Synthèse De BD

Partie Conception

Fait par Brandon Van Bellinghen année 2021/2022

Cours 6 :

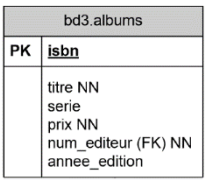
Introduction à la conception ;

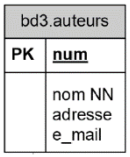
- Rechercher des entités dans un énoncé
- Les représenter sous la forme d'un **Diagramme de Structure de Données (DSD)**
  - Mettre en évidence les **entités** et leurs **attributs.**
  - Définir un **identifiant unique.**
  - Préciser le **contenu** de chaque attribut.
  - Formaliser les **relations** entre les entités.

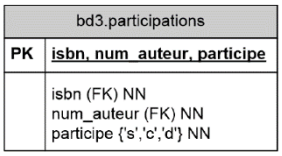
Clé primaire ;

- **But = assurer l'unicité d'un enregistrement / tuples**
- Choix :
  - Identifiant naturel unique
    - ➔ Ex : albums.isbn
  - Numéro auto-incrémenté
    - ➔ Ex : éditeurs.num

  Identifiant composite (concaténation)
    - ➔ Ex : isbn, num_auteur,participe



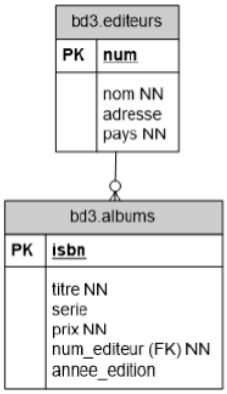




Clé étrangère ;

- **But = assurer l'intégrité référentielle,** garantir la cohérence des références entre les tables

- Conditions pour une intégrité référentielle :
  - Une FK ne peut référer qu'une PK existante
  - Un enregistrement / tuple ne peut pas être supprimé si sa PK est référencée comme FK



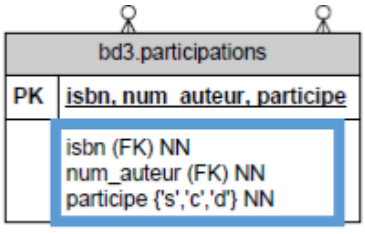
PK concaténées : normes ;

- Conditions pour une clé concaténée :
  - Toutes les parties de la clé doivent être déclarées ''NN''
  - La/Les FK doit/doivent être déclarées.

```
CREATE TABLE bd3.participations

(isbn          char(13) NOT NULL
               REFERENCES bd3.albums(isbn),
num_auteur     int NOT NULL
               REFERENCES bd3.auteurs(num),
participe      char(1) NOT NULL
               CHECK (participe IN ('s','d','c')),

PRIMARY KEY    (isbn, num_auteur, participe)
```

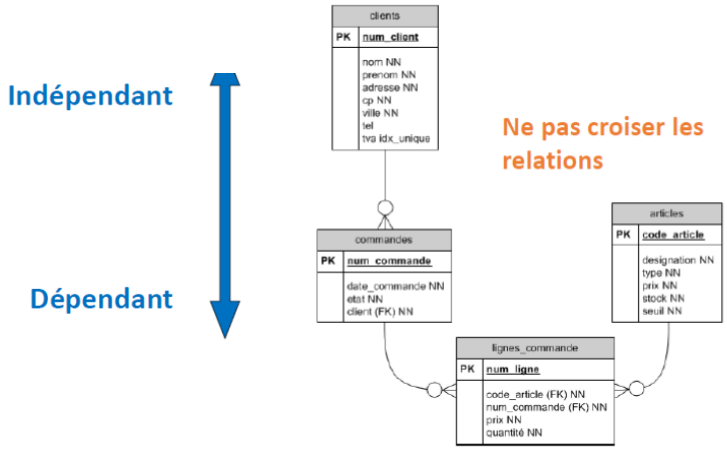


Notations : conventions

- Nom de tables
  - Entièrement en minuscules
  - au pluriel
- Nom de champs entièrement en minuscules
- Pas de caractères spéciaux ou accentués

