# 1) introduction

Nous avons eu l'occasion d'étudier la structure d'une base de données relationnelle, nous allons maintenant apprendre à réaliser des requêtes, c'est-à-dire que nous allons apprendre à créer une base des données, créer des attributs, ajouter de données, modifier des données et enfin, nous allons surtout apprendre à interroger une base de données afin d'obtenir des informations.

Pour réaliser toutes ces requêtes, nous allons devoir apprendre un langage de requêtes : SQL (Structured Query Language). SQL est propre aux bases de données relationnelles, les autres types de bases de données utilisent d'autres langages pour effectuer des requêtes.

Dans ce cours nous allons travailler avec SQLite. SQLite est un système de gestion de base de données relationnelle très répandu. Noter qu'il existe d'autres systèmes de gestion de base de données relationnelle comme MySQL ou PostgreSQL. Dans tous les cas, le langage de requête utilisé est le SQL (même si parfois on peut noter quelques petites différences). Ce qui sera vu ici avec SQLite pourra, à quelques petites modifications près, être utilisé avec, par exemple, MySQL.

Pendant ce cours nous allons travailler avec les 2 tables (relations) suivantes :

Table AUTEURS

| id | nom | prenom | ann_naissance | langue_ecriture |
|---|---|---|---|---|
| 1 | Orwell | George | 1903 | anglais |
| 2 | Herbert | Frank | 1920 | anglais |
| 3 | Asimov | Isaac | 1920 | anglais |
| 4 | Huxley | Aldous | 1894 | anglais |
| 5 | Bradbury | Ray | 1920 | anglais |
| 6 | K.Dick | Philip | 1928 | anglais |
| 7 | Barjavel | René | 1911 | français |
| 8 | Boulle | Pierre | 1912 | français |
| 9 | Van Vogt | Alfred Elton | 1912 | anglais |
| 10 | Verne | Jules | 1828 | français |

Table LIVRES

| id | titre | id_auteur | ann_publi | note |
|---|---|---|---|---|
| 1 | 1984 | 1 | 1949 | 10 |
| 2 | Dune | 2 | 1965 | 8 |
| 3 | Fondation | 3 | 1951 | 9 |
| 4 | Le meilleur des mondes | 4 | 1931 | 7 |
| 5 | Fahrenheit 451 | 5 | 1953 | 7 |
| 6 | Ubik | 6 | 1969 | 9 |
| 7 | Chroniques martiennes | 5 | 1950 | 8 |
| 8 | La nuit des temps | 7 | 1968 | 7 |
| 9 | Blade Runner | 6 | 1968 | 8 |

| id | titre | id_auteur | ann_publi | note |
|---|---|---|---|---|
| 10 | Les Robots | 3 | 1950 | 9 |
| 11 | La Planète des singes | 8 | 1963 | 8 |
| 12 | Ravage | 7 | 1943 | 8 |
| 13 | Le Maître du Haut Château | 6 | 1962 | 8 |
| 14 | Le monde des Ā | 9 | 1945 | 7 |
| 15 | La Fin de l'éternité | 3 | 1955 | 8 |
| 16 | De la Terre à la Lune | 10 | 1865 | 10 |

# 2) requêtes d'interrogation

## a) requêtes d'interrogation "simples"

Quand on désire extraire des informations d'une table, on effectue une requête d'interrogation à l'aide du mot clé SELECT. Voici un exemple de requête d'interrogation :

```
SELECT id, titre, id_auteur, ann_publi, note
FROM LIVRES
```

Cette requête va nous permettre d'obtenir l'id, le titre, l'id de l'auteur, l'année de publication et la note de tous les livres présents dans la table LIVRES.

D'une façon générale, le mot clé SELECT est suivi par les attributs que l'on désire obtenir. Le mot clé FROM est suivi par la table concernée.

Il est possible d'obtenir uniquement certains attributs. Par exemple :

```
SELECT nom, prenom
FROM AUTEURS
```

nous permettra d'obtenir le nom et le prénom de tous les auteurs présents dans la table AUTEURS.

À noter qu'il est possible d'obtenir tous les attributs sans être obligé de tous les noter grâce au caractère étoile * :

```
SELECT *
FROM AUTEURS
```

sera équivalent à

```
SELECT id, nom, prenom, ann_naissance, langue_ecriture
FROM AUTEURS
```

## b) sélectionner certaines lignes : la clause WHERE

Il est possible d'utiliser la clause WHERE afin d'imposer une (ou des) condition(s) permettant de sélectionner uniquement certaines lignes.

La condition doit suivre le mot-clé WHERE :

```
SELECT titre
FROM LIVRES
WHERE note > 9
```

La requête ci-dessus permettra d'afficher uniquement les titres qui ont une note strictement supérieure à 9 (soit "1984" et "De la Terre à la Lune")

Il est possible de combiner les conditions à l'aide d'un OR ou d'un AND :

```
SELECT nom
FROM AUTEURS
WHERE langue_ecriture = 'français' AND ann_naissance > 1900
```

Cette requête permet d'obtenir le nom des auteurs nés après 1900 qui écrivent en français (soit ici "Barjavel" et "Boulle")

Il est aussi possible d'utiliser le OR à la place du AND :

```
SELECT nom
FROM AUTEURS
WHERE langue_ecriture = 'français' OR ann_naissance > 1920
```

Cette requête permet d'obtenir le nom des auteurs qui écrivent en français (quelle que soit leur date de naissance) et des auteurs qui n'écrivent pas en français, mais qui sont nés après 1920 (soit ici "Barjavel", "Boulle", "Verne" et "K.Dick")

## c) mettre dans l'ordre les réponses : la clause ORDER BY

Il est possible de classer les résultats d'une requête par ordre croissant grâce à la clause ORDER BY :

```
SELECT nom
FROM AUTEURS
WHERE langue_ecriture = 'français' ORDER BY ann_naissance
```

La requête ci-dessus permettra d'obtenir le nom des auteurs écrivant en français classé en fonction de leur année de naissance (ici on obtiendra donc : "Verne", "Barjavel", "Boulle")

En rajoutant DESC, on obtient l'ordre décroissant :

```
SELECT nom
FROM AUTEURS
WHERE langue_ecriture = 'français' ORDER BY ann_naissance DESC
```

On obtiendra ici : "Boulle", "Barjavel", "Verne"

À noter que si la clause ORDER BY porte sur une chaine de caractères, on obtient alors l'ordre alphabétique :

```
SELECT nom
FROM AUTEURS
WHERE langue_ecriture = 'français' ORDER BY nom
```

On obtiendra : "Barjavel", "Boulle", "Verne"

## d) la clause DISTINCT

Il est possible d'éviter les doublons dans une réponse grâce à la clause DISTINCT. Imaginons la table suivante :

Table MACHINES

| numero | type | proprietaire |
|---|---|---|
| 1 | X23 | Marc |
| 2 | Y43 | Pierre |
| 3 | Z24 | Kevin |

| numero | type | proprietaire |
|---|---|---|
| 4 | Y44 | Marc |

La requête suivante :

```
SELECT proprietaire
FROM MACHINES
```

donnerait le résultat suivant : Marc, Pierre, Kevin, Marc

Nous avons donc un doublon : Marc apparait 2 fois

Pour éviter ce doublon, nous pouvons écrire :

```
SELECT DISTINCT proprietaire
FROM MACHINES
```

qui nous donnera comme résultat : Marc, Pierre, Kevin

## e) les jointures

Nous avons 2 tables, grâce aux jointures nous allons pouvoir associer ces 2 tables dans une même requête.

