CHAPITRE 1 INTRODUCTION GENERALE

I. NOTIONS INTUITIVES

II. OBJECTIFS ET AVANTAGES BD ET SGBD

III. NOTION DE MODELISATION DES DONNEES

IV. BREF HISTORIQUE, PRINCIPAUX SGBD COMMERCIALISES

I Notions intuitives

· Base de données

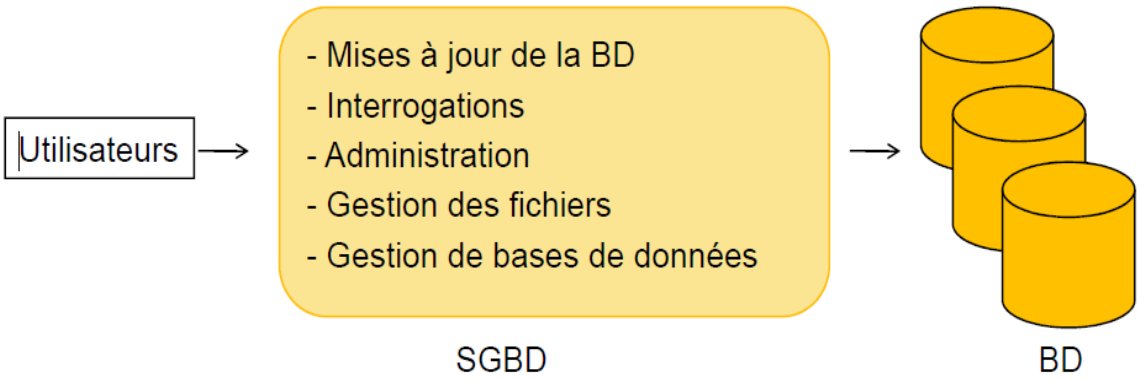
Ensemble structuré de données apparentées qui modélisent un univers réel. Une base de données est faite pour enregistrer des faits, des opérations au sein d'un organisme (administration, banque, université, hôpital, ...)

Une base de données est un ensemble structuré de données enregistrées dans un ordinateur et accessibles de façon sélective par plusieurs utilisateurs.

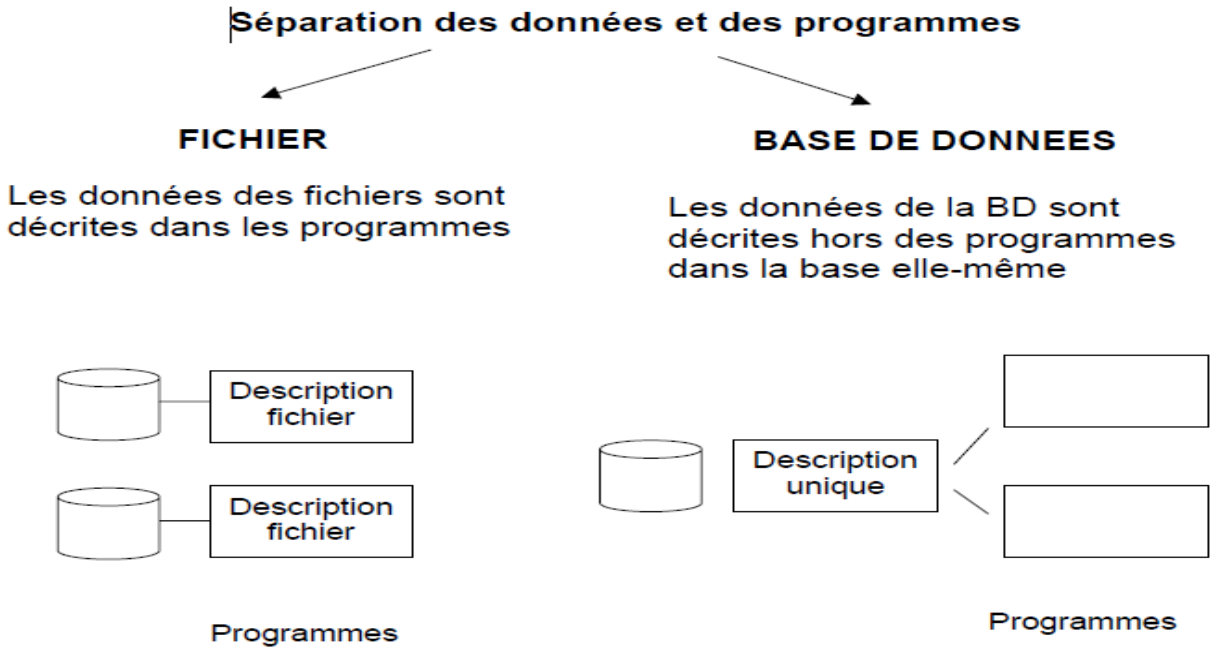
· Système de Gestion de Base de Données (SGBD) DATA BASE MANAGEMENT SYSTEM (DBMS)

Système qui permet de gérer une BD partagée par plusieurs utilisateurs simultanément.

