la récursivité, différents paramétrages permettent de régler ce problème au niveau de la base, comme au niveau de l'instance.

Vues indexées automatisées

Contrairement à Oracle ou PostGreSQL ou les vues matérialisées doivent la plupart du temps être rafraichies avant utilisation, les vues indexées de SQL Server sont toujours synchrones. De plus l'optimiseur est capable de substituer à la volée la vue indexée à tout ou partie de la requête (version Enterprise) ce qui est très pratique lorsque le DBA n'a pas accès au code des programmes clients (cas des applications d'éditeurs).

Pooling automatique

Pour les applications conçues dans .net et utilisant le connecteur ADO, SQL Server effectue du pooling automatique. Le simple fait de distinguer deux chaines de connexion suffit à créer deux pools différents. Pour que le pooling soit efficace, il faut fermer les connexions, une fois le traitement informatique opéré.

Administration centralisée

Par différents biais, SQL Server permet de gérer de manière centralisée un ensemble de serveur. Vous pouvez créer un serveur d'administration centralisée, et ajouter une liste de serveurs SQL de l'entreprise afin de lancer des requêtes envoyées simultanément sur tous les serveurs ou bien créer ou exécuter une même procédure sur tous les serveurs à la fois, ceci dans l'outil SSMS (SQL Server Management Studio)..

De la même façon, le planificateur de tâche et gestionnaire d'alerte (l'Agent SQL) permet de définir des travaux multi-serveurs, comme la sauvegarde, la défragmentation des index le recalcul des statistiques ou la vérification d'intégrité physique des espaces de stockage.

Fonctionnalités pour le développement

Le moteur OLTP de SQL Server est doté de très nombreuses fonctionnalités qu'il serait difficile de toutes énumérer. En voici quelques-unes qui font la différence avec des SGBD plus légers comme MySQL ou PostGreSQL...

Table, index et procédure "In Memory"

SQL Server est doté d'un moteur "In Memory" depuis la version 2014. Les tables et index peuvent être totalement en mémoire (structure et données : data and schema) ou persistantes au niveau de la structure (data only). Les procédures, dans des limites bien définies, peuvent être compilées en mode natif, plutôt qu'interprétées. Les mises à jour restent transactionnées, sans pour autant qu'il soit besoin d'écrire dans les fichiers du journal de transactions (versionnage "in memory"), Cela permet des améliorations très sensibles de performance pour des traitements de calculs complexes portant sur des volumes conséquents de données. Un exemple typique est le calcul des performances financières des cours de bourses en temps réel.

Intra jointure (APPLY)

SQL Server a été le premier avec la version 2005 a introduire l'opérateur d'intra-jointure `APPLY` afin de faciliter la récupération des données encapsulées dans une table virtuelle d'une colonne de table, comme c'est le cas lorsque l'on utilise une fonction table dans une requête dont un des arguments vient d'une des tables de la requête, ou encore lorsque l'on tabularise du XML. L'opérateur `APPLY` doit être précédé du mot clef `CROSS` ou `OUTER` afin d'en définir le comportement. Avec `CROSS` si l'argument est NULL, les lignes concernées sont supprimées du résultat tandis qu'avec `OUTER` elles sont préservées.

L'opérateur `APPLY` est voisin de l'opérateur `LATERAL` de la norme SQL, mais sa syntaxe plus étendue permet des opérations que `LATERAL` ne peut pas réaliser. Oracle vient récemment d'introduire cet opérateur dans sa version 12.1 (2014).

```
SELECT top 10 *
FROM   sys.dm_exec_query_stats AS qs
       CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle)
       OUTER APPLY sys.dm_exec_query_plan(qs.plan_handle)
ORDER BY total_worker_time DESC
```

Dans cet exemple, on utilise la fonction table sys.dm_exec_query_plan, qui restitue le plan d'exécution, à partir du "handle" de plan fournit par l'argument "plan_handle" présent dans la vue sys.dm_exec_query_stats. Cette requête permet de connaître les 10 requêtes ayant consommé le plus de temps CPU depuis le démarrage de l'instance.

Collations (interclassement des littéraux)

SQL Server dispose d'une des plus riches collections de collations (3 887 en version 2014), permettant de piloter 101 langues parmi lesquelles, outre le français, on trouve le Breton et le Corse ! Les collations permettent d'effectuer des recherches en tenant compte ou non de la casse (majuscules, minuscules - CS/CI), des accents et autres caractères diacritiques (accents, cédille, ligatures... - AS/AI), de la largeur du caractères (2 = ^{2} ? - WS) ou des différents types de kana (katakana et hiragana du japonais - KS), ou bien encore d'effectuer des comparaisons binaires tenant compte ou non du code hexadécimal. Les collations étant totalement indépendantes des jeux de caractères (jeux Iso_1) bien qu'elles permettent de les simuler (ASCII, UNICODE, UTF16, EBCDIC...). Le support des collations étant fourni à quatre niveaux : serveur (1), base (2), colonne de table ou de vue (3) et enfin prédicat (4) manipulant des chaines de caractères et tri des colonnes textuelles dans les requêtes.

```
SELECT *
FROM   T_UTILISATEUR_APPLICATIF_UTA
WHERE  UTA_CONNEXION    = 'Éric Blüm' COLLATE French_CI_AI
  AND  UTA_MOT_DE_PASE = 'a²E3wx+Z'  COLLATE Latin1_General_CS_AS_WS
```

Dans l'exemple suivant, on recherche un nom de connexion avec une collations "lâche" (sans tenir compte de la casse ou des accents) et le mot de passe avec une collation "serrée" (binaire) :

La liste des collations disponible sur le serveur peut être obtenue par la requête :

```
SELECT * FROM sys.fn_helpcollations()
```

Données XML

Depuis la version 2005, SQL Server supporte le type XML (fragment) et permet de manipuler du XML via des requêtes XQuery/XPath. Une colonne de type XML peut être typée via une collection de schéma XML (XSD - Xml Schema Definition). L'exemple ci après montre la création d'une collection de schéma XML à un seul élément, puis la création d'une table avec une colonne XML typée avec cette collection de schéma :

```
-- création d'un schéma XML pour validation des données d'une colonne XML
CREATE XML SCHEMA COLLECTION XSC_DONNEES_ENTREPRISE as N'
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:company="http://www.exemple.com/entreprise"
    targetNamespace="http://www.exemple.com/entreprise" elementFormDefault="qualified">
  <xs:element name="personnel">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="employe">
          <xs:complexType>
            <xs:all>
              <xs:element name="nom" type="xs:string" />
            </xs:all>
            <xs:attribute name="matricule" type="xs:ID" />
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>';

-- création d'une table qui possède une colonne de type XML dont les données doivent se conformer au schéma ci
avant
CREATE TABLE T_ENTREPRISE_EMPLOIS_EEP
(EEP_ID         INTEGER PRIMARY KEY IDENTITY,
 EEP_NAME       VARCHAR(12),
 EEP_XML_DATA   XML (XSC_DONNEES_ENTREPRISE) --> création d'une colonne XML typée par XSC_DONNEES_ENTREPRISE
(collection de schémas XML)
 );
```

Une requête SELECT d'extraction peut générer du XML en sortie au lieu d'un résultat tabulaire, via la clause Transact-SQL "FOR XML" pour laquelle il existe de nombreuses options de présentation. L'exemple ci-dessous permet de présenter une hiérarchie XML listant les contraintes d'une table sous forme d'éléments XML :

```
SELECT TABLE_SCHEMA AS schemaSQL, TABLE_NAME AS nomTable,
       CONSTRAINT_NAME AS nomContrainte, CONSTRAINT_TYPE AS typeContrainte
FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS AS contrainte
WHERE TABLE_NAME = 'DepartementFrancais'
FOR XML AUTO, ELEMENTS, ROOT('Contraintes');
```

Cette requête renverra par exemple, les données suivantes :

```
<Contraintes>
  <contrainte>
    <schemaSQL>Reference</schemaSQL>
    <nomTable>DepartementFrancais</nomTable>
    <nomContrainte>pk_DepartementFrancais</nomContrainte>
    <typeContrainte>PRIMARY KEY</typeContrainte>
  </contrainte>
  <contrainte>
    <schemaSQL>Reference</schemaSQL>
    <nomTable>DepartementFrancais</nomTable>
    <nomContrainte>chk_DepartementFrancais_ForceSTSRid</nomContrainte>
    <typeContrainte>CHECK</typeContrainte>
  </contrainte>
  <contrainte>
    <schemaSQL>Reference</schemaSQL>
    <nomTable>DepartementFrancais</nomTable>
    <nomContrainte>UK_DepartementFrancais_NomChefLieu</nomContrainte>
    <typeContrainte>UNIQUE</typeContrainte>
  </contrainte>
  <contrainte>
    <schemaSQL>Reference</schemaSQL>
    <nomTable>DepartementFrancais</nomTable>
    <nomContrainte>UK_DepartementFrancais_Nom</nomContrainte>
    <typeContrainte>UNIQUE</typeContrainte>
  </contrainte>
</Contraintes>
```

Enfin, il est possible de manipuler directement des données XML dans les routines (venant par exemple de fichiers), en utilisant les fonctions et procédures OPENXML, sp_xml_preparedocument, sp_xml_removedocument…

Système d'Information Géographique (SIG) intégré

Depuis la version 2008, SQL Server intègre un SIG dans toutes les versions. Celui-ci permet de construire des objets géométriques (géométrie euclidienne) ou géographique (tenant compte de la courbure spatiale de l'écorce terrestre) à l'aide des types GEOMETRY et GEOGRAPHY (POINT, LINESTRING, POLYGON et combinaisons…). Il est basé sur le standard OGC, tout comme l'add-on spatial PostGIS de PostgreSQL. la version 2012 y a ajouté les objets courbes (CircularStrings, CompoundCurves, CurvePolygons…) et une meilleure précision des calculs (48 bits au lieu de 27 précédemment).

