1

Introduction aux SGBDR

Pour optimiser une base Oracle, il est important d'avoir une idée de la manière dont elle fonctionne. La connaissance des éléments sous-jacents à son fonctionnement permet de mieux comprendre ses comportements. C'est pourquoi nous commencerons, à ce chapitre, par vous présenter ou vous rappeler brièvement ce qu'est un SGBDR et quelques-uns des éléments qui le composent. Ces notions vous seront utiles tout au long du livre. Le but ici n'est pas de faire de vous un spécialiste du fonctionnement interne d'Oracle, mais de vous donner quelques explications sur des concepts que nous référencerons ultérieurement.

Si vous êtes DBA, vous pouvez vous rendre directement au Chapitre 2.

1.1 Qu'est-ce qu'une base de données ?

Une base de données est un ensemble d'informations structurées. Elle peut être de nature :

- hiérarchique ;
- relationnelle ;
- objet ;
- documentaire ;
- …

Actuellement, le marché est principalement composé de bases de données relationnelles avec, parfois, une extension objet. Cet ouvrage traite exclusivement de ce type de bases.

Dans les bases de données relationnelles, les données sont organisées dans des tables à deux dimensions, conformément au modèle relationnel que nous étudierons au Chapitre 2.

1.1.1 Système de gestion des bases de données

Un SGBDR (système de gestion de base de données relationnelle) est un système (logiciel) qui permet de gérer une base de données relationnelle. Les SGBDR peuvent être soit de type client/serveur (Oracle, SQL Server, MySQL, PostgreSQL, etc.), soit de type fichiers partagés (Access, SQL Server CE, Paradox, DBase, etc.). Dans le milieu professionnel, on retrouve principalement des SGBDR client/serveur même si les solutions en fichiers partagés ont eu leur heure de gloire et sont encore utilisées dans certaines applications. L'apparition de SGBDR client/serveur gratuits a fortement contribué à populariser ce modèle ces dernières années. Le modèle client/serveur nécessite généralement la présence d'un serveur, qui traite les requêtes transmises par le client et lui retourne le résultat. Le principal intérêt d'un SGBDR est qu'il va fournir les services suivants (tous les SGBDR ne proposent pas tous ces services) :

- implémentation du langage d'interrogation des données SQL (*Structured Query Language*) ;
- gestion des structures des données et de leur modification ;
- gestion de l'intégrité des données ;
- gestion des transactions ;
- gestion de la sécurité, contrôle d'accès ;
- abstraction de la plateforme matérielle et du système d'exploitation sous-jacent ;
- du point de vue logique, abstraction de l'organisation du stockage sous-jacent ;
- tolérance aux pannes et administration du système ;
- répartition de la charge.

1.2 Modèle de stockage des données

1.2.1 Organisation des données

Les tables sont les objets logiques de base du modèle relationnel. Or, un système d'exploitation ne connaît que la notion de fichiers. Le SGBDR permet de faire le lien entre la représentation logique et le stockage physique dans les fichiers. Nous allons voir comment en étudiant le SGBDR Oracle.

Les objets logiques (tables, index, etc.) sont stockés dans des espaces logiques appelés "tablespaces". Une base de données est constituée de plusieurs tablespaces, lesquels sont composés d'un ou de plusieurs fichiers appelés "datafiles". Ces fichiers peuvent se trouver sur des disques différents.

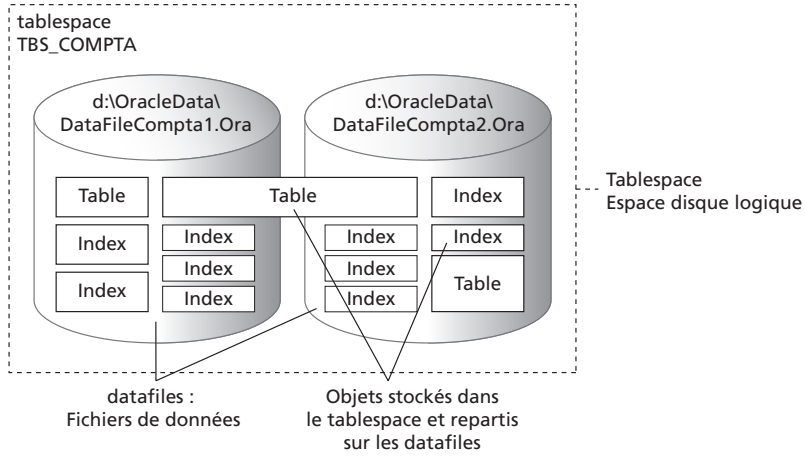


Figure 1.1

Organisation des objets dans un tablespace composé de deux datafiles.