- Si plusieurs mots, ils sont séparés par ‘’_ ‘’
- PK = Primary Key : la PK est toujours soulignée
- FK = Foreign Key
- NN = Not Null

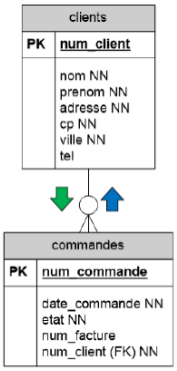
Organisation graphique ;



vocabulaires business ;

- Vocabulaire business
  = vocabulaire du métier
  = vocabulaire du client
- Il est essentiel d’utiliser le **vocabulaire du client** ! Et dans notre cas, le vocabulaire de l’énoncé.

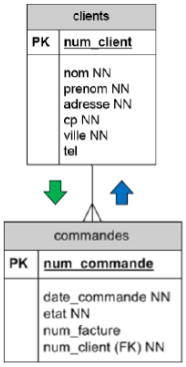
Relation de 1 à 0:N ;



- Une relation = **deux** significations !
- Chaque client peut avoir 0, une ou plusieurs commandes.
- Chaque commande appartient à un seul client.
- La relation implique une FOREIGN KEY (FK) toujours du côté de la fourche.

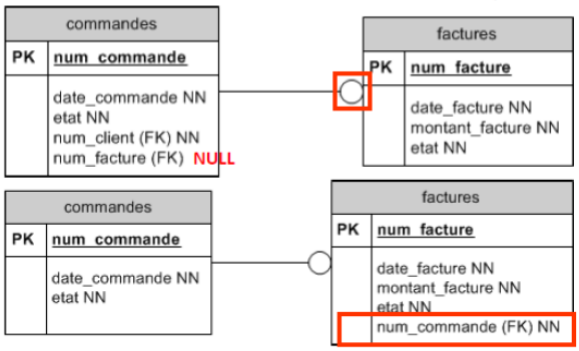
Relation de 1 à 1:N ;

- Chaque client DOIT avoir au moins une commande
- Il peut avoir plusieurs commandes
- Chaque commande appartient à un seul client.

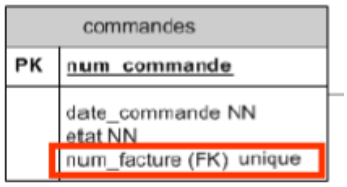


Relation de 1 à 0:1 ;

- Une commande référence une facture ou non

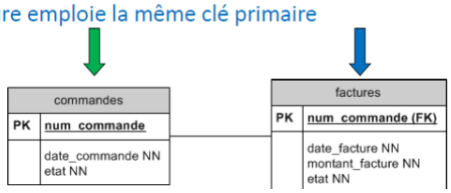


- Afin de s'assurer que 2 ou plusieurs commandes ne référencent pas le même numéro de facture, on peut noter une **contrainte d'unicité** sur l'attribut num_factures (noté ici **unique**).

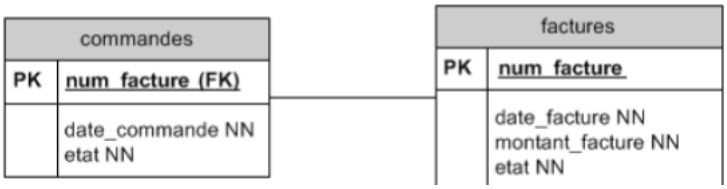


Relation de 1 à 1 ;

- **Si** une commande doit nécessairement correspondre une facture
- La commande est créée avec une clé primaire num_commande
- La facture emploie la même clé primaire



- Remarquons qu'il serait plus judicieux d'utiliser le numéro de factures qui est un élément indispensable d'une facture en bonne et due forme.

