

## Sixième partie VI

### Le modèle entité-association

## Plan du cours

- **Partie I : Introduction aux bases de données relationnelles**
  - Cours 1 : Concepts des bases de données relationnelles
  - Cours 2 : L'algèbre relationnelle
- **Partie II : Utilisation des bases de données relationnelles**
  - Cours 3 : Le langage SQL DML (1)
  - Cours 4 : Le langage SQL DML (2)
  - Cours 5 : Le langage SQL DDL
- **Partie III : Développement des bases de données relationnelles**
  - **Cours 6 : Le modèle entité-association**
  - Cours 7 : Élaboration d'un schéma conceptuel
  - Cours 8 : Production du schéma de la base de données

## Qu'allons nous aborder dans ce cours ?

1. Généralités sur la construction d'une base de données
  - les acteurs et les métiers
  - La démarche
  - Les outils
2. Le modèle entité-association de base
  - Les types d'entités
  - Les attributs
  - Les types d'associations
  - Les contraintes d'intégrité

## Généralités sur la construction d'une base de données

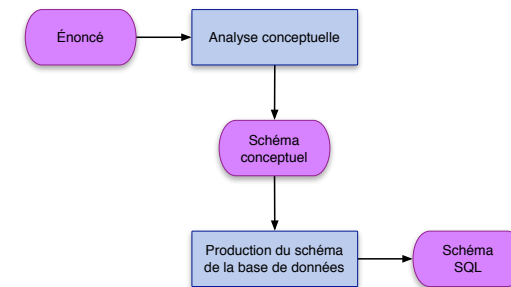
- Lorsque le **domaine d'application** devient complexe il est nécessaire d'avoir une démarche de conception
  - Une base de données réel contient plusieurs centaines de tables contenant plusieurs dizaines de colonnes
  - Il n'est plus possible de raisonner directement sur les tables
- C'est pourquoi on fait appel à un nouveau mode de description appelé **le modèle Entité-association**
  - Ce modèle permet de décrire les concepts du domaine sans se soucier de leur représentation en tables et en colonnes
- C'est un outil de raisonnement et un outil permettant d'exprimer graphiquement un **schéma conceptuel**
  - Le schéma conceptuel est le cahier des charges de la base de données
  - Il permet aux informaticiens et aux utilisateurs de communiquer

## Généralités sur la construction d'une base de données

- Plusieurs rôles, parfois joués par la même personne, apparaissent dans le processus d'élaboration et d'exploitation d'une base de données :
  - **L'utilisateur** : expert du domaine d'application, celui-ci exprime ses besoins d'information mais n'a pas de compétence en informatique
  - **L'analyste** : il dialogue avec l'utilisateur pour cerner ses besoins et les exprimer sous la forme d'un schéma conceptuel
  - **L'implémenteur** : il traduit rigoureusement le schéma conceptuel sous la forme d'une base de données efficace et performante
  - **Le développeur** : il conçoit et développe des applications qui vont utiliser la base de données, e.g., site de vente en ligne
  - **L'administrateur de la base de données** : il veille au jour le jour au bon fonctionnement de la base de données. Il met à jour le SGBD
  - **L'administrateur des données** : il gère les données comme une ressource critique. Il sait où elles se trouvent. Il connaît leur type. Il n'est pas forcément un spécialiste.

## Généralités sur la construction d'une base de données

- La démarche de conception d'une base de données consiste en deux phases :
  1. **L'analyse conceptuelle** au terme de laquelle les besoins en information des utilisateurs sont traduits en un schéma conceptuel
  2. **La production de la base de données** par laquelle le schéma conceptuel est traduit en structures de tables exprimées en SQL



## Généralités sur la construction d'une base de données

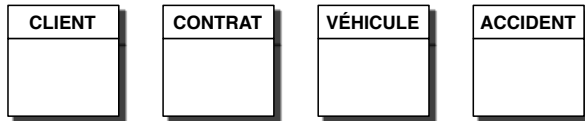
- L'activité d'analyse et de production peuvent rapidement s'avérer fastidieuses et sujettes à erreurs
- Il est possible de trouver des outils pour aider la conception et la production d'une base de données
- Quelques outils :
  - DB-Main (Libre)
  - DBDesigner (Libre)
  - MySQL Workbench (Libre)
  - Microsoft Visio (licence)
  - Maestro SQL (licence)
  - Oracle Designer (licence)
  - PowerAMC (de Sybase) (licence)
  - ERwin (licence)
  - ER/Studio (licence)

