Département Informatique

Module	Bases de Données	
Niveau	Licence 2	

Chapitre II INTRODUCTION AUX BASES DE DONNEES.

Présentation Générale.

I. POURQUOI UTILISER UNE BASE DE DONNEES ?

De nos jours, tout le monde utilise une base de données. Que ce soit directement ou indirectement, nous sommes tous confrontés à l'utilisation d'une base de données.

L'histoire des bases de données **remonte aux années 1960**, avec l'apparition des *bases de données réseau* et des *bases de données hiérarchiques*. Dans les années 1980, ce sont les *bases de données object-oriented* qui ont fait leur apparition. Aujourd'hui, les bases de données prédominantes sont les *SQL, NoSQL* et *bases de données cloud*.

Il est aussi possible de classer les bases de données en fonction de leur contenu : bibliographique, textes, nombres ou images. Toutefois, en informatique, on classe généralement les bases de données en fonction de leur approche organisationnelle. Il existe de nombreux types de bases de données différentes : **relationnelle, distribuée, cloud, NoSQL.**

Les premiers systèmes de gestion des bases de données (SGBD) sont apparus dans les années 1960. Avant cette date les données étaient **stockées dans des fichiers** mais on se rendit compte que cela nécessitait ***beaucoup de traitement et d'effort*** de ***développement et peu d'efficacité***. L'idée principale fut alors d'introduire, entre le système d'exploitation et les applications, une couche de logiciel spécialisée dans la gestion de données structurées. L'un des premiers SGBD fut le système IMS basé sur un ***modèle hiérarchique***. Parallèlement, le Database Task Group (fondé par Charles Bachman) définit la norme CODASYL, s'appuie sur une structuration des ***informations en réseaux***. Les modèles hiérarchie et réseau nécessitaient une connaissance approfondie de la structuration physique des données pour leur exploration.

En 1970, Ted Codd (A. M. Turing Award 1981), proposa **le modèle relationnel** pour la représentation des données en se basant sur la théorie des ensembles. Les données sont ainsi organisées en plusieurs tables ou relations homogènes qui peuvent être interrogées et combinées grâce à des opérateurs ensemblistes. La simplicité de ce modèle lui a valu un succès incomparable. La preuve est que pratiquement tous les SGBD actuels se basent sur ce modèle.

Dans les années 1990, un nouveau type de modèles apparait, il s'agit des modèles semi-structurés. Ces derniers sont mieux adaptés à la gestion et à l'intégration de documents hétérogènes tels qu'ils sont publiés sur le Web. Le standard XML est un représentant de ce type de modèles.

Les bases de données ont donc toujours existé dans les systèmes informatiques. Les modèles ont évolué mais la finalité reste la même : stocker les données pour permettre une exploitation optimale et efficace.

II. SYSTEMES DE GESTION DE FICHIERS

Supposant une entreprise qui doit conserver un volume élevé d'informations :

- ✓ Noms, adresses, salaire, adresse des fournisseurs, quantités, prix des items, bilan financier, etc. Ces informations se retrouvent dans différents systèmes de traitement de fichiers.
- ✓ Système de gestion des stocks, système de facturation, système de préparation de paie, programme de gestion de personnel, etc.

Pour obtenir une information, l'employé doit :

1. Déterminer le système à consulter
2. Trouver la bonne personne concernée. ⟹ ***Perte de temps***

De plus, certaines informations sont souvent conservées en plusieurs endroits.

1. *Duplication de données*
2. *Gaspillage* au niveau du volume de fichiers.

