Chapitre 1: Introduction aux bases de données et aux systèmes de gestion de base de données

1. Bases de données

Définition

Une base de données (BD) est une collection de données **persistantes** et **pertinentes** utilisées par des systèmes d'application de certaines **organisations**. (Chris L. Date [Dat95])

Trois mots clés sont utilisés dans cette définition :

• données persistantes : elles diffèrent d'autres données plus éphémères (de nature transitoire) telles que les données d'entrée et les données de sortie.

• pertinentes : données nécessaires et ciblées pour les différentes applications utilisatrices.

• organisations : peut être un simple individu avec une petite base de données privée (un médecin, un notaire, ...) ou une société complète avec une base de données partagée très importante (une banque, une université, un hôpital,...).

Avantages d'utilisation d'une base de données par rapport aux systèmes traditionnels (fichiers) :

Dans le contexte classique, les fichiers sont conçus pour satisfaire les besoins d'une application. Il s'ensuit qu'une même information peut être représentée dans plusieurs fichiers. Si elle intervient dans deux applications, elle peut même se présenter différemment (format, codage,...). Cela entraîne un risque de conflit : lors d'une mise à jour par exemple, tous les duplicatas ne sont pas mis à jour en même temps par suite de leur non appartenance à l'application qui réalise cette mise à jour. L'utilisation de BD a les avantages suivants :

• **compacité :** plus besoin de fichiers manuels volumineux, fiches cartonnées par exemple où le risque de perte d'information est important,

• **rapidité :** recherche et mise à jour rapide d'information

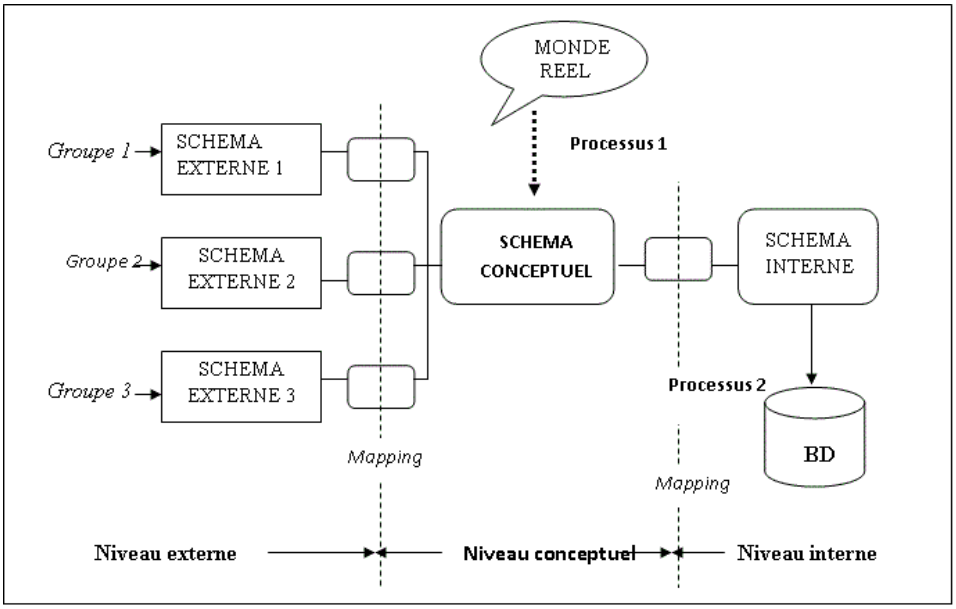
• **données intégrées** : regroupement de plusieurs fichiers de données où toute redondance a été totalement ou partiellement éliminée. D'où la diminution de l'incohérence.

• **données partagées**, i.e., les différents utilisateurs peuvent faire usage de ces données pour des buts divers. Ils peuvent également accéder aux mêmes données simultanément.

2. Niveaux de représentation des données

Trois niveaux sont proposés par le groupe ANSI/SPARC pour l'architecture d'une base de données :

le niveau conceptuel, le niveau externe et Le niveau interne



2.1. Le niveau conceptuel

Le schéma conceptuel est la « **charpente** » d'une BD. Il décrit en termes abstraits la réalité organisationnelle et ses règles de gestion.

Le processus de conception consiste à traduire les objets du monde réel en classes d'objets suivant des modèles de données bien définis.

Il existe plusieurs types de modèles de données classés en trois catégories :

- **Les modèles de 1ère génération** : décennie 60 : Hiérarchique et Réseau
- **Les modèles de 2ième génération** : les décennies 70 et 80
  -E/A (Entités/Association), Relationnel, Réseaux sémantiques
- **Les modèles de 3ième génération** : la décennie 90: Modèle orienté objet

2.2 Le niveau externe

Un schéma externe appelé aussi **Vue** est une perception des données par un programme d'application. C'est un **Sous-schéma** du schéma conceptuel.

