Bases de données

Table des matières

1) Introduction aux bases de données

1.1) Introduction

Les données et la gestion des données sont devenues des enjeux très importants. La quantité de données est gigantesque. Il faut les stocker et les rendre disponibles au plus grand nombre. Depuis plusieurs années, l'utilisation des bases de données relationnelles est la solution.

1.2) Définition d'une base de donnée

Une base de données stocke des informations en rapport avec une activité. Ces informations peuvent être de natures très hétérogènes. Les informations sont structurées et cette structure permet d'insérer, de supprimer, de mettre à jour et d'interroger les informations contenues.

Exemple 1 de base de données :

Vous avez un de nombreux livres et vous voulez stocker des informations à propos de ces livres (auteur, pays, date, édition, etc....)

Exemple 2 de base de données :

Vous disposez chez vous de nombreuses cartes Arduino, capteurs, composants électroniques diverses et variés et vous voulez avoir un inventaire précis de ce que vous avez.

Un Système de gestion de Base de Données (SGBD) peut être vu comme le logiciel qui prend en charge la structuration, le stockage, la mise à jour et la maintenance des données. C'est en fait l'interface entre la base de données et les multiples utilisateurs (ou leurs programmes).

1.3) Historique

Quasiment toutes les bases de données que nous utilisons aujourd'hui sont basées sur les travaux de Edgar Frank Codd (1970). C'est le point essentiel dans l'histoire des bases de données.

On peut établir la chronologie représentée en Figure 1 ainsi que regarder la vidéo.

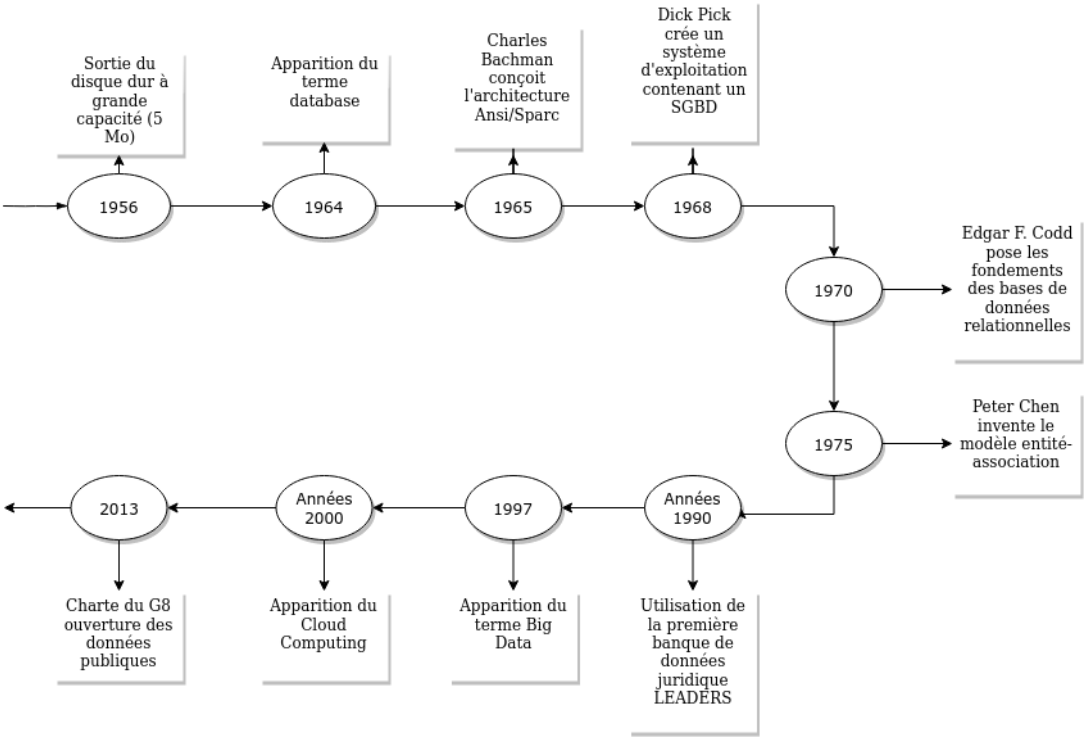
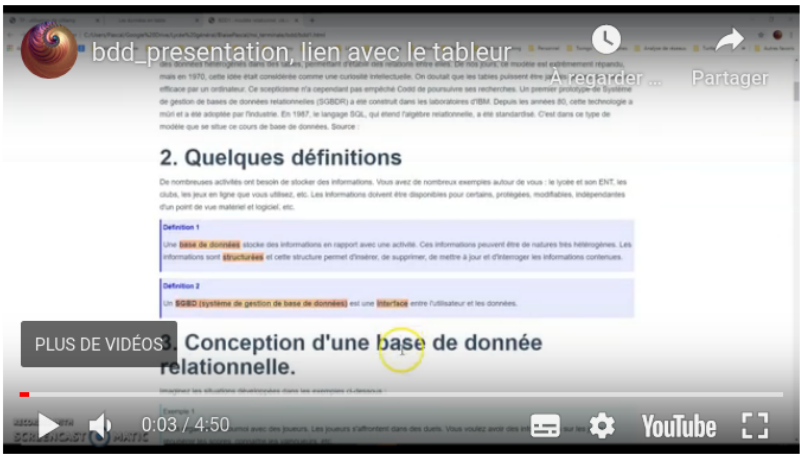