Ainsi, les principaux problèmes de ce système (système de traitement de fichiers):

- ✓ *Redondance de certaines informations*
- ✓ *Ne peut répondre rapidement aux demandes d'information provenant de fichiers multiples.*
- ✓ *Coûts élevés pour les modifications (plusieurs systèmes)*

Avec le temps, il y aura…

- ✓ *Accroissement inutile de* :
  - L'ensemble des fichiers ;
  - La taille des fichiers ;
  - Temps d'accès.
- ✓ *Code développé par différents programmeurs et écrits dans différents langages.*
- ✓ *Formats de fichiers différents*
- ✓ *Inconsistance des données*
- ✓ *Lourd à supporter*

De manière générale les inconvénients de ce système sont :

1. Redondance et inconsistance des données
   - ✓ *Informations identiques répliquées dans plusieurs fichiers.*
     Ex: Institution financière
     Adresse et téléphone d'un employé
     - ✓ *Dans le fichier du système de paie*
     - ✓ *Dans le fichier de gestion du personnel.*

2. *Accroissement inutile :*
   - o *De la taille des fichiers ;*
   - o *Des temps d'accès.*
   - o *Risque d'inconsistance des données si le changement d'adresse ne s'effectue pas dans les deux fichiers.*

3. Difficulté d'accès aux données
   - ✓ *Il faut un programme spécifique pour toute nouvelle demande d'information.*
     Ex : Estimation pour l'augmentation de 10% sur le prix des items vendus du mois dernier.
   - ✓ *Le temps d'accès à une requête non prévue peut être très long.*

4. Isolement des données
   - ✓ Les données sont stockées sous différents formats.
     Ex: Numéro civique, NAS
     Type caractère dans un fichier
     Type entier dans un autre fichier.

- ✓ *Formats de fichiers différents*
- ✓ *Grande difficulté d'écrire un programme d'accès général à toute l'information.*

5. Multiplicité des remises à jour
   - ✓ Les traitements concurrents peuvent générer des erreurs.
     Ex: Mises à jour d'un compte en même temps solde de 400 $
     T1: dépôt de 300 $
     T2: retrait de 500 $
     - Si T1 avant T2 : 400$, 700$, 200$ OK
     - Si T2 avant T1 : 400$, -100$, 200$ ERREUR !!!
   - ✓ *Nécessite un programme superviseur pour gérer les transactions : Difficile, voire impossible.*

6. Sécurité
   - ✓ *La sécurité des données et les accès non-autorisés ne sont pas garanties.*
     Ex: Le personnel ne devrait pas avoir accès au programme de paie.

7. Intégrité des données
   - ✓ *Difficulté d'imposer des contraintes*
     Ex : Le solde ne doit jamais être inférieur à 0.

Idéalement, il devrait y avoir…
- ✓ *Un seul exemplaire de chaque élément de données*
- ✓ *Tous les utilisateurs ont accès aux données en ne communiquant qu'avec la base (sans intermédiaire).*
- ✓ *Mesures de protection pour l'information confidentielle*
- ✓ *La complexité du stockage ne doit pas être apparente à l'utilisateur.*

***Solution** : l'idée principale fut d'introduire entre le système d'exploitation et les applications (entre les utilisateurs et les fichiers physiques) une couche logicielle spécialisée dans la gestion des données qu'on appelle Système de Gestion de Base de Données.*

Les SGBD ont été créés pour résoudre tous ces problèmes !!!

III. BASE DE DONNEES (DATABASE)

III.1. NOTIONS DE BASES

Avant de parler de bases de données, il est important de définir ce que c'est qu'une donnée et de présenter ces caractéristiques. La plupart des gens confondent entre le concept de donnée et d'information. Or une donnée n'est pas une information.

- Les *données* (texte, vidéo, image, valeur numérique) sont des éléments de base, bruts et non interprétés, elles servent de matière première pour le processus d'analyse, de traitement et de communication.
- Les *informations* sont des données traitées et contextualisées pour un utilisateur, elles désignent des données organisées, significatives et interprétables, traitées et stockées par des systèmes informatiques.

⇨ *L'information est le résultat de la transformation des données en quelque chose de significatif et de compréhensible*

Exemple :
La séquence binaire « ***01001000 01100101 01101100 01101100 01101111*** » est une donnée sans signification jusqu'à ce qu'elle soit interprétée. A contrario, l'information est le fruit du traitement des données par des algorithmes, des analyses et des interprétations. Ainsi, une fois interprétée, la

séquence binaire évoquée précédemment se révèle comme étant le code ASCII pour le mot « Hello ». La donnée brute devient alors une information compréhensible et intelligible.

