2012/2013

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU
FACULTE DE GENIE ELECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDES

En vue de l'obtention du diplôme de master académique

Option : Conduite de projets informatiques

Thème :

Conception et réalisation d'une base de données répartie sous SQL Server 2008

Cas : Entreprise NWTRADERS ,citée dans les TP de Microsoft

Proposé et dirigé par :
Mr RAMDANE

Réalisé par :
Melle BANDOU Katia

Remerciements

Je présente mes plus vifs remerciements à mon promoteur Mr RAMDANE et qu'il trouve à travers ces quelques lignes l'expression de ma profonde gratitude pour son soutien, ses encouragements, ses orientations ainsi que ses conseils tout au long de mon travail.

Enfin, je tiens à présenter ma reconnaissance éternelle aux corps administratif et professoral de l'université Mouloud Mammeri de Tizi Ouzou.

Dédicaces

Rien n'est aussi beau à offrir que le fruit d'un labeur qu'on dédie du fond du cœur à ceux que l'on aime jusqu'aux frontières de l'imagination ;

Ma chère mère, mon cher père, sans eux, je n'aurais pas abouti à ce stade d'études. Que dieu puisse m'aider à les honorer, les servir et les combler.

A mes sœurs : Kahina, Lydia et Sihem.

A toute ma famille proche ou lointaine.

A tous(tes) mes amis(es).

A toutes les personnes que j'aime et qui font partie de ma vie.

Sommaire

L'évolution des techniques informatiques depuis les vingt dernières années a permis d'adapter les outils informatiques à l'organisation des entreprises. Vu le grand volume de données manipulées par ces dernières ,la puissance des micro-ordinateurs, les performances des réseaux et la baisse considérable des coûts du matériel informatique ont permis l'apparition d'une nouvelle approche et ce en répartissant les ressources informatiques tout en préservant leur cohérence.

La solution qui s'impose est de distribuer les données et les organiser dans les bases de données sur différents sites de stockage. L'ensemble de ces sites constitue un système de bases de données réparties offrant la possibilité aux utilisateurs de manipuler les différentes bases via un réseau de manière transparente, comme dans une base de données globale.

Mon projet consiste à développer un système d'information dont les données sont intégrées dans un environnement réparti. L'objectif de ce travail est d'essayer de résoudre les problèmes de fragmentation et localisation des données et d'exécution distribuée des requêtes posées par la répartition d'une base de données.

Pour cela, j'ai conçu et mis en œuvre une base de données répartie sous SQL Server 2008 pour la gestion des ventes de produits en ligne au niveau de l'entreprise NWTRADERS, citée dans les TP de Microsoft.

Chapitre I :

Les bases de données réparties

Introduction

Une base de données est un ensemble d'informations structurées mémorisées sur un support permanent. Elle peut être locale, ou répartie dont certaines données sont distantes, en termes d'espaces de stockage, de machines…, de plus elles doivent pouvoir être accédées par plusieurs locaux ou distants. Afin de permettre aux utilisateurs d'interagir avec la base de données de façon cohérente et performante, la présence des bases de données réparties devient indispensable.

1. Définition :

Une base de données répartie BDR est une collection de bases de données localisées sur différents sites, généralement distants. Elles sont mises en relations les unes avec les autres à travers un réseau d 'ordinateurs, perçues par l'utilisateur comme une base de données unique. Elle permet de rassembler des données plus ou moins hétérogènes, disséminées dans un réseau sous forme d'une base de données globale et intégrée.

2. Avantages et inconvénients de BDR :

2.1 Avantages :

- Reflète une structure organisationnelle : amélioration de l'autonomie locale.
- Disponibilité améliorée : une panne dans un site d'un SGBDR ou une rupture de ligne de communication isolant un ou quelques sites n'immobilise pas l'ensemble du système.

- Economie : l'ajout de station de travail à un réseau est nettement moins couteux que l'extension d'un gros système centralisé.
- Facilité d'accroissement (scalability) : ajouter des machines aux réseaux sans toucher à la cohérence de la base de données.

2.2 Inconvénients :

Sécurité : dans un système centralisé, l'accès aux données est un contrôle facile, tandis que dans un système réparti, non seulement il faut contrôler l'accès aux données dupliquées dans les emplacements multiples, mais le réseau lui-même doit être sécurisé.

Coût : la distribution des données entraine des coûts supplémentaires en termes de communication (trafic réseau), et en gestion des communications comme le hardware et software à installer afin de gérer les communications et distributions.

La distribution est coûteuse en termes de personnel utilisé, car il faut payer les administrateurs de chaque site.

3. Caractéristique d'une base de données répartie :

Au niveau de la BDR, la transparence est un principe fondamental qui apparait dans la localisation, le partitionnement et la duplication des données :

- Transparence de la localisation :

La transparence de la localisation des données sous entend que ni les applications ni les utilisateurs n'ont besoin de connaitre la position réelle des tables auxquelles ils accèdent.

Autrement dit, ils ne doivent pas connaitre la localisation physique des données.

Les utilisateurs accèdent à la BD soit directement par le schéma conceptuel soit indirectement à travers les vues externes, mais en aucun cas, ils n'ont les moyens pour accéder aux schémas locaux.

- Transparence de partitionnement :

Les utilisateurs n'ont pas à connaitre les partitionnements de la base de données, ils ne doivent pas savoir si telle information est fractionnée et ne doivent donc pas se préoccuper de la réunifier.

C'est le système qui gère les partitionnements et les modifie en fonction de ses besoins. C'est donc lui qui doit rechercher toutes les partitions et les intégrer en une seule information logique présentée à l'utilisateur.

- Transparence de la duplication :

Enfin, le principe de transparence de la duplication est que les utilisateurs n'ont pas à savoir si plusieurs copies d'une même information sont disponibles. La conséquence directe est que lors de la modification d'une information, c'est le système qui doit se préoccuper de mettre à jour toutes les copies.

4. Système de gestion de bases de données réparties :

Le SGBDR repose sur un système réparti qui est constitué d'un ensemble de processeurs autonomes appelés sites ou serveurs (micro-ordinateurs, stations de travail) reliés par un réseau de communication qui leur permet d'échanger des données (voir la figure ci-dessous) :

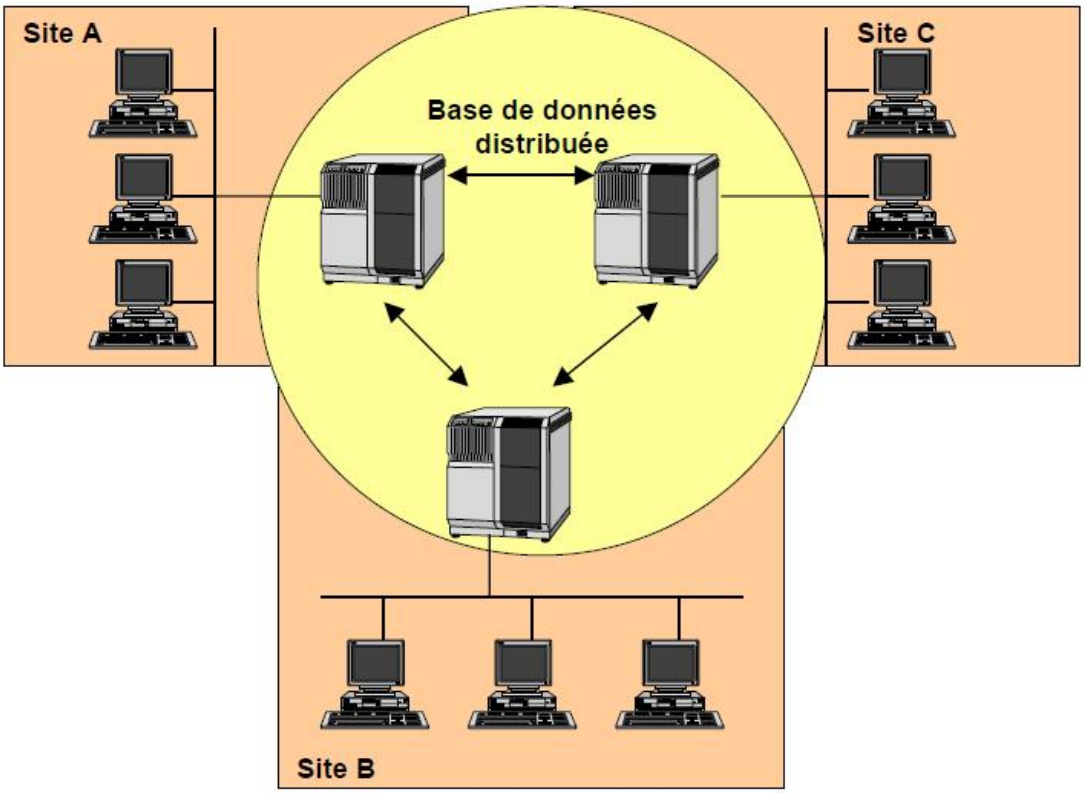


Figure 1 : Schéma de bases de données réparties

Un SGBDR suppose en plus que les données soient stockées sur deux sites au moins. Ceux ci, étant dotés de leur propre SGBD.

4.1 Architecture d'un SGBD réparti :

Cette architecture s'articule autour de trois niveaux de fonctionnalités :

- Niveau local : présent sur chaque serveur, permet d'exporter les données locales. Le niveau local est constitué par un adaptateur local.Ce module réalise le passage du schéma exporté (ce schéma décrit les données exportées par un site vers les sites clients) au schéma local (qui décrit les données d'une base de données locale) et traduit les requêtes en programme d'accès au SGBD local. En sens inverse, il traduit aussi les réponses aux requêtes. Donc, il est véritablement une passerelle depuis le SGBD distribué vers un SGBD local.
- Niveau de communication : requête en provenance d'un site client au serveur de données .Ces sous requêtes référencent le schéma exporté vers ce site client ; En sens inverse, ce niveau transmet les réponses en conformité au schéma exporté.
- Niveau interopérable : Il permet de formuler des requêtes mettant en jeu des vues intégrées de la base ; il assure la décomposition des requêtes en sous requêtes, et le passage de vue intégrées aux différents schémas importés. Une vue intégrée décrit les données de la base de données réparties accédées par une application.

Le schéma importé illustre l'architecture d'un SGBDR avec niveau de schémas :

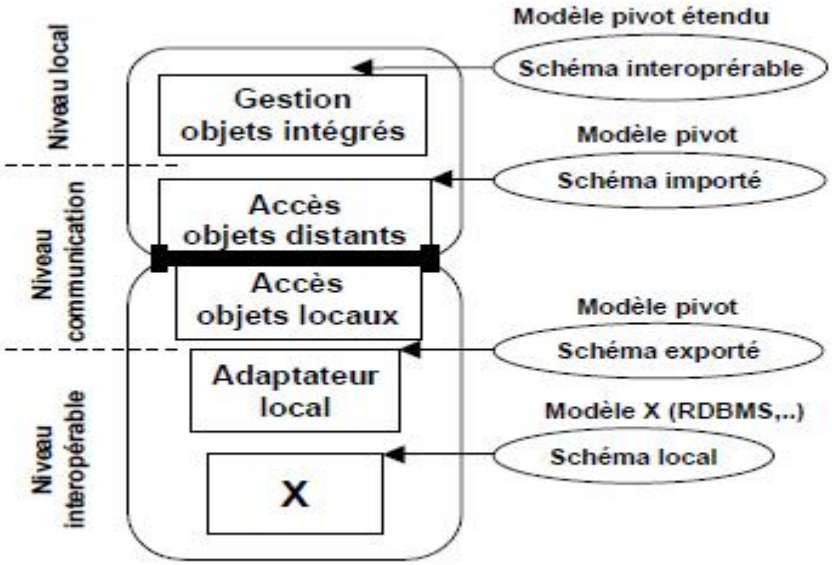


Figure 2 : Architecture d'un SGBDR avec niveau de fonctionnalités

5. Conception d'une base de données répartie :

Dans la phase de conception d'une base de données répartie, l'administrateur doit prendre des décisions dont l'objectif est de minimiser le nombre de transferts entre sites, les temps de transfert, le volume de données transférées.

Deux approches fondamentales sont à l'origine de la conception des bases de données réparties :

5.1 Conception ascendante (bottom up design) :

L'approche se base sur le fait que la répartition est déjà faite, mais il faut réussir à intégrer les différentes bases de données existantes en une seule BD globale.

En d'autre termes, les schémas conceptuels locaux existent et il faut réussir à les unifier dans un schéma conceptuel global et donner aux utilisateurs une vue unique des données implémentées sur plusieurs systèmes à priori hétérogènes (plates-fores et SGBD)

- Etape de la démarche ascendante :

Au premier lieu, il existe plusieurs sites dispersés .Chaque site possède un traducteur. Ce dernier réalise des fonctions telles que la traduction du schéma exporté vers un schéma équivalent en modèle canonique (intermédiaire).

Au deuxième lieu, l'intégration des schémas fait référence au choix de la représentation la plus adéquate pour le schéma global, ainsi que l'identification des éléments de base qui sont liés et l'intégration des schémas intermédiaires.

La figure ci dessous illustre les étapes de la démarche ascendante :

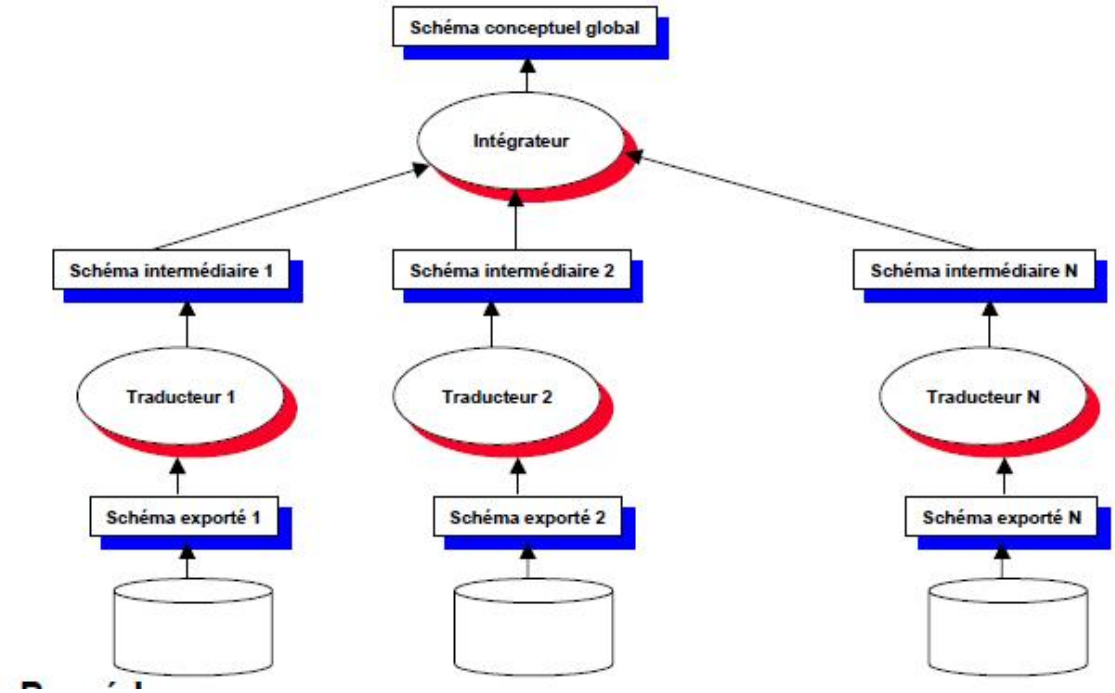


Figure 3 : Etape de la démarche ascendante

5.2 Conception descendante (top down design) :

On commence par définir un schéma conceptuel global de la base de données répartie, puis on le distribue sur les différents sites en des schémas conceptuels locaux. La répartition se fait donc en deux étapes .En première étape, la fragmentation, et en deuxième étape l'allocation de ces fragments aux sites.

5.2.1 La fragmentation :

La fragmentation est le processus de décomposition d'une base de données en un ensemble de sous bases de données. Cette décomposition doit être sans perte d'informations. La fragmentation peut être coûteuse, s'il existe des applications qui possèdent des besoins opposés. On est en quelque sorte dans le cas d'une exclusion mutuelle qui empêche une fragmentation correcte.

Par ailleurs, la vérification des dépendances sur différents sites peut être une opération très longue.

- Objectif de la fragmentation :

Les applications ne travaillent que sur des sous ensembles des relations. Une distribution complète des relations générerait soit beaucoup de trafic, soit une réplication des données avec tous les problèmes que cela occasionne : problème de mise à jour, problème de stockage.

Il est donc préférable de mieux distribuer ces sous ensembles.

L'utilisation de petits fragments permet de faire tourner plus de processus simultanément, ce qui entraine une meilleure utilisation des capacités du réseau d'ordinateurs.

- Type de fragmentation :

a)La fragmentation horizontale :

C'est un découpage d'une table en sous tables par utilisation de prédicats permettant de sélectionner les lignes appartenant à chaque fragment .L'opération de fragmentation est obtenue grâce à la sélection des tuples d'une table selon un ou des critères bien précis.

Exemple :

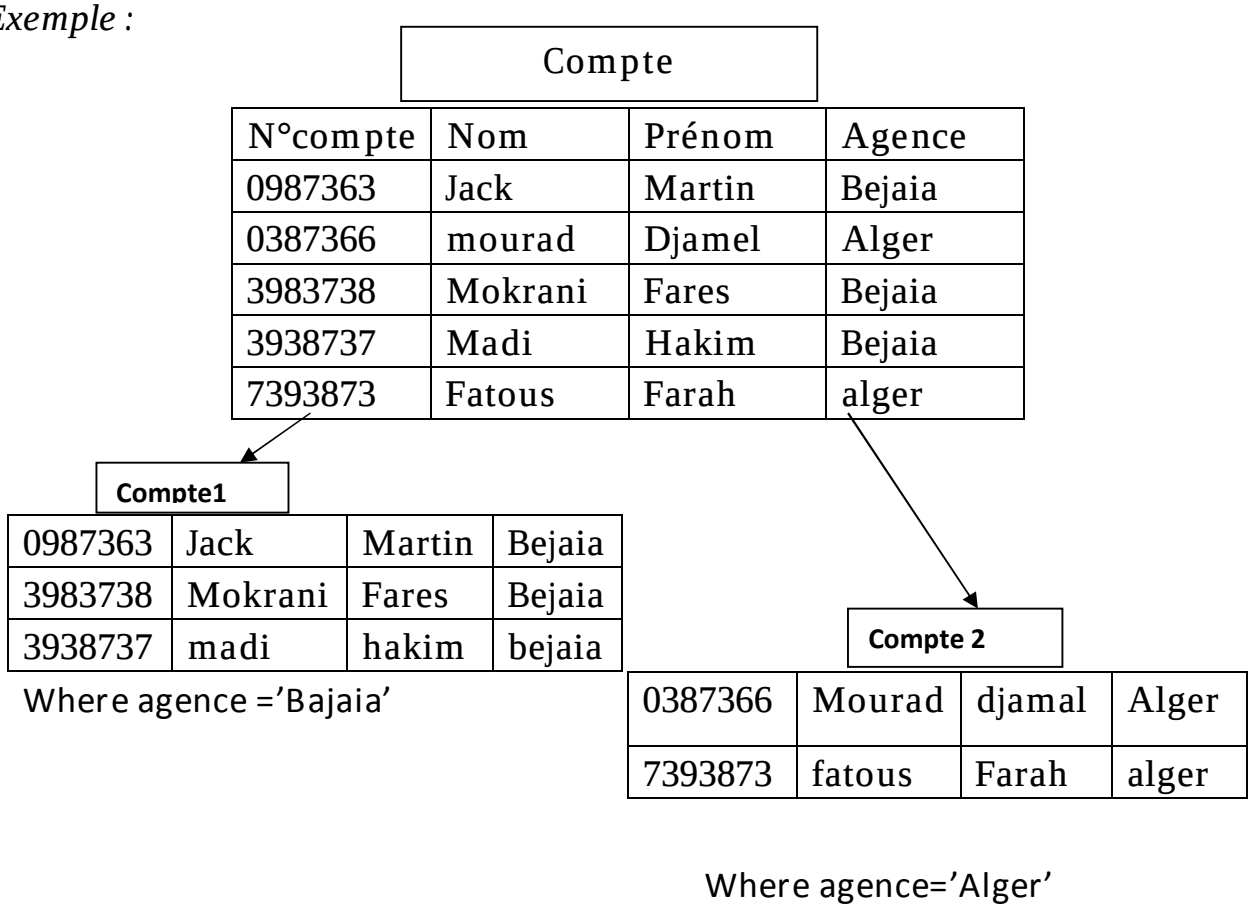


Figure 4: Exemple de fragmentation horizontale

b) La fragmentation verticale :

Elle est le découpage d'une table en sous tables par projection permettant de sélectionner les colonnes composant chaque fragment.

