Introduction aux bases de données

Les données numériques et leur gestion sont omniprésentes dans de nombreuses activités et ce depuis plusieurs décennies. Ces données sont en très grand volume, l'enjeu est donc de les stocker correctement pour en faciliter l'accès et le traitement (mise à jour, recherche, etc.).

Exemple 1

Dans la suite nous prendrons l'exemple d'un CDI qui souhaite gérer l'emprunt de ses livres. Une façon naïve de faire consiste à utiliser un tableur dont voici un extrait (on a limité volontairement le nombre de colonnes mais on voudrait pouvoir rajouter l'année de parution, le nombre de pages, l'éditeur, la date d'emprunt, etc.) :

titre	**auteur**	**nom_emp**	**prenom_emp**	**mail**	**date_retour**
Traité théologico-politique	Spinoza	Doe	John	`jd@mail.org`	2024-11-09
Le temps de l'innocence	Wharthon	Bernard	Victor	`b.v@mail.net`	2024-11-06
Les suppliantes	Eschyle	Dehaze	Olivier	`deho@mail.fr`	2024-10-20
Les sept contre Thèbes	Eschyle	Souvarine	Boris	`bo.sou@mail.io`	2024-10-19
Le temps de l'innocence	Wharthon	Dehaze	Olivier	`deho@mail.fr`	2024-10-13

On voit que ce stockage n'est pas du tout efficace : de nombreuses informations sont redondantes (nom d'un auteur, d'un emprunteur, plusieurs lignes pour un même livre emprunté plusieurs fois). Cela entraîne des difficultés de mise à jour (que faire si un emprunteur change d'adresse mail ?) et des risques de pertes d'information (si on supprime une ligne, où sont stockées les données d'un emprunteur ?).

Pour pallier ces problèmes, on utilisera plutôt une *base de données*.

1 Description d'une base de données

1.1 Les données

Les données d'une base de données sont stockées sous forme de tableaux, appelés *tables* ou *relations*. Une base de données peut contenir un nombre quelconque de tables.
Chaque table est constituée de plusieurs colonnes, appelées aussi *attributs* ou *champs*. Ceux-ci sont fixes.
De plus, les données dans une colonne sont toutes du même *type* (entier, flottant, texte) et ont des valeurs limites (valeur maximale d'un entier, longueur maximale d'une chaîne de caractères, etc.). On dit que les valeurs permises dans une colonne appartiennent à un *domaine*.

On choisira toujours de représenter les dates sous forme d'une chaîne de caractères au format `AAAA-MM-JJ`. Cela est suffisant pour effectuer facilement des comparaisons puisque l'ordre lexicographique correspond alors à celui de l'ordre « naturel » du temps.

Enfin, les lignes d'une table sont appelées *enregistrements* : chacune est un n-uplet de valeurs où n est le nombre de colonnes.

Exemple 2

Le CDI pourrait disposer d'une base de données constituée notamment de :

- la table `Livres` avec les attributs : `titre`, `auteur`, `année_publication`, `éditeur`, `nb_pages` ;
- la table `Élèves` avec les attributs : `prénom`, `nom`, `mail`.

Un enregistrement de la table `Élèves` serait par exemple (John, Doe, `jd@mail.org`).

Cela permet clairement de réduire la redondance [1]. Cependant ces deux tables ne contiennent pas les informations concernant les emprunts. Ne souhaitant pas les inscrire dans l'une des deux tables pour éviter les problèmes précédents, on va donc créer une troisième table `Emprunts` pour les stocker. Mais alors comment faire sans recopier des informations déjà présentes dans les deux tables `Livres` et `Élèves` ?

1. Même si on pourrait faire encore mieux en créant une table `Auteur` à part pour éviter des redondances pour les différents livres d'un même écrivain. Il est important de comprendre que la modélisation effectuée dépend des besoins.

1.2 Les liens entre elles

Dans une base de données, les différentes tables sont reliées entre elles afin de partager des informations tout en limitant la redondance.
En général, même si ce n'est pas obligatoire, cela passe par l'ajout d'un attribut supplémentaire, de type entier, souvent intitulé `id` (pour identifiant), permettant d'identifier de manière unique un enregistrement d'une table.

Exemple 3

Pour le CDI, on aurait alors :

- la table `Livres` avec les attributs : `id`, `titre`, `auteur`, `année_publication`, `éditeur`, `nb_pages` ;
- la table `Élèves` avec les attributs : `id`, `prénom`, `nom`, `mail` ;
- la table `Emprunts` avec les attributs : `id`, `id_livre`, `id_eleve`, `date_emprunt`, `date_retour`.

Définition 4 – Clé primaire

On appelle *clé primaire* un attribut, ou groupe d'attributs, qui permet d'identifier de manière unique un enregistrement.

Exemple 5

Pour la table `Livres`, l'attribut `id` constitue une clé primaire. Le couple (`titre`, `auteur`) pourrait aussi constituer une clé primaire (car on peut raisonnablement estimer qu'un même écrivain ne publie pas deux livres avec exactement le même titre) mais on ne peut en choisir qu'une, prenons la plus simple.

Remarque. Lorsqu'on présente une base de données, pour mettre en avant cette clé primaire, le nom de l'attribut correspondant est parfois souligné ou mis en gras.

Définition 6 – Clé étrangère

On appelle *clé étrangère* un attribut qui est une clé primaire d'une autre table. Elle permet de relier deux tables de manière cohérente.

Exemple 7

Dans la table `Emprunts`, l'attribut `id_livre` est une clé étrangère (c'est une clé primaire de la table `Livres`), de même pour `id_emprunteur` (clé primaire de `Élèves`).

Enfin, ces tables et les liens entre elles sont parfois représentés sous forme de diagramme :