BASE DE DONNÉES

JOURNÉE 1 – APRES-MIDI

Karine BRIFAULT

BASE DE DONNÉES
MODELISER

- La phase de modélisation dans une base de données est **capitale**.

- De sa réussite dépendra

la réussite d'une modelisation

  - La stabilité et la cohérence des données
  - La volumétrie
  - La rapidité des réponses aux requêtes
  - L'évolutivité de la base

envisage d'utilisations possible, faire des bibliothèques et petits modules

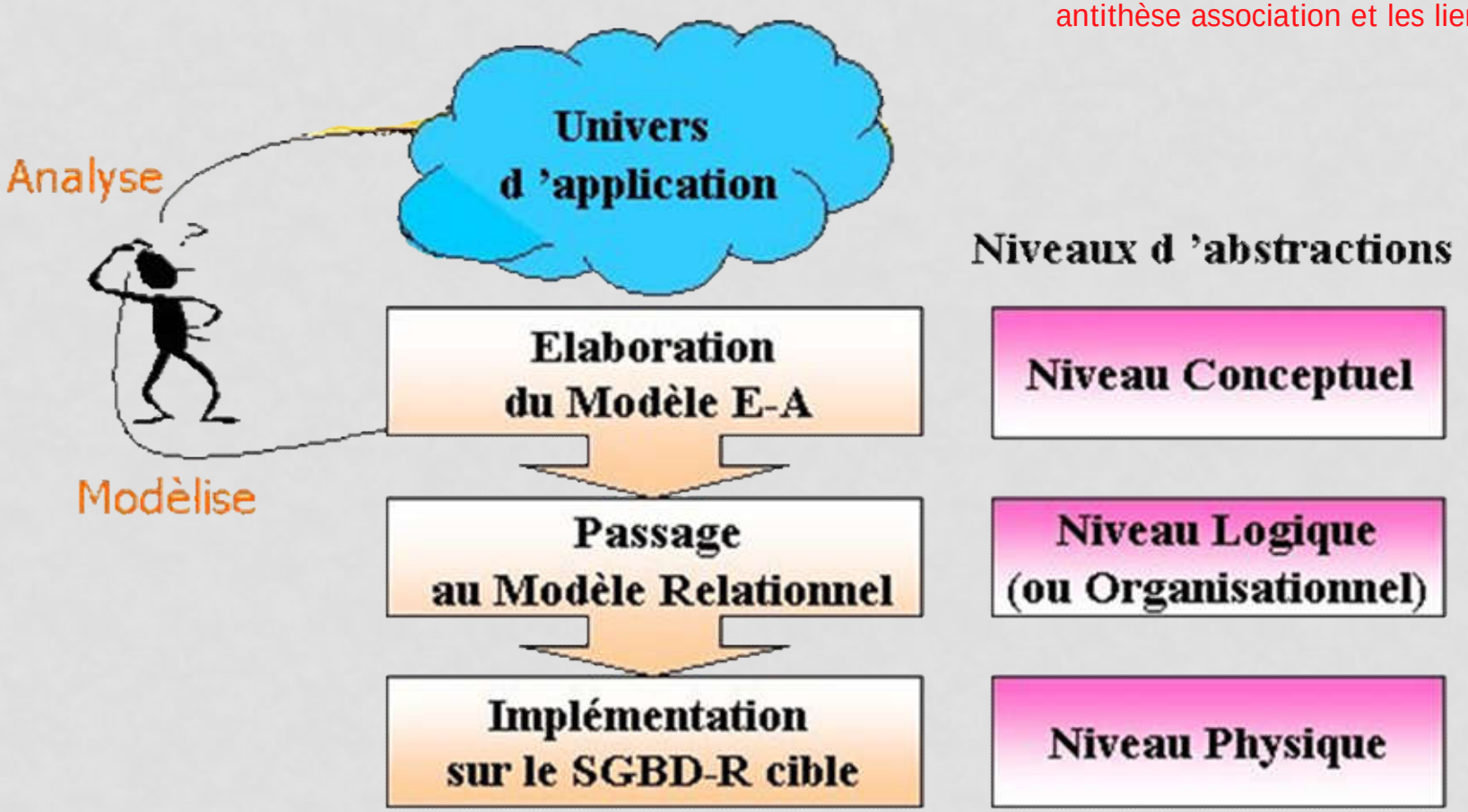
BASE DE DONNÉES
MODELISER

- La modélisation se fait souvent en début de phase du projet.
- On commence par modéliser ses données avant de taper son code.
- **ATTENTION**
  - Contrairement à du code, il est beaucoup plus lourd et compliqué de revenir sur un modèle de données, car de lui dépend toute l'application (les écrans, le code, …)

