CHAPITRE I : INTRODUCTION AUX BASES DE DONNÉES

**Les systèmes à base de fichiers**

• Programmes d’applications qui rendent un service et se basent sur des fichiers qui leurs sont propres

• Il s’agit d’une approche décentralisée.

**Limites des systèmes à base de fichiers**

• Séparation et isolation des données

• Chaque programme maintient son ensemble de données

• Un programme ne connait pas les données potentiellement intéressantes d’un autre programme

• Duplication des données

• Les mêmes données seront utilisées par des programmes différents

• Perte d’espace

• Valeurs différentes ou formats différents pour une même donnée

• Requêtes figées /Prolifération de programmes

• Les programmes sont écrits pour satisfaire des fonctions particulières

• Chaque spécification nouvelle nécessite un nouveau programme

• Aucune sécurité

• Aucun contrôle d’intégrité des données

• Pas de contrôle des accès concurrents

**SGBD**

• Un système de gestion de bases de données (SGBD) est un logiciel qui permet de stocker et de gérer des bases de données

• Un SGBD fournit des capacités de stockage et d’accès aux données persistantes

• Un SGBD permet à plusieurs utilisateurs d'accéder (en lecture et écriture) aux données en même temps, tout en préservant l'intégrité de ces données

• Un SGBD assure la sécurité et la confidentialité des données.

• Un SGBD est robuste aux pannes logicielles, matérielles ou externes (telles que des pannes d’électricité)

**Qu’est-ce qu’une BD?**

• Base de données (BD) : collection de données structurées de manière à être exploitées pour répondre efficacement à des requêtes.

• On représente des entités (exemple: Etudiants, Cours) et leurs attributs (exemple: nom, date de naissance, description)

• On représente les relations entre les entités (exemple: Un étudiant donné suit certains cours)

**Concepts clé**

• La notion de schéma d’une BD, c’est-à dire la structure de données, d’entités que l’on va définir. On parle de l’intention d’une base de données. Etudiant(Id, nom, etc.)

• La notion d’instances du schéma, c’est-à-dire les données dans votre BD. On parle aussi de l’extension d’une base de données. ID001, Simon Diamond

**Notion de requêtes à un SGBD**

• Une requête est une question au sujet des données stockées.

• Les SGBD offrent un langage de requêtes pour pouvoir lancer des requêtes à une base de données.

• Dans le modèle relationnel, le langage s’appelle SQL

**Avantages des BD**

• Stockage de données avec contrôle de la redondance

• Intégrité et consistance des données

• Structures de données et mécanismes d’accès pour des requêtes efficaces

• Concurrence d’accès

• Services de sauvegarde et de recouvrement en cas de panne ou de corruption de données

• Sécurité et confidentialité pour les données

**L’architecture à 3 couches ANSI-SPARC**

* L’architecture de la plupart des SGBD commerciaux est basée sur le modèle ANSI-SPARC (1975).

◦ ANSI: American National Standards Institute

◦ SPARC: Standards Planning and Requirements Committee

Cette architecture stipule qu’un SGBD doit être modélisé en 3 couches. Cette séparation en trois facilite la maintenance du SGBD et des BDs.

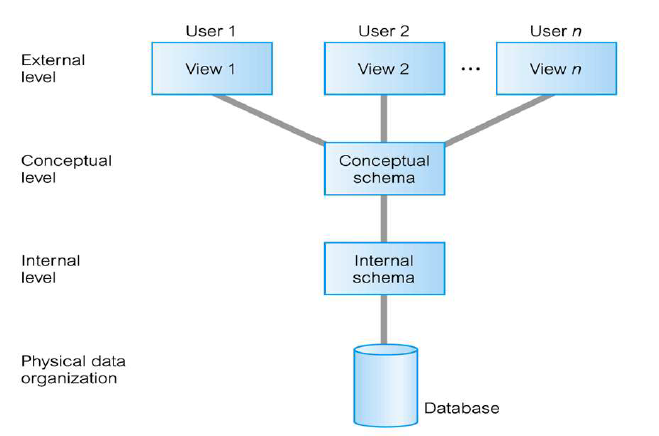
1. Couche externe: décrit les vues des utilisateurs (pour une classe d'utilisateurs, on définit et regroupe les entités désirées).

2. Couche conceptuelle : décrit toutes les entités, les relations, les opérations, les contraintes;

3. Couche interne : stockage physique des données

◦ Décrit comment les données sont stockées, via quelles techniques de compression, quels indexes ou tables de hachage, etc.