- La même information peut être perçue différemment par deux personnes différentes, comme elle peut perdre son importance et devient sans effet avec le temps, chose qui ne s'applique pas à une donnée qui reste tout le temps valable.

⇨ *Pour ces raisons qu'on parle de Bases de Données et non pas de Bases d'informations.*

III.2. DEFINITION DE BDD

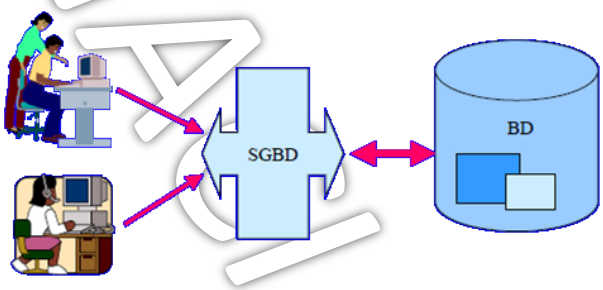
Plusieurs définitions ont été proposées, en peut citer entre autres les définitions suivantes :

- Selon Gardarin, une base de données est un ensemble de données *modélisant les objets d'une partie du monde réel* et servant de support à une application informatique. Pour mériter le terme de base de données, un ensemble de données non indépendantes doit être interrogeable par le contenu, c'est-à-dire que l'on doit pouvoir retrouver tous les objets qui satisfont à un certain critère.
- Une Base de données est un ensemble structuré de données apparentées *qui modélisent un univers réel*. Une BD est faite pour enregistrer des faits, des opérations au sein d'un organisme (administration, banque, université, hôpital, ...). Les BD ont une place essentielle dans l'informatique.
- Une Base de données (BD) est un ensemble cohérent, intégré, partagé de données structurées défini pour les besoins d'une application.
- Une Base de données est un gros ensemble d'informations structurées mémorisées sur un support permanent.

Une base de données est un ensemble structuré de données[1] enregistrées sur des supports accessibles par l'ordinateur[2] pour satisfaire simultanément plusieurs utilisateurs[3] de manière sélective [4]en un temps opportun[5].

IV. SYSTEME DE GESTION DE BASE DE DONNEEs

Un SGBD (en anglais DBMS pour Database Management System) est un logiciel système qui permet d'interagir avec une base de données.



Il permet aux utilisateurs de maintenir, de manipuler (insertion, suppression, mise à jour, recherche efficace) de grandes quantités de données (informations) stockées dans une base de données. Ces données peuvent atteindre quelques milliards d'octets partagée par de multiples utilisateurs simultanément. Les données stockées sont partagées en interrogation et en mise à jour d'une manière transparente. D'autres fonctions complexes peuvent être assurée par le SGBD telle que la protection des données partagées contre les incidents.

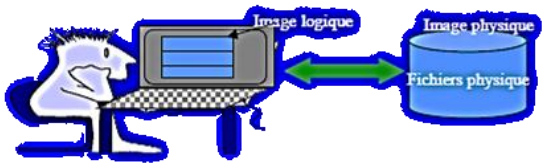
[1] Organisation et description de données
[2] Stockage
[3] Partage de données
[4] Confidentialité
[5] Performance

Contrairement aux systèmes de fichiers, les SGBD permettent de décrire les données de manière séparée de leur utilisation et de retrouver les caractéristiques d'un type de données à partir de son nom (par exemple, comment est décrit un article).

IV.1. OBJECTIFS D'UN SGBD

Les objectifs essentiels et majeurs d'un SGBD est de *garantir l'indépendance des données* par rapport aux programmes (i.e. possibilité de modifier les schémas conceptuel et interne des données sans modifier les programmes), et d'assurer une *abstraction* des données stockées sur disques pour simplifier la vision des utilisateurs (acteurs : administrateurs, programmeurs, non informaticiens,).