BASE DE DONNÉES
MODELISER

La démarche de Conception

antithèse association et les liens



BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Déterminer les entités/classes et attributs :
  - entité/instance de classe = objet décrit par de l'information
    - objet caractérisé uniquement par un identifiant = attribut
    - attribut multi-valué ou avec une association 1:N = entité ou instance
    - attacher les attributs aux ensemble d'entités/classes qu'ils décrivent le plus directement
    - éviter au maximum les identificateurs composites

comment on va faire
que l'objet soit unique

- Identifier les généralisations-spécialisations/héritage
- Définir les associations
  - éliminer les associations redondantes
  - éviter les associations n-aires
  - calculer les cardinalités de chaque association

BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Important : Vous concevez une base pour un usage connu à un instant T. Ne faites pas trop générique ou trop fermé.

- Exemple d'entités
  - **Client** : personne qui peut acheter une ou plusieurs voitures via l'usage d'un contrat
  - **Voiture** : Objet achetable via un contrat par un client
  - **Contrat** : élément utilisé pour les règles d'achats entre un client et une voiture

BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Exemple d'attributs
  - Client

Unique
  - Numéro
  - Nom
  - Adresse

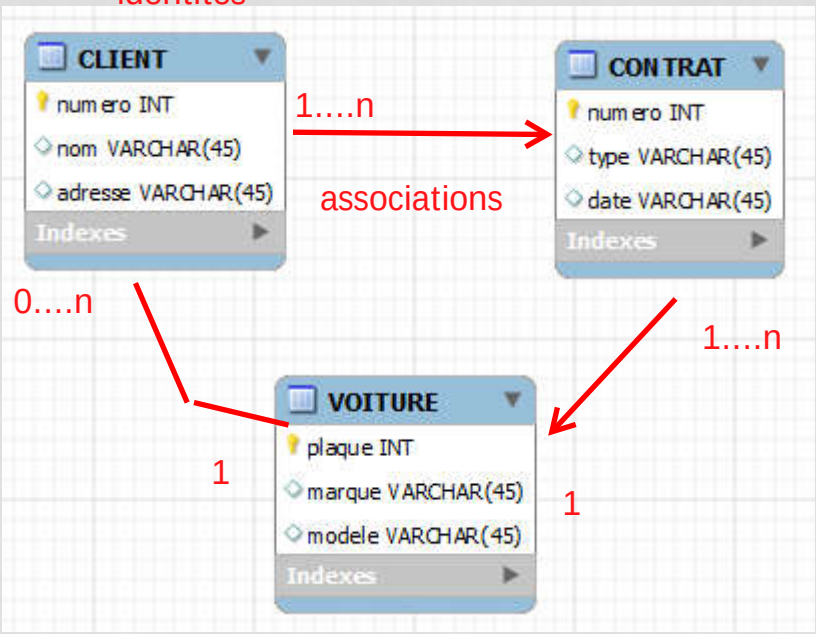
  - Voiture

unique
  - Plaque
  - Marque
  - Modèle

  - Contrat

unique
  - Numéro
  - Type
  - Date

identités

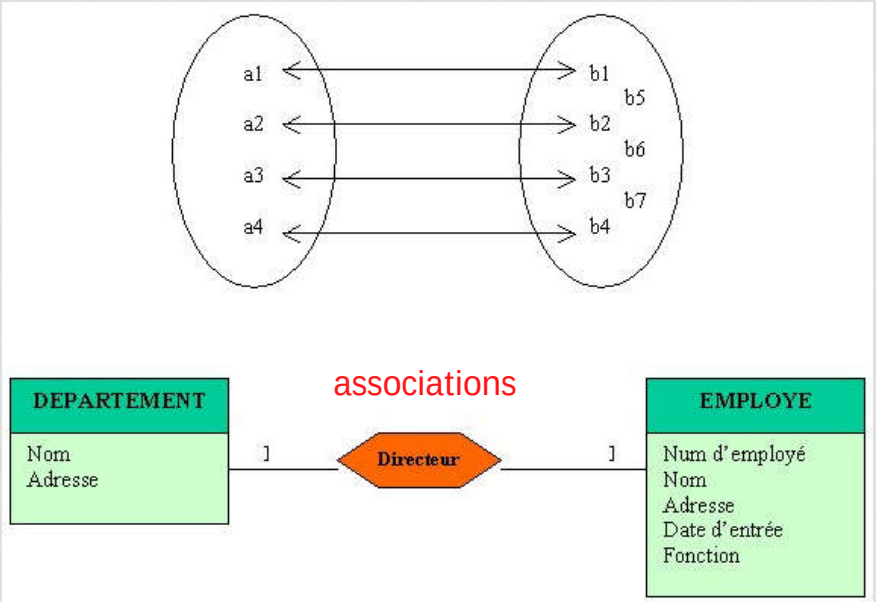


num_client
plaque - voiture

BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

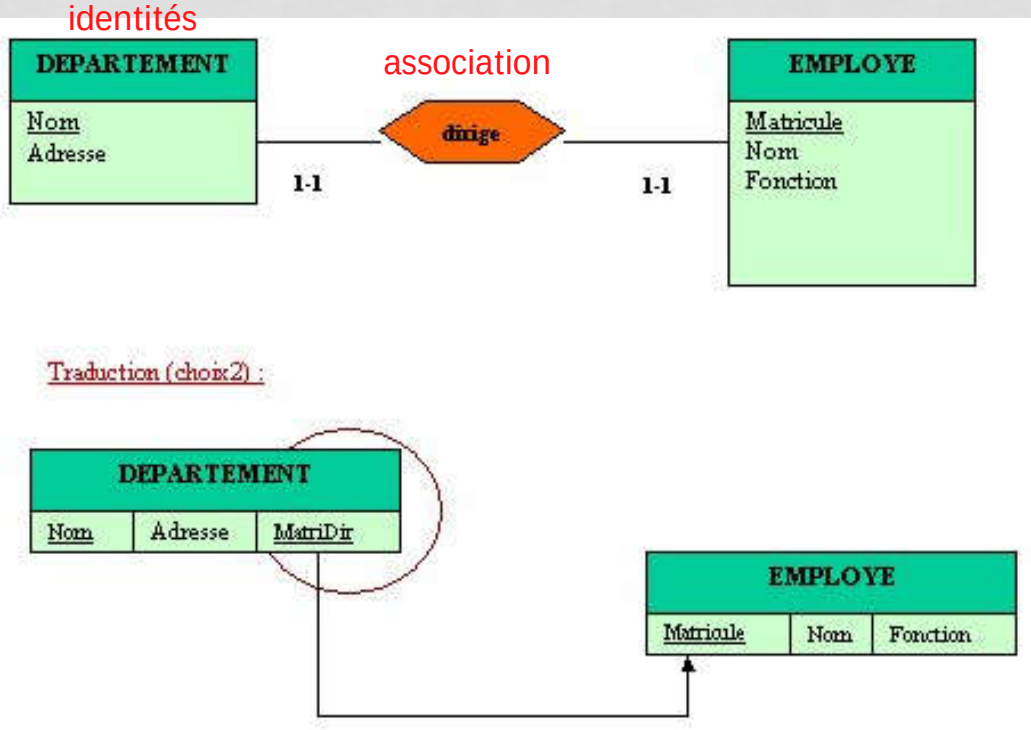
- La cardinalité est le nombre de participation d'une entité à une relation.

- Cardinalité 1 – 1
  - un employé ne peut être directeur que dans un seul département et un département n'a qu'un seul employé comme directeur



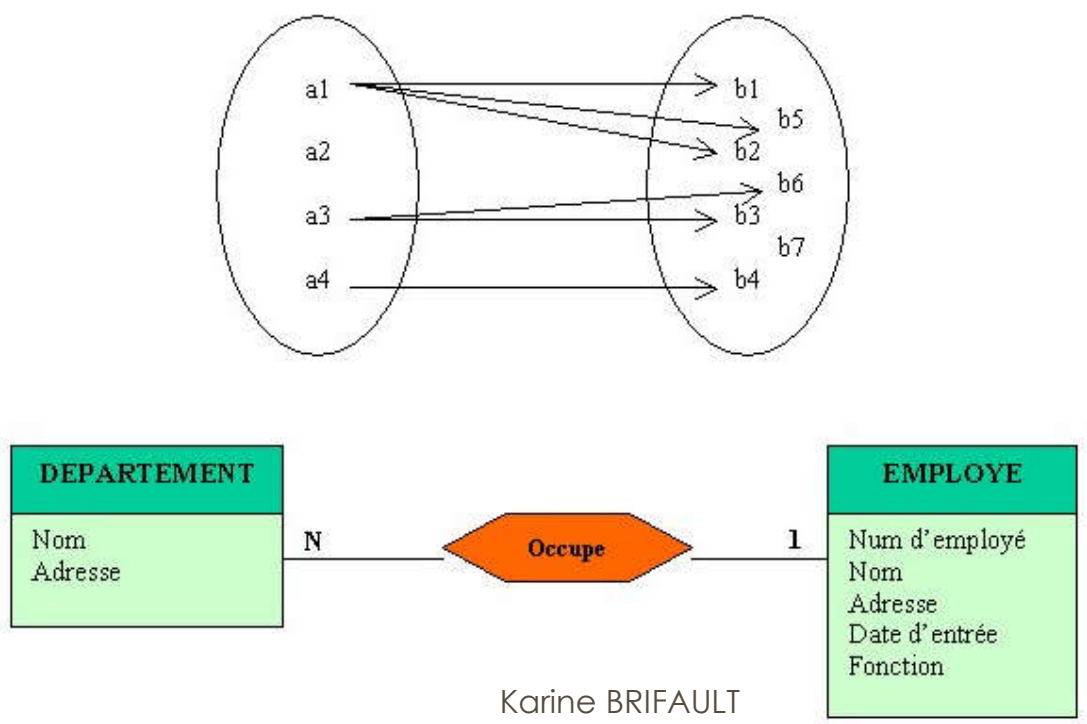
BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Cardinalité 1 – 1
  - Traduction en base de données