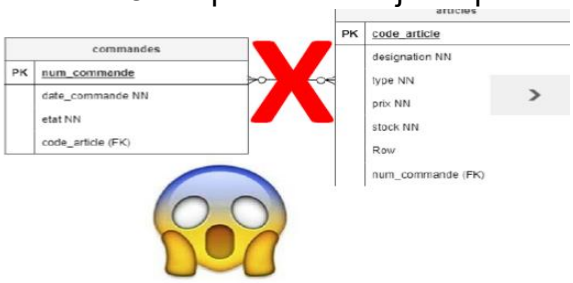


Autres relations entre 2 tables ;

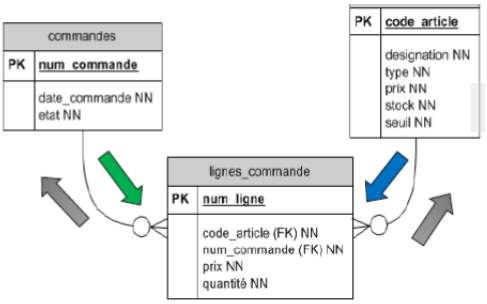
- On peut combiner les terminaisons de relations :
  - Relations 0-1 : 0-1 
  - Une entité a référence éventuellement une entité b
  - Même chose pour b
  - Relation 0-1 : 0-N 
  - Relations 0-1 : 1-N 

Relation de M à N ;

- Une relation de M à N ne peut jamais être représentée directement
- Elle s'implémente toujours par 2 relations de 1 à N

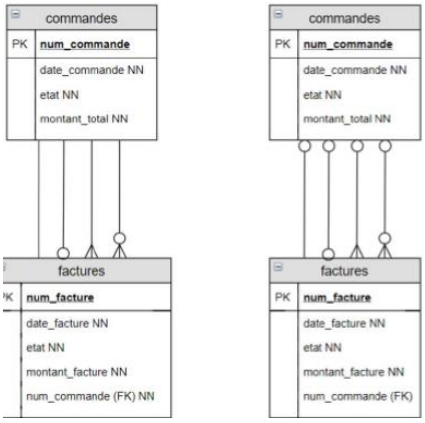


- Une relation de M à N s'implémente toujours par 2 relations de 1 à N



- Une commande peut porter sur plusieurs articles
- Un article peut faire partie de plusieurs commandes

Résumé des 8 relations possibles ;

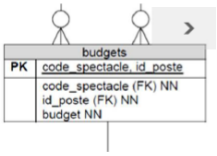


Cours 8 :

PK et FK composées ;

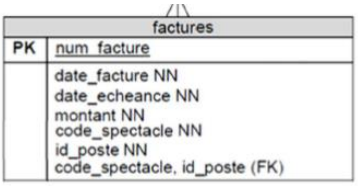
PK composite ;

```
CREATE TABLE budgets (
  code_spectacle char(4) NOT NULL
  references spectacles(code_spectacle),
  id_poste integer NOT NULL
  references postes(id_poste),
  budget double precision NOT NULL,
  PRIMARY KEY (code_spectacle,id_poste)
  )
```



FK composite ;

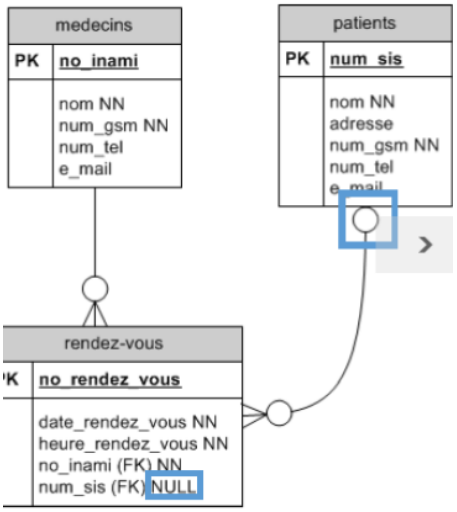
```
CREATE TABLE factures (
 num_facture integer PRIMARY KEY,
 code_spectacle char(4) NOT NULL,
 id_poste integer NOT NULL,
 …,
 FOREIGN KEY(code_spectacle,id_poste)
 REFERENCES budget_depenses(code_spectacle,id_poste)
 )
```



