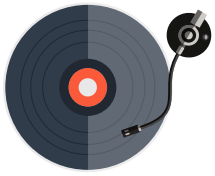
Le langage SQL
EXERCICES

Dernière mise à jour le : 31/10/2023

Exercice 1 : BDD du disquaire - Requêtes portant sur une table



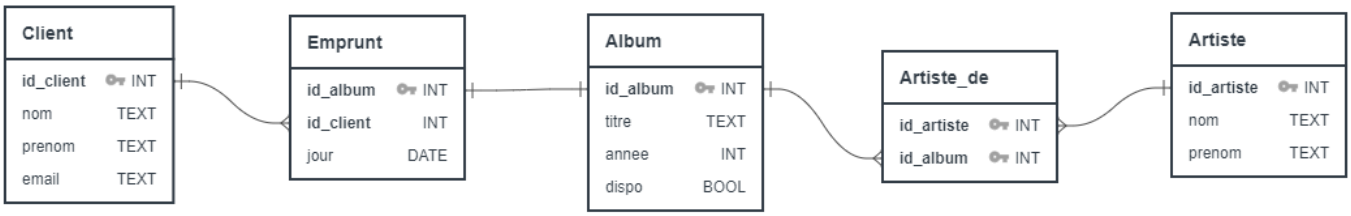
L'objectif de cet exercice est de passer en revue la plupart des requêtes SQL d'interrogation portant sur *une seule table*.

> On travaillera avec la base de données `disquaire.db` que vous trouverez dans l'archive de ce chapitre. Vous utiliserez le logiciel *DB Browser for SQLite* ou le logiciel en ligne sqliteonline.com dans lesquels vous aurez au préalable ouvert la base de données en question.

On rappelle que cette base de données possède le schéma relationnel suivant :

```
Album(id_album INT, titre TEXT, annee INT, dispo BOOL)

Artiste(id_artiste INT, nom TEXT, prenom TEXT)

Artiste_de(#id_artiste INT, #id_album INT)

Client(id_client INT, nom TEXT, prenom TEXT, email TEXT)

Emprunt(#id_client INT, #id_album INT, jour DATE)
```

On peut aussi représenter graphiquement ce schéma par le diagramme suivant :



Réalisé avec l'application quickdatabasediagrams.com

Sélectionner des colonnes

Q1 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT * FROM Album;
```

Réponse :

Q2 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT titre, dispo FROM Album;
```

Réponse :

Q3 : Écrivez et testez une requête permettant de récupérer uniquement les titres et l'année de sortie de chaque album.

Réponse :

Q4 : Écrivez et testez une requête permettant de récupérer tous les attributs des clients.

Réponse :

Q5 : Écrivez et testez une requête permettant de récupérer uniquement l'id_client, le nom et le prénom de chaque client.

Réponse :

Q6 : Écrivez et testez une requête permettant de récupérer uniquement le nom et le prénom de chaque artiste.

Réponse :

Sélectionner des lignes

En plus de sélectionner des colonnes, on peut sélectionner certaines lignes en utilisant la clause `WHERE` suivie de la condition de sélection.

Q7 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT titre FROM Album WHERE dispo=0;
```

Réponse :

Q8 : Écrivez et testez une requête permettant de récupérer les titres de tous les albums sortis en 2000 ou après.
Résultat attendu : 12 enregistrements.

Réponse :

Q9 Écrivez et testez une requête permettant de récupérer tous les albums sortis en 1970.
Résultat attendu : 3 enregistrements.

Réponse :

Q10 : Écrivez et testez une requête permettant de récupérer les titres de tous les albums sortis avant 1950 ou après 2010.
Résultat attendu : 6 enregistrements.

Réponse :

Q11 : Écrivez et testez une requête permettant de récupérer tous les albums disponibles et sortis avant 1990.
Résultat attendu : 13 enregistrements.

Réponse :

Q12 : Écrivez et testez une requête permettant de récupérer tous les clients dont le nom de famille est "Petit".
Résultat attendu : 2 enregistrements.

Réponse:

Q13 : Écrivez et testez une requête permettant de récupérer tous les clients dont le nom de famille est "Chartier".
Résultat attendu : 0 enregistrements.

Réponse:

Q14 : Écrivez et testez une requête permettant de récupérer les noms des artistes ne possédant pas de prénom.
Résultat attendu : 6 enregistrements.

> SOS Lisez cette page pour savoir comment tester si une valeur est absente avec SQLite : [https://sql.sh/cours/where/is)

Réponse :

Trier des données

On peut trier des données en utilisant `ORDER BY` à la fin d'une requête, suivi de l'attribut à trier et de `ASC` (pour un tri croissant) ou `DESC` (pour un tri décroissant).

Q15 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT * FROM Artiste ORDER BY nom ASC;
```

Réponse :

Q16 : Écrivez et testez une requête permettant de récupérer les albums triés dans l'ordre décroissant de leur année de sortie.

