UML et Relationnel : Principes de Transformation

UML et Relationnel

UML

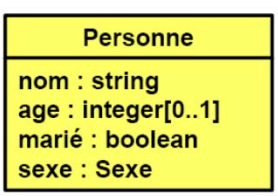
- Type de base
- Classe
- Attribut
- Methode
- Association
- Généralisation
- Contrainte

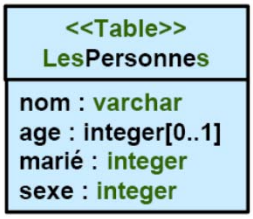
Relationnel

- Type de base
- Table
- Colonne
- Contrainte
  - clé primaire (primary key)
  - clé (constraint C unique)
  - clé étrangère (foreign key)
  - not null ()
- Schéma physique
  - index
  - cluster
  - StoredProcedure

Le "modèle" relationnel est pauvre et de bas niveau

Classes et Attributs

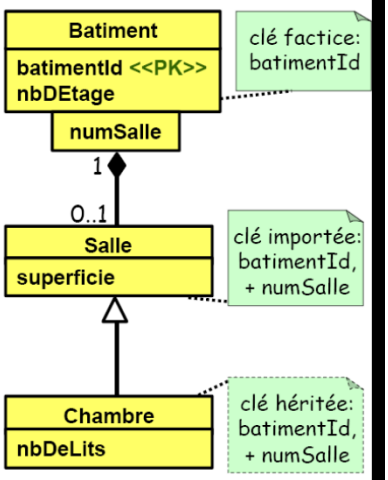




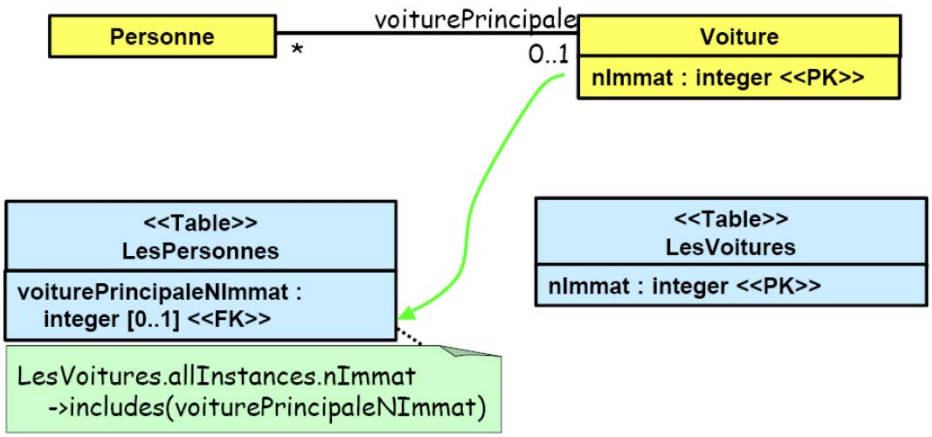
- Une table par classe (sauf généralisation)
- Nom de la table = "Les"+nom de classe+"s"
- Attributs multivalués -> table
- Attributs obligatoire -> contrainte not null
- Traduction des types de bases
  - boolean -> integer
  - string -> varchar
  - enumération -> integer

Identification des clés

- **Pas nécessairement de clé**
  - clé naturelle (e.g. numeroDeCommande)
  - clé factice (e.g. rectangleId)
  - clé importée
  - clé héritée
- **Indiquée avec un stéréotype**
  - Objecteering {PrimaryKey(rank)}
  - <<PK>>, {key=nom, order=n}
  - <<AK>> (Alternative Key)
- **Contrainte**
  - Class.allInstances
    ->isUnique(Tuple(k1=a1,k2=a2, …,))