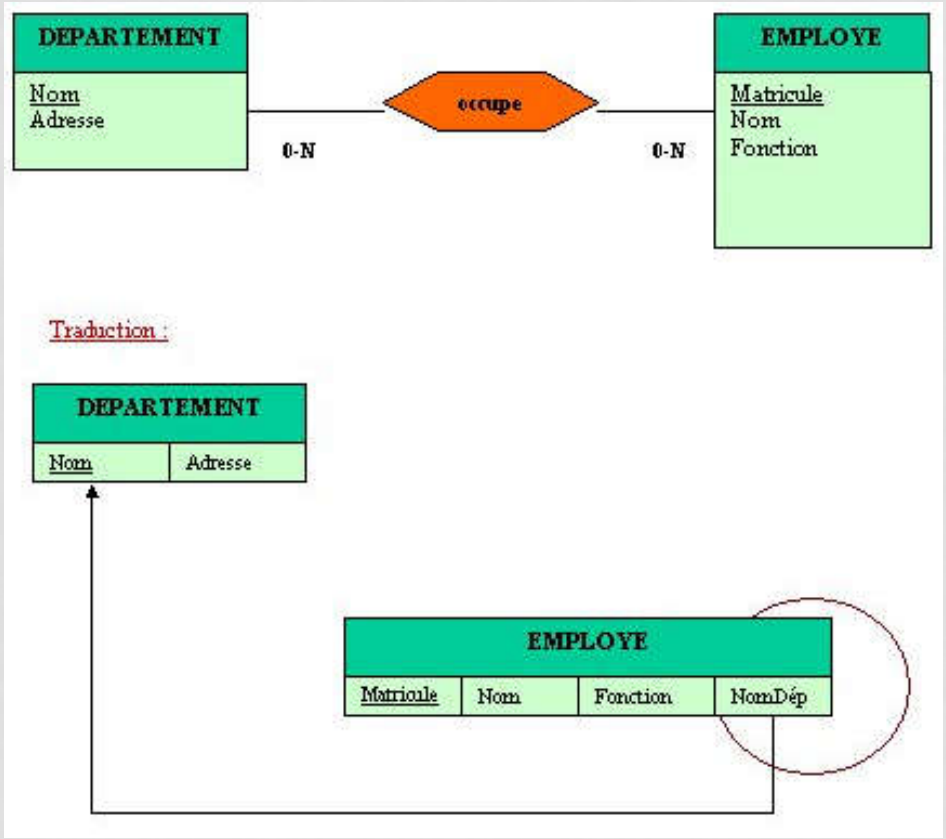
BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Cardinalité 1 – n
  - un département peut occuper plusieurs employés qui réalisent différentes fonctions mais chaque employé ne fait partie que d'un seul département.



