

## Choisir sa base de données

*La gestion du stockage de l'information a toujours été un élément critique de toute application informatique. La gestion des fichiers, la recherche de l'information a très souvent complexifié les développements. Comment choisir ? Quels critères retenir ?*

L'utilisation d'une base de données a permis de simplifier et normaliser cette gestion pour les développeurs. Malgré cet intérêt, son utilisation a longtemps été limitée aux applications critiques et complexes du fait notamment des coûts induits ainsi que des compétences nécessaires à l'installation et à l'administration. L'installation et l'administration des bases de données se sont progressivement simplifiées, ce qui a eu pour effet de répandre leur utilisation, et plus récemment l'apparition de bases de données Open Source a complètement démocratisé leur utilisation à toutes sortes d'applications, simples ou complexes, critiques ou non critiques. Pour définir simplement une Base de Données, nous pouvons parler de collection de données inter reliées de façon cohérente. Un Système de Gestion de Bases de Données (SGDB) est un ensemble de programmes qui permettent à des utilisateurs de saisir, d'organiser et de sélectionner des informations dans la base de données.

Le SGDB le plus courant est un système de gestion de base de données relationnelle ou SGBDR, c'est est un logiciel capable de traiter des données structurées (tables, lignes, colonnes) dans un contexte de concurrence d'accès.

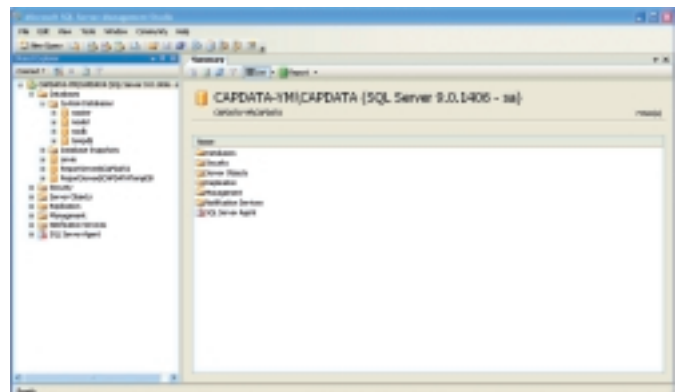
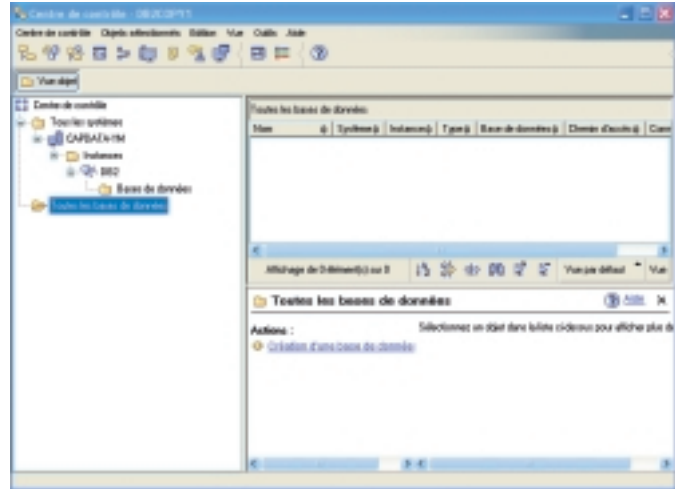
Les grandes caractéristiques d'un SGBDR sont :

- Indépendance face au type de stockage
- Indépendance des données face aux applications
- Accès efficace aux données
- Gestion de l'aspect transactionnel et des accès concurrents
- Manipulation des données simplifiées (Langage SQL)
- Assurance de la cohérence des données
- Sécurité physique (Reprise cohérente en cas d'incident)
- Sécurité logique (Gestion des droits, limitation d'accès, etc.)

Le Langage SQL permet de dialoguer avec un SGBDR. Le succès du langage SQL est dû essentiellement à sa simplicité d'utilisation et au fait qu'il s'appuie sur le schéma conceptuel pour énoncer des requêtes en laissant le SGDB responsable de la stratégie d'exécution. Ce langage permet d'effectuer diverses opérations comme la création, l'extraction, la modification, la suppression, la fusion, etc., sur des collections de données. Le langage SQL sera normalisé à cinq reprises (SQL 86, SQL89 SQL 2, SQL 3, SQL 2003).

### Les SGBDR de notre étude

Les SGBDR abordés dans cette article sont IBM DB2 V9, Oracle 10g, Sybase ASE 15 et Microsoft SQLServer 2005, MySQL v5.0 et PostgreSQL v8. Hormis Microsoft SQL Server, disponible uniquement sur Windows, les autres éditeurs proposent des distributions sur les plates-formes les plus utilisées (Windows, AIX, HP-UX, Solaris, GNU Linux, etc.). Ces distributions sont disponibles soit sous forme de binaires, soit sous forme de sources à recompiler. Actuellement, plus de 50 % des serveurs supportant des bases de données utilisent Windows. Linux connaît une forte progression par rapport à UNIX. Dans les SGBDR pris en compte pour cet article, les bases MySQL, PostgreSQL et



MS SQL sont les distributions les plus simples à installer.