- Faciliter la représentation et la description de données. Tel que plus besoin de travailler directement sur les fichiers physiques (tels qu'ils sont enregistrés sur disque). Un SGBD nous permet de décrire les données et les liens entre elles d'une façon logique sans se soucier du comment cela va se faire physiquement dans les fichiers. On parle alors d'image logique de la base de données, (ou aussi description logique ou conceptuelle ou encore de schéma logique). Ce schéma est décrit dans un modèle de données par exemple le modèle de tables, appelé le modèle relationnel.



- Faciliter la manipulation en travaillant directement sur le schéma logique. On peut insérer, supprimer, modifier des données directement sur l'image logique. Le SGBD va s'occuper de faire le travail sur les fichiers physiques.

- Permettre l'ajout des contraintes permettant d'avoir à tout instant des données cohérentes par exemple l'âge d'une personne supérieur à zéro, salaire supérieur à zéro, etc. Dès que l'on essaie de saisir une valeur qui ne respecte pas cette contrainte, le SGBD le refuse.

- Efficacité des Accès (Temps de réponse & débit global).

- Indépendance physique : permettre le changement du schéma physique (modifier l'organisation physique des fichiers, d'ajouter ou supprimer des méthodes d'accès) sans changer le schéma conceptuel. Cela présente deux avantages : le fait de ne pas manipuler des entités complexes rend les programmes d'application plus simples à écrire, la modification des applications n'est pas obligatoire dans le cas de modification des caractéristiques du niveau physique.

- Indépendance logique **:** possibilité de modification du niveau conceptuel sans changement du schéma externe. Cela a deux avantages : pour les programmes d'application du niveau vue, il n'est pas nécessaire d'avoir une vue globale de l'entreprise. En outre, en cas de modification du schéma du niveau logique, les applications du niveau vue sont réécrites seulement si cette modification entraine celle de la vue.

Ainsi, on peut déduire trois fonctions principales d'un SGBD :

- **DEFINITION ET DESCRIPTION DES DONNEES** : codification et structuration grâce au langage de Description de données (LDD).
- **MANIPULATION ET RESTITUTION DES DONNEES :** (insertion, mise à jour, interrogation) : à l'aide d'un langage de manipulation de Données (LMD).
- **CONTROLE :** (partage, intégrité, confidentialité, sécurité) garce au Langage de Contrôle de Données)

IV.2. LES FONCTIONS DUN SGBD

IV.2.1. DEFINITION ET DESCRIPTION DES DONNEES

Une architecture de trois niveaux (niveaux d'abstraction) de description de données a été définie correspondant d'une part à trois représentations équivalentes de l'information, d'autre part aux champs d'interventions respectifs des principaux acteurs. Pour ces derniers, nous utiliserons la terminologie suivante :

- ✓ *Utilisateur naïf :* du non spécialiste des SGBD au non informaticien.
- ✓ *Concepteur et programmeur d'application* : à partir des besoins des différents utilisateurs, écrit l'application pour des utilisateurs "naïfs".
- ✓ *Utilisateur expert :* informaticien connaissant le fonctionnement interne d'un SGBD et chargé d'administrer la base.

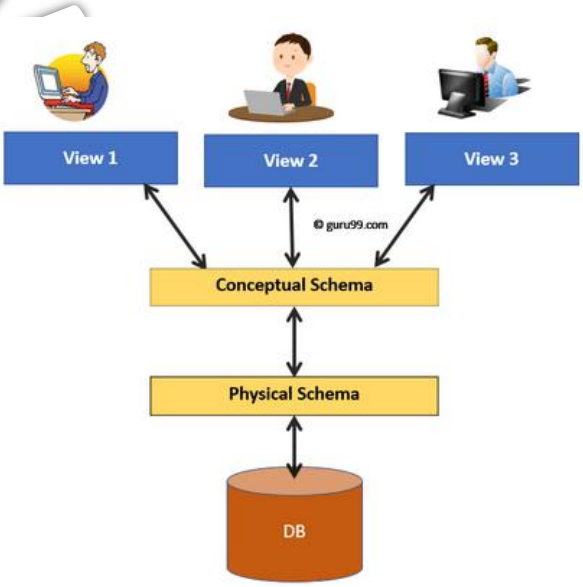
Chaque niveau du SGBD remplit (réalise) un certain nombre de fonctions :