Exemple :

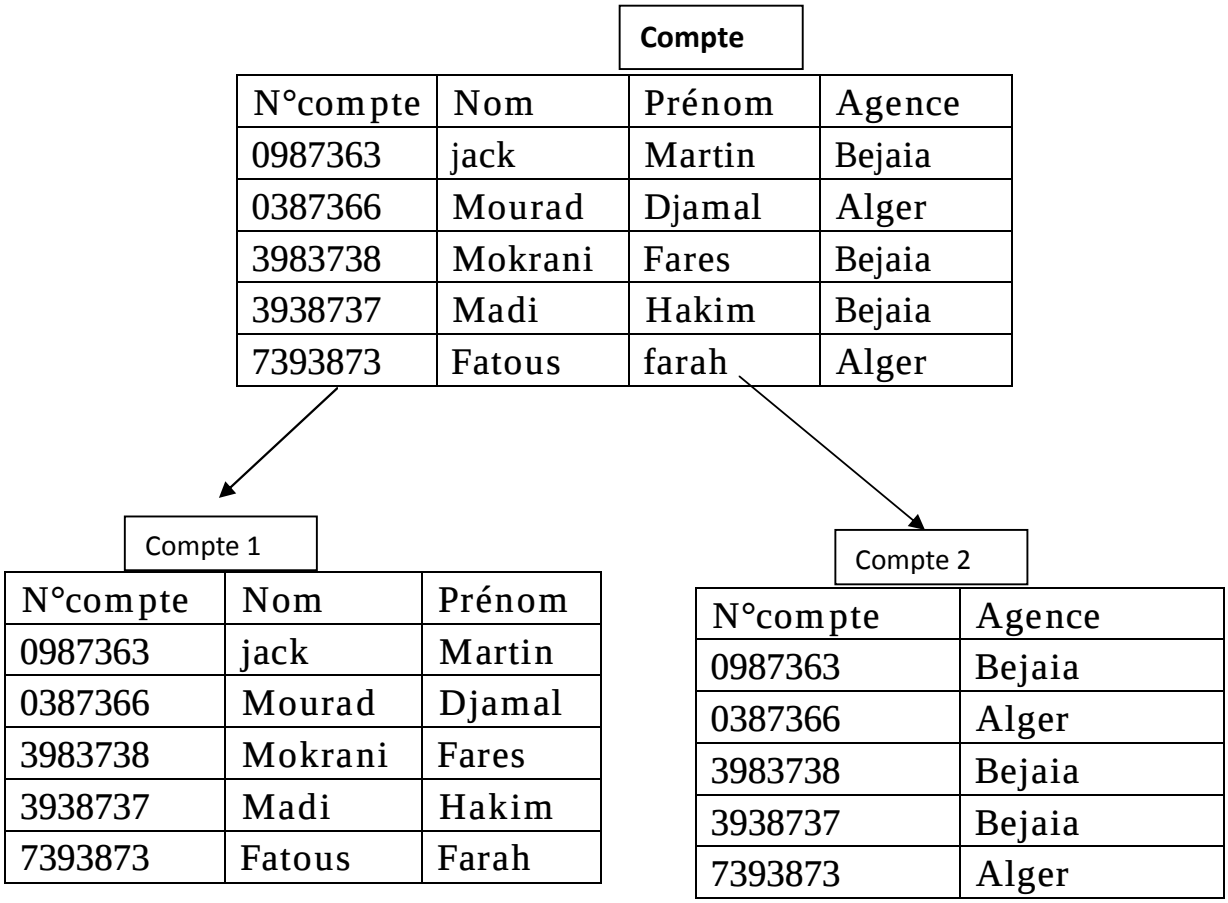


Figure 5 : Exemple de fragmentation verticale

c)La fragmentation mixte :

Elle résulte de l'application successive d'opérations de fragmentation horizontale et verticales sur une relation globale.

- Les règles de fragmentation :

Pour réaliser une fragmentation correcte, il faut respecter les trois règles suivantes :

1. Complétude : pour toute donnée d'une relation globale R, il existe au moins un fragment Ri de la relation R qui possède cette donnée.

2. Reconstitution : pour toute relation R, décomposée en un ensemble de fragments Ri, il existe une opération de reconstruction à définir en fonction de la fragmentation. Pour les fragmentations horizontales, l'opération de reconstruction est une union .Pour les fragmentations verticales, c'est la jointure.

3. Disjonction : une donnée n'est présente que dans un seul fragment, sauf dans le cas de la fragmentation verticale pour la clé primaire qui doit être présente dans l'ensemble des fragments issus d'une relation.

5.2.2 Allocation des fragments :

Suite à la fragmentation des données, il est nécessaire de les placer sur les différentes machines .Un schéma doit être élaboré afin de déterminer la localisation de chaque fragment et sa position dans le schéma global, c'est ce qu'on appelle la location.

- Allocation avec duplication : cette technique consiste à répliquer des parties de la base ,c'est-à-dire les fragments sont dupliqués plusieurs sites selon les besoins (la réplication est détaillée en section suivante).Cette approche est vraiment intéressante car elle améliore considérablement les performances du système, étant donné que les fragments sont dupliqués un peu partout et que les accès aux données sont locaux, évitant ainsi la congestion du réseau et améliorant les temps de réponses. Le principal inconvénient de cette technique est la difficulté des mises à jour de tous les fragments dupliqués.
- Allocation dynamique : avec cette technique, l'allocation d'un fragment peut changer en cours d'utilisation de la BDR, c'est-à-dire qu'un fragment qui se trouve sur un site A à un instant, peut être retrouvé sur un site B à un instant t+1.Cette technique est efficace mais exige le maintien du schéma d'allocation et des schémas locaux.

5.2.3 La Réplication :

La réplication consiste à copier les informations d'une base de données sur une autre .Son objectif principal est de faciliter l'accès aux données en augmentant la disponibilité .Soit par ce que les données sont copiées sur différents sites permettant de repartir les requêtes, soit parcequ' un site peut prendre la relève lorsque le serveur principal s'écroule .Une autre application tout aussi importante est l'amélioration des requêtes sur les données locales,

Et ceci permet d'éviter les transferts de données et d'accroitre la résistance aux pannes.

- Type de réplication :

a)Réplication asymétrique : la réplication asymétrique distingue un site maitre, appelé site « primaire », chargé de centraliser les mises à jour .Il est le seul autorisé à mettre à jour les données et chargé de diffuser les mises à jour aux copies dites secondaires.

Le plus gros problème de la gestion asymétrique est la panne du site primaire. Dans ce cas, il faut choisir un remplaçant, si l'on veut continuer les mises à jour.

On a abouti alors à une technique asymétrique mobile dans laquelle le site primaire change dynamiquement. On distingue l'asymétrie synchrone et l'asymétrie asynchrone.

- **v** Réplication asymétrique synchrone : elle utilise un site primaire qui pousse les mises à jour en temps réel vers un ou plusieurs sites secondaires. La table répliquée est immédiatement mise à jour pour chaque modification par utilisation de trigger sur la table maitre.
- **v** Réplication asymétrique asynchrone : elle pousse les mises à jour en temps différé via une file persistante .Les mises à jour seront exécutées ultérieurement à partir d'un déclencheur externe, l'horloge par exemple.

b) Réplication symétrique : à l'opposé de la réplication précédente, la réplication symétrique ne privilégie aucune copie, c'est-à-dire chaque copie peut être mise à jour à tout instant et assure la diffusion de mises à jour aux autres copies.

Cette technique pose problème de la concurrence d'accès, risquant de faire diverger les copies. Une technologie globale de résolution de conflit doit être mise en œuvre .On distingue l'asymétrie synchrone et la symétrie asynchrone.

- Réplication symétrique synchrone : lors de la réplication symétrique synchrone, il n'y a pas de table maitre .L'utilisation de trigger sur chaque table doit différencier une mise à jour client, à répercuter d'une mise à jour par réplication. Cette technique nécessite l'utilisation de jetons.
- Réplication symétrique asynchrone : dans ce cas, la mise à jour des tables répliquées est différée .Cette technique risque de provoquer des incohérences de données.

6. Décomposition et optimisation des requêtes :

Un traitement réparti fait appel à des données gérées par des SGBDs distincts. Ce traitement contient donc des requêtes qui correspondent à un ensemble d'opérations de recherche et de mise à jour sur des données de la table BDR, formulé à partir d'un schéma externe global. Le SGBDR contrôle et analyse chaque requête puis la décompose en opérations locales afin d'être exécuté par les SGBDs concernés.

L'optimisation est donc indissociable de la requête, car elle entre en jeu à tous les stades du traitement de la requête .Au niveau de la décomposition, l'optimisation permet de simplifier la requête, et cela après avoir éliminé les sous requêtes inutiles ou bien répété plusieurs fois. La figure ci dessous illustre le plan d'exécution répartie d'une requête :

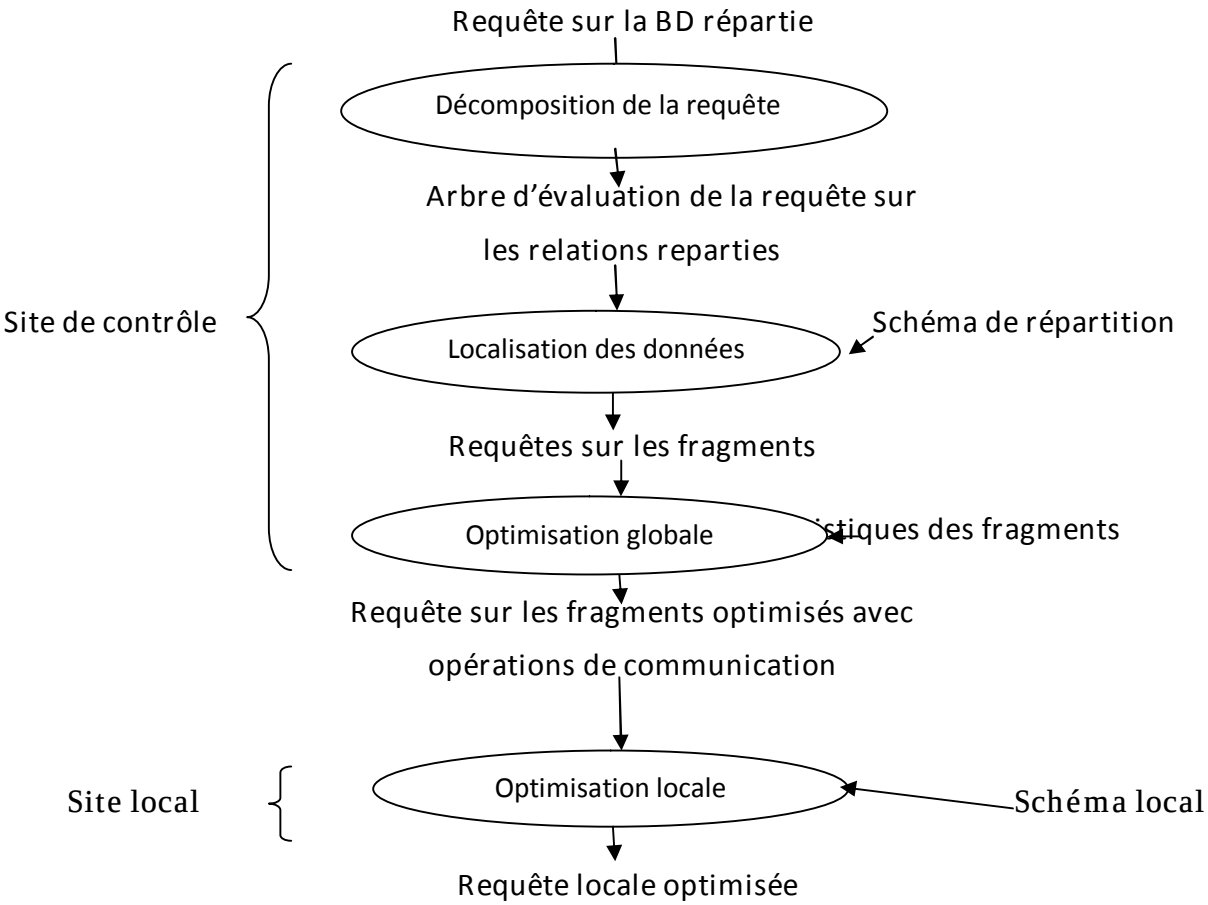


Figure 6 : Décomposition et optimisation d'une requête

- Décomposition de requête :

Cette couche prend une requête exprimée en terme de relation globale et effectue une première optimisation partielle. Le résultat de cette première optimisation est un arbre d'algèbre relationnelle fondé sur les relations globales.

- Localisation des données :

Cette couche prend en ligne de compte la répartition des données sur le système.

Une itération plus poussée de l'optimisation est effectuée en remplaçant les relations globales des feuilles de l'arbre d'algèbre relationnelle par leur algorithme de reconstruction (c'est ce que l'on appelle parfois les programmes de localisation de données) ,c'est-à-dire les opérations d'algèbre relationnelle qui reconstituent les relations globales à partir des fragments constitutifs.

- Optimisation globale :

Cette couche prend en considération des informations statistiques pour trouver un plan d'exécution proche de l'optimal .Le résultat de cette couche est une stratégie d'exécution basée sur les fragments, ou viennent s'ajouter des primitives de communication qui envoient les parties de la requête aux SGBDs locaux pour qu'il les exécute, et qui permettent ensuite de recevoir les résultats.

- Optimisation locale :

Tandis que les deux premières couches s'exécutent sur le site de contrôle (généralement sur le site qui a lancé la requête).Cette couche particulière s'exécute sur chacun des sites locaux impliqués dans la requête.

Chaque SGBD local effectue ses propres optimisations.

Conclusion

Les bases de données réparties sont devenues un domaine important de la gestion d'information. L'objectif général d'une base de données répartie est de permettre à l'utilisateur de manipuler un ensemble de données réparties sur différents sites. L'indépendance de l'environnement répartie est obtenue en assurant la transparence de la localisation, de la fragmentation et de la duplication de données. D'autre objectifs importants sont l'autonomie des sites, la grande disponibilité des données, les performances, l'indépendance et l'hétérogénéité des bases de données locales.

Le chapitre suivant sera consacré à l'étude du SGBD SQL Server 2008.

Chapitre II:

Les bases de données réparties sous SQL Server 2008

Introduction

Dans ce chapitre, je vais présenter en première partie SQL server 2008 dans sa globalité, c'est-à-dire acquérir un aperçu de SQL Server dans son ensemble, à savoir : comprendre la notion de Système de Gestion de Base de données Relationnelle(SGBDR) et le mode de fonctionnement client/serveur, présenter ses composants et les plates formes d'exécution, présenter l'architecture et l'administration Enfin, en deuxième partie, je vais étudier la réplication sous SQL server 2008.

I. Présentation générale de SQL Server 2008 :

1. Définition :

SQL Server est un système de gestion de base de données relationnelles, il est chargé de :

- Stocker les données.
- Vérifier les contraintes d'intégrité définies.
- Garantir la cohérence des données même en cas de panne.
- Assurer les relations entre les données.

Il peut gérer deux types de bases de données :

ü *Les bases OLTP (Online Transactional Processing) :*

C'est des bases de données dans lesquelles les informations sont stockées, de façon directe, afin de réutiliser plus tard ces informations.

ü *Les bases OLAP (Online Analytical Processing) :*

C'est des informations statistiques qui permettent d'extraire les informations sous forme de cube multidimensionnel.

2. Mode de fonctionnement :(Client/serveur)

Toutes les applications qui utilisent SQL Server pour gérer les données, s'appuient sur une architecture (C/S).

- L'application cliente est chargée de :
  - La mise en place de l'interface utilisateur.
  - Elle s'exécute généralement sur plusieurs postes clients simultanément.
- Le serveur, quant à lui, est chargé de :
  - La gestion des données.
  - Repartir les ressources du serveur entre les différentes requêtes des clients.

Les règles de gestion de l'entreprise se répartissent entre le client et le serveur.

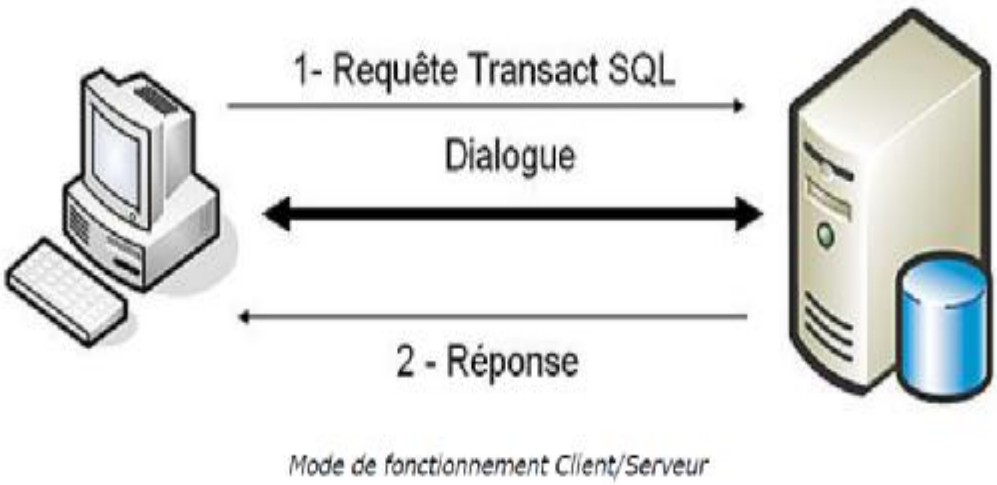


Figure 1 : Mode de fonctionnement (client/serveur)

On peut distinguer trois cas :

- Les règles sont entièrement implémentées sur le client, appelé alors client lourd. Cette solution permet de libérer des ressources au niveau du serveur, mais les problèmes de mise à jour des clients et de développement d'autres applications se posent.
- Les règles sont entièrement définies sur le serveur ; le client est alors un client léger. Cette solution permet d'obtenir des clients qui possèdent peu de ressources matérielles, et autorise une centralisation des règles ce qui rend plus souples les mises à jour. Cependant, de nombreuses ressources sont consommées sur le serveur, et l'interaction avec l'utilisateur risque d'être faible, puisque l'ensemble des contraintes est vérifié, lorsque l'utilisateur soumet sa demande (requête) au serveur.
- Les règles d'entreprises sont définies sur une tierce machine, appelée Middle Ware, afin de soulager les ressources du client et du serveur, tout en conservant la centralisation des règles.