De ce point de vue, MySQL a un net avantage à travers les packages de type Wamp ou Lamp qui associent, en plus du serveur de bases de données MySQL, un serveur apache et les bibliothèques PHP. Ce type de package le rend très attractif pour un développeur qui a choisi ce langage. Les SGBDR historiques sont plus complexes à installer en fonction de la plate-forme choisie et demandent des connaissances préalables plus importantes.

### Développer avec un SGBDR

Tous les SGBDR sont capables de dialoguer avec les langages de développement les plus courants (C, C++, PHP, Java, .NET, Perl, etc.). Le dialogue avec la base de données s'effectuera à travers des requêtes SQL dynamiques ou encapsulées dans des objets compilés (procédures stockées, fonction etc.). Afin de pallier une carence de la norme SQL, chaque éditeur a depuis toujours développé un langage complémentaire (SQL étendu) permettant notamment de gérer la notion de variable et de contrôle de flux (test, boucle ...).

L'utilisation des objets compilés comme les procédures stockées ou les

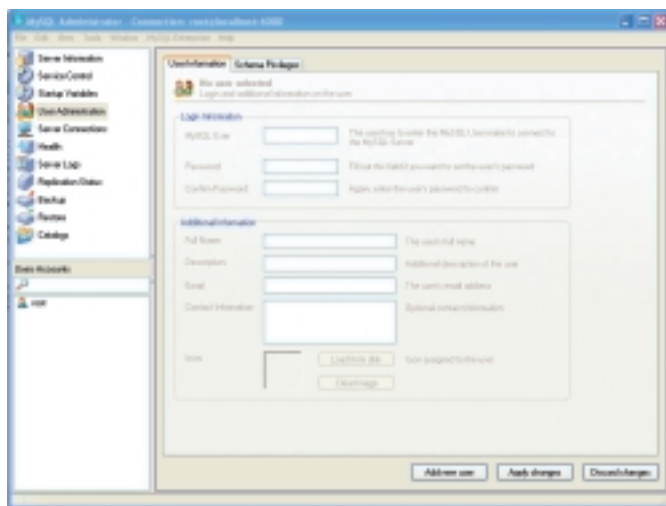
triggers permettant de faciliter la maintenance, la sécurité et la performance de traitements critiques, Par contre leur utilisation lie fortement l'application à l'éditeur du SGBDR. En effet, la syntaxe d'écriture d'un objet compilé n'est pas compatible d'un éditeur à un autre (hormis entre Sybase et MS SQL qui utilisent le même langage étendu, ce qui permet un certain niveau de compatibilité).

Le tableau ci-dessous liste les fonctionnalités de base nécessaires pour un développement et le niveau de compatibilité avec la norme SQL

	DB2	Oracle	Sybase	SQL SERVER	MySQL	PostgreSQL
Norme SQL	SQL92 et SQL99	SQL92 et SQL99	SQL92 et SQL99	SQL99	SQL92 et SQL99	En partie SQL2003
SQL Etendus	SQL PL	PL/SQL	Transact SQL	Transact SQL	SQL	PL/pgSQL
Richesse du langage étendu	Très riche	Très riche	Riche	Riche	moyen	Riche
Transaction	OUI	OUI	OUI	OUI	OUI	OUI
Procédures stockées	OUI	OUI	OUI	OUI	OUI	OUI
Trigger	OUI	OUI	OUI	OUI	OUI	OUI
Curseurs	OUI	OUI	OUI	OUI	OUI	OUI
Vues	OUI	OUI	OUI	OUI	OUI	OUI
Union	OUI	OUI	OUI	OUI	OUI	OUI
Fonctions (UDF)	OUI	OUI	NON	OUI	OUI	OUI
Contraintes	OUI	OUI	OUI	OUI	OUI sur les moteurs InnoDB et Ndb Cluster	OUI

Les SGBD sont aujourd'hui des produits matures et les fonctionnalités de base pour le développement actuellement mises en œuvre par les éditeurs sont sensiblement équivalentes.

Les éditeurs historiques ainsi que PostgreSQL proposent un langage étendu très complet. MySQL propose un langage étendu moins riche, mais suffisant. Il faut également noter que certaines fonctionnalités sont, relativement aux éditeurs historiques, très récentes sur le SGBD MySQL. Par exemple, les procédures stockées ou triggers ne sont disponibles que depuis la version 5 et sont actuellement encore peu implémentées, alors que sur tous les autres moteurs, l'utilisation des objets



compilés est très courante. Il est à noter que les SGBD Open Source sont ceux qui suivent au plus près les normes SQL (PostgreSQL a implémenté une partie de la norme Sql 2003)

MySQL a une approche spécifique, reposant sur plusieurs moteurs de stockage ; chacun répondant à des spécificités particulières et optimisé pour un mode d'utilisation (non transactionnel avec Myisam, transactionnel avec InnoDB etc).

## Unicode

Le stockage classique de jeux de caractères de type latin-1, cp850 ou ISO 8859-1 se fait sur 1 octet, ce qui ne permet pas de stocker des jeux de caractères complexes. Le standard Unicode est un mécanisme universel de codage de caractères. Il définit une manière cohérente de coder des textes multilingues et facilite l'échange de données textuelles. Obligatoire pour la plupart des nouveaux protocoles de l'Internet, il est mis en œuvre dans tous les systèmes d'exploitation et langages informatiques modernes, Unicode est la base de tout logiciel qui veut fonctionner aux quatre coins du monde. Les SGBDR ont, bien sûr, implémenté à différents niveaux les normes permettant de gérer l'unicode ( UCS-2, UCS-4, UTF-8, UTF-16 ).