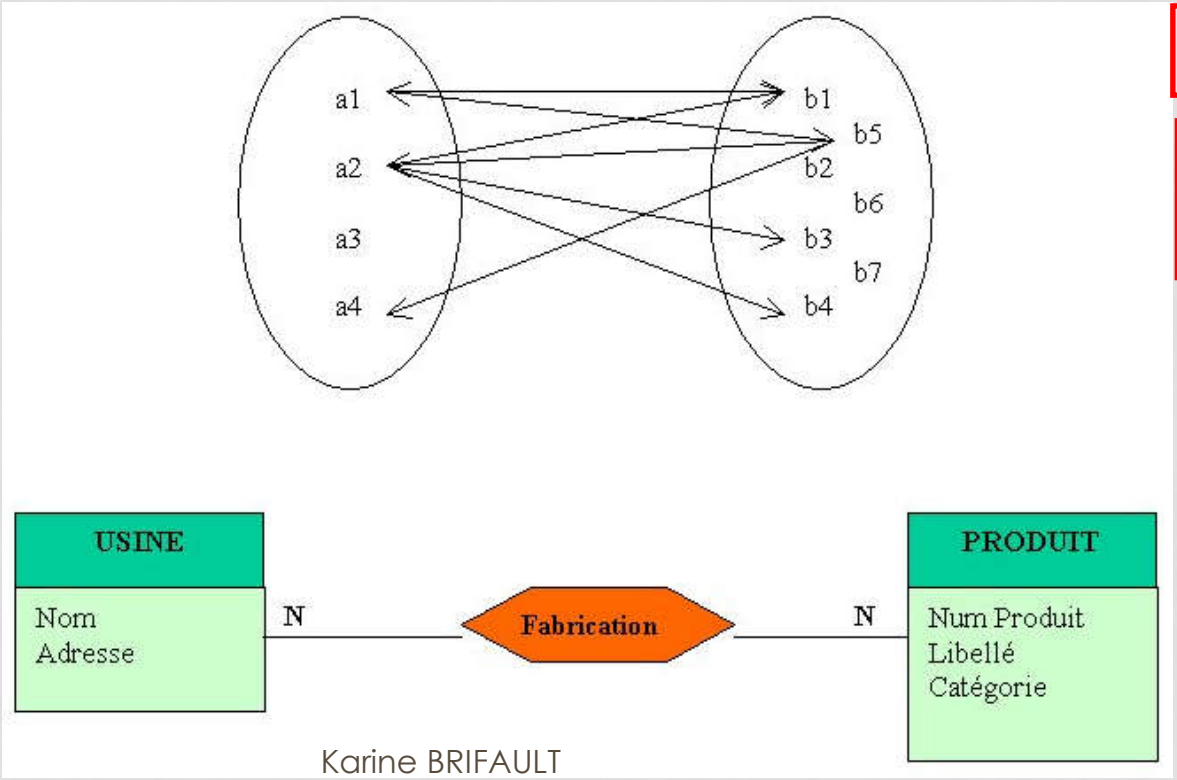
BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Cardinalité 1 – n traduit en base de données



BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Cardinalité n – m
  - un type de produit peut être fabriqué en plusieurs usines et une usine donnée peut fabriquer plusieurs types de produits.

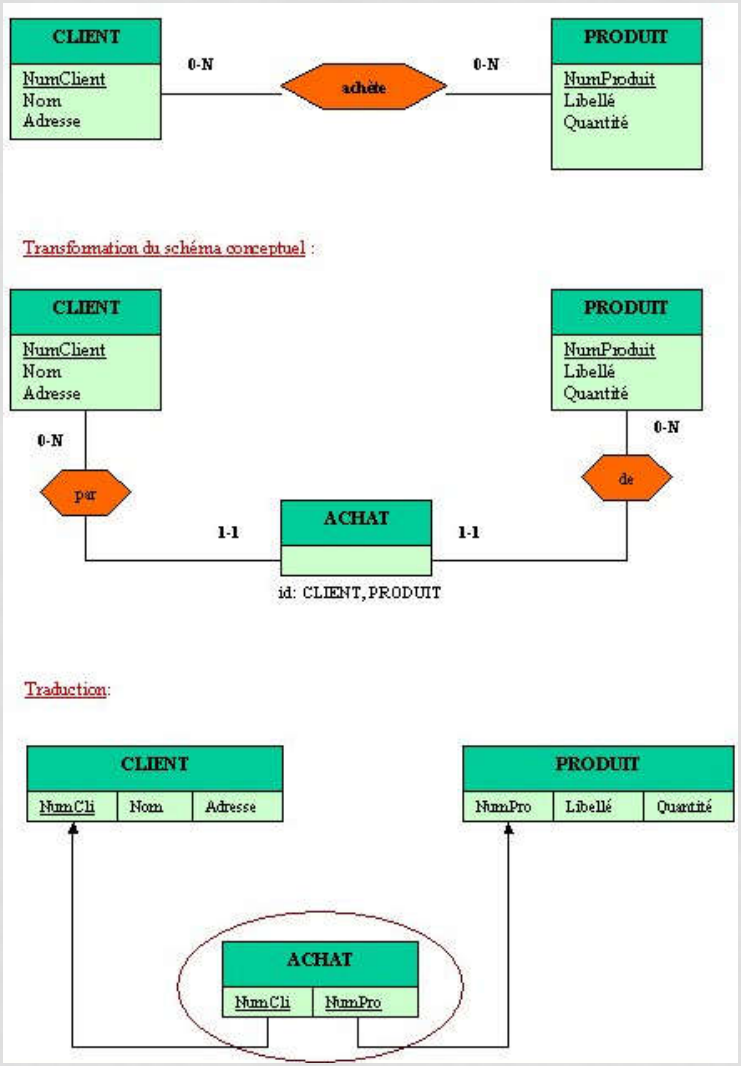


fabrique

id usine
id produit

BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Cardinalité n – m : traduit en base de données
  - On ajoute une table de jointure