- ✓ **Niveau Conceptuel :** également appelé *niveau logique*, est le niveau central. Le schéma conceptuel permet de *décrire l'ensemble des données.*

  Exemple :
  - *Les objets* : Ouvrages, Etudiants,
  - *Les propriétés* des objets : nom, prénom, adresse, titre de des ouvrages, nombre d'exemplaires etc.
  - *Les liens entre les objets :* un Ouvrage peut être emprunté par un Etudiant.
  - *Les Contraintes :* le nombre d'exemplaires d'un ouvrage d'un Ouvrage est supérieur à zéro.

  C'est une définition logique de la BD (représentation) via un *modèle de données* (Est un ensemble de concepts permettant de décrire la structure d'une base de données. Exemple : modèle Relationnel) et est faite par *l'administrateur* de la BD qui identifie et décri les regroupements de données et leurs interactions.

  Cette description va donner lieu à un schéma de base de données qui se compose d'une description des données et leurs relations ainsi qu'un ensemble de contraintes d'intégrité.



- ✓ **Niveau interne :** Ce niveau appelé aussi *niveau physique* permet la gestion : du stockage des données sur des supports physiques, des structures de mémorisation (fichiers) et d'accès (gestion des index, des clés, etc.), partage de données et gestion de la concurrence d'accès et reprise sur pannes (fiabilité) ; Distribution des données et interopérabilité (accès aux réseaux).

  Objectif : Optimiser les performances

- ✓ **Niveau externe :** Ce niveau est aussi appelé *niveau vue,* c'est le plus haut niveau d'abstraction de la base de données. A la différence du niveau conceptuel et interne dont les schémas décrivent toute une base de données, les schémas externes décrivent seulement la partie des données présentant un intérêt pour un utilisateur ou un groupe d'utilisateurs. En d'autres termes, le niveau externe associe à un utilisateur ou à un groupe d'utilisateurs une vue partielle du monde réel. En conséquence, il y a plusieurs vues d'une même BDD.

  Objectif : simplification et confidentialité

Remarque : Pour une BDD, la description conceptuelle et le schéma interne sont uniques. Par contre, plusieurs schémas externes en général peuvent être définis.

IV.2.2. MANIPULATION ET RESTITUTION DES DONNEES

Permettre à tous types d'utilisateurs d'accéder à la base selon leurs besoins et connaissances. Par conséquent, un ou plusieurs :

- *Administrateurs* de la base doivent avoir la possibilité de décrire les données aux niveaux interne et logique,
- *Développeurs* d'applications écrivent des programmes d'application pour les utilisateurs finaux ou pour eux-mêmes, cela est à partir du niveau conceptuel ou externe,
- *Utilisateurs* peuvent manipuler les données via un langage simple dont ils ont besoin.

IV.2.3. LE CONTROLE

✓ **Partage des données** : L'objectif est ici de permettre le partage des données entre différents utilisateurs et applications. Un utilisateur n'a pas à se soucier si quelqu'un d'autre travaille sur les mêmes informations au même moment et peut accéder aux données en consultation ou en mise à jour comme s'il était seul. Le système doit gérer les conflits en refusant ou en retardant éventuellement un ou plusieurs accès. Il faut assurer que le résultat d'une exécution simultanée de transactions est le même que celui d'une exécution séquentielle. Par exemple, ne pas autoriser la réservation du même siège pour deux passagers différents.

