# Les bases de données

Hervé Lehning

Agrégé de mathématiques, écrivain scientifique,
membre de l'ARCSI et commandant de réserve

Nous assistons de nos jours à un déluge de données venant des réseaux sociaux, des sites qu'on visite comme des objets connectés qu'on utilise. En France, cette collecte des données, qui fait le *big data*, est encadrée par la loi. La CNIL (Commission nationale de l'informatique et des libertés) est chargée de la faire appliquer. Les médias nous montrent chaque jour les conséquences dévastatrices occasionnées par des données tombées dans de mauvaises mains.

## Recherche dans une base de données : des méthodes ingénieuses !

Imaginez. Vous entrez dans une bibliothèque pour chercher le livre conseillé par un ami. Il vous a donné son titre et le nom de l'auteur. Vous vous renseignez. On vous envoie à deux fichiers, l'un par thème, l'autre par nom d'auteur. Vous choisissez celui-ci et, très vite, vous trouvez votre auteur, puis la référence du livre que vous cherchez, un numéro de classement dans les rayonnages. Vous les parcourez rapidement et, au numéro dit, trouvez votre livre. Sans le savoir, vous venez de consulter une base de données. Toutes peuvent se représenter comme les livres d'une bibliothèque. De façon générale, on peut l'imaginer comme un tableau dont les lignes représentent les données (les livres de notre bibliothèque) et les colonnes les renseignements les concernant (numéro de classement dans la base, nom de l'auteur, thème, photographie…).

On peut de même se représenter un dictionnaire, le fichier des clients d'une entreprise, celui de ses fournisseurs… Une base de données est donc avant tout un tableau, une liste de fiches de plusieurs natures, quantitatives ou qualitatives. Si elle est « grande », disons qu'elle comporte un million de fiches, sans index, il est difficile d'y trouver ce que l'on cherche. La seule solution serait de parcourir les fiches de la première à la dernière. Si vous avez de la chance, celle que vous cherchez est la première, mais elle peut aussi bien être la dernière. En moyenne, il vous faut consulter la moitié des fiches pour trouver celle que vous cherchez. Une telle recherche est dite

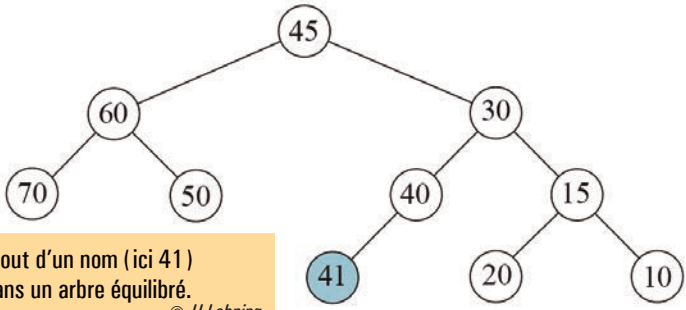
*séquentielle*. Même à l'aide d'un ordinateur, elle n'est praticable que pour de « petites » bases de données car, en moyenne, elle utilise un temps proportionnel au nombre d'éléments de la base.

Pour simplifier la consultation d'une base de données, la première idée est d'utiliser des index classés dans l'ordre suivant un critère, le nom par exemple. Imaginons que nous cherchions le mot « CIJM » parmi un million de fiches classées ainsi. Essayons d'abord celle du milieu ! Si l'on tombe sur le mot cherché, la recherche s'arrête là. Si la fiche porte un nom antérieur à celui recherché, la bonne fiche (si elle existe) se trouve entre les fiches numéro 500 001 et numéro 1 000 0000, sinon entre les numéros 1 et 499 999. On recommence de même. À chaque étape, la longueur de l'intervalle de recherche est divisée par 2. Dans le pire des cas, la consultation sera terminée après vingt essais puisque $2^{20} = 1\,048\,576$.

## Mise à jour d'une base : les atouts d'une structure arborescente

Pour une base qui n'évolue pas (ou peu), tout ceci est parfait. Mais que faire quand on veut ajouter, modifier ou supprimer un élément ? Prenons le cas d'un ajout. Aucune difficulté pour la base elle-même : on place la nouvelle fiche à la suite des autres. Pour les index, c'est plus difficile. Si vous voulez insérer une nouvelle fiche, il faut la placer au bon endroit. Le trouver est facile, il suffit de le chercher comme précédemment. Mais, pour l'insérer, il faut décaler ceux qui le suivent pour lui faire une place. En moyenne, le temps d'insertion est proportionnel à la taille de la base. Il en est de même pour une suppression. Seule une modification ne prend que le temps de la recherche.