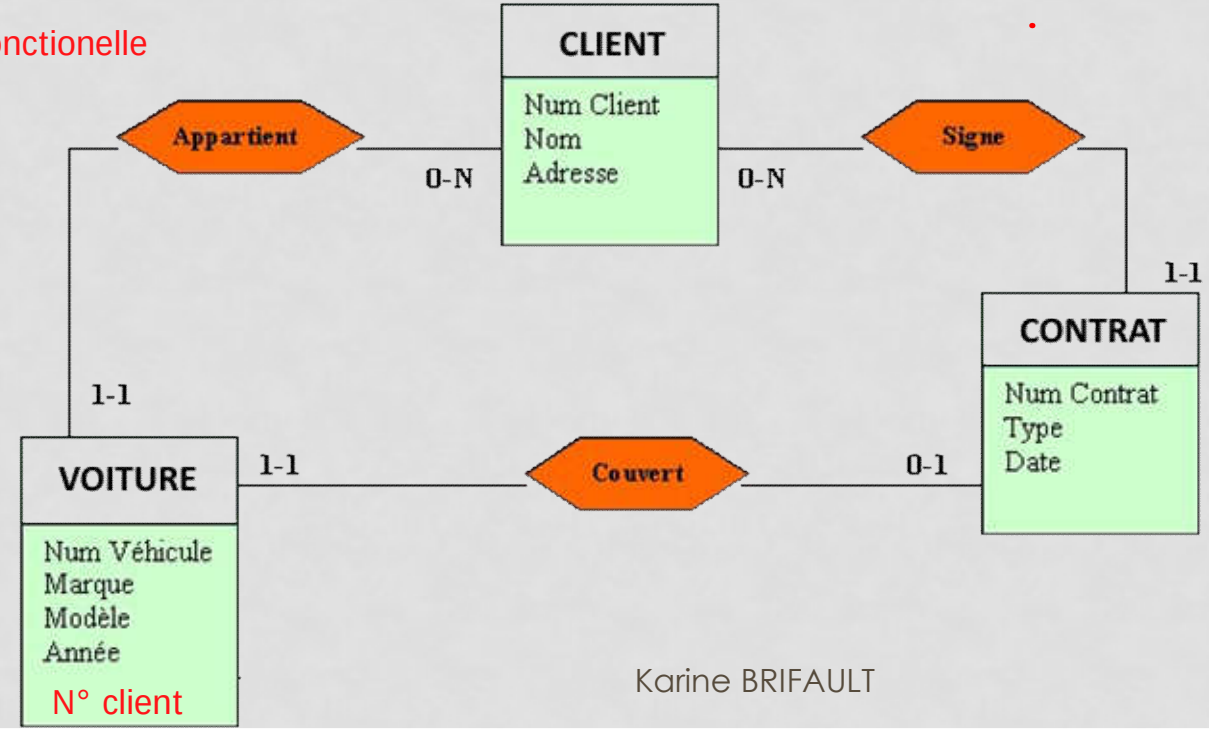


BASE DE DONNÉES
MODELISER ENTITÉS-ASSOCIATIONS

- Exemple de relation entre entités
  - Un client peut passer 1 ou n contrat
  - Une voiture est définit dans un contrat
  - Un contrat lie une voiture à un client

arriver à
Model E/A ----------> SQL

conception fonctionelle



BASE DE DONNÉES
MODELISER EA

- Pour Résumer
  - Déterminer la liste des entités.
    - établir la liste de ses attributs ;
    - parmi ceux-ci, déterminer un identifiant unique (clefs primaires).
  - Déterminer les relations entre les entités.
    - dresser la liste des attributs propres à la relation ;
    - définir les cardinalités.
  - Vérifier le schéma obtenu, notamment :
    - pas de redondance d'information ;
    - ne pas 'exploser' les tables à outrance ;
    - s'assurer qu'il répont aux demandes.
  - Valider avec les utilisateurs.

BASE DE DONNÉES
TRANSFORMER SON MODÈLE

- Il faut transposer son modèle EA en modèle SGBD
- Attention : dans 95% des cas le modèle SGBD est destiné à un SGBD précis.
- Il peut être compliqué de passer d'un modèle pour MySQL à un modèle pour Oracle

BASE DE DONNÉES
RAPPEL

- **Schema** est une base de donnés en GBD
  - Le schéma permet de regrouper plusieurs tables
- **Table / Vue**
  - Définit une entité par sa vision en colonne
- **Colonne**
  - Représente le nom que l'on donne à une information
- **Index** un clé ou un identifiant
  - Structure de la base de données qui cible une ou plusieurs colonne et qui précise comment ranger les informations (par ordre croissant par exemple, ou de manière unique, …)