◦ En dessous du niveau interne, on retrouve le système d’exploitation et le matériel (niveau physique)



**Méthodologie de conception d’une base de données**

3 phases principales

• Modèle conceptuel (Conceptual database design) : Construction du modèle de données utilisées par l’entreprise à partir des spécifications usagers, sans considérer aucun aspect physique

• Modèle logique (Logical database design) :

• Processus de conversion du modèle conceptuel à un modèle logique spécifique comme le modèle relationnel.

• On demeure indépendant de toute considération physique ou d’un SGBD particulier.

• Le schéma obtenu passera ensuite par un processus de normalisation avant d’être converti en modèle physique

• Modèle physique (Physical database design) :

• Description de l’implantation physique de la BD sur un support de stockage secondaire externe

• Le modèle physique décrit les organisations de fichiers, les indexes les plus à même de gérer efficacement les données, ainsi que les contraintes d’intégrité des données

• Ce niveau est dépendant d’un SGBD spécifique.

**LA CONCEPTION D’UNE BD AVEC LE MODÈLE ENTITÉ ASSOCIATION**

**Modélisation conceptuelle d’une BD**

• Processus de construction d'un modèle abstrait de données utilisées dans une entreprise, indépendamment de toutes les considérations physiques.

• Le modèle de données est construit en utilisant les informations contenues dans le cahier de charges des utilisateurs. On parle d’analyse des prérequis.

• Dans les BD, un cahier de charge spécifie les données que l’on veut stocker ainsi que les contraintes sur les données.

Le but de la modélisation ER est de créer un reflet fidèle des données de l’entreprise dans une base de données.

Le modèle ER ne nous donne pas vraiment une description de la base de données. Il nous donne un modèle intermédiaire à partir duquel il est facile de définir une base de données.

**Modèle Entité Association (EA / ER)**

• ER model: Entity Relationship model

• Les modèles EA sont représentés graphiquement.

• Il existe plusieurs notations graphiques: notation UML, notation de Chen, Notation Crow’s foot, Bachman,

IDEF1X etc.

• Dans la notation UML:

• On représente les entités par des rectangles ou boites et les associations par des losanges ou par des lignes simples

• Les attributs sont définis dans les entités

**Entités**

• Groupe d’objets ayant les mêmes propriétés et identifiés par l’entreprise comme ayant une existence indépendante, Etudiant, Professeur

• On distingue les entités ayant une existence physique (Ex: Etudiant) de celle ayant une existence conceptuelle (exemple: Vente)

• Une occurrence ou une instance d’entité : Un objet identifiable de manière unique, Exemple: (001, Amal Zouaq) est une instance de l’entité Professeur

• Une entité forte est une entité dont l’existence ne dépend pas d’une autre entité et qui a sa propre clé primaire, Ex: Étudiant

• Une entité faible est une entité dont l’existence dépend d’une autre entité (forte).

**Attributs**

• Propriétés d’une entité (ou d’une association): décrit ses caractéristiques, Nom, adresse…

• Domaine d’un attribut: ensemble des valeurs possibles d’un attribut, Ville: ‘Kingston’, ‘London’, ‘Aberdeen’

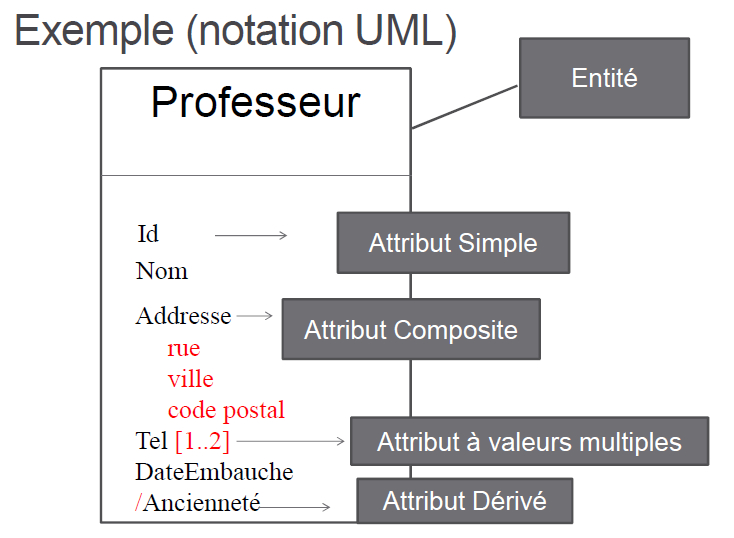
• Attribut simple: Attribut constitué d’un seul composant, Ex: nom, salaire

• Attribut composite: Attribut constitué de plusieurs composants, chacun ayant une existence indépendante, Ex: Adresse peut être décomposée en rue, ville, code postal

• Attribut à valeur simple (Single-valued Attribute): Attribut qui n’a qu’une seule valeur possible pour chaque occurrence de l’entité, Ex: nom