3. Les plateformes possibles :

Il est important de distinguer deux cas :

- ***Les plateformes clientes*** : ce sont les postes sur lesquels les outils d'administration SQL Server peuvent être installés. Il ne s'agit pas des postes qui hébergent une application qui se connecte à une instance SQL Server pour gérer les données. Les outils clients d'administrations peuvent être installés sur tout système d'exploitation Windows 2003, Windows XP Pro ou tout système plus récent.

- Les plateformes serveur : les disponibilités en termes de plateformes sont fonctions de l'édition SQL Server choisie.

Néanmoins, pour héberger une instance de base de données en production, il est nécessaire de disposer d'un serveur performant et fiable. Une plateforme Windows 2003 est donc recommandée.

4. Les composants de SQL Server :

- SQL Server Analysis Service : il permet une analyse poussée des cubes de données définis par l'intermédiaire du Business Intelligence Development Studio.
- SQL Server Integration Service (SSIS) : c'est un outil d'importation et d'exportation de données facile à mettre en place tout en étant fortement paramétrable.
- Reporting Services : il permet de mettre en place des rapports d'analyse des données.
- La réplication des données sur différentes instances permet de positionner les données au plus près des utilisateurs et de réduire les temps de traitement.
- Service Broker permet un travail en mode asynchrone et facilite ainsi la gestion des pics de forte activité en stockant les demandes de travail avant de les traiter.
- L'intégration du CLR dans SQL Server permet de développer procédures et fonctions en utilisant les langagesVB.Net et C#.

- L'intégration du CLR ne vient pas se substituer au Transact SQL, mais se présente comme un complément, afin de pouvoir réaliser un codage simple et performant pour l'ensemble des fonctionnalités qui doivent être présentes sur le serveur.
- Les points de terminaison http permettent à SQL Server d'héberger ses propres services et de faciliter ainsi l'intégration du serveur dans un contexte hétérogène.

5. Architecture de SQL Server :

5.1. Administration :

Le langage naturel de SQL Server est le Transact SQL. Il est donc nécessaire de lui transmettre les instructions dans ce langage. Comme ce langage n'est pas forcément naturel pour l'utilisateur, il est possible de composer l'instruction de façon graphique par SQL Server Management Studio, puis de provoquer son exécution sur le serveur .Les outils graphiques utilisent la bibliothèque SMO (SQL Server Management Object) pour établir un dialogue efficace avec le serveur.SQL SMO englobe et étend SQL DMO.

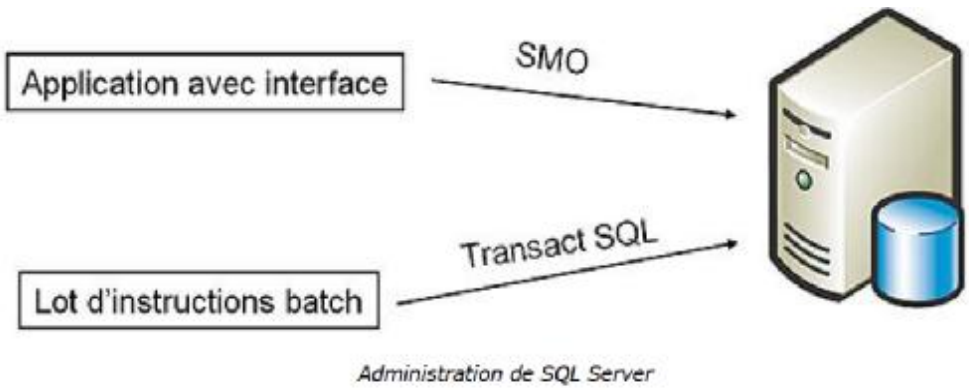


Figure 2 : Administration de SQL Server

5.2. Programmation :

Le développement d'applications clientes, pour visualiser les données contenues dans le serveur, peut s'appuyer sur différentes technologies.

- La DLL SQL Native Client :

C'est une méthode d'accès aux données disponible pour accéder aux données. Avec cette nouvelle API, il est possible d'utiliser l'ensemble des fonctionnalités de SQL Server comme les types personnalisés définis avec les CLR, MARS, XML.

- SQL Native Client :

C'est une API qui permet de tirer profit des fonctionnalités de SQL Server et de posséder un programme qui accède de façon optimum au serveur.
Elle permet aussi l'écriture de programmes clients optimisés mais uniquement capables d'accéder à des données hébergées par un serveur SQL Server.

- *Exemple.*

Ce modèle de programmation correspond à une application client qui souhaite gérer les données. Dans le cas où l'application souhaite être capable de faire des opérations d'administration, il est alors nécessaire d'utiliser la bibliothèque SMO.

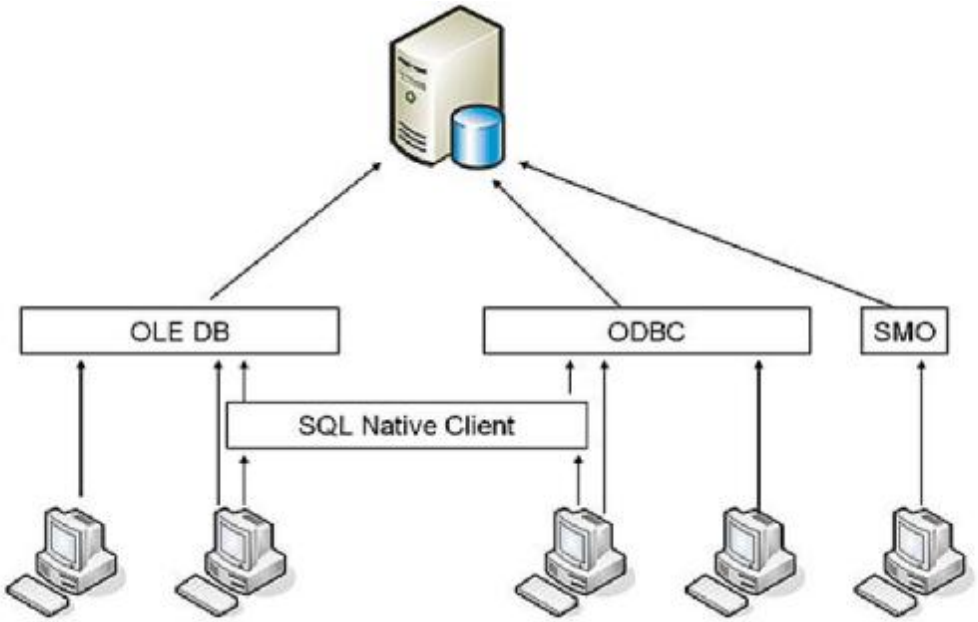


Figure 3 : Exemple d'utilisation de SQL native client

5.3. Base de données SQL Server :

5.3.1 Objets de base de données : il est possible de regrouper ces objets en trois grandes catégories :

- Gestion et stockage des données : tables, type de données, contraintes d'intégrité, valeur par défaut, règles net index.
- Accès aux données : vues et procédures stockées.
- Gestion de l'intégrité complexe : déclencheur (procédure stockée s'exécutant automatiquement lors de l'exécution d'un ordre SQL modifiant le contenu d'une table : INSERT, UPDATE et DELETE).

Le déclencheur est toujours associé à une table et à une instruction SQL. Il permet de mettre en place des règles d'intégrité complexes à cheval sur plusieurs tables ou de maintenir des données non normalisées

5.3.2. Bases de données système:

- Master :

C'est la base de données principale de SQL Server. L'ensemble des données stratégiques pour le bon fonctionnement du serveur y est stocké (comptes de connexion, options de configuration, l'existence des bases de données utilisateurs et les références vers les fichiers qui composent ces bases…).

- Model

Cette base contient l'ensemble des éléments inscrits dans toute nouvelle base utilisateur. Par défaut, il n'y a que les tables système, mais il est possible de rajouter des éléments.

- Tempdb

La base Tempdb est un espace temporaire de stockage partagé. Il permet de gérer les tables temporaires locales ou globales, les tables de travail intermédiaires pour faire des tris par exemple, mais aussi les jeux de résultats des curseurs. La base Tempdb est recréée, avec sa taille initiale, lors de chaque démarrage de l'instance.
Ainsi, aucune information ne peut être conservée de façon persistante à l'intérieur de la base Tempdb. Les objets temporaires sont, quant à eux, supprimés lors de la déconnexion de leur propriétaire.

- Msdb

Elle contient les informations utilisées par le service SQL Server Agent pour déclencher une alerte, prévenir un opérateur ou exécuter une tâche planifiée. Msdb contient également l'historique de l'exécution des tâches.

- Ressources

Cette base en lecture seule contient la définition de tous les nouveaux éléments définis à partir de SQL Server 2005.Les objets systèmes y sont définis bien que logiquement ils apparaissent dans le schéma de l'utilisateur sys.

- Base de données utilisateur :

Elles vont héberger les données fournies par les utilisateurs.

II. Les bases de données réparties sous SQL Server 2008

1. Définition :

La réplication est une puissante fonctionnalité de SQL Server qui permet de distribuer les données et d'exécuter les procédures stockées sur plusieurs serveurs de l'entreprise. La technologie de réplication a considérablement évolué et permet maintenant de copier, déplacer les données à différents endroits et de synchroniser automatiquement les données.

La réplication peut être mise en œuvre entre des bases de données résidant sur le même serveur ou sur des serveurs différents. Les serveurs peuvent être sur un réseau local (LAN), réseau global (WAN) ou sur Internet.

SQL Server distingue deux grandes catégories de réplication :

- ***La réplication de serveur à serveur :***

Elle permet une meilleure intégration ou rapprochement des données entre plusieurs serveurs de base de données

- ***La réplication de serveur à clients :***

Concerne principalement les utilisateurs déconnectés du réseau de l'entreprise et qui souhaitent travailler avec tout ou partie des données de l'entreprise.

2. Les besoins pour la réplication :

SQL Server propose différentes technologies de réplication qu'il est possible d'adapter et de combiner pour répondre le plus exactement possible aux besoins des applications. Chaque technologie présente ses avantages et ses inconvénients. Les trois critères principaux pour sélectionner une technologie de réplication sont :

- La cohérence des données répliquées

1. Cohérence des transactions :

Dans tous les cas, les sites contiennent un jeu de valeurs identique à celui sur lequel toutes les opérations de modification ont été ou auraient pu être effectuées.

- Cohérence transactionnelle immédiate

Avec une telle cohérence, tous les sites, participant à la réplication, ont la garantie de toujours voir les mêmes valeurs au même moment.

Pour assurer la cohérence transactionnelle, SQL Server dispose d'un protocole de validation à deux phases avec tous les sites participants.

Les modifications sont effectuées sur tous les sites ou sur aucun. Une telle solution est bien sûr limitée dans la réalité, car les problèmes du réseau interdisent toute validation de transaction tant que le serveur n'est pas reconnecté au réseau.

Exemple :

Le client effectue une transaction sur le serveur auquel il est connecté. Cette transaction ne sera validée que si elle a pu être exécutée avec succès sur tous les serveurs qui participent à la réplication.

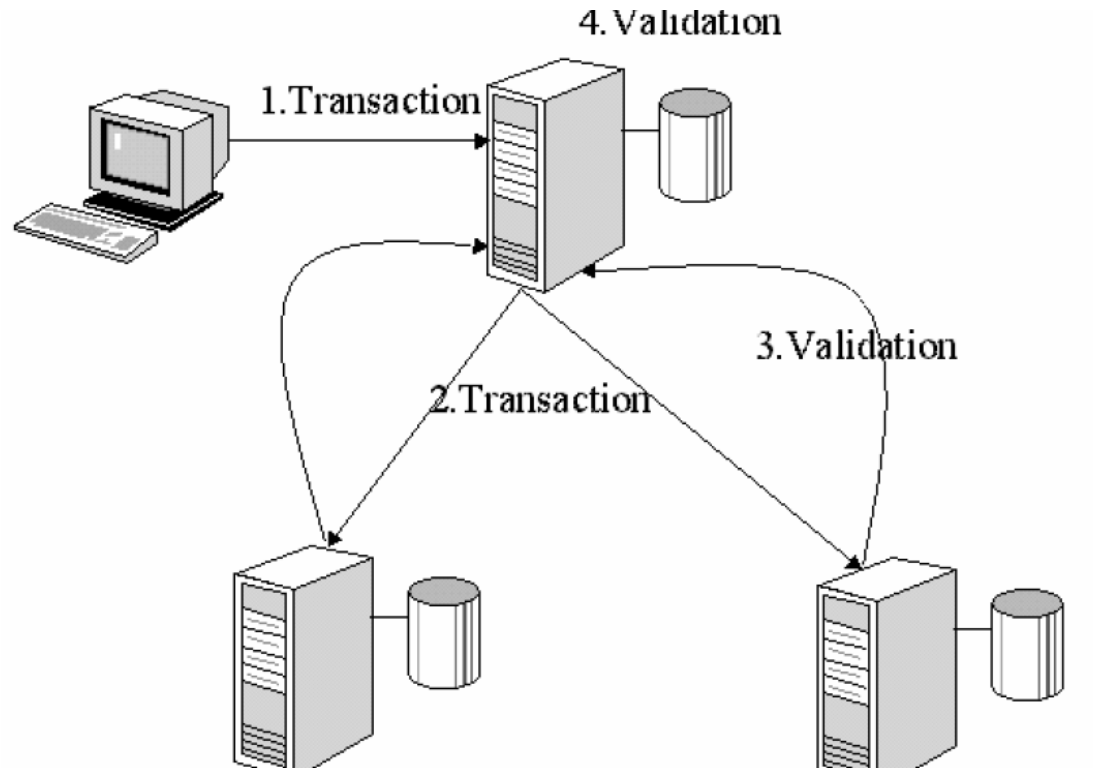


Figure 4 : Cohérence transactionnelle immédiate

ü Cohérence transactionnelle latente :

La cohérence latente des transactions garantit que tous les participants obtiendront les mêmes valeurs que celles contenues sur le site de publication à un instant donné.

Un délai peut s'écouler entre l'instant où la transaction est jouée sur le serveur de publication et l'instant où les modifications sont réfléchies sur les autres sites.

Dans cet exemple, le client envoie une transaction au serveur, elle est immédiatement exécutée localement. Puis sur une base de temps régulière, les serveurs participant à la réplication, rejouent localement l'ensemble des transactions effectuées sur le serveur principal.

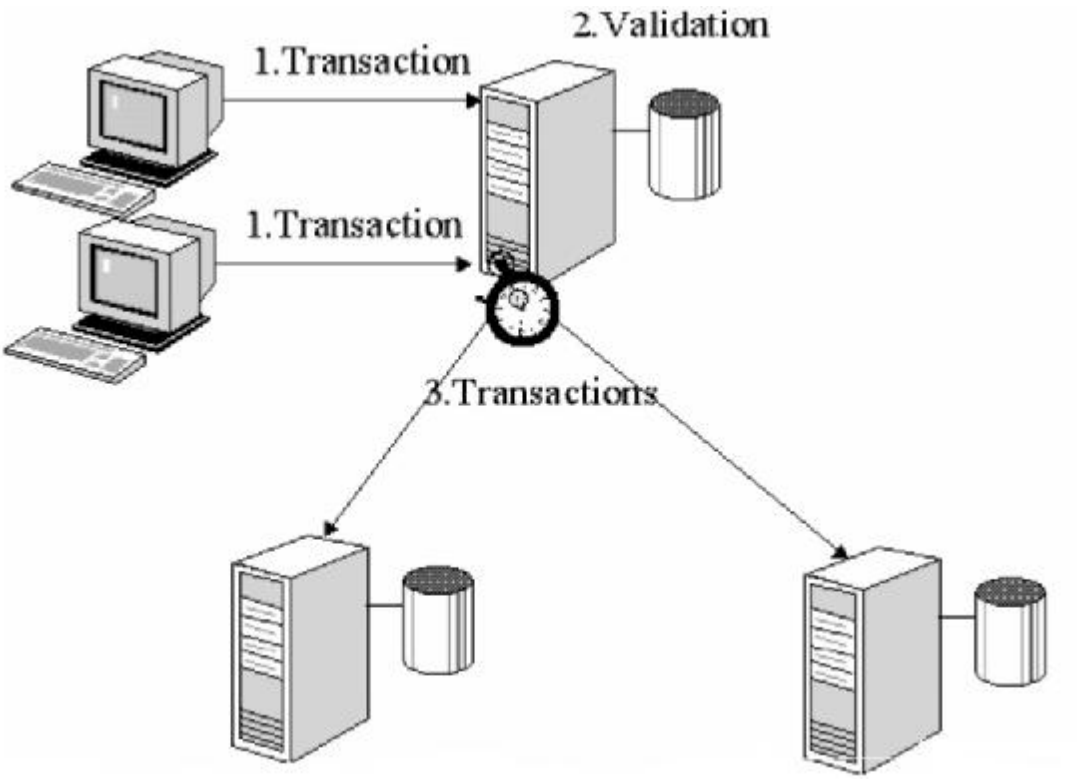


Figure 5 : Cohérence latente

2. Convergence des données

Avec un tel processus, tous les sites finissent par obtenir le même jeu de données, ce qui n'aurait peut être pas pu être obtenu si toutes les modifications avaient été jouées sur un seul serveur. Tous les sites évoluent librement et indépendamment les uns des autres.

La convergence des données est mise en place à l'aide de la réplication de fusion qui tend à amener tous les sites qui y participent vers la gestion du même jeu de données.

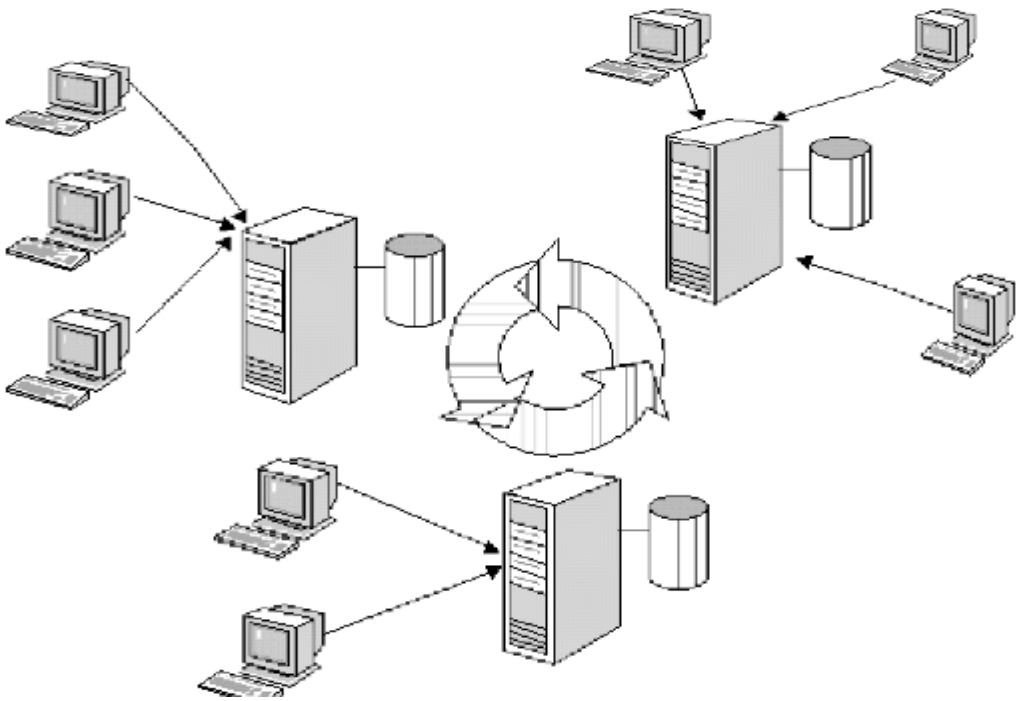


Figure 6: Convergence des données

Ø L'autonomie des sites :

Elle est mesurée par l'impact que possède une opération effectuée sur un site sur les autres sites. L'autonomie est parfaite lorsqu'un site peut évoluer totalement librement. Le site autonome ne se soucie pas des opérations qui peuvent intervenir sur les autres sites. Par exemple, lorsque la réplication garantit la convergence des données, l'autonomie des sites est à son maximum, car chaque serveur SQL peut évoluer librement par rapport aux autres sites.