En général, les jointures consistent à associer des lignes de 2 tables. Elles permettent d'établir un lien entre 2 tables. Qui dit lien entre 2 tables dit souvent clé étrangère et clé primaire.

Analysons la requête suivante :

```
SELECT *
FROM LIVRES
INNER JOIN AUTEURS ON LIVRES.id_auteur = AUTEURS.id
```

Le "FROM LIVRES INNER JOIN AUTEURS" permet de créer une jointure entre les tables LIVRES et AUTEURS ("rassembler" les tables LIVRES et AUTEURS en une seule grande table). Le "ON LIVRES.id_auteur = AUTEURS.id" signifie qu'une ligne quelconque A de la table LIVRES devra être fusionnée avec la ligne B de la table AUTEURS à condition que l'attribut id_auteur de la ligne A soit égal à l'attribut id de la ligne B.

Par exemple, la ligne 1 (id=1) de la table LIVRES (que l'on nommera dans la suite ligne A) sera fusionnée avec la ligne 1 (id=1) de la table AUTEURS (que l'on nommera dans la suite B) car l'attribut id_auteur de la ligne A est égal à 1 et l'attribut id de la ligne B est aussi égal à 1.

Autre exemple, la ligne 1 (id=1) de la table LIVRES (que l'on nommera dans la suite ligne A) ne sera pas fusionnée avec la ligne 2 (id=2) de la table AUTEURS (que l'on nommera dans la suite B') car l'attribut id_auteur de la ligne A est égal à 1 alors que l'attribut id de la ligne B' est égal à 2.

Dans notre exemple l'attribut "id_auteur" de la tables LIVRES est bien une clé étrangère puisque cet attribut correspond à l'attribut "id" de la table "AUTEURS".

La requête ci-dessus permettra d'obtenir le résultat suivant :

| id | titre | id_auteur | ann_publi | note | id | nom | prenom | ann_naissance | langue_ecriture |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1984 | 1 | 1949 | 10 | 1 | Orwell | George | 1903 | anglais |
| 2 | Dune | 2 | 1965 | 8 | 2 | Herbert | Frank | 1920 | anglais |
| 3 | Fondation | 3 | 1951 | 9 | 3 | Asimov | Isaac | 1920 | anglais |
| 4 | Le meilleur des | 4 | 1931 | 7 | 4 | Huxley | Aldous | 1894 | anglais |

| id | titre | id_auteur | ann_publi | note | id | nom | prenom | ann_naissance | langue_ecriture |
|---|---|---|---|---|---|---|---|---|---|
| | mondes | | | | | | | | |
| 5 | Fahrenheit 451 | 5 | 1953 | 7 | 5 | Bradbury | Ray | 1920 | anglais |
| 6 | Ubik | 6 | 1969 | 9 | 6 | K.Dick | Philip | 1928 | anglais |
| 7 | Chroniques martiennes | 5 | 1950 | 8 | 5 | Bradbury | Ray | 1920 | anglais |
| 8 | La nuit des temps | 7 | 1968 | 7 | 7 | Barjavel | René | 1911 | français |
| 9 | Blade Runner | 6 | 1968 | 8 | 6 | K.Dick | Philip | 1928 | anglais |
| 10 | Les Robots | 3 | 1950 | 9 | 3 | Asimov | Isaac | 1920 | anglais |
| 11 | La Planète des singes | 8 | 1963 | 8 | 8 | Boulle | Pierre | 1912 | français |
| 12 | Ravage | 7 | 1943 | 8 | 7 | Barjavel | René | 1911 | français |
| 13 | Le Maître du Haut Château | 6 | 1962 | 8 | 6 | K.Dick | Philip | 1928 | anglais |
| 14 | Le monde des Ā | 9 | 1945 | 7 | 9 | Van Vogt | Alfred Elton | 1912 | anglais |
| 15 | La Fin de l'éternité | 3 | 1955 | 8 | 3 | Asimov | Isaac | 1920 | anglais |
| 16 | De la Terre à la Lune | 10 | 1865 | 10 | 10 | Verne | Jules | 1828 | français |

À noter que pour éviter toute confusion il est souvent judicieux d'ajouter le nom de la table juste devant le nom de l'attribut : on écrira AUTEURS.id au lieu de simplement id, en effet, si on écrivait seulement id, il n'y aurait aucun moyen de distinguer l'id de la table LIVRES et l'id de la table AUTEURS. Je vous conseille d'adopter cette écriture systématiquement en cas de jointure, même quand cela n'est pas obligatoire (par exemple on aurait pu écrire id_auteur à la place de LIVRES.id_auteur puisqu'il y a uniquement un id_auteur dans la table LIVRES), cela vous permettra d'éviter certains déboires.

Dans le cas d'une jointure, il est tout à fait possible de sélectionner certains attributs et pas d'autres (aucune obligation de sélectionner tous les attributs des 2 tables :

```
SELECT LIVRES.titre, AUTEURS.nom, AUTEURS.prenom
FROM AUTEURS
INNER JOIN LIVRES ON LIVRES.id_auteur = AUTEURS.id
```

On obtiendra alors une jointure uniquement avec l'attribut titre de la table LIVRES et les attributs nom, prenom de la table AUTEURS.

| titre | nom | prenom |
|---|---|---|
| 1984 | Orwell | George |
| Dune | Herbert | Frank |
| Fondation | Asimov | Isaac |
| Le meilleur des mondes | Huxley | Aldous |

| titre | nom | prenom |
|---|---|---|
| Fahrenheit 451 | Bradbury | Ray |
| Ubik | K.Dick | Philip |
| Chroniques martiennes | Bradbury | Ray |
| La nuit des temps | Barjavel | René |
| Blade Runner | K.Dick | Philip |
| Les Robots | Asimov | Isaac |
| La Planète des singes | Boulle | Pierre |
| Ravage | Barjavel | René |
| Le Maître du Haut Château | K.Dick | Philip |
| Le monde des Ā | Van Vogt | Alfred Elton |
| La Fin de l'éternité | Asimov | Isaac |
| De la Terre à la Lune | Verne | Jules |

### f) utilisation du WHERE dans les jointures

Suite à une jointure il est possible de sélectionner certaines lignes grâce à la clause WHERE :

```
SELECT LIVRES.titre,AUTEURS.nom, AUTEURS.prenom
FROM LIVRES
INNER JOIN AUTEURS ON LIVRES.id_auteur = AUTEURS.id
WHERE LIVRES.ann_publi>1965
```

On obtient avec cette requête le résultat suivant :

| titre | nom | prenom |
|---|---|---|
| Ubik | K.Dick | Philip |
| La nuit des temps | Barjavel | René |
| Blade Runner | K.Dick | Philip |

### g) les jointures plus complexes

Pour terminer avec les jointures, vous devez savoir que nous avons abordé la jointure la plus simple (INNER JOIN). Il existe des jointures plus complexes (CROSS JOIN, LEFT JOIN, RIGHT JOIN), ces autres jointures ne seront pas abordées dans ce cours.