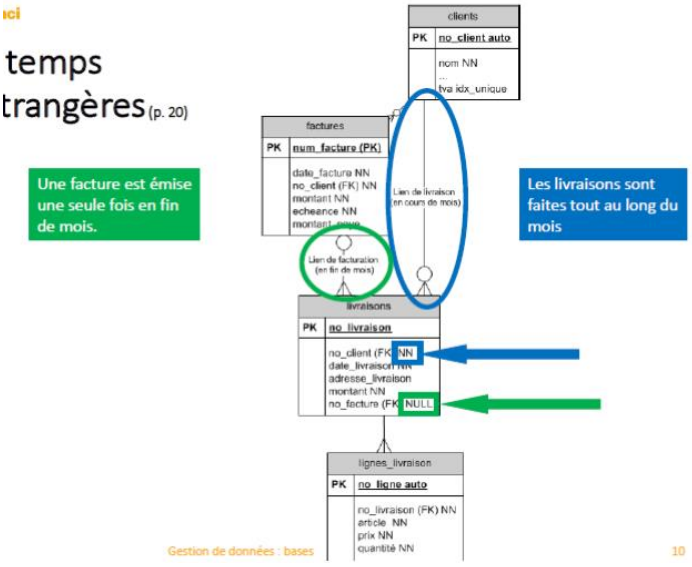
Clé étrangère NULL ;

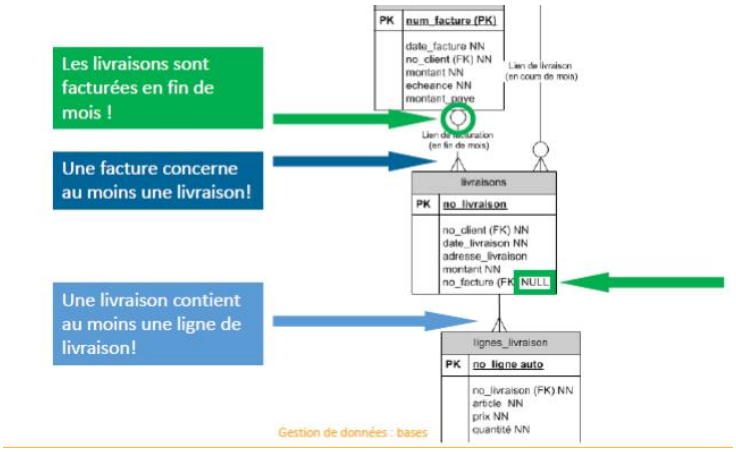
Un rendez-vous peut être attribué à 0 ou 1 patient.

➔ num_sis dans la table rendez-vous est une FK qui peut être NULL



Influence du temps sur les clés étrangères ;

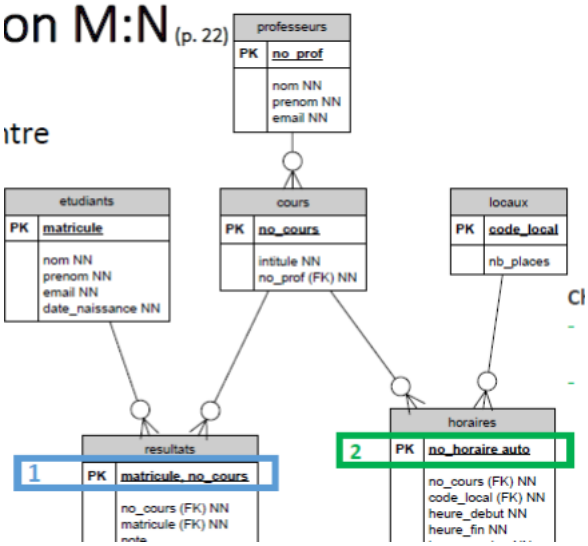




Relations multiples entre tables ;

Double relation M :N ;