• Attribut à valeurs multiples (Multi-valued Attribute): Attribut qui a plusieurs valeurs possibles pour chaque occurrence de l’entité, Ex: Tel

• Attribut dérivé : représente une valeur qui est dérivable (calculable) de la valeur d’un autre attribut ou d’un ensemble d’attributs, et non nécessairement dans la même entité, Ex: durée peut être calculée à partir de dateFrom et DateTo



**Clés**

* Chaque occurrence d’entité doit être identifiable de manière unique via un ou plusieurs attributs que l’on appelle une clé, Exemple: pmatricule d’un étudiant
* Une clé est un ensemble d’attributs dont les valeurs identifient exactement une occurrence dans l’ensemble des occurrences/instances de l’entité. Une clé doit avoir:
  + une propriété d’unicité: Il ne peut pas y avoir deux instances de l’entité qui ont la même clé
  + une propriété de minimalité: la clé contient uniquement les attributs nécessaires pour respecter la propriété d’unicité
  + La propriété d’unicité ne doit pas varier dans le temps!, Exemple : Nom
  + Plusieurs types de clés existent qui doivent respecter les propriétés d’unicité et de minimalité:
  + Clé candidate : Un ensemble minimal d’attributs qui identifient de manière unique chaque occurrence de l’entité, Etudiant: matricule, NAS
  + Clé primaire : Une clé candidate choisie pour identifier de manière unique chaque occurrence de l’entité, Etudiant: Matricule
  + Clé composite : Une clé candidate qui consiste en plus d’un attribut, Achat : IdProduit, NumClient, DateAchat
  + Superclés (superkeys) : Un ensemble d’attributs d’une entité est une superclé si pour chaque instance de cette entité, l’ensemble des valeurs de ces attributs identifient l’instance de manière unique. On a la propriété d’unicité mais pas nécessairement de minimalité. Une clé candidate est une superclé mais pas l’inverse!
  + Dans un diagramme UML, on peut souligner les attributs qui indiquent une clé primaire ou utiliser la notation PK (Primary Key)

• ATTENTION: une entité ne peut pas être un attribut d’une autre entité!

• Lorsque vous vous surprenez à vouloir mettre une entité à l’intérieur d’une autre entité, vous devez plutôt spécifier une association!

Clés et entités faibles

• Dans un modèle conceptuel, on n’indique pas de clé primaire pour une entité faible

• Une entité faible peut être identifiée uniquement en fonction de la clé primaire de l’entité forte à laquelle elle est associée (combinée avec un ou plusieurs de ses attributs).

• On parle de clé partielle pour l’entité faible qui est soulignée par un trait non continu --------

• L’entité forte est alors dite propriétaire et identifiante.

• Les entités faibles doivent avoir une participation totale dans l’association qui les relie à l’entité propriétaire. On parle alors d’association identifiante.

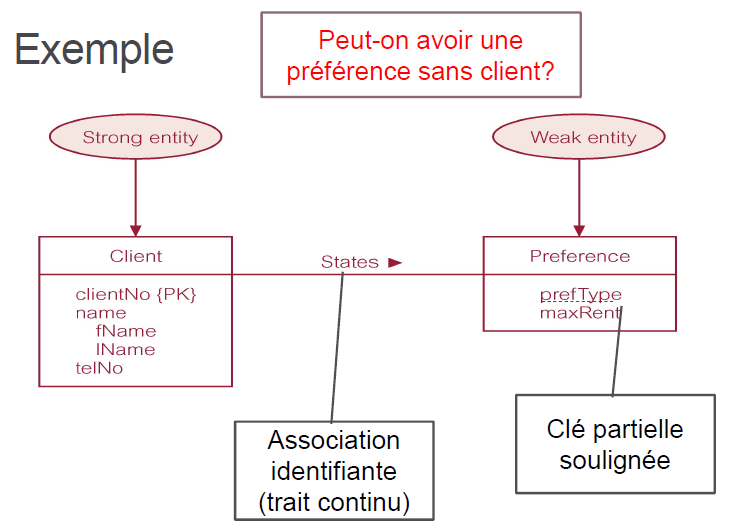
• En UML (Microsoft Visio), les associations identifiantes sont représentées avec un trait continu

**Associations**

• Une association capture comment les entités sont reliées les unes aux autres

• On peut penser à des associations comme à des verbes, qui relient deux entités ou plus, Ex: un professeur appartient à un département

• Une occurrence d’association est une occurrence identifiable de manière unique, qui contient une occurrence de chacune des entités qui participent à l’association (p012689, GIGL)



• Degré d’une association: Nombre d’entités qui participent dans l’association

• Une association de degré:

• Un est reflexive ou récursive

• Deux est binaire (binary)

• Trois est ternaire (ternary)

• Plusieurs est n-aires (n-ary)