Réponse :

Q17 : Écrivez et testez une requête permettant de récupérer les titres par ordre alphabétique des albums sortis entre 1980 et 2010.
Résultat attendu : 10 enregistrements.

Réponse :

Q18 : Écrivez et testez une requête permettant de récupérer les noms et prénoms de tous les clients, triés par ordre alphabétique des noms.

Réponse :

Q19 : Écrivez et testez une requête permettant de récupérer les noms et prénoms de tous les clients, triés par ordre alphabétique des noms, puis des prénoms en cas de noms identiques.
Vous vérifierez bien que les clients portant le même nom sont bien triés alphabétiquement selon leur prénom (ce qui n'était pas le cas de la précédente requête).

> SOS On peut trier des données selon plusieurs attributs en les écrivant après `ORDER BY` dans l'ordre souhaité et en les séparant par des virgules.

Réponse :

Supprimer les doublons grâce à `DISTINCT`

On peut supprimer les doublons grâce à `DISTINCT`.

Q20 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT DISTINCT id_client FROM Emprunt;
```

Réponse :

Q21 : Écrivez et testez une requête permettant de récupérer les titres distincts des albums.
Résultat attendu : 29 enregistrements (car un album était en double).

Réponse :

Compter les enregistrements

On peut compter le nombre d'enregistrements (ou tuples) en utilisant la fonction `COUNT()`.

Q22 : Traduisez par une phrase la requête suivante. *Vérifiez **ensuite** votre réponse en l'exécutant.*

```
SELECT COUNT(*) AS nbClients FROM Client;
```

> On rappelle que `AS` permet de nommer le résultat de la requête, ici on le nomme `nbClients`. Essayez d'enlever le `AS nbClients` et observez le résultat de la requête.

Réponse :

Q23 : Écrivez et testez une requête permettant de récupérer le nombre d'emprunts en cours, que l'on notera `nbEmprunts`.
Résultat attendu : 11.

Réponse :

Q24 : Écrivez et testez une requête permettant de récupérer le nombre de clients ayant le nom "Petit", que l'on notera `nbPetit`.
Résultat attendu : 2.

Réponse :

Q25 : Écrivez et testez une requête permettant de récupérer le nombre de titres différents d'albums, que l'on notera `nbAlbumsDistincts`.
Résultat attendu : 29.

> SOS Vous lirez la page suivant pour savoir comment faire : https://sql.sh/fonctions/agregation/count.

Réponse :

Recherche par motif

On peut effectuer des requêtes effectuant des recherches de certains motifs.

Par exemple, on peut chercher tous les clients dont l'adresse email contient le domaine "domaine.net". La requête s'écrirait :

```sql
SELECT * FROM Client WHERE email LIKE "%domaine.net%";
```

Analyse :

- On a remplacé le `=` qui fait une recherche exacte par `LIKE`. Ainsi, `email LIKE "%domaine.net%"` est évaluée à vrai si et seulement si l'attribut `email` correspond au motif `"%domaine.net%"`.
- Dans un motif le symbole `%` est un *joker* et peut être substitué par n'importe quelle chaîne de caractères.

Q26 : Écrivez et testez une requête permettant de récupérer les titres de tous les albums contenant le mot `"Love"`.
Résultat attendu : 2 enregistrements.

Réponse :

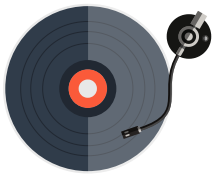
Q27 : Écrivez et testez une requête permettant de récupérer le nombre d'albums dont le titre contient la lettre `"a"`.
Résultat attendu : 16 enregistrements.

Réponse :

Q28 : Écrivez et testez une requête permettant de récupérer les noms et prénoms de tous les artistes *commençant* par la lettre `"M"`.
Résultat attendu : 3 enregistrements.

Réponse :

Exercice 2 : BDD du disquaire - Requêtes croisées



> On travaillera toujours avec la base de données `disquaire.db` que vous trouverez dans l'archive de ce chapitre. Vous utiliserez le logiciel *DB Browser for SQLite* ou le logiciel en ligne sqliteonline.com dans lesquels vous aurez au préalable ouvert la base de données en question.

L'exercice précédent portait sur des requêtes ne portant à chaque fois que sur une seule table. C'est malheureusement insuffisant pour chercher certaines informations qui nécessitent de croiser plusieurs tables.

Introduction

> On présente dans cette introdution la notion de **jointure** qui permet de croiser plusieurs tables. Prenez le temps de lire attentivement cette partie, de tester les requêtes proposées, de les modifier, etc. pour bien comprendre cette notion fondamentale des bases de données relationnelles.

Imaginons que l'on veuille connaître les clients ayant des emprunts en cours. Ces derniers sont ceux présents dans la table `Emprunt` et on peut les obtenir avec la requête