- Relation de M:N entre
  - Étudiants et cours
  - Locaux et cours

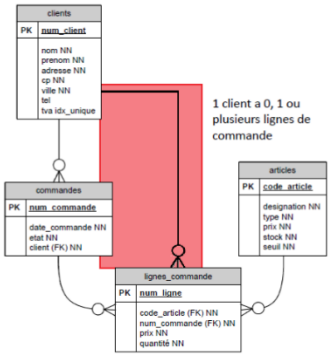


Choix des clés primaires :
- Concaténation (1)
- La combinaison 'matricule et no_cours' est unique : un étudiant ne peut être inscrit qu'une seule fois à un même cours

Choix des clés primaires :
- Numéro auto-incrémenté (2)
- La combinaison 'no_cours et code_local' n'est pas unique : un cours pourrait être donné plusieurs fois dans un même local

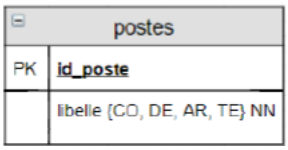
Relation induite :

- Relation 1:N entre
  - Clients et commandes
  - Commandes et lignes_commandes
    - ➔ Relation 1:N induite entre clients et lignes_commande
- Une relation induite n'est pas dessinée.



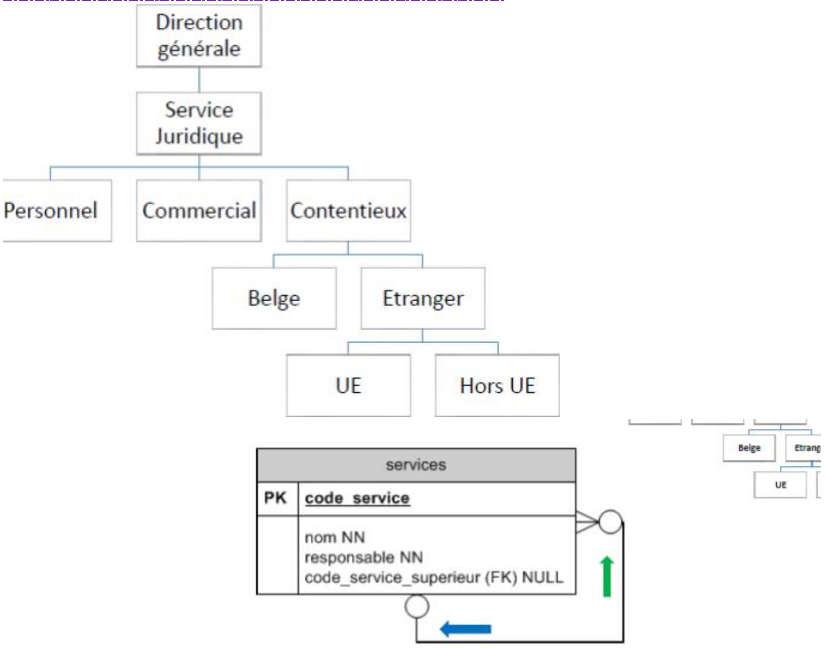
Enumération :

- Permet de limiter les valeurs autorisées pour un champ spécifique
- S'écrit entre accolades { }
- Si on reprend l'exemple des ''postes'' du théâtre du trac, on pourrait par exemple avoir 4 postes : costumier, décorateur, artiste et technicien. Les libellés seraient donc limités à une liste de 4 possibilités.



Cours 8 ;

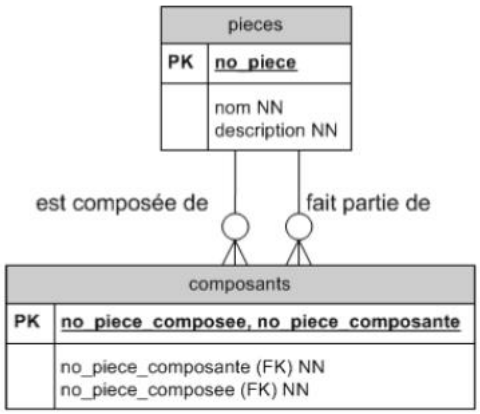
Relation cycliques hiérarchique ;