Ensemble de logiciels systèmes permettant aux utilisateurs d'insérer, de modifier, et de rechercher efficacement des données spécifiques dans une grande masse d'informations (pouvant atteindre plusieurs milliards d'octets) partagée par de multiples utilisateurs. Un SGSBD est caractérise par un modèle de description des données.



Des fichiers aux Base de Données



La multiplication des fichiers entraînait la redondance des données, ce qui rendait difficile les mises à jour.

D'où l'idée *d'intégration* et de *partage* des données

II Objectifs et avantages des SGBD

Que doit permettre un SGBD ?

- **Décrire les données**

Indépendamment des applications (de manière intrinsèque)

Langage de définition des données
DATA DEFINITION LANGUAGE (DDL)

- **Manipuler les données**

Interroger et mettre à jour les données sans préciser d'algorithme d'accès

Langage de manipulation des données
DATA MANIPULATION LANGUAGE (DML)

- **Contrôler les données**

Intégrité des données (respect de contraintes qui peuvent être programmées)

Confidentialité des données (contrôle des droits d'accès, autorisation)

Langage de contrôle des données
DATA CONTROL LANGUAGE (DCL)

- **Partage**

Une BD est partagée entre plusieurs utilisateurs en même temps

Contrôle des accès concurrents

- **Sécurité**

Reprise après panne, journalisation

- **Performances d'accès**

Index

- **Indépendance physique**

Pouvoir modifier les structures de stockage ou les index sans que cela ait de répercussion au niveau des applications. Les disques, les méthodes d'accès, les modes de placement, le codage des données ne sont pas apparents

- **Indépendance logique**

Permettre aux différentes applications d'avoir des vues différentes des mêmes données. Permettre au DBA de modifier le schéma logique sans que cela ait de répercussion au niveau des applications.

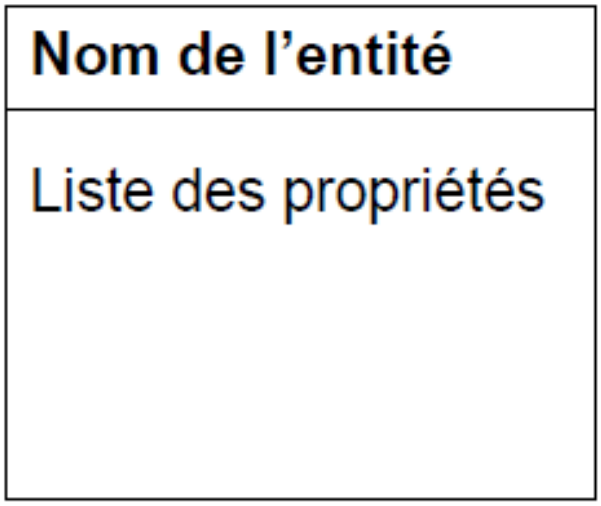
III Notion de modélisation des données

• Le modèle Entité-Association

EA en français, ER en anglais (pour Entity Relationship).Formalisme retenu par l'ISO pour décrire l'aspect conceptuel des données à l'aide d'*entités* et d'*associations*

• Le concept d'entité

Représentation d'un objet matériel ou immatériel
Par exemple un employé, un projet, un bulletin de paie



Les entités peuvent être regroupées en **types d'entités.** Par exemple, on peut considérer que tous les employés particuliers sont des **instances** du type d'entité générique EMPLOYE

Par exemple l'employé nommé DUPONT est une instance ou occurrence de l'entité EMPLOYE

- **Les propriétés**

Données élémentaires relatives à une entité

Par exemple, un numéro d'employé, une date de début de projet

• On ne considère que les propriétés qui intéressent un contexte particulier

• Les propriétés d'une entité sont également appelées des attributs, ou des caractéristiques de cette entité

- **L'identifiant**

Propriété ou groupe de propriétés qui sert à identifier une entité.

L'identifiant d'une entité est choisi par l'analyste de façon à ce que deux occurrences de cette entité ne puissent pas avoir le même identifiant.

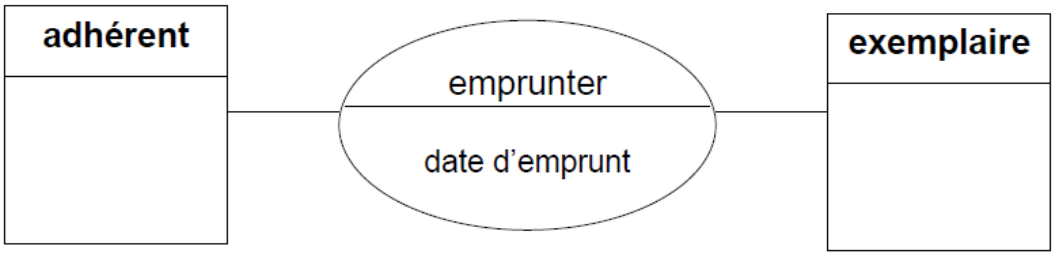
Par exemple, le numéro d'employé sera l'identifiant de l'entité EMPLOYE