## Le modèle Entité-association

- **Le modèle Entité-association** est né dans les années 70 de la nécessité de décrire les besoins en information des utilisateurs d'une base de données indépendamment d'une technologie
- L'identification des concepts du domaine utilise un **modèle**, i.e., **une manière de voir les choses**, auquel a été associé une notation graphique
  - Les notations graphiques peuvent varier un peu
- **La manière de voir les choses** du modèle Entité-association est très simple. Elle consiste à structurer le monde en :
  1. classes d'entités dotées de propriétés
  2. associations les unes avec les autres
- Cette simplicité permet à des utilisateurs peu expérimentés de le manipuler et de le traduire aisément en un schéma de base de données

## Les types d'entités (1/2)

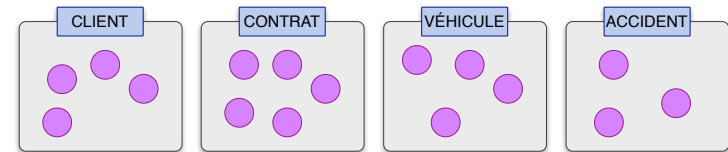
- Le domaine d'application est perçu comme étant constitué d'un **ensemble d'entités abstraites**
- Par exemple dans le contexte de l'assurance automobile, on peut distinguer les entités suivantes :
  - les clients, les véhicules, les contrats, les accidents
- Chaque entité du domaine appartient à un **type d'entités**
- On pourra les représenter comme suit :



- Un type d'entités peut correspondre à des objets concrets inanimés, des objets concrets animés, des conventions abstraites, ou des événements

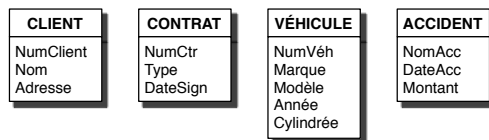
## Les types d'entités (2/2)

- Contrairement aux tables d'une base de données, les types d'entités non pas à proprement parler de **contenu**
- Chaque type d'entités **représente** une population souvent de taille variable

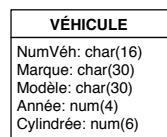


## Les attributs (1/2)

- Chaque client est caractérisé par un numéro, un nom et une adresse
- On modélisera ces faits en dotant le type d'entités CLIENT des **attributs** NumClient, Nom, Adresse

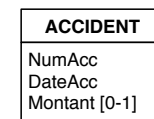


- On spécifiera le type de chaque attribut : numérique, caractère, date, etc. ainsi que sa longueur et son unité si nécessaire



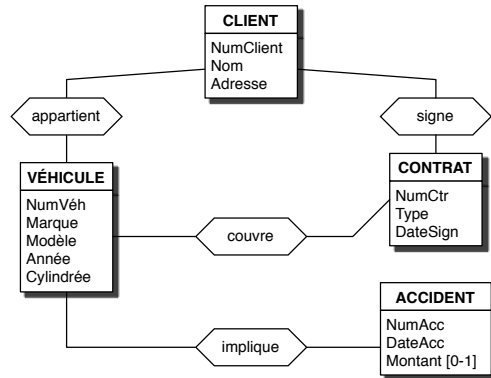
## Les attributs (2/2)

- Il se peut que la valeur de d'un attribut ne soit pas connue au moment où une entité est décrite
- On peut alors déclarer un attribut **facultatif** ou **obligatoire**
  - Un attribut facultatif est indiqué par le symbole de cardinalité [0-1] qui indique qu'à toute entité est associé de 0 à 1 valeur
  - Un attribut obligatoire est indiqué par le symbole de cardinalité [1-1] qui indique qu'à toute entité est associé de 1 valeur
- Par défaut, tous les attributs sont obligatoires



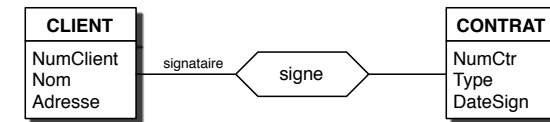
## Les types d'associations

- Une **association** est une relation qui lie deux entités
- Une association porte un nom qui décrit cette relation