	DB2	Oracle	Sybase	SQL SERVER	MySQL	PostgreSQL
UTF-8	OUI	OUI	OUI		OUI	OUI
UTF-16		OUI	OUI	OUI	OUI	
UTF-32						
UCS-2	OUI				OUI	
UCS-4						

L'utilisation du stockage unicode se fera essentiellement à travers les types NCHAR et NVARCHAR. Tous les SGBDR sont capables à différents niveaux de traiter l'UNICODE.

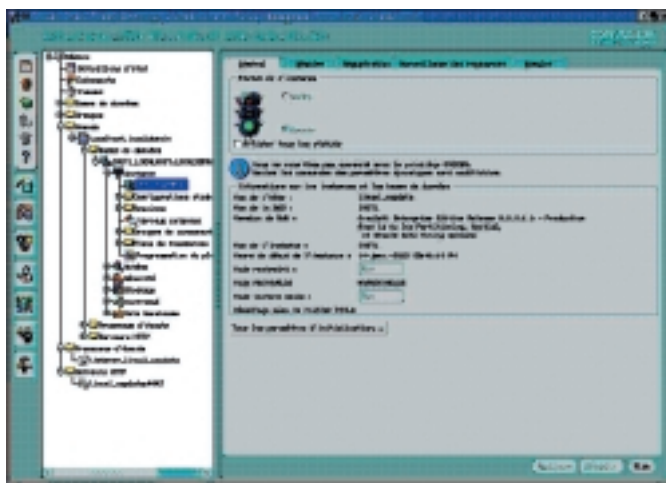
La gestion du "caractère set" et du langage peuvent être gérées à travers la notion de collation au niveau de la colonne pour MySQL et MS SQL Server. Les autres SGBD gèrent ces paramètres au niveau de l'instance ou au plus fin au niveau de la base.

## XML

XML (eXtensible Markup Language) est une norme d'échange de documents informatisés et est rapidement devenu un standard pour structurer les documents à transférer et à échanger à travers un réseau. XML est le standard émergent pour l'intégration de données en SOA (Architecture Orientée Services). Le SGBDR n'est pas forcément le moteur de stockage idéal pour les documents XML auquel nous préférons des bases "natives" de type hiérarchique, explicitement construites pour stocker du XML.

Toutefois, les bases relationnelles proposent deux solutions pour stocker les documents XML. La première stocke le document en bloc dans le champ d'une table tandis que la seconde extrait l'information du document pour la décomposer et la ranger dans différentes tables et champs.

La première solution préserve l'intégrité du document, par contre, elle rend difficile la manipulation des données à l'intérieur du document. La seconde simplifie la manipulation des données, mais elle ne permet pas de conserver le document dans sa forme originale et oblige donc à un mapping de schémas et des requêtes complexes pour les restituer. Si nous voulons conserver l'intégrité du document et faciliter la manipu-



lation nous serons par conséquent obligés, dans un moteur relationnel, d'effectuer un double stockage coûteux.

Une fois les documents stockés, il faudra pouvoir manipuler les formats XML. Pour cela nous pourrons utiliser XQUERY qui est un langage de manipulation et d'interrogation d'un document XML. C'est en quelque sorte aux données XML ce que SQL est aux données relationnelles. Très fortement lié à XPath, il s'en différencie par sa plus grande richesse fonctionnelle.

Bases propriétaires	DB2	Oracle	Sybase	SQL SERVER	MySQL	PostgreSQL
Stockage en base non structuré	OUI	OUI	OUI	OUI	OUI	OUI
Stockage en base structuré	OUI	OUI	OUI	OUI	NON	NON
Stockage format natif	OUI	NON	NON	NON	NON	NON
Licences en plus	OUI pour le format natif	NON	OUI	NON	NON	NON
XQUERY / Xpath	OUI	OUI	OUI	OUI	NON	NON
Import / export	OUI	OUI	OUI	OUI	Outil client MySQL avec option --xml	

La nouvelle version DB2 v9, avec sa technologie Hybride (relationnel et XML) offre un stockage natif de documents XML. Tous les acteurs historiques sont capables de gérer des documents XML. Les SGBD Open Source sont actuellement en retard sur ce sujet.

## La performance

Pour des applications standard sur une architecture correctement dimensionnée, la performance dépendra essentiellement de la qualité du développement et du paramétrage du serveur. Hormis quelques architectures spécifiques, la performance n'est pas l'argument principal pour le choix d'un SGBDR.

Les benchmark de type TPCC ou TPCB (<http://www.tpc.org/>) peuvent donner à un certain moment des éléments de performance pour un SGBDR, mais très souvent chaque benchmark est effectué en fonction de critères difficilement comparables avec d'autres résultats (version du serveur, architecture matérielle, etc.)

Le tableau ci-dessous compare un certain nombre de fonctionnalités

permettant d'adapter le paramétrage de l'instance dans le but d'obtenir de bonnes performances.

