Factorisation

Introductions aux Bases de Données

Nathanaël Martel

Factorisation

$$Ax + Bx = x(A + B)$$

Factorisation

Imaginez une table «client» contenant des adresses :

Client
Nom Prénom Téléphone Adresse Complément d’adresse Code postale Ville Pays

Factorisation

Avec cette structure :

- **Un client n’a qu’une seule adresse**
- **S’il change d’adresse, il n’est pas possible de conserver l’ancienne**
- **Si l’adresse n’est pas renseigner, cela fait beaucoup de champs vide**

Factorisation

Factorisation, étape 1 :

- **Stocker les adresses à part**

Client
Nom Prénom Téléphone

Adresse
Adresse Complément d'adresse Code postale Ville Pays

Factorisation

Avec cette structure :

- **Plusieurs adresses par client,**
- **Y compris des adresses obsolète**
- **Il y a un enregistrement uniquement pour les client qui ont une adresse enregistré**

Factorisation

Supposons, que nous ayons des clients qui habitent quasiment tous dans la même ville :

- **Le pays est toujours le même**
- **La ville est «quasiment» toujours la même**

Factorisation

Factorisation, étape 2 :

- **Stocker les adresses à part**

Client
Nom Prénom Téléphone

Adresse
Adresse Complément d'adresse Code postale

Ville
Ville

Pays
Pays

Factorisation

Avantage :

- **Optimisez la taille de la base**
- **Permettre à l'utilisateur de faire des choix plus facilement**
- **Valider la cohérence des données**
  - Une Ville est dans un seule Pays (idem code postale)
  - Il est possible de vérifier l'existence d'une ville / pays

Factorisation

Inconvénient :

- **Requêtes plus compliqué**
  - jointures…
- **Maintenance de la base plus compliqué**
  - La simple lecture d'une table ne suffit pas toujours à comprendre son fonctionnement

Factorisation

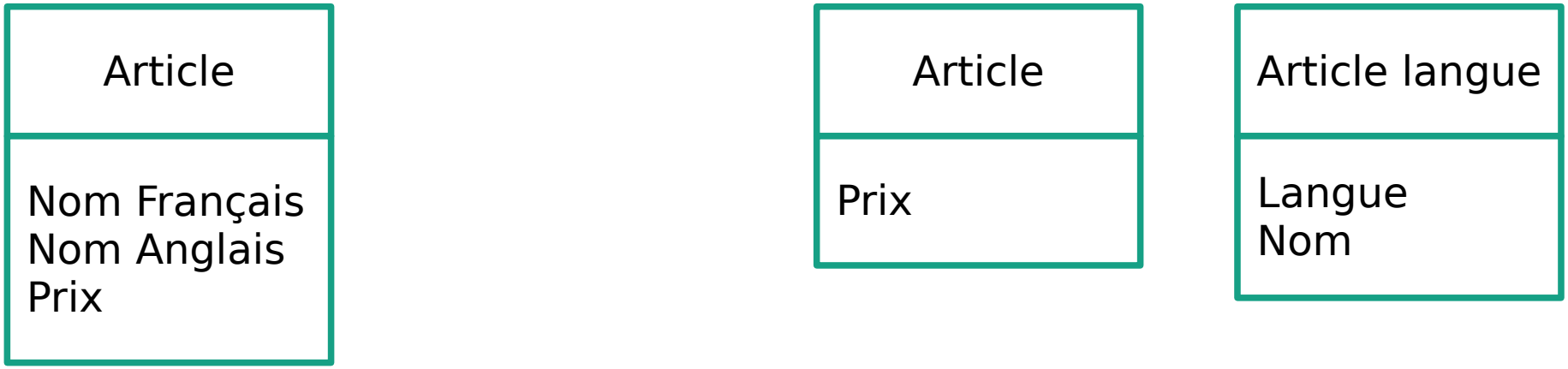
Requêtes pour avoir le pays d'un client :

- `SELECT * FROM client ;`
-
```
SELECT * FROM client
LEFT OUTER JOIN adresse
ON (client.adresse_id=adresse.id)
LEFT OUTER JOIN ville ON
(adresse.ville_id=ville.id)
LEFT OUTER JOIN pays ON
(ville.pays_id=pays.id)
```

Factorisation

Autre utilisation : Internationalisation (i18n).

Si les enregistrements existent en plusieurs langue, on va stocker les textes dans une table à part.



Factorisation

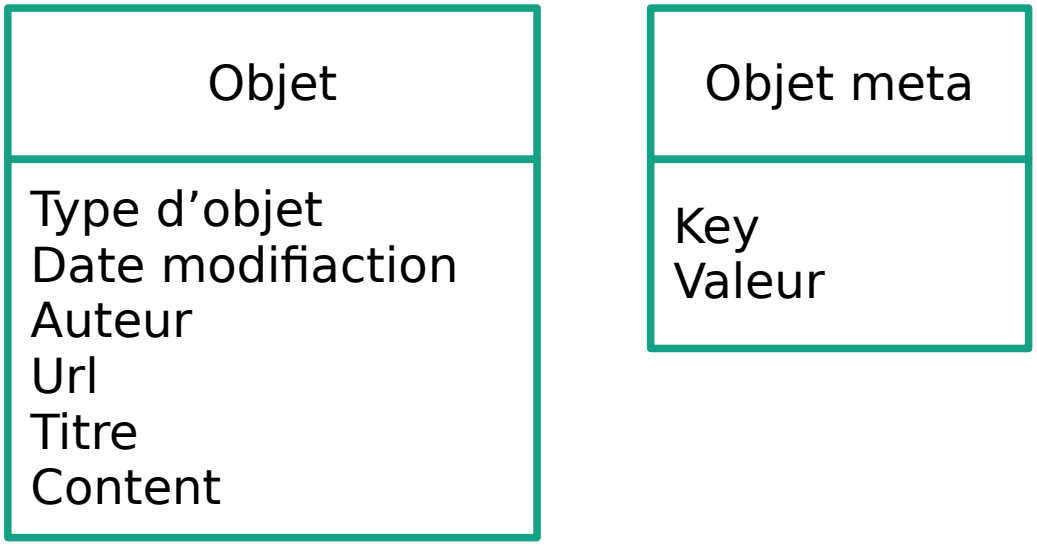
Avantage :

- **Pas besoins de faire modifier la structure de la table pour ajouter une langue**

Factorisation

Utilisation extrême :

Tout stocker dans une même table



Factorisation

Exemples, l'objet peut être :

- **Un billet de blog**
  - les metas ne seront peut être pas utilisé
- **Un produits à vendre**
  - Les metas contiendrons le prix, le stock, le code bar
- **Une commande**
  - Les metas contiendrons les informations du client (adresses facturation / livraison), les produits de la commande, le montant total…

Factorisation

Avantage :

- **Très grande souplesse**

Inconvénient :

- **Niveau d'abstraction supplémentaire**
  - La modélisation «métier» ne correspond pas aux tables de la base de données
- **Difficile à maintenir**
- **Certaines requêtes sont très difficile à faire :**
  - tous les produits dont le stock est inférieur à 5

Factorisation

Utilisations :

- **Certaine informations redondante (adresse)**
- **Internationalisation**
  - Ajout de langue sans toucher à la structure
- **CMS**
  - On peut faire rapidement une interface qui gère tous les types d'objet : la structure étant la même, il n'y a qu'un seule code.

Conclusion

La factorisation est un équilibre à trouver.