

[Logiciel](#)

SQLite : une base de données SQL

Pré-requis

Installation

- Installez le paquet  **sqlite3** ou en ligne de commande :

```
...@...:~$ sudo apt install sqlite3
```

Configuration

Utilisation

Avec un client graphique : SQLiteManager

SQLiteManager : un client graphique pour gérer les bases SQLite

En ligne de commande

Lancez dans un terminal :

```
...@...:~ $ sqlite3
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>
```

Le prompt change et signale que vous êtes dans le **terminal SQLite** :

Lister les commandes

- **.help** : Lister les commandes :

```
sqlite> .help
```

`.archive ...`

Manage SQL archives

Quitter sqlite

- **.exit** : Quitter le programme sqlite
- **.quit** : Quitter le programme sqlite

Modifier le format de sortie

- **.mode MODE ?TABLE?** : Mode de sortie, MODE est à choisir parmi :
 - **csv** : valeurs séparées par des virgules
 - **column** : colonnes alignées à gauche (voir `.width`)
 - **html** : `<table>` (code HTML)

```
sqlite> .mode html
```

```
<TR><TH>id</TH>
<TH>titre</TH>
<TH>auteur</TH>
<TH>resume</TH>
<TH>num</TH>
<TH>timent</TH>
<TH>éditeur</TH>
</TR>
<TR><TD>1</TD>
<TD>tintin au congo</TD>
<TD>hergé</TD>
<TD>Tintin est au congo.</TD>
<TD>5.0</TD>
<TD></TD>
<TD>casterman</TD>
</TR>
<TR><TD>2</TD>
<TD>le nid des marsupilamis</TD>
<TD>franquin</TD>
<TD>Un reportage incroyable</TD>
<TD>6.0</TD>
<TD></TD>
<TD></TD>
</TR>
<TR><TD>3</TD>
<TD>la déesse</TD>
<TD>moebius</TD>
<TD>une aventure extraordinaire</TD>
<TD>7.0</TD>
<TD>2011-02-03</TD>
<TD></TD>
</TR>
```

affichera :

- **insert** : commande SQL **insert** pour la table TABLE
- **line** : Une valeur par ligne
- **[list]** : Valeurs délimitées par la chaîne de séparation `.separator`

- **tabs** : valeurs séparées par des tabulations
- **tcl** : Liste TCL des éléments

Afficher le nom des colonnes / Changer l'aspect des colonnes (.mode column)

- **.header(s) on|[off]** : Affiche (ou non) les titres. Par exemple,

```
sqlite> .header on
sqlite> .mode column
```

| nom | age | membre |
|-----|-----|--------|
| dan | 23 | oui |
| bob | 45 | non |

affichera les résultats comme ceci :

L'affichage par défaut

```
sqlite> .header off
sqlite> .mode list
```

```
dan, 23, oui
bob, 45, non
```

affiche comme ceci :

- **.width NUM1 NUM2 ...** : largeur des colonnes [par défaut, 10 caractères]. Par exemple,

```
sqlite> .width 2 15 10 20 3
```

affichera:

| id | titre | auteur | resume | num | date_creation | éditeur |
|----|-----------------|----------|-----------------|-----|---------------|-----------|
| 1 | tintin au congo | hergé | Tintin est au c | 5.0 | | casterman |
| 2 | le nid des mars | franquin | Un reportage in | 6.0 | | |
| 3 | la déesse | moebius | une aventure ex | 7.0 | 2011-02-03 | |

- **.separator STRING** : change le séparateur utilisé par le mode de sortie et par .import. En mode liste,

```
sqlite> .separator ", "
```

```
dan, 23, oui
bob, 45, non
```

affichera :

Rappel des paramètres

- **.show** Affiche les valeurs actuelles des différents paramètres :

```
sqlite> .show
echo: off
eqp: off
explain: auto
headers: off
mode: html
```

```

nullvalue: ""
output: stdout
colseparator: "|"
rowseparator: "\n"
stats: off
width:
filename: :memory:

