CHAPITREI:

Introduction aux bases de données

1. Introduction

L'informatique évolue vers le traitement de masses d'informations de plus en plus grandes dans des environnements répartis géographiquement où doivent cohabiter des matériels hétérogènes. Dans ce contexte, les bases de données sont utilisées de façon intensive pour de nombreux domaines d'application tels que le domaine médical, les administrations ou les associations. Les applications concernées par l'utilisation d'un SGBD (Système de gestion de base de données) possèdent des caractéristiques différentes tant au niveau du volume de données concernées qu'au niveau de la complexité de ces données et des traitements informatiques à réaliser. Néanmoins, le regroupement des données dans une base de données gérée par un système de gestion de base de données apporte de nombreux avantages dans la plupart des cas d'utilisation.

Le domaine informatique bien qu'étant jeune, a une évolution croisière. Jadis, la gestion et le traitement des données se faisaient par la méthode classique à laquelle l'on a pu dégager ces défauts suivants:

- La redondance de données ;
- La dépendance pleine entre données et traitement ;
- Le manque de normalisation au niveau de stockage de données.

Pour remédier à cette situation, il a été mis au point la notion de base de données répondant aux questions suivantes:

- L'accès aux données selon les multiples critères ;
- L'intégration des données ;
- La relation entre les données.

La notion qui remplace avantageusement celle de fichiers.

- L'ordre dans le stockage de données ;
- L'utilisation simultanée des données par différents utilisateurs.

2. Qu'est-ce qu'une base de données ?

Le concept de Base de Données (BDD) est apparu vers 1960, face au nombre croissant d'informations que les entreprises devaient gérer et partager :

- Base de données - Un ensemble organisé d'informations avec un objectif commun. Plus précisément, on appelle base de données un ensemble structuré et organisé permettant le stockage de grandes quantités d'informations afin d'en faciliter l'exploitation (ajout, mise à jour, recherche et consultions de données).

- Base de données informatisée - Une base de données informatisée est un ensemble structuré de données enregistrées sur des supports accessibles par l'ordinateur, représentant des informations du monde réel et pouvant être interrogées et mises à jour par une communauté d'utilisateurs.

La gestion et l'accès à une base de données sont assurés par un ensemble de programme que constitue le système de gestion de base de données (SGBD).

Ainsi la notion de base de données est généralement couplée à celle des réseaux informatiques afin de pouvoir mettre en commun les informations d'où le nom de « base ». On parle souvent de système d'information pour désigner toute structure regroupant les moyens mis en place pour partager les données. **[1]**

3. Critères d'une base de données

Une base de données doit répondre aux trois critères suivants :

- L'exhaustivité : C'est la présence dans cette base de tous les enseignements qui ont trait aux applications en question.
- Le non redondance des données : Non répétition d'une donnée plusieurs fois.
- La structure : C'est l'adaptation du mode de stockage de données au traitement ; structuration que la base doit avoir est liée à l'évolution de la technologie. **[1]**

4. Système de Gestion de Base de données : SGBD

Ensemble des programmes et des langages de commande qui permettent de :

- Définir des "bases de données", et des relations entre les éléments de chaque base ;
- Spécifier le traitement de ces données : interrogations, mises à jour, calculs, extractions...

Le SGBD reçoit des commandes aussi bien des programmes d'application que des utilisateurs : il commande les manipulations de données, généralement par l'intermédiaire d'un SGF. **[3]**

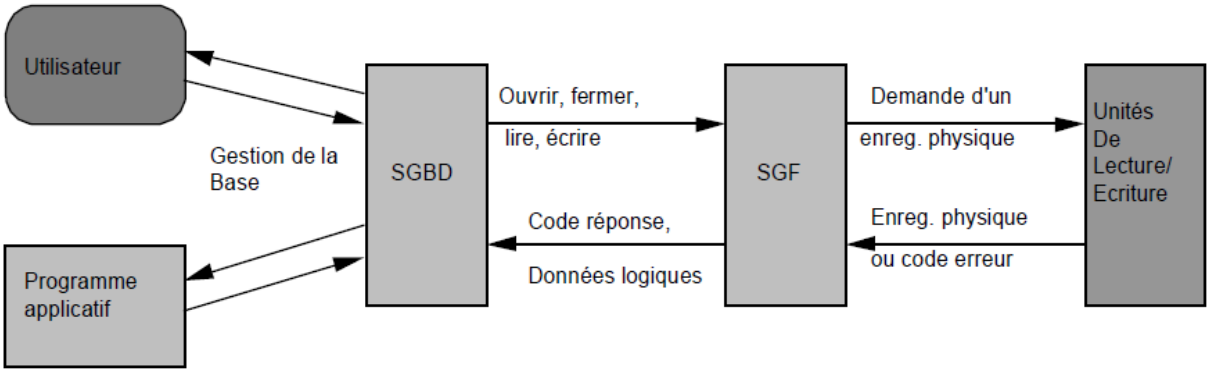

Figure I .1: Système de Gestion de Base de données « SGBD ».

