Informatique en CPGE (2018-2019)
Bases de Données Relationnelles

Comment gérer des données à l'aide de systèmes informatiques ? Supposons que des données sont stockées sur un serveur qui se trouve quelque part dans le monde. Un utilisateur a besoin d'accéder à ce serveur par un réseau afin par exemple, d'obtenir une information, de modifier des données, d'en supprimer ou d'en rajouter. L'utilisateur va écrire sa question ou requête dans un navigateur de manière simple (formulaire, mots-clés) et cette question sera transformée en un programme dont l'exécution permettra d'obtenir le résultat souhaité.

Le système de fichiers sur un ordinateur ou un smartphone est un système élémentaire qui permet de gérer des données. Chaque fichier peut représenter un texte, une photo, une musique, un film, etc. On peut effectuer des recherches, ajouter ou supprimer des fichiers. Mais interroger ou modifier des bases de données qui peuvent être à l'échelle mondiale nécessite un système beaucoup plus complexe. Un système de gestion de bases de données doit permettre une plus grande rapidité d'accès aux données, un mode multi utilisateurs, assurer la sécurité, la confidentialité, l'intégrité, utiliser un langage "universel", etc.

1 Principes et architecture

Un système de gestion de base de données facilite la manipulation des données par l'utilisateur qui ne doit pas se soucier de "comment cela fonctionne dans la machine". Il est le médiateur entre la machine et la personne.

Ce que voient les utilisateurs, l'organisation physique dans la machine et la logique de l'organisation des données (le modèle relationnel) sont indépendants.

1.1 Le concept de client-serveur

L'utilisateur travaille sur une machine à l'aide d'une application ; il est le client. La base de données est gérée par un serveur (une autre machine). Plusieurs utilisateurs (personnes ou programmes) peuvent effectuer des demandes au serveur simultanément.

1.2 Architecture trois-tiers

Le plus souvent, l'utilisateur n'accède pas directement à la base de données. C'est l'application utilisée qui communique avec le serveur de base de données. Nous distinguons alors trois niveaux : le niveau utilisateur, le niveau applicatif et le niveau base de données.

2 Le modèle relationnel

Prenons un exemple. Tous les élèves de première et deuxième année peuvent avoir pendant une semaine des cours de soutien en informatique tous les jours avec un professeur qu'ils choisissent.

Pour organiser cela, on dresse donc un tableau qui ressemble à celui-ci :

NumEleve	Nom	Prenom	Classe	NomProf	Numprof	Salle
1				Java	1	I403
2				Céplus	2	I307
3				Java	1	I403
4				Java	1	I403
5				Python	3	I508
6				Céplus	2	I307
...						

Trois professeurs, Mme Java, M. Céplus et M. Python, participent et chacun des trois travaille dans sa salle. Un tableau de ce type peut se rencontrer dans de nombreux domaines, par exemple lorsque des passagers réservent un vol Nice-Paris à une date donnée. Lorsque le nombre de lignes devient grand, la modification d'un tel tableau peut être longue et source de plusieurs erreurs.

Par exemple : M. Python change de salle, M. Céplus sera absent et remplacé par M. Cémoins qui est dans une autre salle, les élèves qui avait choisi M. Céplus veulent changer pour Mme Java ...

La séparation d'objets reliés permet une avancée importante :

NumEleve	Nom	Prenom	Classe	Numprof
1				1
2				2
3				1
4				1
5				3
6				2
...				

Numprof	Nom	Salle
1	Java	I403
2	Céplus	I307
3	Python	I508
4	Cémoins	I215

Les modifications sont maintenant plus simples à gérer.

Dans le modèle relationnel, les données sont organisées en tableaux à deux dimensions qui s'appellent **relations**.

2.1 Les relations

Une relation regroupe un ensemble de données homogènes concernant un même élément (ex : élèves, professeurs, ...). Les données sont organisées sous forme de colonnes (ex : Nom, Prenom, ...). Chaque colonne, appelée **attribut** ou bien **champs** , caractérise la relation. Les valeurs des attributs sont présentées sous forme de lignes, appelées **n-uplets**, ou **tuples** en anglais, et chaque n-uplet est unique ; ces valeurs ont un type (entier, texte, flottant, ...) et appartiennent à un domaine (les entiers naturels inférieurs à 256, texte comportant au maximum 8 caractères, ...)

Une relation est donc un sous-ensemble, caractérisé par un nom, du **produit cartésien** de domaines.

Le **produit cartésien** d'un ensemble de domaines D_1, D_2, ..., D_n noté $D_1 \times D_2 \times \ldots \times D_n$ est l'ensemble des n-uplets $(v_1, v_2, \ldots, v_n)$ tels, pour tout i, $v_i \in D_i$.

2.2 Notion de clé primaire

Dans toute relation, un attribut, ou un groupe d'attributs, permet d'identifier de manière unique les valeurs des autres attributs. On l'appelle une **clé candidate**. S'il y a plusieurs clés candidates, on en privilégie une, nommée la **clé primaire**. Certaines relations peuvent contenir un attribut qui est la clé primaire d'une autre relation et qui est alors appelée **clé étrangère**.

Un **schéma de relation** se présente sous cette forme :

Relation (attribut 1, attribut 2, ..., attribut N)

Clé primaire : attribut 1

Clé étrangère : attribut N en référence à attribut 1 de Relation X

Par exemple :
Eleves (Ideleve, Nom, Prenom, Adresse, CP, Ville, Tel, Classe)
Clé primaire : Ideleve
Clé étrangère : Classe en référence à une autre relation

Un schéma de base de données est un ensemble de schémas de relations liés par des attributs communs.