- **Les associations**

Représentation d'un lien entre deux entités ou plus

• Une association peut avoir des propriétés particulières

Par exemple, la date d'emprunt d'un livre



- **Les cardinalités**

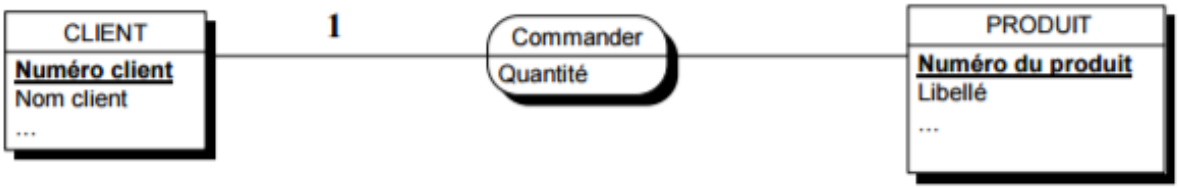
Les cardinalités sont des couples de valeur que l'on trouve entre chaque entité et ses associations liées. Donc, pour une association de 2 entités, il y a 4 cardinalités à indiquer (2 de chaque côté). Il y a trois valeurs typiques : 0, 1 et N (plusieurs). Pour les associations à 2 entités, ce sont des valeurs qui permettent d'indiquer combien de fois au minimum et au maximum une occurrence d'entité peut être liée à une autre occurrence d'entité.

De manière plus générale, les cardinalités d'une entité dans une association expriment le nombre de fois qu'une occurrence de cette entité peut être impliquée dans une occurrence de l'association, au minimum et au maximum.

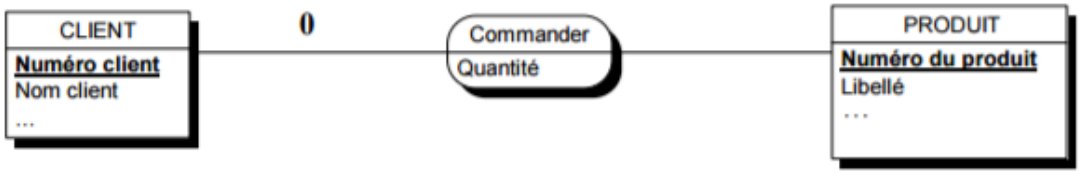
Les cardinalités traduisent des règles de gestion. Ce sont des règles propres à l'organisation étudiée, qui sont décidées par les gestionnaires et décideurs. Ces règles expriment des contraintes sur le modèle.

La cardinalité minimale

Elle est exprimée presque toujours par l'une des deux valeurs 0 ou 1. Elle traduit combien de fois au minimum une occurrence de l'entité participe à l'association, autrement dit, si une occurrence est obligatoirement associée à une autre ou pas. Exemple Pour la cardinalité minimale entre client et commander, il faut se poser la question : Pour un client donné, combien de fois au minimum il commande ? Ou encore mieux Est-il obligatoire qu'un client effectue une commande de produit ? Cela dépend des REGLES DE GESTION de l'entreprise. Si la règle de gestion est « tout client doit passer au moins une commande sinon ce n'est pas un client » on met la cardinalité mini à 1.



Mais on peut très bien imaginer que l'entreprise veut aussi mémoriser les clients potentiels (prospects), qui n'ont encore rien commandé. Dans ce cas, un client peut très bien ne pas avoir encore commandé, et on met la cardinalité mini à 0.



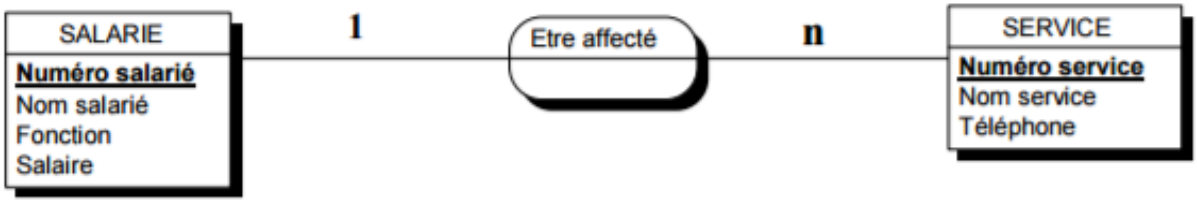
La cardinalité maximale

Elle traduit combien de fois au maximum une occurrence d'entité peut être en relation avec une occurrence de l'association. Cela peut être plusieurs fois (si c'est un nombre indéterminé, on indique la valeur n) ou une seule fois. Cette cardinalité répond à la question : la participation d'une occurrence doit-elle être unique ou bien peut-elle être multiple ? ou bien combien de fois au maximum une occurrence est elle impliquée dans l'association ? Si l'association est binaire (relie seulement deux entité), la question peut être aussi : Une occurrence de l'entité peut-elle être reliée à plusieurs occurrences de l'autre entité ou bien ne peut-elle être reliée qu'à une seule autre occurrence au plus?