## 3) requêtes d'insertion

Il est possible d'ajouter une entrée à une table grâce à une requête d'insertion :

```
INSERT INTO LIVRES
(id,titre,id_auteur,ann_publi,note)
VALUES
(17,'Hypérion',11,1989,8);
```

On emploie les mots clés INSERT INTO suivi de la table concernée (ici LIVRES). Ensuite on indique les noms des attributs que l'on désire ajouter (ici (id,titre,auteur,ann_publi,note)), et enfin pour terminer les valeurs de chaque attribut (ici (17,'Hypérion','Simmons',1989,8), attention à bien respecter l'ordre : 17 correspond à id, Hypérion correspond au titre, 11 correspond à id_auteur, 1989 correspond à ann_publi et 8 correspond à note).

# 4) requêtes de mise à jour

“UPDATE” va permettre de modifier une ou des entrées. Nous utiliserons “WHERE”, comme dans le cas d’un “SELECT”, pour spécifier les entrées à modifier. Voici un exemple de modification :

```
UPDATE LIVRES
SET note=7
WHERE titre = 'Hypérion'
```

Cette requête permet de modifier la note du(des) livre(s) ayant pour titre Hypérion

# 5) requêtes de suppression

“DELETE” est utilisée pour effectuer la suppression d’une (ou de plusieurs) entrée(s). Ici aussi c’est le “WHERE” qui permettra de sélectionner les entrées à supprimer :

```
DELETE FROM LIVRES
WHERE titre='Hypérion'
```

Cette requête permet de supprimer le(les) livre(s) ayant pour titre Hypérion

Attention à l’utilisation de cette requête DELETE notamment si on oublie le WHERE. Un :

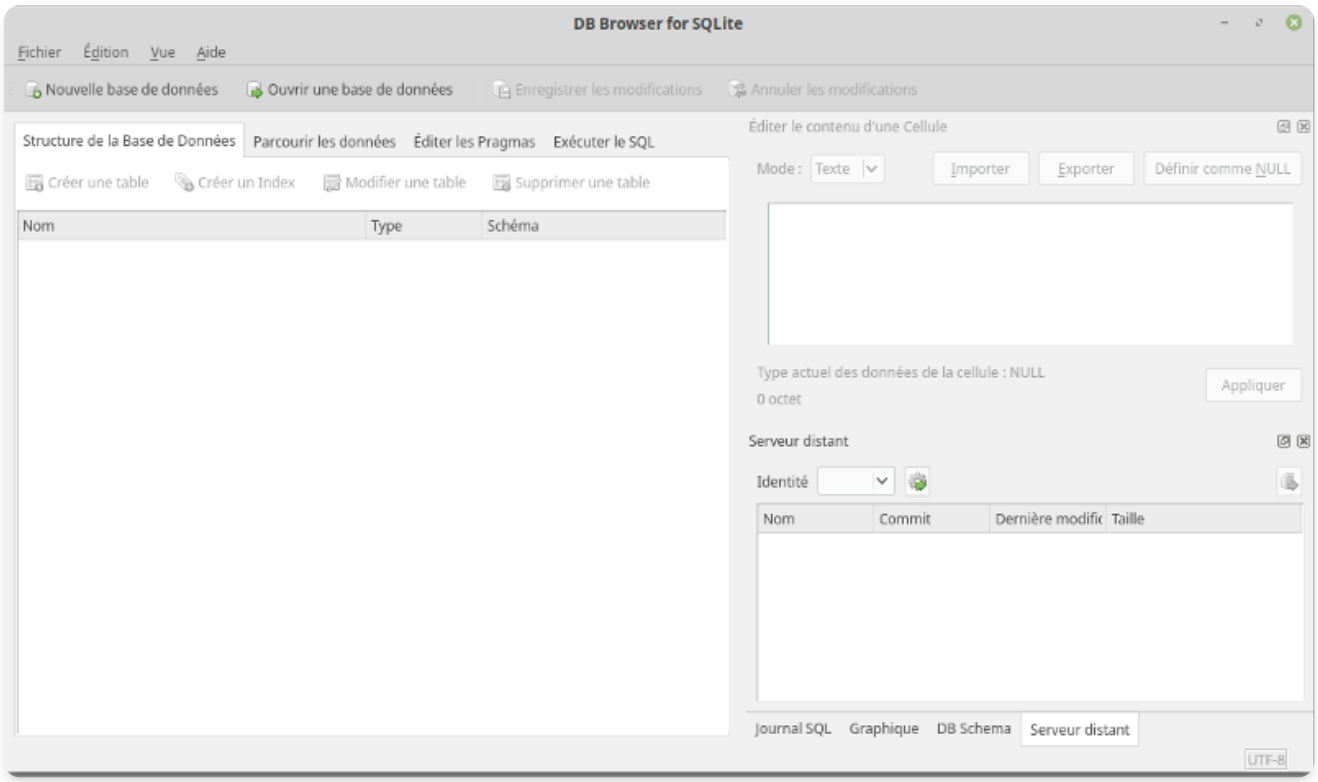
```
DELETE FROM LIVRES
```

supprimerait toutes les entrées de la table LIVRES

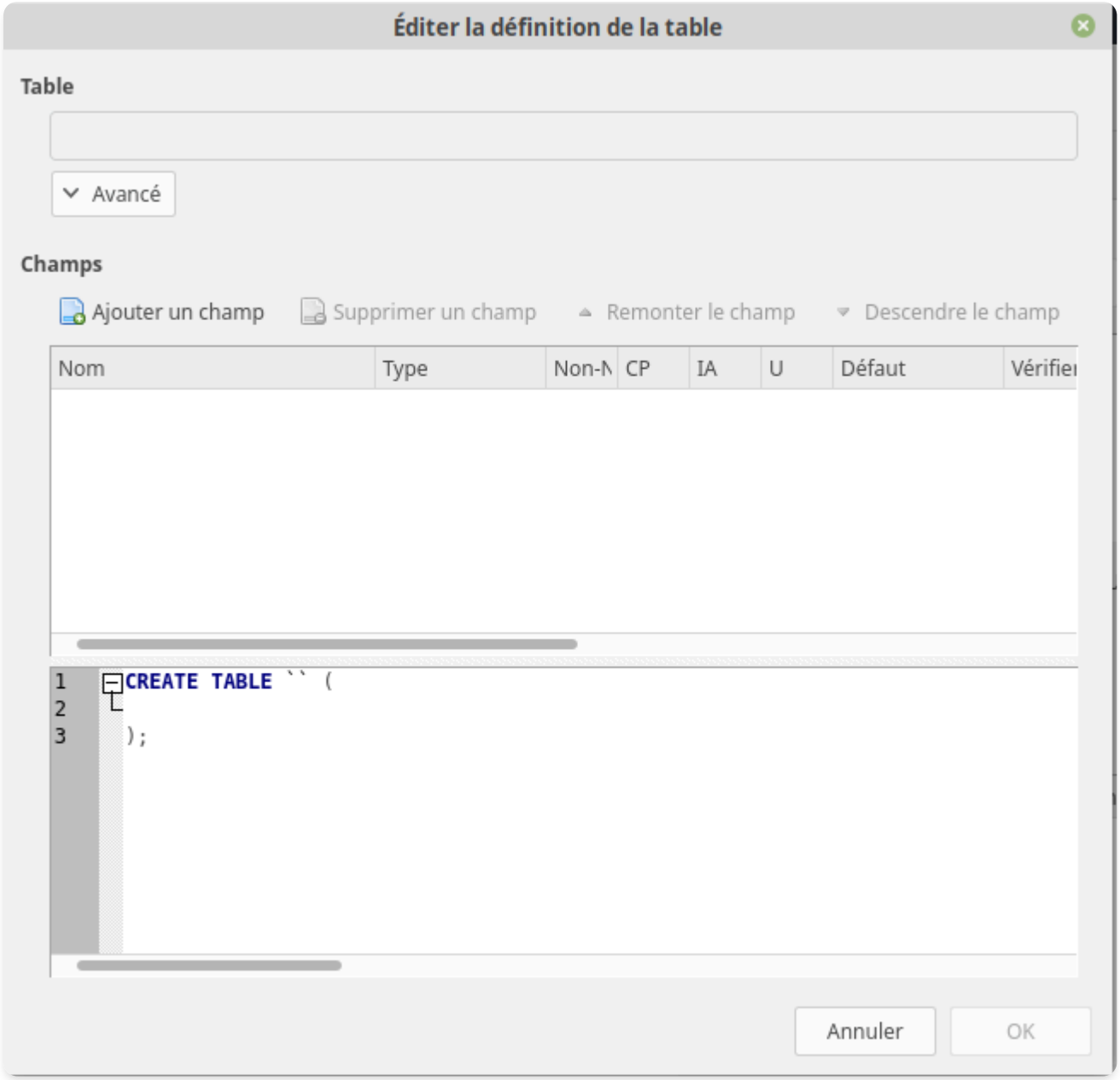
ir créer une base de données et effectuer des requêtes sur cette dernière, nous a
utiliser le logiciel “DB Browser for SQLite” : https://sqlitebrowser.org/