Les index spatiaux permettent de rendre « cherchable » certaines méthodes telles que STContains, STDistance, STEquals, STIntersects, STOverlaps, STTouches, STWithin…

Stockage de fichiers électroniques : FileStream et FileTable

Conformément à la norme SQL qui propose le DATALINK, SQL Server met en œuvre un outil similaire appelé FileStream permettant de stocker les fichiers à titre de fichiers dans le système d'exploitation, mais sous la responsabilité et le contrôle du serveur de bases de données (sérialisation, transactionnement, fiabilité, sécurité…). Ceci permet, entre autres, une sauvegarde intègre et synchrone des données relationnelles et non relationnelles, par

exemple pour les images des produits d'un site web, comme les manuels électroniques PDF des machines d'un parc pour la maintenance. Cet outil est notablement utilisé pour la GED. Les fichiers électroniques pouvant faire l'objet d'une indexation textuelle.

En complément et depuis la version 2012, SQL Server propose le concept de FileTable, qui n'est autre qu'une table virtuelle associée à un point d'entrée (répertoire) du système informatique, afin de permettre la gestion bidirectionnelle des fichiers électroniques : les fichiers manipulés par le langage SQL sont accessibles dans le système de fichiers de Windows et réciproquement : on peut insérer un fichier ou un répertoire dans la table par le biais d'une requête SQL et ce dernier sera vu dans l'arborescence système.

L'outil de gestion de la connaissance de l'entreprise Sharepoint, utilise massivement ce concept pour gérer les documents électronique qui y sont déposés, et toutes les versions transitoires.

Web Services

Introduit avec la version 2008, la possibilité de créer des services Web natifs directement dans le serveur à l'aide de points de terminaison SOAP/HTTP, a été considérée comme obsolète dans la version 2008 R2 et retirée dans la version 2012. Néanmoins, SQL Server permet de créer, et utilise en interne, de nombreux points de terminaison pour des services particuliers tel que la mise en miroir des bases ou Service broker...

Service Broker

L'objet de Service Broker est de fournir un outil de messagerie de base de données permettant de gérer des flux de données entre serveurs SQL de manière asynchrone, sérialisé et transactionnés par le biais de files d'attente qui sont des tables systèmes. Cela permet de créer des architecture SODA (Service Oriented Database Architecture) c'est-à-dire des bases de différentes instances qui dialoguent entre elles au niveau des données et non plus au niveau applicatif comme c'est le cas de SOA.

Cela permet de faire des bases de données réparties collaboratives et réparties, du "grid computing", ou encore du WorkFlow, et par exemple de fournir une meilleure surface d'attaque pour les accès à très forte concurrence. Ce système se distingue de la réplication de données, par le fait que les informations sont principalement véhiculées sous forme XML (chiffré), sur la couche HTTP (Internet par exemple) par le biais de services Web et qu'il n'existe aucune obligation pour les bases et les tables de provenance et de destinations d'être identiques sur le plan de la structure.

Un exemple intéressant est de remplacer des outils de messageries traditionnels tels que MQ Series, par un tel dispositif, bien plus fiable...

Service Broker pouvant être utilisé via l'édition gratuite SQL Server Express, il a été utilisé notablement dans l'industrie pour gérer les informations de production de nombreuses machines d'un même parc afin de remonter ces données vers un serveur central. C'est le cas notamment pour la chaine de production des vaccins d'un célèbre institut français. [réf. nécessaire]

Fonctionnalités pour l'administrateur

En sus des fonctionnalités pour le développeur, SQL Server présente de nombreux outils pour faciliter l'administration. En voici quelques-uns.

Réplication de données

À ne pas confondre avec la « haute disponibilité » qui consiste à dupliquer l'intégralité d'une base de données, SQL Server dispose de 8 modes de réplication de données (réplication de certaines informations de certaines tables venant de bases de données vers d'autres bases sur d'autres serveur) parmi lesquelles les modes transactionnel, point à point, par cliché ou de fusion. La réplication de, et vers Oracle est aussi supportée dans l'édition Enterprise.

Les sites web de vente en ligne comme Cdiscount, Fnac.com ou Vente-privee.com utilisent la réplication, à la fois pour dissocier les données du FO (Front Office - essentiellement données du site web) et du BO (Back Office - gestion des factures, expéditions, stock...), mais aussi pour élargir la surface d'attaque afin d'absorber la charge des nombreux utilisateurs.

Haute disponibilité

La duplication des bases de données pour établir un système hautement résilient est assuré par quatre moyens différents :

- Le « ***log Shipping*** » par envoi de sauvegarde du journal de transaction et ré-application des transactions sur les bases de secours. Système à basculement manuel asynchrone. Pertes limitée à la latence d'envoi des journaux. Opère base par base.
- Le « ***mirroring*** » par envoi des pages modifiées puisées du journal de transaction vers une base de secours. Système transparent pour les applications (.net ou ODBC) à mode synchrone ou asynchrone à basculement manuel ou automatique (si synchrone) base par base. Latence de basculement nulle si synchrone et en mode automatique. Un seul miroir est toléré par base et pour le basculement automatique un serveur témoin (version Express gratuite par exemple) est nécessaire de façon à éviter le syndrome du « split brain ».
- Le « ***clustering*** » par redondance des serveurs (jusqu'à 64 nœuds passifs ou actifs) partageant une même baie de disque (*Share Nothing*), auquel on peut ajouter un système de réplication des IO de la baie afin d'éviter le SPOF (*Single Point Of Failure*). Permet de faire basculer l'instance en entier manuellement ou automatiquement suivant des règles à mettre en place. Données toujours synchrones. Temps de basculement dépendant du scénario établi, mais en général quelques dizaines de secondes à quelques minutes.
- La technologie « **AlwaysOn** » : combinant *mirroring* et Failover Cluster (FCI pour *Failover Cluster Instance*) permettant d'associer différentes bases au sein de groupes de disponibilité ce qui assure un basculement collectif de toutes les bases en cas de dysfonctionnement d'une d'entre elles au sein de groupe. De même que pour le mirroring, le système est transparent pour les applications (.net ou ODBC) à mode synchrone ou asynchrone à basculement manuel ou automatique (si synchrone) groupe par groupe. La latence de basculement est nulle si le mode synchrone est adopté. Le nombre de « réplicas » d'un même groupe de disponibilité est limité à 10 : 8 asynchrones et 2 synchrones. Un même groupe de disponibilité pouvant contenir plus d'une centaine de bases.

Gestion de la stratégie (politiques d'administration - policy management)

Il est possible de créer des règles de gestion et d'en planifier le fonctionnement. Il s'agit de vérifier un attribut d'une facette, via une règle qui peut être vérifiée de différentes manières. Quelques exemples:

- vérifier une fois par jour que chacune des bases a bien été sauvegardée
- forcer une règle de nommage pour les tables, les vues, les procédures
- interdire l'utilisation systématique du schéma SQL par défaut dbo

Cliché de base de données (database snapshot)

Ce procédé permet de créer une nouvelle base à partir d'une base existante qui représente les données à l'heure à laquelle le « cliché » a été lancé. Cette copie en, lecture seule, est effectuée instantanément quel que soit le volume de la base initiale. Cela permet par exemple :

- d'obtenir avec certitude l'ensemble des données de la base à une heure précise
- de pouvoir comparer en développement les données initiales et finale après l'exécution d'une procédure
- revenir en arrière soit par requête de modification inter-base, soit par restauration de la base depuis le cliché

Partitionnement

Présent depuis la version 2005 dans l'édition Enterprise, le partitionnement permet de découper tables et index suivant un critère de donnée de la table. la solution réutilisable est d'une grande simplicité et repose sur trois étapes :

- La création des valeurs pivots du partitionnement via une fonction de partitionnement (CREATE PARTITION FUNCTION...);
- La création d'un schéma de ventilation des partitions dans les différents espaces de stockage (CREATE PARTITION SCHEME...);
- La création de la table ou l'index exploitant le schéma de partitionnement à l'aide d'un critère qui doit être une colonne non NULL présente dans la définition de l'objet.

Une fois ces étapes accomplies la fonction et le schéma de partitionnement peuvent être réutilisées pour un autre objet à des fins d'alignement des partitions, comme ce peut être le cas d'une table des factures et de sa subordonnée table de détail ou de lignes de facture.

Il est ensuite possible de gérer les partitions (déplacer, fusionner, découper...) à chaud à l'aide des commandes ALTER PARTITION FUNCTION... SPLIT RANGE / MERGE RANGE et ALTER TABLE... SWITCH PARTITION... qui rendent la main immédiatement et agissent à bas niveau, en tâche de fond, pour opérer les déplacements.

Plan de maintenance

Microsoft SQL Server fournit de nombreux assistants pour effectuer des tâches souvent ingrates comme l'import/export de données ou la sauvegarde/restauration. Mais s'il en est un bien utile au débutant, c'est l'outil de gestion des plans de maintenance qui permet en quelques clics de réaliser l'ensemble des tâches indispensable à la survie et au maintien des performances d'une base. Cet outil permet entre autres d'effectuer de manière récurrentes, les travaux suivants :

- sauvegarde ;
- vérification d'intégrité physique ;
- défragmentation des index ;
- recalculs des statistiques.

... et d'en planifier l'exécution.

Le gouverneur de ressources

Parce que la cohabitation sur un même pied d'égalité de tous les utilisateurs d'une même base peut vite poser des problèmes, SQL Server intègre un gouverneur de ressources permettant d'allouer plus ou moins de ressources (RAM, CPU, disques...) à tel ou tel groupe d'utilisateurs. Le cas classique est le mélange des utilisateurs friands de tableaux de bords et états de toutes natures (par exemple les contrôleurs de gestion) très consommateurs de données, pénalisant la production des données des utilisateurs effectuant de la saisie unitaire...

Gestion de la qualité des données

SQL Server intègre deux modules distinct pour gérer des données de qualité dans les bases :

- **Data Quality Services** (DQS) : permet de piloter la qualité des données des bases relationnelles et décisionnelle de manière interactive et par l'intermédiaire d'une base de connaissance : nettoyage, correction, enrichissement, normalisation, déduplication...
- **Master Data Services** (MDS) : est la solution SQL Server de gestion des données de référence.