Il peut contenir des informations complémentaires (par exemple des informations de calcul).

Remarque

Les différents schémas externes permettent la validation du schéma conceptuel par l'utilisation de technique de construction du schéma conceptuel à partir de l'intégration de vues.

2.3 Le niveau interne

C'est le niveau relatif à la mémoire physique. Il concerne la manière selon laquelle les données sont réellement stockées, il correspond au schéma interne.

Remarque

Le processus de développement d'une BD comprend deux grandes phases :

- Une **phase de conception** dont le résultat est le schéma conceptuel. La validation s'effectue à travers les schémas externes

- Une **phase de réalisation** dont le résultat est le schéma interne et la BD

3. Système de Gestion de Base de Données (SGBD):

3.1.Définition

Un système de gestion de base de données (SGBD) est un logiciel qui prend en charge tous les accès à la BD.

3.2. Objectifs d'un SGBD :

Un SGBD dispose généralement d'un certains nombre d'objectifs

a) Définition des données

permet de définir les données (schémas externes, schéma conceptuel , schéma interne et tous les liens correspondants) sous une forme non compilée et de les convertir dans la forme objet appropriée. Le SGBD doit donc être muni d'un langage de définition de données (LDD).

b) Non redondance

La conception intégrée de la BD grâce à la notion de schéma de données (qui est une représentation globale des informations à l'aide de modèles de données pour un ensemble d'applications) évite les redondances constatées dans les SGF.

c) Cohérence

Les données de la base obéissent à des règles appelées contraintes d'intégrité (CI).

Une CI est une assertion que doit vérifier le SGBD à chaque fois que la donnée sur a quelle elle est définie est sollicitée (déchargement de l'utilisateur lors des opération de création, modification, suppression).

Une BD cohérente est une BD dont les CI sont toujours vérifiées lors d'opérations sur la BD.

Exemples

-spécialité licence a pour valeur ('ACAD', 'ISIL', 'GTR')

-une note est comprise entre 0 et 20

d) Efficacité des accès aux données

Les accès aux données seront plus efficaces que dans les SGF grâce notamment :

-au développement d'index sophistiqués

-à l'existence de plusieurs chemins d'accès à une donnée

-à l'existence de techniques d'optimisation de requêtes qui sélectionnent le chemin optimal à une donnée.

e) Administration centralisée

C'est le rôle de l'administrateur de la BD qui a pour fonction de :

• Définir des structures de stockage,

• Définir des structures de données,

• Assurer le suivi et le contrôle de leur évolution.

f) Indépendance physique

permet à l'administrateur de la BD de modifier l'organisation physique des données (ajouter une table d'index...) sans modification des programmes d'application déjà existants.

g) Indépendance logique

permet de modifier le schéma conceptuel (ajouter de nouvelles classes d'objets, de nouvelles associations,...) sans modifier les programmes d'applications.

h) Partageabilité

Le SGBD doit permettre à plusieurs applications de partager les données. Pour cela, il doit gérer les conflits d'accès (les détecter et les solutionner) . Ceci est possible grâce à la notion de **transaction** et de définition d'**algorithmes de gestion de la concurrence.**

Remarque

Une transaction est une unité de traitement séquentiel exécutée pour le compte d'un utilisateur. C'est une suite finie d'actions (opérations) portant sur des objets de la BD dont l'exécution la maintient dans un état cohérent.

Exemple de transaction

Opération de transfert d'une somme d'argent X d'un compte bancaire C1 à un compte C2.

i) Sécurité et confidentialité

Les données doivent être protégées contre les pannes et contre les accès mal intentionnés.

- Protection contre les pannes : On distingue deux types de pannes.

  - Pannes simples caractérisées par la perte du contenu de la mémoire centrale
  - Pannes catastrophiques caractérisée par la perte du contenu des mémoires secondaires.

Dans les deux cas, il est possible de récupérer un état cohérent des données grâce à l'existence et à la gestion de journaux de transactions, qui gardent la trace d'exécutions antérieures.

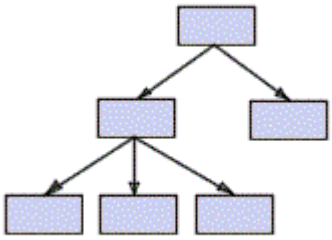
- Protection contre les accès mal intentionnés : La définition de droits d'accès, de mots de passe etc. gérés par le SGBD permet la confidentialité des données de la base et leur inviolabilité.