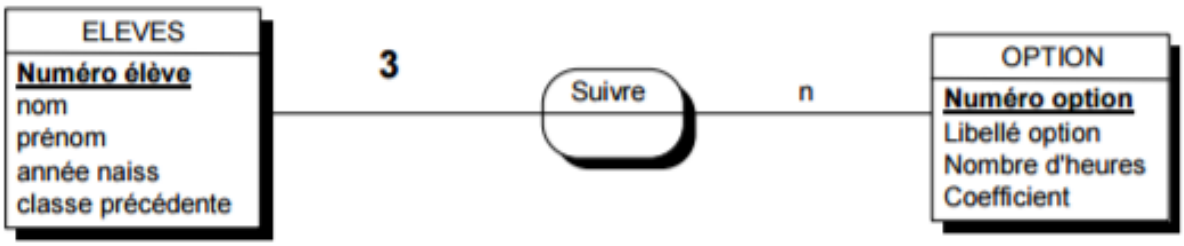
Si la réponse est « au plus une fois » (participation unique), la cardinalité maximale prend pour valeur 1. Si la réponse est « plusieurs » (participation multiple), la cardinalité maximale prend la valeur N.

Exemple RG (règles de gestion)

- Un salarié est affecté au plus à un seul service.
- Dans un service sont affectés plusieurs salariés



Il arrive (mais c'est rare) qu'une cardinalité maximale ait une valeur limitée. Exemple : RG : Un élève peut suivre au maximum 3 options.



Les différents types d'associations

- **Les associations binaires concernant 2 entités**

On distingue trois catégories d'associations en fonction des cardinalités maximales de ses branches:

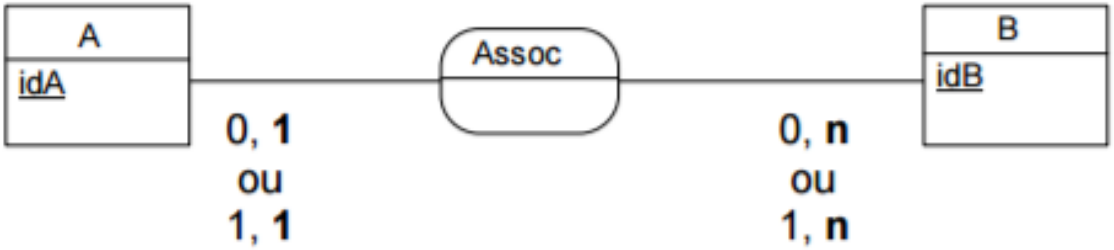
- les associations hiérarchiques encore appelées associations [1, n] ou associations fonctionnelles

- les associations non hiérarchiques, encore appelées associations [n, n] ou non fonctionnelles

- les associations [1, 1], les 2 branches ont pour cardinalité maximale 1. Ce cas est rare.

a)Les associations hiérarchiques [1,n]

Ce sont les associations où d'un côté la cardinalité maximale est à 1 et de l'autre côté la cardinalité maximale est à n.



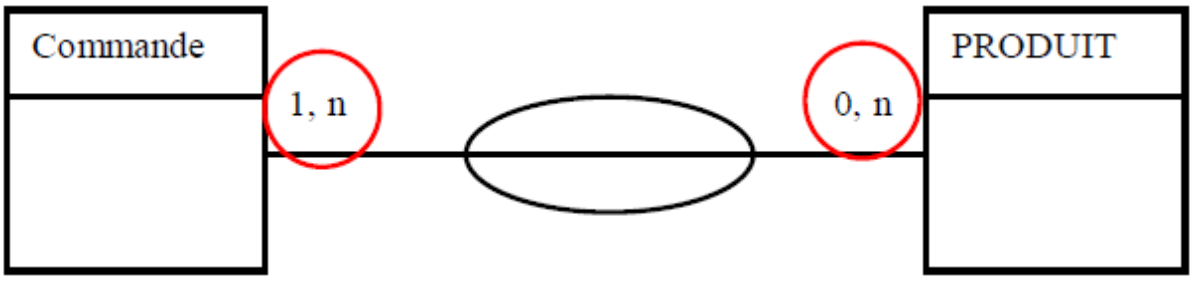
Cela signifie qu'une occurrence de A est reliée au plus à une seule occurrence de B. C'est-à-dire si on connaît une occurrence de A alors on saura forcément quelle est la seule occurrence de B qui correspond (si elle existe). On dit que A détermine B. C'est un lien de dépendance fonctionnelle. B dépend fonctionnellement de A.

L'entité qui correspond à la branche du côté du 1 est parfois appelée entité fils et l'entité correspondant à la branche du côté n est parfois appelée entité père. Cette appellation découle de l'analogie : un fils n'a qu'un seul père, et un père peut avoir plusieurs fils.

b) Les associations non hiérarchiques [n,n]

Toute association non hiérarchique (de type (0 ou 1-N) - (0 ou 1-N)) devient une relation. La clé primaire est formée par la concaténation (juxtaposition) l'ensemble des identifiants des entités reliées. Toutes les propriétés

éventuelles deviennent des attributs qui ne peuvent pas faire partie de la clé.