Bases propriétaires	DB2	Oracle	Sybase	SQL SERVER
Bases propriétaires	DB2	Oracle	Sybase	SQL Server
Optimiseur statistique	OUI	OUI	OUI	OUI
Cache de données	OUI	OUI	OUI	OUI
Cache de requêtes	OUI	OUI	OUI	OUI
Cache de procédures	OUI	OUI	OUI	OUI
Forcer l'optimiseur	OUI	OUI	OUI	OUI
Types d'index	Btree	Btree, Bitmap	Btree, Hash	Btree
Prepare statement	OUI	OUI	OUI	OUI
Multi-processeurs	OUI	OUI	OUI	OUI
Version 64 bits	OUI	OUI	OUI	OUI
Répartition des objets (Tablespace, devices, etc.)	OUI	OUI	OUI	OUI
Tables en mémoire	OUI	OUI	OUI (Caches nommés)	OUI
Parallélisme	OUI	OUI	OUI	OUI
Verrouillage	Lecture en attente sur verrouillage	Multi-versioning	Lecture en attente sur verrouillage	Lecture en attente sur verrouillage (SQL Server 2005, Lecture image avant avec "snapshot")
Granularité du verrouillage	ligne	ligne	ligne	ligne
Assistant de performance	OUI	OUI (option payante)	NON	OUI
Limitation de ressources au niveau de l'utilisateur	OUI	OUI	OUI	OUI

Bases Open sources	MySQL	PostgreSQL
Optimiseur statistique	OUI	OUI
Cache de données	OUI	OUI
Cache de requêtes	OUI	OUI
Cache de procédures	5.1	NON
Forcer l'optimiseur	OUI	OUI
Types d'index	Btree, Hash, Rtree	Btree, Rtree, Hash
Prepare statement	OUI	OUI
Multi-processeurs	OUI	OUI
Version 64 bits	OUI	OUI
Répartition des objets (Tablespace, devices, etc.)	OUI	OUI
Tables en mémoire	OUI (engine MEMORY)	NON
Parallélisme	Version 5.1	OUI
Verrouillage	Multi-versioning	Multi-versioning
Granularité du verrouillage	Ligne, page, table (en fonction des moteurs)	ligne
Assistant de performance	NON	NON
Limitation de ressources au niveau de l'utilisateur	OUI	OUI

Sybase et SQL Server (sans l'utilisation des snapshot SQL 2005) offrent un système transactionnel sans un système de lecture "multi-versioning", ce qui a pour conséquence de rendre plus délicat le développement et la gestion des accès concurrents (risque plus important de verrouillage et de deadlock etc.).

Des outils d'assistance à la performance apparaissent sur DB2, Oracle et SQL Server. Ces outils apportent des recommandations sur le paramétrage du serveur, l'indexation, etc. Pour l'instant, la plupart de ces conseils proposés par ces outils demandent une analyse fine et ne peuvent pas être appliqués automatiquement.

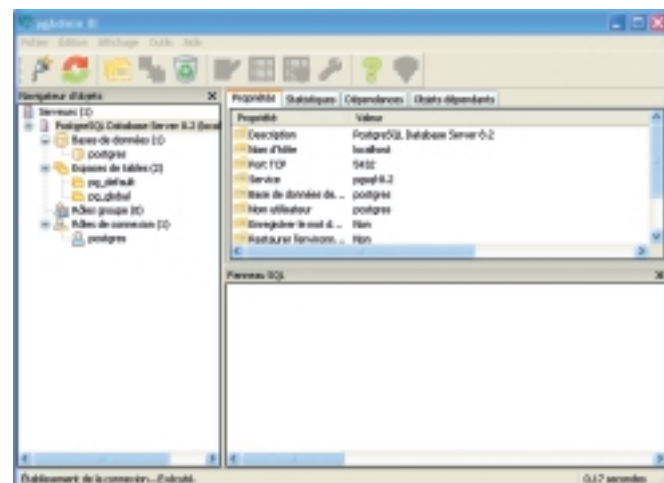
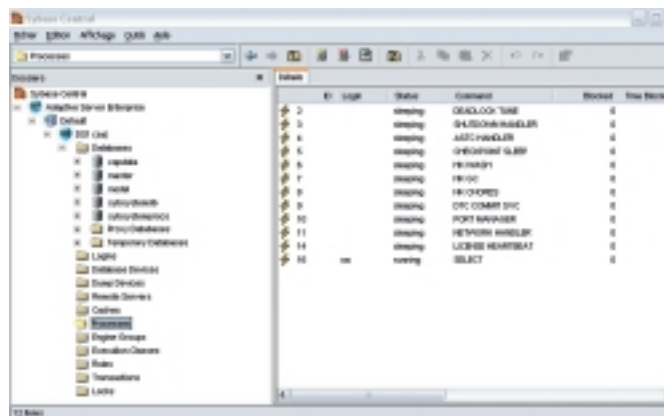
Hormis les assistants de performance, les fonctionnalités demandées pour répondre aux besoins de performance sont disponibles sur tous les SGBDR étudiés. Par contre, certaines de ces fonctionnalités comme la gestion des caches ou les limitations de ressources sont beaucoup moins sophistiquées sur les bases Open Source. Malgré la mise en œuvre de toutes ces fonctionnalités, l'application peut connaître des problèmes de performance. Il est par conséquent important de disposer d'outils permettant d'identifier et de diagnostiquer les problèmes rencontrés.