4. Modèles de données

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 faisait ressentir. Il existe cinq modèles de SGBD, différenciés selon la représentation des données qu'elle contient :

4.1 Le modèle hiérarchique:

C'est le premier modèle de SGBD. Les données sont classées hiérarchiquement, selon une arborescence descendante. Ce modèle utilise des pointeurs entre les différents enregistrements Père, fils, frère



Le système le plus représentatif est le **SGBD IMS** développé par IBM.

Caractéristiques :

-LMD de type procédural,

-Seuls les liens **1-1** et **1-N** sont pris en compte.

Exemple: base de données "Approvisionnement"

Un fournisseur est décrit par son numéro NF, son nom NOM, son code CODE et la ville où il est localisé VILLE.

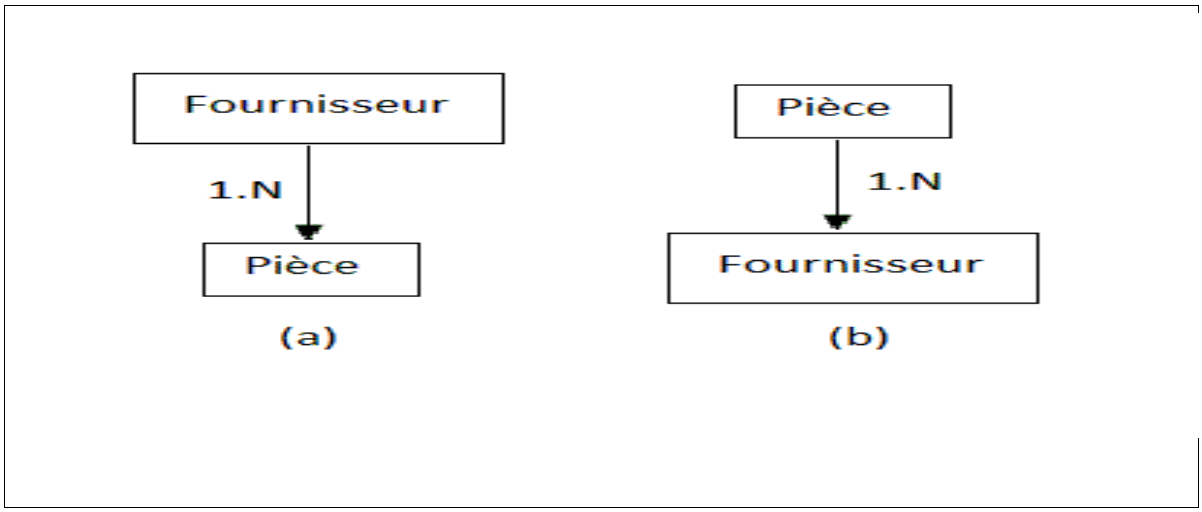
Une pièce est décrite par son numéro NP, son nom NOM, son poids POIDS, son matériau MATERIAU et la ville où elle est stockée VILLE.

Décrivons l'association qui lie les fournisseurs aux pièces qu'ils commercialisent et inversement :

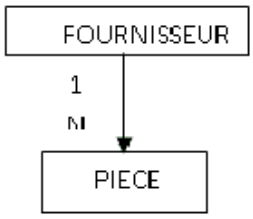
Un fournisseur f fournit une pièce p en une certaine quantité q.

Un fournisseur peut fournir plusieurs pièce et une pièce peut être fournit par plusieurs fournisseurs (association (**N-M**)).

La contrainte de la hiérarchie (**1-N**) ne permet pas cette association. Selon les objectifs de l'application, le concepteur doit choisir entre le schéma **(a)** ou le schéma **(b)** de la figure suivante.



Exemple d'implémentation du cas (a)



F1	Haroun	120	Alger			
	P1	Vis	Fer	12	Alger	300
	P2	Boulon	Acier	17	Tunis	200
	P3	Ecrou	Zinc	17	Paris	400
	P4	Ecrou	Fer	14	Alger	200
	P5	Came	Zinc	12	Tunis	100
	P6	Clou	Fer	19	Alger	100
F2	Bouzid	110	Tunis			
	P1	Vis	Fer	12	Alger	300
	P2	Boulon	Acier	17	Tunis	400
F3	Mamir	130	Tunis			
	P2	Vis	Fer	12	Alger	200
F4	Kaci	120	Alger			
	P2	Vis	Fer	12	Alger	200
	P4	Ecrou	Fer	14	Alger	300
	P5	Came	Zinc	12	Tunis	400

Considérons des requêtes d'interrogation de données

• Q1 : Trouver les pièces fournies par F2?