FIGURE 1 – Frise chronologique

1.4) Pourquoi ne pas utiliser un tableur pour réaliser une base de données



Exercice 1 : Regarder la vidéo ci-dessus. En déduire les avantages et les inconvénients du tableur pour réaliser une base de données.

2) Conception des bases de données relationnelles

2.1) Notion de relation, d'attribut et de de domaine d'attribut

La notion de relation est au cœur des bases de données relationnelles. Une relation peut être vue comme un tableau à 2 dimensions, composé d'une en-tête et d'un corps. Le corps est lui-même composé de t-uplets (lignes) et d'attributs (colonnes). L'en-tête contient les intitulés des attributs de l'entité LIVRE. Le corps contient les données proprement dites. À noter que l'on emploie aussi le terme "table" à la place de "relation". Un exemple de relation peut se voir à la Figure 2

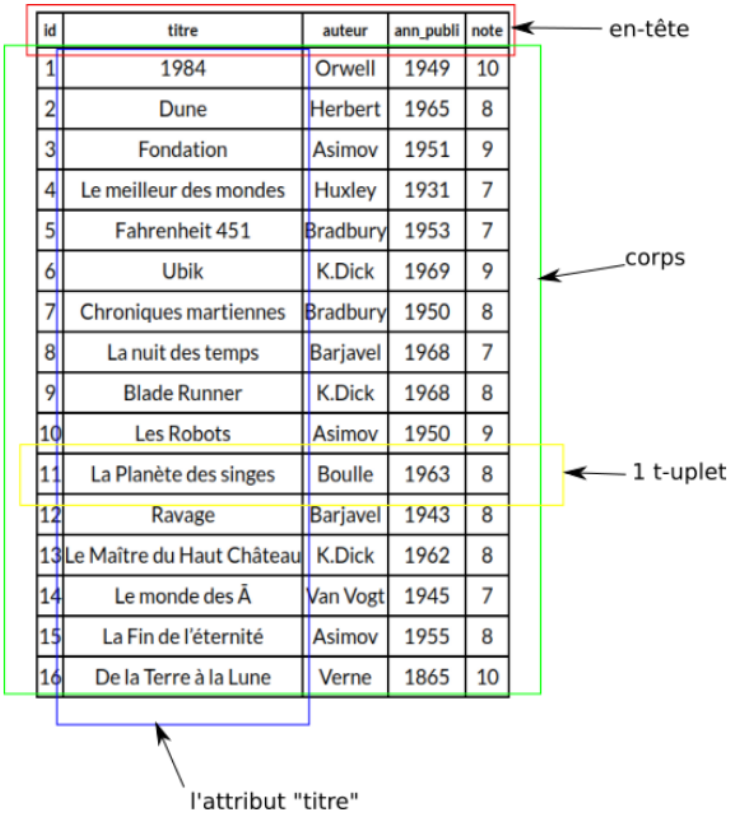


FIGURE 2 – Entité livre

Le t-uplet encadré en jaune sur le schéma ci-dessus contient les valeurs (ou éléments) suivantes : 11, La Planète des singes, Boulle, 1963 et 8. L'attribut "titre" est composé des valeurs suivantes : 1984, Dune, Fondation, Le meilleur des mondes, Fahrenheit 451, Ubik, Chroniques martiennes, La nuit des temps, Blade Runner, Les Robots, La Planète des singes, Ravage, Le Maître du Haut Château, Le monde des Ā, La Fin de l'éternité et De la Terre à la Lune.

Pour chaque attribut d'une relation, il est nécessaire de définir un **domaine** : Le domaine d'un attribut donné correspond à un ensemble fini ou infini de valeurs admissibles.

Par exemple : le domaine de l'attribut "id" correspond à l'ensemble des entiers (la colonne "id" devra obligatoirement contenir des entiers).

Autre exemple : le domaine de l'attribut "titre" correspond à l'ensemble des chaînes de caractères . Dernier exemple, le domaine de l'attribut "note" correspond à l'ensemble des entiers positifs.

2.2) Notion de clé primaire

Afin de na pas avoir deux t-uplets identiques, on définit la notion de "clef primaire" (ou Primary Key).