À l'inverse, la cohérence transactionnelle immédiate impose une autonomie quasiment nulle des sites qui y participent, car une transaction doit être approuvée par tout le monde avant d'être validée. Si un seul serveur ne peut pas, alors la transaction n'est validée nulle part.

Ø Le partitionnement des données pour éviter les conflits :

Il est possible de répartir les données sur plusieurs sites afin que chaque site travaille avec son propre jeu de données, rigoureusement distincts les uns des autres. Ainsi, les transactions intervenant sur chaque site ne mettent en jeu que les données du site et la cohérence globale est conservée. Si par exemple, chaque agence possède un fichier client sur une zone géographique bien déterminée, un client ne peut alors être géré que par une seule agence et toute source de conflit est alors exclue. Le partitionnement des données permet d'éviter tout conflit de données, ce qui est préférable car la résolution des conflits est un processus lourd, qui demande beaucoup de temps machine. Plus les conflits sont nombreux, plus la situation est difficile à gérer.

Le partitionnement des données permet de fonctionner avec une cohérence des données latente car chaque site ne modifie que son propre jeu de données. L'application de cette cohérence est moins lourde à mettre en œuvre que la cohérence transactionnelle immédiate qui repose sur un processus de validation à deux phases.

3. Les modèles de la réplication

Dans SQL Server, les différents modèles de réplication utilisent la métaphore "Editeur Abonné" afin de concevoir au mieux les modèles de réplication.

a. L'Editeur

Comme un éditeur de livres ou de journaux, un serveur Editeur met à la disposition des autres serveurs des données pour mettre en œuvre la réplication. L'éditeur conserve toutes les données publiées (celles qui participent à la réplication) et tient à jour les modifications intervenues sur ces données. Pour les données publiées, l'éditeur est toujours unique.

b. Le Distributeur

Il s'agit du serveur SQL qui contient la base de distribution, c'est à dire celle qui contient toutes les informations utilisées par les abonnés pour tenir à jour les données qu'ils contiennent. Ces deux rôles peuvent être tenus par la même machine.

c. Les Abonnés :

- Abonnement envoyé :

C'est le distributeur qui se charge d'envoyer la mise à jour des données distribuées à tous les abonnés. Ce type d'abonnement est particulièrement bien adapté lorsque le temps de mise à jour des abonnés doit être réduit au minimum.

- Abonnement extrait

C'est l'abonné qui décide de souscrire ou non un abonnement. C'est par la suite l'abonné qui va demander régulièrement des mises à jour. Ce type d'abonnement convient bien lorsque les abonnés sont très nombreux, car la charge de travail serait trop importante pour le serveur Distributeur. Les abonnements extraits sont également bien adaptés aux utilisateurs nomades, car lorsque l'utilisateur vient se reconnecter au réseau de l'entreprise, c'est son propre serveur qui demande la mise à jour de ses données.

d. Les Agents

Les agents de réplication sont :

- L'agent de capture instantanée : son rôle est de fournir une image exacte de la base répliquée à un instant précis. Cette image prend en compte les structures et les données. La totalité de cette capture est stockée dans des fichiers de capture instantanée et des informations de synchronisation sont stockées dans la base de distribution. Cet agent s'exécute sur le distributeur.
- L'agent de lecture du journal : cet agent, actif uniquement dans le cadre de la réplication transactionnelle, scrute le journal de base de données à laquelle il est attaché et copie les transactions qui concernent la réplication vers la base de distribution. Si des réplications transactionnelles sont définies sur plusieurs bases, alors chacune d'entre elle possède son propre agent de lecture du journal. Cet agent s'exécute sur le distributeur.

- L'agent de distribution : exécuté sur le serveur de distribution (abonnement poussé) ou bien sur l'abonné (abonnement extrait), l'agent de distribution a en charge d'appliquer la capture instantanée et les transactions enregistrées dans la base de distribution. Chaque abonné dispose d'une instance spécifique de l'agent de distribution.

4. Types de réplication

Il existe trois types de réplication fournis par SQL Server, le choix s'effectue selon les besoins représentés dans la figure suivante.

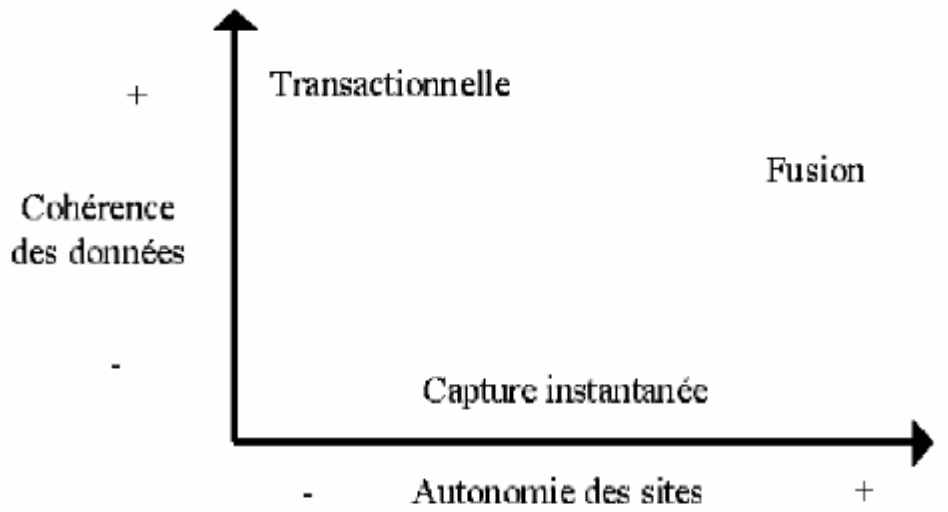


Figure 7 : Type de réplication selon les besoins

ü Capture instantanée :

Cette réplication consiste à prendre une image instantanée des données publiées dans la base. Ce type de réplication demande une surcharge de travail peu importante pour le serveur éditeur car l'opération est ponctuelle.

Les abonnés sont mis à jour en recopiant la totalité des données publiées plutôt que d'effectuer uniquement les modifications (INSERT, UPDATE et DELETE) qui sont intervenues. Cette réplication convient bien pour des publications de petit volume sinon les mises à jour des abonnés peuvent nécessiter des ressources réseau importantes. Cette réplication est également bien adaptée dans le cas de diffusion d'un grand volume d'informations mises à jour de façon ponctuelle et globale.

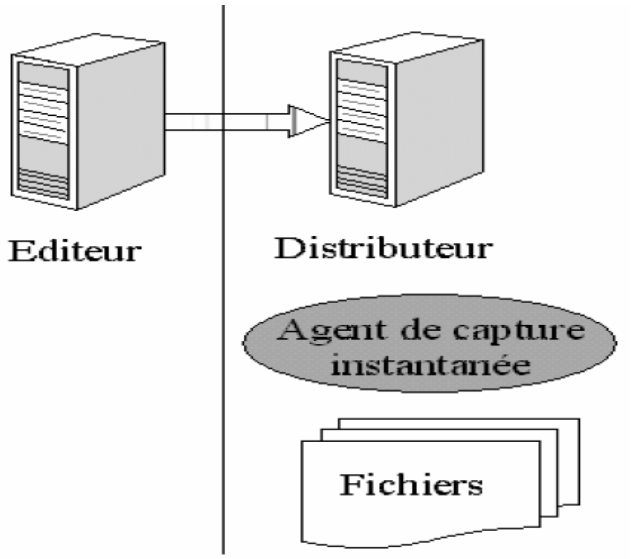


Figure 8: La réplication par capture instantanée

- Transactionnelle : capture instantanée puis mise à jour immédiate des abonnés.

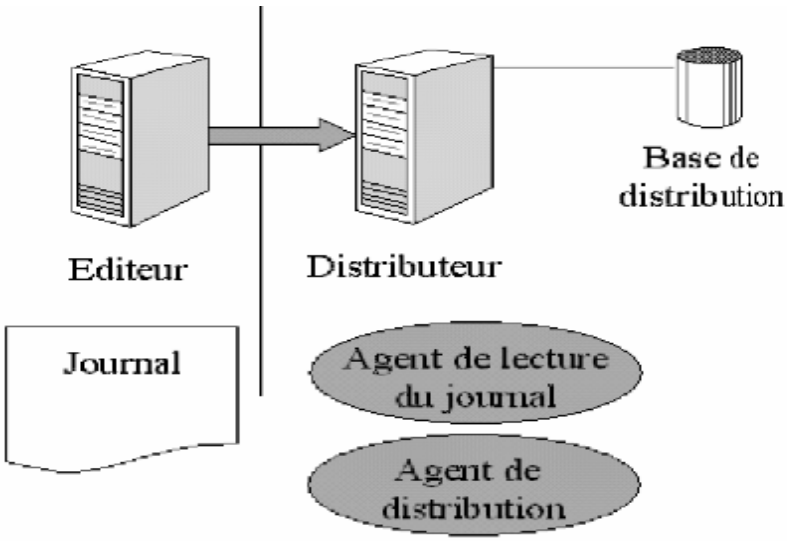


Figure 9 : La réplication transactionnelle

- **Fusion :** Il s'agit ici de surveiller les modifications d'une base de données source et de synchroniser les valeurs entre l'éditeur et les abonnés. Ces derniers peuvent tous effectuer des opérations de mise à jour sur les données distribuées. Si l'éditeur conserve la maîtrise de la publication, ce n'est pas toujours les opérations effectuées sur l'éditeur qui prennent le pas sur celles effectuées sur l'abonné. Toutes les modifications apportées à la base cible sont reportées dans la base source.

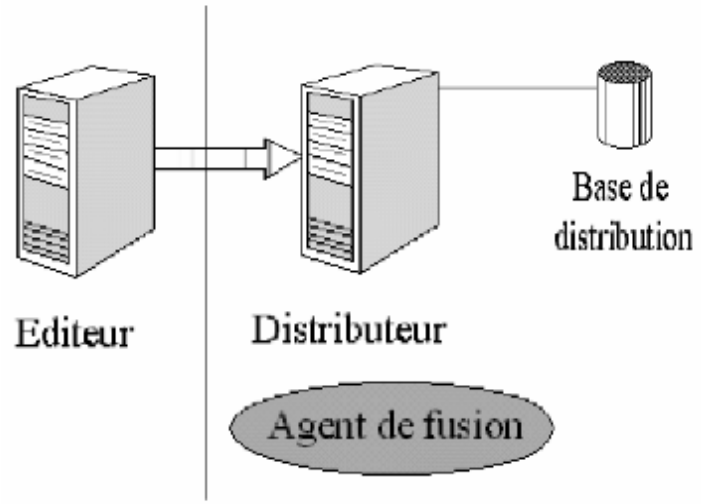


Figure 10 : La réplication par fusion

Conclusion :

Avec SQL Server 2008, le rapprochement entre le développeur, le designer, et l'entreprise est omni présent, tous les outils fournis permettent un développement, un suivi et une intégration plus rapide des applications. Les bases de données deviennent un point central de l'entreprise car elles stockent toutes les informations tant au niveau métier, SQL Server 2008 est donc un produit répondant, par sa fiabilité, sa maturité et sa solidité aux exigences actuelles du marché.

Chapitre III:

Conception

Introduction

Dans ce chapitre, je vais faire la conception d'une base de données répartie pour l'entreprise NWTRADERS, citée dans les TP de Microsoft qui est spécialisée dans la vente de produits en ligne.

1. Infrastructure réseau de NWTRADERS :

L'entreprise est implémentée dans vingt (20) pays, décrite comme ceci :

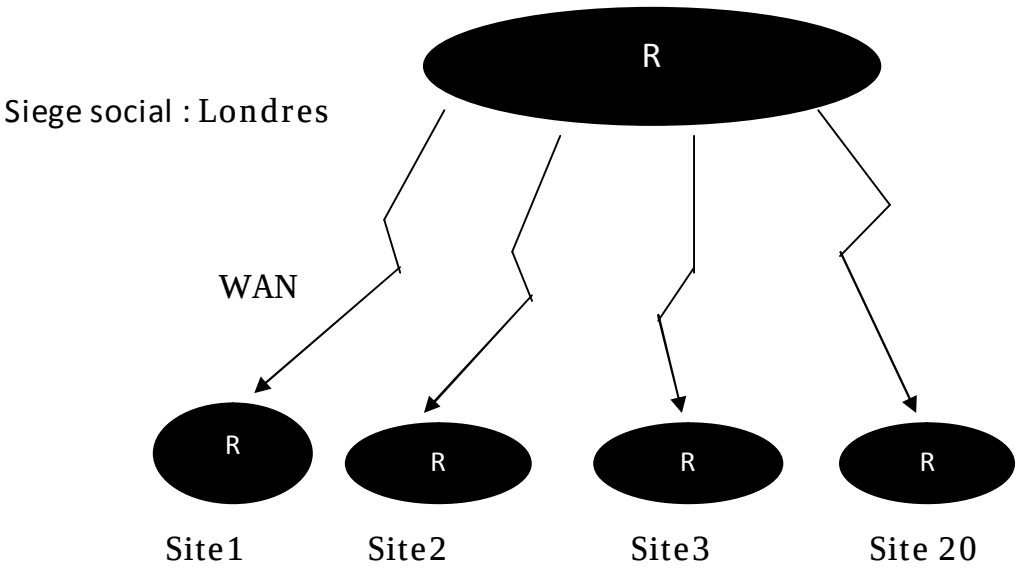


Figure 1 : Infrastructure réseau de NWTRADERS

2. Définition des sites :

On définit trois sites qui sont géographiquement éloignés :

- Site de Londres (Siege social).
- Site de Paris : gère la vente de produits au niveau de paris.
- Site de Tunis : gère la vente de produits au niveau de Tunis.

3. Construction du modèle organisationnel des traitements :

L'objectif de cette étape est de fournir une représentation schématique de l'organisation du processus de vente de produits au niveau de l'entreprise NWTRADERS, c'est à cette étape qu'on doit répondre aux questions OU ? QUI ?et QUAND ?

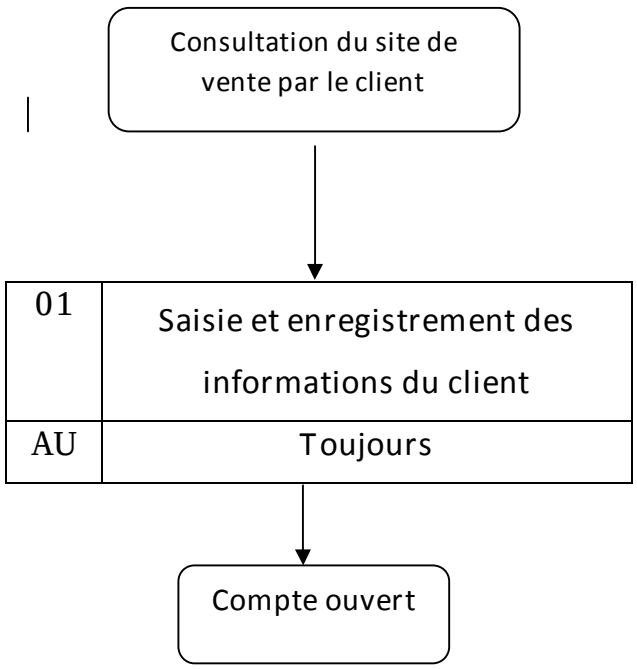
A la première interrogation répondra le site concerné, àla seconde le choix entre automatique ou manuel et à la dernière question précisera le déroulement dans le temps des différentes actions.

- Liste des procédures :

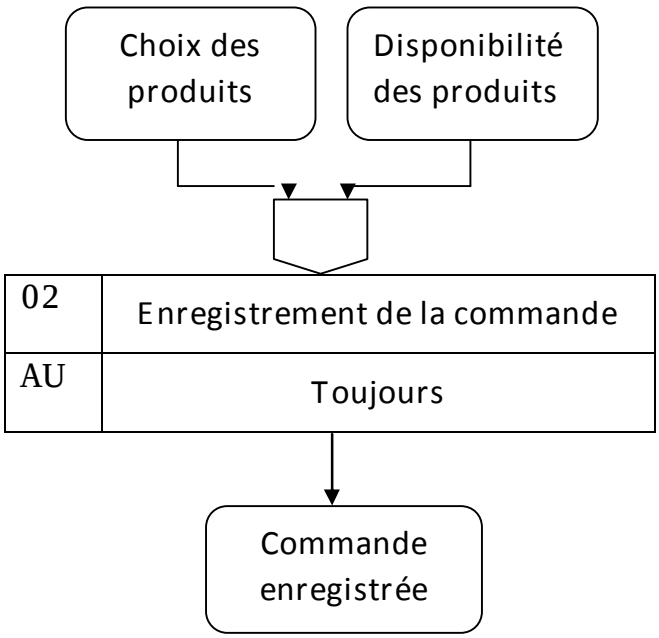
Numéro de la procédure	*Nom de la procédure*
01	Ouverture du compte
02	Commande
03	Facturation
04	Payement
05	Vente(Livraison)
06	Consultation des commandes
07	Consultation des factures
08	Consultation des détails de livraison

Ø Description des procédures :

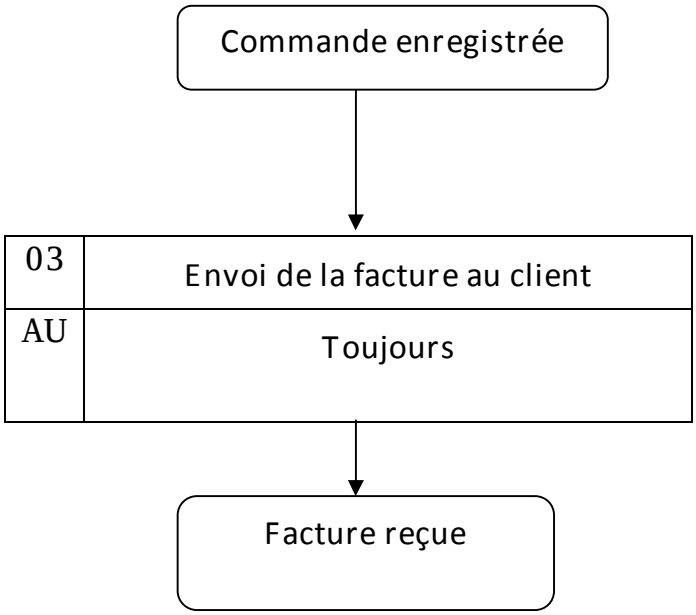
§ *Procédure N°01* : Ouverture du compte.

