Bases de données

Bases de données relationnelles

Une **base de données** est un ensemble d'informations structurées mémorisées sur un support permanent (serveur de base de données). Par exemple dans le cas d'une gestion d'entreprise, de la liste des membres du personnel, pour chaque personne, on enregistre le nom, le prénom, le sexe, la date de naissance, l'adresse, la fonction dans l'entreprise, etc. Toutes ces données ont la même structure, si elles sont gérées par des moyens informatiques, on dit qu'elles constituent une base de données. On utilise aussi le sigle BDD, et le terme anglais correspondant est Data Base.

Les bases de données

Manipulation de données

Le développement des traitements informatiques nécessite la manipulation de données de plus en plus nombreuses. Leur organisation et leur stockage constituent un enjeu essentiel de performance.

Le recours aux bases de données relationnelles est aujourd'hui une solution très répandue. Ces bases de données permettent d'organiser, de stocker, de mettre à jour et d'interroger des données structurées volumineuses utilisées simultanément par différents programmes ou différents utilisateurs. Cela est impossible avec les représentations tabulaires.

Différentes familles de base de données

Il existe différentes familles de bases de données :

- les bases de données hiérarchiques (années 60) : les enregistrements de la base sont liés selon une structure arborescente peu adapté pour décrire des structures complexes, répondre aux besoins réels et suivre l'évolution des systèmes d'information
  exemple : base de registre de Microsoft Windows

- les bases de données relationnelles (1970) : l'information est organisée dans des tableaux à deux dimensions reliés entre eux par des clefs. Ce sont de loin les plus couramment utilisées ...

- les bases de données objet (années 90) : les informations sont groupées sous formes de collections d'objets (conteneur logique qui englobe des informations et des traitements)

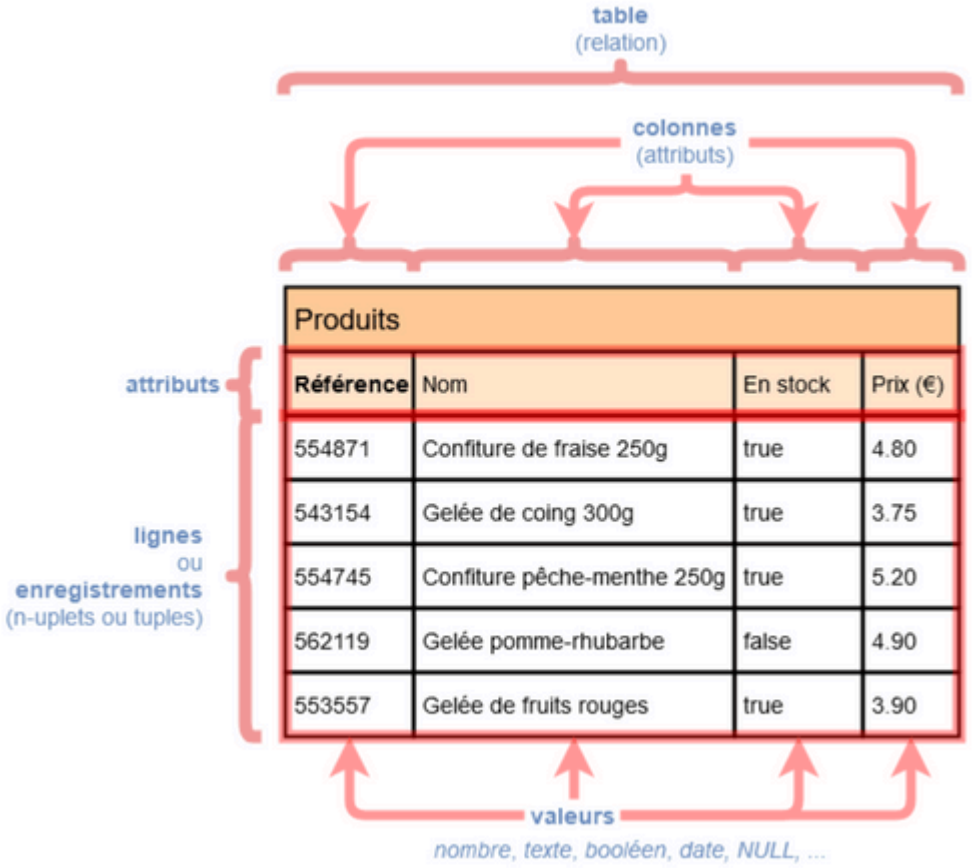
- les bases de données NoSQL (Not only SQL) (2009) : enregistrement en structures flexibles, selon un paradigme non relationnel, favorisant l'interrogation des données adapté aux grosses infrastructures matérielles constituées de nombreux serveurs organisés en clusters (grappes), essentiellement pour des applications Web (cloud, réseaux sociaux, ...) exemples : BigTable de Google, HBase (Facebook), Cassandra (Twitter) ...