• Association réflexive: association où la même occurrence d’entité participe plus d’une fois dans des rôles différents

• E1 supervise E2

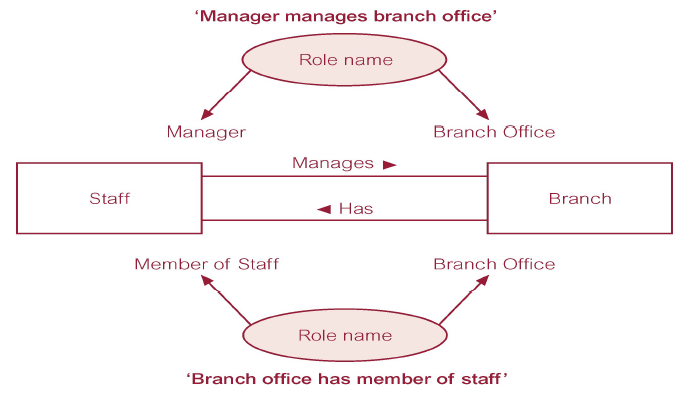
• E1 est supervisé par E3

• On peut assigner des noms de rôles aux deux bouts de l’association réflexive

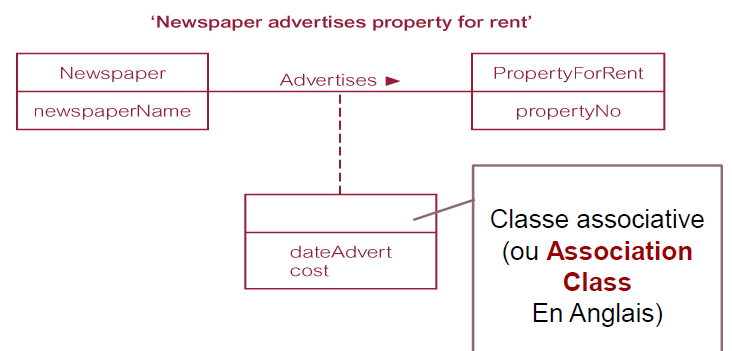
• Une même entité peut participer à des associations différentes, ou à différents “rôles” dans la même association

• Des indicateurs de rôle sont ajoutés au diagramme ER

• Deux associations entre la même paire d’entités et rôles :



• Une association peut aussi avoir des attributs qui caractérisent l’association et non pas les entités qui y participent



• Les associations ont des contraintes qui représentent des règles d’entreprise établies par les usagers ou l’entreprise.

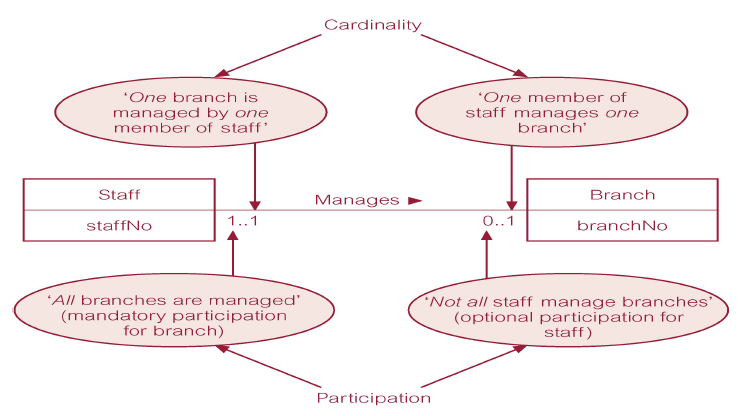
• La principale contrainte est la notion de multiplicité: le nombre des occurrences possibles d’une entité qui peut être reliée à une autre entité par l’association.

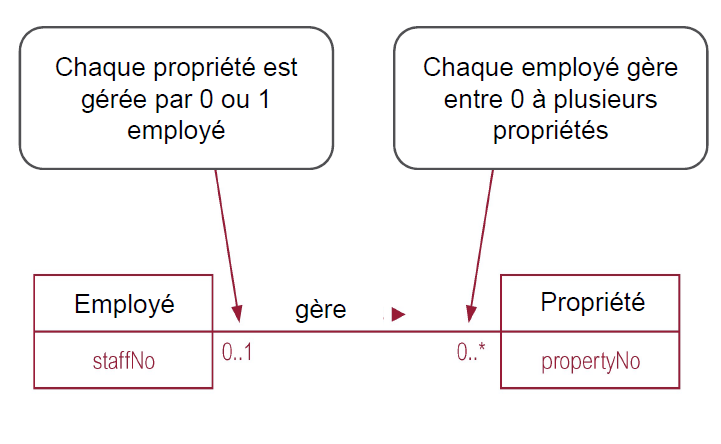
**Multiplicité**

• La multiplicité est composée de deux restrictions sur les associations: la participation et la cardinalité