Une clé primaire est un attribut dont la valeur permet d'identifier de manière unique un t-uplet de la relation. Autrement dit, si un attribut est considéré comme clé primaire, on ne doit pas trouver dans toute la relation 2 fois la même valeur pour cet attribut.

Exemple : Dans la relation précédente, l'attribut auteur ne peut pas être une clé primaire ni l'année de publication. Par contre, il n'est pas inconcevable que le titre du livre soit une clé primaire mais en général, on crée un attribut particulier pour jouer le rôle de clé primaire.

2.3) Notion de clé étrangère

Revenons à notre relation "LIVRES". Nous désirons maintenant un peu enrichir cette relation en ajoutant des informations supplémentaires sur les auteurs, nous obtenons alors la relation de la Figure 3.

id	titre	nom_auteur	prenom_auteur	date_nai_auteur	langue_ecriture_auteur	ann_publi	note
1	1984	Orwell	George	1903	anglais	1949	10
2	Dune	Herbert	Frank	1920	anglais	1965	8
3	Fondation	Asimov	Isaac	1920	anglais	1951	9
4	Le meilleur des mondes	Huxley	Aldous	1894	anglais	1931	7
5	Fahrenheit 451	Bradbury	Ray	1920	anglais	1953	7
6	Ubik	K.Dick	Philip	1928	anglais	1969	9
7	Chroniques martiennes	Bradbury	Ray	1920	anglais	1950	8
8	La nuit des temps	Barjavel	René	1911	français	1968	7
9	Blade Runner	K.Dick	Philip	1928	anglais	1968	8
10	Les Robots	Asimov	Isaac	1920	anglais	1950	9
11	La Planète des singes	Boulle	Pierre	1912	français	1963	8
12	Ravage	Barjavel	René	1911	français	1943	8
13	Le Maître du Haut Château	K.Dick	Philip	1928	anglais	1962	8
14	Le monde des Ã	Van Vogt	Alfred Elton	1912	anglais	1945	7
15	La Fin de l'éternité	Asimov	Isaac	1920	anglais	1955	8
16	De la Terre à la Lune	Verne	Jules	1828	français	1865	10

FIGURE 3

Nous avons ajouté 3 attributs ("prenom_auteur", "date_nai_auteur" et "langue_ecriture_auteur"). Nous avons aussi renommé l'attribut "auteur" en "nom_auteur".

Comme vous l'avez peut-être remarqué, il y a pas mal d'informations dupliquées, par exemple, on retrouve 3 fois "K.Dick Philip 1928 anglais", même chose pour "Asimov Isaac 1920 anglais"...Cette duplication est-elle indispensable ? Non ! Est-elle souhaitable ? Non plus ! En effet, dans une base de données, on évite autant que possible de dupliquer l'information (sauf à des fins de sauvegarde, mais ici c'est toute autre chose). Si nous dupliquons autant de données inutilement c'est que notre structure ne doit pas être la bonne ! Mais alors, comment faire pour avoir aussi des informations sur les auteurs des livres ?

La solution est relativement simple : travailler avec 2 relations au lieu d'une seule et créer un "lien" entre ces 2 relations (voir Figure 4).

Nous avons créé une relation AUTEURS et nous avons modifié la relation LIVRES : nous avons remplacé l'attribut "auteur" par un attribut "id_auteur".

Comme vous l'avez sans doute remarqué, l'attribut "id_auteur" de la relation LIVRES permet de créer un lien avec la relation AUTEURS. "id_auteur" correspond à l'attribut "id" de la relation AUTEURS. L'introduction d'une relation AUTEURS et la mise en place de liens entre cette relation et la relation LIVRES permettent d'éviter la redondance d'informations.

id	nom	prenom	ann_naissance	langue_ecriture
1	Orwell	George	1903	anglais
2	Herbert	Frank	1920	anglais
3	Asimov	Isaac	1920	anglais
4	Huxley	Aldous	1894	anglais
5	Bradbury	Ray	1920	anglais
6	K.Dick	Philip	1928	anglais
7	Barjavel	René	1911	français
8	Boulle	Pierre	1912	français
9	Van Vogt	Alfred Elton	1912	anglais
10	Verne	Jules	1828	français

(a) Table Auteur

id	titre	id_auteur	ann_publi	note
1	1984	1	1949	10
2	Dune	2	1965	8
3	Fondation	3	1951	9
4	Le meilleur des mondes	4	1931	7
5	Fahrenheit 451	5	1953	7
6	Ubik	6	1969	9
7	Chroniques martiennes	5	1950	8
8	La nuit des temps	7	1968	7
9	Blade Runner	6	1968	8
10	Les Robots	3	1950	9
11	La Planète des singes	8	1963	8
12	Ravage	7	1943	8
13	Le Maître du Haut Château	6	1962	8
14	Le monde des Ā	9	1945	7
15	La Fin de l'éternité	3	1955	8
16	De la Terre à la Lune	10	1865	10

(b) Table livre

FIGURE 4

Pour établir un lien entre 2 relations RA et RB, on ajoute à RA un attribut x qui prendra les valeurs de la clé primaire de RB. Cet attribut x est appelé **clé étrangère** (ou Foreign Key). Cet attribut correspond à la clé primaire d'une autre table, d'où le nom.