Les définitions sont cependant de peu d'intérêt pour déterminer si un système est vraiment un SGBD ou s'il s'agit simplement d'un système d'information classique ou d'un système de fichiers. Il faut mieux définir le SGBD en précisant certaines des fonctions qu'il doit remplir :

- L'intégration des données afin d'éviter l'incohérence d'éventuelles données dupliquées (tout est intégré dans un seul ensemble cohérent).
- La séparation entre les moyens de stockage physique des données et la logique des applications.
- Un contrôle unique de toutes les données afin de permettre l'utilisation simultanée par plusieurs utilisateurs.
- La possibilité d'utiliser des structures de fichiers et des méthodes d'accès complexes, de façon à ce que les relations correctes entre les données puissent être exprimées et les données utilisées le plus efficacement dans un grand nombre d'applications.
- Des facilités pour le stockage, la modification, la réorganisation, l'analyse

et la consultation des données, sans que le système impose des restrictions à l'utilisateur.

- Des contrôles de sécurité afin d'empêcher l'accès illégal à certaines données.
- Des contrôles d'intégrité pour prévenir une modification indue des données (exemple: contrôle d'exactitude, de validité).
- La compatibilité avec les principaux langages de programmation, les programmes-sources existants, et les données extérieures à la base.
- Posséder une capacité de stockage élevée.
- Pouvoir répondre à des requêtes avec un niveau de performances adapté.
- Fournir des facilités pour la gestion des méta-données. **[4]**

4.1 Objectifs de l'approche SGBD

• Pour pallier aux inconvénients des méthodes classiques de gestion de fichiers, les SGBD visent quatre objectifs : intégration et corrélation, flexibilité (indépendance), disponibilité, sécurité.

• Ces objectifs exigent une distinction nette entre les données et les procédures de manipulation de ces données : aux données, on associera une fonction **d'administration des données**, aux procédures de manipulation une **fonction de programmation**.

4.1.1 Intégration et corrélation

Dans les systèmes classiques, chaque application gère ses données dans ses propres "fichiers", d'où :

❖ Un risque de redondance, et un danger d'incohérence des données.

- La même donnée peut appartenir à plusieurs applications, induisant une déperdition de stockage.
- Toute modification de cette donnée est à enregistrer plusieurs fois : si cette mise à jour multiple n'est pas effectuée correctement, les données deviennent incohérentes.
- Le coût de la mise à jour augmente du fait de la multiplication des entrées-sorties physiques.

❖ Une difficulté pour créer de nouveaux traitements

- Les nouvelles applications entraînent des duplications supplémentaires de données.
- Leur intégration avec les applicatifs en exploitation entraîne des modifications importantes. Dans l'approche SGBD, un "réservoir" commun (**intégration**) est constitué, représentant une modélisation (**corrélation**) aussi fidèle que possible de l'organisation réelle de l'entreprise :
  - ❖ Toutes les applications puisent dans ce réservoir, les données qui les concernent, évitant ainsi les duplications.
  - ❖ Mais le partage des données entre les utilisateurs pose le problème de la synchronisation des accès concurrents.

4.1.2 Flexibilité ou indépendance

- Dans les systèmes classiques, tout changement intervenant dans le stockage des données (support, méthode d'accès physique) entraîne des modifications lourdes des applications correspondantes.
- L'approche SGBD poursuit trois objectifs, pour assurer l'indépendance des données par rapport aux traitements :
  - ❖ Indépendance physique: tout changement de support, de méthode d'accès reste transparent au niveau de l'utilisateur.
  - ❖ Indépendance logique : les programmes d'application sont rendus transparents à une modification dans l'organisation logique globale, par la définition de sous-schémas couvrant les besoins spécifiques en données.
  - ❖ Indépendance vis-à-vis des stratégies d'accès : l'utilisateur n'a plus à prendre en charge l'écriture des procédures d'accès aux données. Il n'a donc pas à intégrer les modifications tendant à optimiser les chemins d'accès (ex: création d'index).

4.1.3 Disponibilité

- ❖ Le choix d'une approche SGBD ne doit pas se traduire par des temps de traitement plus longs que ceux des systèmes antérieurs.
- ❖ L'utilisateur doit ignorer l'existence d'utilisateurs concurrents.
- ❖ L'aspect "performance" est donc crucial dans la mise en œuvre d'une base de données. Un tel objectif ne peut être atteint que si la conception

d'une base de données est menée de façon rigoureuse avec un découpage fonctionnel adéquat. Les règles et contraintes inhérentes sont évoquées lors de l'apprentissage d'une méthodologie d'analyse (exemple MERISE).

4.1.4 Sécurité

La sécurité des données recouvre deux aspects :

- **L'intégrité**, ou protection contre l'accès invalide (erreurs ou pannes), et contre l'incohérence des données vis-à-vis des contraintes de l'entreprise.
- **La confidentialité**, ou protection contre l'accès non autorisé ou la modification illégale des données.