- Chaque service a **zéro ou un** service hiérarchiquement supérieur
- Chaque service a **zéro, un ou plusieurs** services subordonnés

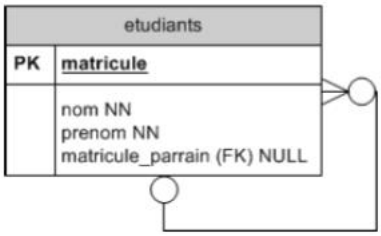
Relation cyclique non hiérarchique ;

Une pièce peut être composée de plusieurs composants et une pièce peut faire partie de plusieurs composants.



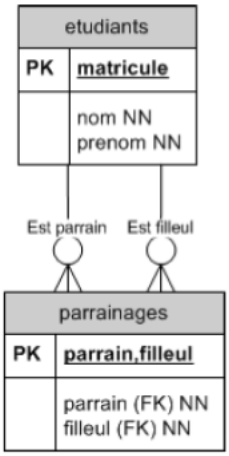
Parrainage : relation cyclique hiérarchique ;

- Un étudiant (filleul) peut être parrainé (ou non) par un autre étudiant
- Un étudiant (parrain) peut parrainer (ou non) plusieurs autres étudiants



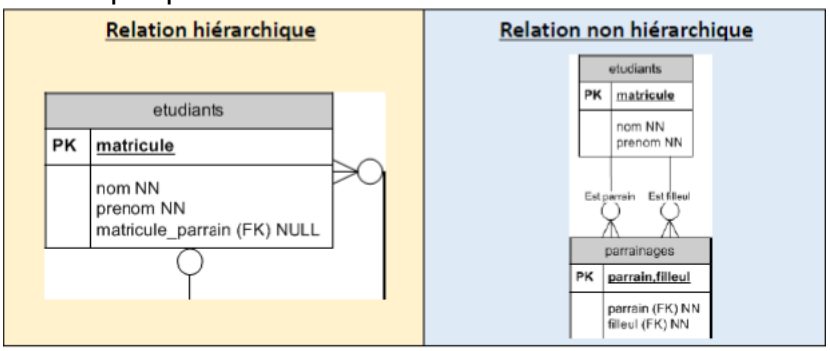
Parrainage : relation cyclique non hiérarchique ;

- Un étudiant (filleul) peut être parrainé par plusieurs autres étudiants
- Un étudiant (parrain) peut parrainer plusieurs autres étudiants

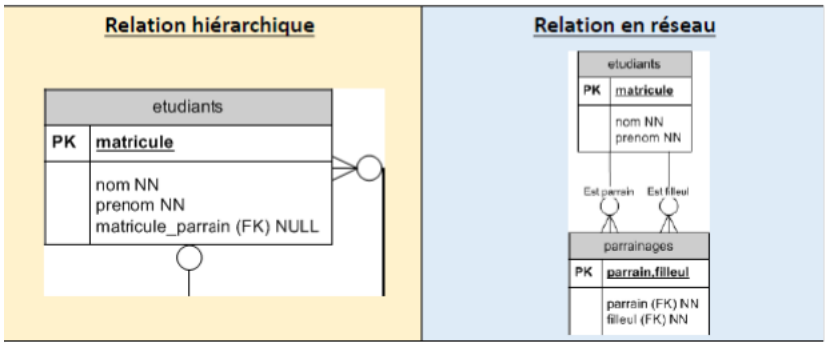


Parrainage : SELECT ;

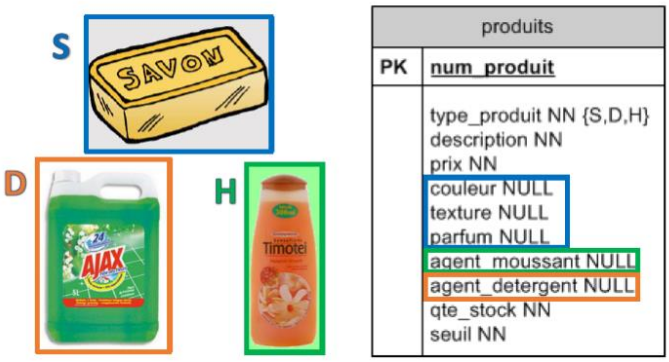
- Pour chaque parrain (matricule, nom et prénom), affichez le nombre de filleuls qu'il possède.