Dans l'exemple ci-dessus, l'attribut "id_auteur" de la relation LIVRES permet bien d'établir un lien entre la relation LIVRES et la relation AUTEURS, "id_auteur" correspond bien à la clé primaire de la relation AUTEURS, conclusion : "id_auteur" est une clé étrangère.

3) Base relationnelle

Dernière définition, on appelle schéma relationnel l'ensemble des relations présentes dans une base de données. Quand on vous demande le schéma relationnel d'une base de données, il est nécessaire de fournir les informations suivantes :

- Les noms des différentes relations pour chaque relation ;
- la liste des attributs avec leur domaine respectif pour chaque relation ;
- la clé primaire et éventuellement la clé étrangère

Voici un exemple pour les relations LIVRES et AUTEURS :

- AUTEURS(id : INT, nom : TEXT, prenom : TEXT, ann_naissance : INT, langue_ecriture : TEXT)
- LIVRES(id : INT, titre : TEXT, #id_auteur : INT, ann_publi : INT, note : INT)

Les attributs soulignés sont des clés primaires, le # signifie que l'on a une clé étrangère.

On peut représenter comme sur la Figure 5 la base de données élaborées précédemment.

Exercice 2 :

Faire l'analyse d'une base de données dans laquelle vous voulez stocker :

- Le nom d'un film à regarder ;

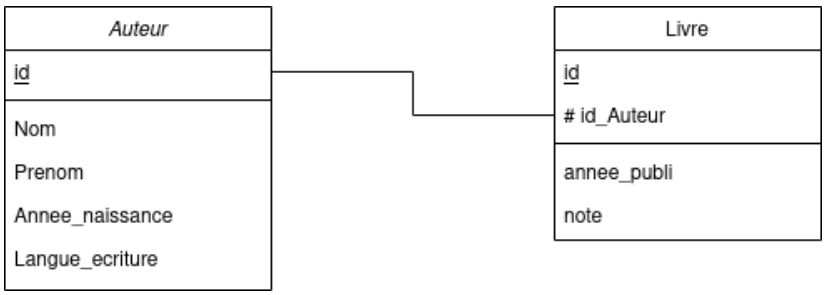


FIGURE 5 – BDD

- Le réalisateur de ce film (Nom, prénom)
- La date de sortie de ce film ;
- L'identité des acteurs principaux (nom, prénom)
- Genre du film (policier, comédie, etc...)
- Un descriptif du film (texte bref)

Questions :

1. Quels objets peut-on mettre en évidence pour cette base de données que l'on pourrait appeler FILM
2. Établir un tableau de vos données.
3. Représenter vos données à l'aide d'une représentation du type du cours.
4. Etablir les liens entre vos données.

Pour représenter une base de données, vous pouvez utiliser soit DrawIO en ligne soit le logiciel yEd en ligne ou à télécharger

4) Les contraintes d'intégrité

Il existe un certain nombres de règles à respecter pour respecter l'intégrité d'une base de données. Ces règles visent à préserver la cohérence des données et garantir une stabilité de notre base dans le temps.

Il existe des catégories de contraintes d'intégrité à respecter :

- contrainte d'entité :
- contrainte de domaine
- contrainte de référence

4.1) Contraintes d'entité

Par définition, une relation est un ensemble de tuples. Un ensemble n'ayant pas d''éléments en double, il ne peut pas exister deux fois le même tuple dans une relation. Toute relation doit donc posséder une clé unique (clé primaire).

4.2) Contraintes de domaine

Les données que nous souhaitons stocker dans notre base de données ont des formats différents. On parle alors de domaine. On peut s'inspirer des types de données des langages de programmation que nous avons étudiés (integer, booléens, float, char, string).

Les **contraintes de domaine** doivent permettre de :

- représenter les données sans perte d'information ;
- d'éviter les erreurs de saisies.

4.3) Contraintes de référence

Correctement construite, une base de données fait appel à des données situées dans différentes relations. Pour que les données restent utilisables et cohérentes, il ne faut pas que l'on puisse détruire des données qui dépendent les une des autres. C'est le rôle de l'intégrite referentielle de protéger ces relations. Autrement dit, l'intégrité référentielle vérifie qu'une valeur de clé étrangère existe bien en tant que valeur de clé primaire dans une autre table.