```
SELECT * FROM Emprunt;
```

qui produit la réponse suivante :

id_client	id_album	jour
1	5	2021-09-10
3	8	2021-08-18
3	24	2021-08-18
5	25	2021-09-12
5	6	2021-10-10
9	20	2021-09-28
11	14	2021-10-08
7	15	2021-10-08
7	19	2021-10-08
7	16	2021-10-15
16	29	2021-10-01

Mais ce n'est pas très satisfaisant car on aimerait plutôt afficher les noms, prénoms et adresse email de ces clients plutot que `id_client`.

Le problème est que les noms, prénoms, adresses email sont uniquement présents dans la table `Client`. Il est nécessaire de faire **une jointure** entre les deux tables `Emprunt` et `Client`.

Première jointure

Une jointure consiste à créer toutes les combinaisons de lignes des deux tables ayant un attribut de même valeur (l'attribut `id_client` dans notre exemple). Pour effectuer une jointure, on utilise `JOIN` !

Ainsi, la requête

```
SELECT *
FROM Emprunt
JOIN Client ON Emprunt.id_client = Client.id_client;
```

crée la jointure des deux tables `Emprunt` et `Client`, représentée ci-dessous.

id_client	id_album	jour	id_client_1	nom	prenom	email
1	5	2021-09-10	1	Dupont	Florine	dupont.florine@domaine.net
3	8	2021-08-18	3	Marchand	Grégoire	greg.marchand49@music.com
3	24	2021-08-18	3	Marchand	Grégoire	greg.marchand49@music.com
5	25	2021-09-12	5	Pacot	Jean	jpacot@music.com
5	6	2021-10-10	5	Pacot	Jean	jpacot@music.com
9	20	2021-09-28	9	Dubois	Philippe	pdubois5@chezmoi.net
11	14	2021-10-08	11	Fournier	Marie	mfournier@abc.de
7	15	2021-10-08	7	Moreau	Alain	amoreau1@abc.de
7	19	2021-10-08	7	Moreau	Alain	amoreau1@abc.de
7	16	2021-10-15	7	Moreau	Alain	amoreau1@abc.de
16	29	2021-10-01	16	Bernardin	Stéphanie	sbernard1@chezmoi.net

Analyse :

- La jointure (`SELECT * FROM Emprunt JOIN Client`) a permis de recopier toutes les colonnes des deux tables.
- Le choix des lignes à conserver, appelée *condition de jointure*, suit le mot clé `ON`. Cela permet de fusionner uniquement les lignes vérifiant la condition `Emprunt.id_client = Client.id_client`, autrement dit les lignes pour lesquelle l'attribut `id_client` est

identique donc celles concernant un même client.

> Essayez d'enlever le `ON ...`, vous constaterez que toutes les lignes sont fusionnées, ce qui est absurde car une même ligne peut alors correspondre à deux clients distincts.

- Vous avez constaté que l'on a préfixé chaque attribut par le nom de la table auquel il appartient. Cela permet de faire la différence entre deux attributs portant le même nom dans deux tables différentes, et c'est une bonne pratique de toujours le faire même lorsqu'il n'y a pas d'ambiguïté.

Ce sont les clés étrangères qui permettent de faire le lien entre les tables, il est donc normal que la condition de jointure fasse intervenir `id_client` (puisque c'est une clé étrangère de la table `Emprunt` qui la lie à la table `Client`).

On peut combiner une jointure avec la clause `SELECT` pour n'afficher que ce qui nous intéresse. Par exemple, si on ne veut que les noms, prénoms et adresses email des clients ayant des emprunts en cours ainsi que les albums empruntés et le jour d'emprunt, on peut faire la requête