## Les types d'associations

- Lorsqu'un type d'entité intervient dans un type d'associations, on dit qu'il joue un **rôle**
- Un rôle est utilisé pour caractériser l'extrémité d'une association
- Par défaut, c'est le nom de l'entité qui tiendra lieu de rôle



## Les types d'associations

- Les **propriétés** associées à un type d'associations  $R$  entre  $A$  et  $B$  permettent de caractériser pour chacun des types d'entités  $A$  et  $B$ , à combien d'associations  $R$  peut et doit participer
- On mesurera cette propriété en indiquant, pour chaque entité le nombre minimum et maximum d'associations auxquelles toute entité participe
- Les propriétés peuvent être découpées en 2 catégories :
  1. Les **propriétés fonctionnelles** qui expriment les associations fonctionnelles entre entités
  2. Les **propriétés à caractère facultatif/obligatoire** qui expriment le caractère obligatoire ou facultatif d'une association entre entités
- Quelque soit le type de propriété, elle peut être synthétisée sous la forme d'une propriété de **cardinalité**

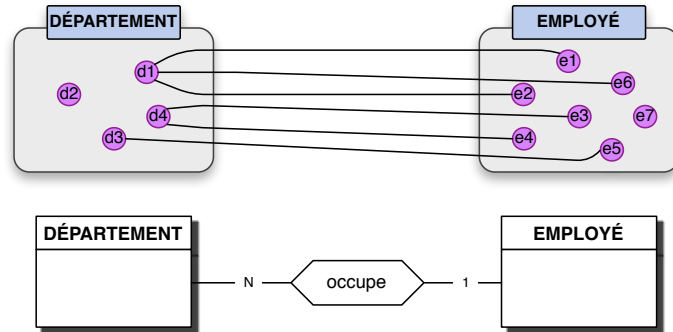
## Les types d'associations

On distingue 4 propriétés fonctionnelles qui décrivent le nombre maximum d'entités  $B$  pour chaque entité  $A$  et inversement :

1. **un-à-plusieurs**
2. **un-à-un**
3. **plusieurs-à-plusieurs**
4. **plusieurs-à-un**

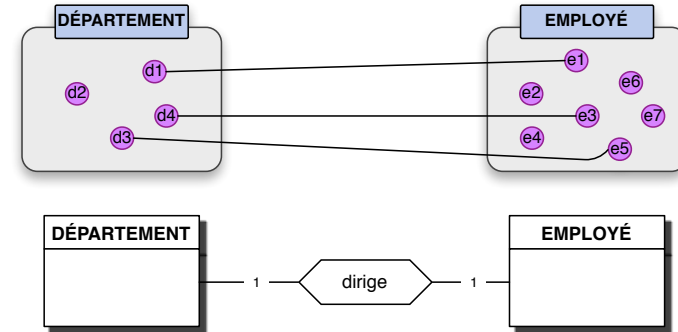
## Les types d'associations

- Le schéma indique qu'un département peut occuper plusieurs employés (N), mais que qu'un employé n'est occupé que par un seul département (1)



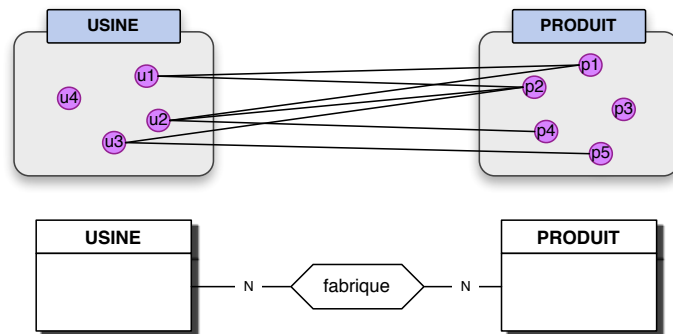
## Les types d'associations

- Le schéma indique qu'un département possède un directeur qui n'est pas l'employé