Les objets sont attachés à un seul tablespace mais ils peuvent, par contre, être répartis sur plusieurs datafiles (voir Figure 1.1). Il n'y a aucun moyen d'influer sur la localisation des objets au niveau des datafiles, il est seulement possible de définir le tablespace associé à un objet. (Les objets partitionnés peuvent, eux, être attachés à plusieurs tablespaces. Nous les étudierons au Chapitre 5, section 5.3.4, "Partitionnement des données".)

Les datafiles, et donc les tablespaces, sont constitués de blocs de données (*data block*), dont la taille est configurée à la création du tablespace. Ce paramètre varie généralement entre 2 Ko et 16 Ko et, par défaut, est de 4 ou 8 Ko suivant la plateforme

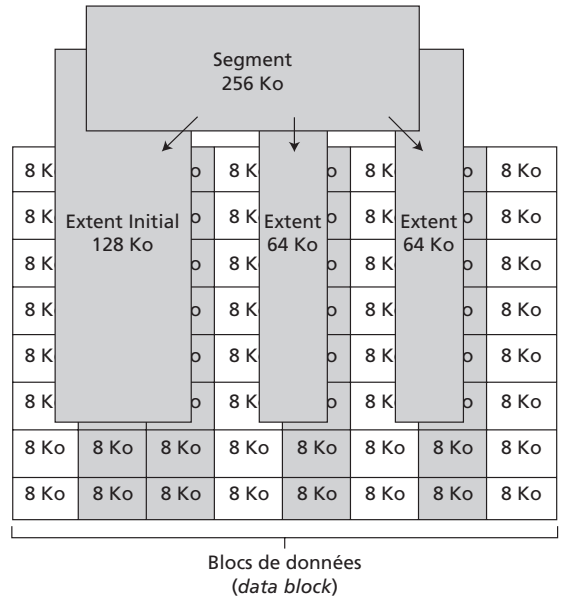
(au long de l'ouvrage, nous retiendrons une taille de 8 Ko qui est une valeur assez communément utilisée).

Chaque objet ou constituant d'objet ayant besoin de stockage s'appelle un "segment". Par exemple, pour une table classique, il y a un segment pour la table elle-même, un segment pour chacun de ses index et un segment pour chacun de ses champs LOB (voir Chapitre 2, section 2.2.1, "Les types sous Oracle").

Quand un segment a besoin d'espace de stockage, il alloue un *extent*, c'est-à-dire un ensemble de blocs de données contigus. Un segment est donc composé d'un ensemble d'extents qui n'ont pas forcément tous la même taille. (Les clauses `INITIAL` et `NEXT` spécifiées dans l'instruction de création de l'objet associé au segment permettent d'influer sur ces paramètres.) La Figure 1.2 présente la décomposition hiérarchique d'un segment en blocs de données. Il est à noter que, si un segment n'a plus besoin de l'espace qu'il a précédemment alloué, il ne le libère pas nécessairement.

Figure 1.2

Décomposition d'un segment en extents puis en blocs.



MS SQL Server

Sous SQL Server, la logique est relativement proche, sauf qu'un niveau supplémentaire est intégré, le niveau base de données (*database)*. À la différence d'Oracle, une instance SQL Server contient plusieurs bases de données, qui ont chacune leurs espaces logiques nommés *data file group* (équivalents des tablespaces) eux-mêmes composés de *data file*.

MySQL

Sous MySQL, l'organisation du stockage est déléguée au moteur de stockage. Plusieurs moteurs sont supportés, MyISAM et InnoDB le plus fréquemment.

Le moteur MyISAM attribue plusieurs fichiers à chaque table.

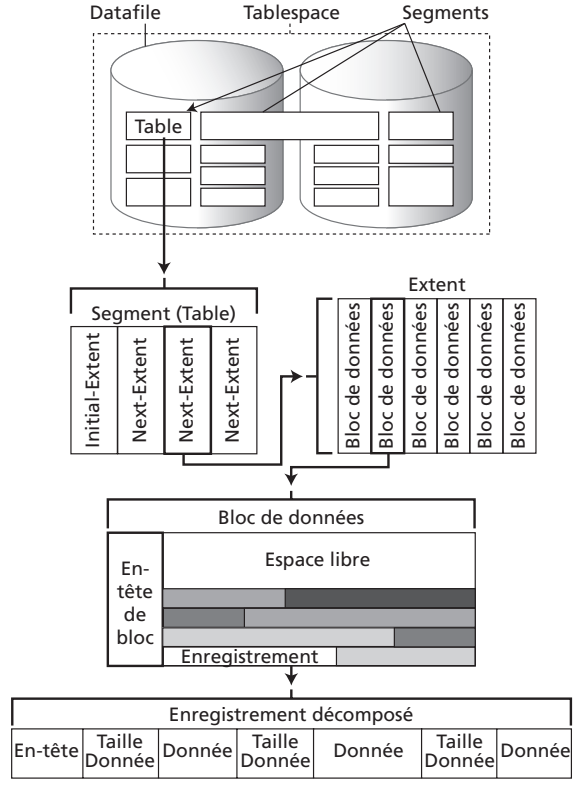
- Un fichier `.frm` pour décrire la structure de la table. Ce fichier est géré par MySQL et non pas par le moteur de stockage. Un fichier `.myd` qui contient les données.
- Un fichier `.myi` qui contient les index.