✓ **Intégrité des données** grâce à la définition de contraintes sur les données. Le SGBD veille à ce que toutes les contraintes soient vérifiées à chaque mise à jours (suppression, insertion et modification). En d'autres termes, souvent une donnée ne peut pas prendre une valeur quelconque. **Exemples** : une note doit être supérieur ou égale à 0 et ne doit pas dépasser 20, un salaire mensuel doit être supérieur à 40 000 Dinar et doit raisonnablement rester inférieur à 200 000 Dinar. Un SGBD doit veiller à ce que ces règles soient respectées par les applications lors des modifications des données et ainsi assurer la cohérence des données. Ces règles sont appelées contraintes d'intégrité

✓ **Sécurité des données** : La sécurité des données consiste à :

- Refuser les accès aux personnes non autorisées ou mal intentionnés. Le système doit présenter un mécanisme de vérification des droits d'accès aux objets de la base. Par exemple : un employé peut connaître seulement les salaires des personnes qu'il dirige mais pas le salaire des autres employés de l'entreprise.
- Protéger les données contre les pannes. Le système doit garantir des reprises après panne tout en restaurant la BD dans le dernier état cohérent avant la panne.

✓ **Confidentialité :** plusieurs utilisateurs peuvent utiliser en même temps une base de données, se pose le problème de la confidentialité des données. Des droits doivent être gérés sur les données, droits de lecture, mise à jour, création… qui permettent d'affiner.

V. LES PROPRIETES DES TRANSACTIONS DE BDD

Au sein d'une base de données, le terme de « transaction » désigne *les opérations apportant des modifications aux données,* d'une autre manière n'importe quelle opération effectuée au sein d'une base de données.. Par exemple,

- Un virement bancaire provoquant le débit du compte de l'émetteur et le crédit du compte du bénéficiaire est une transaction.
- Création d'un nouvel enregistrement ou d'une mise à jour des données

Une transaction est une unité logique de travail, c'est à dire une séquence d'instructions, dont l'exécution assure le passage de la BD d'un état cohérent à un autre état cohérent.

Ces transactions doivent toutefois présenter **quatre propriétés** visant à garantir leur validité même en cas d'erreur ou de pannes informatiques. Ces quatre propriétés sont **l'Atomicité, la Cohérence, l'Isolation et la Durabilité**. Ces quatre principes permettent d'assurer que *les transactions de bases de données* soient traitées de façon *fiable*. Ils sont gérés par le moteur de base de données, et doivent impérativement être respectés. Leur implémentation n'est pas des plus simples, car **tous les paramètres de chaque transaction lancée doivent être confrontés à ces quatre attributs au même moment**.

ACID est un acronyme résumant les quatre propriétés élémentaires d'une transaction au sein d'une base de données : Atomicité, Cohérence, Isolation, Durabilité. Découvrez la signification précise de ces propriétés.

V.1. ATOMICITE

L'atomicité des transactions au sein des bases de données signifie que t**ous les changements apportés aux données doivent être totalement effectués (réalisés)**, ou **pas du tout.** Soit tous les changements apportés par la transaction sont enregistrés **pour** la postérité, soit aucun ne l'est.

En outre, l'atomicité permet **d'éviter que les changements prennent effet** en cas de panne de l'application ou du serveur de la base de données en plein milieu de la transaction. Ainsi, la base de données ne risque pas d'être corrompue par des opérations imprévisibles et à moitié complétées.

Exemple : Sur 5000 lignes devant être modifiées, si la modification d'une seule échoue, alors la transaction entière doit être annulée. C'est primordial, car chaque ligne modifiée peut dépendre du contexte de modification d'une autre, et toute rupture de ce contexte peut avoir des conséquences catastrophiques.

V.2. COHERENCE

La **cohérence** signifie que les transactions doivent respecter les *contraintes d'intégrité des données* de la base de données. Ainsi, une transaction qui commence avec un ensemble de données cohérent (respectant les contraintes d'intégrité) doit résulter par un ensemble de données cohérent.

Pour maintenir la cohérence, un SGBD peut **abandonner les transactions qui risquent de provoquer une incohérence**. Précisons que cette cohérence ne s'applique pas aux étapes intermédiaires de la transaction.

Pour assurer la cohérence au fil du temps, **il est possible d'utiliser une base de données temporelle**. Ceci permet de spécifier les contraintes de données qui doivent traverser la dimension temporelle.