```
SELECT Emprunt.id_album, Emprunt.jour, Client.nom, Client.prenom, Client.email
FROM Emprunt
JOIN Client ON Emprunt.id_client = Client.id_client;
```

qui produit le résultat

id_album	jour	nom	prenom	email
5	2021-09-10	Dupont	Florine	dupont.florine@domaine.net
8	2021-08-18	Marchand	Grégoire	greg.marchand49@music.com
24	2021-08-18	Marchand	Grégoire	greg.marchand49@music.com
25	2021-09-12	Pacot	Jean	jpacot@music.com
6	2021-10-10	Pacot	Jean	jpacot@music.com
20	2021-09-28	Dubois	Philippe	pdubois5@chezmoi.net
14	2021-10-08	Fournier	Marie	mfournier@abc.de
15	2021-10-08	Moreau	Alain	amoreau1@abc.de
19	2021-10-08	Moreau	Alain	amoreau1@abc.de
16	2021-10-15	Moreau	Alain	amoreau1@abc.de
29	2021-10-01	Bernardin	Stéphanie	sbernard1@chezmoi.net

Combiner les jointures

Plutôt que d'afficher l'`id_album`, qui est peu lisible, on peut préférer afficher le titre de l'album. Pour cela, on peut faire une nouvelle jointure :

```
SELECT Album.titre, Emprunt.jour, Client.nom, Client.prenom, Client.email
FROM Emprunt
JOIN Client ON Emprunt.id_client = Client.id_client
JOIN Album ON Emprunt.id_album = Album.id_album;
```

Analyse : On a ajouté la dernière ligne qui permet de faire une jointure sur l'attribut `id_album` entre la table produite par la requête précédente et la table `Album`. Et on a remplacé la première colonne `Emprunt.id_album` par `Album.titre` pour faire apparaître les titres des albums comme souhaité :

titre	jour	nom	prenom	email
Axis : Bold As Love	2021-09-10	Dupont	Florine	dupont.florine@domaine.net
Riding With The King	2021-08-18	Marchand	Grégoire	greg.marchand49@music.com
Continuum	2021-08-18	Marchand	Grégoire	greg.marchand49@music.com
Continuum	2021-09-12	Pacot	Jean	jpacot@music.com

Thriller	2021-10-10	Pacot	Jean	jpacot@music.com
Power Up	2021-09-28	Dubois	Philippe	pdubois5@chezmoi.net
Let It Be	2021-10-08	Fournier	Marie	mfournier@abc.de
44/876	2021-10-08	Moreau	Alain	amoreau1@abc.de
Songs in the Key of Life	2021-10-08	Moreau	Alain	amoreau1@abc.de
Lady Soul	2021-10-15	Moreau	Alain	amoreau1@abc.de
Leave the Light On	2021-10-01	Bernardin	Stéphanie	sbernard1@chezmoi.net

On peut combiner tout ce qui a été vu précédemment (dans l'exercice 1 par exemple) avec les jointures, on peut par exemple ajouter des conditions. La requête

```
SELECT Album.titre, Client.nom, Client.prenom, Client.email
FROM Emprunt
JOIN Client ON Emprunt.id_client = Client.id_client
JOIN Album ON Emprunt.id_album = Album.id_album
WHERE Client.nom = "Moreau";
```

produit le résultat

titre	nom	prenom	email
44/876	Moreau	Alain	amoreau1@abc.de
Lady Soul	Moreau	Alain	amoreau1@abc.de
Songs in the Key of Life	Moreau	Alain	amoreau1@abc.de

Utiliser des alias

Certaines requêtes peuvent commencer à être assez longues à écrire. Pour réduire leur longueur on peut utiliser des *alias* pour les noms de table grâce au mot clé `AS`.

Ainsi, la requête précédente peut aussi s'écrire