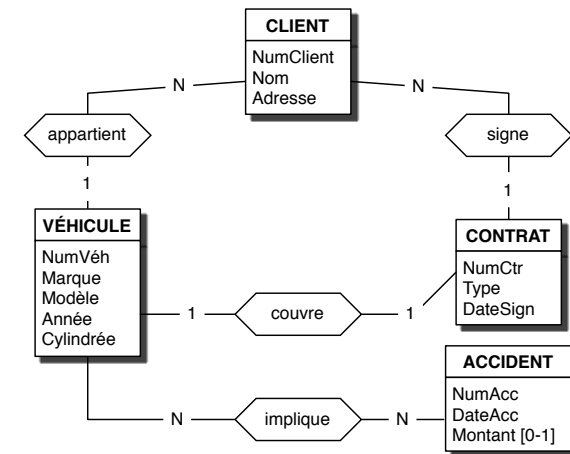


## Les types d'associations

- Le schéma indique qu'une usine fabrique plusieurs produits et qu'un produit peut être fabriqué par plusieurs usines

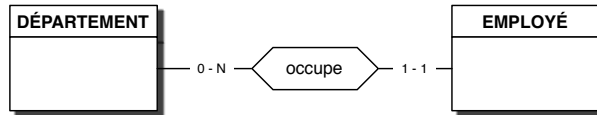


## Les types d'associations



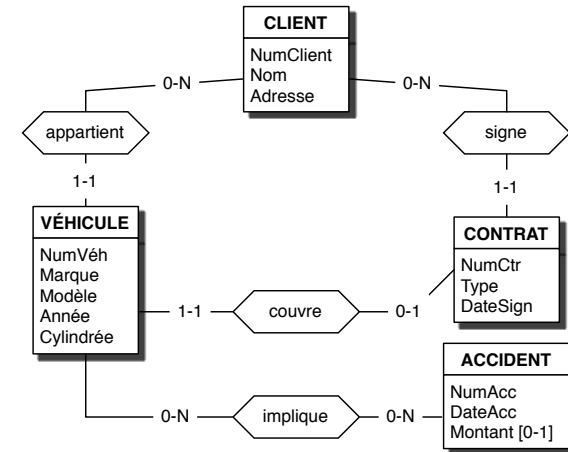
## Les types d'associations

- On peut imposer qu'un type d'associations soit **obligatoire** pour un type d'entités qui y participe
- Exemple :



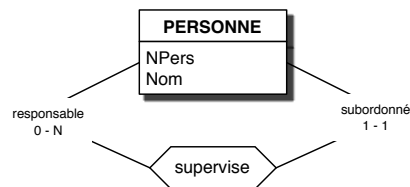
- On exprime par là que tout employé est occupé par un département
- Autrement dit, toute entité DÉPARTEMENT est associée, via *occupe*, à un nombre quelconque (de 0 à N) d'entités EMPLOYÉ, et toute entité EMPLOYÉ est associée, via *occupe*, à exactement une (de 1 à 1) entité DÉPARTEMENT.

## Les types d'associations



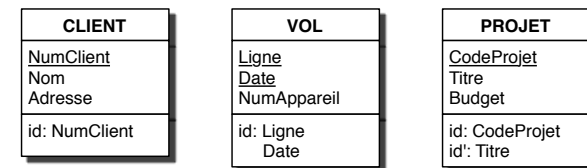
## Les types d'associations

- On peut définir un type d'associations entre un type d'entités et lui-même, définissant ainsi par là un type d'association **cyclique**
- Exemple :



## Les identifiants

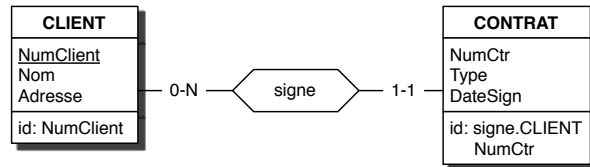
- Un type d'entités est généralement doté d'un attribut qui l'identifie
- Cet attribut est unique, i.e., une seule entité ne peut posséder la même valeur pour cet attribut
- Cet attribut est appelé **identifiant**
- Exemple :



- Un **identifiant primaire** sera précédé par la clause *id* et un **identifiant secondaire** par la clause *id'*
- Les attributs constituant l'identifiant primaire seront également soulignés