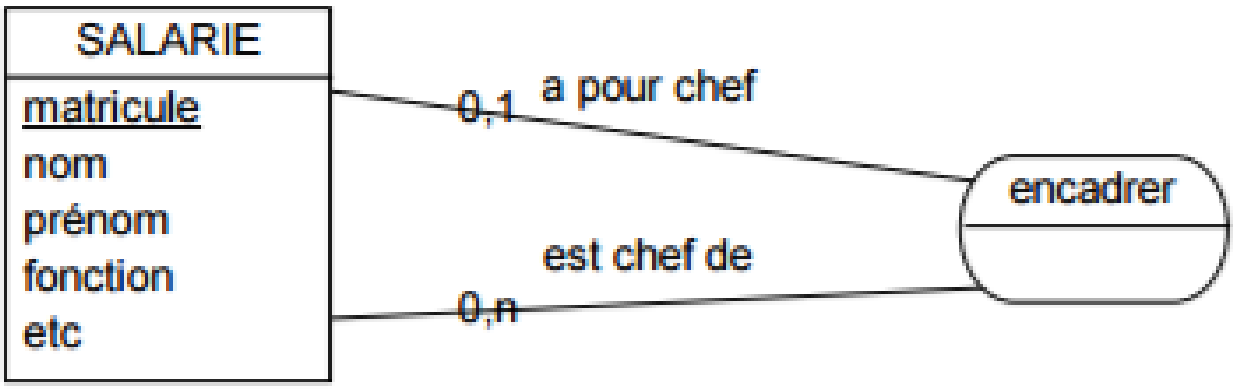
c) Les associations de type [1, 1]

Il s'agit des cas exceptionnel et très rare.

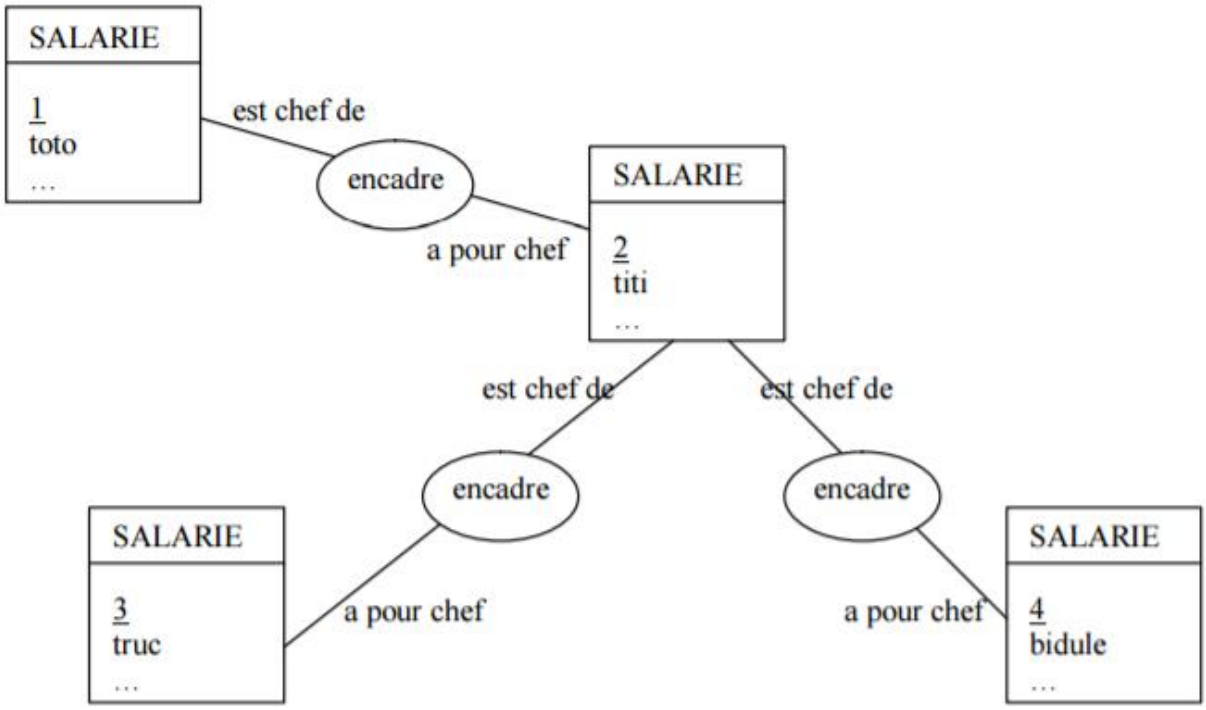
Les autres types d'association

- **Les associations réflexives**

Une association réflexive est une association reliant des occurrences de la même entité. Ces associations sont quasiment toujours binaire (2 branches).