• La participation décrit si une partie ou toutes les occurrences de l’entité participent à l’association. On parle alors de participation partielle versus participation totale

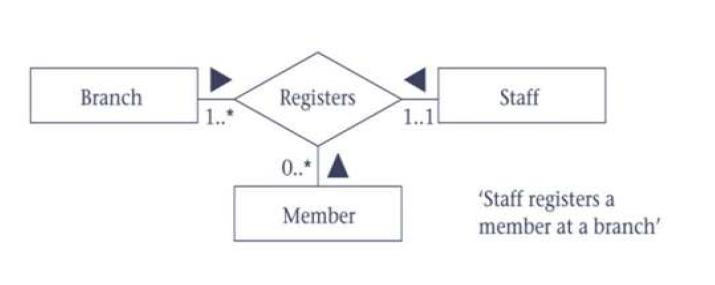
• La cardinalité décrit le nombre maximal possible d’occurrences de l’association pour l’entité qui y participe





Multiplicité des associations complexes

• Le nombre des occurrences possibles d’une des entités d’une association n-aire lorsque les (n-1) autres valeurs sont fixées



• Un membre du personnel enregistre un client à une succursale donnée et participation totale (1..1), Un client peut être enregistré par un membre du personnel dans plusieurs une succursales et participation partielle (0..\*), Dans une succursale plusieurs membres du personnel peuvent enregistrer plusieurs clients et participation totale (1..\*).

**Bonnes pratiques de modélisation**

• Être fidèles aux spécifications de l’application/usager

• Utiliser des attributs si possible

• Éviter la duplication de données et les informations redondantes sauf pour des raisons de performance (redondance contrôlée)

**Remplacer les associations n-aires par des entités faibles**

• On peut remplacer une association n-aire (n>2) par une entité de la manière suivante :

• On remplace le type association n-aire par une entité faible

• On crée des associations binaires identifiantes entre la nouvelle entité et toutes les entités de l’association n-aire.

• La cardinalité de chacune des associations binaires créées est 1..1 du côté des entités propriétaires et 0..n ou

1..n du côté de la nouvelle entité créée (qui remplace l’association n-aire)

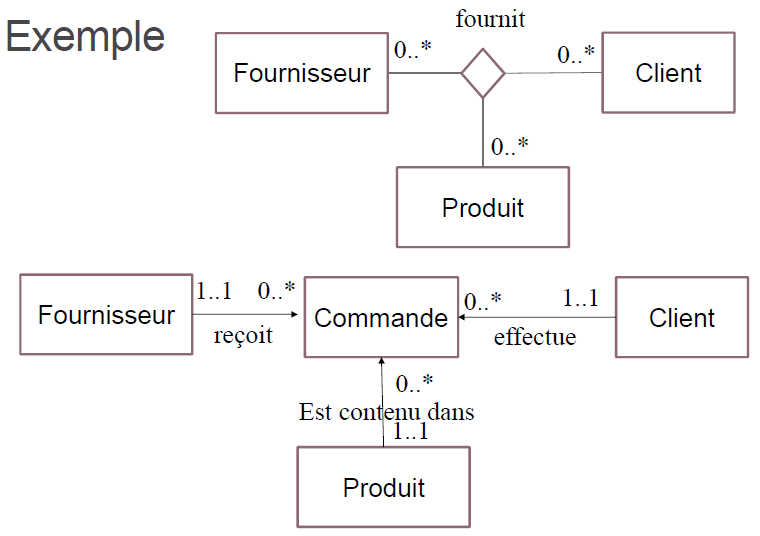
**Remplacer les associations n-aires par des entités fortes**

• Alternativement, on peut remplacer l’association n-aire par une entité forte en lui ajoutant une clé artificielle

• On crée des associations binaires entre la nouvelle entité et toutes les entités de l’association n-aire.

• La cardinalité de chacune des associations binaires créées est 1..1 du côté des entités propriétaires et 0..n ou 1..n du côté de la nouvelle entité créée (qui remplace l’association n-aire)

• Le choix entre une entité forte ou faible dépend de la situation à modéliser.



CHAPITRE II : CONCEPTION D’UNE BD ET MODÈLE RELATIONNEL

**Le modèle EA Etendu (EER)**

• On rajoute des concepts sémantiques avec notamment:

• Le concept de spécialisation/généralisation (Héritage, is-a)

• Le concept d’agrégation/de composition (part of)

**Spécialisation / Généralisation (Hiérarchie ISA)**

• A ISA B signifie que chaque occurrence d’une entité A est aussi une occurrence de l’entité B.