Pour ne pas trop affecter les performances, la sécurité doit également être prise en compte dès la phase de conception. **[3]**

4.2 Architecture de SGBD

4.2.1 Architecture Client-Serveur

Depuis les années 80, les SGBD sont basés sur une **architecture clients-serveur.**

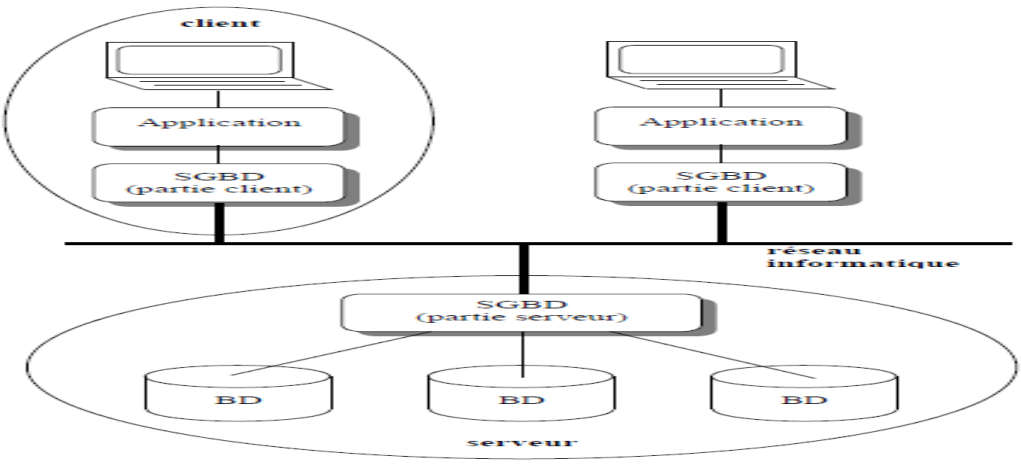

Figure I .2: Architecture Client-serveur. [2]

Serveur : On appelle logiciel serveur un programme qui **offre un service** sur le réseau. Le serveur accepte des requêtes, les traite et renvoie le résultat au demandeur. Le terme serveur s'applique à la machine sur lequel s'exécute le logiciel serveur. (Gère les données partagées et exécute le code du SGBD).

Clients : On appelle logiciel client un programme qui **utilise le service** offert par un serveur. Le client Communiquent avec le serveur envoie une requête et reçoit la réponse. Le client peut-être raccordé par une liaison temporaire.

Qu'appelle-t-on architecture client/serveur ?

C'est la description du **fonctionnement coopératif** entre le serveur et le client. Les services internet sont conçus selon cette architecture. Ainsi, chaque application est composée de logiciel serveur et logiciel client. A un logiciel serveur, peut correspondre plusieurs logiciels clients développés dans différents environnements: Unix, Mac, PC...; la seule obligation est le respect du protocole entre les deux processus communicants. Ce protocole étant décrit dans un RFC (Request For Comment). **[9]**

4.2.2 Architecture centralisée

Ce type d'architecture est appelée solution sur site central **(***Mainframe*). Historiquement, les applications sur site central ont été les premières à proposer un accès multiutilisateurs. Dans ce contexte, les utilisateurs se connectent aux applications exécutées par le serveur central à l'aide des terminaux se comportant en esclaves. C'est le serveur central qui prend en charge l'intégralité des traitements y compris l'affichage qui est simplement déporté sur des terminaux. **[1]**

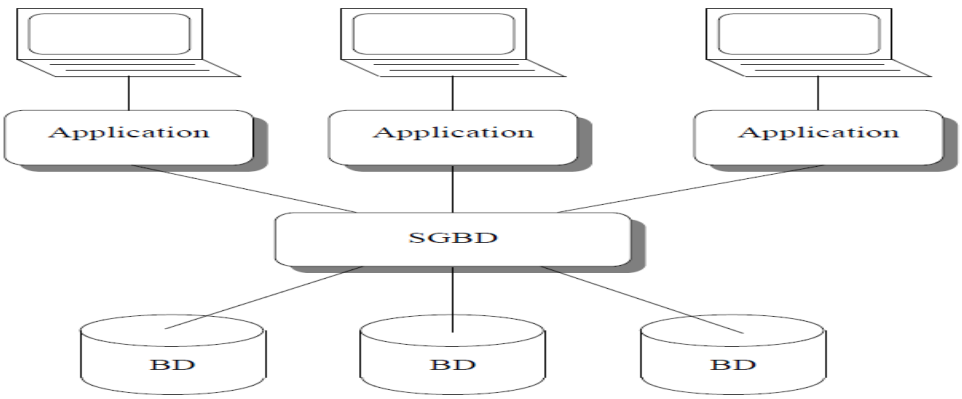

Figure I .3: Architecture centralisée. [2]

4.3 Architecture des systèmes

4.3.1 Architecture trischématique ou architecture ANSI/SPARC

Dans l'architecture trischématique on établit quatre niveaux de description du système de base de données qui sont : le *niveau interne*, le *niveau conceptuel*, *le niveau externe et le niveau physique*.

Le processus de transformation des requêtes et des résultats qui sortent d'un niveau à un autre s'appelle correspondance ou mapping.