Modèle relationnel

Les bases de données relationnelles ont été mises au point en 1970 par **Edgar Franck Codd**, informaticien britannique (1923-2003). Ces bases de données sont basées sur la théorie mathématique des ensembles.

Les relations

> 📢 Définition : Le modèle de données relationnel représente une base de données comme un ensemble de **relations** (tables à deux dimensions) qui contiennent :
>
> - un ensemble de **n-uplets ou tuples** (les lignes – ou enregistrements – de la table) dont les entrées (des valeurs qui peuvent être de type nombre, texte, booléen ou bien NULL) appartiennent à un domaine.
> - des **attributs** (les colonnes de la table), dont les noms peuvent servir de clefs.



> 📌 définition : **Ordre ou degré d'une relation**
> On appelle ordre (ou degré) le nombre d'attributs d'une relation.

Par exemple : la relation Produits est d'ordre 4.

> 📌 définition : **Cardinal d'une relation**
> On appelle cardinal le nombre d'enregistrements d'une relation.

Par exemple : la relation Produits est de cardinal 5.

Dans une même base de données, plusieurs relations peuvent avoir des attributs de même nom. Pour désigner l'attribut *attr1* d'une relation *rel1* , on le notera : *rel1.attr1*

Par exemple : Produits.Référence

Domaines

> 📌 définition : **Le domaine** d'un attribut est un ensemble, fini ou infini, de valeurs admissibles.

Par exemple :

- le domaine de l'attribut « Prix » est l'ensemble des nombres à virgule flottante (FLOAT) exprimés avec deux chiffres après la virgule.

- le domaine de l'attribut « Nom » correspond à l'ensemble des chaînes de caractères (TEXT).

Le domaine d'un attribut est renseigné au moment de la création de la relation.

Relations et clés

Les relations d'une base sont reliées (ou « connectées ») par **certaines des valeurs** qu'elles contiennent : chaque enregistrement d'une table contient un groupe d'informations relatives à un sujet et les différents sujets sont connexes.

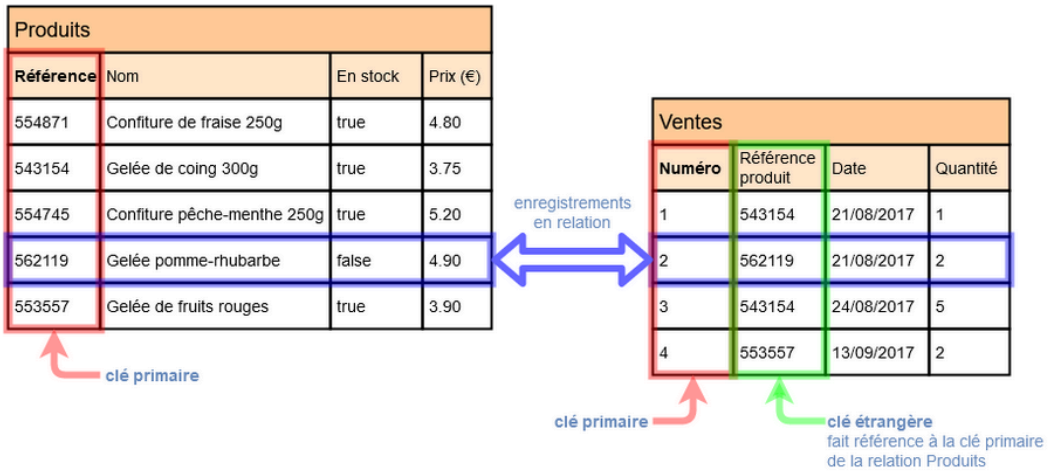
Par exemple : les relations Produits et Ventes sont connectées puisque l'objectif de la base de données est de gérer la vente de produits !

Ces liens existants entre les informations sont stockés dans les champs (attributs) des enregistrements sous forme de **clés**.

> 📢 définition : Pour une relation donnée, une clé est un groupe d'attributs permettant d'identifier un unique enregistrement de la relation.

```
Pour qu'un groupe d'attributs forme une clé, il faut être sûr que de
```

On parle de clé candidate si le groupe d'attributs de la clé est minimal, c'est-à-dire que si on retire un seul des attributs de ce groupe, l'unicité n'est plus vérifiée.



> 📢 **clé primaire** : c'est l'une des clés candidates de la relation, choisie pour être utilisée comme clé étrangère dans une autre relation.

> 📢 **clé étrangère** : attribut d'une relation dont les valeurs sont des références à une clé primaire d'une autre relation.

Schéma relationnel

Les tables constituent la structure logique du modèle.