• B est appelée une super-classe et A une sous-classe

• Super-classe: Une entité (ex: Compte) qui inclut un ou plusieurs sous-groupes (Ex: chèque, épargne) d’occurrences

• Sous-classe: un sous-groupe distinct d’occurrences d’une entité donnée

• L’association entre une super-classe et une sous-classe est une association 1..1

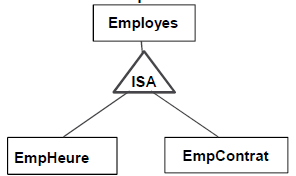
• Une sous-classe possède généralement des attributs spécifiques en plus de ceux de la super-classe.

• Les attributs du parents sont des attributs hérités.

• Deux contraintes s’appliquent à une relation de spécialisation/généralisation :

• Une contrainte de couverture: Elle détermine si chaque occurrence de la super-classe doit participer comme occurrence d’une sous-classe. Elle peut avoir la valeur obligatoire (mandatory) ou optionnelle (optional). Les occurrence d’une superclasse ne sont pas forcément des occurrences d’une sous-classe. Y a-t-il des employés qui ne sont ni dans EmpHeure ni dans EmpContrat?

• Une contrainte de superposition : Décrit la relation entre les membres des sous-classes et indique si un même membre d'une superclasse peut être un membre de plus d'une sous-classe. Elle peut avoir la valeur disjointe (Or) ou non-disjointe (And). Les sous-classes peuvent être disjointes ou non. Patrick peut-il être à la fois dans EmpHeure et dans EmpContrat?



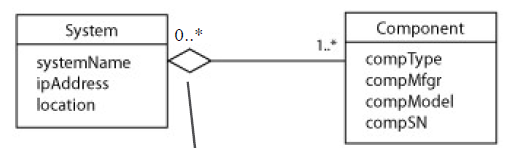
**Agrégation / Composition**

• En UML, ce sont des associations qui relient des entités qui représentent des parties et une entité englobante (un tout). Part-whole relations

• Agrégation : représente une relation 'HAS-A' ou 'ISPART-OF‘ entre des entités. Les entités “parties” peuvent changer d’entité englobante ou exister indépendamment de cette entité

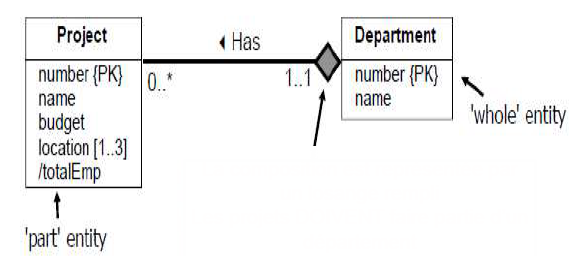
• On dit que la participation de l’entité “composante” dans une agrégation est partielle

• Le losange vide est le symbole de l’agrégation

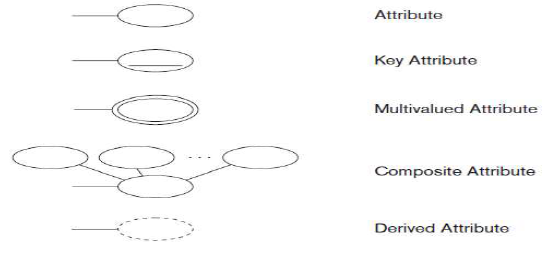


• Composition: c’est une forme plus forte d’agrégation où la partie ne peut exister sans son entité englobante.

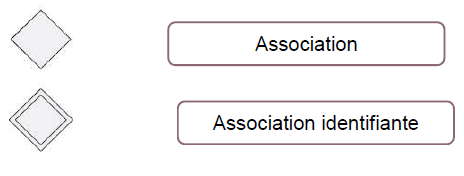
La composition est représentée par un losange rempli. Les projets DOIVENT faire partie d’un département

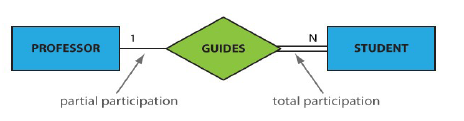
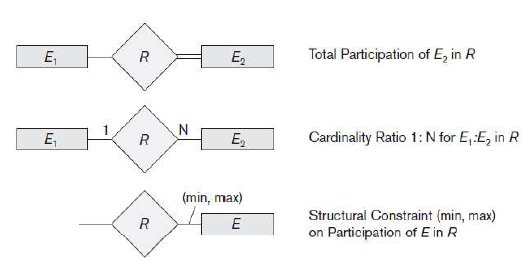


**Notation de CHEN**



Entité faible et clé partielle : On indique la clé partielle d’une entité faible par un trait discontinu sous l’attribut concerné.

Une association identifiante représentée par un losange double relie une entité faible à une entité forte.



