SQL

1 Base de données

Introduction

Une base de données (BD) est un moyen permettant de mémoriser de façon pérenne des données et de pouvoir exprimer de façon précise les relations et liens entre les données.
Un système de gestion de base de données (SGDB) est une structure permettant de mettre en place et d'utiliser une ou plusieurs bases de données. Il doit gérer :

- la persistante des données ;
- la structure des données, les contraintes, les domaines ;
- un grand volume de données ;
- l'exécution de transactions concurrentes tout en préservant l'intégrité des données ;
- la sécurité des données : contrôle d'accès, tolérance de panne.

Pour les définitions et la terminologie utilisée on s'appuiera sur l'exemple suivant :

Id	**Auteur**	**Titre**	**Éditeur**
124	Émile Zola	Au bonheur des dames	Gallimard
15	Émile Zola	Germinal	Hachette
28	Honoré de Balzac	Eugénie Grandet	Gallimard
132	René Barjavel	La nuit des temps	Pocket

Définitions

- **Table ou Relation** : dans une base de données relationnelle, le tableau précédent est appelé **table** ou **relation**.
- **Enregistrement, tuple ou n-uplet** : une ligne de la table.

  Donner un tuple de la table de l'exemple :

  .....

- **Schéma** : le schéma d'une table est défini par :
  - le nombre d'attributs (nombre de colonnes de votre table) par tuple ;

    Donner le nombre d'attributs de la table de l'exemple :

    .....

  - un nom (différent) pour chaque attribut ;

    Donner un des attributs de la table de l'exemple :

    .....

  - pour chaque attribut, le domaine dans lequel il prend ses valeurs (entier, flottant, chaîne de caractère, un binaire...).

  Donner le domaine de l'attribut « Id » : .....

  Donner le domaine de l'attribut « Titre » : .....

  .....

  Le nombre d'attributs s'appelle l'arité.
  Pour l'exemple précédent on a donc le schéma suivant :

  schéma : Livre (Id : Entier ; Auteur : Chaîne ; Titre : Chaîne ; Éditeur : Chaîne)

- La valeur d'une relation est l'ensemble de ses enregistrements.
- La cardinalité d'une relation est le nombre de ses enregistrements.

Table - Relation (vocabulaire et notation)

Soit $S = (A_1, \cdots, A_n)$ un schéma.
Une table (ou relation) R sur le schéma S est un ensemble fini de n-uplets de $\text{dom}(A_1) \times \cdots \times \text{dom}(A_n)$.

Notation : pour tout $e \in R$ et $A \in S$, on note $e.A$ la composante du n-uplet associée à l'attribut A.
On dit que $A \in S$ est **une clé unique** pour S si et seulement si : $\forall (e, e') \in R^2,\ e.A = e'.A \Rightarrow e = e'$.
On dit que $A \in S$ est **la clé primaire, « Primary key »** si elle caractérise le tuple. En général noté « id ».

Remarque :

« Auteur » n'est pas une clé primaire du schéma ci-dessus car : .....

Mais ..... est une clé unique du schéma ci-dessus.

Exemple

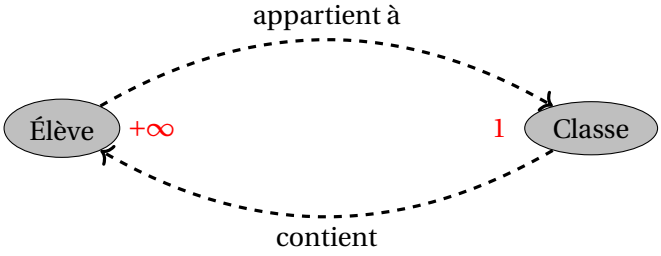
Liens entre des tables

Dans une base de données relationnelle, les relations vous permettent d'éviter les données redondantes. Par exemple, si vous concevez une base de données qui suit les informations sur les élèves et leur classe, vous pouvez avoir une table nommée « Eleve » qui stocke des informations sur chaque chaque élève, telles que son nom, son prénom et sa classe. Il existe également des informations que vous pourriez vouloir conserver sur la classe, telles que son nom, son professeur principal et son niveau.
On considère les deux tables suivantes :

- Élève de schéma (**Id**,Nom,Prenom,Id_Classe)
- Classe de schéma (**Id**,Nom,Professeur_principal,Niveau)

L'attribut *Id_Classe* du schéma *Élève* est associé à la clé primaire du schéma *Classe*.
Ainsi, chaque élément de la table *Élève* est associé à un unique élément de la table *Classe*. Les informations d'une classe seront ainsi mémorisées une unique fois dans notre base.
L'attribut *Id_Classe* de la table *Élève* est appelée **clé étrangère ou « Foreign key ».**

Le lien entre les tables *Classe* et *Élève* peut être représenté de la façon suivante :



Une classe contient des élèves mais chaque élève est dans une seule classe

Exercice 1 : Pour la gestion d'une bibliothèque, on considère les relations suivantes :

- *Livre* de schéma (Id, Titre ,IdAuteur , IdEditeur)
- *Éditeur* de schéma (Id, Nom, Adresse)
- *Auteur* de schéma (Id, Nom, Prénom)

On suppose que dans notre base un ouvrage a un unique auteur et un unique éditeur.

