Bases de données relationnelles

Bases de données

Objectifs et prérequis. **Prérequis :** les fichiers CSV de première (lire une table enregistrée dans un fichier, la filtrer, la trier).
À l'issue de ce chapitre, vous saurez :

- expliquer ce qu'une base de données et son SGBD apportent par rapport à un simple fichier ;
- citer les quatre services rendus par un SGBD : persistance des données, gestion des accès concurrents, efficacité de traitement des requêtes et sécurisation des accès ;
- repérer les anomalies d'une table unique et expliquer comment le découpage en plusieurs tables les supprime ;
- employer le vocabulaire du modèle relationnel : relation, attribut, domaine, n-uplet, schéma ;
- identifier la clé primaire et les clés étrangères d'un schéma relationnel ;
- distinguer la structure d'une base de données de son contenu ;
- prévoir si le SGBD accepte ou refuse une opération, d'après les contraintes d'intégrité.

1 Le rôle d'une base de données

Un lycée garde la trace de ses élèves, de leurs classes, de leurs notes et de leurs absences. Ces informations sont nombreuses, liées entre elles (une note concerne un élève et une matière) et consultées chaque jour par des centaines de personnes : élèves, familles, professeurs, vie scolaire.

Une **base de données** rassemble de telles informations de façon organisée : chaque information y a une place définie, ce qui permet de la retrouver, de la croiser avec d'autres et de la mettre à jour.

En première, vous avez traité des tables enregistrées dans des fichiers CSV. Un tel fichier convient à une table qu'un seul programme utilise. Il ne suffit plus quand les données sont nombreuses et partagées :

- une coupure de courant pendant un enregistrement peut laisser le fichier à moitié écrit ;
- si deux personnes modifient le fichier en même temps, la dernière à l'enregistrer efface les modifications de l'autre ;
- pour retrouver une ligne, il faut lire le fichier depuis le début ;
- qui peut ouvrir le fichier peut y lire toutes les données.

Le logiciel qui gère une base de données s'appelle un **système de gestion de base de données**, ou **SGBD**. C'est lui qui range les informations, répond aux questions qu'on lui pose et applique les modifications. On lui adresse des **requêtes** : une requête est une demande, question ou modification, écrite dans un langage de requête. Pour les bases relationnelles, ce langage est presque toujours **SQL**, que comprennent tous les grands SGBD. Une base relationnelle présente les données sous forme de tables : on décrit ce que contiennent les tables, et le SGBD décide lui-même de la façon de ranger les octets sur le disque.

2 Les services rendus par un SGBD

Un SGBD rend quatre grands services, qui répondent aux limites d'un simple fichier.

La persistance des données. Les données sont gardées sur un disque et survivent à une coupure de courant ou à un redémarrage. Une modification interrompue par une panne est appliquée en entier ou pas du tout : la base ne reste jamais à moitié modifiée.

La gestion des accès concurrents. De nombreuses personnes utilisent la base au même moment. Le SGBD fait passer l'une après l'autre les modifications d'une même donnée, grâce à des **verrous** : un verrou réserve une donnée à une seule modification, le temps qu'elle se termine. Si deux personnes réservent en même temps le dernier siège d'un avion, la première demande traitée obtient la place, et la seconde la trouve déjà prise.

L'efficacité de traitement des requêtes. Grâce à des **index**, le SGBD retrouve une information parmi des millions sans parcourir toutes les lignes. Un index joue le rôle de l'index placé à la fin d'un livre, qui indique directement la page où chercher un mot.

La sécurisation des accès. Chaque utilisateur reçoit des droits : lire telle table, modifier telle autre. Au lycée, un élève consulte ses notes, un professeur saisit les notes de ses classes, et seul l'administrateur de la base gère les comptes et leurs droits.

③ Une table unique et ses anomalies

Un opérateur de téléphonie veut garder la trace des appels de ses clients. Première idée : tout écrire dans une seule table, avec une ligne par appel, qui donne le nom du client, son adresse, le numéro qui appelle, le numéro appelé et la durée de l'appel en secondes.

Client	Adresse	Numéro	Appelé	Durée (s)
Léon Dupont	3 rue du Marché	0612345678	0687654321	455
Léon Dupont	3 rue du Marché	0612345678	0611111111	62
Léon Dupont	3 rue du Marché	0633333333	0611111111	145
Léon Dupont	22 rue de la Gare	0644444444	0655555555	302

L'adresse de Léon Dupont est recopiée à chaque appel qu'il passe : on parle de **redondance**. Cette répétition crée trois sortes d'anomalies.

Anomalie de mise à jour. Si Léon Dupont déménage, il faut modifier toutes les lignes qui le concernent. Si l'on en oublie une, la table donne deux adresses différentes pour le même client et devient incohérente. La table ne permet d'ailleurs pas de savoir si la dernière ligne concerne le même Léon Dupont, après un déménagement, ou un homonyme.