Ces problèmes sont, dans une forte proportion, des requêtes mal optimisées, et parfois un mauvais paramétrage de l'instance. Pour un développeur, il est très important de comprendre certains mécanismes de fonctionnement du SGBD comme la gestion des transactions et le verrouillage. La prise en compte, dès le départ, de ces mécanismes, est indispensable pour garantir un bon niveau de performance pendant toute la durée de vie du projet.

En effet, au départ du projet, hormis lorsque l'on fait des tests de montée en charge, ce qui est plutôt rare et coûteux, les requêtes s'exécutent rapidement, compte tenu du faible nombre d'accès concurrents et de la faible volumétrie, même lorsqu'elles sont mal écrites. C'est très souvent lorsque le projet rencontre le succès, avec notamment une forte augmentation de la volumétrie et du nombre d'accès concurrents, que l'on prend conscience du fait des problèmes de performance rencontrés, que les requêtes n'avaient pas initialement respecté un certain nombre de principes en phase avec les mécanismes de fonctionnement des SGBD.

Pour ces dernières il est important d'identifier leur plan d'exécution (Explain PLAN), le nombre d'E/S effectués, ou leur temps d'exécution. Chaque éditeur propose ses propres outils qui peuvent être plus ou moins riches et conviviaux.

Outils de tuning	
DB2	<ul style="list-style-type: none"> <li>• db2expln (Tuning de code SQL, analyse comportement optimiseur)</li> <li>• Visual explain, database monitoring</li> <li>• DB2 Query patroler</li> </ul>
Oracle	<ul style="list-style-type: none"> <li>• statspack (Indicateurs de performance d'une instance à un moment donné)</li> <li>• explain, tkprof (Tuning de code SQL, analyse comportement optimiseur)</li> <li>• ADDM : regroupe le management pack et le tuning pack</li> </ul>
Sybase	<ul style="list-style-type: none"> <li>• sp_sysmon : Indicateurs de performance d'une instance à un moment donné)</li> <li>• showplan, statistics io/time : Tuning de code SQL, analyse comportement optimiseur</li> <li>• Monitor Historical Server (Collecte des indicateurs de performance et historisation)</li> <li>• Les tables MDA (12.5.1)</li> </ul>
SQL Server	<ul style="list-style-type: none"> <li>• Profiler : Tuning de code SQL, analyse comportement optimiseur</li> <li>• Perfmon : outil générique de performance Windows, qui comportent un grand nombre de compteurs SQL Server.</li> <li>• DTA : database tuning advisor : permet de mémoriser de l'activité et de simuler des cas de figures</li> <li>• Showplan, statistics I/O</li> </ul>
MySQL	<ul style="list-style-type: none"> <li>• Explain (Tuning de code SQL, analyse comportement optimiseur)</li> <li>• Journal des requêtes lentes et des requêtes qui n'utilisent pas les index</li> </ul>
PostgreSQL	<ul style="list-style-type: none"> <li>• explain (Tuning de code SQL, analyse comportement optimiseur)</li> </ul>



Les bases ouvertes (MySQL et PostgreSQL) ont des outils ou commandes minimalistes, mais néanmoins suffisants. Les autres éditeurs ont de nombreux outils, souvent très riches fonctionnellement, mais pas toujours très simples à implémenter et à utiliser, hormis l'outil "profiler" de SQL Server qui apparaît comme étant l'outil le plus simple à installer et à utiliser.

## L'administration

La performance et la disponibilité des bases de données dépendent également des tâches d'administration qui sont régulièrement effectuées. En plus des sauvegardes qui apparaissent comme la tâche d'administration la plus importante, il est important, en fonction des besoins, de réorganiser, vérifier et optimiser les accès aux données. Ces tâches ont tendance à s'automatiser et deviennent progressivement transparentes pour le DBA. Dans un souci de disponibilité de la base, on va demander à ce que les actions d'administration soient les moins bloquantes possibles, et puissent être effectuées dynamiquement.

Les sauvegardes peuvent se dérouler suivant trois techniques différentes :

- **Base fermée** : le serveur de base de données doit être arrêté et une sauvegarde de tous les fichiers physiques est effectuée. Méthode qui fonctionne dans tous les cas quel que soit le SGBDR. Attention à ne pas oublier de sauvegarder tous les fichiers utiles pour la base de données.
- **Base ouverte** : la sauvegarde est effectuée à travers des commandes

au niveau du SGBDR, sans interrompre l'activité sur la base. La sauvegarde base ouverte permet de travailler sur des bases disposant de peu ou pas de plage disponible pour les tâches d'administration (Accès 24/7, international, etc.)

- **Par Cliché** : la base est "gelée" au niveau mise à jour le temps de la recopie des données mais reste accessible en lecture. Cette solution est souvent couplée avec un système de "snapshot" proposé par les fabricants de baie de disques. Essentiellement utilisés pour des grosses volumétries.