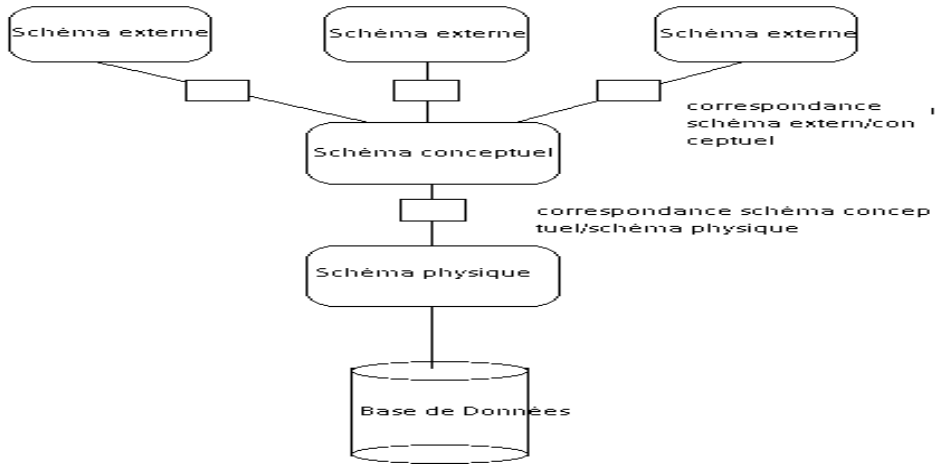

Figure I .4 : Architecture fonctionnelle d'un SGBD : ANSI-SPARC

Le niveau externe :

Le niveau externe (appelé aussi niveau vue), comprend une quantité de vues utilisateurs ; chaque utilisateur décrit une partie de la base qui convient à ses besoins. Chaque groupe d'utilisateurs s'intéresse uniquement à son propre schéma externe et les SGBD doivent transformer toute demande d'utilisateur de haut niveau en requêtes de schéma conceptuel puis en requêtes de schéma interne appliquées aux données stockées.

Le niveau conceptuel :

Dans le niveau conceptuel on décrit la structure générale de la base de données du point de vu de la communauté des utilisateurs ; c'est un schéma conceptuel qui masque les détails des structures de stockage physique des données et qui ne se soucie pas de l'implémentation physique des données ni de la façon dont chaque groupe d'utilisateurs voudra se servir de la base de données ; ce niveau se concentre sur la description des entités, du type des données, des relations existant entre les entités et des opérations des utilisateurs.

Le niveau interne :

Le niveau interne est un schéma qui décrit la structure de stockage physique de la base de données. Il s'appuie sur un système de gestion de fichiers pour définir la politique de stockage ainsi que le placement des données.

Le niveau physique :

Est donc responsable du choix de l'organisation physique des fichiers ainsi que de l'utilisation de méthodes d'accès en fonction de la requête.

4.3.2 Indépendances des données

Indépendance logique.

L'architecture définit ci-dessus permet de garantir l'indépendance des données par rapport aux programmes ; elle permet de modifier le schéma de la base de données à un niveau sans restructurer les autres. L'indépendance logique des données est la possibilité qui fait qu'on puisse modifier le niveau conceptuel sans remettre en cause les schémas externes ou les programmes d'application. L'ajout ou le retrait de nouveaux objets ne modifient pas les éléments qui n'y font pas explicitement référence.

Indépendance physique

Quand on peut changer le schéma physique et qu'on peut modifier l'organisation physique des fichiers, rajouter ou supprimer des méthodes d'accès sans remettre en cause le schéma conceptuel, alors on a une indépendance physique de la BD.

Le but de ce niveau d'indépendance est de rendre transparente la gestion physique des données aux programmes d'application. **[5]**

Cette architecture logique permet donc d'identifier la logique de structuration d'un système de gestion de bases de données. D'un point de vue fonctionnel, il est possible d'identifier plusieurs composants que l'on retrouve dans la plupart des SGBD. La figure **IV.5** illustre cette composition.

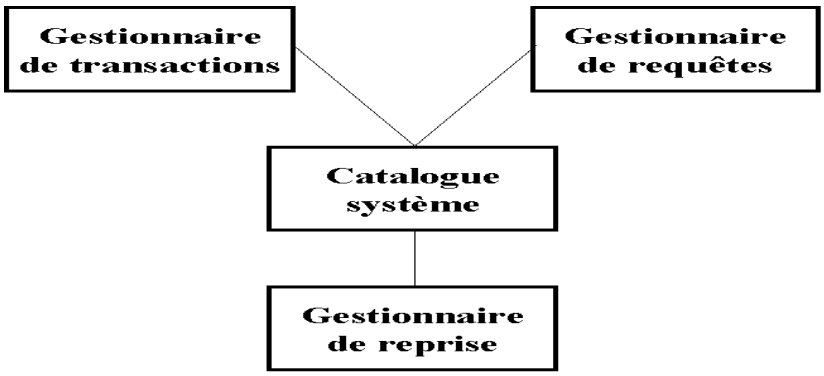

Figure I .5 : Composants d'un SGBD.

4.3.3 Le catalogue système ou dictionnaire

De données est un composant au cœur de la communication entre les autres composants. Il contient toutes les méta-données utiles au système. Ces méta-données correspondent à la description des données (type, taille, valeurs autorisées, etc.), aux autorisations d'accès, aux vues et autres éléments systèmes. Le catalogue correspond à la réalisation de l'architecture à trois niveaux décrite précédemment. Le catalogue contient la description des différents schémas (physique, conceptuel et externes) ainsi que les règles de passage d'un schéma vers l'autre.