## activité 3.1

Après avoir lancé le logiciel “DB Browser for SQLite”, vous devriez obtenir ceci :

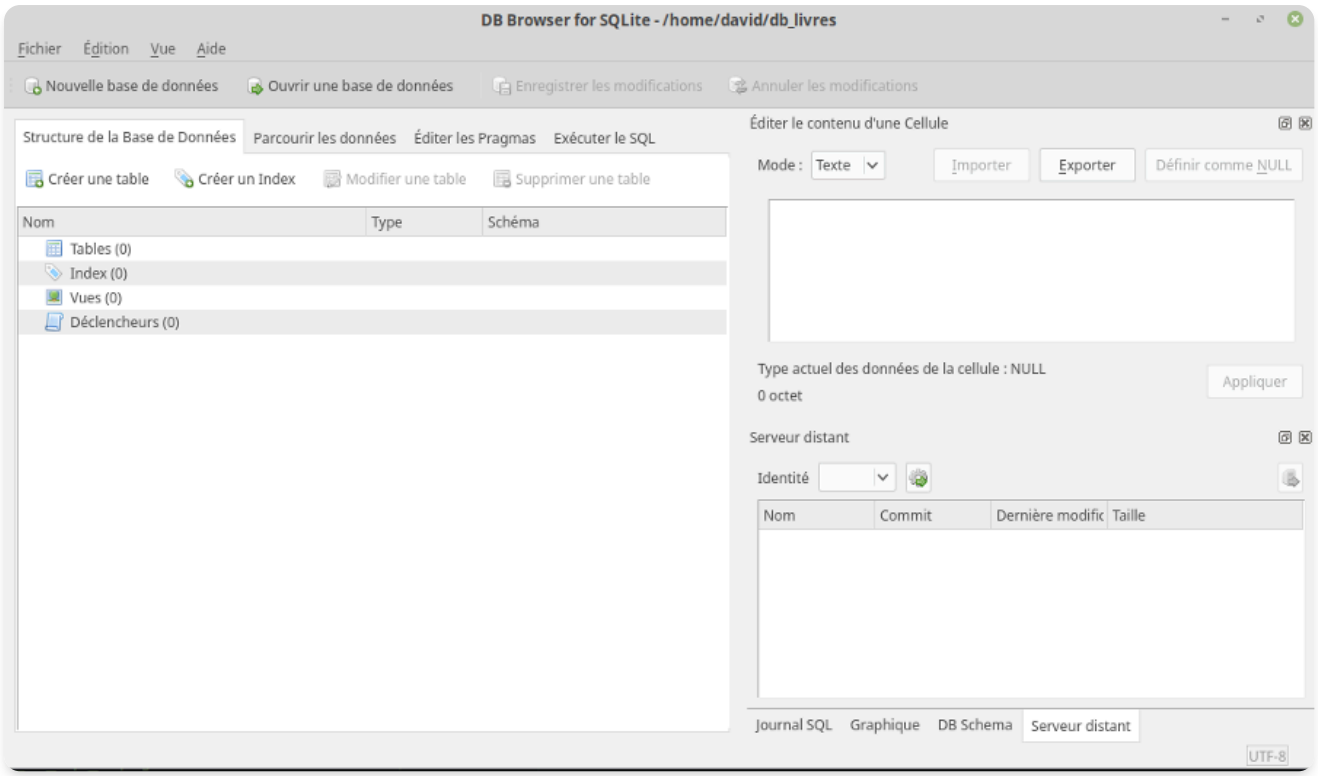



Cliquez sur Nouvelle base de données. Après avoir choisi un nom pour votre base de données (par exemple “db_livres.db”), vous devriez avoir la fenêtre suivante :