--> Le schéma d'une relation précise le **nom de la relation** ainsi que l'ensemble des **attributs** .

Exemple :

```
Produits (Référence, Nom, En stock, Prix)
```

--> Le **schéma relationnel** d'une base de données est l'**ensemble** des schémas des relations qui la composent.

Exemple :

```
Produits (Référence, Nom, En stock, Prix)
Ventes (Numéro, Référence produit, Date, Quantité)
```

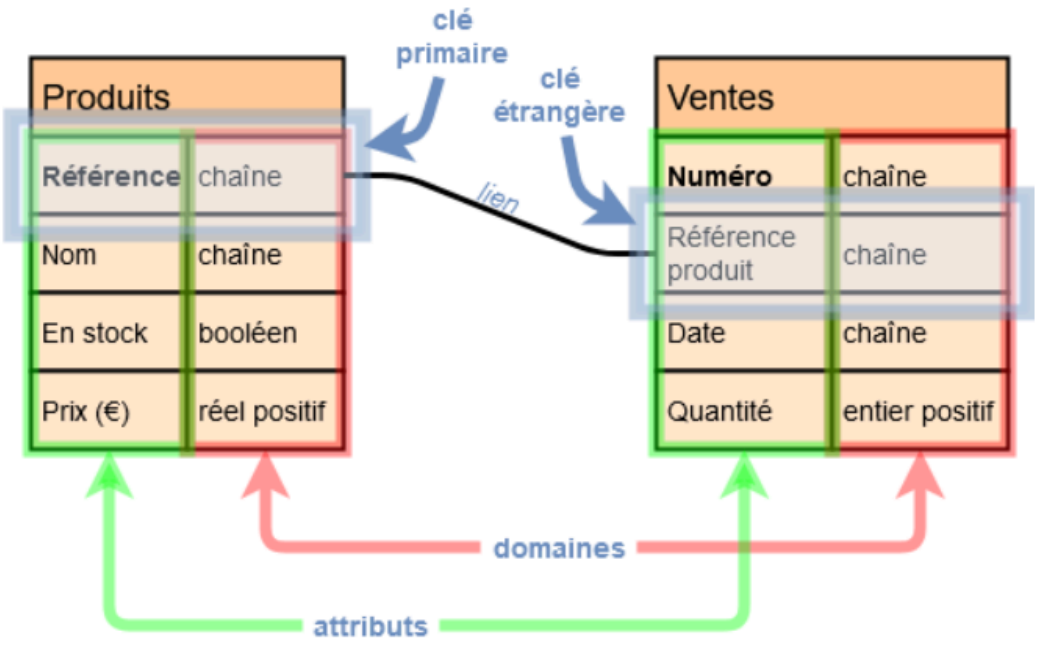
Il est possible de faire apparaitre sur un schéma relationnel les clés primaires en les soulignant et étrangères précédées (ou suivies) d'un '#' des relations :

On peut également préciser les domaines des différents attributs :

Exemple :

```
Produits (Référence : TEXT, Nom : TEXT, En stock : BOOL, Prix :
Ventes (Numéro : INT, #Référence produit : TEXT, Date : DATE, Qu
```

On peut représenter un schéma de relation sous une forme graphique, par exemple à l'aide d'un diagramme :



Base de données relationnelle

Pour gérer une base de données, il faut utiliser une suite de programmes appelé **système de gestion de base de données**.

Des systèmes de gestion de bases de données (SGBD) de très grande taille (de l'ordre du pétaoctet) sont au centre de nombreux dispositifs de collecte, de stockage et de production d'informations. Les SGBD permettent :

- **de gérer la lecture, l'écriture ou la modification** des informations de la base
- **de gérer les autorisations d'accès à la base** : un utilisateur peut être autorisé à lire et écrire dans la base de données alors qu'un autre utilisateur peut n'avoir la possibilité que de de lire les informations
- **de gérer les accès concurrents à la base** : plusieurs personnes peuvent avoir besoin d'accéder en même temps aux informations contenues dans une base données, ce qui peut parfois poser problème, notamment lorsqu'il s'agit de modifier la même donnée au même moment
- **d'assurer la maintenance de la base** : les fichiers d'une base de données sont stockés sur des disques durs dans des ordinateurs. Pour palier aux éventuelles pannes matérielles, une même base de données est dupliquée sur plusieurs ordinateurs. Toutes les copies de la base doivent être parfaitement synchronisées, et ce le plus rapidement possible.

Pour préserver l'intégrité d'une base de données, le SGBD se charge de vérifier que toutes les valeurs d'une clef étrangère d'une relation correspondent bien à des valeurs présentes dans la clef primaire de l'autre relation.

\newpage

Contraintes d'intégrité

> 📢 Définition: Une contrainte d'intégrité est une règle, établie à la création d'une base de données, qui définit la cohérence des données de la base.