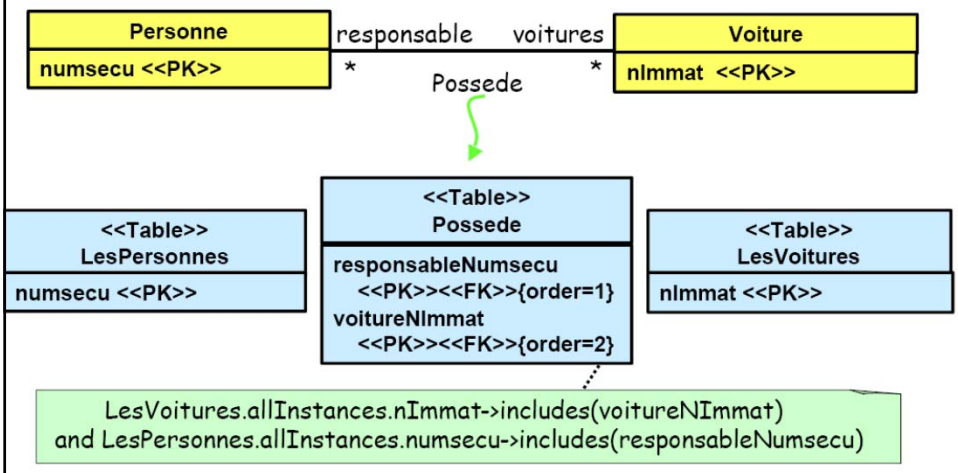


Traduction des Associations * -- 1

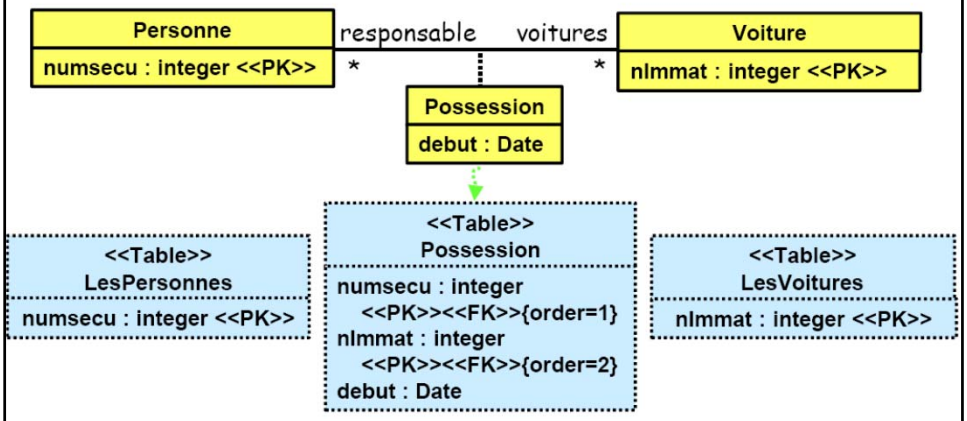


- recopie de la clé étrangère
- pas de problème de navigabilité

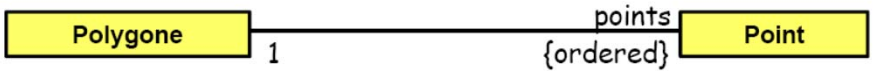
Traduction des Associations * -- *



Traduction des Classes Associatives



Traduction de {ordered}

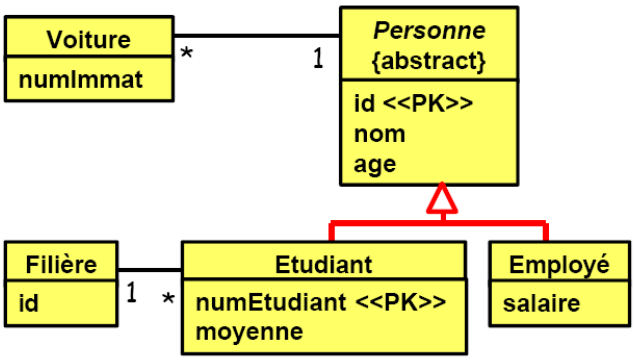


- pas d'ordre en relationnel
- diverses implémentations ni triviales, ni régulières ...

- requêtes complexes et difficile à comprendre
- relationel = niveau assembleur

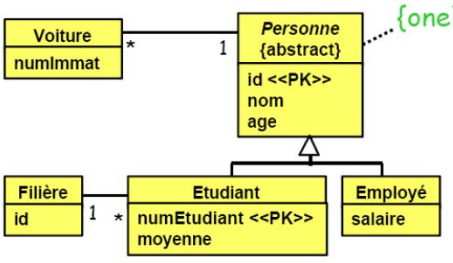
Traduction de la Généralization

- pas de généralisation en relationnel

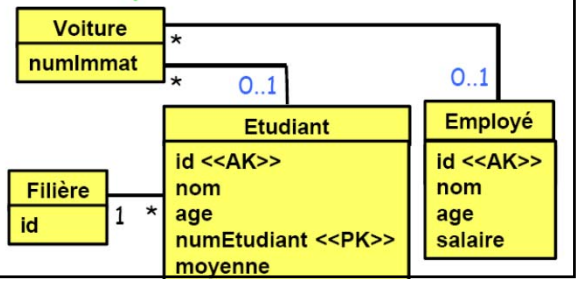


- **trois stratéges pour l'éliminer** :
  - "héritage" {oneTablePerConcreteClass}
  - "éclatement" {oneTablePerClass}
  - "union" {oneTable}