Service d'envoi d'email intégré

La messagerie de base de données est l'outil de gestion des envois de mails de SQL Server et repose sur Service Broker. Ce système permet de définir plusieurs profils d'envoi de mail (par exemple un pour les applications et l'autre pour l'administration) avec redondance possible des serveurs SMTP. L'envoi d'email est notablement utilisé par l'Agent SQL pour notifier parfois la réussite, mais souvent les échecs, des tâches planifiées (par exemple qu'une sauvegarde n'a pu se faire). La procédure msdb.dbo.sp_send_dbmail permet d'envoyer un mail, y compris avec pièces jointes, résultat de requête, contenu sous forme texte ou enrichi...

Les outils d'audit

SQL Server est riche en outils d'audit, que ce soit pour la sécurité, l'évolution des données ou les performances. Nous allons en décrire quelques-uns.

Audit de sécurité (Database Audit)

Outre l'audit C2 en voie d'obsolescence pour être remplacé par les critères communs (ISO 15408), SQL Server offre la possibilité de tracer tout ce qui relève de la sécurité par le biais de l'audit de base de données, et toutes les actions possibles aussi bien au niveau serveur qu'au niveau d'une base de données particulière.

Ce système est perfectionné au point qu'il est possible d'arrêter automatiquement le serveur en cas de défaillance du système d'audit, afin que des intrus ne profitent pas de cet état de dysfonctionnement de la traçabilité pour effectuer des manœuvres illicites !

Évolution des données

SQL Server dispose de deux modes permettant de tracer automatiquement l'évolution des données (INSERT, UPDATE, DELETE...) :

- CHANGE TRACKING : disponible dès l'édition Standard permet d'indiquer quelles lignes de quelles tables ont évolué dans un laps de temps cible et paramétrable.
- CHANGE DATA CAPTURE (CDC) : disponible dans l'édition Enterprise, copie les informations des mises à jour dans des tables de travail avec indication de la date et heure des changements effectués.

CHANGE TRACKING oblige à scruter les tables de la base, mais soulage la volumétrie globale, tandis que CDC n'interroge pas les données de production afin de soulage le verrouillage.

Ces deux modules sont généralement utilisés, soit l'un, soit l'autre, pour alimenter en mode différentiel de grand data warehouse.

Performances

La quantité d'outil est si nombreuse que beaucoup d'utilisateurs les ignorent, cherchant ailleurs dans des outils tiers - souvent payants - ce qu'ils ont déjà et gratuitement à portée de main.

Data Management View (DMV)

Ce sont des données système présentant des statistiques d'exécution du moteur SQL utilisable sous forme tabulaire par des pseudo vues (en fait la plupart du temps des fonctions internes retournant un état tabulaire des données collectées en mémoire). Cela permet d'établir différents diagnostics tels que :

- les temps d'attente : sys.dm_os_wait_stats
- l'activité en cours dans le serveur : sys.dm_exec_connections, sys.dm_exec_sessions, sys.dm_exec_requests;
- l'utilisation et la fragmentation des index : sys.dm_db_index_usage_stats, sys.dm_db_index_operational_stats, sys.dm_db_index_physical_stats;

- les index estimés manquants : sys.dm_db_missing_index_details, sys.dm_db_missing_index_groups, sys.dm_db_missing_index_group_stats;
- les traitements les plus couteux : sys.dm_exec_query_stats, sys.dm_exec_procedure_stats, sys.dm_exec_trigger_stats;
- les transactions (et leur corollaire les verrous) : sys.dm_os_tran_* (plus de 10 DMV)...
- la gestion du cache : sys.dm_os_memory_* (plus de 10 DMV)...
- la gestion des fichiers : sys.dm_db_file_space_usage, sys.dm_db_log_space_usage, sys.dm_io_virtual_file_stats, sys.dm_os_volume_stats...
- les threads : sys.dm_os_threads, sys.dm_os_tasks, sys.dm_os_workers, sys.dm_os_dispatcher_pools, sys.dm_os_nodes...

Rapports

Basés sur les DMV, des rapports exploitent et présentent les données de manière synthétique. Ils sont disponibles à différents niveaux de l'arborescence de l'explorateur d'objet de SQL Server Management Studio, principalement, au niveau serveur (instance) et base.

Compteurs de performance

SQL Server publie de nombreux compteurs de performances requêtables directement en SQL ou analysables via l'analyseur de performances (perfmon.exe). Ces compteurs sont lisibles dans la vue sys.dm_os_performance_counters.

Alertes

SQL Server permet de mettre en place à travers l'Agent SQL des alertes sur des erreurs ou des métriques franchissant un seuil en interrogeant des données de compteurs de performance ou via WMI (Windows Management Instrumentation). Ce type d'alerte permet par exemple d'être informé par email de la saturation des journaux de transaction ou le remplissage des disques du serveur.

Assistant de paramétrage du moteur de bases de données (Database Engine Tuning Advisor)

Cet outil permet d'effectuer un diagnostic de pose d'index, de statistiques, de partitionnement ou de création de vues indexées quand on lui fournit un lot de requêtes.

Profiler SQL

Le profiler SQL permet de renifler les requêtes avant et après exécution afin de fournir des métriques à des fins d'analyse essentiellement dans le but d'améliorer les performances (bien qu'il existe un mode "deprecated" permettant de voir le code obsolète de certaines applications).

Ces données sont enregistrées dans une table ou un fichier (transférable dans une table) et doivent ensuite être analysées. Différents templates permettent des niveaux de finesses de scrutation plus ou moins grand.

Le template "replay" permet de capturer une charge et ses métriques qui pourra être rejoué dans les mêmes conditions (simulation des différents utilisateurs en parallèle) à des fins de benchmark notamment pour des évolutions de versions de SQL Server, comme de hardware. Pour rejouer et comparer, il faut utiliser le package RML disponible gratuitement sur le site de Microsoft.

Événements étendus (XE : eXtended Events)

En complément et futur remplaçant du profiler, XE permet une intrusion plus profonde que le profiler puisqu'il peut aller scruter presque toutes les métriques des opérations en mémoire, dans les threads ou les IO des disques (niveau système). Cela en sus de ce que fait déjà le profiler.

Collecteur de données (Data Collector)

Parce que les données des DMV sont par principe volatiles (n'existant qu'en mémoire, elles sont perdues à chaque redémarrage) ou limitées (dans un anneau mémoire) il a été décidé de créer cet outil qui enregistre cycliquement différentes données de différentes sources (la plupart venant de DMV) à des fins de rétro-analyse par le biais de rapports spécifiques. Pour ce faire, une base de données de collecte doit être créée et vous devez lancer les différents collecteurs prédéfinis, ou bien créer vos propres collecteurs.

Distributed Replay

Ce dispositif permet de rejouer l'activité d'un serveur capturé par une trace du Profiler SQL, en simulant une activité proche de la réalité, par le biais d'une distribution des requêtes sur différents postes de travail le tout orchestré par un contrôleur. Le but étant par exemple de comparer la conformité d'exécution d'une application sur une nouvelle version de SQL Server ou lors d'un changement de machine (amélioration des performances).

Fonctionnement

Microsoft SQL Server propose plusieurs fonctionnements :

Langages

Pour les requêtes, SQL Server utilise le langage SQL dans un des dialectes les plus conformes à la norme SQL, Microsoft se faisant un point d'honneur à corriger de version en version quelques errements du passé hérité du moteur Sybase. Le dialecte utilisé est le T-SQL (Transact-SQL), une implémentation de SQL qui prend en charge les procédures stockées, les fonctions utilisateurs ou UDF (User Defined Function) et les déclencheurs (*trigger*). Les langages XQuery et XPath sont utilisés à différents niveaux pour manipuler les données XML au sein même des requêtes SQL.

Pour les transferts de données, SQL Server utilise le format TDS (Tabular Data Stream) qui a été implémenté dans d'autres bases de données (en particulier dans son homologue Sybase) et dont les spécifications sont publiques. Une implémentation Open Source d'un client TDS est disponible et constitue la base du client SQL Server du projet Mono : *FreeTDS*[2].

SQL Server étant doté de deux moteurs de bases de données, l'un relationnel et l'autre décisionnel, il est possible de faire des requêtes en langage MDX ou DMX spécifique à l'analyse de données pour les bases décisionnelles.

Liens d'intégrité

Dans le moteur relationnel de SQL Server, il est possible de définir des liens entre les tables par le biais de contraintes déclaratives de façon à garantir fortement l'intégrité des données entre table de référence et table fille. Ces liens d'intégrité peuvent être utilisés pour modifier ou supprimer en chaîne des lignes liées et supportent les règles d'évolution de type NO ACTION, CASCADE, SET DEFAULT et SET NULL.

Transactions

SQL Server est un SGBD transactionnel. Il est capable de préparer des modifications sur les données d'une base et de les valider ou de les annuler de façon atomique, c'est-à-dire en "tout ou rien". Cela garantit la cohérence et l'intégrité des informations stockées dans la base. Lors d'une transaction, les ensembles de données contenant les lignes de données en cours de modification par cette transaction sont verrouillés. Les autres utilisateurs doivent attendre la fin

de la transaction pour pouvoir les modifier à nouveau. En revanche, les utilisateurs concurrent peuvent éventuellement lire les données, mêmes verrouillées en modification par une autre transaction, en fonction du niveau d'isolation choisit.

Niveau d'isolation et verrouillage

Les verrouillages s'effectuent au niveau des lignes, pages, extensions, tables ou base de données. SQL Server ne verrouille que les ressources dont il a besoin (par défaut les lignes) et en fonction des besoins peut verrouiller à un niveau plus élevé (partition ou table). Cela évite aux utilisateurs d'attendre la fin d'une transaction pour mettre à jour des lignes de données qui n'ont pas été touchées par une modification et permet de diminuer la quantité de ressources consommées. Le verrouillage peut être pessimiste - les utilisateurs concurrents ne pourront accéder aux lignes de cette transaction - ou optimiste - dans ce cas les utilisateurs concurrent pourront accéder à la dernière version des lignes.