Le moteur InnoDB implémente le concept de tablespace qui contient tous les éléments de la base de données (tables, index, etc.). Les fichiers `.frm` gérés par MySQL sont présents en plus du tablespace.

Les blocs de données contiennent les enregistrements (lignes ou *row*) et des structures de contrôle. La Figure 1.3 montre un schéma complet de l'organisation du stockage des données.

Figure 1.3

Schéma illustrant le stockage des données dans une base Oracle.



1.2.2 Le RowID

Le RowID (identifiant de ligne) est une information permettant d'adresser directement un enregistrement, sans avoir à parcourir aucune liste. C'est une sorte de pointeur. Il est composé de :

- l'identifiant de l'objet ;
- l'identifiant du datafile ;
- l'identifiant du bloc dans le datafile ;
- l'identifiant de la ligne dans le bloc.

Cette information sera rarement manipulée directement par vos applications. Elle est surtout destinée à un usage interne au SGBDR. Cependant, dans certains cas, il peut être intéressant d'y recourir depuis une application, par exemple pour désigner un enregistrement n'ayant pas de clé primaire ou pour adresser plus rapidement un enregistrement pour lequel vous avez récupéré le RowID. Le RowID d'un enregistrement s'obtient en interrogeant la pseudo-colonne `rowid`.

```
SQL> select nocmd, noclient, datecommande, rowid from cmd;
     NOCMD   NOCLIENT DATECOMMANDE ROWID
---------- ---------- ------------ ------------------
     14524     106141 16/04/2004   AAARnUAAGAAASIfAAG
     14525     131869 16/04/2004   AAARnUAAGAAASIfAAH
     14526      83068 16/04/2004   AAARnUAAGAAASIfAAI
     14527     120877 16/04/2004   AAARnUAAGAAASIfAAJ
     14528      34288 16/04/2004   AAARnUAAGAAASIfAAK
     14529     103897 16/04/2004   AAARnUAAGAAASIfAAL
6 rows selected
```

Voir l'Annexe A, section A.1, pour plus d'informations sur les RowID.

1.2.3 Online Redo Log et Archived Redo Log

Les fichiers Online Redo Log sont des fichiers binaires qui contiennent toutes les modifications appliquées aux datafiles récemment. Ils permettent notamment de réparer les datafiles en cas d'arrêt brutal de la base.

Les fichiers Archived Redo Log sont des fichiers Redo Log archivés. Ils sont créés uniquement si le mode *archivelog* est actif. Ils permettent de réparer les datafiles, de compléter une restauration, de maintenir une base de secours en attente (*stand by database*) et sont aussi nécessaires pour certaines fonctions Oracle Warehouse.

Leur gestion a un coût en termes d'espace disque utilisé, donc si vous n'en avez pas besoin, désactivez cette fonction.

MS SQL Server

Les fichiers LDF, qui sont les journaux de transactions, jouent un rôle analogue aux Online Redo Log.

MySQL

Le moteur InnoDB intègre un mécanisme analogue aux Online Redo Log. Le moteur MyISAM n'a pas d'équivalent.

1.2.4 Organisation des tables

Les données sont organisées dans des tables à deux dimensions : les *colonnes*, ou champs, qui ne sont pas supposées évoluer différemment du modèle de données, et les *lignes*, enregistrements qui varient au fil du temps quand des données sont ajoutées ou modifiées. Par défaut, dans de nombreux SGBDR, la structure de stockage adoptée est une structure dite de "tas" (*heap*).

Ce type de structure stocke les enregistrements en vrac, sans ordre particulier, dans une zone de données. Quand un enregistrement est détruit, il libère de l'espace qu'un autre enregistrement pourra éventuellement réutiliser.