```
SELECT a.titre, c.nom, c.prenom, c.email
FROM Emprunt AS e
JOIN Client AS c ON e.id_client = c.id_client
JOIN Album AS a ON e.id_album = a.id_album
WHERE c.nom = "Moreau";
```

Analyse : `Emprunt AS e` permet de renommer la table `Emprunt` par `e`, ce qui permet de raccourcir les écritures du type `Emprunt.id_client` en `e.id_client`. Idem pour `c` et `a` qui sont les alias respectifs des tables `Client` et `Album`.

À vous de jouer !

Q1 : Écrivez et testez une requête permettant de récupérer l'adresse email de chaque client ayant un emprunt en cours.
Résultat attendu : 11 enregistrements (comme un peu au-dessus).

Réponse :

Q2 : Écrivez et testez une requête permettant de récupérer les noms, prénoms et adresses email des clients ayant un emprunt en cours et dont la date d'emprunt est postérieure au 2 octobre 2021.
Résultat attendu : 5 enregistrements.

Réponse :

Q3 : En réalité, l'attribut `dispo` (de la table `Album`) n'est pas utile car on peut retrouver tous les albums empruntés avec une jointure. **En supposant que l'attribut `dispo` n'existe pas**, écrivez et testez une requête permettant de récupérer le titre de tous les albums empruntés.
Résultat attendu : 11 enregistrements.

Réponse :

Q4 : Toujours sans utiliser l'attribut `dispo`, écrivez et testez une requête permettant de récupérer le titre de tous les albums empruntés par le client dont l'attribut `id_client` vaut 7.
Résultat attendu : 3 enregistrements.

Réponse :

Q5 : Toujours sans utiliser l'attribut `dispo`, écrivez et testez une requête permettant de récupérer le nom, le prénom et le titre de tous les albums empruntés par le client dont l'attribut `id_client` vaut 7.
Résultat attendu : 3 enregistrements (comme à la question précédenet, le client en question est Alain Moreau).

Réponse :

Q6 : Écrivez et testez une requête permettant de récupérer les titres ainsi que les noms et prénoms des artistes de chaque album.
Résultat attendu : 34 enregistrements.

Réponse :

Q7 : Écrivez et testez une requête permettant de récupérer les titres et l'année de sortie de tous les albums de Michael Jackson.
Résultat attendu : 3 enregistrements.

Réponse :

Q8 : Écrivez et testez une requête permettant de récupérer les titres et l'année de sortie de tous les albums de Sting, rangés par ordre croissant d'année de sortie.
Résultat attendu : 3 enregistrements.

Réponse :

Q9 : Écrivez et testez une requête permettant de récupérer les noms et prénoms des artistes de l'album initulé `"Don't Explain"`.
Résultat attendu : 2 enregistrements.

Réponse :

Exercice 3 - Questions interactives en ligne avec SQLZOO

1. Travail sur SELECT...

... sur une base de données des prix Nobel.

Lien vers l'activité : https://sqlzoo.net/wiki/SELECT_from_Nobel_Tutorial

Correction disponible ici : https://github.com/jisaw/sqlzoo-solutions/blob/master/select-from-nobel.sql

2. Travail sur SUM et COUNT ...

... sur une base de données sur les pays du monde.

Lien vers l'activité : https://sqlzoo.net/wiki/SUM_and_COUNT

Correction disponible ici : https://github.com/jisaw/sqlzoo-solutions/blob/master/sum-and-count.sql

3. Travail sur JOIN ...

... sur une base de données sur l'Euro 2012.

Lien vers l'activité : https://sqlzoo.net/wiki/The_JOIN_operation

> **Attention : Vous pouvez vous arrêter à la question 8**. En effet, à partir de la question 9, il s'agit d'utiliser la clause `GROUP BY` qui n'est pas au programme de Terminale NSI (mais libre à vous de vous y intéresser ou pas).

Correction disponible ici : https://github.com/jisaw/sqlzoo-solutions/blob/master/join.sql

Exercice 4 - Modifier une table

Les données stockées dans une base de données n'ont pas vocation à être figées, elles peuvent être modifiées au cours du temps grâce à des **requêtes de mise à jour** de la base de donnnées.

Nous allons voir les requêtes permettant d'ajouter des données à une table, de modifier les données d'une table et de supprimer les données d'une table.

Avant cela, faisons une petite digression sur la création de tables dans une base de données.

Création d'une table avec `CREATE TABLE`

> La création d'une base de données n'est pas au programme de Terminale NSI. Néanmoins, connaître les requêtes permettant de créer une table permet de mieux appréhender les requêtes de mise à jour que l'on verra ensuite.

Créer une base de données consiste à créer les tables de la base. Pour créer une table, on utilise `CREATE TABLE`.

Exemple

Pour créer la table `Artiste` correspondant à la relation suivante :

```
Artiste(id_artiste INT, nom TEXT, prenom TEXT)
```

on peut exécuter cette commande SQL :