Pour assurer ce fonctionnement, SQL Server permet de piloter le niveau d'isolation de six manières différentes. Les 4 niveaux prévus par la norme SQL

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

... qui repose sur le verrouillage pessimiste et deux niveaux "optimistes" :

- SNAPSHOT
- READ COMMITTED SNAPSHOT

Le niveau d'isolation étant dynamique, il peut donc changer au cours d'une même transaction. Le niveau d'isolation READ COMMITTED SNAPSHOT opère exactement à la manière d'Oracle ou de PostGreSQL en substituant au niveau d'isolation READ COMMITTED un versionnement des lignes pour un verrouillage optimiste.

Journalisation des transactions

SQL Server permet de jouer les transactions de trois manières différentes :

- validation automatique : validation ou annulation (en cas d'erreur) immédiate de chaque ordre SQL lancé (par défaut) ;
- mode implicite : la connexion entame une transaction et la finalisation (COMMIT ou ROLLBACK manuel) termine la transaction pour en commencer une nouvelle (SET IMPLICIT_TRANSACTIONS ON) ;
- mode explicite : l'initialisation d'une transaction doit de faire par la commande BEGIN TRANSACTION.

Contrairement à Oracle ou PostGreSQL, SQL Server transactionne même les ordre du DDL (CREATE, ALTER, DROP...) et du DCL (GRANT, REVOKE...).

Il est possible de définir des points de contrôle de la transaction (commande SAVE TRANSACTION) afin de permettre une annulation partielle (ROLLBACK...).

Les transactions sont enregistrées dans le journal de transaction ("UNDO log"), les modifications des données sont immédiatement disponibles en mémoire et seront écrites dans les fichiers de données lors de points de contrôle (check point) lancés de manière périodique et asynchrone. Il est cependant possible de forcer un point de contrôle grâce à l'instruction CHECKPOINT.

Le journal des transactions enregistre différemment les données et cela de trois manières différentes :

- Mode complet : toutes les modifications sont enregistrées dans le journal. Les transactions terminées dont les données sont écrites sur le disque seront supprimées du journal de transaction lors des sauvegardes transactionnelles.
- Mode journalisé en bloc (BULK LOGGED) : journalise de façon minimale les opérations reproductibles (par exemple insertion de données de fichiers, construction d'index...). Ce mode permet d'accélérer le

fonctionnement du journal. Mais comme dans le mode complet, le journal devra être purgé par une sauvegarde transactionnelle.

- Mode simple : mode similaire au mode journalisé en bloc, à l'exception du fait que le journal est auto-purgé à chaque point de contrôle.

Dans les modes complet et journalisé en bloc, il faut sauvegarder le journal de transaction périodiquement ("REDO log"), sous peine de le voir grossir indéfiniment. Et grâce à la sauvegarde transactionnelle il est possible de restaurer la base telle qu'elle était à n'importe quel point du temps à la seconde près ou à la transaction près (avec cependant quelques limites pour le mode journalisé en bloc).

Il est possible d'imbriquer des transactions mais comme le concept de transaction est un état de la transaction, il n'est pas possible par ce biais de faire des annulations partielles. Il existe cependant d'autres moyens pour préserver une partie des données de l'annulation de la transaction, notamment par le biais des variables table.

SQL Server supporte les transactions distribuées via DTC (Database Transaction Coordinator) qui implémente le "commit à deux phases" conformément au modèle XA.

Fichiers

Les bases de données sont contenues physiquement dans des fichiers. Les fichiers portent généralement les extensions :

- MDF (Main Database File) pour le premier fichier de données, comportant certaines pages techniques vitales
- NDF (Next Database File) pour les autres fichiers de données
- LDF (Log Database File) pour les fichiers du journal de transaction

Les fichiers sont divisés en blocs de 8196 octets appelés pages et organisées par ensembles de 8 pages consécutives appelées extensions. Jusqu'à la version 2000, les lignes étaient limitées à une taille maximale de 8060 octets (les colonnes de type LOBs (image, text, ntext) n'étant pas comptabilisées dans cette limite). Depuis la version 2005 il est possible de dépasser largement cette limite et d'aller jusqu'à 2 milliards d'octets. En revanche la taille utile de la page est de 8096 octets.

Les bases de données ne peuvent fonctionner que lorsque tous les fichiers sont présents.

Les fichiers de données sont regroupés logiquement dans la base de données dans des **groupes de fichiers** qui constituent la destination des objets de la base (table, index...) Un groupe de fichier est considéré comme l'unité de stockage de toute table ou index et si ce groupe contient plusieurs fichiers, alors SQL Server répartit les données dans tous les fichiers (équivalent à un RAID 0), les opérations de lecture et d'écriture étant effectuées en parallèle dans les différentes fichiers, sous le contrôle direct du moteur de stockage de SQL Server. Ces fichiers et groupes de fichiers peuvent être sauvegardés de façon indépendante même s'il existe une interdépendance logique entre les objets d'un fichier et d'un autre (intégrité référentielle en particulier).

Sécurité

Du fait de son aspect multibase, SQL Server dispose d'une sécurité à deux niveaux : niveau serveur, par le biais des comptes de connexion, et niveau base, par le biais des utilisateurs SQL.

À partir de la version 2005, de profondes modifications de sécurité ont eu lieu dans la stratégie de SQL Server.

En sus, la stratégie de sécurité porte aussi sur les fonctionnalités du serveur. Par exemple, la procédure stockée *xp_cmdshell*[3] permettant de lancer des commandes systèmes est désactivée, et seul le DBA peut la réactiver. De même, dès ouverture du serveur, tous les fichiers de toutes les bases sont verrouillés en lecture comme en écriture afin d'empêcher tout accès indésirable.

Comptes de service

Les services exécutant chaque instance de SQL Server utilisent un compte de service (compte système pour l'exécution du service Windows). Ce dernier doit être choisi avec soin pour éviter d'éventuelles failles de sécurité sur le serveur. Celui-ci peut être : Service Système, Service Local (à partir de Windows 2003), Service Réseau (à partir de Windows 2003), compte utilisateur Windows local, compte utilisateur du domaine. Il est fortement conseillé de verrouiller les autorisations sur les comptes de service. Dans ce sens, et à partir de la version 2012, des comptes spécifiques aux autorisations très restreintes, sont créés lors de l'installation pour chacun des services de SQL Server (SQL Server, Agent SQL, SSAS, SSIS, SSRS...).

Comptes de connexion

Le compte de connexion permet de se connecter à une instance de SQL Server. À ces comptes, il convient de définir les privilèges qui s'appliquent au niveau serveur. Le privilège minimum étant le privilège "CONNECT". Un exemple de privilège de niveau serveur est la possibilité de créer toute base de données. Dans ce cas c'est le privilège CREATE ANY DATABASE.

SQL Server s'appuie par défaut sur le système d'authentification de Windows (Kerberos ou Natif) de qui permet d'accéder à une instance SQL Server par le biais d'un groupe Windows ou d'un utilisateur Windows. Lors de la connexion à la base de données, l'utilisateur est identifié grâce à son login Windows. Il n'a donc pas besoin de donner son mot de passe lors de la phase d'authentification au serveur.

Comme il existe des cas où l'utilisateur ne peut pas être identifié par son login Windows (utilisation de GNU/Linux, d'une page Web...) une méthode d'identification directe à SQL Server peut être mise en place. Elle doit l'être explicitement par l'administrateur de l'instance. Ceci suppose la création d'un nom de compte SQL et d'un mot de passe, et pour ce dernier on peut calquer la politique de conformation du mot de passe sur celle de Windows.

Lors de l'accès à une ressource extérieure, le processus SQL Server agit de 3 manières différentes : par emprunt d'identité lorsque la couche Windows est correctement configurée, l'utilisateur ne peut accéder par l'intermédiaire de SQL Server qu'aux ressources auxquelles il aurait droit s'il y accédait directement ; par le compte de service de l'instance, lorsque l'utilisateur est sysadmin et pour certaines tâches ; il ne permet pas l'accès, dans tous les autres cas. Dans le cas des comptes de connexion SQL, et pour l'accès à une ressource extérieure (par exemple l'envoi d'un fax dans une procédure stockée) SQL Server ajoute la possibilité de créer des informations de connexion qui sont des comptes Windows enregistrés à cet effet. Les utilisateurs SQL peuvent donc être associés à ces comptes Windows pour l'accès aux ressources externes.

Utilisateur SQL

Un compte de connexion peut être mappé à un utilisateur SQL soit dans une base particulière, soit dans plusieurs voire toutes. Le nom de compte de connexion et le nom d'utilisateur peuvent être différentes (et il vaut mieux qu'il le soit). Un utilisateur SQL dans une base peut être doté de privilèges qui l'autorisent à lancer différentes commandes sur divers objets. Les privilèges peuvent être établis directement sur un objet de la base ou bien sur un « conteneur » comme le schéma SQL ou la base tout entière.

Exemple d'attribution de différents privilèges de niveau schéma SQL à un utilisateur nomme USR_BANQUE :

```
GRANT SELECT, INSERT, UPDATE, EXECUTE
   ON SCHEMA::FINANCE
   TO USR_BANQUE;
```

Rôles

SQL Server dispose de rôles pré-établis au niveau serveur comme au niveau base. Il est même possible de créer ses propres rôles (nouveau pour le niveau serveur depuis 2012). Un rôle étant une collection de privilèges attribuable à un compte de connexion au niveau serveur ou à un utilisateur SQL quand ce rôle a été créé pour une base spécifique.