Une contrainte peut agir au niveau de la valeur d'une donnée, de la relation où elle est enregistrée, ou encore d'un ensemble de relations.

Contrainte de domaine

> Elle est définie par le **domaine** que l'on spécifie pour chaque attribut d'une relation.

Le choix du domaine est important pour assurer l'intégrité des données.

```
Exemple, pour l'attribut Prix de la relation Produits :

Si on choisit le domaine FLOAT, on autorise la possibilité d'avoir c

Il faut préférer le domaine DECIMAL(5,2) (ensemble des nombres décim
```

Selon les SGBD, on pourra également limiter les nombres à des entiers positifs, les chaînes de caractère à un ensemble fini de mots, ...

Contrainte de relation

> Il faut spécifier **une clé primaire pour chaque relation** d'une base de données. Ainsi le SGBD pourra s'assurer qu'il sera possible avec cette clé d'accéder à un unique enregistrement, et interdira la création de doublons.

```
Exemple : le SGBD renverra un message d'erreur si l'on tente de crée
```

Si on omet de spécifier une clé primaire pour une relation, l'application qui exploite les données pourrait avoir de graves dysfonctionnements !

Contrainte de clé étrangère

Les clés étrangères d'une relation doivent être définie lors de la création de la relation. Ainsi, le SGBD sera chargé de vérifier qu'une valeur de clé étrangère correspond bien à une des valeurs de clé primaire de la table à laquelle elle se réfère.

Exemple :

```
Chaque valeur de l'attribut `Référence` produit de la relation `Vent
```

Si on tente de donner la valeur 536124 à à la Référence produit d'une vente, cela provoquera une erreur !

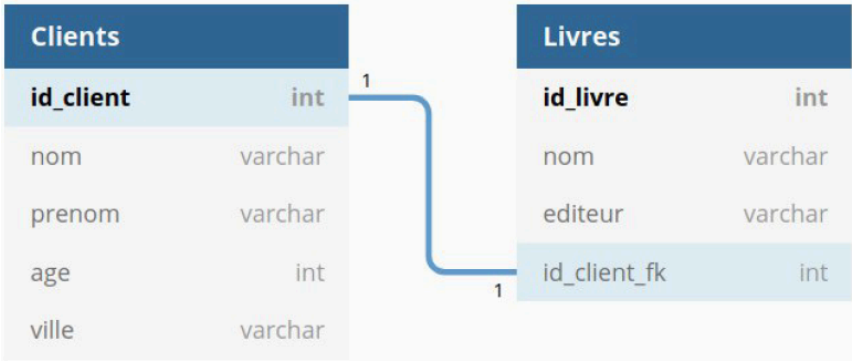
Contrainte d'intégrité référentielle

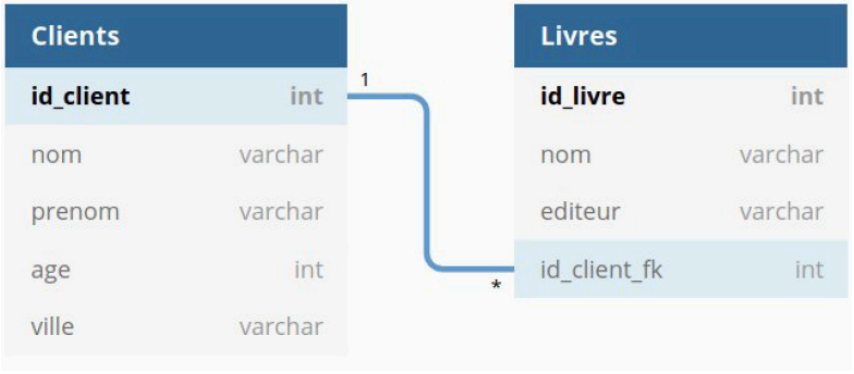
Elle est définie par trois règles :

- une clé étrangère doit être une valeur qui fait partie des valeurs de clé primaire de la relation à laquelle elle se réfère
- un enregistrement d'une relation ne peut pas être supprimé s'il possède des enregistrements liés
- la clé primaire d'un enregistrement d'une relation ne peut pas être modifié s'il possède des enregistrements liés

Liens entre les tables

Les tables peuvent être reliées entre elles à l'aide des contraintes de référence présentées au paragraphe précédent.
Il existe 3 types de relations entre les tables illustrées ci-dessous :

Relation	Description
1-1 : one-to-one	 La clef étrangère peut se situer indifféremment dans l'une ou l'autre des tables

1-N : one-to-many	 La clef étrangère est du côté du “many”
N-M : many-to-many	 Il faut ajouter une nouvelle table entre Client et Livres avec 2 clefs étrangères afin de pouvoir faire 2 relations one-to-many