Sous Oracle, en mode MSSM (*Manual Segment Space Management*), les blocs contenant de l'espace libre, c'est-à-dire dont le pourcentage d'espace libre est supérieur au paramètre `PCTFREE` de la table, sont listés dans une zone de la table nommée FREELIST. Les blocs dans lesquels de l'espace a été libéré à la suite de suppressions d'enregistrements et dont le pourcentage d'espace utilisé repasse en dessous de la valeur du paramètre `PCTUSED` sont, eux aussi, répertoriés dans la FREELIST.

L'organisation des tables sous forme d'index est assez commune, les enregistrements sont stockés dans l'ordre de la clé primaire. On dit alors que la table est organisée en index (IOT, *Index Organized Table*). Nous étudierons ce type de table au Chapitre 5, section 5.3.2, "Index Organized Table".

MS SQL Server

Les tables sans index clustered sont organisées en tas, celles ayant un index clustered ont une organisation de type IOT.

MySQL

Sous MySQL, les tables utilisant le moteur MyISAM sont organisées en tas, celles utilisant le moteur InnoDB ont une organisation de type IOT.

1.2.5 *Row Migration* et *Row Chaining*

La migration d'enregistrement (*Row Migration*) est le mécanisme qui déplace un enregistrement qui ne tient plus dans son bloc, après une mise à jour de ses données ayant entraîné un accroissement de sa taille. Lors de la migration, le SGBDR insère un pointeur à l'emplacement initial de l'enregistrement qui désigne le nouvel emplacement que le SGBDR alloue pour y placer les données. Ainsi, on pourra toujours accéder à l'enregistrement en utilisant le RowID initial qui reste valide. Cependant, il faut noter que l'utilisation de ce pointeur sera source d'E/S (entrées/sorties) supplémentaires. En effet, l'accès à un enregistrement au moyen de son RowID se fait habituellement en *une* lecture de bloc, alors que celui à un enregistrement ayant migré nécessite de lire *deux* blocs (celui qui est pointé par le RowID plus celui qui est désigné par le pointeur). Afin de limiter l'apparition de ce phénomène, Oracle n'insère des lignes dans un bloc que si un certain pourcentage d'espace libre demeure à l'issue de cette opération (paramètre `PCTFREE` de la table ; par défaut, il vaut 10 %).

Le chaînage d'enregistrement (*Row Chaining)* est le mécanisme qui gère le fait qu'un enregistrement ne peut pas tenir dans un unique bloc de données (8 Ko par défaut) car sa taille est supérieure. Le processus va scinder les données et les répartir dans plusieurs blocs en plaçant dans chaque bloc un pointeur vers les données situées dans le bloc suivant. Comme lors de la migration d'enregistrement, ce mécanisme va causer des E/S supplémentaires.

Voir l'Annexe A, section A.2, pour plus d'informations sur ces mécanismes.

1.2.6 Le cache mémoire

Les accès disque sont très lents comparés aux accès en mémoire vive (RAM). Approximativement et en simplifiant : un disque dur a un temps d'accès qui se compte en millisecondes, alors que celui de la mémoire se compte en nanosecondes. La mémoire vive est donc un million de fois plus rapide que les disques durs.

Par ailleurs, la quantité de mémoire vive sur les serveurs étant de plus en plus importante, une optimisation assez évidente consiste à implémenter un système de cache mémoire dans les SGBDR.

Le but du cache mémoire est de conserver en mémoire une copie des informations les plus utilisées afin de réduire les accès au disque dur. Sous Oracle, ces caches mémoire se trouvent dans SGA (*System Global Area*), une zone mémoire principale, qui englobe diverses zones, parmi lesquelles :

- **Database Buffer Cache.** Contient les blocs manipulés le plus récemment.
- **Shared Pool.** Contient les requêtes récentes ainsi que le plan d'exécution qui leur est associé. Cette zone contient aussi un cache du dictionnaire des données.

Le buffer cache contient une copie mémoire des blocs les plus manipulés afin de réduire le nombre d'accès disque. Cela aura pour effet d'améliorer notablement les performances. Dans de nombreux cas, la quantité de mémoire disponible pour le buffer cache sera plus faible que celle des données manipulées. De fait, il faudra choisir quelles sont les données à conserver dans le cache et quelles sont celles qui doivent céder leur place à de nouvelles données. Ce choix sera fait par un algorithme de type MRU (*Most Recently Used*) qui privilégiera le maintien en mémoire des blocs les plus utilisés récemment. Afin d'éviter que le parcours d'une grosse table entraîne un renouvellement complet du cache, l'accès à ce type de tables est pondéré défavorablement au profit des petites tables, pondérées, elles, positivement. Ces dernières resteront plus longtemps dans le cache, alors que, généralement, les grosses tables n'y seront pas chargées entièrement.

MS SQL Server

Sous SQL Server, on retrouve un mécanisme tout à fait similaire.