4.3.4 Le gestionnaire de requêtes

Est responsable du traitement des commandes des utilisateurs visant à stocker, rechercher et mettre à jour des données dans la base de données. En utilisant les informations stockées dans le catalogue, il interprète ces requêtes et les traduit en des requêtes d'accès physique aux données susceptibles d'être traitées par le système de gestion de fichiers.

4.3.5 Le gestionnaire de transactions

Est responsable de traiter les transactions. Une transaction est un ensemble d'opérations d'accès et de mise à jour de données émises par un utilisateur. Ces opérations doivent être traitées comme un tout et le gestionnaire de transaction doit assurer d'une part l'indivisibilité des opérations soumises, la cohérence du système après l'exécution de l'ensemble des opérations, l'isolation des traitements par rapports aux autres traitements qui peuvent être soumis de façon concurrente et enfin la persistance des résultats une fois l'ensemble des opérations achevées.

4.3.6 Le gestionnaire de reprise

Est un élément essentiel d'un SGBD qui doit remplacer le système de gestion de fichier traditionnel afin de minimiser les effets d'une panne sur la base de données. Un tel système ne peut évidemment pas être sûr ou sécurisé à 100 %. Néanmoins, il est indispensable que le gestionnaire de reprise puisse pallier à certaines pannes qui peuvent avoir différentes causes telle qu'une division par zéro dans un programme, à un problème de disque défectueux ou à une panne d'alimentation. L'objectif essentiel du gestionnaire de reprise est de restaurer la base de données dans un état cohérent. Vue les causes très différentes de panne et les difficultés liées à cette gestion, cet élément est un élément central qui concerne environ 10 % ou plus du code d'un SGBD. **[6]**

4.4 Les opérations sur les données

Il existe 4 opérations classiques (ou *requêtes*) :

1. La *création* (ou *insertion*).

2. La *modification* (ou *mise-à-jour*).

3. La *destruction*.

4. La *recherche*.

Ces opérations correspondent à des commandes du LMD (Langage de Manipulation des Données). La plus complexe est la *recherche* en raison de la variété des critères.

Pour l'utilisateur, une bonne requête a les caractéristiques suivantes. Tout d'abord elle s'exprime facilement : l'idéal serait de pouvoir utiliser le langage naturel, mais celui-ci présente trop d'ambiguïtés. Ensuite le langage ne devrait pas demander d'expertise technique (syntaxe compliquée, structures de données, implantation particulière ...). Il est également souhaitable de ne pas attendre trop longtemps (à charge pour le SGBD de fournir des performances acceptables). Enfin, et peut-être surtout, la réponse doit être fiable.

Une bonne partie du travail sur les SGBD consiste à satisfaire ces besoins. Le résultat est ce que l'on appelle un *langage de requêtes*, et constitue à la fois un sujet majeur d'étude et une caractéristique essentielle de chaque SGBD. Le langage le plus répandu à l'heure actuelle est SQL**. [7]**

4.5 Optimisation

L'optimisation (d'une requête) s'appuie sur *l'organisation physique des données*. Les principaux types d'organisation sont les fichiers séquentiels, les index (denses. secondaires, arbres B) et le regroupement des données par hachage.

Un module particulier du SGBD, *l'optimiseur*, tient compte de cette organisation et des caractéristiques de la requête pour choisir le meilleur séquencement des opérations.

5. Types de base de données

On retrouve fréquemment les bases de données suivantes :

- Référence (qui regroupe des informations bibliographiques ou factuelles).
- Texte intégral (qui regroupe des documents à caractère textuel).
- Multimédia (qui regroupe documents sonores, visuels, etc.).

Une base de données peut se trouver sur n'importe quel support : disque dur, cédérom, sur un serveur et accessible en réseau interne ou en ligne (à distance).

5.1 SGBD réparti ou SGBD distribué

Système gérant une collection de BD logiquement reliées, réparties sur différents sites en fournissant un moyen d'accès rendant la distribution transparente.

Deux approches fondamentales sont à l'origine de la conception des bases de données réparties : la conception descendante '*Top down design*' et la conception ascendante '*Bottom up design*'.

5.1.1 Conception descendante

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.

L'approche top down est intéressante quand on part du néant. Si les BDs existent déjà, la méthode bottom up est utilisée. **[8]**