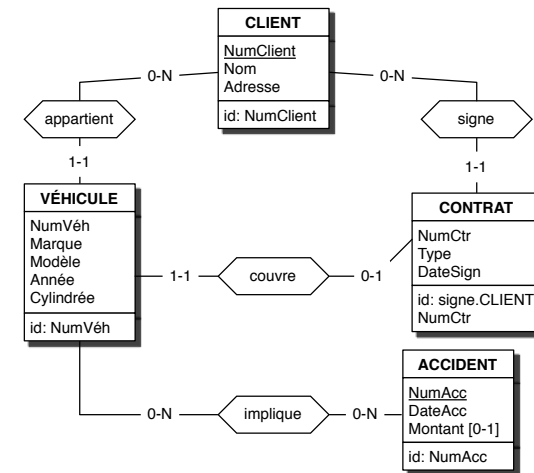
## Les identifiants

- Un **identifiant hybride** est constitué de types d'entités voisins et d'attributs locaux
- Exemple :

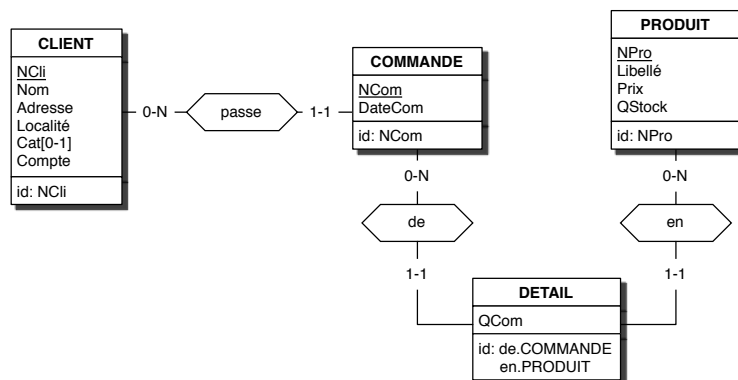


- Plusieurs contrats peuvent être signés par un même client et plusieurs contrats peuvent avoir la même valeur de NumCtr, mais il ne peut exister plus d'un contrat signé par le même client et portant le même numéro NumCtr

## Les identifiants

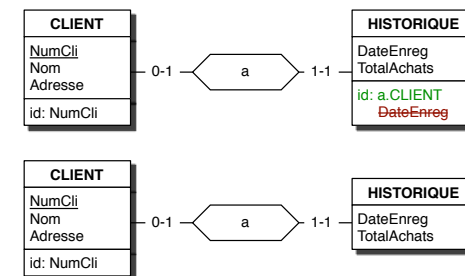


## Les identifiants



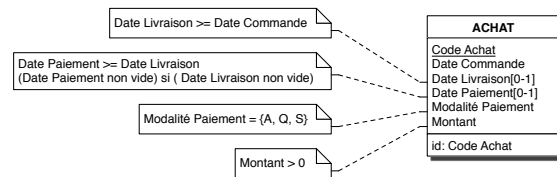
## Les identifiants

- Un **identifiant est minimal** si aucun de ses composants ne peut être retiré sans qu'il perde le statut d'identifiant
  - Par exemple : (NumAcc, Date) constitue un identifiant minimal de l'entité ACCIDENT
  - Il faut veiller à ce qu'un schéma conceptuel ne contienne que des **identifiants minimaux**



## Les contraintes d'intégrité

- Les **contraintes d'intégrité** formalisent certaines propriétés importantes du domaine d'application que les données devront respecter
- On peut distinguer :
  - Les contraintes de base**
    - Les identifiants (primaires et secondaires)
    - Les attributs obligatoires
    - Les contraintes de cardinalité des rôles
  - Les contraintes additionnelles**
    - On désigne sous ce terme toutes les propriétés que l'on voudrait voir définies dans le schéma mais que le modèle ne permet pas d'exprimer de manière explicite. Ces contraintes seront notées sous forme d'annotation.



## Exercice : la modélisation d'une structure administrative

- Modéliser un sous-ensemble d'une structure administrative composée :
  - d'une direction caractérisée par un identifiant et le nom de son président
  - de plusieurs départements localisés et sous l'autorité d'une direction
  - de plusieurs services qui composent les départements identifiés par un nom et un responsable
  - Chaque service a la charge de dossiers identifiés par un numéro et dotés d'un titre et d'une date d'enregistrement
  - Dans chaque service travaillent des employés identifiés par un numéros et caractérisés par leur nom et leur adresse

## Exercice : la modélisation d'une structure administrative