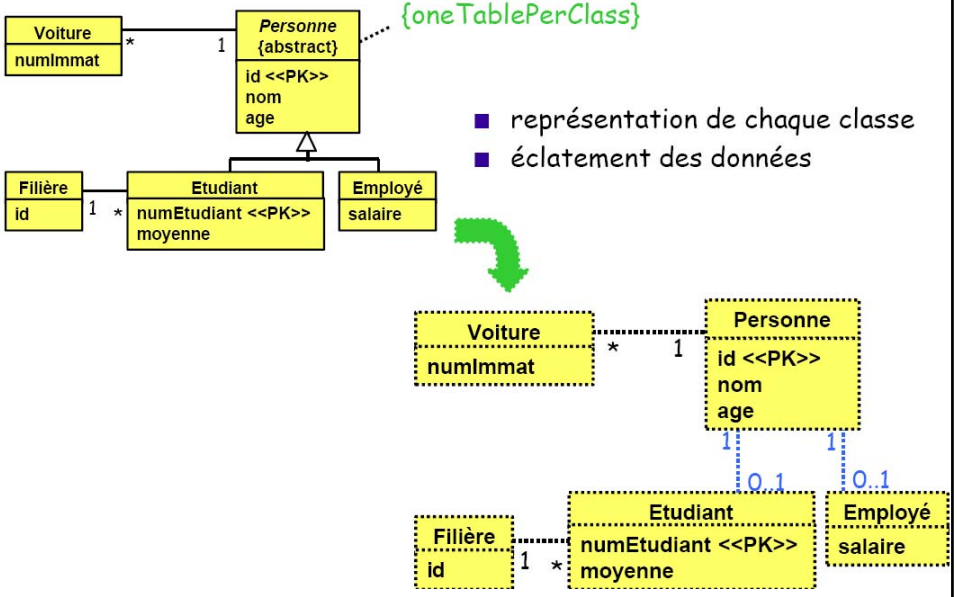
"Mise à plat" de la Généralisation



- élimination des classes abstraites
- "push-down" des attributs hérités

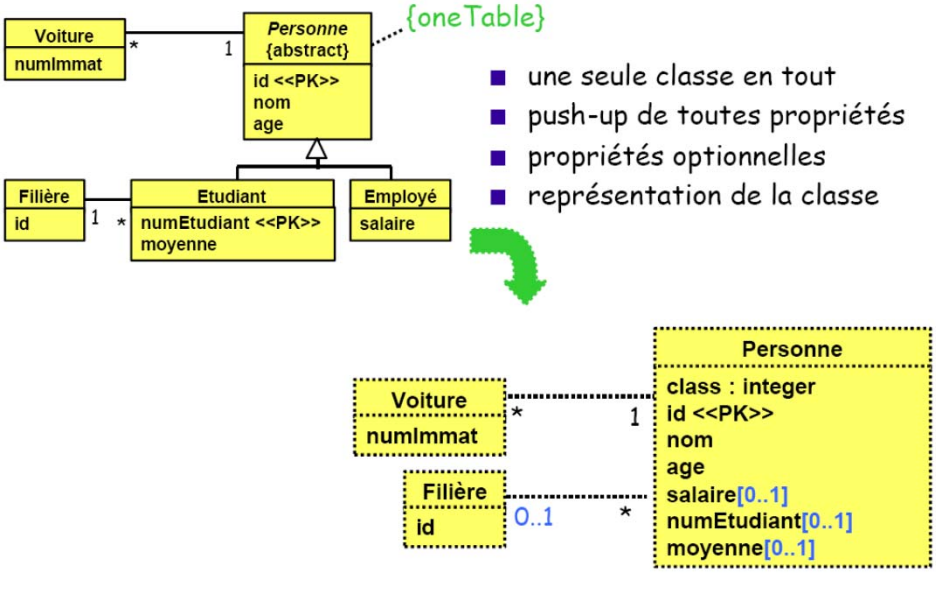


"Eclatement" de la Généralisation



- représentation de chaque classe
- éclatement des données

"Union" de la Généralisation



- une seule classe en tout
- push-up de toutes propriétés
- propriétés optionnelles
- représentation de la classe

Conclusion

- **co-transformation**
  modèle-contraintes <-> schéma-requêtes-trigger
- "modèle" relationnel = niveau assembleur
- très populaire dans l'industrie
- schémas complexes difficiles à comprendre
- transformation UML <-> relationnel
  - certaines "conventionnelles"
  - d'autres plus complexes
- ingénierie vs. rétro-ingénierie
- différents livres, différents outils

Sous Rational

- Activer le data Modeler sous Rational Rose
  – Add-ins/Addin-Manager
- Créer un ou des packages
- Mettre les classes dans leur package
- Mettre les classes en persistantes
- Utiliser le Data Modeler
  – Bouton droit sur le package dans l'inspecteur d'objet puis Transform to Data