Anomalie de suppression. Effacer le seul appel d'un client fait disparaître ses coordonnées en même temps.

Anomalie d'insertion. On ne peut pas enregistrer une nouvelle cliente, Inès Martin, tant qu'elle n'a passé aucun appel : chaque ligne de la table décrit un appel.

3.1 Découper en plusieurs tables

Pour supprimer ces anomalies, on découpe les informations en plusieurs tables, chacune décrivant une sorte d'objet : les clients, leurs téléphones et les appels.

Clients

`id_client`	`nom`	`prenom`	`adresse`
156	Dupont	Léon	3 rue du Marché
10234	Dupont	Léon	22 rue de la Gare

Telephones

numero	id_client
0612345678	156
0633333333	156
0644444444	10234

Appels

id_appel	appelant	appele	duree
1	0612345678	0687654321	455
2	0612345678	0611111111	62
3	0633333333	0611111111	145
4	0644444444	0655555555	302

Chaque information n'est plus écrite qu'une seule fois : l'adresse d'un client n'apparaît que dans la table Clients. Les tables restent reliées par des valeurs communes : l'identifiant du client relie un téléphone à son propriétaire, et le numéro de téléphone relie un appel au téléphone qui l'a passé.

Les trois anomalies disparaissent : un déménagement ne modifie qu'une ligne de la table Clients, supprimer un appel ne touche pas aux coordonnées du client, et un nouveau client s'enregistre dans la table Clients avant même son premier appel.

④ Le vocabulaire du modèle relationnel

Ce découpage suit le **modèle relationnel**, proposé en 1970 par Edgar Codd, chercheur chez IBM.

— Chaque table s'appelle une **relation** et porte un nom : Clients, Telephones, Appels.

— Chaque colonne est un **attribut**. Un attribut porte un nom et possède un **domaine**, c'est-à-dire l'ensemble des valeurs qu'il peut prendre, le plus souvent donné par un type : des entiers pour `id_client`, des chaînes de caractères pour `nom` et `adresse`. Dans un fichier CSV, au contraire, toutes les valeurs sont des chaînes de caractères.

— Chaque ligne est un **n-uplet**, aussi appelé **enregistrement** : elle donne une valeur à chaque attribut. Dans la table Clients, `(156, 'Dupont', 'Léon', '3 rue du Marché')` est un n-uplet.

Le **schéma** d'une relation décrit sa structure : son nom, puis ses attributs avec leurs domaines.

Clients(id_client : INT, nom : TEXT, prenom : TEXT, adresse : TEXT)

Les domaines s'écrivent avec les types du langage SQL : `INT` pour un entier, `TEXT` pour une chaîne de caractères, `REAL` pour un nombre flottant, `DATE` pour une date. On les omet souvent pour alléger l'écriture :

Clients(id_client, nom, prenom, adresse)

⑤ Les clés

5.1 Clé primaire

Dans la table Clients, deux lignes portent le même nom et le même prénom : deux clients différents peuvent s'appeler Léon Dupont. Le nom ne permet donc pas de distinguer les clients. On se sert d'un attribut à part, l'identifiant du client, qui joue le rôle de clé primaire.

La **clé primaire** d'une relation est un attribut, ou un groupe d'attributs, qui identifie chaque n-uplet de manière unique. Elle respecte deux règles :

— l'**unicité** : deux n-uplets n'ont jamais la même valeur de clé primaire ;

— la **non-nullité** : la clé primaire n'est jamais vide.

Dans un schéma, on souligne la clé primaire :

Clients(id_client, nom, prenom, adresse)

5.2 Clé étrangère

Dans la table Telephones, l'attribut `id_client` fait référence à la clé primaire de la table Clients : c'est une **clé étrangère**. Une clé étrangère est un attribut dont chaque valeur désigne une ligne d'une autre table, par la clé primaire de cette ligne. C'est elle qui relie les tables entre elles.

Pour savoir à qui appartient le téléphone 0644444444, on lit son identifiant de client, 10234, puis on cherche cette valeur dans la table Clients : c'est le client de la rue de la Gare.

Une clé étrangère doit toujours désigner une ligne qui existe : on ne peut pas attribuer un téléphone à un client inconnu. Dans un schéma, on précède la clé étrangère d'un dièse :

Telephones(numero, #id_client)

⑥ Le schéma relationnel

L'ensemble des schémas des relations forme le **schéma relationnel** de la base :

— **Clients**(id_client, nom, prenom, adresse)

— **Telephones**(numero, #id_client)

— **Appels**(id_appel, #appelant, appele, duree)

L'attribut souligné est la clé primaire ; un attribut précédé de `#` est une clé étrangère. Dans la table Appels, `appelant` désigne un numéro de la table Telephones. L'attribut `appele` n'est pas une clé étrangère : le numéro appelé peut appartenir au client d'un autre opérateur, et ne figurer dans aucune table de la base.