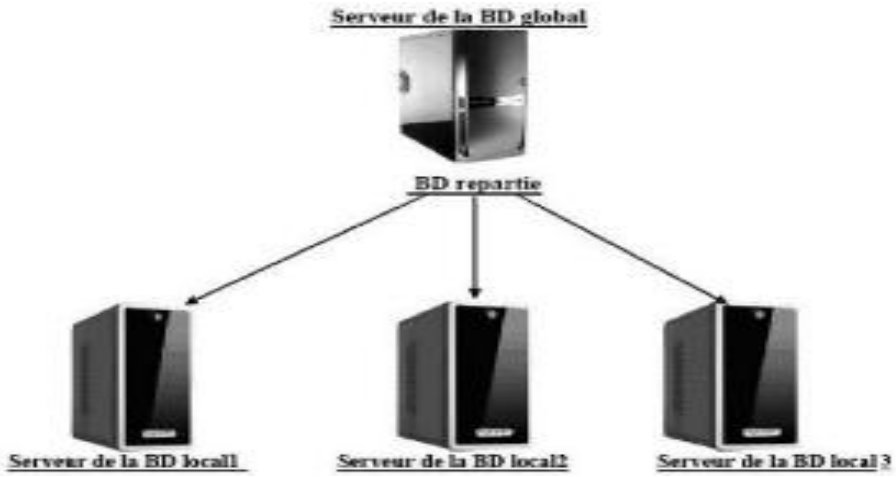

Figure I .6 : Architecture de la conception descendante.

5.1.2 Conception ascendante *(Base de données fédérée):*

Base de données fédérée (BDF) est une BD répartie hétérogène, c'est-à-dire constituée à partir de sources de données de nature variées : fichiers classiques, fichiers de textes, documents HTML, XML, BD relationnelle ou objet, etc. L'objectif est de fournir aux utilisateurs une vue intégrée de différentes données de l'entreprise soit dynamiquement sur demande, soit en la matérialisant périodiquement dans un entrepôt de données. (Plusieurs BD hétérogènes capables d'interopérer *via* une vue commune (modèle commun)).

Cette approche se base sur le fait que la répartition est déjà faite, mais il faut réussir à intégrer les différentes BDs existantes en une seule BD globale. En d'autre terme, les schémas conceptuels locaux existent et il faut réussir à les unifier dans un schéma conceptuel global. **[8]**

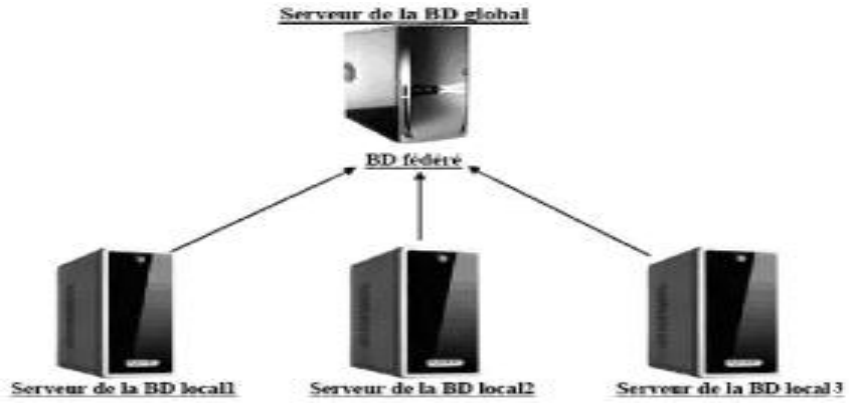

Figure I.7 : Architecture de la conception ascendante.

6. Sécurité et confidentialité d'une base de données

La base de données doit être sécurisée contre :

- Les indiscrétions : Par un mot de passe.
- Les erreurs : Des contrôles doivent être mis en place pour vérifier que des contraintes d'intégrités sont respectées.
- Les destructions : En cas d'incident (panne logicielle, panne matérielle ou panne d'électricité), des procédures de sauvegarde et reprise doivent être prévues afin de relancer le système sans avoir recommencé les saisies par la transaction. **[1]**

7. Conception d'une base de données

La conception d'un système d'information n'est pas évidente car, il faut réfléchir sur l'ensemble de l'organisation, que l'on doit mettre en place. La phase de conception nécessite des méthodes permettant de mettre en place un modèle sur lequel il faut s'appuyer. La modélisation consiste à créer une représentation virtuelle d'une réalité de telle façon à faire ressortir les points auxquels l'on s'intéresse, Elle nécessite une analyse approfondie du monde réel ainsi que des besoins des futurs utilisateurs.

Des méthodes de conception de BDD ont donc été développées : UML, par exemple.

Une fois le schéma conceptuel de haut-niveau établi, il est traduit en termes du modèle conceptuel du SGBD choisi. **[2]**

8. Qui intervient sur une BDD ?

L'**administrateur** (une personne ou une équipe) :

Il définit le schéma conceptuel de la BDD et le fait évolué, comme il fixe les paramètres de l'organisation physique de façon à optimiser les performances, et aussi il gère les droits d'accès et les mécanismes de sécurité.

Les **programmeurs d'applications :**

Ils définissent les schémas externes et construisent les programmes qui alimentent ou exploitent la BDD en vue d'applications particulières. Ils utilisent pour cela le langage de bases de données du SGBD, éventuellement couplé avec un langage de programmation classique.

Les **utilisateurs finals** :

Ils accèdent à la BDD au travers des outils construits par les programmeurs d'application ou pour les plus avertis au travers du langage de requêtes. **[2]**

9. Principaux modèles logiques de SGBD [3]

Les bases de données sont apparues à la fin des années 60, à une époque où la nécessité d'un système de gestion de l'information souple se fait ressentir. Il existe cinq modèle de SGBD, les différenciés selon la représentation des données qu'elle contient :

9.1 Modèle hiérarchique

Le modèle hiérarchique est une forme de système de gestion de base de données qui lie des enregistrements dans une structure arborescente de façon à ce que chaque enregistrement n'ait qu'un seul processeur. Il s'agit du premier modèle de SGBD.