```
CREATE TABLE Artiste (
    id_artiste INTEGER PRIMARY KEY,
    nom TEXT,
    prenom TEXT
);
```

Remarque : On a bien précisé le nom et le type de chaque attribut, et indiqué avec `PRIMARY KEY` quelle était notre clé primaire.

À vous de jouer !

Q1 : Ouvrez *DB Browser for SQLite* ou sqliteonline.com :

- si vous utilisez *DB Browser for SQLite*, cliquez sur "Nouvelle Base de Données" puis appelez-la `bac.db` et validez. Fermez ensuite la fenêtre de l'assistant de création de table. Vous êtes désormais prêt pour la suite !
- si vous utilisez *sqliteonline.com*, remplacez la requête dans la fenêtre SQL par `DROP TABLE demo;` pour effacer la table `demo` présente au départ. Vous êtes désormais prêt pour la suite !

Q2 : Écrivez la requête permettant de créer une table `Note` correspondant à la relation suivante. *Vous vérifierez ensuite dans la structure de la base de données que la table a bien été créée avec les bons attributs et les bons types.*

```
Note(id_eleve INT, nom TEXT, prenom TEXT, maths INT, anglais INT, info INT)
```

Réponse :

Insertion de valeurs dans la table avec `INSERT INTO ... VALUES`

Pour insérer des enregistrements (= des lignes) dans une table, on utilise la commande `INSERT INTO ... VALUES`

Exemple

Pour insérer les 3 enregistrements suivants dans la table `Artiste`.

id_artiste	nom	prenom
1	Clapton	Éric
2	Mayall	John
3	Hendrix	Jimi

on écrit la requête

```
INSERT INTO Artiste VALUES (1, 'Clapton', 'Éric'),
                           (2, 'Mayall', 'John'),
                           (3, 'Hendrix', 'Jimi');
```

Analyse :

- Après `INSERT INTO` (que l'on traduit par "insérer dans") on indique le nom de la table (ici `Note`) dans laquelle on veut insérer des données ;
- Puis on indique grâce au mot clé `VALUES` les enregistrements que l'on veut insérer, ces derniers étant séparés par des virgules s'il y en a plusieurs (on n'oublie pas le `;` pour terminer) ;
- Avec cette requête, les valeurs des différents enregistrements (ou n-uplets) doivent être données dans le même ordre que lors du `CREATE TABLE`. Néanmoins, il est possible de les passer dans un ordre différent comme on l'explique juste en-dessous.

Si on désire passer les valeurs des enregistrements dans un ordre différent de celui de la création de la table, il suffit de préciser l'ordre juste après le nom de la table :

```
INSERT INTO Artiste (prenom, nom, id_artiste) VALUES ('John', 'Mayer', 4);
```

Pour voir le code SQL qui a permi de créer la base de données du disquaire, vous pouvez suivre ce lien. Regardez notamment comment on a lié les tables avec les clés étrangères en utilisant `REFERENCES` et comment on définit la réunion de plusieurs attributs comme clé primaire. Vous constaterez également que l'on n'a pas utilisé le type `TEXT` pour le domaine des attributs de type "texte". Enfin, vous trouverez dans ce fichier .sql les ordres d'insertions des différents enregistrements de chaque table.

À vous de jouer !

Q3 : Écrivez la requête permettant d'insérer les trois enregistrements suivants dans la table `Note`.

id_eleve	nom	prenom	maths	anglais	info
1	Marchand	Alice	14	13	11
2	Muller	Marie	10	18	17
3	Prenel	Laura	13	14	15

Réponse :

Q4 : Écrivez et testez une requête permettant d'afficher tous les enregistrements de la table `Note` afin de vérifier que les données ont bien été ajoutées.

Réponse :

Q5 : Écrivez et testez la requête permettant d'ajouter l'enregistrement `(3, 'Dupont', 'Arthur', 18, 14, 13)` à la table `Note`. Quelle est l'erreur provoquée ? Expliquez-la (voir Chapitre 1 si nécessaire).

Réponse :

Q6 : Écrivez et testez une requête permettant d'ajouter l'enregistrement concernant Arthur mais avec la clé primaire `id_eleve` égale à 4. *Vous vérifierez que l'enregistrement a bien été ajouté !*

Réponse :

À ce stade, vous devriez obtenir la table `Note` suivante :

id_eleve	nom	prenom	maths	anglais	info
1	Marchand	Alice	14	13	11
2	Muller	Marie	10	18	17
3	Prenel	Laura	13	14	15
4	Dupont	Arthur	18	14	13

Modification de valeurs avec `UPDATE ... SET`

Il est possible de modifier des valeurs existantes dans une table, avec `UPDATE`.

Exemple

Dans la table `Client`, le client

id_client	nom	prenom	email
4	Michel	Valérie	vmichel5@monmail.com

a changé d'adresse email. Pour modifier cette adresse dans la table `Client`, on peut écrire :