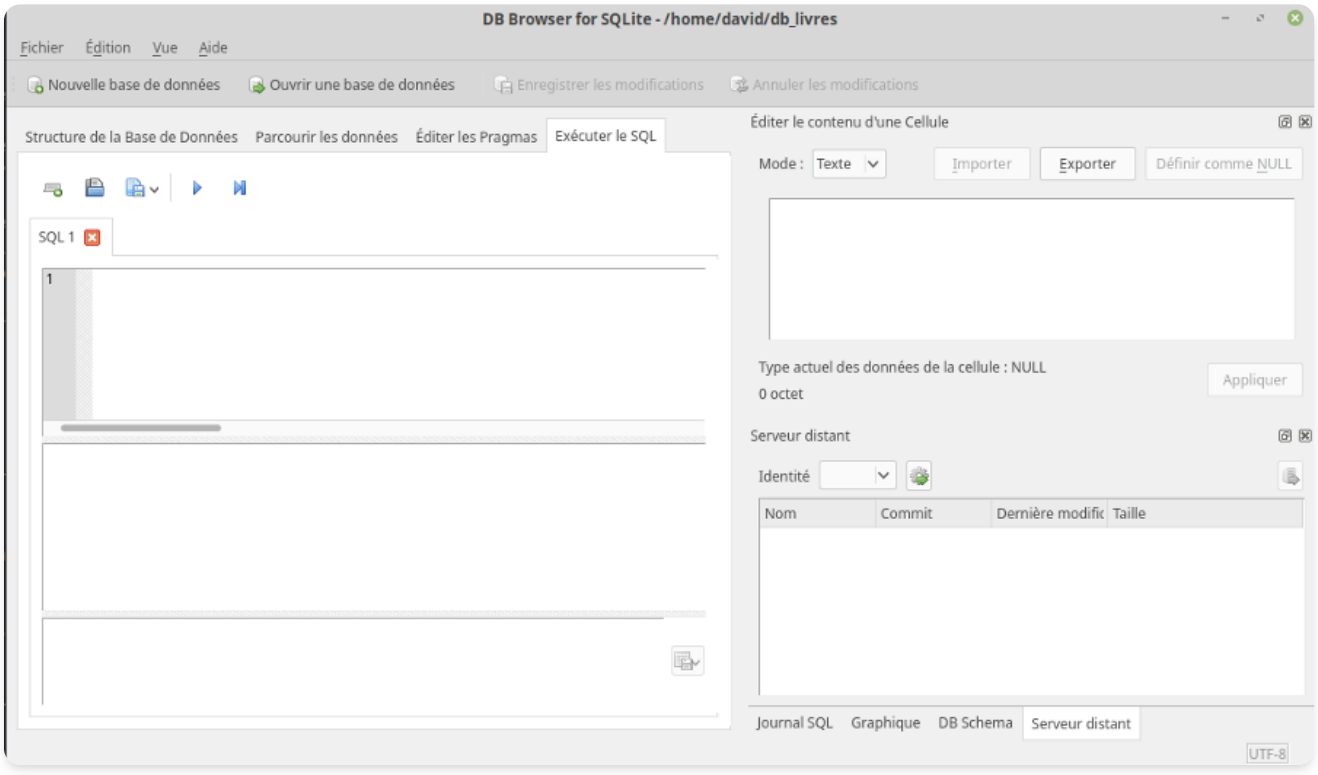



Cliquez alors sur Annuler

Notre base de données a été créée :



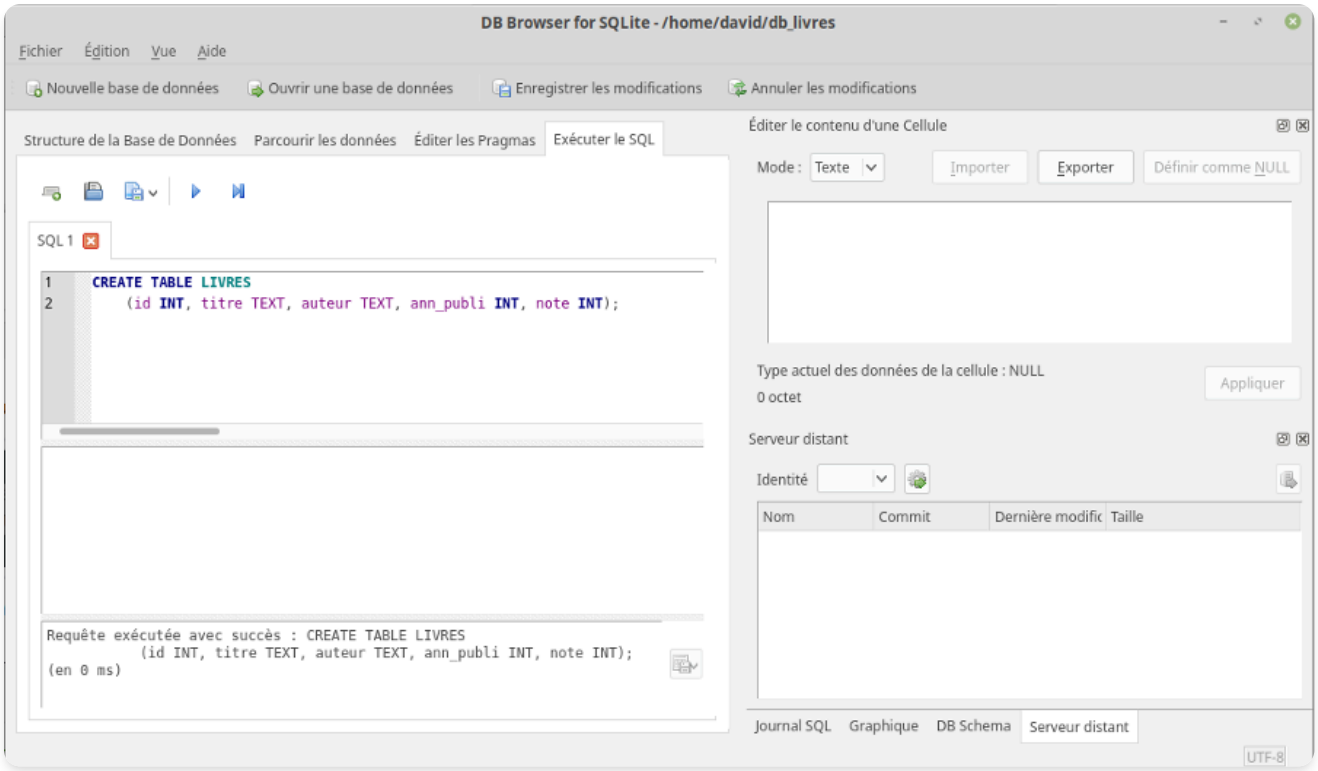

mais pour l'instant elle ne contient aucune table (aucune relation), pour créer une table, cliquez sur l'onglet "Exécuter le SQL". On obtient alors :




Copiez-collez le texte ci-dessous dans la fenêtre "SQL 1"

```
CREATE TABLE LIVRES
    (id INT, titre TEXT, auteur TEXT, ann_publi INT, note INT, PRIMARY KEY (id));
```

Cliquez ensuite sur le petit triangle situé au-dessus de la fenêtre SQL 1 (ou appuyez sur F5), vous devriez avoir ceci :




Comme indiqué dans la fenêtre, “Requête exécutée avec succès” !

Vous venez de créer votre première table.

Revenons sur cette première requête :

Le *CREATE TABLE LIVRES* ne devrait pas vous poser de problème : nous créons une nouvelle table nommée “LIVRES”.

La suite est à peine plus complexe :

nous créons ensuite les attributs :

- id
- titre
- auteur
- ann_publi
- note

Nous avons pour chaque attribut précisé son domaine : id : entier (INT), titre : chaîne de caractères (TEXT), auteur : chaîne de caractères, ann_publi : entier et note : entier

L’attribut “id” va jouer ici le rôle de clé primaire, nous avons donc précisé dans notre requête que id jouera le rôle de clé primaire (PRIMARY KEY (id)). Notre système de gestion de base

de données nous avertira si l'on tente d'attribuer 2 fois la même valeur à l'attribut"id".