La formulation de Q1 en **DL/1** (nom du langage de manipulation de données IMS) :

```
GET UNIQUE fournisseur WITH        NF = 'F2'
NEXT:  GET NEXT pièce FOR THIS fournisseur
pièce FOUND ? IF NOT
Exit.
PRINT NP
 GO TO NEXT
Exit.
```

GET UNIQUE .... : permet de rechercher le premier segment répondant au critère de recherche

GET NEXT .... : permet de passer au segment suivant.

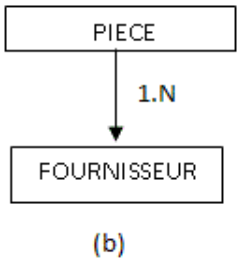
• Q2: Trouver les fournisseurs qui fournissent P2?

```
GET TO START OF DATA
NEXT 1 :GET NEXT fournisseur
Fournisseur FOUND? IF NOT  EXIT
 GET UNIQUE pièce FOR THIS fournisseur WITH NP = 'P2'
Pièce FOUND ? IF NOT GO TO NEXT1
PRINT NF
GET NEXT 1.
Exit
```

On constate que si Q1 et Q2 sont symétriques quand au rôle joué par fournisseur et pièce, leur formulation ne l'est pas à cause de la hiérarchie qui impose une séquence d'analyse des données enregistrées dans la base. On constate que l'on n'a pas une formulation naturelle des question qui permettent l'extraction de données de la base

ce qui demande de la part des utilisateurs du SGBD une haute compétence technique. Par conséquent, cela rend difficile l'écriture, la mise au point et la maintenance des programmes d'application.

Si la requête Q2 est fréquente, une autre solution consiste à ajouter à la base de données les instances correspondantes à l'implémentation du cas (b).



(b)

P1	Vis	Fer	12	Alger	
	F1	Haroun	120	Alger	300
	F2	Bouzid	110	Tunis	300
P2	Boulon	Acier	17	Tunis	
	F1	Haroun	120	Alger	200
	F2	Bouzid	110	Tunis	400
	F3	Mamir	130	Tunis	200
	F4	Kaci	120	Alger	200
P3	Ecrou	Zinc	17	Paris	
	F1	Haroun	120	Alger	400
P4	Ecrou	Fer	14	Alger	
	F1	Haroun	120	Alger	200
	F4	Kaci	120	Alger	300
P5	Came	Zinc	12	Tunis	
	F1	Haroun	120	Alger	100
	F4	Kaci	120	Alger	400
P6	Clou	Fer	19	Alger	
	F1	Haroun	120	Alger	100

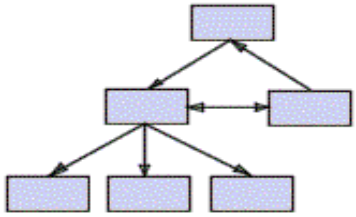
Inconvénients :

- Redondance

- Très forte dépendance entre la description de la structure de données et la manière dont ces dernières sont enregistrées sur le support à accès direct.

4.2 Le modèle réseau

Comme le modèle hiérarchique ce modèle utilise des pointeurs vers des enregistrements. Toutefois la structure n'est plus forcément arborescente dans le sens descendant C'est un ensemble de nœuds et d'arcs. Les nœuds représentent les objets et les arcs représentent les associations entre ces objets.



• Description des objets :

Les concepts définissant les objets sont : l'atome, l'agrégat, l'article ou enregistrement logique.

un **atome** (data item) est la plus petite unité de données possédant un nom. Un atome est représenté par une valeur dans la base

un **agrégat** (data aggregate) est une collection d'atomes rangés consécutivement dans la base et portant un nom.

un **article** ou enregistrement logique (record) est une collection d'agrégats et d'atomes rangés côte à côte dans la base de données et constituant l'unité d'échanges entre la BD et les applications.

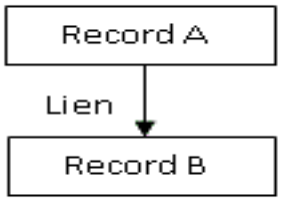
• Description des associations :

Nous distinguons trois types de liens dans le modèle réseau :

1-1 : à un objet de A correspond un et un seul objet de B et vice versa

1-N : à un objet de A peut correspondre un à plusieurs objets de B mais inversement, à un objet de B ne peut correspondre qu'un seul objet de A (lien hiérarchique).

M-N : à un objet de A peut correspondre un à plusieurs objets de B et inversement.



4.3. Le modèle réseau CODASYL

En 1969 a eu lieu la conférence appelée CODASYL (Conference On Data System Language) dont un groupe, le DBTG (Data Base Task Group) a normalisé et présenté des concepts communs pour la définition de schémas réseaux.