Si à une valeur d'un attribut A correspond une et une seule valeur d'un attribut B, on dit que l'attribut B est en **dépendance fonctionnelle** de l'attribut A.
Par exemple : à un identifiant élève correspond un unique nom d'élève.

2.3 La normalisation relationnelle

Dans l'élaboration des relations, la normalisation relationnelle permet d'éviter la redondance des données et facilite leur mise à jour.

2.3.1 La première forme normale

Une relation est en première forme normale si tous ses attributs sont en dépendance fonctionnelle de la clé primaire et ne contiennent qu'une seule information.
Ex : Eleves (Ideleve, Nom Prenom, Adresse, CP, Ville, Tel, Classe) Clé primaire : Ideleve
Ici les attributs sont en dépendance fonctionnelle de la clé primaire mais l'attribut "Nom Prenom" contient deux informations, le nom et le prénom de l'élève. Il faut donc séparer cet attribut en deux attributs :
Eleves (Ideleve, Nom, Prenom, Adresse, CP, Ville, Tel, Classe)

2.3.2 La deuxième forme normale

Lorsque la clé primaire est constituée de plusieurs attributs, on dit qu'une relation est en deuxième forme normale si elle est en première forme normale et si tous les attributs sont en dépendance fonctionnelle de l'intégralité de la clé primaire et pas seulement que d'une partie de celle-ci.

2.3.3 La troisième forme normale

Une relation est en troisième forme normale si elle est en deuxième forme normale et si tous les attributs sont en dépendance fonctionnelle directe de la clé primaire et uniquement de la clé primaire.
Eleves (Ideleve, Nom, Prenom, Adresse, CP, Ville, Tel, Classe, Salle)
Cette relation n'est pas en troisième forme normale car l'attribut Salle est en dépendance fonctionnelle de la clé primaire Ideleve, mais aussi de l'attribut Classe.

3 Algèbre relationnelle

3.1 Vocabulaire des bases de données

Il est important d'avoir toujours en tête le vocabulaire utilisé en algèbre relationnelle et celui utilisé avec les bases de données :
relation = table, attribut = champ = colonne, tuple = n-uplet = ligne.

3.2 Opérateurs usuels sur les ensembles

Ces opérateurs ne s'appliquent que sur des relations compatibles : des relations $A(A_1, A_2, \ldots, A_n)$ et $B(B_1, B_2, \ldots, B_n)$ qui ont le même nombre d'attributs et où pour tout i, les attributs A_i et B_i ont le même domaine.

3.2.1 Union

$A \cup B$ est la relation qui inclut tous les n-uplets appartenant à A ou à B (au sens mathématiques). Les doublons sont éliminés.

3.2.2 intersection

$A \cap B$ est la relation qui inclut tous les n-uplets appartenant à A et à B (au sens mathématiques).

3.2.3 Différence

$A - B$ est la relation qui inclut tous les n-uplets appartenant à A mais pas à B.

3.3 Opérateurs spécifiques aux bases de données

On travaille sur les relations :
Eleves (Ideleve, Nom, Prenom, Adresse, CP, Ville, Tel, Numprof)
Profs (Id, Nom, Prenom, Tel, Salle)

3.3.1 Opérateur de projection

L'opérateur est noté π. Par exemple : $\pi_{\text{Nom, Prenom}}\big(\text{Eleves}\big)$.
On ne retient que les n-uplets des attributs indiqués par l'opérateur, ici Nom et Prenom. Les doublons sont éliminés.

3.3.2 Opérateur de sélection

L'opérateur est noté σ. Par exemple : Exemple : $\pi_{\text{Nom, Prenom}}\big(\sigma_{\text{Ville='Nice'}}\big(\text{Eleves}\big)\big)$
On ne retient que les n-uplets vérifiant une propriété indiquée par l'opérateur, ici Ville='Nice'.

3.3.3 Opérateur de jointure

L'opérateur est noté $\bowtie$. Par exemple : Eleves $\bowtie_{\text{Eleves.Numprof=Profs.Id}}$ Profs

Ceci est équivalent à une sélection sur le produit cartésien :

$$\sigma_{\text{Eleves.Numprof=Profs.Id}}\big(\text{Eleves} \times \text{Profs}\big)$$

Le produit cartésien Eleves $\times$ Profs contient toutes les associations possibles entre une valeur de Eleves et une valeur de Profs.

3.3.4 Opérateur de renommage

L'opérateur est noté ρ. Par exemple : $\rho_{\text{Adresse}\leftarrow\text{Rue}}\big(\text{Eleves}\big)$
Condition : les attributs Adresse et Rue ont le même domaine.
On obtient alors le nouveau schéma : Eleves (Ideleve, Nom, Prenom, Rue, CP, Ville, Tel, Numprof)

3.3.5 Division cartésienne

Si S et R sont deux relations, la relation $S \div R$ est la plus grande relation (pour l'inclusion) telle qu'il existe une relation R' vérifiant $((S \div R) \times R) \cup R' = S$ et $((S \div R) \times R) \cap R' = \emptyset$.

En particulier, si $S = R_1 \times R_2$, alors $S \div R_2 = R_1$.