```

Gestion des bases

```

* **.databases** : Liste les noms et les fichiers des bases de données associées
* **.tables ?TABLE?** : Liste les noms des tables dans la base courante. Si TABLE est spécifié, ne liste que les tables correspondant au motif TABLE.
* **.import FILE TABLE** : Importer les données de FILE dans TABLE
* **.indices ?TABLE?** : Afficher les noms de tous les indices ; si TABLE est spécifiée, n'affiche que les indices correspondant au motif TABLE
* **.output FILENAME** : Envoyer la sortie vers FILENAME<cli prompt='>'>sqlite> .output bd.txt

```

```
sqlite> select * from bd; sqlite> cat bd.txt sqlite> .quit</cli>
```

- **.output stdout** : Envoyer la sortie vers l'écran

Dumper une table depuis SQLite en format SQL pour sauvegarder la structure et les données sur un disque

- **.dump ?TABLE? ...** : Dump de la base de données dans un format texte SQL. Si TABLE est spécifié, ne dumper que les tables correspondant au motif TABLE

```
sqlite> .dump bd
```

```

PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE "bd" (id integer primary key, titre VARCHAR(30), auteur VARCHAR(30), resume TEXT, num double, date_creation date);
INSERT INTO "bd" VALUES(1,'tintin au congo','hergé','Tintin est au congo.',5.0,NULL);
INSERT INTO "bd" VALUES(2,'le nid des marsupilamis','franquin','Un reportage incroyable',6.0,'2011-02-04');
INSERT INTO "bd" VALUES(3,'la déesse','moebius','une aventure géniale',7.0,'2011-02-04 17:06:23');
COMMIT;

```

- Rediriger la sortie vers un fichier puis dumper la table depuis SQLite

```

sqlite> .output bd.sql
sqlite> .dump bd

```

Le résultat n'est plus affiché dans le terminal, mais redirigé vers le fichier bd.sql. Pour le vérifier il suffit d'afficher le contenu du fichier:

```

sqlite> .quit
...@...:~ $ cat bd.sql

```

- Lire directement un fichier dumpé depuis sqlite :
 - Effacez la table de la base:

```
sqlite> drop table bd;
```

- Puis lisez le fichier sauvegardé:

```
sqlite> .read bd.sql  
sqlite> select * from bd;
```

Affichera:

```
1|tintin au congo|hergé|Tintin est au congo.|5.0|  
2|le nid des marsupilamis|franquin|Un reportage incroyable|6.0|2011-02-04  
3|la déesse|moebius|une aventure géniale|7.0|2011-02-04 17:06:23
```

3. Dumper une base en format SQL pour sauvegarder sa structure, ses tables et ses données :

```
....$ sqlite3 livres.db .dump > livres.sql
```

4. Récupérer un fichier dumpé pour recréer la base :

- Effacez la base originale :

```
...@...:~ $ rm -r livres.db
```

- Récupérez la base depuis le fichier de sql :

```
...@...:~ $ sqlite3 livres.db < livres.sql
```

- Connectez-vous à la base< :

```
...@...:~ $ sqlite3 livres.db
```

- Faites une requête pour vérification :

```
sqlite> select * from bd;
```

Affichera:

```
1|tintin au congo|hergé|Tintin est au congo.|5.0|  
2|le nid des marsupilamis|franquin|Un reportage incroyable|6.0|2011-02-04  
3|la déesse|moebius|une aventure géniale|7.0|2011-02-04 17:06:23
```

Manipuler une base

- **Créer une base - ouvrir une base** : Lancez sqlite3 avec le nom de la base :

```
...@...:~ $ sqlite3 livres.db
```

Si la base n'existe pas, elle sera créée. Toutes les commandes qui suivront concerneront cette base.