Schéma SQL

Un schéma étant un conteneur d'objet dans une base, SQL Server permet de définir autant de schéma SQL qu'on le souhaite, ceci afin d'organiser les différentes objets dans la base et surtout de gérer la sécurité au niveau de schéma de manière proactive. Ainsi la création de nouveaux objets, si ces objets sont placés dans les bons schémas, automatise la mise en place de la sécurité par héritage. Il n'y a donc aucune commande d'aucune sorte à passer d'un point de vue gestion de privilèges, pour que la sécurité soir immédiatement appliquée.

Enfin, et contrairement à certains SGBDR (Oracle par exemple) il n'y a pas d'intimité ni de confusion entre un schéma SQL et son propriétaire (l'utilisateur qui l'a créé) ce qui par exemple facilite grandement la migration d'un objet d'un schéma à l'autre et garantit une meilleure sécurité.

Exemple de migration d'une table d'un schéma SQL à l'autre :

```
ALTER SCHEMA BANQUE TRANSFER dbo.T_CLIENT;
```

Chiffrement

SQL Server permet de chiffrer les données des tables par différentes algorithmes intégrés à SQL Server. Les clefs de chiffrement et les certificats peuvent être importés depuis un fichier externe ou directement créés dans la base, SQL Server étant sa propre autorité de certification. Il existe cependant une troisième possibilité qui consiste à utiliser un boitier électronique de génération et conservation des clefs appelé HSM (Hardware Security Module), destiné à garantir l'inviolabilité des clefs. Ces équipements électroniques se placent sur le réseau et s'autodétruisent en cas de tentative d'accès illégal que ce soit physique ou logique.

Une sauvegarde de la base incorpore toutes les clefs et certificats pour assurer chiffrement et déchiffrement. Mais la restauration d'une base sur un autre serveur que celui d'origine impose un déchiffrement suivi d'un rechiffrement. Les éléments composant l'architecture de chiffrement étant interdépendants les uns des autres depuis une clef secrète détenue au niveau de l'instance. Il est cependant nécessaire, pour procéder au chiffrement ou déchiffrement d'ouvrir la clef soit en possédant un certificat, soit en donnant le mot de passe.

SQL Server permet de chiffrer les données des tables ou de chiffrer le code des routines.

Enfin, SQL Server permet de chiffrer le stockage des données et non plus les données par le biais de TDE (Transparent Data Encryption). Les fichiers (journaux de transaction, données des tables et index) sont chiffrés au moment de l'écriture physique sur les disques et déchiffrés au moment de la lecture physique des disques vers la mémoire. Ainsi les données figurent en clair en mémoire, ce qui permet de conserver les performances tout en chiffrant l'intégralité de la base.

Objets de la base de données

Tables

Les tables sont les objets qui contiennent effectivement les données dans la base. Elles sont de deux types :

1. Les **tables systèmes** contiennent les informations permettant le bon fonctionnement de la base de données. Par défaut sous SQL Server 2000 elles ne sont pas modifiables. Sous SQL Server 2005 de nouvelles tables systèmes apparaissent mais elles ne sont pas accessibles aux utilisateurs. Pour pouvoir toujours accéder à ces

informations l'équipe de développement a ajouté un grand nombre de vues systèmes. Celles-ci remplacent les anciennes tables systèmes et certaines commandes du DBCC. Un grand nombre de nouvelles vues systèmes a aussi été ajouté. Sous SQL Server tous les objets systèmes font partie du schéma *sys* et sont stockés dans une base de données appelée SystemResource.

2. Les **tables utilisateurs** contiennent les données des utilisateurs.

Une table peut contenir jusqu'à 30 000 colonnes dont la somme des tailles n'excède pas 2 milliards d'octets. Les colonnes de types BLOBs (VARCHAR(max), NVARCHAR(max), VARBINARY(max)) étant stockées en dehors de la ligne.

Publicité pour Microsoft SQL Server 2005 sur écran vidéo géant sur un bateau-mouche à Shanghai

Les tables peuvent être dotées d'une clé primaire. La clé primaire est toujours indexée. D'autres contraintes sont également disponibles au niveau des tables telles que : les contraintes uniques (automatiquement indexées), les valeurs par défaut, les contraintes de vérification (check), les clés étrangères.

Vues

Une vue est une requête nommée de la base de données. Elle est interrogée de la même façon qu'une table.

Les données renvoyées par la vue sont reconstituées à partir des données contenues dans les tables à chaque appel de la vue (sauf pour les vues indexées qui stockent les résultats). SQL Server remplace le nom des vues dans la requête qui va être exécutée par leurs définitions, puis la requête ainsi obtenue est compilée, optimisée et exécutée par le moteur. Le plan compilé de la requête ainsi obtenue est ensuite stocké pour permettre la réutilisation lors d'un prochain appel. Comme SQL Server dispose d'une optimisation sémantique en plus de l'optimisation statistique, les branches des jointures inutiles à l'exploitation d'une vue sont systématiquement coupées pour optimisation, lorsque cela est possible.

Dans certaines conditions, il est possible de mettre à jour les valeurs dans une vue. Voici des exemples de ces conditions:

1. Une vue sur plusieurs tables ne peut modifier qu'une seule de ces tables.
2. Si la vue contient des fonctions, utilise "DISTINCT" ou "GROUP BY", la modification ne fonctionnera pas.
3. Il ne doit pas y avoir transformation des données de colonnes (opération, appel de fonction...)
4. La vue possède un déclencheur de type INSTEAD OF (dans ce dernier cas, la logique de modification est celle figurant dans le code du déclencheur).

SQL Server implémente des types particuliers de vues qui sont les vues partitionnées et les vues indexées (équivalent des vues matérialisées synchrones d'Oracle).

Index

L'indexation dans SQL est assurée de quatre manières : l'indexation relationnelle, l'indexation verticale, l'indexation textuelle et l'indexation spécifique à certains types d'objets.

Indexation relationnelle

Elle porte sur le contenu intégral des colonnes. Le mécanisme interne de ces index repose sur la notion d'arbre équilibré (B-Tree ou Balanced Tree en anglais).

Il existe deux types d'*index* :

1. Les index clusterisés (CLUSTERED INDEX)
2. Les index non-clusterisés (NONCLUSTERED INDEX)

Il ne peut y avoir qu'un seul index clusterisé par table puisque ce type d'index est en fait un "tri" de la table. Les lignes de la table sont donc logiquement triées dans l'ordre de l'index clusterisé, ce qui explique qu'il n'y ne peut y en avoir qu'un seul. Autrement dit, l'index clusterisé étant un arbre-b, les feuilles de l'index sont directement les lignes dans la table.

Un index non clusterisé ne reflète pas l'ordre physique des lignes de la table.

Les deux types d'index sont construits à partir d'une clé, composée d'un certain nombre de colonnes. Ils permettent de retrouver rapidement une donnée à partir de tout ou partie des colonnes de cette clé.

Ainsi, si nous considérons une clé (A, B, C), on peut faire des recherches sur les champs (A), (A, B) et (A, B, C). Les champs sont considérés dans l'ordre de la clé ; il n'est donc pas possible de faire une recherche sur, par exemple (B, C) ou (C) avec cet index. En revanche une recherche sur (A, C) profitera aisément de l'indexation de la colonne A.

Une table peut contenir 999 index. Chaque index peut contenir 16 colonnes dont la somme des tailles n'excède pas 900 octets. Les index contiennent en plus des données des colonnes de l'index un signet (bookmark) vers les données de la table. Celui-ci peut être soit un pointeur vers une ligne, soit une des clefs de l'index clusterisé (en anglais : *index clusterised*).

L'optimisateur de SQL Server choisit quel index utiliser. Il est possible qu'il n'utilise pas l'index que vous avez créé car le coût de la recherche *via* cet index peut être plus grand que de lire la table au complet. Pour faire ces choix, l'optimisateur utilise entre autres les statistiques de la table et la présence ou non de contraintes. Il est possible par contre de forcer l'optimisateur à utiliser un index avec la clause WITH(nom_index), cette pratique n'est toutefois pas recommandée.

L'index est organisé en arbre-b. Ceci permet de classer plus rapidement les informations que s'il fallait les insérer dans une table séquentielle. Cette disposition étant très gourmande en ressources de stockage, la version 2005 de SQL Server permet de séparer les données de recherche et des données de traitement intégrées à l'index.

SQL Server permet de créer des colonnes calculées, qui, si elles sont persistantes (PERSISTED), peuvent être indexées.

Il n'y a pas d'autre structure d'index que l'arbre-b. Par exemple, on ne trouve pas d'*index bitmap*, qui permettraient de faciliter les recherches sur des champs aux contenus peu variés, mais il suffit de créer une table de référence pour simuler un index bitmap. De la même manière il n'y a pas d'index en hachage. Mais il suffit de créer une colonne calculée de hachage (fonction CHECKSUM) et de l'indexer pour obtenir un effet similaire. Enfin, il n'est pas possible de créer un index sur expression, mais il suffit de créer une colonne calculée comportant une telle expression et de l'indexer pour obtenir le même effet.

Indexation verticale

SQL Server permet de créer des index dit "verticaux" (columnstore index) sur un ensemble de colonnes permettant d'accéder à toutes les colonnes indexées sur le même plan. En effet, dans un index relationnel multicolonne, l'ordre de chaque colonne dans la clé de l'index n'autorise que certaines combinaisons de recherches. Ce n'est pas le cas avec l'indexation verticale, qui permet toutes combinaisons de recherches. Par exemple, avec la clef (A, B, C), il est possible de faire des recherches aussi bien sur (A), (B) ou (C), mais aussi sur toutes combinaisons de deux ou trois colonnes (A, B), (B, A), (A, C), (C, A), (B, C), (C, B), (A, B, C), (A, C, B), (B, A, C), (B, C, A), (C, A, B), (C, B, A).

Dans la version 2012, ces index sont en lecture seule et donc essentiellement destinés aux bases de données de type DataWarehouse pour le décisionnel. Dans la version 2014, ces index sont modifiables.