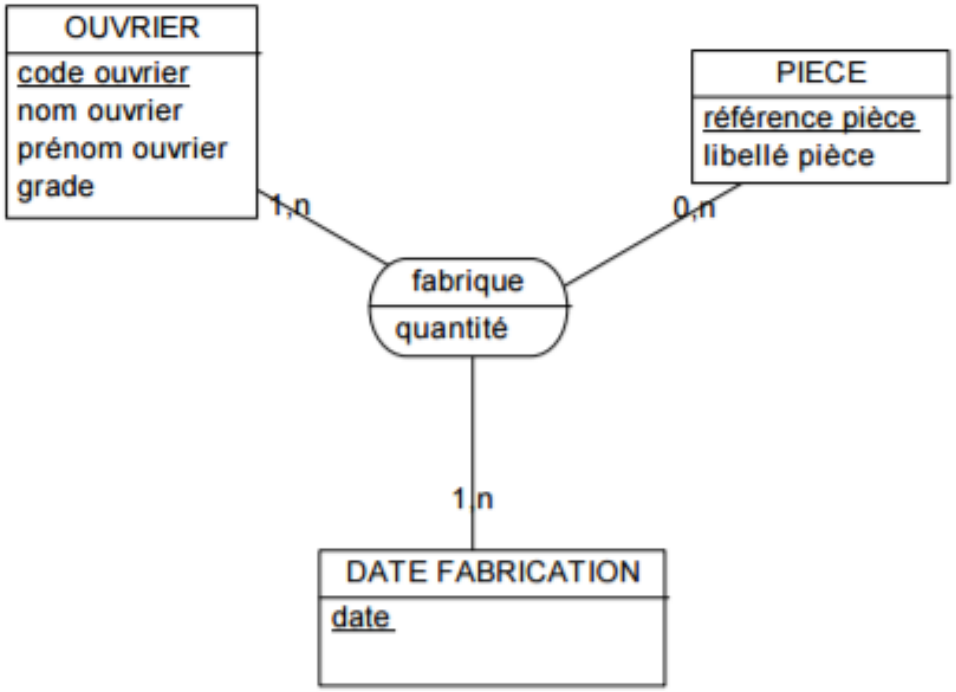


Exemples d'occurrences : diagramme d'occurrences Remarquez l'importance du rôle dans le diagramme d'occurrence



- **Les associations de dimension 3 ou plus (ternaires ou plus)**

Une association peut relier plus de 2 entités ensemble, le plus souvent trois. On parle alors d'association ternaire. On utilise une association ternaire quand on a besoin de connaître une occurrence de chaque entité pour avoir une information. Ex : Pour connaître la quantité de chacune des pièces fabriquée par chaque ouvrier à une date donnée, on utilise une association ternaire entre OUVRIER, PIECE et DATE. La quantité est une donnée portée par cette association.



Une occurrence de l'association fabrique implique une seule occurrence de chaque entité.

Exemple :

Ouvrier	Pièce	Date	quantité
Dupont	1	17/02	50
Dubois	2	17/02	100
Martin	1	17/02	40
Dupont	3	17/02	55
Dubois	3	17/02	20
Dupont	1	16/02	40
Martin	1	16/02	40

...

L'association ternaire implique aussi que : - Pour un ouvrier, on peut avoir plusieurs pièces différentes à la même date (il peut fabriquer plusieurs types de pièces le même jour). - Une pièce peut être fabriquée par plusieurs ouvriers différents le même jour. - A des dates différentes, un même ouvrier peut fabriquer les mêmes pièces. L'identifiant d'une association ternaire est formée de la concaténation (juxtaposition) des identifiants des 3 entités reliées. Dans notre exemple, on ne peut donc pas avoir plusieurs occurrences de "fabrique" qui concernent Dupont, la pièce n°1 à la date du 17/02.

IV Bref historique, principaux systèmes

Années 60 Premiers développements des BD

- Fichiers reliés par des pointeurs
- Systèmes IDS 1 et IMS 1 précurseurs des SGBD modernes

Années 70 Première génération de SGBD

- Apparition des premiers SGBD
- Séparation de la description des données de la manipulation de celles-ci par les applications
- Modéles hiérarchique et réseau CODASYL
- Langages d'accès navigationnels
- SGBD IDMS, IDS 2 et IMS 2

Années 80 Deuxième génération

- Modèle relationnel
- Les SGBDR représentent l'essentiel du marché BD (aujourd'hui)
- Architecture répartie client-serveur

Années 90 Troisième génération

- Modèles de données plus riches
- Systèmes à objets

OBJECTSTORE, O2

Principaux systèmes

- Oracle
- DB2 (IBM)
- Ingres
- Informix
- Sybase
- SQL Server (Microsoft)
- O2
- Gemstone

Sur micro

CHAPITRE 2 LE MODELE RELATIONNEL

I. LES CONCEPTS

II. LES DÉPENDANCES FONCTIONNELLES

III. LES RÈGLES D'INTÉGRITÉ

IV. LES FORMES NORMALES

I. Les Concepts

- **LE DOMAINE**

Ensemble de valeurs atomiques d'un certain type sémantique.