Clés :

Tous les types d'objets représentés par des nœuds sont identifiés par des clés

Une **clé** est un atome ou un agrégat qui permet d'identifier de manière unique chaque record.

Liens :

Les types de liens supportés par la norme CODASYL sont les liens 1-1 et 1-N.

Le lien CODASYL est appelé **SET**. Un SET est un lien entre un enregistrement propriétaire appelé **OWNER** et un ou plusieurs enregistrements membres appelé **MEMBER**.

Un enregistrement ne peut être à la fois propriétaire et membre d'un même SET (pas de boucle). Cependant un enregistrement peut être propriétaire de plusieurs SET différents et ou membre de plusieurs SET différents.

Dans certains cas, le lien possède des propriétés qui relient les objets en relation. On les appelle des **données d'intersection.**

lien M-N

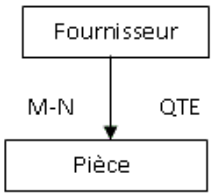
Le lien M-N ne peut être représenté directement dans la norme **CODASYL**, étant donné que celle-ci ne prend en charge que les liens de type hiérarchique (*owner-member*).

Les liens **M-N** nécessitent des transformations : Un artifice est utilisé qui consiste à créer un enregistrement intermédiaire, avec deux liens **1-N**.

Le record intermédiaire aura pour clé la concaténation des clés des records composant le lien **M-N** et pour propriétés les données d'intersection si elles existent.

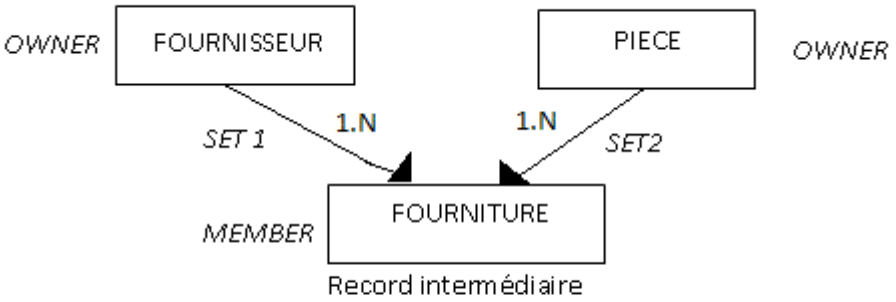
Exemple

Le lien entre fournisseur et pièce est un lien M-N ayant une propriété d'intersection : la quantité fournie.



Transformation du réseau fournisseur-pièce

On crée le record intermédiaire FOURNITURE puis on associe FOURNISSEUR à FOURNITURE puis FOURNITURE à PIÈCE. On constitue ainsi 2 lien SET1 et SET2.

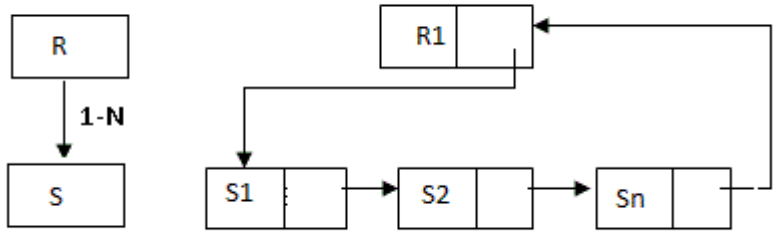


SET 1 : FOURNISSEUR- FOURNITURE
SET2 : PIECE- FOURNITURE

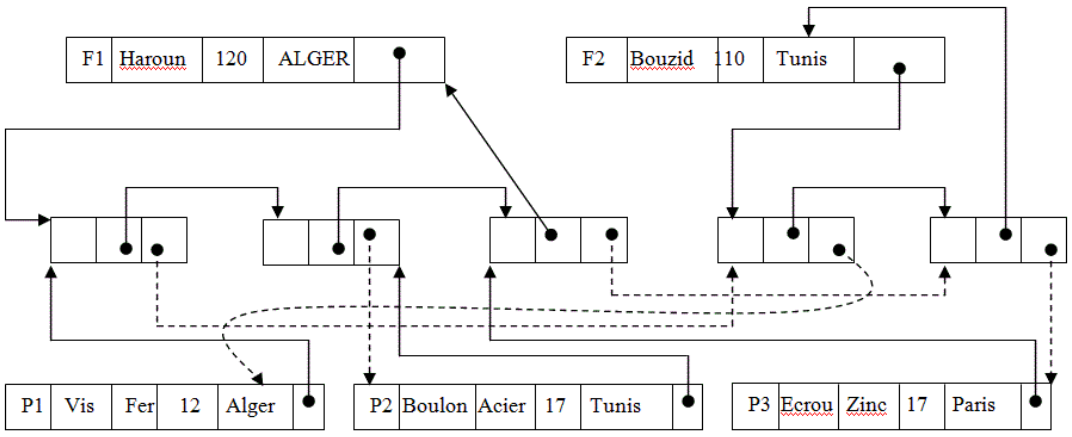
Implémentation des liens codasyl

Le lien CODASYL est implémenté grâce à la liste circulaire (ou anneau)

schéma d'occurrences :



Exemple



Le LMD (langage de manipulation) du modèle réseau doit permettre le parcours des anneaux.