Une fois la technique choisie, il faut distinguer trois modes de sauvegarde :

- **FULL** : sauvegarde la base complète

- **TRANSACTIONNEL** : sauvegarde du journal de transaction

- **DIFFERENTIEL** : sauvegarde de ce qui a été modifié depuis la dernière sauvegarde complète. L'avantage par rapport à une sauvegarde incrémentale, c'est que lorsque l'on recharge une base il suffit de recharger la base complète suivie de la dernière sauvegarde différentielle

	DB2	Oracle	Sybase	SQL Server	MySQL	PostgreSQL
Vérification de bases	OUI	OUI	En ligne	OUI	OUI	NON
Ordonnancement de tâches intégré	OUI	OUI	OUI	OUI	OUI	Utilisation du scheduler OS AT, CRON etc.
Réorganisation des tables et index	OUI	En ligne	OUI	En ligne	OUI	OUI
Calcul des statistiques	OUI	AUTO	AUTO	AUTO	OUI	AUTO
Sauvegarde base ouverte	OUI	OUI	OUI	OUI	- Mysam : Via replication - Innodb : Outil HotBackup de la société Innodb (licence supplémentaire) - NDB cluster : OUI	OUI
Sauvegardes transactionnelles	OUI	OUI	OUI	OUI	OUI	OUI
Sauvegardes différentielles	OUI	OUI	NON	OUI	NON	NON
Compression des fichiers de sauvegarde à la volée	OUI	OUI	OUI	NON	OUI	OUI
Sauvegarde sur plusieurs fichiers	OUI	OUI	OUI	OUI	NON	NON
Cliché (*)	OUI	OUI	OUI	OUI	OUI	NON

Toutes les bases étudiées proposent les outils nécessaires aux différentes tâches d'administration. Sur les SGBD historiques, ces traitements sont plus rapides et moins bloquants sur des volumétries importantes comparé aux outils des bases Open source.

Le système qui assure la sauvegarde complète proposée par Sybase et MS SQL Server apparaît comme étant le plus simple à mettre en œuvre. Pour MySQL la multiplication des moteurs a tendance à complexifier les tâches d'administration, notamment au niveau des sauvegardes. De ce

fait, les outils proposés sont divers et variés et dans certains cas peu performants sur de gros volumes. PostgreSQL propose un système simple et moins performant sur les gros volumes comparé à Oracle, DB2, Sybase ou MS SQL Server.

Tous les éditeurs proposent des outils permettant d'administrer les serveurs de base de données à travers des interfaces graphiques.

## Outils d'administration

DB2	• Centre de contrôle
Oracle	• Enterprise Manager
Sybase	• Sybase Central
SQL Server	• Sql Advantage • SQL Server management Studio
MySQL	• MySQL administrator • MySQL Query Browser
PostgreSQL	• Projet open source : phppgadmin

## La gestion de la sécurité

Ce que l'on demande à un SGBDR c'est de pouvoir assurer la sécurité d'accès aux données en proposant une gestion de l'identification, des droits sur les données et du cryptage de l'information. Quand on parle de cryptage de l'information, cela se situe généralement soit au niveau de la connexion pour sécuriser les échanges avec le serveur de base de données, soit au niveau de la base elle-même en sécurisant les données. Dans ce dernier cas, le cryptage sera assuré soit en utilisant directement au niveau de l'application des fonctions de cryptage/décryptage, soit en utilisant une fonctionnalité qui propose un cryptage automatique au niveau de la base en utilisant des systèmes de certificat.

Des bases comme DB2, Oracle et Sybase proposent des systèmes de cryptage directement au niveau de la base, les autres SGBD proposent du cryptage uniquement à travers l'utilisation de fonctions.

	DB2	Oracle	Sybase	SQL Server	MySQL	PostgreSQL
Gestion des utilisateurs au niveau OS	OUI	OUI	NON (Hormis windows)	OUI	NON	OUI
Gestion des utilisateurs Au niveau SGBD	OUI	OUI	OUI	OUI	OUI	OUI
Gestion des droits sur les objets	OUI	OUI	OUI	OUI	OUI	OUI
Gestion des rôles et des groupes	OUI	OUI	OUI	OUI	NON	OUI
Interfaçage avec un annuaire LDAP	OUI	OUI	OUI	OUI	NON	OUI
Outil d'audit	OUI	OUI	OUI	OUI	OUI (log de requêtes), mais fort impact sur les performances	OUI

Comparer aux autres SGBDR, MySQL n'a pas pour l'instant toutes les fonctionnalités attendues pour faciliter la gestion des accès. Cela devrait se corriger en partie sur les versions prochaines. Oracle propose les outils les plus sophistiqués pour protéger et tracer tous les accès à la base de données (gamme de produits " Vault " etc.)

## La haute disponibilité

L'internationalisation des accès et la possibilité d'accès 24/24 via Internet conduit de plus en plus à mettre en œuvre des solutions de haute disponibilité. Pour répondre à ce besoin, les éditeurs de base de donnée proposent des solutions cluster. Le principe du cluster est de proposer plusieurs points d'accès aux données (nœuds). Si un nœud est perdu, les utilisateurs pourront continuer à accéder aux données via les autres nœuds disponibles. Suivant les éditeurs, nous serons face à une organisation de type ACTIF/ACTIF ou ACTIF/ PASSIF.