Ex. :

NOM_VILLE = { Cotonou, Parakou, Come}

• Les domaines sont les ensembles de valeurs possibles dans lesquels sont puisées les données

• Deux ensembles peuvent avoir les mêmes valeurs bien que sémantiquement distincts

Ex. :

NUM_ELV = { 1, 2, … , 2000 }

NUM_ANNEE = { 1, 2, … , 2000 }

- **LA RELATION**

Sous ensemble du produit cartésien de plusieurs domaines

$R \subset D1 \times D2 \times ... \times Dn$

D1, D2, ... , Dn sont les domaines de R
n est le degré ou l'arité de R

Ex.:
Les domaines :

NOM_ELV = { dupont, durant }

PREN_ELV = { pierre, paul, jacques }

DATE_NAISS = {Date entre 1/1/1990 et 31/12/2020}

NOM_SPORT = { judo, tennis, foot }

La relation ELEVE

ELEVE ⊂ NOM_ELV × PREN_ELV × DATE_NAISS

ELEVE = { (dupont, pierre, 1/1/1992),

(durant, jacques, 2/2/1994) }

La relation INSCRIPT

INSCRIPT ⊂ NOM_ELV × NOM_SPORT

INSCRIPT = { (dupont, judo), (dupont, foot),

(durant, judo) }

Définition d'une relation : R (A1 : D1, A2 : D2, … An : Dn), avec :
R = nom de la relation

Ai = noms des attributs

Di = domaines de définition des attributs (valeurs possibles)

n = cardinalité de la relation

Exemple :

PERSONNE (Nom : char(20), Prénom : char(20), Age : integer, Adresse : varchar(50), CP : integer, Ville : char(20))
Ce qui est écrit ci-dessus constitue en fait le schéma de la relation **PERSONNE**.
La relation **PERSONNE** est représentée sous la forme d'une table :

Nom	Prénom	Age	Adresse	CP	Ville
Dupont	Pierre	50	7, rue du Port	17000	La Rochelle
Martin	Alain	33	4, place de la Gare	87000	Limoges

- Une ligne de la table constitue un élément de la relation, ou n-uplet. Elle

Représente aussi une personne, instance de la relation PERSONNE. D'un point de vue logique (mathématique), il s'agit d'un prédicat qui met en relation les attributs de la relation.

- Il n'y a pas d'ordre sur les lignes (ni sur les colonnes) dans une table / relation.

- Il n'y a pas non plus d'information sur l'organisation physique (stockage des

données) qui est de ce fait cachée à l'utilisateur.

• LES ATTRIBUTS

Chaque composante d'une relation est un attribut

- Le nom donné à un attribut est porteur de sens
- Il est en général différent du nom de domaine
- Plusieurs attributs peuvent avoir le même domaine

• LE SCHÉMA D'UNE RELATION

Le schéma d'une relation est défini par :
- Le nom de la relation
- La liste de ses attributs

On note : R (A1, A2, ... , An)

Ex.:

ELEVE (NOM, PRENOM, NAISS)
INSCRIPT (NOM_ELV, SPORT)
TRAJET (VD, VA)

II. LES DÉPENDANCES FONCTIONNELLES

• Dépendance fonctionnelle

Principales contraintes d'intégrité formelles dans une Base de Données.

Une dépendance fonctionnelle (ou DF) indique une implication vérifiée universellement entre deux groupes d'attributs A et B : à une valeur pour A correspond toujours la même valeur de B.

● On la note A → B (A détermine B).

● A est parfois appelé « la source » et B « le but ».

● Soit R(A, B, C). L'attribut B est dit fonctionnellement dépendant de l'attribut A **si étant donné 2 n-uplets <a1, b1, c1> et <a2, b2, c2>**
a1 = a2 ⟹ b1=b2

● Ou encore, A détermine B si étant donné une valeur de A, il lui correspond une valeur unique de B.

• Système de règle d'inférences définit par Armstrong en 1974 :

• Réflexivité : $Y \subseteq X \Rightarrow X \rightarrow Y$ (ex. $A \rightarrow A$; $AB \rightarrow A$)

• Augmentation : $X \rightarrow Y \Rightarrow \forall Z, XZ \rightarrow YZ$ ou $XZ \rightarrow Y$

• Transitivité : $X \rightarrow Y, Y \rightarrow Z \Rightarrow X \rightarrow Z$

- Autres règles déduites des 3 premières
  - Pseudo-transitivité : $X \rightarrow Y, WY \rightarrow Z \Rightarrow WX \rightarrow Z$

    Démonstration

    Augmentation à $X \rightarrow Y \Rightarrow WX \rightarrow WY$

    Transitivité : $WX \rightarrow WY, WY \rightarrow Z \Rightarrow WX \rightarrow Z$

  - Union (ou composition) : $X \rightarrow Y, X \rightarrow Z \Rightarrow X \rightarrow YZ$

    Démonstration

    Augmentation à $X \rightarrow Y \Rightarrow X \rightarrow YX$

    Augmentation à $X \rightarrow Z \Rightarrow XY \rightarrow ZY$

    Transitivité : $X \rightarrow YX, XY \rightarrow ZY \Rightarrow X \rightarrow ZY$

- Autres règles déduites des 3 premières
  - Décomposition : $X \rightarrow Y$ et $Z \subseteq Y \Rightarrow X \rightarrow Z$

    Démonstration

    Réflexivité à $Z \subseteq Y \Rightarrow Y \rightarrow Z$

    Transitivité : $X \rightarrow Y, Y \rightarrow Z \Rightarrow X \rightarrow Z$

    Exemples : $AB \rightarrow CD \Rightarrow AB \rightarrow C$ et $AB \rightarrow D$

✓La clé d'une relation

Attribut (ou groupe minimum d'attributs) qui détermine tous les autres.