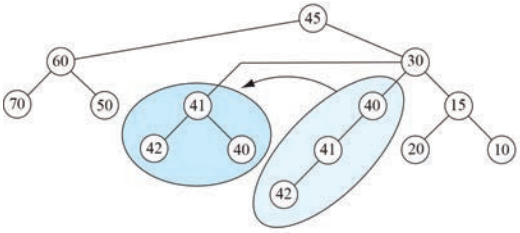
Pour simplifier la gestion des bases de données, une solution est l'utilisation d'une *structure d'arbre*, analogue à celle des arbres généalogiques. Plus précisément, elle est composée de *nœuds* portant le critère de recherche en *étiquette*, chaque nœud ayant un *père* (sauf le premier, la racine) et deux *fils* (sauf les derniers, les *feuilles*). Pour chaque nœud, l'étiquette du fils droit est antérieure et celle du fils gauche, postérieure à celle de son père. De plus, tout le long de l'arbre, les hauteurs des fils d'un même père diffèrent d'au plus une unité. Un tel arbre est appelé un *arbre de recherche équilibré*. Imaginons que l'index d'une base de données soit structuré ainsi. Pour y chercher un nom, partons de la racine. Si ce nom est antérieur à la racine, il se trouve (s'il existe effectivement) dans son fils droit, sinon dans son fils gauche. La recherche se poursuit ainsi en parcourant les branches de l'arbre. En moyenne, son temps d'exécution est proportionnel à la hauteur de l'arbre, et non pas à son nombre d'éléments.




Ajout d'un nom (ici 41)
dans un arbre équilibré.
© *H.Lehning*

Les éléments d'un arbre équilibré peuvent être des nombres ou des objets de toute nature, du moment qu'il est possible de les comparer deux à deux. Pour chercher l'élément 40, on le compare à la racine (45). Il est strictement inférieur donc, s'il figure effectivement dans l'arbre, il appartient à son fils droit. On le compare alors à la racine du fils droit (30). Il est strictement supérieur, donc appartient à son fils gauche. Et ainsi de suite. Si maintenant on souhaite ajouter l'élément 41, on opère de même (en bleu sur la figure). Il se trouve qu'ici l'arbre reste équilibré. Ce n'est pas forcément le cas.

Comment ajouter un élément dans un tel index ? Si l'index est vide, aucune difficulté. La belle affaire, direz-vous. Méfiez-vous, l'idée est plus subtile qu'il n'y paraît ! Si l'arbre n'est pas vide, il a une racine. Si l'élément à ajouter est antérieur à cette racine, il convient de l'ajouter à son fils droit, sinon à son fils gauche. Et on recommence avec ce fils. Cette seule idée suffit pour insérer le nouvel élément. Un tel algorithme est dit *récursif*. Quel est son temps d'exécution ? Le même que celui d'une recherche ! Il est proportionnel à la hauteur de l'arbre. Bien entendu, il reste un problème délicat : ce nouvel arbre n'est plus forcément équilibré. Il faut le rééquilibrer. Ceci se fait par des « rotations » des parties déséquilibrées. Cet algorithme d'ajout d'un élément dans la base clôt la question : on a décrit les deux opérations essentielles liées aux bases de données (la consultation et la mise à jour).




Rééquilibrage d'un arbre.
L'arbre se déséquilibre lorsque l'on ajoute
les deux éléments 41 et 42. Pour le rééquilibrer,
il suffit d'opérer une rotation sur son élément médian (41).
La partie bleu clair est donc remplacée par la partie bleu foncé.
© *H.Lehning*

## Le partitionnement de documents : classer, résumer, affecter…

Retournons dans notre bibliothèque. Elle est partitionnée en plusieurs rayons : romans, livres pratiques, sciences… De même, on peut vouloir partitionner un lot de photos en paysages, bâtiments, animaux, humains… Si la taille de la collection de photos est grande, il est important de le faire automatiquement, de trouver un algorithme réalisant l'opération pour nous. Pour cela, on définit une distance euclidienne sur l'espace des photos, ce que l'on peut faire en comparant les pixels des images. Pour d'autres types de documents, la première étape du partitionnement est le choix de cette distance euclidienne, du nombre *k* de classes et d'un représentant de chaque classe, donc ici d'une photo de paysage, d'une photo de bâtiment…
La première étape de l'algorithme est d'attribuer une classe à chaque document : celle dont le représentant est « le plus proche ». On recommence alors en remplaçant le représentant de chaque classe par le centre de gravité (la moyenne) de la classe… et on recommence ainsi jusqu'à obtenir un partitionnement stable. Cette procédure est appelée *algorithme des k-moyennes*. Elle est utile pour analyser des données sophistiquées en les résumant à *k* exemples. Elle permet aussi à une équipe de ventes de définir des cibles pour leurs campagnes de marketing.

Les nuages (ou *clouds*) sont indispensables aux grandes bases de données. Leurs avantages sont nombreux. Plus besoin de disposer de tous les logiciels sur son ordinateur, il suffit d'utiliser ceux se trouvant dans le nuage qui, de plus, sont toujours à jour. De même, vous disposez de toute la mémoire nécessaire pour vos besoins quotidiens. Bien des gens le font en déposant des vidéos sur des sites comme YouTube (Google, 2006), par exemple.