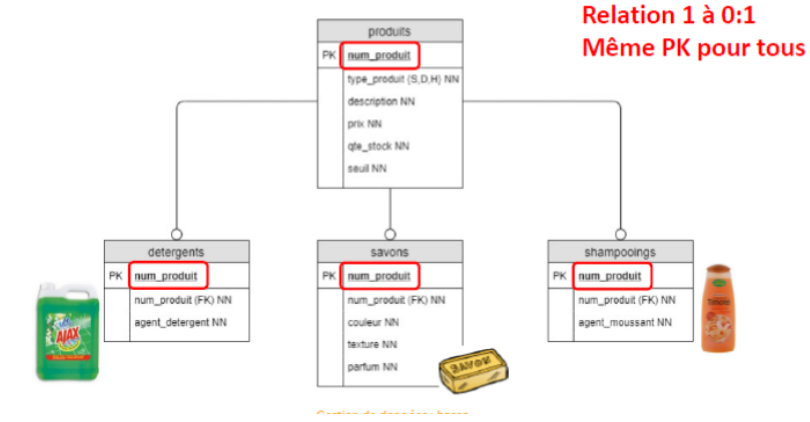
- Pour chaque parrain (matricule, nom et prénom), affichez ses filleuls (matricule, nom, prénom).

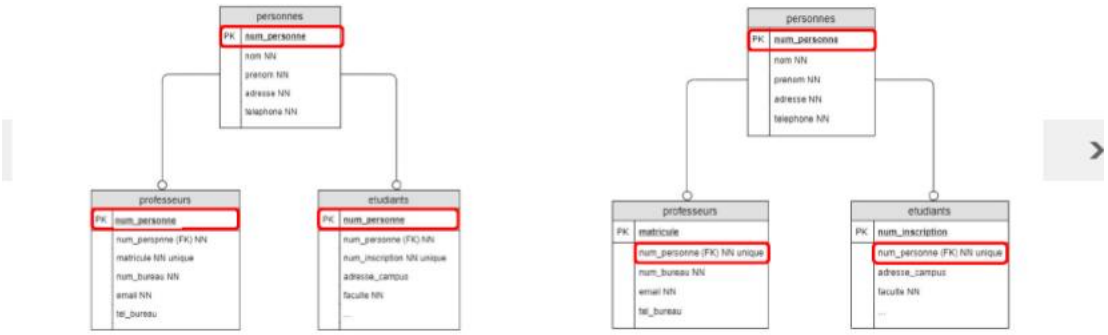


Tables spécialisées ;



- Problème : les propriétés encadrées dépendent du type de produit (savon, shampooing, détergent)
- Elles ne sont pas communes à tous les produits
- Il y a beaucoup de champs qui seront NULL





Attention ! Les étudiants et les professeurs ont une clé unique distincte.
Deux solutions peuvent être implémentées.

Sauvegarde des données ;

- Mettre en sécurité les données contenues dans un système informatique
  - Back-up
  - Archivage

Back-up ;

- Pourquoi ?
  - Permettre de restaurer un système informatique dans un état de fonctionnement par suite d'un incident
  - Faciliter la restauration d'une partie d'un système informatique
- Comment ?
  - Copie des données sur un support indépendant du système initial
- La restauration = opération inverse, càd réutiliser des données sauvegardées.

Archivage ;

- Pourquoi ?
  - Désengorger des tables de mouvements, souvent volumineuses et rapidement périmées.
  - Raisons légales.
  - Statistiques.