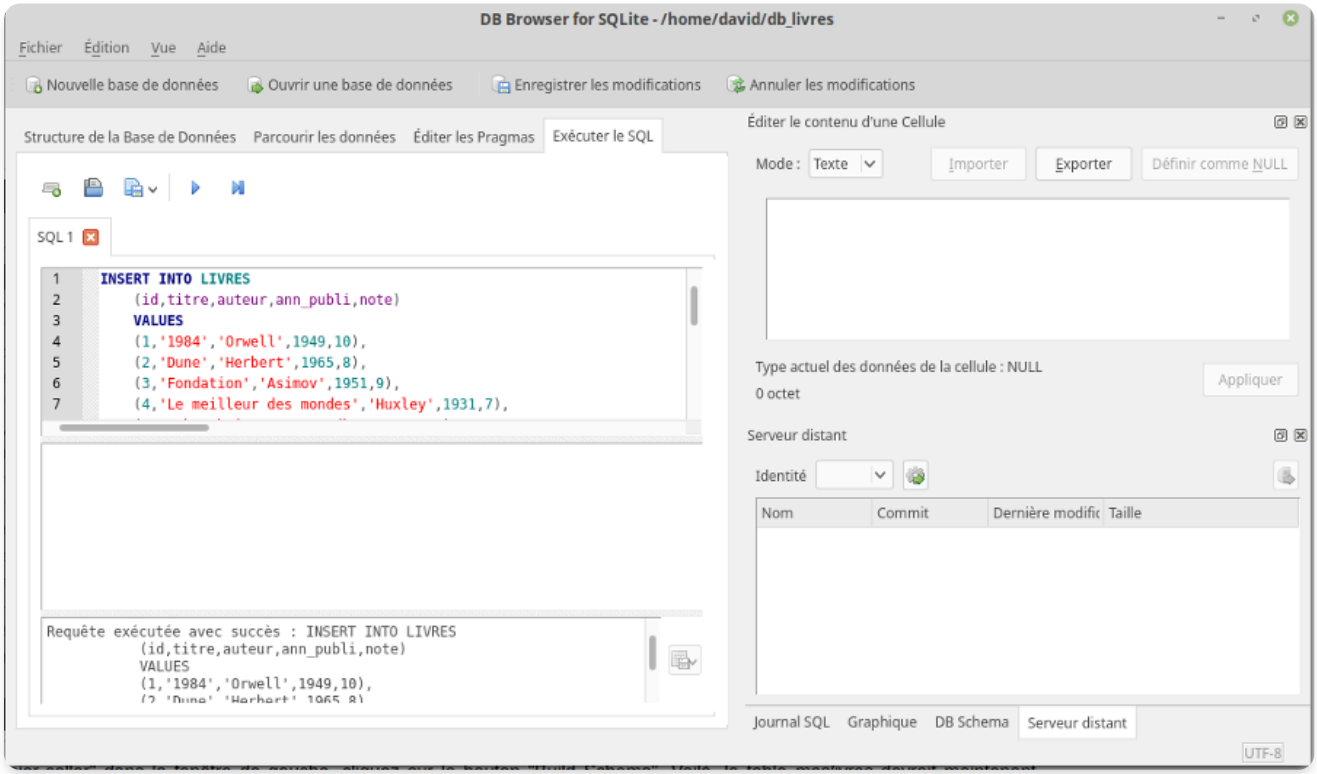
## activité 3.2

Nous allons maintenant ajouter des données à notre table.

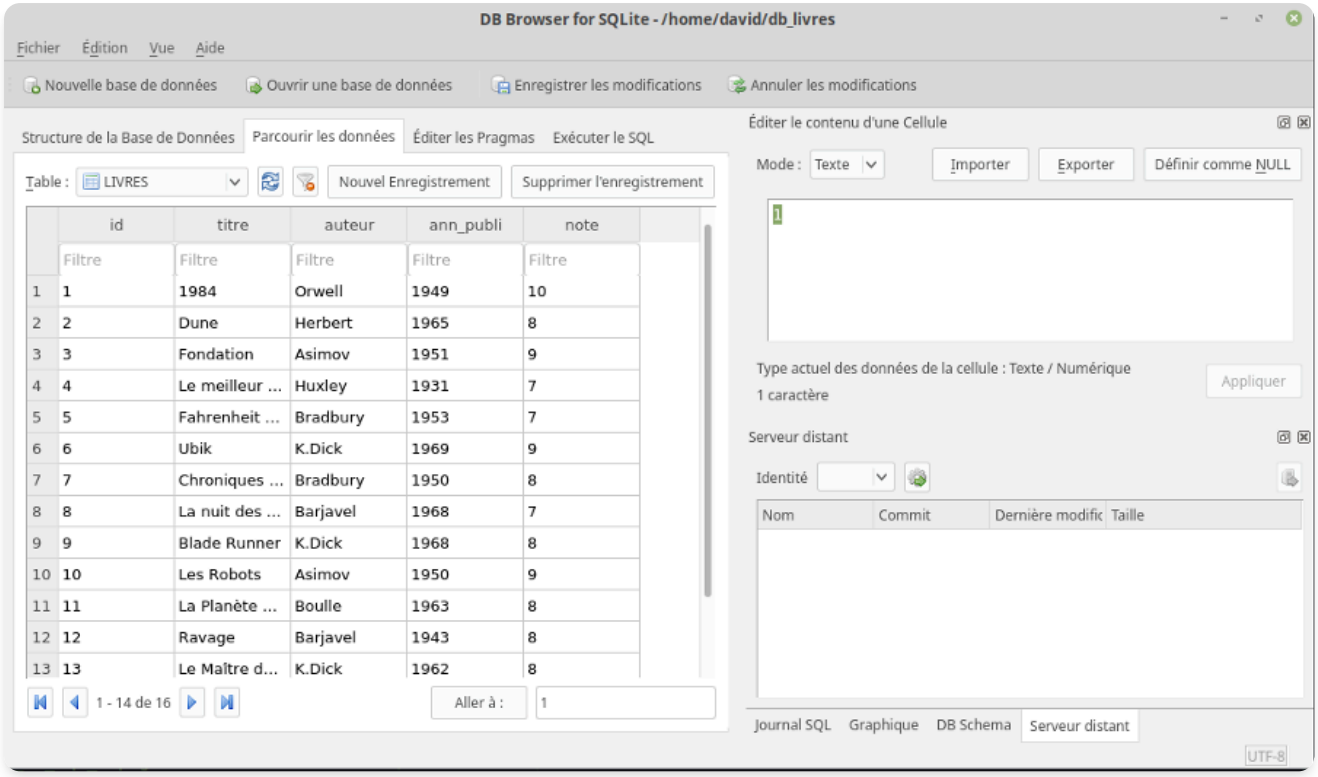
Toujours dans l'onglet "Exécuter le SQL", après avoir effacé la fenêtre SQL 1, copiez-collez dans cette même fenêtre la requête ci-dessous :

```
INSERT INTO LIVRES
    (id,titre,auteur,ann_publi,note)
    VALUES
    (1,'1984','Orwell',1949,10),
    (2,'Dune','Herbert',1965,8),
    (3,'Fondation','Asimov',1951,9),
    (4,'Le meilleur des mondes','Huxley',1931,7),
    (5,'Fahrenheit 451','Bradbury',1953,7),
    (6,'Ubik','K.Dick',1969,9),
    (7,'Chroniques martiennes','Bradbury',1950,8),
    (8,'La nuit des temps','Barjavel',1968,7),
    (9,'Blade Runner','K.Dick',1968,8),
    (10,'Les Robots','Asimov',1950,9),
    (11,'La Planète des singes','Boulle',1963,8),
    (12,'Ravage','Barjavel',1943,8),
    (13,'Le Maître du Haut Château','K.Dick',1962,8),
    (14,'Le monde des Ã','Van Vogt',1945,7),
    (15,'La Fin de l'éternité','Asimov',1955,8),
    (16,'De la Terre à la Lune','Verne',1865,10);
```

Un message devrait vous préciser que votre requête a été exécutée avec succès :



La table LIVRES contient bien les données souhaitées (onglet “Parcourir les données”) :