Indexation textuelle et sémantique

L'indexation textuelle (en fait un catalogue d'indexation) permet de faire des recherches en texte clair ("full text", conforme à la norme SQL avec le prédicat CONTAINS) ou à la manière de Google à l'aide du prédicat FREETEXT. Il est permis de rechercher différents mots dans un même document (texte des colonnes d'une table ou document électronique stocké dans une table) par égalité, début, proximité, mais aussi par forme fléchie (conjugaison, pluriels, féminin, etc.), synonymisation, extension (notamment pour les sigles)... ainsi qu'avec des poids relatifs (fonction table CONTAINSTABLE et FREETEXTTABLE).

L'indexation textuelle peut porter sur des colonnes de type chaîne de caractères ou XML, mais aussi sur des fichiers électroniques qu'ils soient stockées directement en table (dans un BLOB) ou à titre de fichier par le biais du FILESTREAM ou dans un FILETABLE. SQL Server reconnait plus de 50 formats de fichier nativement et permet de rajouter de nombreux autres types de fichiers électroniques à l'aide des "ifilter".

SQL Server supporte la plupart des langues et permet des recherches combinées multilingues. Pour cela des listes de "mots noirs" peuvent être établies pour chacune des langues indexées. De plus SQL Server indexe tous les mots sans exception y compris les mots noirs, là ou certains SGBDR se limitent à des mots ayant une longueur minimale (MySQL par exemple) ou excluent les mots noirs (PostgreSQL par exemple). Enfin, si la casse n'a pas d'importance pour l'indexation textuelle, l'utilisateur peut choisir s'il veut une indexation textuelle prenant en compte ou pas les caractères diacritiques, tels que les accents ou les ligatures. L'indexation textuelle permet aussi la recherche de métadonnées dans les fichiers électroniques, par exemple l'auteur d'un fichier Word.

Avec la version 2012, SQL Server a rajouté l'indexation sémantique qui permet, comme son nom l'indique de faire des recherches sémantiques. Cette fonctionnalité nécessite l'ajout d'une base de données de traitement sémantique destinée à établir des statistiques sur les textes indexés. la recherche sémantique porte sur la signification des documents. Ceci permet l'extraction automatique de balises, la découverte de contenu connexe et la navigation hiérarchique à travers des documents ayant un contenu similaire à un texte de référence. Un exemple pratique consiste à interroger l'index de ressemblance de document pour identifier dans un lot de curriculum vitæ, ceux qui correspondent le mieux à une description de poste.

Indexation des objets

SQL Server permet l'indexation de certains types de données objet, comme le XML (quatre types d'index), ou le spatial (types *geometry* et *geography*). Comme il est possible de rajouter vos propres objets, à condition de les décrire sous forme .NET, il est aussi possible de les indexer, à condition pour que vous vous arrangiez qu'ils soient « *bit ordered* » dans l'organisation de leur stockage.

Procédures stockées

Il est possible de définir des procédures stockées codées en Transact-SQL comme en .NET (SQL CLR). Une procédure stockée est une suite d'instructions qui peut modifier les données de la base et encapsuler des transactions et qui renvoient une ou plusieurs valeurs.

Les procédures stockées sous SQL Server peuvent prendre en paramètre et/ou retourner des entiers, des chaînes de caractère, des dates, des curseurs, des tables, des tables virtuelles et tout autre type défini dans SQL Server par défaut ou par les utilisateurs.

Les principaux avantages de l'utilisation des procédures stockées sont :

- la transaction de traitements complexes (plusieurs transactions peuvent s'enchainer ou s'imbriquer)
- la sécurité accrue (le code d'une procédure peut être dépersonnalisé)
- une meilleure réutilisation des plans compilés
- la possibilité de gérer les dépendances entre le code SQL et les objets du moteur.

Déclencheurs (trigger)

Le déclencheur constitue du code stocké qui s'exécute après (AFTER) ou à la place (INSTEAD OF) d'une action particulière (insertion, modification, suppression) sur une table ou vue. Il est possible de coder autant de déclencheurs AFTER qu'on le souhaite sur une même action (INSERT, UPDATE ou DELETE) d'une même table ou vue. En cas de pluralité de déclencheurs sur une même action d'une même table, l'ordre des déclencheurs est arbitraire, sauf pour les 3 premiers que l'on peut imposer. Les déclencheurs INSTEAD OF doivent être uniques par événement/table.

Deux tables virtuelles sont créées pour manipuler les données insérées, modifiées ou supprimées : *inserted* (pour l'insertion et la modification) et *deleted* (pour la suppression et la modification). Il est impossible de modifier le contenu de ces tables virtuelles. En revanche, SQL Server permet de modifier la table cible du déclencheur, comme d'annuler la transaction implicite qui a déclenché le trigger par annulation de la transaction (ROLLBACK).

Le déclencheur particulier *INSTEAD OF* est utilisable lorsque l'on souhaite intercepter l'événement, ne pas le laisser se dérouler, mais coder une autre action. Ce type de déclencheur est utilisable par exemple dans le cadre d'un mapping relationnel/objet où l'on manipule des vues composées de multiples jointures où l'on souhaite que l'utilisateur puisse faire des mises à jour (INSERT, UPDATE, DELETE) directement sur ces vues. Dans ce cas, les données INSERT, UPDATE ou DELETE sont découpées pour chacune des tables en jeu et une série d'ordres spécifiques à chaque table à mettre à voir est lancée au travers une transaction.

Depuis SQL Server 2005, la possibilité de créer des triggers DDL a été ajoutée. Ces déclencheurs DDL agissent après apparition des événements de type CREATE, ALTER ou DROP, et ces événements peuvent être interceptés au niveau du serveur (par exemple un CREATE DATABASE) ou au niveau de la base (par exemple un DROP TABLE). Pour obtenir des détails sur l'événement intercepté, un paquet de données XML (que l'on obtient par la fonction EVENTDATA()) donne de multiples précisions sur l'origine et la nature de l'événement. On y trouve même la requête SQL qui l'a déclenché.

Un type particulier de déclencheur permet d'intercepter les connexions au serveur (trigger FOR LOGON).

Fonctions (UDF : User Defined Function)

Depuis SQL Server 2000, il est possible de créer des fonctions en Transact SQL et depuis 2005 en .net (SQL CLR).

Ces fonctions sont de 3 types :

- Scalaire : Plusieurs instructions renvoient alors une valeur de type simple
- Table à instructions multiples : plusieurs instructions renvoient alors une table
- Table en-ligne (vue paramétrée) : une instruction de type SELECT renvoie une table

Il existe cependant des restrictions quant aux fonctionnalités utilisables dans le corps d'une fonction. Il n'est pas possible d'utiliser de fonctions non déterministes et la fonction ne doit pas modifier des données ou des paramètres système de manière permanente ou durable. Ce qui interdit par exemple l'utilisation d'un INSERT sur une table ou l'utilisation de la fonction GETDATE().

SQL Server 2005 a rajouté :

- la possibilité de créer des fonctions d'agrégation (calcul statistique) par l'intermédiaire de CLR, c'est-à-dire d'un codage à l'aide d'un langage de la plate-forme .net ;
- des fonctions particulières liées au partitionnement des données (PARTITION FUNCTION).

À la différence d'une procédure stockée, une fonction ne peut pas contenir :

- d'ordre déterminant les frontières d'une transaction (BEGIN TRANSACTION, COMMIT, ROLLBACK)
- d'ordre de mise à jour des données (INSERT, UPDATE, DELETE, SELECT), sauf pour les fonctions table multi-instructions et ce uniquement pour la table de retour.
- de code non déterministe (barrière partiellement levée avec SQL Server 2005, les fonctions telles que GETDATE étant autorisées, mais RAND est toujours interdite)
- de SQL dynamique
- de curseur
- d'un appel de procédure stockée

C'est logique, car le but d'une fonction est avant tout d'être utilisée au sein d'une requête.

Aspects objet

Depuis la version 2005, SQL Server incorpore des aspects objets parmi lesquels on trouve :

- la possibilité de réaliser des types complexes structurés via un langage externe de type .net et de les utiliser aussi bien en tant que colonne de table qu'en tant que type d'une variable dans une routine (procédure, déclencheur ou fonction).
- le type XML interrogeable via XQuery/XPath et indexable. SQL server intégrant un parseur XML et des procédures de manipulation de flux XML.
- la possibilité de générer un flux XML à partir du résultat d'une requête SELECT à la place d'un "dataset".
- le type hierarchyId (apparu en version 2008) facilitant la structuration des données hierarchique (arbres).
- un véritable SIG (Système d'Information Géographique - apparu en version 2008) avec l'arrivée des types GEOMETRY et GEOGRAPHY au standard OGC avec l'interfaçage sous forme WKT (Well known text) ou WKB (Well Known Binary), et dotés de nombreuses méthodes de manipulation.
- l'implantation de services Web que l'on peut créer directement dans le SGBDR.
- le stockage de fichiers sous la responsabilité exclusive ou partielle du SGBDR (FILESTREAM) et leur manipulation transactionnelle permettant une sauvegarde cohérente des données relationnelles et non relationnelles (très utile pour une GED par exemple) - @À partir de 2008.
- la représentation transparente sous forme de tables (FILETABLE) des arborescences de stockage des fichiers de l'OS Windows (répertoires et fichiers) et leur manipulation par des requêtes SQL - à partir de 2012.

Outils

SQL Server Management Studio (SSMS)

Cet outil est fourni avec les versions payantes de SQL Server. Il permet de se connecter et d'administrer les différents moteurs SQL Server (SSRS, SSIS, SSAS et le moteur relationnel). Il permet pour le moteur relationnel de développer des scripts TSQL, avec la possibilité de regrouper l'ensemble de ceux-ci au sein d'une solution (comme sous Visual Studio). On peut aussi enregistrer ceux-ci grâce à Visual SourceSafe ou Team Foundation Server. Pour la version Express (gratuite) l'outil simplifié SQL **Server Management Studio Express Edition** (SSMSE) est disponible gratuitement.

Avant 2005 cet outil était séparé en deux modules appelés "Enterprise Manager" et "Query Analyzer".