Ex.:

PRODUIT (**no_prod**, nom, prixUHT)

no_prod → (nom, prixUHT)

no_prod est une clé

• Une clé détermine un n-uplet de façon unique

• Pour trouver la clé d'une relation, il faut examiner attentivement les hypothèses sur le monde réel

• Une relation peut posséder plusieurs clés, on les appelle clés candidates

✓Clé primaire

Choix d'une clé parmi les clés candidates

✓Clé étrangère ou clé secondaire

Attribut (ou groupe d'attributs) qui fait référence à la clé primaire d'une autre relation

Ex.:

CATEG (**no_cat**, design, tva)

PRODUIT(no_prod, nom, marque, no_cat, prixUHT)

no_cat dans PRODUIT est une clé étrangère

CLÉ ÉTRANGÈRE = CLÉ PRIMAIRE dans une autre relation

III LES RÈGLES D'INTÉGRITÉ

Les règles d'intégrité sont les assertions qui doivent être vérifiées par les données contenues dans une base. Le modèle relationnel impose les contraintes structurelles suivantes :

- **INTÉGRITÉ DE DOMAINE**
- **INTÉGRITÉ DE CLÉ**
- **INTÉGRITÉ RÉFÉRENCIELLE**

INTÉGRITÉ DE DOMAINE

Les valeurs d'une colonne de relation doivent appartenir au domaine correspondant

INTÉGRITÉ DE CLÉ

Les valeurs de clés primaires doivent être :
- uniques
- non NULL

INTÉGRITÉ RÉFÉRENCIELLE

Les valeurs de clés étrangères sont 'NULL' ou sont des valeurs de la clé primaire auxquelles elles font référence.

IV. LES FORMES NORMALES

Différentes formes de normalisation de la relation universelle, le plus souvent obtenues par décomposition, afin d'obtenir un schéma de base de données qui soit

Sans redondance,
Sans anomalies de mise à jour,
Sans perte de données,
Sans perte de DF.

✓ 1ère Forme Normale 1FN

Une relation est en 1FN si chacun des attributs ne prend ses valeurs que dans un domaine constitué de valeurs élémentaires (i.e. atomiques). La 1FN consiste à éviter les domaines composés de plusieurs valeurs (pas de données composées, pas de listes).

Contre-exemple
ELEVE (**no_elv**, nom, prenom, liste_notes)
Un attribut ne peut pas être un ensemble de valeurs
Décomposition
ELEVE (**no_elv**, nom, prenom)
NOTE (no_elv, no_matiere, note)

✓ 2ème Forme Normale 2FN

Un schéma de relation est en 2FN si et seulement si :
● il est en 1FN,
● il n'admet pas de dépendance de clé partielle, c'est à dire une DF d'une partie stricte d'une clé vers des attributs non clés.

• C'est la phase d'identification des clés

• Cette étape évite certaines redondances

• Tout attribut doit dépendre fonctionnellement de la totalité de la clé

Contre-exemple
une relation en 1FN qui n'est pas en 2FN
COMMANDE (**date**, **no_cli**, **no_pro**, qte, prixUHT)
elle n'est pas en 2FN car la clé **= (date, no_cli, no_pro)**, et le prixUHT ne dépend que de no_pro
Décomposition
COMMANDE (**date**, **no_cli**, **no_pro**, qte)
PRODUIT (**no_pro**, prixUHT)

✓ 3ème Forme Normale de BOYCE-CODD BCNF

Un schéma de relation est en 3FN ssi :
● il est en 2FN,
● chaque attribut non clé est pleinement et directement dépendant des clés.
ou
● tout attribut n'appartenant pas à une clé ne dépend pas d'un attribut non clé.
ou encore
● il n'admet pas de DF transitive, c'est-à-dire d'un ensemble d'attributs non inclus dans une clé vers un autre ensemble d'attributs non (sur-)clé. Toutes les DF des clés vers les attributs non clés sont élémentaires et directes.

Contre-exemple
une relation en 2FN qui n'est pas en 3FN
VOITURE (**matricule**, marque, modèle, puissance)

on vérifie qu'elle est en 2FN ; elle n'est pas en 3FN car la clé = **matricule**, et la puissance dépend de (marque, modèle)

Décomposition

VOITURE (**matricule**, marque, modèle)

MODELE (**marque, modèle**, puissance)