Période	Enchainement des phases	Poste de travail
Avant chaque commande du client		Site de (Paris, Tunis)

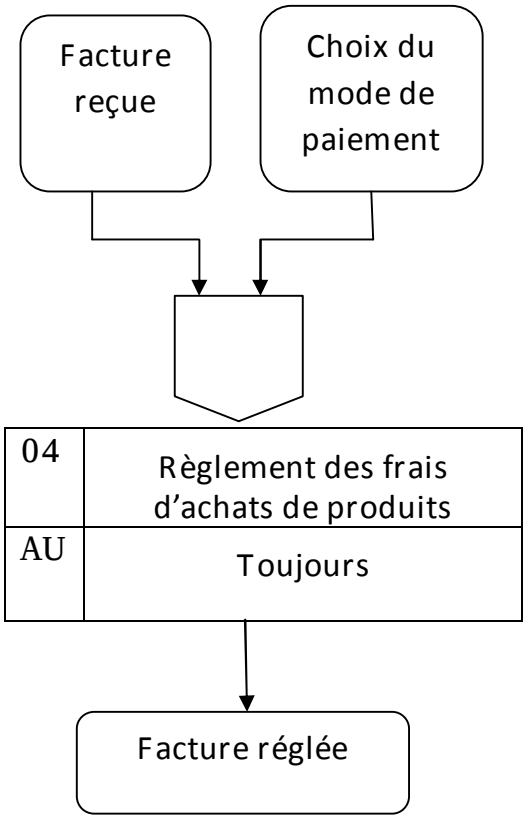
Procédure N°02 : Commande

Période	Enchainement des phases	Poste de travail
A la commande Du client		Site de (Paris, Tunis)

Procédure N°03 : Facturation

Période	Enchainement de phases	Poste de travail
Après chaque commande		Site (Paris, Tunis)

Procédure N°4 : Paiement

Période	Enchainement des phases	Poste de travail
Après chaque réception de la facture		Site (Paris,Tunis)

Procédure N°5 : Vente(Livraison)

Période	Enchainement des phases	Poste de travail
Après chaque règlement de frais	Facture payée 05 : Saisie et enregistrement d'un détail de livraison Au : Toujours Produit vendu	Site de (Paris,Tunis)

Procédure N°6 : Consultation des commandes

Période	Enchainement des phases	Poste de travail
Après chaque enregistrement de la commande dans les deux sites Paris et Tunis	Commande enregistrée 06 : Consultation des commandes et des mises à jour AU : Toujours Copie des commandes enregistrées	Site de Londres

Procédure N°7 : Consultation des factures

Période	Enchainement des phases	Poste de travail
Après chaque enregistrement des factures dans les deux sites Paris et Tunis	Facture enregistrée 07 : Consultation des factures et des mises à jour AU : Toujours Copie des factures enregistrées	Site de Londres

Procédure N°8 : Consultation du détail de livraison

Période	Enchainement des phases	Poste de travail
Après chaque enregistrement d'un détail de livraison au niveau des site (Paris,Tunis)	Produit vendu (livré) 08 : Détail de livraison enregistré AU : Toujours Fin de la procédure de vente	Site de Londres

§ *Site de Paris, Tunis comporte le même schéma local* :

4. Construction des schémas locaux :

Selon le modèle organisationnel de traitement, on a affecté les procédures comme suit :

4.1 Affectation des données :

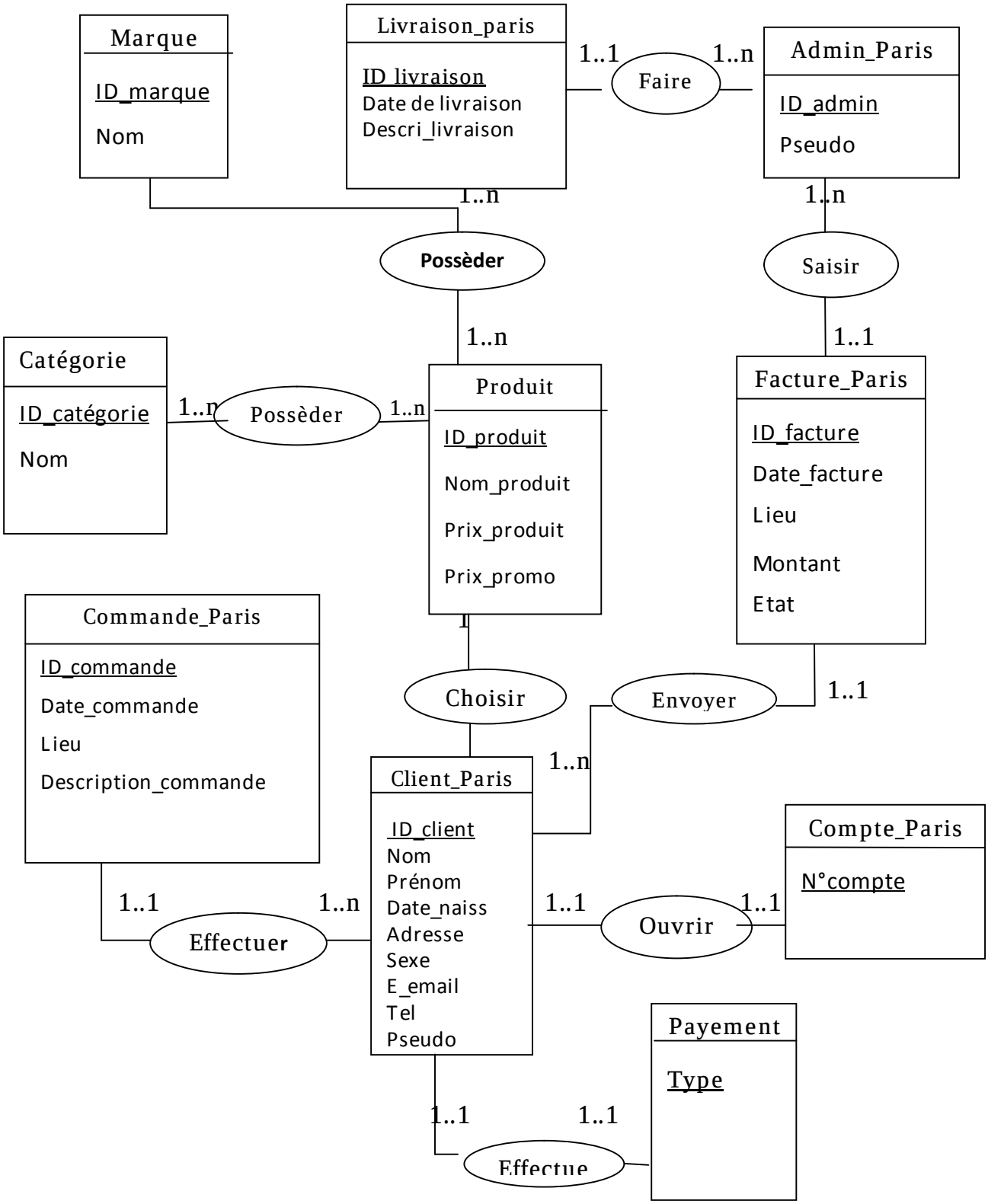
Le site	Nom de la phase	Le bloc de données
Site de paris et Tunis	Saisie du client	• ID_client • Nom • Prénom • Date _naiss • Sexe • Adresse • E-mail • Téléphone • Pseudo
	Saisie de la commande	• ID_commande • Date de commande • Lieu • Description_commande
	Saisie de la facture	• ID_facture • Date de facturation • Lieu • Montant facture • Etat
	Saisie des détails de livraison	• Date de livraison • N°client • Produit livré

Le site	Nom de la phase	Le bloc de données
Londres	Saisie de la liste des produits mis en vente	• ID_produit • Nom_produit • Prix_produit • Prix_promo
	Consultation des commandes	• ID_commande • Date de commande • Lieu • Description_commande
	Consultation des factures	• ID_facture • Date de facturation • Lieu • Montant facture
	Consultation des détails de livraison	• ID_livraison • Date de livraison • Descrip_livraison

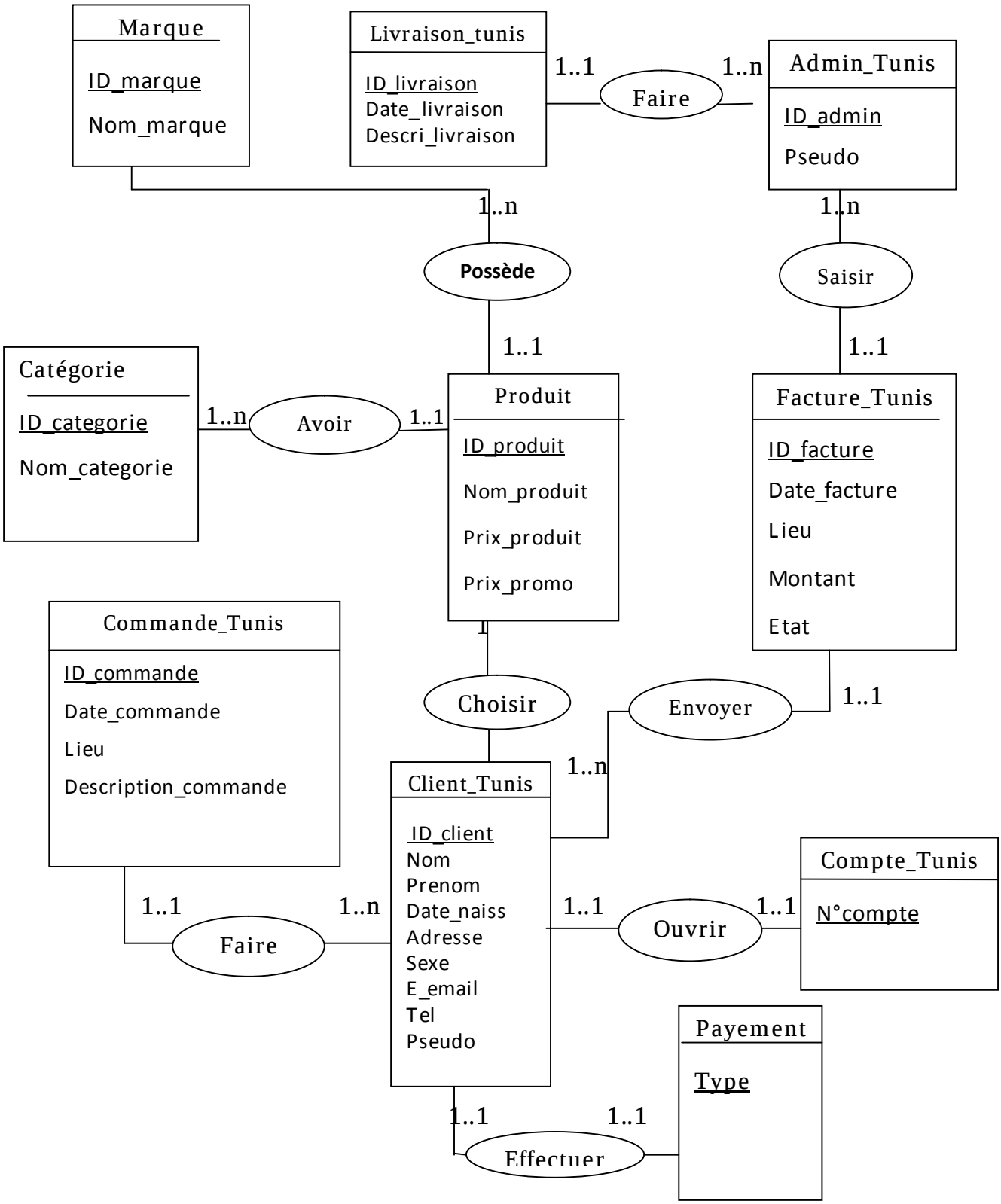
4.2 Construction des schémas locaux :

On construit le schéma local selon le bloc de données utilisées pour chaque site

Ø Schéma local du site de Paris



Ø Schéma local du site de Tunis



5. Construction du modèle logique(MLD) :

Les tables	Les attributs
Produit	ID_produit,nom,prix_produit,prix_promo
Marque	ID_marque, nom_marque,description_marque
Catégorie	ID_categorie, nom_categorie
Commande_Paris	ID_commande,description_commande,lieu.date_commande
Client_Paris	ID_client, nom, prénom, date_naiss, adresse,e_mail,tel,pseudo
Admin_Paris	ID_admin,pseudo.
Paiement	Type
Facture_Paris	ID_facture, date_facture, montant,lieu ,etat
Commande_Tunis	ID_commande,description_commande,lieu.date_commande
Client_Tunis	ID_client, nom, prénom, date_naiss, adresse,e_mail,tel,pseudo
Admin_Tunis	ID_admin,pseudo.
Compte_Paris	N°compte
Compte_Tunis	N°compte
Facture_Paris	ID_facture, date_facture, montant,lieu ,etat
Livraison_paris	ID_livraison,Date_ livraison,Descri_livraison
Livraison_tunis	ID_livraison,Date_ livraison,Descri_livraison

Toutes les tables seront crées au niveau du site de Paris et Tunis, un accès a distance sera fait par le site de Londres pour des éventuelles consultations.

La table client sera fragmentée horizontalement dans les deux sites paris et Tunis.

La figure ci dessous montre la procédure de l'implémentation de ma base de donnés répartie :

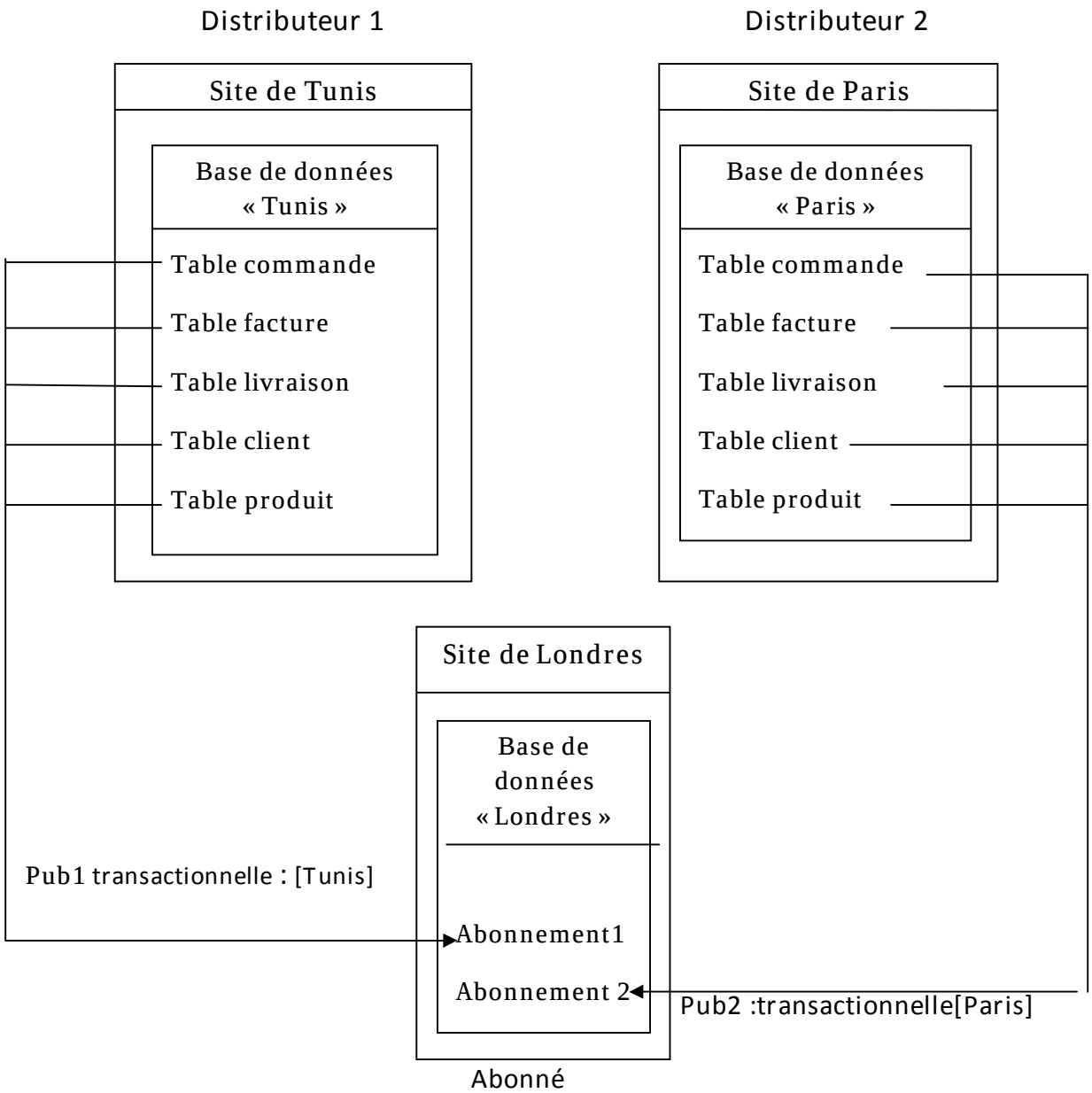


Figure2 : Procédure d'implémentation de la base de données répartie de NWTRADERS

Conclusion :

Apres avoir présenté le processus de conception, à savoir le modèle de traitement ainsi que la création du MLD relationnel et enfin la localisation des données, je vais passer maintenant à la phase de réalisation.

Chapitre IV:

Réalisation

Introduction :

Dans ce chapitre, je vais présenter le support matériel et logiciel qui m'a servi d'appui pour le développement et l'implémentation de mon application .Par la suite, je vais présenter mon application qui couvre la création d'une base de données réparties sur trois sites et ainsi la présentation de quelques interfaces.

1. présentation du matériel utilisé :

La réalisation de mon application est faite sur un micro-ordinateur ; les caractéristiques de ce dernier sont les suivantes :

Ordinateur : est un PC portable décrit comme ceci :

- Un micro-processeur Intel pentium dual-core ;
- Fréquence d'horloge 2.2 GHz ;
- RAM de 4Go ;
- Disque Dur de 250 Go ;
- Lecteur CD/DVD ;

2. Environnement d'exécution :

Durant ma réalisation, j'ai utilisé une plate forme Windows avec son système d'exploitation Windows Server 2003, sur lequel est installé le logiciel suivant :

- Le SGBD SQL Server 2008 pour l'implémentation de ma base de données.

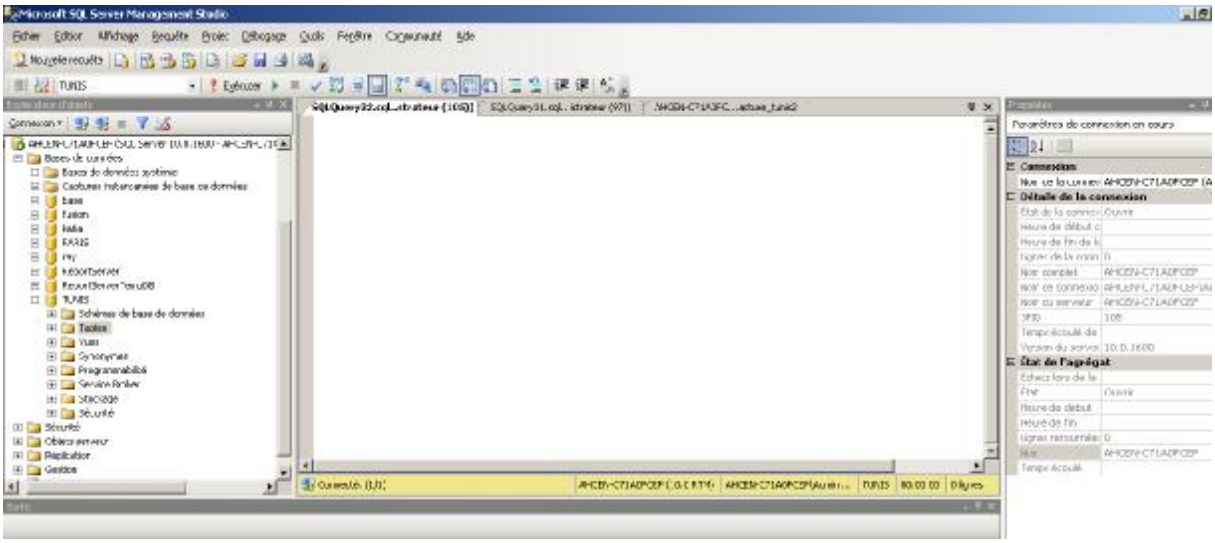


Figure 1 : Interface de la page d'accueil de SQL Server 2008

3. Environnement de programmation :

Pour la création des interfaces de mon application, j'ai utilisé Visual Basic 6.0 qui est un langage de programmation. Outre une interface utilisateur graphique, il dispose de caractéristiques telles que la manipulation déventements, un accès à Win32 API, la programmation orientée objet, la gestion d'erreurs, la programmation structurée. C'est un langage interprété.
L'environnement de développement intégré de Visual Basic permet de créer, exécuter et déboguer des programmes Windows dans une seule application (à savoir Visual Basic).