Pour lire un schéma relationnel, on se pose trois questions :

1. Quelles sont les tables ? Ici : Clients, Telephones et Appels.
2. Quels sont leurs attributs ? Par exemple, la table Clients en a quatre.
3. Quels liens les clés étrangères établissent-elles ? Ici, chaque téléphone appartient à un client, et chaque appel part d'un téléphone.

⑦ Les contraintes d'intégrité

Un schéma ne suffit pas à garantir des données cohérentes : rien n'empêcherait d'écrire une durée d'appel négative, deux clients sous le même identifiant, ou un téléphone rattaché à un client qui n'existe pas. On ajoute donc des **contraintes d'intégrité** : ce sont des règles que le SGBD vérifie à chaque insertion, modification ou suppression. Si une opération viole une contrainte, le SGBD la refuse, et les données restent intactes.

Chacune des trois contraintes protège l'une des notions du modèle relationnel.

La contrainte de domaine porte sur les attributs : chaque valeur doit appartenir au domaine de son attribut. Le domaine peut être plus restreint que le type : une durée d'appel est un entier positif ou nul, un nom ne peut pas être vide.

La contrainte de relation porte sur la clé primaire : elle doit rester unique et ne jamais être vide. Le SGBD refuse donc d'enregistrer un second client sous l'identifiant 156.

La contrainte de référence porte sur les clés étrangères : chacune doit désigner une ligne qui existe. Le SGBD refuse donc d'enregistrer un téléphone pour le client 999, qui n'existe pas. Il refuse aussi de supprimer le client 156, ou de changer son identifiant, tant que des téléphones y font référence : ces téléphones désigneraient alors un client qui n'existe plus.

Les contraintes se déclarent en SQL, au moment de créer les tables : le chapitre *Modification des données et définition des tables* montre comment.

8 Structure et contenu d'une base

Une base de données a une **structure** et un **contenu**. La structure est donnée par le schéma relationnel et par les contraintes d'intégrité : les tables, leurs attributs avec leurs domaines, leurs clés, et les règles que les données doivent respecter. Le contenu est l'ensemble des n-uplets enregistrés dans les tables.

La structure est fixée à la création de la base et change rarement ; le contenu change à chaque insertion, modification ou suppression. Enregistrer un nouvel appel modifie le contenu ; ajouter un attribut `date_naissance` à la table Clients modifie la structure. À chaque changement du contenu, le SGBD vérifie que les données respectent toujours la structure.

9 Activité

Exercice 1 *Opération acceptée ou refusée ?* Une base de commandes comporte deux tables :

— **Produits**(id_produit, nom, prix), où le prix d'un produit doit être strictement positif ;

— **Commandes**(id_commande, #id_produit, quantite), où chaque commande porte sur un produit.

Voici leur contenu.

Produits

id_produit	nom	prix
1	Cahier	2.50
2	Stylo	1.20

Commandes

id_commande	id_produit	quantite
1	2	5

On tente les opérations suivantes, l'une après l'autre. Pour chacune, indiquer si le SGBD l'accepte ou la refuse, et justifier en nommant la contrainte d'intégrité concernée.

1. Ajouter à la table Produits la ligne `(3, Gomme, 0.80)`.
2. Ajouter à la table Produits la ligne `(1, Colle, 2.00)`.
3. Ajouter à la table Produits la ligne `(4, Trousse, -2.00)`.
4. Ajouter à la table Commandes la ligne `(2, 5, 3)`.

5. Supprimer le stylo (produit 2) de la table Produits.
6. Supprimer la gomme (produit 3) de la table Produits.

L'essentiel à retenir.

— Une **base de données** organise des informations nombreuses et liées entre elles. Le logiciel qui la gère, le **SGBD**, rend quatre services : la persistance des données, la gestion des accès concurrents, l'efficacité de traitement des requêtes et la sécurisation des accès.
— Une **table unique** répète les informations, ce qui crée des anomalies de mise à jour, de suppression et d'insertion ; on découpe alors les données en plusieurs tables.
— Dans le **modèle relationnel**, une relation est une table, un attribut est une colonne avec son domaine, et un n-uplet est une ligne ; le schéma d'une relation donne son nom et ses attributs.
— La **clé primaire** identifie chaque n-uplet (unicité et non-nullité) ; une **clé étrangère** désigne une ligne d'une autre table par sa clé primaire.
— La **structure** d'une base, c'est-à-dire son schéma relationnel et ses contraintes, se distingue de son **contenu**, les n-uplets enregistrés.
— Le SGBD vérifie les **contraintes d'intégrité** de domaine, de relation et de référence, et refuse toute opération qui les viole.