1. Définir les domaines des différents attributs (chaîne de caractère, entier...).

.....

.....

.....

.....

2. Quels sont les liens entre les tables ?

3. Pour chaque table, proposer la clé primaire et les éventuelles clés étrangères et uniques.

	clé primaire	clés étrangères	clés uniques
Livre			
Éditeur			
Auteur			

4. On ajoute une table *Utilisateur* de schéma (Id, Nom, Prénom).
   (a) Préciser les domaines des attributs de cette nouvelle table.

   .....
   .....
   .....

   (b) Proposer une solution afin de permettre à un utilisateur d'emprunter un ou plusieurs livres sachant qu'un livre ne peut pas être emprunté deux fois.

   .....

   (c) Modifier la structure de la base de données afin de pouvoir mémoriser des livres ayant plusieurs auteurs.

   .....

2 Langage SQL

Descripteur

Un **descripteur** (uni-terme ou multi-termes) est choisi pour éviter toute ambiguïté de sens. Sont dès lors évacués : les synonymes (par exemple : MALARIA et PALUDISME) ; les homographes (par exemple : PECHE (fruit) et PECHE (action de pêcher)).

Projection

Une projection consiste à obtenir une partie des attributs des enregistrements d'une table.
La commande SQL est : `SELECT attributs FROM table`.
Par exemple, pour obtenir les auteurs et les titres de la table Livre : `SELECT Auteur, Titre FROM Livre`.

Sélection

Une **sélection** consiste à obtenir une partie des enregistrements d'une table. La commande SQL est :

```
SELECT * FROM table WHERE Conditions
```

.
Remarque : le symbole `*` permet de ne pas effectuer de projection : les enregistrements retournés contiennent la totalité des attributs.
On peut effectuer des sélections complexes (portant sur plusieurs attributs) à l'aide des opérateurs.

Opérateurs	sens
+,-,*	Opérateurs arithmétiques (permet des opérations sur des nombres)
=,<,>	Opérateurs de comparaisons
<=	⩽
>=	⩾
AND	opérateur logique « et »
OR	opérateur logique « ou » (l'un ou l'autre ou les deux)
NOT	opérateur logique « différent de »

Exemple : pour obtenir les enregistrements relatifs à l'éditeur Gallimard :

```
SELECT * FROM Livre WHERE Éditeur Like "Gallimard";
```

Vous noterez les guillemets autour des chaînes de caractères.

3 Une première base

Utilisation du logiciel

Vous irez sur l'ENT, puis sur Capytale avec le code : `daf7-12570922`.

Vous noterez qu'il n'y a aucun accent dans les attributs pour éviter des bugs de lectures en informatique

Exercices en langage SQL

Pour récupérer la table que nous utiliserons aujourd'hui aller sur Capytale avec le code : `93b7-12570877`
Nous allons travailler avec la base de données Base_Commerce.sqlite.
Cette base regroupe différentes tables :

- *client* de schéma `(id,Nom,Prenom,Ville,Telephone,Type);`
- *ville* de schéma `(Nom,Departement,Population);`
- *article* de schéma `(id,Designation,Prix,Rayon);`
- *facture* de schéma `(id,idClient,Date,Reduction);`
- *achat* de schéma `(idArticle,idFacture,Quantite).`

1. Testez et expliquer les commandes suivantes :
   - `SELECT * FROM client WHERE id<10`

     .....
   - `SELECT Nom FROM client WHERE Ville LIKE "Paris"`

     .....
   - `SELECT * FROM client WHERE id<10 and Ville LIKE "Paris"`

     .....
   - `SELECT Nom, Ville FROM client WHERE Ville LIKE "_e%"`

     .....
2. Quel est le rôle du caractère *? du caractère %? du caractère _?

   * : .....

   % : .....

   _ : .....

- Tester

  `SELECT Ville FROM client WHERE Ville LIKE "_e%"`

  `SELECT DISTINCT Ville FROM client WHERE Ville LIKE "_e%"`

  Quelle différence y-a-t-il entre les deux requêtes :

  .....

3. En s'inspirant des requêtes précédentes, déterminez :
   - les informations sur les clients habitant à Paris et dont le prénom débute par la lettre E;

     .....
   - les noms et prénoms des clients qui ont la lettre B dans leur prénom.

   Modifications de tables

   .....
4. Testez la commande suivante : `UPDATE client SET Type=0`.

   Que constatez-vous pour les enregistrements de la table `client`?