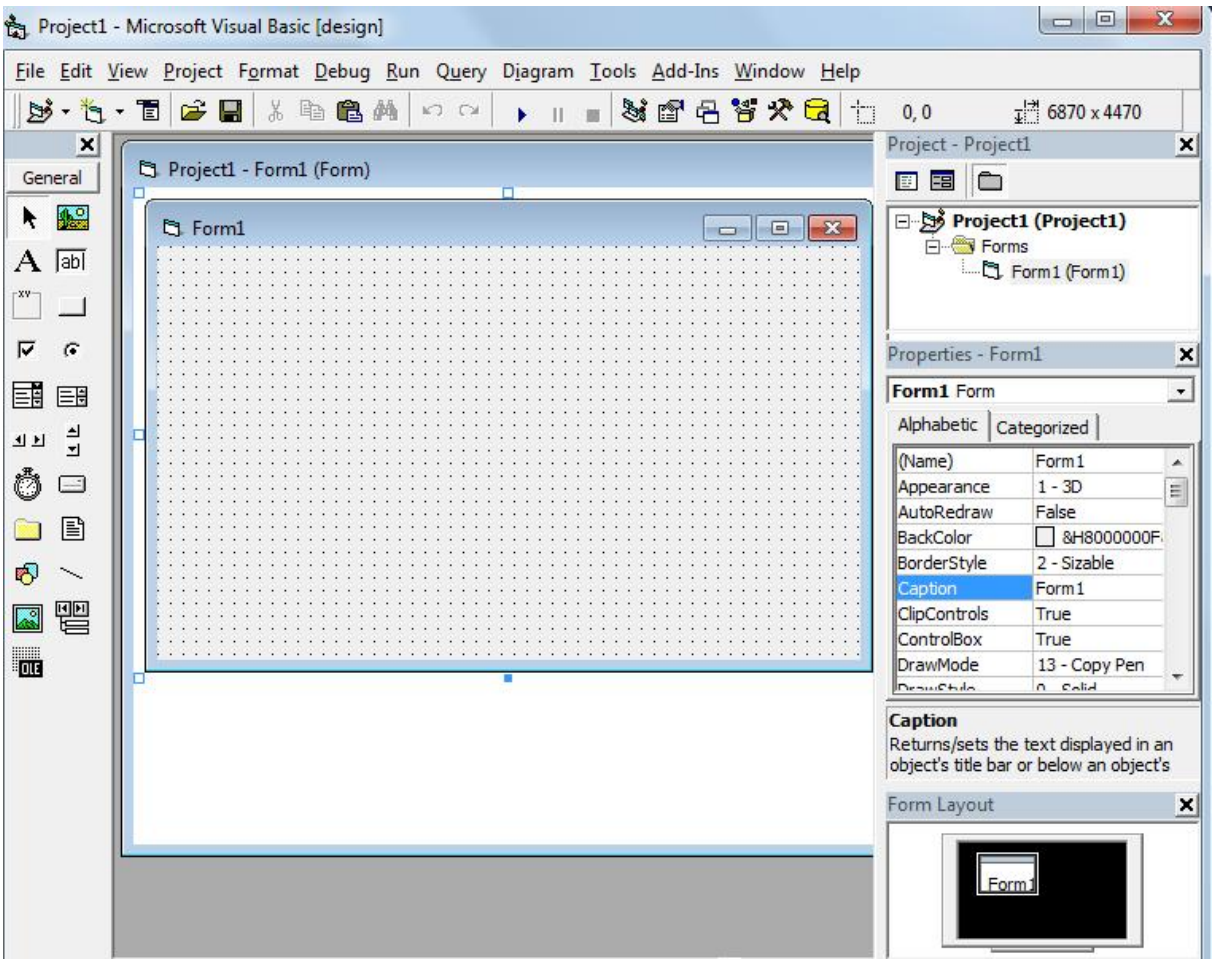
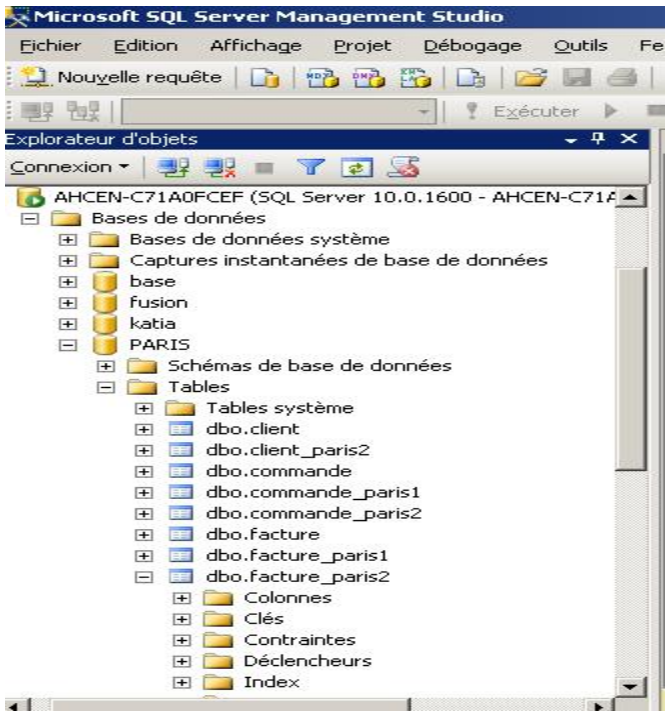


Figure 2 : Interface de la page d'accueil de visual basic 6.0

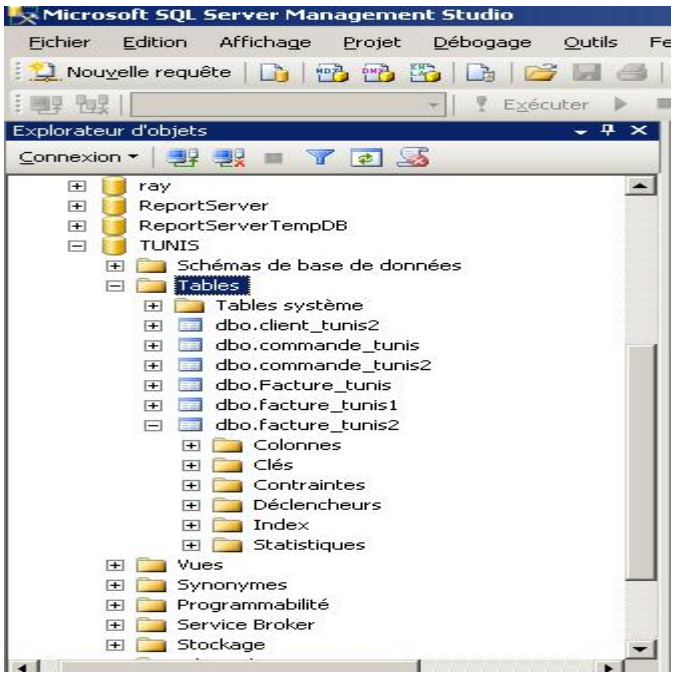
4. Création des objets de la base de données :

Pour la création de la base de données, j'ai utilisé la console de SQL Server 2008 « Microsoft SQL Server Management Studio ».après la création d'un espace base de données, je suis passée à la création des objets (tables, attributs).
Dans mon cas, je créerai trois bases de données sur trois sites différents.

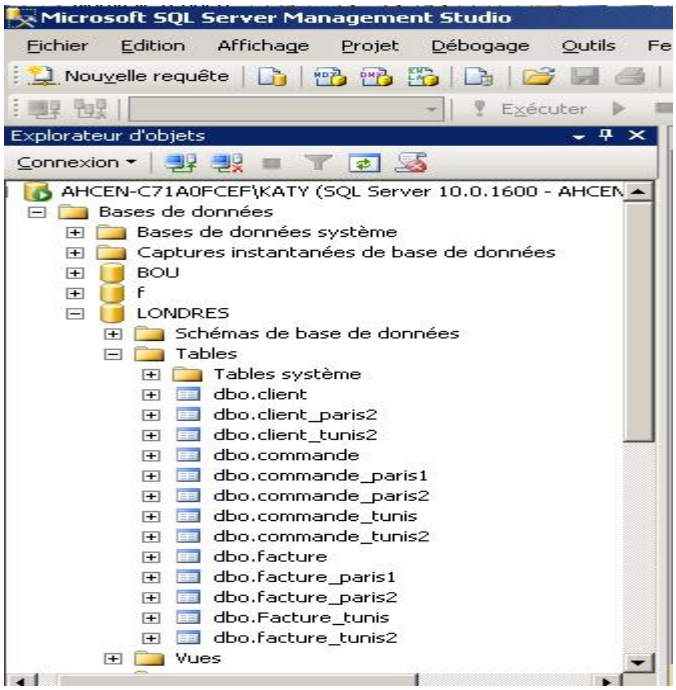
4.1 La base de données du site de Paris



4.2 La base de données du site de Tunis :



4.3 La base de données du site de Londres :

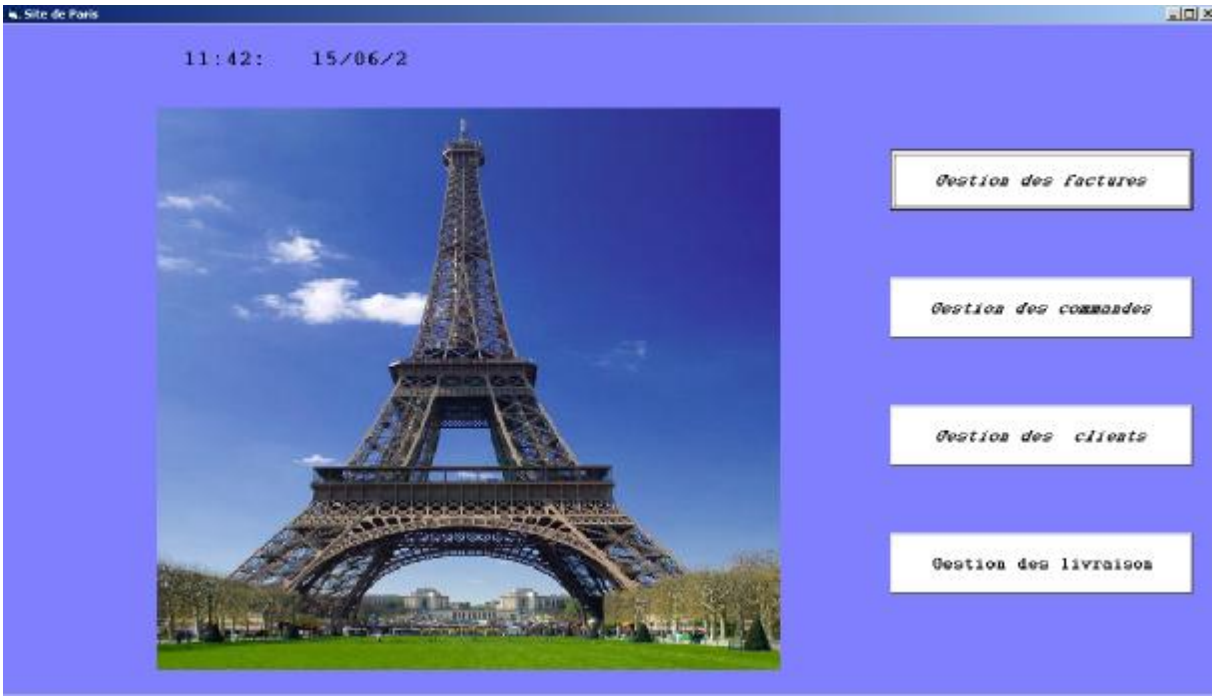


5. Présentation de quelques interfaces :

- Interface d'accueil :

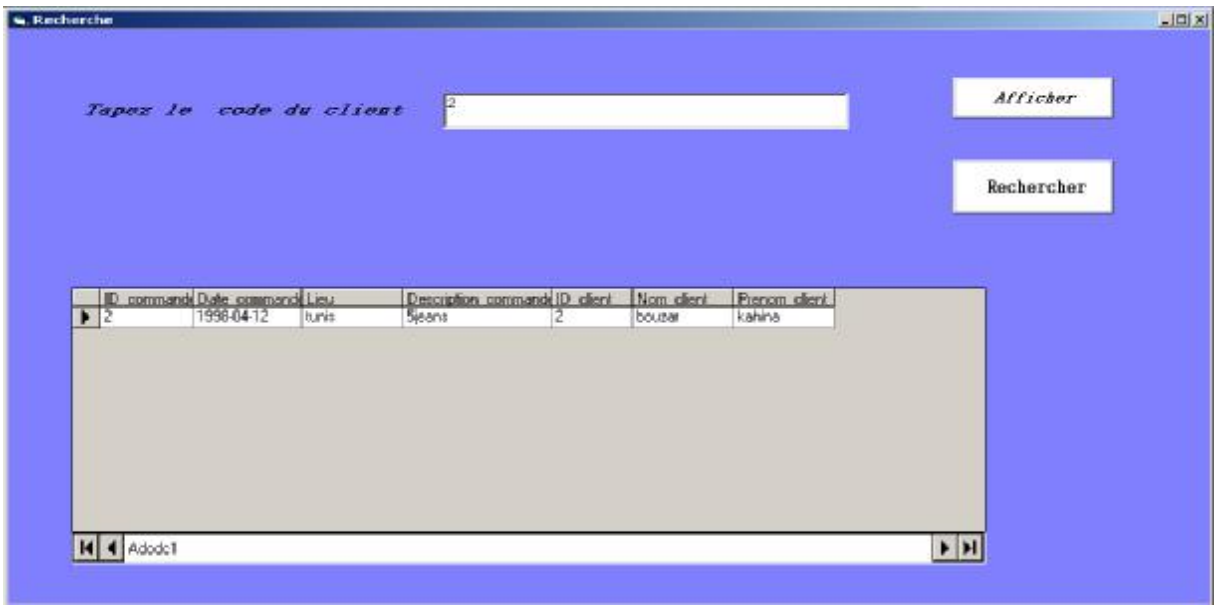
Ø Interface du site de Tunis :

Ø Interface du site de Paris :

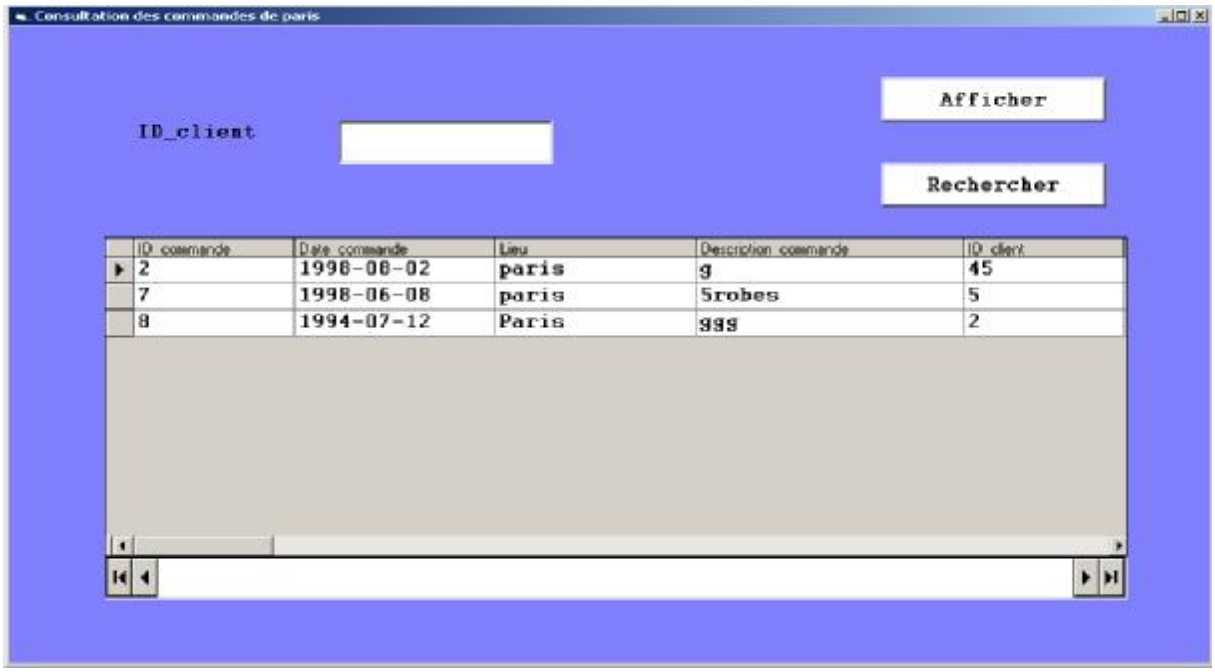


Ø Interface du site de Londres :

Ø Interface de recherche de commande :



Ø Interface de consultation des commandes :



Conclusion générale

Le travail réalisé durant ce projet de fin d'études qui consistait à la conception et à la réalisation d'une base de données répartie sous SQL Server 2008 pour l'entreprise NWTRADERS , m'a permis d'acquérir de nouvelles connaissances sur le SGBD SQL Server 2008 et sur le langage de programmation Visual Basic (VB) version 6.0 .

Bien que l'implémentation de ma base de données répartie soit limitée à trois sites : Paris, Tunis et Londres. Il serait intéressant de prévoir l'accessibilité de ce système aux autres sites restants dissimulés à travers les cinq continents.

Enfin, Je souhaite que cette modeste contribution puisse répondre favorablement aux besoins des futurs utilisateurs et servir comme un outil d'aide et de documentation pour les futurs étudiants.