Instructions :

FIND permet de désigner un record et de positionner un pointeur (**point courant)** qui contient l'adresse du record sélectionné.

GET permet la lecture du record sélectionné.

Reconsidérons les requêtes précédentes Q1et Q2

Q1 Trouver les pièces fournies par F2	Q2 Trouver les fournisseurs qui fournissent P2
FIND fournisseur WITH NF = 'F2'	FIND piece WITH NP = 'P2'
NEXT : FIND NEXT LINK FOR THIS Fournisseur	NEXT : FIND NEXT LINK FOR THIS piece
LINK FOUND ? IF NOT	LINK FOUND ? IF NOT
EXIT	EXIT
FIND Piece FOR THIS LINK	FIND Fournisseur FOR THIS LINK
GET piece	GET fournisseur
PRINT NP	PRINT NF
GO TO NEXT	GO TO NEXT
EXIT	EXIT

On constate que des questions symétriques mettent en œuvre des algorithmes symétriques, ce qui est un avantage par rapport à l'approche hiérarchique. Mais, on rencontre des problèmes de stratégie d'accès (comme dans le modèle réseau).

Par exemple, si l'on veut trouver la quantité des pièces P2 fournies par F2. Faut-il rechercher le fournisseur F2 et retrouver P2 ou inversement. Selon le cas, le temps de réponse est différent.

La réalisation des opérations de mise à jour (insertion, suppression, modification) est simplifiée par rapport au modèle hiérarchique :

Le majeur inconvénient de la structure réseau réside dans la nécessité de parcourir des chaînages et de gérer des points courants. La gestion des points courants est très délicate à cause des interactions des instructions du LMD sur ces points courants.

4.4. Le modèle "Entité-Association" ("Entity-Relationship" E/R)

Introduction

C'est un modèle conceptuel indépendant des SGBD

Il est généralement utilisé en mode graphique. Sa simplicité l'a rendu très populaire.

Les schémas de données qu'il permet de construire peuvent être traduits presque automatiquement dans les modèles des SGBD des différentes générations.

Concepts

Il est basé sur les concepts ENTITE TYPE/ ASSOCIATION TYPE et sur deux formes d'abstraction : la classification et l'instanciation.

Une association est une combinaison d'entités dans laquelle chacune d'elle joue un rôle spécifique.

Les entités et les associations sont caractérisées par des propriétés

Les entités, associations et propriétés sont classées et définies par des types

Un type définit en intension ou extension une population d'objets de même nature. On parle aussi de classe d'entité ou classe d'association.

Une entité ou une association sont caractérisés par le doublet « attribut valeur »

Le schéma de l'entité type définit l'entité type en intention . De même, Le schéma d'association type définit l'association type en intention.

Exemple

Soit l'entité-type ou la classe Employé

Le schéma de cette entité-type s'écrit : Employé(Nom, Date_naissance, Salaire).

Une **occurrence** ou **instance** de cette entité-type : employé ayant les propriétés :

(Nom, ‘Ali'), (Date naissance, ‘251268'), (Salaire, ‘400000')

Une **extension** de la classe Employé est un ensemble d'instances ou d’occurrences :

{Attaf, 251268, 40000}

{Benmohamed, 121065, 200000}

{Benmiloud, 050763, 100000}

Le **schéma** d'association-type Affectation s'écrit :

Affectation (Employé, Projet : Date-deb, Date-fin)

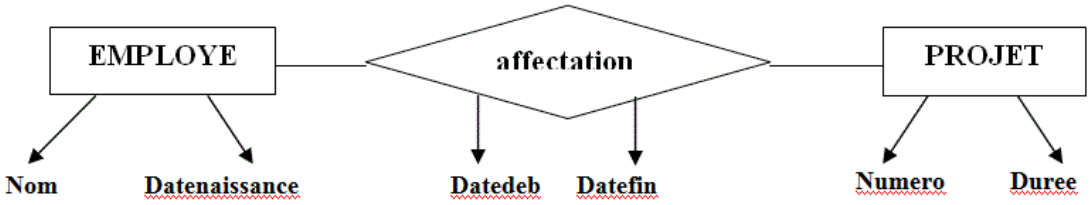
Une **occurrence** de la classe association est :

Affectation(Employé, Projet) : (Date-deb, ‘010196'), (Date-fin, '010199')

Association binaire

Graphiquement une entité-type est représenté par un rectangle alors qu'une association-type est représentée par un losange.