SQL Server Business Intelligence Development Studio / SQL Server Data Tools

Tout comme SSMS, cet outil est fourni avec les versions payantes de SQL Server. Il s'agit d'une surcouche de l'outil générique de développement Microsoft Visual Studioadapté pour la création de projets BI (Analysis Services, Integration Services ou Reporting Services). Tous ces projets se retrouvent dans le groupe "Projets Business Intelligence".

Profiler (générateur de profil)

Cet outil permet de capturer l'activité d'une base de données. Il permet aux administrateurs de vérifier les éléments et les requêtes qui prennent du temps sur la base de données. Éventuellement, il est possible de rejouer la capture sur un serveur de test. Il a tendance à être progressivement remplacé par le système des "événements étendus" (eXtended Events).

Services et moteurs

En fait MS SQL Server est une suite composée de cinq services principaux :

- Le moteur relationnel (OLTP) appelé **SQL Server ;**
- Le moteur décisionnel (OLAP) appelé **SSAS** (*SQL Server Analysis Services*) incluant un moteur de stockage pour les cubes, des algorithmes de forage (*data mining*) et différents outils de BI (Business Intelligence) ;
- Un ETL (Extract Transform and Load) appelé **SSIS** (SQL Server Integration Services) destiné à la mise en place de logiques de flux de données, notamment pour alimenter des entrepôts de données (data warehouse) ;
- Un outil de génération d'état appelé **SSRS** (SQL Server Reporting Services) permettant de produire des rapports sous différentes formes et exploitant les ressources du moteur décisionnel (bases "resportServer...") à la fois pour y stocker les rapports mais aussi y cacher les données de ces derniers afin de faire du "warmup" ;
- Un système de planification de travaux et de gestion d'alerte appelé **Agent SQL** qui utilise lui aussi les services du moteur SQL (base msdb).

D'autres services lui sont associés pour certains besoins :

- **SQL Full-Text Filter Daemon launcher** : pour charger certains documents (thésaurus et dll ifilter) externe nécessaires à la recherche "plain texte"
- **SQL Browser** : service de diffusion des services SQL destiné à faciliter la recherches des services SQL sur le réseau
- **MS DTC** (Data Transaction Coordinator) : coordinateur de transactions distribuées, utilisé pour des mises à jour en validation à deux phases (2PC) combinant différents serveurs transactionnels, pas forcément du monde Microsoft (XA, X/Open, TIP, WS...)

SQL Server inclut aussi de nombreux outils de développement :

- pour les bases transactionnelles, par le biais du **SQL Server Management Studio** (SSMS)
- pour les bases décisionnelles, le forage des données, le reporting et l'ETL, par le biais d'une surcouche de Visual Studio appelé **BIDS** (Business Intelligence Development Studio) puis **SQL Server Data Tools** depuis la version 2012.

Plus d'une centaine d'outils périphériques sont disponibles, soit lors de l'installation (SQL Profiler, Database Tuning Advisor, Data Collector...), soit directement sur le site de Microsoft (SQLdiag, SQLioSim, SQL Server Best Practices Analyzer...), soit à travers le site communautaire opensource Codeplex (RML Utilities, PAL, Open DBDiff...).

Une instance de SQL Server est une installation de tout ou partie des services SQL Server sur une machine Windows et peut héberger de nombreuses bases de données. Un même OS supportant jusqu'à 50 instances différentes (ce qui n'est pas conseillé en production).

SQL Server 2014 n'est pas limité en nombre de cœurs ou de RAM, que par la capacité de Windows. Une base est cependant limitée à 524 pétaoctets (Po), mais comme SQL Server est multibase et peut héberger 32760 bases dans une même instance, la capacité globale du système permet de prendre en charge théoriquement 17 166 240 Po de données. Avec 50 instances sur une même machine, cette limite théorique est repoussée à 858 312 000 Po...

SQL Server existe en différentes éditions : CE (Compact Edition - solution embarquée pour les smartphones), Express (plusieurs déclinaisons - gratuite), Web (en mode SPLA : *Service Provider License Agreement* - auprès d'hébergeurs Web), Standard, BI et Enterprise - ceci pour la production. L'édition Developper (équivalente à l'édition Enterprise) est destinée au développement.

SQL Server

Il s'agit du moteur de bases de données. À chaque instance de SQL Server correspond un service Windows Sql Server. Ce service peut être repéré dans les services Windows sous les noms **MSSQL$Nom_instance** pour les instances nommées et **MSSQLServer** pour l'instance par défaut.

Chaque instance créée possède au départ 4 à 6 bases de données systèmes. Le moteur SQL Server s'appuie sur la base de données système **master** qui contient la définition des autres bases de données.

Les autres bases systèmes sont :

- **msdb** : utilisée par l'agent SQL Server, la réplication, Data Transformation Services, Integration Services...
- **model** : utilisée comme modèle pour créer de nouvelles bases de données. Toute nouvelle base de données est une copie de celle-ci.
- **tempdb** : Base de données temporaire, elle sert à stocker les données temporaires créées par des utilisateurs ou générées par des curseurs, des créations d'index, lors d'un tri volumineux. Sous SQL Server 2005, elle contient aussi les versions d'enregistrement générées par les niveaux d'isolation SNAPSHOT, le contenu des pseudo tables inserted et deleted utilisées dans les triggers et celles générées par les opérations d'index en ligne. Cette base de données système est systématiquement vidée au démarrage du service SQL Server.
- **distribution** : N'est présente que dans le cadre d'une réplication si votre serveur joue le rôle du distributeur.
- **mssqlsystemresource** : Présente uniquement à partir de SQL Server 2005. Elle n'est pas accessible directement, et elle contient tous les objets systèmes (hors tables).

SQL Server Agent

Il s'agit de l'agent de maintenance de l'instance de SQL Server. À chaque instance de SQL Server correspond un service Windows Sql Server Agent. Ce service peut être repéré dans les services Windows sous les noms **SQLAgent$Nom_instance** pour les instances nommées et **SQLAgent** pour l'instance par défaut. Le moteur SQL Server Agent s'appuie sur la base **msdb**. Ce moteur permet de gérer les sauvegardes, les plans de maintenance, les travaux planifiés, la surveillance de la base, les alertes administratives...

L'Agent a aussi comme rôle la surveillance du service de SQL Server et le déclenchement automatique du redémarrage de celui-ci en cas d'arrêt inattendu.

Full Text Search

Moteur d'index / recherche de texte intégral.

Distributed Transaction Coordinator

Connu aussi sous le nom de MS DTC, sert à gérer les transactions distribuées. C'est-à-dire les transactions entre plusieurs serveurs de transactions SQL Server, entre un serveur SQL Server et des serveurs de base de données autres, entre plusieurs sources de données différentes, qu'ils soient des moteurs de base de données ou de simples composants.

Notification Services

Service apparu sous SQL Server 2000 permet de requêter régulièrement la base de données et en fonction de ces requêtes de notifier des groupes abonnés à ces évènements. Ce service n'est plus fourni en standard depuis SQL Server 2008. SQL Server nom de code Denali, apporte néanmoins une fonctionnalités similaire dans SQL Server Reporting Services et Sharepoint 2010, connu sous le nom d'Alerting.

Décisionnel

La plate-forme décisionnelle SQL Server se compose d'un ETL, apparu avec SQL 7, d'un moteur multidimensionnel, également apparu sous SQL 7 et d'un moteur de rapports, ajouté en 2004 sous SQL 2000. Avec la version SQL Server 2005, un studio de développement destiné au métier du décisionnel a été intégré dans Visual Studio 2005 : BI development Studio.

Cartographie

SQL Server 2008 supporte de nouveau type de données primitives : le type géographique où intervient une projection (latitudes et les longitudes) et le type géométrique sans projection pour manipuler les données relatives aux points, aux lignes et aux polygones. Le premier logiciel compatible SQL Server a été le logiciel Manifold System `(en)` puis Mapinfo, Geomédia.

Integration Services (SSIS : SQL Server Integration Services)

Service apparu sous SQL Server 2005, il est le remplaçant de Data Transformation Services (DTS). L'ETL a été complètement reconstruit, et se positionne en concurrence des autres outils ETL professionnels du marché.

L'objectif est l'intégration de l'entreprise par ses données. SQL Server Integration Services (SSIS) est une plate-forme complète d'intégration de données. SSIS, n'est pas un simple ETL. C'est une plate-forme complète d'intégration de données, offrant un certain nombre d'outils graphiques de développement et de gestion, de services, d'objets programmables et d'interfaces API (Application Programming Interfaces). SSIS contient un moteur de workflow prenant en charge une logique complexe et peut être utilisé pour un large éventail d'opérations de maintenance de base de données et d'opérations sophistiquées de transfert de données. L'architecture SSIS est essentiellement composée de deux parties : le moteur d'exécution de transformation de données, qui gère le flux de contrôle d'un package, et le moteur de flux de données ou moteur de pipeline de transformation de données, qui gère le flux des données à partir des sources de données, via les transformations et jusqu'aux cibles de destination.

Analysis Services

SQL Server Analysis Services (SSAS)[4,5] est un service apparu sous SQL Server 7, connu à cette époque sous le nom de OLAP Services. Il permet de générer des cubes OLAP, données agrégées et multidimensionnelles. Il permet également d'implémenter des algorithmes de Data Mining.

Reporting Services

Ce service[6] apparu sous SQL Server 2000 est un moteur de génération d'états. Deux services web le composent, l'un permettant son administration, l'autre la génération, l'abonnement, le rendu des rapports. Les rendus se font sous Excel, PDF, HTML, et Word (à partir de la version 2008).

Éditions

Datacenter Edition

Apparue avec SQL Server 2008 R2, elle supportera jusqu'à 256 processeurs logiques, ainsi que la virtualisation illimitée et une quantité de mémoire illimitée.