Bibliographie et webographie

- Mémoire ingénieur, Université Mouloud Mammeri de Tizi Ouzou, conception et réalisation d'une base de données reparties sous SQL server 2000 réalisé par Mr Hammache ET Mr Kerbiche promotion 2003/2004.
- Mémoire ingenieur, Université Mouloud Mammeri de Tizi Ouzou, conception et réalisation d'une base de données reparties sous oracle pour le service personnel enseignant de la faculté génie électrique et informatique.
- Mémoire master II, Université Mouloud Mammeri de Tizi Ouzou, conception et réalisation d'une base de données repartie sous oracle, cas d'étude « ENIEM ».Réalisé par Melle BOUAZIZ Malika et Melle BELGAID Naima promotion 2011/2012.
- Ouvrage : SQL Server 2008 ADMINISTRATION auteur : Jérôme GABILLAUD

Site web :

www.developpez.com

www.siteduzero.com

Annexes

Mise en œuvre

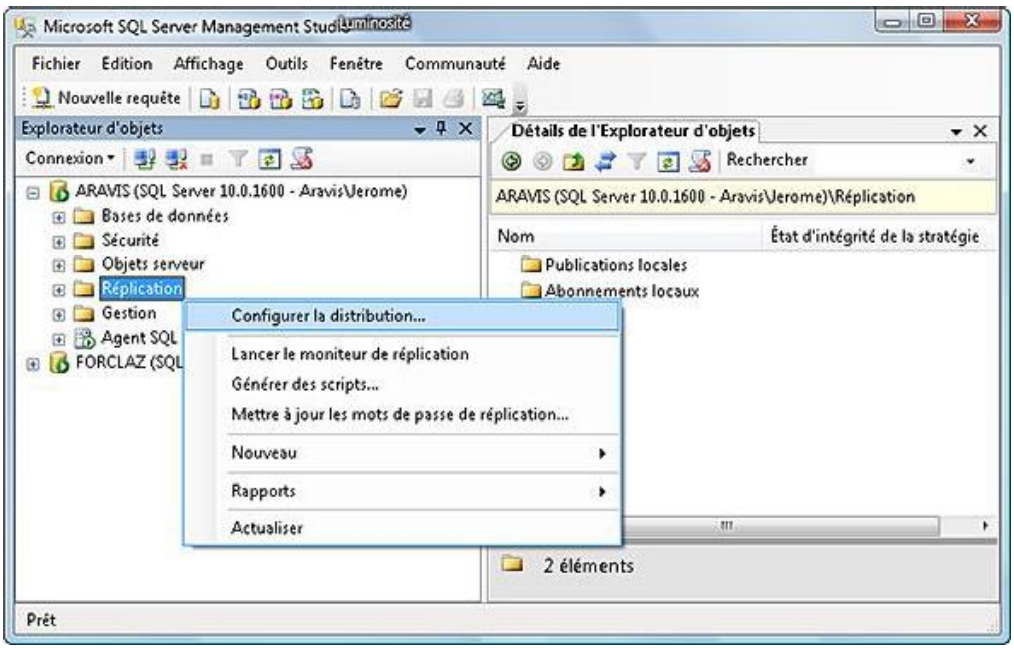
Quels que soient le modèle et le type de réplication choisis, la mise en œuvre doit toujours respecter les même étapes :

- la mise en œuvre du distributeur,
- la mise en œuvre de l'éditeur,
- la mise en œuvre des abonnés,
- la définition des publications.

Comme SQL Server considère par défaut que l'éditeur et le distributeur résident sur le même serveur, l'installation de ces deux états est confondue.

Pour pouvoir mettre en œuvre la réplication à partir de SQL Server Management Studio, tous les serveurs SQL qui y participent doivent y être inscrits.

SQL Server Management Studio propose différents assistants graphiques pour mettre en place, surveiller et paramétrer l'environnement de réplication. Tous ces éléments sont accessibles depuis le nœud **Réplication** de l'explorateur d'objets de SQL Server Management Studio.



1. Le distributeur

Il s'agit du poste qui va gérer le répertoire partagé pour la capture instantanée ainsi que la base de distribution. Le distributeur stocke les modifications effectuées sur les données puis les transmet aux abonnés.

a. Les concepts

Le distributeur doit être installé avant de mettre en place les éditeurs qui l'utilisent. Pour créer un distributeur, il faut être administrateur du système (membre du groupe local Administrateurs et par le biais des connexions approuvées se connecter à SQL Server en tant qu'administrateur). Lorsque le distributeur est installé, il est possible de connaître ses propriétés locales et distantes.

La base distribution

Elle contient toutes les transactions qui sont en attente d'envoi vers les abonnés. Dans le cadre d'une réplication transactionnelle elle est alimentée par l'agent de surveillance du journal qui y transfère toutes les transactions qui interviennent sur des données répliquées. Cette base est automatiquement créée lors de la configuration du distributeur, mais il est possible de :

- spécifier un serveur de distribution distant lors de l'installation de l'éditeur,
- définir plusieurs bases de distribution pour un même éditeur.

Accès au répertoire partagé

Le répertoire de distribution, utilisé par la réplication de capture instantanée, doit être disponible pour tous les agents de distribution qui sont susceptibles de l'utiliser. Cette utilisation est fonction du type de réplication mis en œuvre et ne se pose que lors de l'utilisation d'un serveur de distribution distant.

La mémoire

La mémoire vive présente sur le distributeur doit être en quantité suffisante afin de répondre promptement aux différentes demandes des abonnés. La quantité de mémoire nécessaire est fonction du volume des données répliquées et du nombre d'abonnés.

La désinstallation

Il est possible de désinstaller un distributeur par le biais de l'assistant **Réplication, Désactiver l'assistant Publication** et **Distribution**. Les effets sont les suivants :

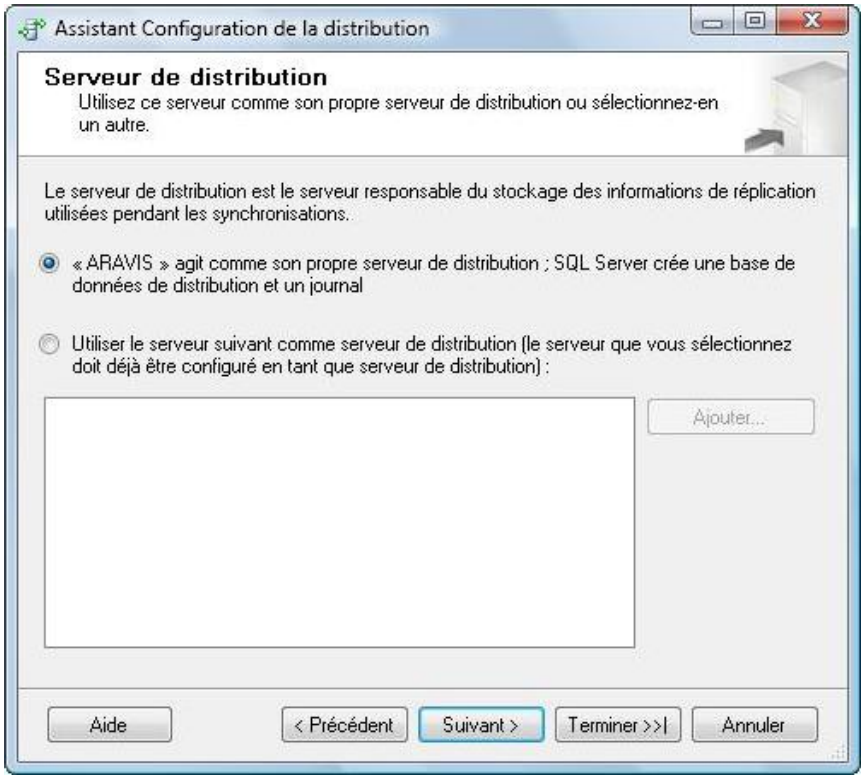
- les bases de données distribution du serveur sont supprimées,
- tous les éditeurs qui utilisent ce distributeur sont désactivés et toutes les publications sont supprimées,
- tous les abonnements sont supprimés, mais les données d'abonnement restent sur les abonnés.

b. La mise en place

La mise en place d'un distributeur demande des privilèges d'administrateur, c'est-à-dire être membre du rôle de serveur **sysadmin.** La création du distributeur se fera soit par le biais des procédures stockées, soit par le biais de SQL Server Management Studio.

La configuration du distributeur est accessible par deux chemins différents dans SQL Server Management Studio.

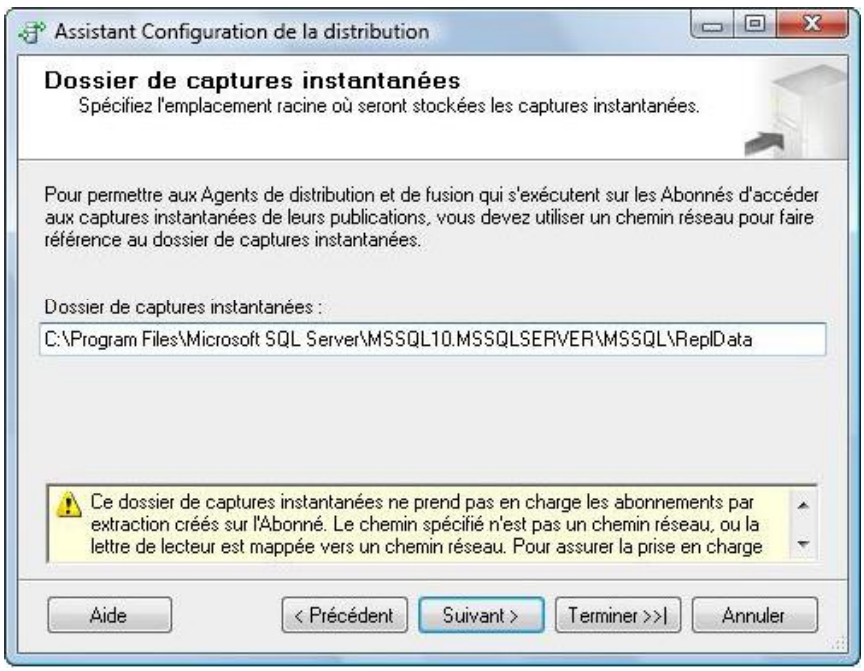
- Soit en sélectionnant explicitement le choix **Configurer la distribution** dans le menu contextuel associé au nœud **Réplication** depuis l'explorateur d'objets.
- Soit en demandant la création d'une nouvelle publication (choix **Nouveau - Publication**) depuis le menu contextuel associé au nœud **Réplication.** L'assistant de création d'une publication est alors exécuté et cet assistant permet de sélectionner/configurer un distributeur pour la réplication.



L'assistant de publication se propose de créer un distributeur

Dans le cas où la demande porte sur la configuration de la distribution, l'assistant de configuration de la distribution est exécuté. Cet assistant va tout d'abord demander de préciser le serveur qui va jouer le rôle de distributeur. Il est alors possible de sélectionner le serveur qui hébergera également les publications (éditeur/distributeur) ou bien de sélectionner un distributeur distant.

À l'étape suivante, l'assistant permet de spécifier le dossier utilisé pour les fichiers de capture instantanée. Un dossier local est proposé par défaut. Si la réplication de capture instantanée doit être accessible depuis le réseau, il est nécessaire de préciser un nom de chemin réseau. L'avertissement situé en bas de la fenêtre porte précisément sur ce point.

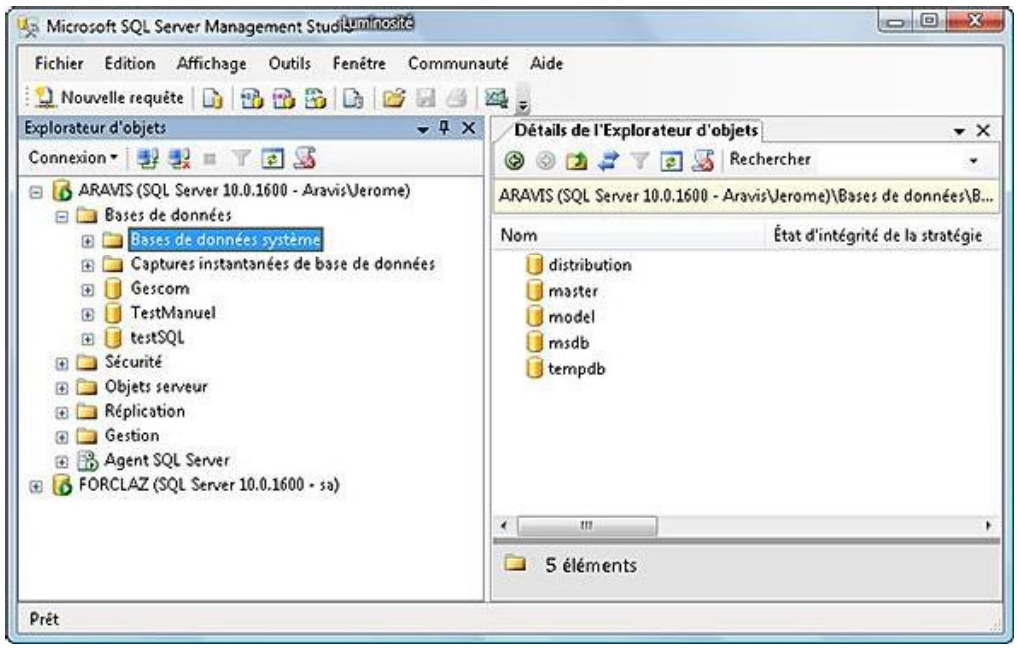


Puis l'assistant permet de préciser le nom de la base de distribution ainsi que l'emplacement physique du fichier de données et du journal. Comme le montre l'écran ci-dessous, la base se nomme par défaut distribution et les dossiers proposés sont ceux spécifiés par défaut.

Ensuite, l'assistant demande de préciser quels sont les éditeurs autorisés à travailler avec ce distributeur.

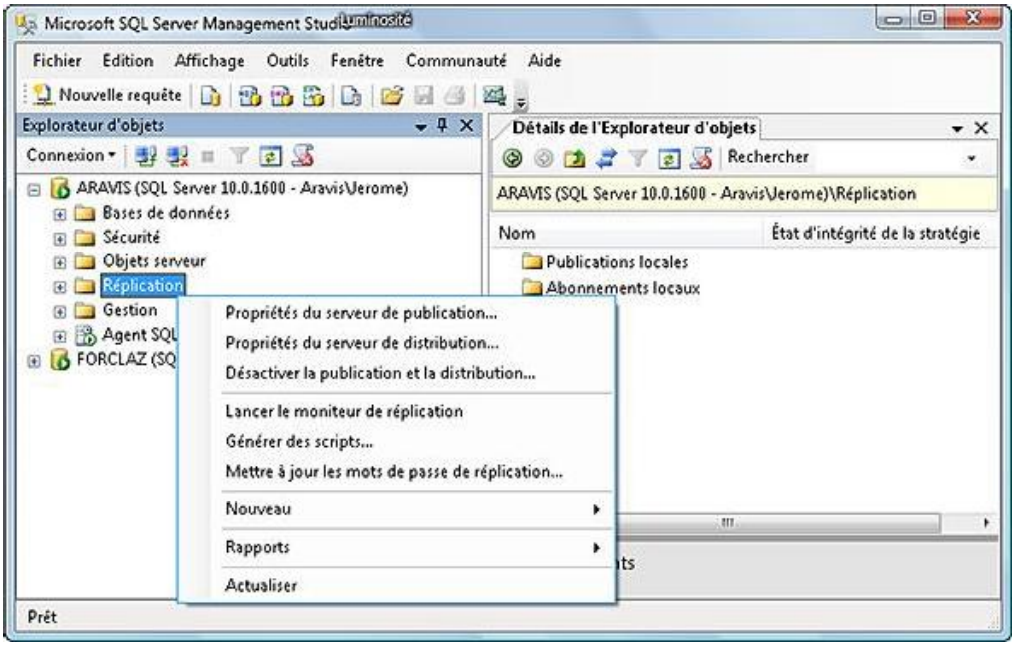
Enfin, l'assistant offre la possibilité soit de réaliser le paramétrage immédiatement, soit de générer les scripts Transact SQL correspondant aux différents choix spécifiés avec l'assistant, pour une exécution ultérieure de la mise en place du distributeur.

Tant que l'assistant n'est pas terminé, aucune modification n'est effectuée sur le serveur.

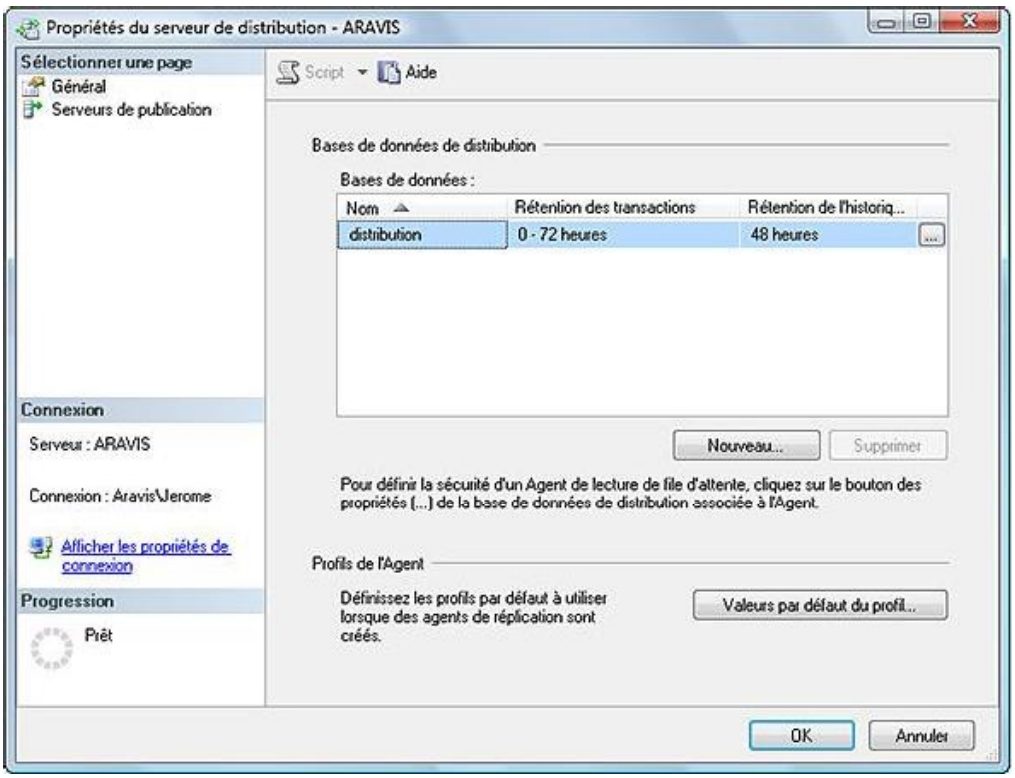


Apparition de la base de donnée distribution

Les options accessibles depuis SQL Server Management Studio sont légèrement différentes afin de pouvoir réaliser toutes les opérations liées à l'administration du serveur de distribution. Ces nouvelles fonctionnalités sont principalement accessibles par le menu contextuel associé au nœud **Réplication** de l'explorateur d'objets.



Le menu contextuel propose de visualiser les propriétés du serveur de distribution. Les propriétés du serveur de distribution permettent de paramétrer la distribution en elle-même comme la durée de rétention des transactions et celle de l'historique. Il est également possible de modifier la liste des éditeurs autorisés à utiliser ce serveur de distribution.



Il est également possible de lancer le moniteur de réplication afin de suivre le travail effectué par chaque agent qui participe à la réplication. Le moniteur de réplication permet de connaître l'état de chaque abonné et de suivre les opérations de synchronisation.

Le menu contextuel offre également la possibilité de désactiver la distribution.

Toujours depuis l'explorateur d'objets, il est possible de visualiser la base de données système distribution qui a été créée à l'aide de l'assistant.

2. L'éditeur

Après la création du distributeur, il est possible de créer un éditeur qui utilisera ce distributeur.

Un distributeur peut être commun à plusieurs éditeurs et un éditeur peut utiliser plusieurs distributeurs.