BASE DE DONNÉES
RAPPEL

- **Clef primaire**
  - Représente la colonne (ou les colonnes) qui servira d'identifiant unique pour le tuple.
  - Doit être unique et non nulle
  - En générale, elle est gérée par un index

- **Clef étrangère**
  - Quand une colonne d'une table A fait référence à une clef primaire du table B, on dit que cette colonne est une clef étrangère
  - Peut être nulle ou non
  - En générale, elle est gérée par un index

BASE DE DONNÉES
MODELISER

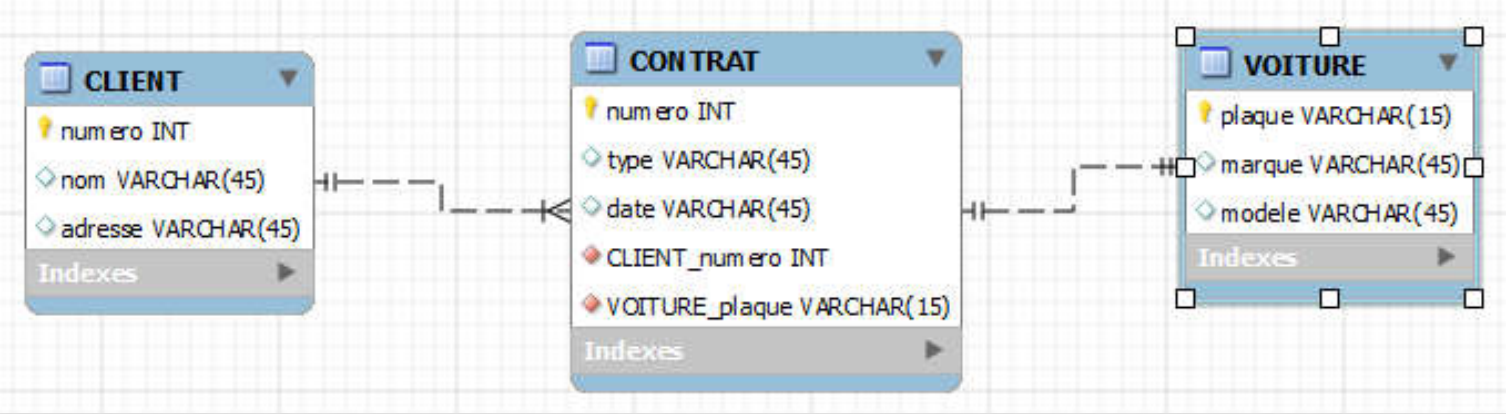
- Identifie les clefs primaires
  - Client
    - **Numéro**
    - Nom
    - Adresse
  - Voiture
    - **Plaque**
    - Marque
    - Modèle
  - Contrat
    - **Numéro**
    - Type
    - Date

Ou

  - Contrat
    - **Plaque**
    - **NumeroClient**
    - Type
    - Date

BASE DE DONNÉES MODELISER

- Exemple de modélisation finale



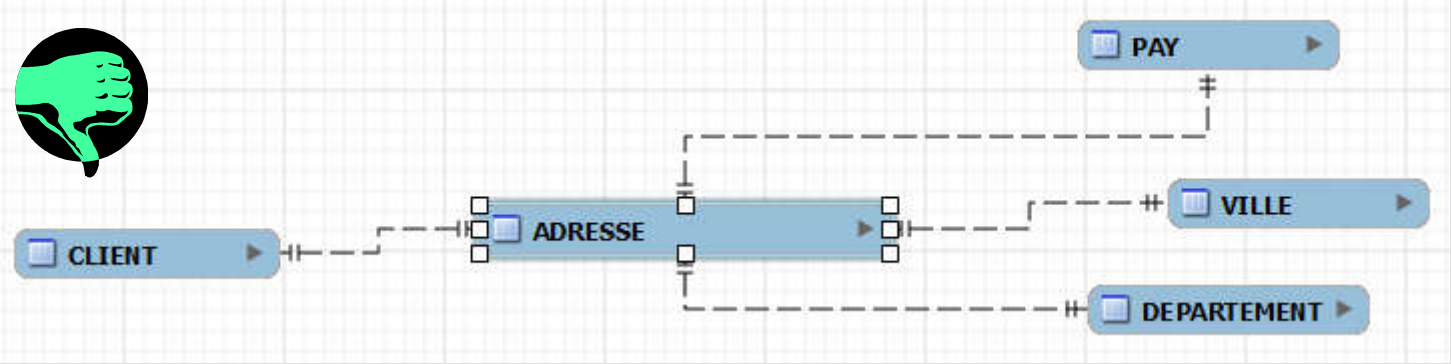
BASE DE DONNÉES
BIEN TRANSFORMER

- **Une clef primaire par table**
  - Eviter les clefs primaires sur plusieurs colonnes
  - Attention : différencier clef primaire technique / clef primaire fonctionnelle