Une association binaire relie deux entités.

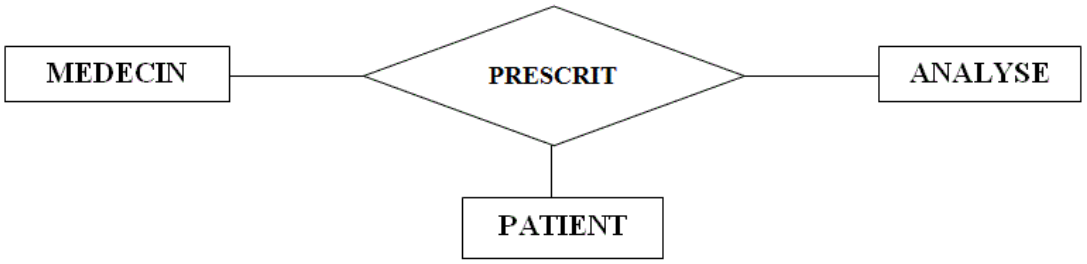


Association N-aire

Une association N-aire relie plus de deux entités.

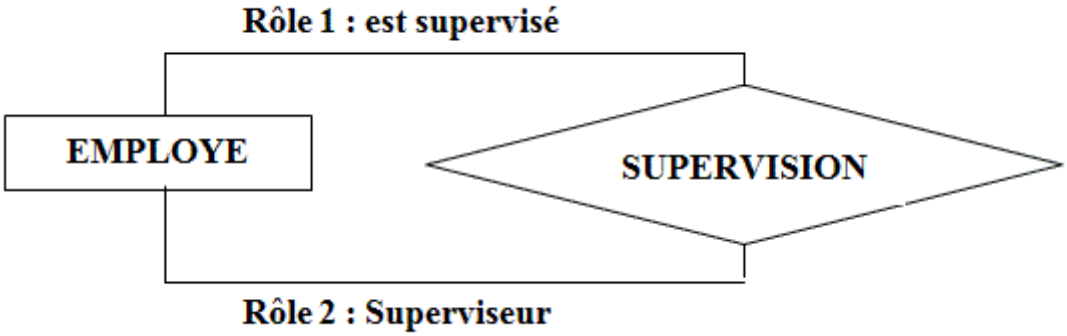
On appelle aussi ternaire, une association reliant trois entités.

L'association PRESCRIT relie un médecin à un patient et aux analyses qu'il doit faire.

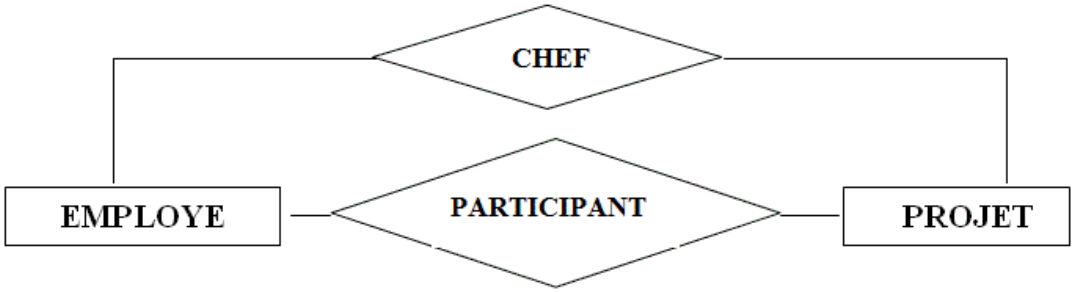


Association récursive

Une entité type peut jouer plusieurs rôles dans une association type



Une entité peut jouer plusieurs rôles dans ses relations avec une autre entité type



- Une **clé** est un attribut ou groupe d'attribut dont les valeurs permettent d'identifier de manière unique les entités d'un même type.. La spécification de la clé d'une entité type implique une contrainte d'unicité sur l'extension de l'entité type.