9.1.1 Caractéristiques générales du modèle :

- Forte dépendance entre la description de la structure des données et la manière dont celles-ci sont enregistrées sur le support physique.
- Les éléments de base du modèle sont des enregistrements logiques reliés entre eux pour constituer un arbre ordonné.
- Les entités (ou **segments**) constituent les nœuds, celui de plus haut niveau portant le nom de racine ; les branches (pointeurs logiques entre entités) constituent les **liens**. Chaque segment est une collection d'objets appelés champs (ou **Fields**).
- Chaque segment a obligatoirement un père (sauf la racine), et peut avoir plusieurs fils.

9.1.2 Avantages:

- Rigueur des structures et des chemins d'accès.
- Simplicité relative de l'implémentation.
- Adéquation parfaite du modèle à une entreprise à structure arborescente.

9.1.3 Inconvénients :

- Les accès se font uniquement depuis la racine.
- La structure interdit les liens N:M, ne permettant que le lien 1:N. La représentation d'autres relations impose de ce fait une redondance de l'information.
- Les "anomalies" que l'on constate lors des opérations de mise à jour (insertion, destruction, modification) : l'élimination d'un nœud entraîne l'élimination de tous les segments de niveau inférieur qui lui sont rattachés (risque de perdre des données uniques)
- Indépendance logique très réduite : la structure du schéma doit refléter les besoins des applications.
- Pas d'interface utilisateur simple.

9.2 Modèle réseau

Ce modèle utilise des pointeurs vers des enregistrements.

Evolution du modèle hiérarchique intégrant les résultats du travail du groupe CODASYL (comité de langage de programmation), qui avait démarré l'étude d'une extension de COBOL pour manipuler les bases de données. En 1969, il donne ses premières recommandations concernant syntaxe et sémantique du LDD et du LMD.
Même si cette vue est un peu simplificatrice, une base en réseau peut être décrite comme un certain nombre de fichiers comportant des références les uns vers les autres. Les entités sont connectées entre elles à l'aide de pointeurs logiques :

- Un enregistrement d'un ensemble de données A est associé à une série d'enregistrements (ou records) d'un autre ensemble de données B. On constitue ainsi des SET, ou COSET, structure fondamentale du modèle en réseau.
- Le lien entre les enregistrements de A et ceux de B est 1:N.
- Le COSET comporte un type d'enregistrement "propriétaire" (l'enregistrement de A est dit OWNER) et un type d'enregistrement "membre" (les enregistrements de B sont MEMBER).

9.2.1 Avantages et inconvénients du modèle

- Aucune restriction dans la conception : un type de "record" peut à la fois être propriétaire et membre de plusieurs sets.
- Représentation naturelle des liens maillés N:M.
- Pas d'anomalies pour les opérations de stockage.
- Commercialisation importante des systèmes correspondants (DMS, IDMS, TOTAL, IDS II, SOCRATE...),mais pas d'indépendance par rapport aux stratégies d'accès.
- Procéduralité importante des langages de manipulation ; l'utilisateur doit "naviguer" dans le réseau logique constitué par les enregistrements et les chaînes de pointeurs.

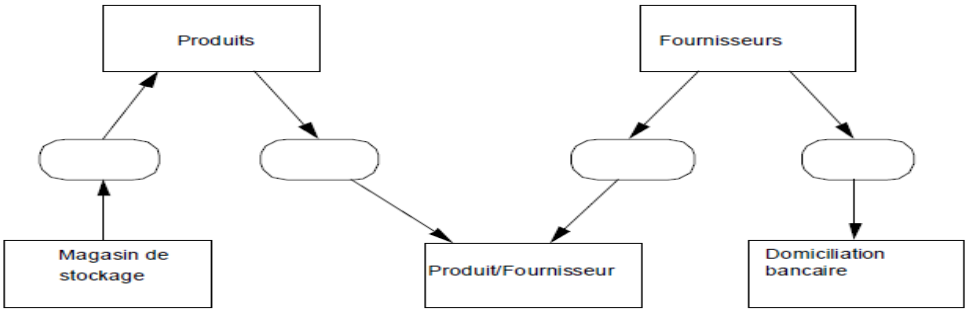

Figure I .8: Schéma représentant le sous-système d'information.

9.3 Modèle relationnel

C'est un article publié en 1969 par un mathématicien du centre de recherche IBM, **Codd**, qui définit les bases de ce modèle relationnel. Codd s'est intéressé au concept d'information et a cherché à le définir sans se préoccuper de la technique informatique, de ses exigences et de ses contraintes. Il a étudié un

modèle de représentation des données qui repose sur la notion mathématique de "relation".