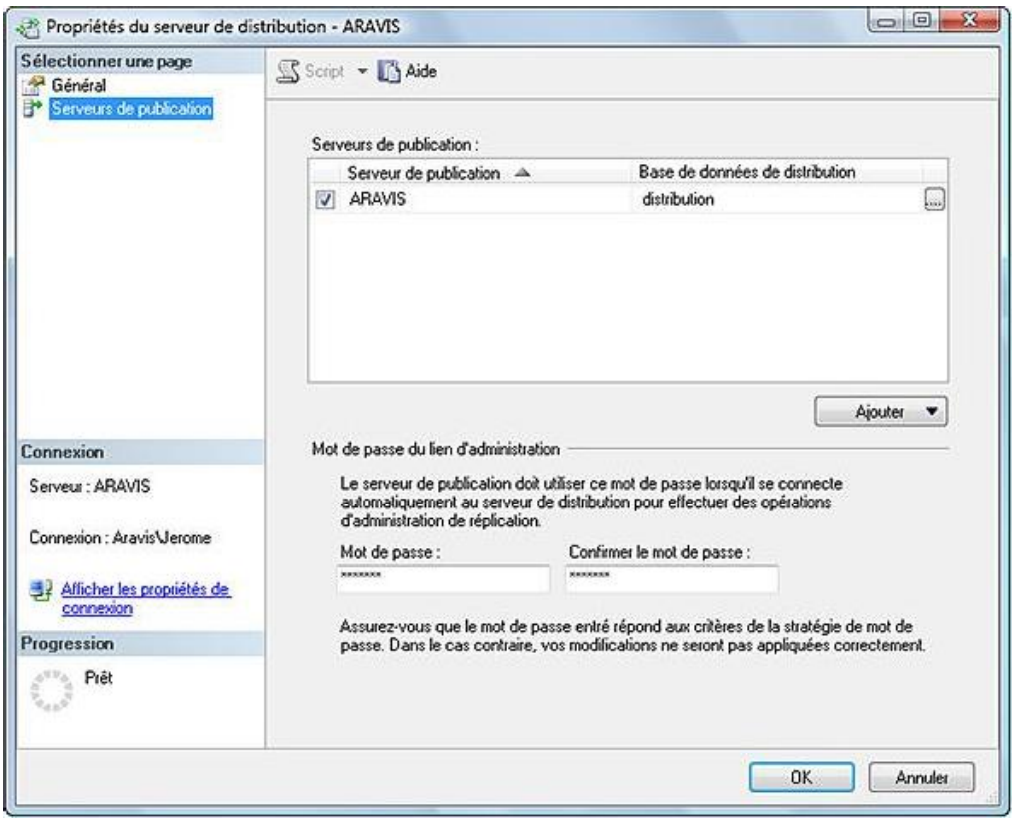
Lors de l'ajout d'un éditeur on peut :

- activer l'éditeur et l'autoriser à utiliser le distributeur.
- afficher ou modifier les options de l'éditeur.

L'ajout, la suppression et la modification d'un éditeur peuvent être réalisés de différentes façons :

SQL Server Management Studio

C'est à partir des propriétés du distributeur qu'il est possible de gérer (ajouter/supprimer) les serveurs de publication (éditeur) qui sont autorisés à utiliser ce distributeur.



Il est possible d'ajouter un distributeur SQL Server ou Oracle. Les procédures d'inscription étant différentes, le bouton **Ajouter** propose le choix entre les deux types d'éditeur avant de les autoriser à utiliser le distributeur.

Pour supprimer l'autorisation accordée à un distributeur, il faut désactiver le serveur de publication depuis les propriétés du serveur de distribution.

Transact SQL

Les opérations relatives à l'ajout et au retrait d'un éditeur sur le distributeur sont possibles par l'intermédiaire des procédures :

- **sp_adddistpublisher** pour ajouter un nouveau serveur de publication sur le distributeur. L'utilisation de cette procédure réclame l'exécution de la procédure sp_get_distributor afin de déterminer si l'instance de SQL Server est configurée ou non en tant que distributeur.

- **sp_dropdistpublisher** pour supprimer le serveur de publication du distributeur.

Modification de la base de données de distribution

Si un éditeur change de base de distribution, cela revient à tout recommencer depuis le début soit : désactiver l'éditeur, le supprimer de la base de données de distribution initiale, activer l'éditeur avec une autre base de données de distribution. Sur l'éditeur, il convient de créer les publications et les articles et d'activer les abonnés qui peuvent recevoir des données en provenance de l'éditeur via le distributeur.

3. Les publications

Il existe deux sortes de publications, celles de données et celles de procédures stockées.

Une publication, dans le cadre de la réplication SQL Server, correspond à un ensemble d'éléments (articles) qui participent à la réplication. Suivant le type de réplication choisi, ces articles peuvent être des tables, des tables partitionnées, des procédures stockées, des fonctions, des vues, des vues indexées, des types de données définis par l'utilisateur.

Lorsqu'une table participe à une publication, elle est nommée **article**. L'article correspond à la totalité de la table ou bien à un sous-ensemble de lignes ou de colonnes. Un article peut également correspondre à l'exécution d'une procédure stockée, cependant cette fonctionnalité n'est pas disponible dans la cadre d'une réplication de fusion.

C'est lors de la création d'une publication que l'éditeur va utiliser les ressources mises à sa disposition par le distributeur pour héberger les fichiers de capture instantanée ainsi que la copie des transactions sur la base de distribution.

Au moment de la création d'une publication, les éléments suivants doivent donc être résolus :

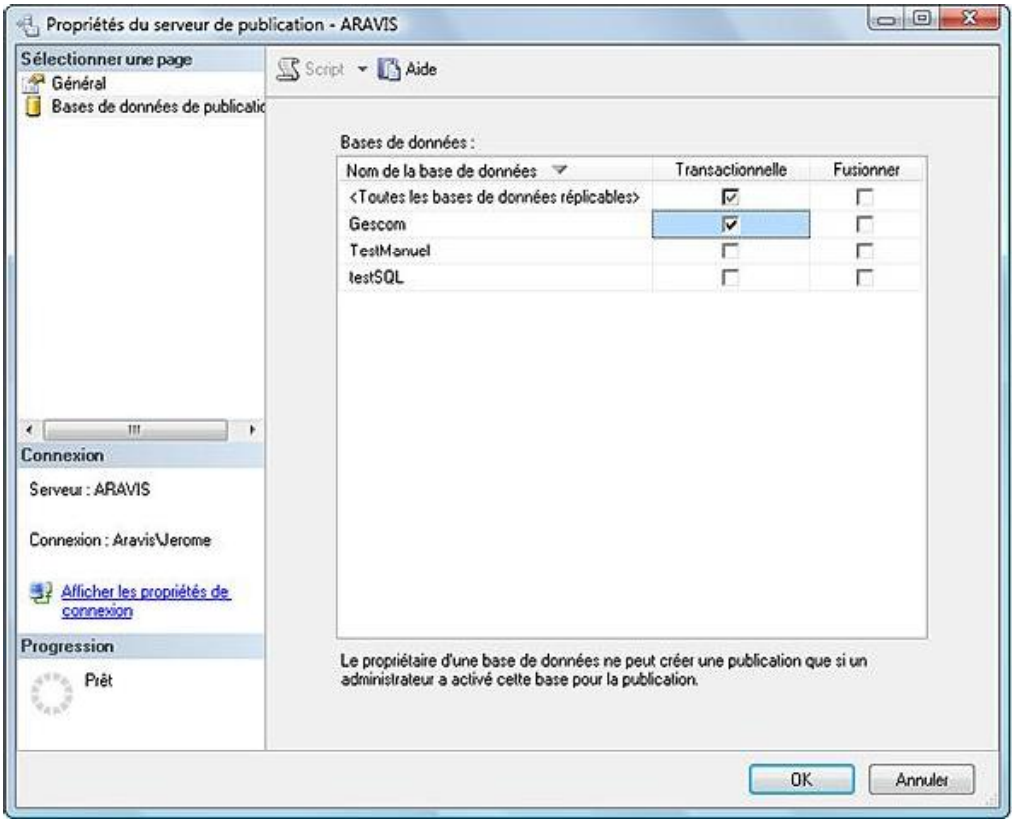
- le nom de la publication et sa description,
- le serveur de distribution à utiliser,
- le type de réplication (capture instantanée, transactionnelle ou fusion) à laquelle la publication va participer,
- les articles.

Tous ces éléments vont devoir être spécifiés lors de l'exécution de l'assistant de création d'une publication.

Création et modification

La création d'une publication par le propriétaire de la base de données (ou un utilisateur membre du rôle db_owner) n'est possible que si l'administrateur du serveur a activé la base pour participer à une réplication. Lors de cette activation, l'administrateur précise si les publications définies sur la base peuvent s'inscrire dans une réplication transactionnelle ou de fusion.

L'activation de la publication sur les bases est possible depuis la fenêtre présentant les propriétés du serveur de publication. Le choix **Propriétés du serveur de publication** depuis le menu contextuel associé au nœud **Réplication** de l'explorateur d'objets permet d'afficher cette fenêtre.

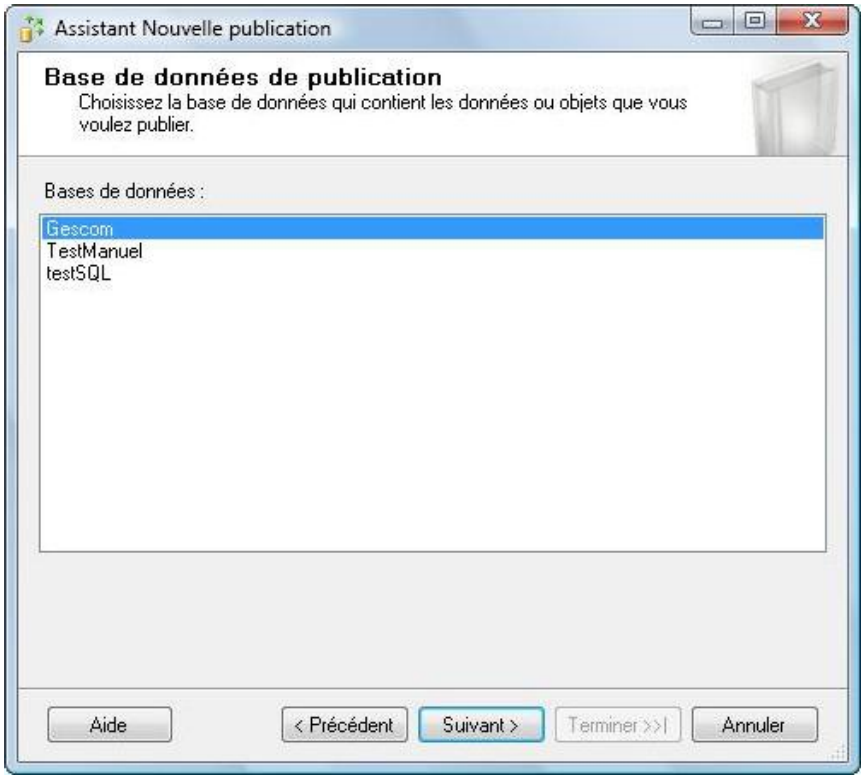


La base Gescom est activée pour la réplication transactionnelle

La création d'une publication est également réalisée par l'intermédiaire d'un assistant sous SQL Server Management Studio. Pour créer une nouvelle publication, il faut sélectionner **Nouvelle publication** dans le menu contextuel associé au nœud **Réplication - Publications locales** de l'explorateur d'objets.

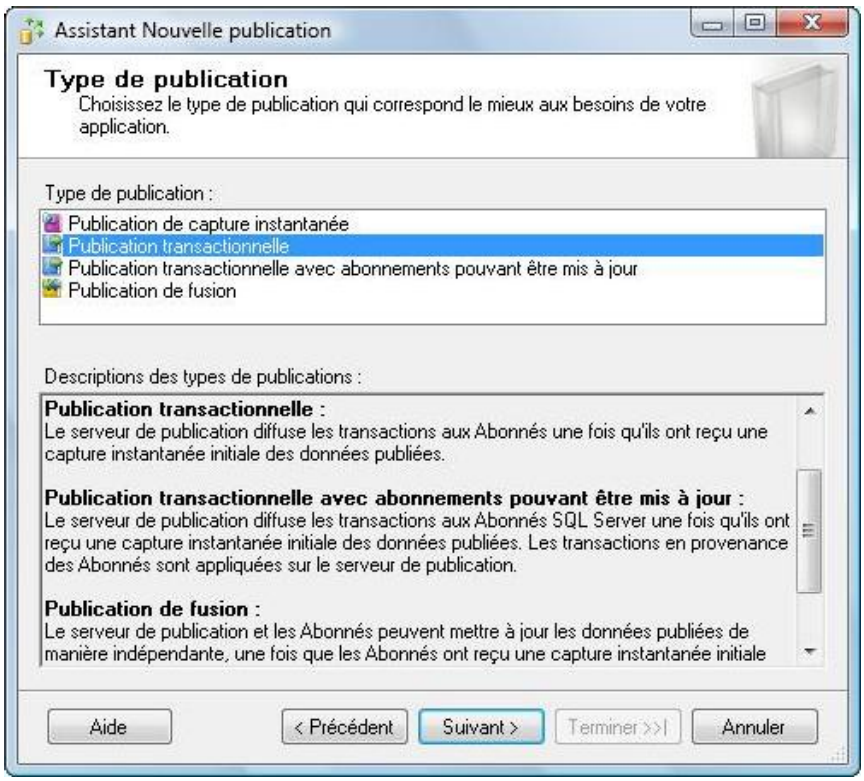
La première étape de l'assistant consiste à sélectionner la base de données qui contient le ou les objets qui vont participer à la publication. Cette boîte de dialogue montre toutes les bases de données utilisateurs, qu'elles soient ou non activées pour la réplication.

Dans l'exemple ci-dessous, seule la base Gescom est activée pour la réplication, alors que toutes les bases sont présentes dans la fenêtre.



L'assistant demande alors de préciser le type de réplication à laquelle cette publication va participer.

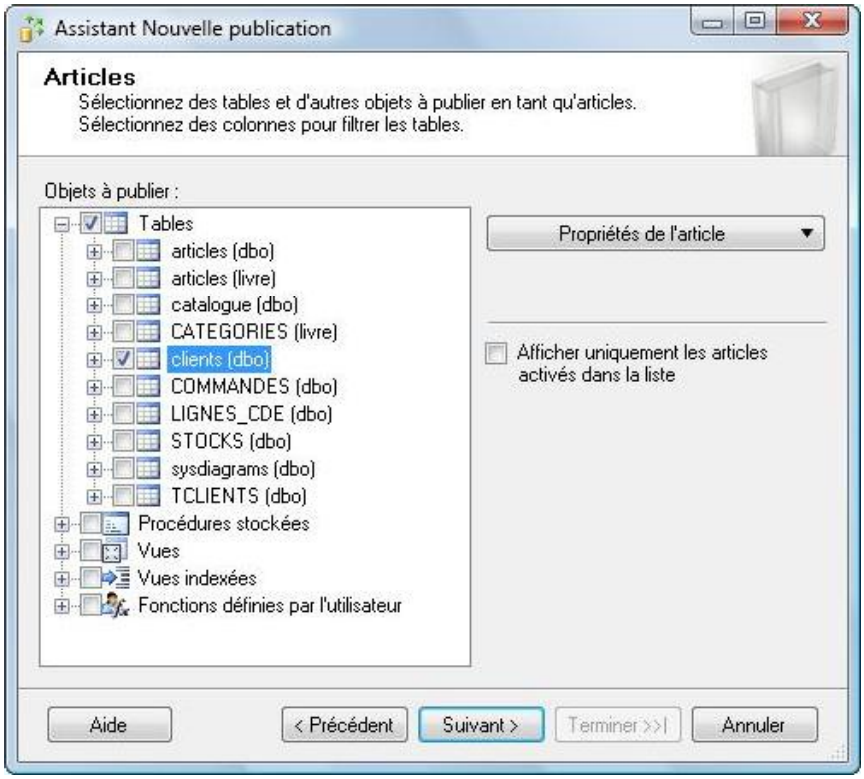
Dans l'exemple ci-dessous, la publication est définie dans le cadre d'une réplication transactionnelle, c'est-à-dire que les données sont modifiées sur l'éditeur et que ces modifications sont reportées sur les abonnés par le processus de réplication.



Le choix se porte sur une capture instantanée

Après que le type de réplication est défini, il est possible de sélectionner les éléments de la base qui participent à cette publication, c'est-à-dire définir les articles.

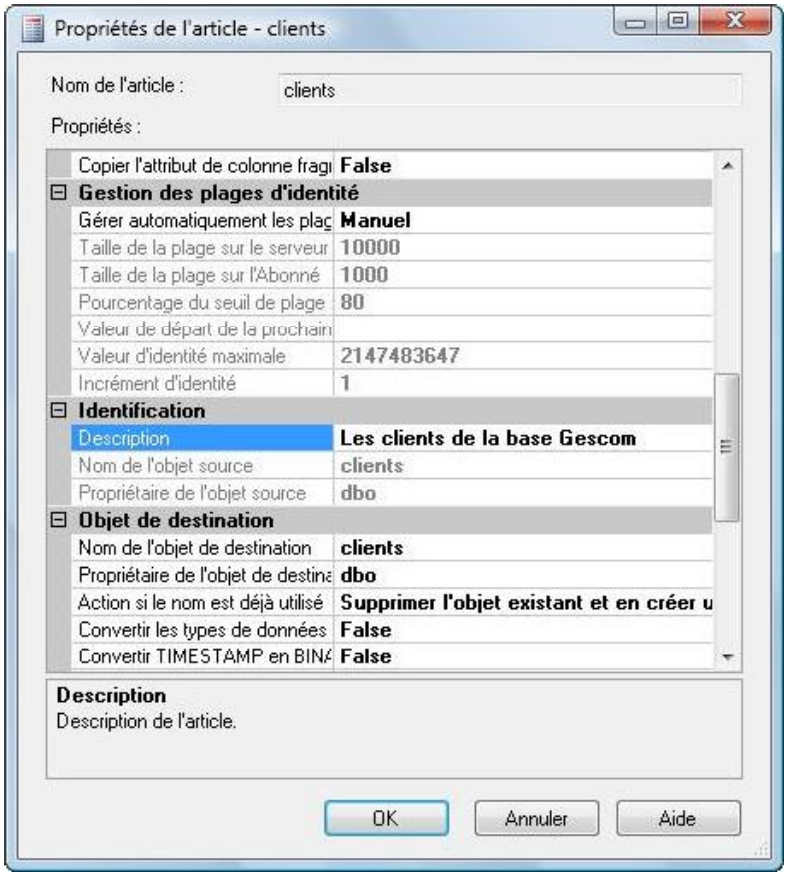
Dans l'exemple ci-dessous, la publication va compter un seul article : la table des clients.



Choix des articles à publier

Seules les tables qui possèdent une clé primaire peuvent participer à une publication en tant qu'article.

Pour chaque article publié, il est intéressant d'afficher ses propriétés afin de saisir une description pour chaque article et fixer le schéma et le nom de la table de destination.



Fixer les propriétés générales de chaque article

Les propriétés de l'article publié permettent de définir le comportement à adopter sur les index, les index XML, les valeurs par défaut, es autorisations.

L'assistant offre ensuite la possibilité de définir des filtres pour limiter le volume de données publiées. Par exemple, est-il nécessaire de publier la totalité de la table clients ou bien faut-il limiter la réplication aux seuls clients d'une zone géographique bien définie ?

Résumé

L'évolution des techniques informatiques depuis les vingt dernières années a permis d'adapter les outils informatiques à l'organisation des entreprises. Vu le grand volume de données manipulées par ces dernières ,la puissance des micro-ordinateurs, les performances des réseaux et la baisse considérable des coûts du matériel informatique ont permis l'apparition d'une nouvelle approche et ce en répartissant les ressources informatiques tout en préservant leur cohérence.

La solution qui s'impose est de distribuer les données et les organiser dans les bases de données sur différents sites de stockage. L'ensemble de ces sites constitue un système de bases de données réparties offrant la possibilité aux utilisateurs de manipuler les différentes bases via un réseau de manière transparente, comme dans une base de données globale.

Mon projet consiste à développer un système d'information dont les données sont intégrées dans un environnement réparti. L'objectif de ce travail est d'essayer de résoudre les problèmes de fragmentation et localisation des données et d'exécution distribuée des requêtes posées par la répartition d'une base de données.

Pour cela, j'ai conçu et mis en œuvre une base de données répartie sous SQL Server 2008 pour la gestion des ventes de produits en ligne au niveau de l'entreprise NWTRADERS, citée dans les TP de Microsoft.