Exemple :

Clé de EMPLOYE : Nom - Clé de PROJET : Numéro

Clé de MEDECIN: N°MEDECIN - Clé de PATIENT: N°PATIENT

Clé de ANALYSE : N°ANALYSE

- La clé d'une association type est la **combinaison** des clés des entités types participantes

Exemple

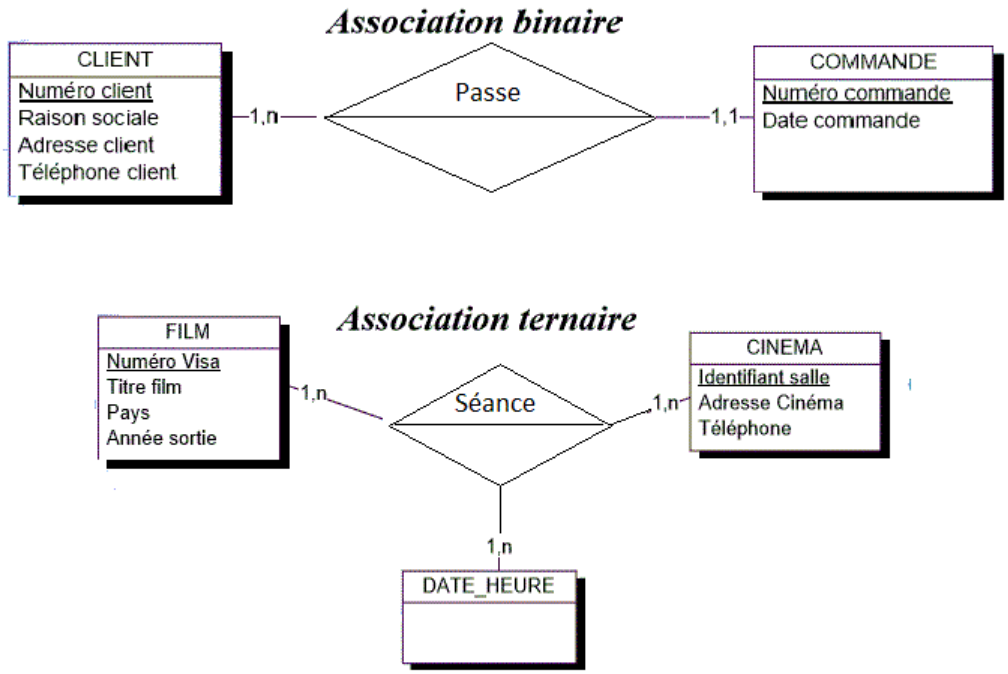
Clé de AFFECTATION : (Nom, Numéro)

Clé de PRESCRIT : (N°MEDECIN,N°PATIENT, N°ANALYSE)

- La **cardinalité** est le nombre d'associations auxquelles une entité peut participer. Il existe trois cardinalité prédéfinies

- **1-1** : a une instance de l'objet A est associée une et une seule instance de l'objet B.
- **1-n** : a une instance de l'objet A est associée une ou plusieurs instances de l'objet B.

  n-m a une instance de l'objet A est associée une ou plusieurs instances de l'objet B et inversement.



4.5. Le modèle relationnel

Les données sont représentées sous forme de tables à deux dimensions (lignes et colonnes). La manipulation de ces données se fait selon la théorie mathématique des relations. Ce modèle est appelé ''relationnel'' car le terme mathématique **relation** est le terme le plus approprié pour une table.

Etudiant

Nom	Prénom	Matricule	Groupe	Section

Module

Cod-Mod	Coéfficient

Etudiant-Module

Matricule	Cod-Module	EMD1	EMD2

Terminologie du modèle relationnel :

Scalaire : c'est la plus petite unité sémantique de données (valeur atomique) telle que la valeur du numéro d'un fournisseur, du poids d'une pièce...

Domaine : ensemble donné de valeurs scalaires, toutes de même type. Par exemple, le domaine des numéros des fournisseurs, le domaine des villes ...

Attribut : champ définissant une propriété. Il doit être défini sur exactement un seul domaine : les valeurs des attributs doivent être prises dans ce domaine. Par exemple, l'attribut NF dans la relation Fournisseur et l'attribut NF dans la relation Fourniture, sont tous deux définis sur le domaine des numéros de fournisseur.

Dans les langages de programmation, on défini un domaine par un type de données.

Relation : Une relation R, sur un ensemble de domaines D1,D2,..., Dn (non nécessairement distincts) est constituée de deux parties, un en-tête (schéma de la relation) et un corps (ensemble de lignes).

L'en-tête est un ensemble fixé d'attributs A1, A2,..., An , tel que chaque attribut Ai est défini sur un domaine Dj (j = 1,2, ..., n). Les noms des attributs A1 , A2 , ..., An sont tous distincts.

Le corps consiste en un sous-ensemble de tuples ou n-uplets.

Tuple : (appelé aussi **uplet** ou enregistrement) correspond à une ligne d'une relation

Cardinalité : c'est le nombre de tuples d'une relation.

Degré : c'est le nombre d'attributs d'une relation.

Remarque

Une relation correspond à une table. Un n-uplet correspond à une ligne d'une table et un attribut à une colonne.