• On peut représenter les multiplicités de différentes manières dans la notation de Chen

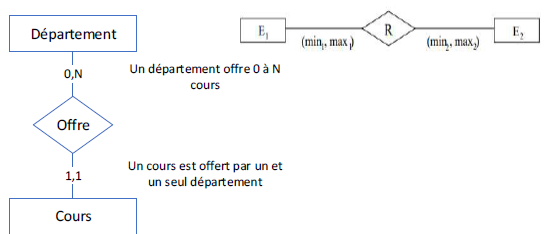
• Ici seule la cardinalité des relations est indiquée par un chiffre, la participation est indiquée visuellement.

• Tous les étudiants sont guidés par un professeur. Leur participation à l’association est totale.

• Un professeur peut ne guider aucun étudiant. La participation

**Notation (min, max)**

On peut aussi indiquer la multiplicité, c’est-à dire la participation et la cardinalité (min, max) mais dans ce cas attention: la multiplicité se trouve alors du même côté que l’entité à laquelle elle réfère



**Le Modèle Relationnel**

C’est un modèle où toutes les données prennent la forme de relations (tables). Il est constitué de : Relations, Clés, Contraintes d’intégrité et les vues.

**Relation**

• Une relation est un ensemble non ordonné qui regroupe des instances ayant des propriétés communes

• Une relation est une table avec des colonnes et des lignes (structure logique). Elle correspond souvent à une entité du schéma conceptuel

• Un tuple est une instance de la relation qui est physiquement stockée dans la BD.

• Le degré d’une relation est son nombre d’attributs (Ne pas confondre avec le degré d’une association dans le modèle conceptuel)

• La cardinalité d’une relation est le nombre de tuples d’une relation

• Un attribut est une colonne de la relation

• Le domaine d’un attribut est l’ensemble des valeurs possibles de l’attribut

• Formellement, une relation r(R) est une relation mathématique de degré n sur les domaines dom(A1),…, dom(An), qui est un sous-ensemble du produit cartésien des domaines qui définissent R:

• (𝑅) ⊑ (𝑑𝑜𝑚 𝐴1 𝑋 𝑑𝑜𝑚 𝐴2 𝑋 … 𝑋 𝑑𝑜𝑚 )

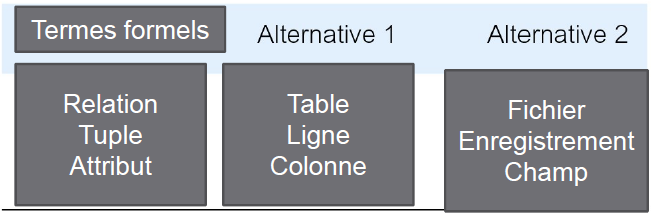
• Le nom d’une relation est distinct des autres relations dans une même BD

• Chaque cellule d’une relation contient exactement une valeur atomique ( Les attributs composites et multi-valués du modèle conceptuel ne sont pas acceptés)

• Chaque tuple est distinct; il n’existe pas de tuples dupliqués

• L’ordre des attributs n’est pas important en autant que l’on conserve la correspondance entre les attributs et leurs valeurs

• Comme une relation est définie comme un ensemble de tuples, théoriquement, l’ordre des tuples n’a pas d’importance (Toutefois ils sont stockés physiquement dans un ordre donné)



**Clés relationnelles**

• Super clé (Superkey)

• Clé candidate

• Clé primaire

• Clé alternative (Alternate Key)

• Clé étrangère (Foreign Key) ( Attribut ou ensemble d’attributs dans une relation qui correspond à la clé primaire d’une autre relation. Elle permet de représenter les associations entre deux entités dans le modèle conceptuel!)

**Contraintes d’intégrité**

• Une contrainte d’intégrité est une condition qui doit être satisfaite par toutes les instances stockées dans la base de données. Exemple: GPA>=0 et GPA<=4.3

• On spécifie les contraintes d’intégrité lors de la création du schéma de la base de données. On les vérifie lorsqu’une modification de la BD est effectuée

• Ces contraintes permettent d’éviter des erreurs lors de l’insertion et de la mise à jour des données et donc d’imposer la consistance et la cohérence des données

Contraintes du domaine

• Chaque attribut dans une relation a un nom distinct

• Nous avons déjà vu que les attributs sont associés à des types de données et à un domaine de valeurs ( NAS. L’ensemble des nombres de 9 chiffres valides qui représentent le numéro d’assurance sociale au Canada.