Enterprise Edition

Elle supporte un nombre de processeurs et une taille de mémoire vive illimités (limités par le système d'exploitation). Il existe une version 32 bits (x86) et 2 versions 64 bits (ia64 et x64, x64 uniquement pour SQL Server 2005) et n'a pas de limite quant à la taille des bases de données. Elle ne peut s'installer que sur un Windows version serveur. Inclut toutes des fonctionnalités du moteur, dont les fonctions de haute disponibilité. Les fonctionnalités pour le décisionnel sont toutes incluses de la génération d'états avec Reporting Services à l'utilisation de cubes OLAP avec Analysis Services en passant par le transfert de données avec Integration Services (SQL Server 2005) ou Data Transformation Services (SQL Server 7 & 2000). Elle peut en outre fonctionner en cluster jusqu'à 8 nœuds.

Developer Edition

Il s'agit d'une édition pour les développeurs qui dispose des mêmes fonctionnalités que l'édition Entreprise. Cependant la licence contient des restrictions quant à son utilisation. La licence interdit une exploitation en production. Elle est exclusivement destinée à une utilisation en Test. Cependant il n'y a aucune restriction sur les produits disponibles.

Standard Edition

Elle supporte jusqu'à 4 processeurs et une taille de mémoire vive illimitée (limitée par le système d'exploitation). Il existe une version 32 bits (x86) et 2 versions 64 bits (ia64 et x64, les deux uniquement pour SQL Server 2005) et n'a pas de limite quant à la taille des bases de données. Elle ne peut s'installer que sur un Windows version serveur. Cependant SQL Server 2005 permet son installation sur Windows de bureau. Inclus toutes des fonctionnalités du moteur, sauf certaines fonctions de haute disponibilité. Les fonctionnalités pour le décisionnel sont toutes incluses de la génération d'états avec Reporting Services (pour SQL Server 2000, ce composant est à charger séparément) à l'utilisation de cubes OLAP avec Analysis Services en passant par le transfert de données avec Integration Services (SQL Server 2005) ou Data Transformation Services (SQL Server 7 & 2000). Elle peut en outre fonctionner en cluster jusqu'à 2 nœuds depuis sa version 2005.

Workgroup Edition

Nouvelle édition apparue avec SQL Server 2005. Elle supporte jusqu'à 2 processeurs et 3 Go de mémoire vive. Pour le moment, elle n'existe qu'en version 32 bits et n'a pas de limite quant à la taille des bases de données. Inclus la majeure partie des fonctionnalités du moteur, dont la possibilité de participer à une réplication en tant qu'éditeur. Les fonctionnalités pour le décisionnel incluses se limitent à la génération d'états avec Reporting Services. Cette édition disparaît avec la version 2012 de SQL Server[7].

Web Edition

Nouvelle édition apparue sous SQL Server 2008. Proche de l'édition Standard en termes de fonctionnalité, elle ne supporte cependant ni la mise en miroir, ni les composant décisionnel de SQL Server. Sa licence ne permet son utilisation que comme serveur de base de données pour un site ou service Web. Elle est disponible en abonnement mensuel spécialement pour les herbergeurs en plus d'être disponible plus classiquement à l'achat.

Personal Edition

Édition existant avec SQL Server 7 et SQL Server 2000. Elle supporte jusqu'à deux processeurs et 2 Go de mémoire vive. N'existe qu'en version 32 bits et n'a pas de limite quant à la taille des bases de données. Inclut la majeure partie des fonctionnalités du moteur, dont la possibilité de participer à une réplication en tant qu'abonné. Elle n'a pas de fonctionnalités pour le décisionnel. En termes de licence, le fait de posséder une licence d'accès client (CAL) SQL Server suffit à utiliser cette édition.

Windows Internal Database / SQL Server Embedded

Similaire à SQL Server edition Express, avec cependant les limites de mémoire, de nombre de processeurs et de taille de base de données supprimés, elle est cependant utilisable que localement et ne peut héberger que des bases de données d'application Microsoft. Des produits tels que Sharepoint ou certaines fonctionnalités Active Directory de Windows s'en servent.

MSDE / Express Edition

Microsoft a édité une édition gratuite composée uniquement du moteur de base de données relationnel, bridé à 4 Go d'espace disque (10 Go dans la version 2008 R2). Elle permet, néanmoins de créer jusqu'à 32760 bases différentes soit plus de 320 To de données à partir de la version 2008 R2.

Compact Edition

Appelée auparavant : Pocket PC / Mobile / Everywhere Edition. Il s'agit d'une édition légère adaptée à l'utilisation sur des PDA ou smartphones munis de Windows Mobile. La version Compact (version 3.5), qui succède aux éditions Mobile s'ouvre elle aux postes de travail classiques à base de Windows. Cette édition peut participer à une réplication en tant qu'abonné.

Licences

SQL Server supporte 5 systèmes de licences :

- **Licence par utilisateur** : SQL Server peut utiliser tous les processeurs du serveur (jusqu'à la version 2008 R2, maximum 16 (édition Standard) ou 20 (édition Entreprise) Cores dans la version 2012) mais est limité au nombre d'utilisateurs spécifié. Chaque personne physique se servant d'une application utilisant SQL Server est considéré comme un utilisateur de la base de données
- **Licence par périphérique** (depuis SQL Server 2005) : SQL Server peut utiliser tous les processeurs du serveur (jusqu'à la version 2008 R2, maximum 16 (édition Standard) ou 20 (édition Entreprise) Cores dans la version 2012) mais est limité au nombre de périphériques spécifié. Chaque périphérique physique accédant directement ou indirectement à SQL Server est considéré comme un utilisateur de la base de données
- **Licence par processeur** : SQL Server utilise le nombre de processeurs spécifiés dans la licence et peut accepter un nombre indéfini d'utilisateurs. Ce mode de licence est valable jusqu'à l'édition 2008 R2 et disparaît après.
- **Licence par cœur** : Il faut acheter autant de licences "Core" qu'il n'y a de cœur dans le serveur ou dans la machine virtuelle. Le nombre d'utilisateurs est illimité. Ce mode de licence n'est valable qu'à partir de l'édition 2012[7]. Le prix est compté par paires de cœur et démarre à 4.
- **Licence SPLA** : pour l'édition Web, il s'agit d'une licence sous forme locative. Le prix modique permet de concurrencer des SGBDR « libre » comme MySQL ou PostGreSQL.

Les types de licence envisageable pour une utilisation de SQL Server derrière un frontal Web est soit la licence par processeur ou cœur soit le mode SPLA limité à l'édition Web.

Outils complémentaires intégrant un serveur SQL server

Certains outils connexes à SQL Server méritent d'être cités...

BizTalk

Solution Microsoft pour l'EAI (Enterprise Application Integration) et l'intégration de données B2B (business-to-business).

Sharepoint

Solution globale de gestion de la connaissance d'entreprise (KB - Knowledge base) permettant la gestion de contenu (via site web et forum) et de documents électronique partagés, la recherche d'information, ainsi que la possibilité de créer des formulaires et la présentation de statistiques décisionnelles tel que tableaux de bord, indicateurs clef (KPI - Key Performance Indicator)...

Dynamics

C'est l'ERP (Enterprise Resource Planning) de Microsoft.

TFS (Team Foundation Server)

Est une forge logicielle permettant la gestion des code sources, des builds, le suivi des ressources, la planification des tâches...

SCOM (System Center Operations Manager)

Système de gestion centralisée, de supervision et de maintenance du système informatique (serveurs, PCs, périphériques en ligne) et de l'infrastructure réseau.

WSUS (Windows Server Update Services)

Service de distribution concernant les OS et applications Microsoft pour la mise à jour des différentes machines Windows d'un parc informatique (utilise la version Express gratuite)

Outils complémentaires pouvant utiliser un serveur SQL server

StreamInsight

StreamInsight est une plate-forme prévue pour développer et déployer à grande échelle des applications de traitement des événements complexes (Complex Event Processing - CEP). Elle est basée sur deux composants essentiels : le StreamInsight Serveur pour la gestion de flux des données et StreamInsight Framework pour la création des interfaces.

EDF utilise StreamInsight pour la gestion des incidents de l'ensemble de son parc informatique.

SQL Server et le Big Data

Très engagé dans ce mouvement, notamment par le biais de Hadoop, Microsoft propose différentes solutions pour gérer le big data.

HDInsight (Hadoop)

HDInsight est une solution Hadoop proposée sur le cloud Azure ou localement (à des fins de développement).

Les outils Map/Reduce, Hive, Sqoop et le langage Pig facilite le transfert et la manipulation des données depuis SQL Server vers Hadoop. Un SDK basé sur le framework .NET permet d'écrire des batchs en utilisant Visual Studio. Une installation locale d'Hadoop destinée aux développeurs est disponible au travers de HDInsight Server.

Polybase

Polybase est un outil destiné à attaquer différentes sources de données relationnelles (SQL Server), décisionnelles (SQL Server Parallel Data Warehouse - PDW) ou encore big data, à travers différents processus comme Map Reduce ou MPP (Massive Parallèle Processive)

Références

1. https://docs.microsoft.com/en-us/sql/linux/quickstart-install-connect-docker
2. FreeTDS (ftp://ftp.astron.com/pub/freetds/)
3. Par défaut, sur SQL Server 2014, il n'est pas possible de lancer une commande système à partir de SQL Server (https://msdn.microsoft.com/fr-fr/library/ms190693.aspx)
4. https://technet.microsoft.com/fr-fr/library/ms175609%28v=sql.90%29.aspx
5. http://www.microsoft.com/france/serveur-cloud/sql/2012/business-intelligence.aspx#analysis
6. http://www.microsoft.com/france/serveur-cloud/sql/2012/business-intelligence.aspx#reporting
7. SQL Server 2012, quoi de neuf ? (http://www.mslicensing.fr/actualite/sql-server-2012-quoi-de-neuf)

Voir aussi

Articles connexes

- Microsoft SQL Server Desktop Engine
- SQL Server Express

Sur les autres projets Wikimedia :

Microsoft SQL Server (https://commons.wikimedia.org/wiki/Category:Microsoft_SQL_Server?uselang=fr), sur Wikimedia Commons

Microsoft SQL Server, sur Wikibooks