1 Problématique

La *base de données* étant crée, il faut maintenant pouvoir en modifier son contenu en fonction des demandes de l'utilisateur.

Quelles sont les instructions permettant de manipuler les données d'une base ?

2 Insérer des données

2.1 Syntaxe

La collection de bandes dessinées étant amenées à augmenter, la première opération nécessaire est de pouvoir ajouter une entité dans une table. Le code 3 ajoute l'auteur *Riad Sattouf* dans la table *Auteurs*.

```
1 INSERT INTO Auteurs values (309, "Sattouf", "Riad");
```

Code 1 – Insertion d'un auteur

La nécessité de connaître un *id* non utilisé peut être contraignant. Heureusement dans le schéma de la relation, cet attribut est affublé de l'étiquette *AUTOINCREMENT*. Le code 2 ajoute un nouvel auteur.

```
1 INSERT INTO Auteurs (nom, prenom) values ("Tardi", "Jacques");
```

Code 2 – Insertion en spécifiant les attributs

Activité 1 :

1. Télécharger et utiliser la base *bd-avec-emprunts.db*.
2. Ajouter les deux auteurs.
3. Ajouter trois emprunteurs :
   - Alice Knuth né le 19 avril 2002,
   - Bob Nelson né le 24 juillet 1990,
   - Christophe Viroulaud né le 08 décembre 1987.

2.2 Respect des contraintes d'intégrité

Les contraintes d'intégrité sont vérifiées au moment de l'insertion. Ainsi le code 3 renvoie une erreur.

```
1 INSERT INTO Auteurs values (309, "Giraud", "Jean");
```

Code 3 – Erreur d'insertion

Christophe Viroulaud

Activité 2 :

1. Quelle contrainte d'intégrité n'est pas respectée lors de cette tentative d'insertion ?
2. La requête 4 provoquera-t-elle une erreur ? Pour quelle raison ?

```sql
INSERT INTO Auteurs (nom, prenom) values ("Sfar", "Joann");
```

Code 4 – Provoque-t-on une erreur ?

3 Sélectionner des données

3.1 Syntaxe

Récupérer des données de la base est une autre manipulation indispensable afin de les utiliser ensuite dans une application. Le code 5 renvoie les informations des tous les auteurs.

```sql
SELECT id, nom, prenom FROM Auteurs;
```

Code 5 – Sélectionner des données

Si on sait que l'on doit récupérer toutes les colonnes, on peut utiliser le symbole *.

```sql
SELECT * FROM Auteurs;
```

Code 6 – Sélectionner toutes les données

3.2 Contrainte sur la sélection

Les codes 5 ou 6 renvoient toutes les entités de la table *Auteurs* ce qui ne représente que peu d'intérêt. Le code 7 renvoie tous les auteurs qui ont *Christophe* pour prénom.

```sql
SELECT nom FROM Auteurs WHERE prenom = "Christophe";
```

Code 7 – Clause WHERE

La clause *WHERE* évalue une expression booléenne. Les opérateurs de comparaison classiques peuvent être utilisés, ainsi que les opérateurs logiques (AND, OR, NOT).

```sql
SELECT nom FROM Auteurs WHERE prenom = "Christophe" AND NOT nom = "
   Arleston";
```

Code 8 – Expression booléenne

Christophe Viroulaud

Activité 3 :

1. Tester les requêtes précédentes.
2. Sélectionner les bandes dessinées dont l'*id du genre* est supérieur à 10.
3. Sélectionner les bandes dessinées dont le premier tome est sorti en 2010 ou après.

3.3 Sélectionner une chaîne de caractère approchante

Il peut être fastidieux d'effectuer une recherche exacte sur une chaîne de caractère. Le code 9 renvoie toutes les bandes dessinées qui contiennent *Astérix* dans leur titre.

```
1 SELECT * FROM Bandes_dessinees WHERE titre LIKE "%Astérix%";
```

Code 9 – Recherche de chaîne de caractère

La chaîne %Astérix% est un motif où le % est un *joker*. Il signifie que SQL peut le remplacer par n'importe quelle chaîne. Si on sait qu'il n'y aura qu'un seul caractère on peut utiliser le *joker* _. Ainsi le code 10 renvoie toutes les lignes qui contiennent *Astérix, Asterix....*

```
1 SELECT * FROM Bandes_dessinees WHERE titre LIKE "%Ast_rix%";
```

Code 10 – Joker

Activité 4 :

1. Tester les requêtes précédentes.
2. Sélectionner les auteurs dont le nom commence par un *T*.

4 Modifier des données

Il peut être nécessaire de modifier le contenu de certaines entités. Ainsi l'emprunteur *Christophe Viroulaud* n'est pas né en 1987 mais en 1977.

```
1 UPDATE Emprunteurs SET naissance = "1977-12-08" WHERE nom = "Viroulaud";
```

Code 11 – Modification d'une date de naissance

La clause *WHERE* se construit sur les mêmes principes que précédemment. L'exécution de la requête renvoie le nombre d'entités (enregistrements) affectées. Il peut être nul.

5 Supprimer des données

5.1 Syntaxe

Enfin la syntaxe suivante permet de supprimer une ligne d'une table.

```sql
DELETE FROM Emprunteurs WHERE nom = "Viroulaud";
```

Code 12 – Modification d'une date de naissance

Il est possible de supprimer plusieurs lignes en une seule requête.

5.2 Respect des contraintes

Le code 13 renvoie une erreur : il n'est pas possible de supprimer un emprunteurs s'il a encore des bandes dessinées en sa possession. Ainsi dans la table *Emprunts* l'attribut *id_emprunteurs* est référencé comme une clé étrangère de l'attribut *id* de la table *Emprunteurs*.

```sql
DELETE FROM Emprunteurs WHERE id = 1;
```

Code 13 – Contrainte de référence

Activité 5 :

1. Que doit réaliser la requête 12 ?
2. Tester les requêtes 11, 12 et 13.
3. Pour quelle raison la requête 14 renverra une erreur ?

```sql
DELETE FROM Bandes_dessinees WHERE isbn = 2205050699;
```

Code 14 – Cette requête renvoie une erreur

4. Quelle requête doit-on réaliser préalablement avant d'effectuer la requête 14 ? Que signifie-t-elle dans la vie réelle ?

À retenir

Toute exécution de requête est **définitive**. Il peut être pertinent d'effectuer une copie de sauvegarde avant d'effectuer d'importantes manipulations.

Christophe Viroulaud