9.3.1 Définitions

- Une relation est un ensemble de tuples (lignes), dont l'ordre est sans importance. Les colonnes de la table sont appelées attributs ou champs. L'ordre des colonnes est défini lors de la création de la table.
- Une clé est un ensemble ordonné d'attributs qui caractérise un tuple. Une clé primaire le caractérise de manière unique, à l'inverse d'une clé secondaire.
- On dit qu'un attribut A est un **déterminant** si sa connaissance détermine celle de l'attribut B (B dépend fonctionnellement de A).

9.3.2 Caractéristiques du modèle

- Schéma de données facile à utiliser : toutes les valeurs sont des champs de tables à deux dimensions.
- Améliore l'indépendance entre les niveaux logique et physique : pas de pointeurs visibles par l'utilisateur.
- Fournit aux utilisateurs des langages de haut niveau pouvant éventuellement être utilisés par des non-informaticiens (SQL, L4G) et un ensemble d'opérateurs basé sur l'algèbre relationnelle : union, intersection, différence, produit cartésien, projection, sélection, jointure, division.
- Optimise les accès aux bases de données.
- Améliore l'intégrité et la confidentialité : unicité de clé, contrainte d'intégrité référentielle.
- Prend en compte une variété d'applications, en gestion et en industriel.
- Fournir une approche méthodologique dans la construction des schémas.

Il est évident qu'une base de données relationnelle présente des avantages tels que :

- L'indépendance des utilisateurs vis à vis de la structure logique, la structure physique, la stratégie d'accès aux données.
- La puissance de représentation grâce à conception rigoureuse du schéma (approche méthodologique).

- La rapidité d'écriture du code.
- La manipulation par des non informaticiens pour des cas simples sinon la connaissance de la structure de la base de données et de la maîtrise de l'algèbre relationnelle sont nécessaires.
- L'approche non procédurale permettant un traitement uniforme de la définition, de la manipulation et du contrôle des données.
- L'évolutivité.
- La prise en compte des contraintes d'intégrité.

9.4 Modèle déductif

Un SGBD Déductif est un SGBD qui peut faire des déductions sur la base **des règles** et des **faits** enregistrés dans la base de données.

Le but est d'utiliser des méthodes semblables à celles pratiquées pour la déduction en intelligence artificielle.

Une base de données déductive (BDD) est constituée de: **BDE** et **BDI.**

Base de données extensionnelle (BDE): Ensemble des faits connus (tuples) dans la BDD relationnelle.

Base de données intentionnelle (BDI): Ensemble des faits ou règles déduits.

Les BDD déductives ne sont pas une nouvelle technologie de SGBD, mais elles rajoutent des fonctionnalités supplémentaires qui permettent de générer des données dérivées (ou déduites) à partir de données déjà existante dans la base. **[16]**

9.5 Modèle objet

Modèle objet est l'annuaire, qui est capables de stocker une multitude d'informations. Il stocke l'information dans des objets, très souvent une fiche individuelle, une machine, une ressource. à laquelle on associe des valeurs, ses attributs. **[10]**

9.6 Modèle multidimensionnel

Il permet de stocker différentes données numériques aux croisements des "n" axes correspondant aux "n" dimensions de la base.
Il est alors possible de naviguer dans cet espace, à différents niveaux d'agrégats (zooms, rotation d'axes, etc.) : ces bases de données sont appelées cubes ou hypercubes en informatique décisionnelle et sont souvent utilisés dans les métiers du contrôle de gestion. **[10]**
Les bases multidimensionnelles sont le plus souvent formées par agrégats de bases pouvant être relationnelles, en tout cas hétérogènes. **[11]**

9.7 Modèle semi-structuré

La popularité du web et l'essor de XML ont contribué à l'émergence des bases de données semi-structurées et des bases de documents. Les modèles classiques de bases de données relationnelles ou objets supportent difficilement les données semi-structurées qui sont complexes, hétérogènes, distribuées, parfois incomplètes. La force des modèles semi-structures est de ne plus imposer de structure a priori dans le schéma mais de la définir a posteriori dans les données elles-mêmes. Plus récemment, les chercheurs s'intéressent à l'extraction de connaissances à partir de données semi-structurées du web. L'objectif de bases de données semi-structurées est de faire le point sur les avancées actuelles dans le domaine des bases de données semi-structurées d'un point de vue théorique et de recenser les applications originales qui s'appuient sur ce formalisme. **[12]**

10. Conclusion :

En d'autres termes, « les bases de données assurent une gestion de données exactes, complètes et disponibles à tout moment, depuis n'importe où et dans la forme voulue ».

Les SGBD les plus évolués actuellement disponibles disposent de la plupart de ces fonctions, mais pas toutes. Il en résulte des différences significatives entre leurs caractéristiques, leur fonctionnement et leurs usages possibles.

Il convient de signaler également que les bases de données sont plus qu'une nouvelle technique de stockage et de manipulation des données. Elles impliquent une nouvelle approche de la conception et de l'utilisation des systèmes d'information et peuvent avoir des conséquences organisationnelles qui sortent largement du cadre du service informatique. Elles obligent les utilisateurs à considérer les données comme une ressource de l'entreprise qui

doit être gérée comme le sont les ressources traditionnelles (personnel, locaux, moyens de production, capitaux) pour être accessible à un grand nombre d'utilisateurs.