Ainsi, s'il arrive qu'un changement risque de perturber l'intégrité des données, alors soit le système doit modifier les données dépendantes, soit la transaction doit être interdite.

VI.3. ISOLATION

L'**isolation** signifie que **les écritures et lectures des transactions réussies ne seront pas affectées** par les écritures et lectures d'autres transactions, qu'elles soient ou non réussies. Les transactions isolées peuvent être « sérialisées », ce qui signifie que l'état final du système peut être atteint en effectuant les transactions une par une.

D'une autre manière l'isolation signifie que **les transactions lancées au même moment ne doivent jamais interférer (interagir) entre elles**, ni même agir selon le fonctionnement de chacune. Par exemple, si une requête est lancée alors qu'une transaction est en cours, le résultat de celle-ci ne peut montrer que l'état original ou final d'une donnée, mais pas l'état intermédiaire. De fait, les transactions doivent s'enchaîner les unes à la suite des autres, et non de manière concurrentielle.

V.4. DURABILITE

Enfin, **la durabilité** garantit que **les transactions réussies survivront de façon permanente** et ne seront pas affectées par d'éventuelles pannes ou problèmes techniques. Les changements apportés aux données doivent être permanents. Plus précisément, ce sont les effets logiques des données modifiées sur les futures transactions qui doivent être permanents.

La simple écriture des données sur le disque ne suffit pas pour atteindre la durabilité. Pour cause, le disque peut par exemple tomber en panne. Il est **nécessaire que le DBMS écrive des logs** sur les changements effectués. Ces logs doivent être permanents, et éventuellement redondants.

En cas de panne du disque, de l'OS ou du DBMS, la base de données sera redémarrée et **le DBMS pourra vérifier si les données sont synchronisées avec les logs**. En effet, les logs contiennent les effets de toutes les transactions effectuées. Ce n'est pas forcément le cas des fichiers de données. Si des informations sont manquantes, les opérations enregistrées dans les logs seront à nouveau effectuées.

Durabilité signifie que **toutes les transactions sont lancées de manière définitive**. Une base ne doit pas afficher le succès d'une transaction, pour ensuite remettre les données modifiées dans leur état initial. Pour ce faire, toute transaction est sauvegardée dans un fichier journal, afin que, dans le cas où un problème survient empêchant sa validation complète, celle-ci puisse être correctement terminée lors de la disponibilité du système.

Exemple :
Une transaction qui transfert 1000 Euro du compte A vers le compte B
1. Lire(A)
2. A := A − 1000
3. Ecrire(A)
4. Lire(B)
5. B := B + 1000
6. Ecrire(B)

- La base est cohérente si la somme (A+B) ne change pas suite à l'exécution de la transaction (**cohérence**).
- Si la transaction "échoue" après l'étape 3, alors le système doit s'assurer que les modifications de A ne soient pas persistantes (**atomicité**).
- Une fois l'utilisateur est informé que la transaction est validée, il n'a plus à s'inquiéter du sort de son transfert (**durabilité**).
- Si entre les étapes 3 et 6, une autre transaction est autorisée à accéder à la base, alors elle "verra" un état incohérent (A + B est inférieur à ce qu'elle doit être). L'**isolation** n'est pas assurée. La solution triviale consiste à exécuter les transactions en **séquence**.

VI. TYPES DE MODELES DE DONNEES

Un modèle est une abstraction de la réalité sur laquelle on peut opérer. Il existe plusieurs modèles et parmi eux on trouve :

VI.1. MODELE SEMANTIQUE

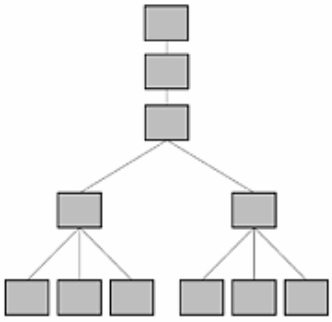
Le modèle sémantique est parmi les modèles de bases de données les moins courants. Il comprend des informations sur la façon dont les données stockées sont rattachées au monde réel.