• Un domaine D est un ensemble de valeurs atomiques

• Les contraintes du domaine spécifient que dans chaque

tuple la valeur de chaque attribut A doit être une valeur

atomique du domaine dom(A)

Contraintes de clé

• Il ne peut pas exister deux tuples qui ont la même combinaison de valeurs pour un sous-ensemble d’attributs que l’on appelle une super-clé

• Une super-clé doit avoir la propriété d’unicité

• Une super-clé minimale est une clé (primaire)

• Une contrainte de clé définit qu’une clé (primaire) doit avoir les propriété de minimalité et d’unicité. Elle sert à identifier de manière unique chaque tuple de la relation

Contraintes d’intégrité : La contrainte sur les valeurs Null

• Elle représente la valeur d’un attribut inconnu ou non applicable au tuple courant

• Permet de représenter les données incomplètes ou exceptionnelles. Cela n’est pas la même chose que zéro ou espace!

• On peut contraindre un attribut à ne pas être null avec NOT NULL. Par exemple: un nom ne peut pas être “null”

Contraintes d’intégrité d’entité

• Intégrité d’entité (Entity Integrity): dans une relation, aucun attribut de la clé primaire ne peut avoir la valeur

null

Contraintes d’intégrité : Intégrité référentielle

• Intégrité des références ou référentielle: elle est spécifiée entre deux relations.

• Elle établit qu’un tuple dans une relation qui référe à une autre relation doit spécifier un tuple existant dans cette relation ou une valeur nulle

• L’intégrité référentielle se base sur la notion de clé étrangère (Foreign Key = FK).

Clé étrangère

• Un ensemble d’attributs FK d’une relation R1 est une clé étrangère de R1 qui référence une relation R2 si les conditions suivantes sont respectées:

1. Les attributs de FK sont du même domaine que la clé primaire de R2

2. Soit la valeur de FK dans un tuple ti existe comme valeur de clé primaire dans un tuple tj de R2 soit elle est nulle.

Contraintes d’intégrité : Contraintes générales

• Ce sont des règles additionnelles spécifiées par des usagers ou des administrateurs qui définissent ou restreignent certains aspects du modèle

• Ex: salaire maximum / Nombre max d’employés par branche / GPA max, etc.

• On parle aussi de contraintes d’intégrité sémantique

• On utilise pour cela des commandes du langage SQL telles que les triggers ou les assertions

**La traduction d’un modèle conceptuel vers un modèle relationnel**

**Entités Fortes**

• Les entités fortes sont transformées directement en relations (tables) qui incluent les attributs simples de l’entité ,

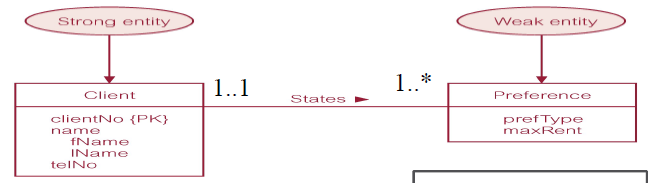
SalesPerson (SalespersonNumber, name, commissionPerc,year)

**Entités faibles**

• Une entité faible est reliée à une entité propriétaire via une association identifiante. Dans ce cas, une occurrence de l’entité faible ne peut pas être identifiée de manière unique sans la clé primaire de l’entité forte (propriétaire)

• Une entité faible doit avoir une participation totale dans la relation identifiante qui la relie à l’entité propriétaire.

• La clé primaire d’une entité faible est partiellement ou complètement dérivée de l’entité propriétaire forte qui lui est reliée.



Preference (clientID, prefType, maxRent)

**Attributs composés**

• On ne garde que leurs constituants simples dans la table/relation représentant l’entité à laquelle ces attributs appartiennent. Ex: Attribut composé Adresse de l’entité Professeur, On ne garde que: • Num, • Rue, • Ville, • Code postal

Attributs multi-valués

• Soit une entité de base avec un attribute multi-valué

• Créer une nouvelle relation (table) pour représenter l’attribut multi-valué.

• Inclure la clé primaire de l’entité de base dans la nouvelle relation (elle sera une clé étrangère)

• A moins que l’attribut multi-valué soit lui-même une clé candidate de l’entité, la clé primaire de la nouvelle relation sera la combinaison de l’attribut multi-valué et de la clé primaire de l’entité. Exemple : Une succursale a l’attribut telNo[1..3]

• Créer une relation (table) Succursale qui contient toutes les autres infos (attributs simples) de la succursale

• Créer une table Telephone qui référence la succursale

Solution complète:

• Succursale(succ, adresse)

• Primary key: Succ

• Telephone (telNo, succNo)

• Primary key: TelNo

• Foreign Key: SuccNo references Succursale(succ)

Les associations binaires (1:\*)

• Dans chaque relation binaire 1:\*, l’entité sur le côté 1 de l’association est désignée comme A, tandis que celle du côté \* est désignée comme l’entité B

• Pour représenter cette association, on poste une copie de la clé primaire de A en tant que clé étrangère dans la relation (table) qui représente B.