```
UPDATE Client SET email = 'valerie.michel@email.fr'
WHERE id_client = 4;
```

Analyse :

- Après `UPDATE` on indique le nom de la table dans laquelle on veut modifier une valeur (ici `Client`) ;
- Ensuite, on écrit `SET` puis une expression de la forme `attribut = valeur` qui permet de définir une nouvel valeur `valeur` pour l'attribut `attribut` (ici `'valerie.michel@email.fr'` est la nouvelle valeur de l'attribut `email`) ;
- Enfin, on précise avec `WHERE` la condition permettant de sélectionner les enregistrements sur lesquels la modification doit être apportée (ici une seule ligne car la condition `id_client = 4` ne correspond qu'à un seul enregistrement).

À vous de jouer !

Q7 : Il y a eu une erreur dans la saisie de l'élève numéro 3 : son prénom n'est pas Laur**a** mais Laur**e**. Écrivez et testez une requête permettant d'effectuer la correction. *Vous vérifierez que la modification a bien été effectuée !*.

Réponse :

Q8 : Un autre erreur a été détectée : les notes de Mathématiques et d'Anglais de Marchand Alice ont été inversées. Écrivez et testez **une** requête permettant d'effectuer la correction. *Vous vérifierez que la modification a bien été effectuée !*.

> SOS Vous lirez la page suivant pour savoir comment modifier plusieurs valeurs simultanément avec `UPDATE ... SET` : https://sql.sh/cours/update.

Réponse :

Q9 : Pour des raisons d'équité entre établissement, il a été décidé que les notes de mathématiques de tous les élèves devaient augmenter d'un point. Écrivez et testez une requête permettant d'effectuer la correction. *Vous vérifierez que la modification a bien été*

effectuée !.

> SOS Pour ajouter un nombre à une valeur, on peut procéder comme en Python pour ajouter un nombre à une variable.

Réponse :

À ce stade, vous devriez obtenir la table `Note` suivante :

id_eleve	nom	prenom	maths	anglais	info
1	Marchand	Alice	14	14	11
2	Muller	Marie	11	18	17
3	Prenel	Laure	14	14	15
4	Dupont	Arthur	19	14	13

Suppression de valeurs

Il est possible de supprimer une ligne d'une table en utilisant `DELETE`.

Exemple

Le client Marchand Grégoire a rendu l'album `Continuum` (dont l'attribut `id_album` vaut 25) qu'il avait emprunté. Il faut supprimer la ligne correspondante dans la table `Emprunt` :

id_client	id_album	jour
1	5	2021-09-10
3	8	2021-08-18
3	24	2021-08-18
5	25	2021-09-12
5	6	2021-10-10
9	20	2021-09-28
11	14	2021-10-08
7	15	2021-10-08
7	19	2021-10-08
7	16	2021-10-15
16	29	2021-10-01

Pour cela, on peut écrire l'ordre suivant :

```
DELETE FROM Emprunt
WHERE id_album = 25;
```

Analyse :

- Après `DELETE` on indique dans quelle table on veut supprimer une ligne avec `FROM [nom_table]` ;
- Ensuite on précise avec `WHERE` la condition permettant de sélectionner les enregistrements à supprimer (ici une seule ligne est supprimée car la condition `id_album = 25` ne correspond qu'à un seul enregistrement).

On peut vérifier que la ligne correspondante a bien été supprimée de la table `Emprunt` :

```
SELECT * FROM Emprunt;
```

id_client	id_album	jour
1	5	2021-09-10
3	8	2021-08-18
3	24	2021-08-18
5	6	2021-10-10
9	20	2021-09-28
11	14	2021-10-08
7	15	2021-10-08
7	19	2021-10-08
7	16	2021-10-15
16	29	2021-10-01

Remarque : Avec le schéma de la base de données il faut aussi mettre à jour la table `Album` puisque l'album en question est à nouveau disponible. La requête de mise à jour suivante permet de faire cela :