VI.2. MODELE ENTITE-ASSOCIATION

Le modèle Entité-Association (EA) en français, ER en anglais (Entity Relationship) permet de décrire l'aspect conceptuel des données à l'aide d'entités et d'associations.

VI.3. MODELE HIERARCHIQUE

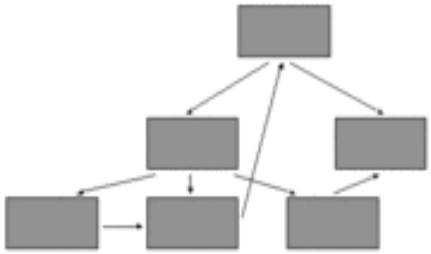
Le modèle hiérarchique (*ou arbre*) organise les données dans une structure arborescente, où chaque enregistrement n'a qu'un seul parent (racine). Les enregistrements frères et sœurs sont triés dans un ordre particulier. Ce modèle convient à la description de plusieurs relations du monde réel. Exemple : ADABASE (1970), System 2000 (1967).



- ✓ **Exemple**
  Par exemple, le *canard* appartient à la famille des *anatidés* qui elle-même appartient à l'ordre des *ansériformes* qui lui-même appartient à la classe des *oiseaux* qui elle-même appartient au sous-embranchement des *vertébrés* qui lui-même appartient au règne *animal*.
- ✓ **Histoire et limites**
  Les structures de données hiérarchiques ont été largement utilisées dans les premiers systèmes de gestion de base de données de type *mainframe*. Elles ont toutefois montré des limites pour décrire des structures complexes, répondre aux besoins réels et suivre l'évolution des systèmes d'information. Comme on le voit dans l'exemple cité plus haut, l'organisation hiérarchique des bases de données est particulièrement adaptée à la modélisation de *nomenclatures*, mais si le principe de relation « 1 vers N » n'est pas respecté (le *canard* n'appartient bien qu'à une seule *famille* mais, par exemple, un *malade* peut être en relation avec plusieurs *médecins*), alors la hiérarchie doit être transformée en un réseau.

VI.4. MODELE RESEAU

Modèle réseau (ou *graphe*) est un modèle *hiérarchique étendu* qui autorise relations transverses (i.e. relations plusieurs-à-plusieurs entre des enregistrements liés). Un enregistrement peut être un membre ou un enfant dans plusieurs ensembles. Cela permet de traduire des relations complexes. Ex : TOTAL (1978).



Le **modèle réseau** est une manière de représenter les données dans le cadre d'une base de données. Ce modèle est en mesure de lever de nombreuses difficultés du modèle hiérarchique grâce à la possibilité d'établir des liaisons de type n-m en définissant des associations entre tous les types d'enregistrements.

Ce modèle est une extension du modèle hiérarchique, les liens entre objets peuvent exister sans restriction. Pour retrouver une donnée dans une telle modélisation, il faut connaître le chemin d'accès (les liens), ceci rend encore les programmes dépendants de la structure de données.

VI.5. MODELE RELATIONNEL

Dans le modèle relationnel, les informations décomposées et organisées sont stockées dans des tables. Exemple : 80% des SGBD sont relationnelles, ORACLE (85% du marché), DB2, SQL Server, ACCESS, etc. Le schéma relationnel est l'ensemble des RELATIONS qui modélisent le monde réel ; tel que les relations représentent les entités du monde réel (par exemple : des personnes, des objets, etc.) ou les associations entre ces entités.

Remarque : Avant d'aboutir à une base de données sous sa forme finale (i.e. implémenter dans le SGBD) il faut qu'on passe par l'étape de *conception* et de *modélisation*, durant laquelle on présente les objets de la réalité et les interactions entre eux de manière à ce qu'ils soient facilement manipulables. Dans notre cas on utilise le modèle sémantique *entité/association* qu'on détaillera dans le chapitre suivant.