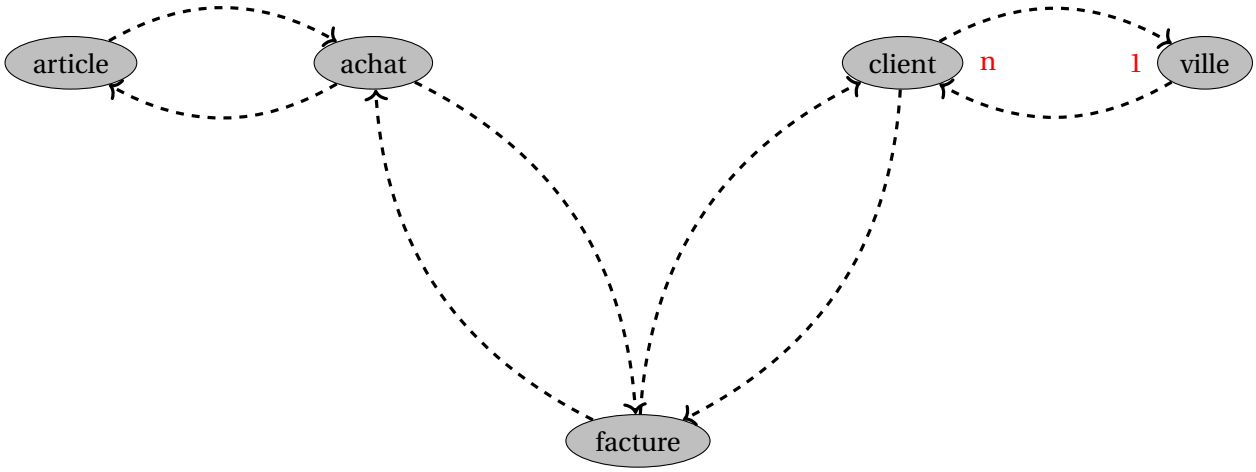
   .....

   De manière général, si nous ne voulons changer que les information de certains clients :

   `UPDATE table SET colonne_1 = 'valeur 1', colonne_2 = 'valeur 2', colonne_3 = 'valeur 3' WHERE condition`
5. Insérer un nouveau client avec la commande `INSERT INTO` *nom_table* un tuple. Attention de donner au nouveau client un identifiant unique et pas de nouvelle ville....
6. Supprimer un client avec `DELETE FROM client WHERE Nom like ....` (attention à l'importance du WHERE ici!)

Représentation des différentes relations

On peut représenter les relations entre les différentes tables de la façon suivante :



1. Il manque les « 1 » et les « $+\infty$ » pour certaines relations « 1 à plusieurs ». Compléter le graphique.
2. Existe-t-il une relation « plusieurs à plusieurs »? Quel est le rôle de la table *achat*?

   .....

Quelques requêtes sur une unique table

Afin de répondre aux questions suivantes, on pourra utiliser les opérateurs suivants (mais pas uniquement) :

- *IN* `( ... WHERE Attribut IN (x,y,z))`
- *BETWEEN* `( ... WHERE Attribut BETWEEN X AND Y)`
- *AND, OR, NOT*

1. Quelles villes se trouvent dans la Manche ou dans le Calvados?

   .....

2. Quelles villes ont entre 10 000 et 20 000 habitants ?

   .....

3. Quels articles ont un prix compris entre 10 et 20 euros mais différent de 12 ?

   .....

4. Quels articles ont un prix de 10, 100 ou 200 euros ?

   .....

Requêtes sur plusieurs tables

1. Tester les deux requêtes suivantes et expliquer les résultats obtenus :

   (a)
   ```
   SELECT id,client.Nom,client.Ville,ville.Nom FROM client,ville
   ```

   .....

   (b)
   ```
   SELECT client.Nom,Prenom,ville.Nom,ville.population
   FROM client JOIN ville ON Ville=ville.Nom
   ```

   .....

   (c) Expliquer le sens de `client.Nom` et `ville.Nom` utilisés pour ces deux requêtes.

   .....

   .....

2. Écrire les requêtes permettant de déterminer :

   (a) les noms et prénoms des clients qui ne sont pas normands ;

   .....

   (b) le nom du client et les dates de facture de ce client ayant pour identifiant le numéro 100 ;

   .....

   (c) déterminer les noms et les dates de facturation des clients parisiens ;

   .....

   (d) les identifiants des articles achetés par le client identifié par le numéro 100.

3. On considère la requête suivante :

   ```
   SELECT idArticle,idFacture,Date FROM client JOIN facture ON client.id=facture.idClient
           JOIN achat ON achat.idfacture=facture.id WHERE facture.idclient=100
   ```

   (a) Tester cette requête.

   (b) Quel est le but de cette requête ?

   .....

   (c) Expliquer la construction de cette requête.

   .....

4. Déterminer la désignation des articles achetés par le client ayant pour identifiant le numéro 100.

   .....

   .....

   .....