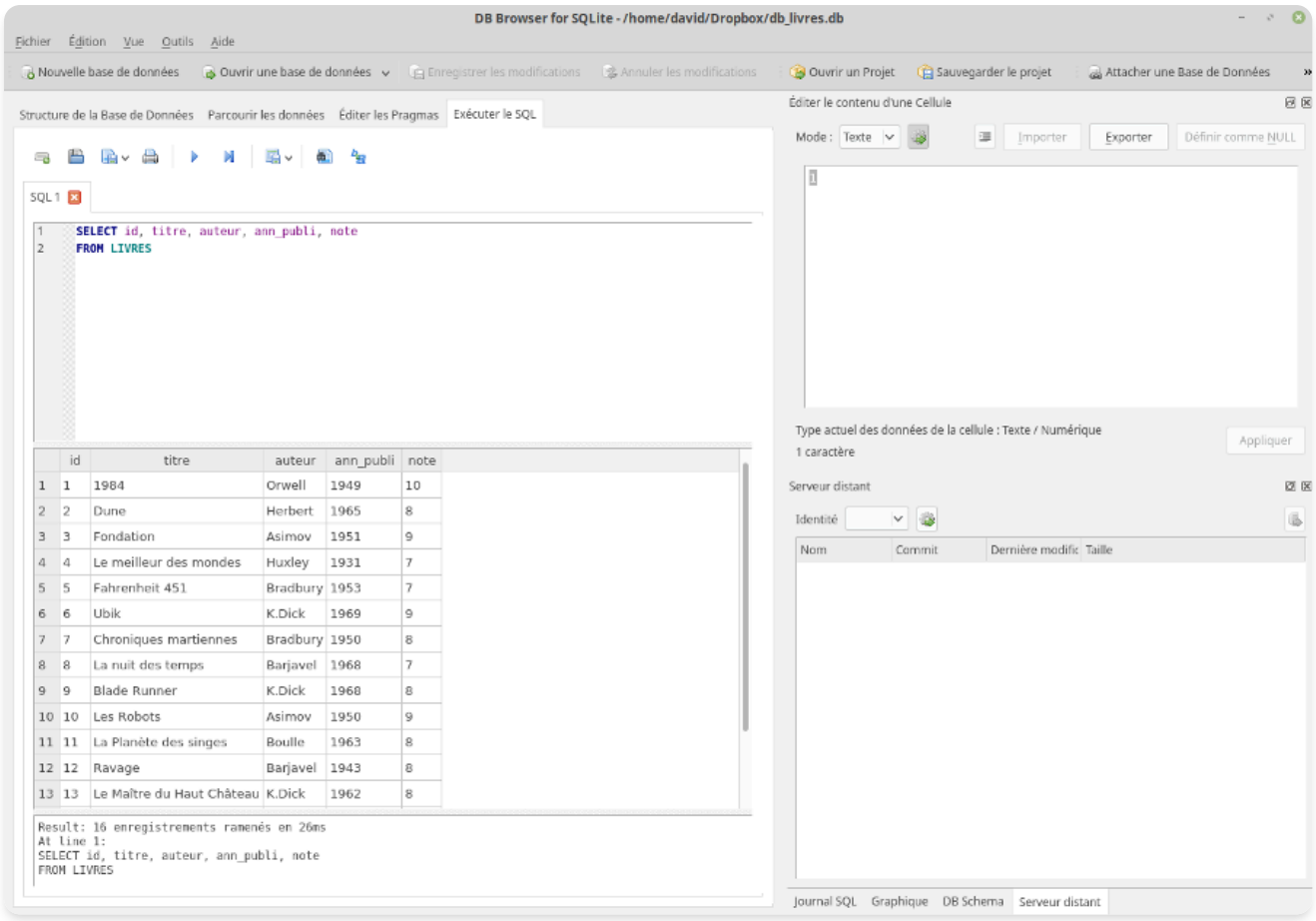

## activité 3.3

Saisissez la requête SQL suivante :

```
SELECT id, titre, auteur, ann_publi, note
FROM LIVRES
```

puis appuyez sur le triangle (ou la touche F5)

Après un temps plus ou moins long, vous devriez voir s’afficher ceci :




## activité 3.4

Effectuez une requête qui permettra d'obtenir le titre et l'auteur de tous les livres présents dans la table LIVRES.

## activité 3.5

Saisissez et testez la requête SQL suivante :

```
SELECT titre, ann_publi
FROM LIVRES
WHERE auteur='Asimov'
```

Vérifiez que vous obtenez bien uniquement les livres écrits par Isaac Asimov.

## activité 3.6

Écrivez et testez une requête permettant d'obtenir uniquement les titres des livres écrits par Philip K.Dick.

## activité 3.7

Saisissez et testez la requête SQL suivante :

```
SELECT titre, ann_publi
FROM LIVRES
WHERE auteur='Asimov' AND ann_publi>1953
```

Vérifiez que nous obtenons bien le livre écrit par Asimov publié après 1953.

## activité 3.8

Écrivez une requête permettant d'obtenir les titres des livres publiés après 1945 qui ont une note supérieure ou égale à 9.

## activité 3.9

Écrivez une requête SQL permettant d'obtenir les livres de K.Dick classés du plus ancien au plus récent.

## activité 3.10

Créez une nouvelle base de données que vous nommerez par exemple db_livres_auteurs.db, puis créez une table AUTEURS à l'aide de la requête SQL suivante :

```
CREATE TABLE AUTEURS
(id INT, nom TEXT, prenom TEXT, ann_naissance INT, langue_ecriture TEXT, PRIMARY
```

Créez ensuite une deuxième table (LIVRES) :

```
CREATE TABLE LIVRES
(id INT, titre TEXT, id_auteur INT, ann_publi INT, note INT, PRIMARY KEY (id), FO
```

Comme vous l'avez sans doute remarqué nous avons précisé dans notre requête que l'attribut "id_auteur" jouera le rôle de clé étrangère : liaison entre "id_auteur" de la table LIVRES et "id" de la table AUTEURS (FOREIGN KEY (id_auteur) REFERENCES AUTEURS(id)).

Ajoutez des données à la table AUTEURS à l'aide de la requête SQL suivante :

```
INSERT INTO AUTEURS
(id,nom,prenom,ann_naissance,langue_ecriture)
VALUES
(1,'Orwell','George',1903,'anglais'),
(2,'Herbert','Frank',1920,'anglais'),
(3,'Asimov','Isaac',1920,'anglais'),
(4,'Huxley','Aldous',1894,'anglais'),
(5,'Bradbury','Ray',1920,'anglais'),
(6,'K.Dick','Philip',1928,'anglais'),
```

```
(7,'Barjavel','René',1911,'français'),
(8,'Boulle','Pierre',1912,'français'),
(9,'Van Vogt','Alfred Elton',1912,'anglais'),
(10,'Verne','Jules',1828,'français');
```

Ajoutez des données à la table LIVRES à l'aide de la requête SQL suivante :

```
INSERT INTO LIVRES
(id,titre,id_auteur,ann_publi,note)
VALUES
(1,'1984',1,1949,10),
(2,'Dune',2,1965,8),
(3,'Fondation',3,1951,9),
(4,'Le meilleur des mondes',4,1931,7),
(5,'Fahrenheit 451',5,1953,7),
(6,'Ubik',6,1969,9),
(7,'Chroniques martiennes',5,1950,8),
(8,'La nuit des temps',7,1968,7),
(9,'Blade Runner',6,1968,8),
(10,'Les Robots',3,1950,9),
(11,'La Planète des singes',8,1963,8),
(12,'Ravage',7,1943,8),
(13,'Le Maître du Haut Château',6,1962,8),
(14,'Le monde des Ã',9,1945,7),
(15,'La Fin de l'éternité',3,1955,8),
(16,'De la Terre à la Lune',10,1865,10);
```

## activité 3.11

Saisissez et testez la requête SQL suivante :

```
SELECT titre,nom, prenom
FROM LIVRES
INNER JOIN AUTEURS ON LIVRES.id_auteur = AUTEURS.id
```

Rappel : attention, si un même nom d'attribut est présent dans les 2 tables (par exemple ici l'attribut id), il est nécessaire d'ajouter le nom de la table devant afin de pouvoir les distinguer (AUTEURS.id et LIVRES.id).