Dans le cas d'une organisation ACTIF / PASSIF, le nœud PASSIF est considéré comme un nœud de secours pour une application et sera activé uniquement lors de la perte du nœud ACTIF. Dans le cas d'une organisation ACTIF / ACTIF, les accès aux données pourront se faire à travers les 2 nœuds simultanément (cette solution a pour avantage de répartir la charge sur les différents nœuds pour une même application) Deux architectures types se dégagent par rapport aux solutions étudiées. La première architecture appelée "share nothing" s'articule sur des nœuds complètement indépendants les uns des autres. Chaque nœud a ses propres ressources CPU, mémoire, réseau et disque. Implémentée par les solutions DB2 et MySQL

La deuxième architecture appelée "Share disk", met en commun les ressources disques, cela veut dire que 2 nœuds peuvent accéder en même temps à une même ressource disque. Cette solution oblige à voir des disques partagés entre ces différents nœuds à travers des baies SAN, NAS ou autres. Le modèle Shared disk" est implémenté en mode ACTIF / ACTIF, par la solution Oracle RAC et dans la future version Sybase ASE

Bases propriétaires	DB2	Oracle	Sybase	SQL Server
Type	ACTIF / ACTIF	ACTIF / ACTIF	ACTIF / PASSIF	ACTIF / PASSIF
Couche cluster obligatoire	Cluster standard du marché	CRS ou cluster standard du marché	Cluster standard du marché	Cluster Windows
Le basculement vu par le client	manuel	TAF	Par codage au niveau de l'application	Par codage au niveau de l'application

Bases Open sources	MySQL	PostgreSQL
Type	ACTIF / ACTIF	ACTIF / PASSIF
Couche cluster obligatoire	Utilisation du moteur Ndb Cluster	Projets open source (Slony)
Le basculement vu par le client	manuel	Transparent via le projet open source PgPool

En matière de solution ACTIF / ACTIF, Oracle est le leader avec une solution éprouvée qui est utilisée en production depuis plusieurs années sur système ouvert. Avec son moteur NdbCluster, MySQL apporte une solution à faible coût de type ACTIF / ACTIF. Même si cette solution est encore " jeune " et impose actuellement un certain nombre de contraintes d'utilisation, elle pourrait devenir progressivement une solution de plus en plus intéressante. DB2 apporte également sa propre solution ACTIF / ACTIF que l'on rencontre surtout sur système propriétaire. La solution cluster de Microsoft est une architecture largement utilisée et éprouvée. Afin de pouvoir utiliser tous les nœuds y compris les nœuds PASSIF, les applications seront réparties à travers des services cluster différents. Dans ce cas nous pourrions dire que pour une application un

nœud sera considéré comme ACTIF et pour une autre application ce même nœud sera considéré comme PASSIF.

## La réplication

Les systèmes de réplication permettent de dupliquer des informations entre plusieurs bases de données. Fortement utilisée dans les années 90 sur des architectures distribuées afin de contourner les faibles débits des réseaux, la réplication est de nos jours plus souvent utilisée afin d'alimenter des serveurs de secours, répartir des traitements en séparant notamment les traitements de type transactionnel (TP) et décisionnel, ou répartir la charge en lecture.

Bases propriétaires	DB2	Oracle	Sybase	SQL Server
Outil de réplication intégré	OUI	OUI	OUI	OUI
Outil de réplication demandant une licence	WebSphere Information Integrator	Advance Replication	Replication server	NON
Réplication partielle	OUI	OUI	OUI	OUI
Réplication hétérogène	DB2 -> autre Autre -> DB2	Oracle -> autre	Autre -> Sybase Sybase -> autre	MSSQL -> autre Oracle -> MSSQL

Bases Open sources	MySQL	PostgreSQL
Outil de réplication intégré	OUI	NON
Outil de réplication demandant une licence	NON	Projet open source externe ( Slony )
Réplication partielle	OUI	NON
Réplication hétérogène	NON	NON

Sybase et Microsoft sont les éditeurs qui apportent le système de réplication le plus sophistiqué et le plus utilisés en production. Par ailleurs, MySQL apporte un système de réplication simple, robuste et facile à mettre en œuvre, ce qui séduit de plus en plus d'utilisateurs avec essentiellement un usage en serveur de secours ou pour faire une répartition de charge

## Quel SGBD pour quel type de projets ?

Si on classifie de manière arbitraire les applications en petites, départementales (volumétrie > 5 Go et < 50 Go et utilisateur > 10 et < 100 ) et entreprise, et si on regarde les fonctionnalités disponibles sur les différents SGBD étudiés il apparaît que tous les SGBD sont capables de répondre à des architectures départementales.

Pour de " petites applications ", du fait du coût et de la simplicité de mise en œuvre les bases Open Source MySQL et PostgreSQL peuvent s'avérer être un excellent choix. Pour des applications entreprise, les bases historiques que sont Oracle, DB2, SQL Server et Sybase s'avèrent être de bons choix, et notamment du fait qu'actuellement l'administration, la sécurité et la gestion des ressources des bases Open source n'offrent pas toutes les facilités et les performances attendues. La rapide progression d'un SGBDR comme MySQL, si celle-ci se confirme sur les deux prochaines années, nous laisse à penser que si telle est la volonté de la société MySQL AB, le SGBD MySQL pourrait progressivement être de plus en plus utilisé pour des applications de type entreprise.

Notons que cette progression en terme d'extension, d'applications de type départemental vers des applications de type entreprise (forte criti-

cité en termes de volumétrie, d'accès concurrents, de disponibilité etc.), a été constaté ces dernières années pour le SGBD MS SQL Server. Il est probable que l'amélioration de la fiabilité du système d'exploitation Windows, l'amélioration de la robustesse du moteur de bases de données SQL Server, associés à l'apport de nouvelles fonctionnalités ont majoritairement contribué à cette progression.