auto incrémentation soit en lon lon

8 bits = 0 - 255

- Ne pas faire trop de jointure, sauf si le besoin fonctionnel l'impose



BASE DE DONNÉES
BIEN TRANSFORMER

- Donnez des noms fonctionnelles à vos tables et vos colonnes
  - Attention : pas trop long, limitez vous à 25 caractères
- Typez bien vos colonnes
  - Entier : gros / petit / signé ?
  - Flottant : nombre de chiffre après la virgule
  - Chaine de caractères : taille ?
  - …
- Ne pas passer d'un diagramme UML à un diagramme SQL
  - Objet (Java, .NET, C++ …)⇔Relationnel (SQL, PL/SQL, …) : deux univers très différents

BASE DE DONNÉES
BIEN TRANSFORMER

- **ATTENTION**
  - Toutes les bases de données n'ont pas les mêmes typages
  - Restez simple, mais cohérent
  - Exemple
    - Code postal : entier ou chaine de caractères ?
    - Mot de passe : chiffre / une chaine de caractères / blob ?
    - Date : date, timestamp, chaine de caractères ?
    - Numéro de sécurité social : chiffre / chaine de caractère ?

BASE DE DONNÉES
SQL POUR SGBD

- Après avoir 'dessiné' sa base de données il faut réaliser un script qui la fabriquera au sein du SGBD.
- Afin de rester normalisé, le script sera en SQL. Plus précisément ce sera un DDL (**Data Definition Language)**
- On se limitera à certaines requêtes SQL
  - CREATE, ALTER, DROP, RENAME

BASE DE DONNÉES
SQL POUR SGBD

- Le **SQL** : langage standardisé qui permet de manipuler des données et même parfois la base de données.
- C'est du code, avec une syntaxe stricte.
- Ce code a pour vocation d'être portable, mais en réalité il peut très vite devenir dépendant (voir spécifique) à votre SGBD.

BASE DE DONNÉES
SQL POUR SGBD

- Créer un schéma (en MySQL)
  - CREATE SCHEMA IF NOT EXISTS ***monShema*** CHARACTER SET = "UTF8";

- Créer une table (en MySQL), ordre **CREATE**

```
CREATE TABLE IF NOT EXISTS monShema.CLIENT (
  numero INT NOT NULL,
  nom VARCHAR(45) NULL,
  adresse VARCHAR(45) NULL,
PRIMARY KEY (numero)
 );
```

BASE DE DONNÉES
SQL POUR SGBD

- ATTENTION : les types en MySQL
  - TINYINT[(length)]
  - | SMALLINT[(length)]
  - | MEDIUMINT[(length)]
  - | INT[(length)]
  - | INTEGER[(length)]
  - | BIGINT[(length)]
  - | REAL[(length,decimals)]
  - | DOUBLE[(length,decimals)]
  - | FLOAT[(length,decimals)]
  - | DECIMAL(length,decimals)
  - | NUMERIC(length,decimals)
  - | DATE
  - | TIME
  - | TIMESTAMP
  - | DATETIME
  - | CHAR(length) [BINARY | ASCII | UNICODE]
  - | VARCHAR(length) [BINARY]
  - | TINYBLOB
  - | BLOB
  - | MEDIUMBLOB
  - | LONGBLOB
  - | TINYTEXT
  - | TEXT
  - | MEDIUMTEXT
  - | LONGTEXT
  - | ENUM(value1,value2,value3,...)
  - | SET(value1,value2,value3,...)

BASE DE DONNÉES
SQL POUR SGBD

- Pour supprimer schéma ou une table on utilisera l'ordre **DROP**
  - DROP SCHEMA ***monShema***;
  - DROP TABLE **monShema.CLIENT**;

BASE DE DONNÉES
SQL POUR SGBD

- Pour modifier une table on utilisera l'ordre **ALTER**
  - ALTER TABLE **monShema.CLIENT** DROP COLUMN **nom**;
  - ALTER TABLE **monShema.CLIENT** ADD COLUMN **age** INT NULL;
  - ALTER TABLE **monShema.CLIENT** RENAME monShema.utilisateur;
  - ALTER TABLE **monShema.CLIENT** MODIFY age VARCHAR(3);

BASE DE DONNÉES
SQL POUR SGBD

- Ordre de création des tables
  - On part toujours de la plus libre (celle qui a le moins de contraintes)
  - Idéalement, il ne doit pas y avoir de cycle
  - Dans le pire des cas, il est possible de désactiver les contraintes le temps de la création de la base.