## activité 3.12

Écrivez une requête SQL permettant d'obtenir les titres des livres publiés après 1945 ainsi que le nom de leurs auteurs.

## activité 3.13

Nous allons repartir avec une nouvelle base de données. Créez une nouvelle base de données nommée “db_livres.db”.

Créez ensuite une table LIVRES à l’aide de la requête suivante :

```
CREATE TABLE LIVRES
(id INT, titre TEXT, auteur TEXT, ann_publi INT, note INT, PRIMARY KEY (id));
```

Ajoutez des données à la table LIVRES à l’aide de la requête SQL suivante :

```
INSERT INTO LIVRES
(id,titre,auteur,ann_publi,note)
VALUES
(1,'1984','Orwell',1949,10),
(2,'Dune','Herbert',1965,8),
(3,'Fondation','Asimov',1951,9),
(4,'Le meilleur des mondes','Huxley',1931,7),
(5,'Fahrenheit 451','Bradbury',1953,7),
(6,'Ubik','K.Dick',1969,9),
(7,'Chroniques martiennes','Bradbury',1950,8),
(8,'La nuit des temps','Barjavel',1968,7),
(9,'Blade Runner','K.Dick',1968,8),
(10,'Les Robots','Asimov',1950,9),
(11,'La Planète des singes','Boulle',1963,8),
(12,'Ravage','Barjavel',1943,8),
(13,'Le Maître du Haut Château','K.Dick',1962,8),
(14,'Le monde des Ã','Van Vogt',1945,7),
(15,'La Fin de l’éternité','Asimov',1955,8),
(16,'De la Terre à la Lune','Verne',1865,10);
```

## activité 3.14

À l’aide d’une requête SQL, ajoutez à la table LIVRES le livre suivant :

- id : 17
- titre : “2001 : L’Odyssée de l’espace”
- auteur : “Clarcke”
- année de publication : 1968
- note : 7

## activité 3.15

Écrivez une requête permettant d’attribuer la note de 10 à tous les livres écrits par Asimov publiés après 1950. Testez cette requête.

## activité 3.16

Écrivez une requête permettant de supprimer les livres publiés avant 1945. Testez cette requête.

## exercice 3.1

Un ski-club utilise une base de données constituée de 2 tables :

- une table ADHERENTS
- une table STATIONS

Dans la table ADHERENTS on trouve un attribut “ref_station” qui permet de connaître les stations de ski préférées des adhérents.

Table ADHERENTS

| num_licence | nom | prenom | annee_naissance | ref_station |
|---|---|---|---|---|
| 12558 | Doe | John | 1988 | 5 |
| 13668 | Vect | Alice | 1974 | 6 |
| 1777 | Dect | Bob | 1967 | 3 |
| 13447 | Beau | Tristan | 1999 | 4 |
| 1141 | Pabeau | John | 1975 | 3 |

table STATIONS

| ref | nom | altitude_max |
|---|---|---|
| 3 | Le grand Bornand | 2050 |
| 4 | La clusaz | 2616 |
| 5 | Flaine | 2510 |
| 6 | Avoriaz | 2466 |

1. Comment appelle-t-on l’attribut ref_station de la table ADHERENTS ?
2. Écrire la requête SQL permettant d’obtenir le nom des stations ayant une altitude maxi strictement supérieure à 2500 m.
3. Écrire une requête SQL permettant d’obtenir le numéro de licence des adhérents nés après 1980 et ayant pour prénom John.
4. Donnez le résultat de la requête SQL suivante :

```
SELECT nom
FROM ADHERENTS
WHERE num_licence > 2000 OR  ref_station = 3
```

5. Donnez le résultat de la requête SQL suivante :

```
SELECT STATIONS.nom
FROM STATIONS
INNER JOIN ADHERENTS ON ADHERENTS.ref_station = STATIONS.ref
WHERE annee_naissance > 1975
```

## exercices du bac

- Sujet 1 2021 Exercice 4
- Sujet 2 2021 Exercice 3
- Sujet 3 2021 Exercice 2
- Sujet 4 2021 Exercice 1
- Sujet 5 2021 Exercice 1
- Sujet 7 2021 Exercice 4
- Sujet 8 2021 Exercice 1
- Sujet 10 2021 Exercice 3
- Sujet 1 2022 Exercice 3
- Sujet 2 2022 Exercice 2
- Sujet 3 2022 Exercice 4
- Sujet 4 2022 Exercice 1
- Sujet 5 2022 Exercice 3
- Sujet 6 2022 Exercice 4
- Sujet 7 2022 Exercice 3
- Sujet 8 2022 Exercice 2
- Sujet 9 2022 Exercice 4
- Sujet 10 2022 Exercice 3
- Sujet 11 2022 Exercice 2
- Sujet 12 2022 Exercice 3
- Sujet 13 2022 Exercice 1
- Sujet 14 2022 Exercice 3

## Ce qu'il faut savoir

- Pour consulter des données, ajouter une entrée, modifier une entrée ou supprimer une entrée dans une base de données relationnelle, il est nécessaire d'effectuer des "requêtes SQL" (utilisation du langage SQL)
- Pour ajouter des entrées à une table, on utilisera "INSERT" (exemple : INSERT INTO LIVRES (id,titre,auteur,ann_publi,note) VALUES (1,'1984','Orwell',1949,10);)
- Pour interroger une table, on utilisera "SELECT" (exemple : SELECT titre FROM LIVRES WHERE auteur='Asimov')
- Pour modifier une entrée, on utilisera "UPDATE" (exemple : UPDATE LIVRES SET note=7 WHERE titre = 'Hypérion')
- Pour supprimer une entrée, on utilisera "DELETE" (exemple : DELETE FROM LIVRES WHERE titre='Hypérion')
- Pour réaliser une jointure, il est possible d'utiliser "INNER JOIN" (exemple : SELECT FROM LIVRES INNER JOIN AUTEURS ON LIVRES.id_auteur = AUTEURS.id)

## Ce qu'il faut savoir faire

- Vous devez être capable d'effectuer des requêtes SQL simples (utilisation de "INSERT", "SELECT", "UPDATE" et "DELETE")
- Vous devez être capable d'effectuer une jointure entre 2 tables (utilisation de "INNER JOIN")