Pour des applications Web, les bases de données les plus utilisées sont MS SQL Server et MySQL. Les critères de choix sont essentiellement le coût et la facilité d'installation, (notamment à travers les packages de type WAMP et LAMP pour MySQL)

Actuellement, tous les éditeurs historiques, dans le but de permettre à leurs clients d'initier des projets à moindre coût, proposent une version gratuite, limitée en terme d'utilisation et correspondant à des configurations de faible importance :

- Oracle 10g Express Edition, (limitée à 4 Go sur disque, 1Go de mémoire, 1 CPU en environnement Linux/Windows),
- DB2 Express-C (limitée à 4 Go de mémoire, 2 CPU),
- Sybase ASE Express Edition (limitée à 5Go sur disque, 2Go de mémoire, 1 CPU en environnement Linux),
- SQL Server Express Edition (limitée à 4Go sur disque, 2Go de mémoire, 1 CPU).

L'offre Open source, de par son prix, est une solution séduisante, mais qui dit open source ne dit pas forcément gratuit pour autant, en fonction de l'usage qui est fait du programme et suivant les codes qui dictent les

règles de l'open source. En effet, si vous développez par exemple une application qui utilise MySQL et que vous vendez votre solution au lieu de la proposer en Open source, vous devez payer une licence commerciale à MySQL AB. La traduction officielle en français de la FAQ sur la licence GPL se trouve à l'adresse : <http://www.gnu.org/licenses/gpl-faq.fr.html>

Comme nous le disions, par son coût et sa facilité d'installation, les bases Open source séduisent de plus en plus d'entreprises, y compris les grands comptes. Grâce à sa politique commerciale MySQL arrive en tête de la préférence des bases Open source, même si à ce jour certaines fonctionnalités communes apparaissent comme plus riches sur PostgreSQL (le langage procédural PL / pgSQL par exemple)

Du fait que cette base constituée de plusieurs moteurs est propriété en grande partie de la société MySQL AB, elle apporte une certaine garantie sur la pérennité du produit. Pour PostgreSQL, l'approche est différente, car ce moteur de base de données repose sur une communauté, ce qui ne va pas forcément rassurer les grandes entreprises, sauf si celles-ci décident d'investir dans la communauté ou la connaissance des sources afin de garantir une certaine pérennité du produit.

#### ■ Yves MOULIN

Directeur Technique

Cap Data Consulting - Spécialiste des bases de données

CONSEIL, SERVICE, FORMATION, INFOGERANCE SGBD

[www.capdata.fr](http://www.capdata.fr)

## Maîtriser les relations développeur – base de données

*Même pour celles qui ne sont pas qualifiées d'application de gestion, de nos jours la plupart des applications vont adresser la problématique de gestion de données. C'est dire combien le développeur est amené à côtoyer quotidiennement les bases de données.*

Par ailleurs, ces dernières années ont connu l'émergence de méthodologies agiles qui ont contribué à modifier les relations historiques entre le développeur et le DBA. Il y a encore quelques décennies, la base de données était considérée comme un concept monolithique au travers duquel l'ensemble des développeurs d'un projet recevait la vision unique du DBA, maître absolu des modèles et du jeu de test. On assiste aujourd'hui, et aussi grâce à la puissance matérielle de plus en plus disponible sur les PC, à un foisonnement d'instances et schémas qui sont des replica d'un maître unique de la structure et du contenu des données du projet. Ainsi sur le seul périmètre de développement, donc sans parler des phases de qualification et de production, on retrouve plusieurs configurations qui combinent les éléments suivants selon la nature et la taille du projet :

- La base de données du développeur : véritable bac à sable du développeur, elle lui permet d'être isolé du reste du groupe et d'effectuer des modifications sans impact sur les autres membres de l'équipe. Seules les modifications jugées concluantes seront promues sur la base de données du projet ;
- La base de données du projet : c'est la référence à la fois de la structure et de contenu tant sur les données de référence que du jeu d'es-

sai. Elle est sous la responsabilité d'un DBA ou éventuellement un développeur avec des connaissances de base de données. Elle alimente régulièrement les bases de données des développeurs du projet et aussi celle d'intégration continue ;

- La base de données d'intégration : c'est dans cette base que s'exécuteront les tests prévus dans le cadre de l'assurance qualité. Il n'est pas rare que cette base de données soit de la responsabilité des DBA de production.

La responsabilité de ces différentes bases de données est partagée entre le DBA et le développeur. Le dénouement de cette coresponsabilité dépend très souvent de la relation entre le DBA et le développeur. Cette relation est quant à elle le fruit de la culture et de l'existant de l'entreprise. La culture mainframe a généralement débouché sur le DBA perché dans sa tour d'ivoire et qui décide de tout sur " ses " bases de données. A l'inverse, l'adoption des " petits " systèmes a largement contribué à avoir des DBA accessibles et proches des développeurs.

Le contexte " tour d'ivoire ", surtout lorsque les infrastructures sont mutualisées, conduit fréquemment à des relations conflictuelles entre le DBA et le développeur. Ces conflits sont liés à deux objectifs antagonistes : le DBA veut fournir une plate-forme qui respecte la qualité de