```
UPDATE Album
SET dispo = 1
WHERE id_album = 25;
```

À vous de jouer !

Q10 : L'élève Muller Marie ne devrait pas faire partie de la table `Note` car elle ne fait pas partie du même lycée que les autres élèves. Écrivez et testez une requête permettant de supprimer la ligne correspondante. *Vous vérifierez que la suppression a bien été effectuée !*

Réponse :

Exercice 5 - Modification de la base du disquaire

> On travaillera à nouveau avec la base de données `disquaire.db` que vous trouverez dans l'archive de ce chapitre. Vous utiliserez le logiciel *DB Browser for SQLite* ou le logiciel en ligne sqliteonline.com dans lesquels vous aurez au préalable ouvert la base de données en question.

Q1 : Le prénom du client "Robert Pascal" a mal été saisi dans la base de données. En effet, son prénom n'est pas *Pascal* mais *Pascale*. Écrivez et testez une requête permettant corriger cette erreur. *Vous vérifierez que la correction a bien été effectuée !*

Réponse :

Q2 : Un titre d'album a mal été saisi dans la base de données. Il faut remplacer *Riding With The King* par *Riding with the King*. Écrivez et testez une requête permettant corriger cette erreur. *Vous vérifierez que la correction a bien été effectuée !*

Réponse :

Q3 : Le client "Durand Julien" souhaite que les informations personnelles le concernant soient supprimées de la base de données (c'est son droit avec le RGPD). Écrivez et testez la requête permettant d'effectuer ces suppressions. *Vous vérifierez que la suppression a bien été effectuée !*

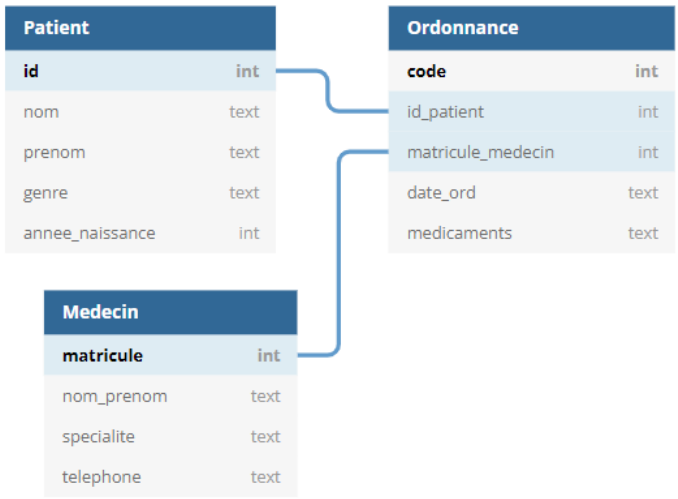
Réponse :

Q4 : Essayez de supprimer le client "Dupont Florine". Quelle est l'erreur provoquée ? Expliquez-la (voir Chapitre 1 si nécessaire).

Réponse :

Exercice 6

On veut créer une base de données `baseHopital.db` qui contiendra les trois tables suivantes (les clés primaires sont en gras) :



On suppose que les dates sont données sous la forme `jj-mm-aaaa`.

Q1 : Donner les commandes SQLite permettant de créer ces tables.

Réponse :

Q2 : On a oublié une colonne pour noter les codes postaux des patients. Donner la commande SQLite permettant cet ajout.

> SOS Vous lirez la page suivant pour savoir comment ajouter une colonne à une table avec `ALTER TABLE`:

> https://sql.sh/cours/alter-table. *Cette commande n'est a priori pas au exigible en Terminale NSI !*

Réponse :

Q3 : Mme Anne Wizeunid, née en 2000 et demeurant 3 rue des Pignons Verts 12345 Avonelit doit être enregistrée comme patiente. Donner la commande SQLite correspondante.

Réponse :

Q4 : Le patient numéro 100 a changé de genre et est maintenant une femme. Donner la commande SQLite modifiant en conséquence ses données.

Réponse :

Q5 : Par souci d'économie, la direction décide de se passer des médecins spécialisés en épidémiologie. Donner la commande permettant de supprimer leurs fiches.

Réponse :

Q6 : Donner la requête permettant d'obtenir la liste des prénoms et noms des patientes habitant le Finistère triées dans l'ordre croissant des âges.

Réponse :

Q7 : Donner la liste des patient(e)s ayant été examiné(e)s par un(e) psychiatre en avril 2020.

Réponse :

Exercice 7 - S'entraîner en ligne

> Vous utiliserez le site développé par Nicolas Revéret, un grand merci à lui pour cette ressource !

Rendez-vous sur la page https://nreveret.forge.apps.education.fr/exercices_bdd/ qui vous permettra de vous frotter à des exercices complets mais aussi de réviser de manière ciblée certaines notions.

Références :

- Exercices sur le langage SQL proposés par Gilles Lassus pour la découverte du site sqlzoo.net (Exercice 3).
- Livre *Prepabac NSI, Tle*, G. Connan, V. Petrov, G. Rozsavolgyi, L. Signac, éditions HATIER, pour l'idée de l'exercice 4 ainsi que l'exercice 6.

Germain BECKER, Lycée Mounier, ANGERS

Voir en ligne : info-mounier.fr/terminale_nsi/base_de_donnees/langage-sql-exercices.php