Malgré ces avantages, l'*informatique en nuage* (ou *cloud computing*) possède deux gros inconvénients. Le premier est de dépendre d'Internet. En cas de coupure de réseau, vous n'avez plus accès à ce que vous avez externalisé. Le second est plus grave, il s'agit du manque de confidentialité. Vous ne savez plus où transitent vos données. Pourquoi pas chez vos pires ennemis ? Pour le comprendre, il est bon de savoir où elles se trouvent physiquement. Dans quel pays, sous quelles menaces (légales ou illégales) ? En fait, un nuage correspond à un certain nombre de centres de données (*data centers* en anglais), dont il est bon de savoir où, et sous quelle législation, ils se trouvent. Ils sont grands consommateurs d'énergie et grands producteurs de chaleur, ce qui milite pour les placer dans des endroits froids, proches d'agglomérations dont ils peuvent fournir une partie du chauffage urbain.

Un centre de données, élément d'un nuage. Où se trouve-t-il ?

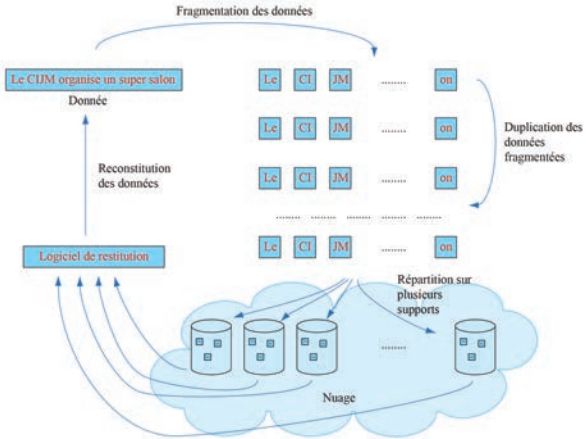
© *Más grande del mundo, 2018*

On parle de « nuages » pour signifier que l'on ne sait pas bien où sont situées les données que l'on y place car elles sont en fait fragmentées entre plusieurs centres de données qui, contrairement au *cloud*, sont des lieux physiques.

Plus précisément : un *cloud* est toujours un espace virtuel ; les données contenues dans un cloud sont fragmentées ; les fragments de données sont toujours dupliqués et répartis sur un ou plusieurs supports physiques ; un *cloud* possède une fonction de restitution des données permettant de reconstituer les données qui ont été fragmentées.

Fonctionnement d'un nuage.

© *H.Lehning*




## Le chiffrement des données : la clé de la confidentialité

Si on ignore les algorithmes utilisés, il est en principe impossible de savoir où se trouve l'information, et on ne peut avoir accès qu'à des fragments qui ne permettent pas de retrouver le sens des données. Ces impossibilités

sont cependant fondées sur le secret de la méthode utilisée, ce qui est une grave faiblesse face à l'espionnage ou à l'intelligence économique. Pour garantir une véritable confidentialité, il est prudent de chiffrer les données.

Il existe un système permettant d'assurer une confidentialité correcte, à base d'un chiffre symétrique dont la clef est transmise par un chiffre asymétrique. Ce procédé de chiffrement est en principe solide, mais il n'est pas utilisable sur les «petits» objets connectés, en particulier ceux que l'on implante dans le corps humain comme les pacemakers ou les pompes à insuline. L'utilisation d'un chiffre asymétrique, très gourmand en énergie, ferait chauffer l'objet, ce qui serait source d'insécurité ! Il en résulte que les «petits» objets connectés sont difficiles à protéger. Ce sont donc des cibles faciles pour les pirates, pour se constituer des réseaux de zombies (ou *botnets*), c'est-à-dire d'ordinateurs dont ils se sont rendus maîtres pour lancer des actions malveillantes, telles des opérations de hameçonnage (le tristement fameux *phishing*). Cela peut sembler de la science-fiction. Pourtant, cela s'est déjà produit ! Les objets connectés se comptant par dizaines de milliards, le marché est immense… pour des personnes mal intentionnées.

***H.L.***

## Les systèmes de chiffrement

Il existe deux grands systèmes de chiffrement. Premièrement, les *systèmes symétriques*, pour lesquels savoir chiffrer implique savoir déchiffrer. La clef doit être tenue secrète. Le chiffrement est rapide, le temps étant de l'ordre de celui d'une addition. Le plus connu est le système AES (*advanced encryption standard*, 1997).

Deuxièmement, les *systèmes asymétriques*, pour lesquels savoir chiffrer n'implique pas de savoir déchiffrer. La clef de chiffrement est publique, celle de déchiffrement est secrète. Le plus connu est le système RSA (Rivest, Shamir, Adleman, 1977), où la clef publique est liée au produit $N = p \times q$ de deux «grands» nombres premiers $p$ et $q$ (idéalement de plus de deux cents chiffres), et la clef secrète est liée à $p$ et $q$. La difficulté de la factorisation de N si on ne connaît ni $p$, ni $q$ explique ce mystère. Ce chiffrement est lent et gourmand en énergie, le temps étant de l'ordre d'une exponentiation.

### Pour en savoir (un peu) plus :

***Toutes les mathématiques du monde***. Hervé Lehning, Flammarion, 2017.

***La bible des codes secrets***. Hervé Lehning, Flammarion, 2019.