- **Détruire une base** : Il suffit d'effacer son fichier .db :

```
...@...:~ $ rm livres.db
```

- **Créer une table** : Dans sqlite, dans une base existante, lancer la commande :

```
sqlite> CREATE TABLE bandedessinée (id integer primary  
key, titre VARCHAR(30), auteur VARCHAR(30), resume TEXT,  
num double, date_creation date);
```

Si le prompt apparaît après avoir tapé la commande, c'est qu'il manque le “;” à la fin de la requête. Ajoutez-le juste après le prompt validez.
Les types de données SQLite3 sont : NULL, INTEGER, REAL, TEXT et BLOB. Ce qui donne par exemple :

```
sqlite> CREATE TABLE bandedessinée (id integer primary  
key, titre TEXT, auteur TEXT, resume TEXT, num REAL,  
date_creation INTEGER);
```

- **Insérer des valeurs dans la table** : Un exemple :

```
sqlite> INSERT INTO "bandedessinée" VALUES(1, 'tintin au  
congo', 'hergé', 'Tintin est au congo.', 5.0, NULL);  
sqlite> INSERT INTO "bandedessinée" VALUES(2, 'le nid des  
marsupilamis', 'franquin', 'Un reportage incroyable',  
6.0, date('now'));  
sqlite> INSERT INTO "bandedessinée" VALUES(3, 'la  
déesse', 'moebius', 'une aventure géniale', 7.0,  
strftime("%Y-%m-%d %H:%M:%S", 'now', 'localtime'));
```

- **Simple requête pour visualiser le contenu de la table** :

```
sqlite> select * from bandedessinée;
```

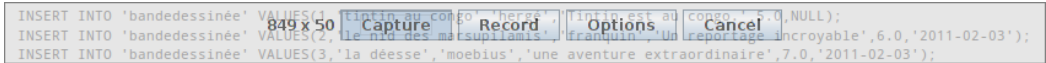
affichera :

```
1|tintin au congo|hergé|Tintin est au congo.|5.0|  
2|le nid des marsupilamis|franquin|Un reportage incroyable|6.0|2011-02-03  
3|la déesse|moebius|une aventure extraordinaire|7.0|2011-02-03 18:36:25
```

- **Requête de visualisation d'une table formatée en sortie COMME une insertion de valeur** :

```
sqlite> .mode insert bandedessinée
sqlite> select * from bandedessinée;
```

Affichera :



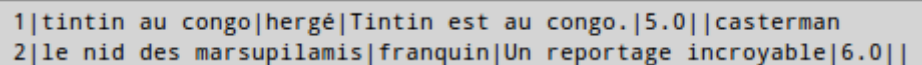
```
INSERT INTO 'bandedessinée' VALUES(1,'tintin au congo','herge','Tintin est au congo.',5.0,NULL);
INSERT INTO 'bandedessinée' VALUES(2,'le nid des marsupilamis','franquin','Un reportage incroyable',6.0,'2011-02-03');
INSERT INTO 'bandedessinée' VALUES(3,'la déesse','moebius','une aventure extraordinaire',7.0,'2011-02-03');
```

Quelques exemples de requêtes :

- **Limiter une requête par nombre d'éléments :**

```
sqlite> select * from bandedessinée limit 2;
```

affiche :

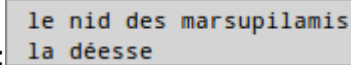


```
1|tintin au congo|herge|Tintin est au congo.|5.0||casterman
2|le nid des marsupilamis|franquin|Un reportage incroyable|6.0||
```

- **Sélectionner les titres de la table bandedessinée enregistrés depuis février :**

```
sqlite> select titre from bandedessinée where
strftime('%m', date_creation)='02';
```

affiche :



```
le nid des marsupilamis
la déesse
```

7. Effacer une valeur dans la table :

```
sqlite> DELETE FROM "bandedessinée" WHERE id = 3;
```

8. Ajouter une colonne à la table :

```
sqlite> ALTER TABLE "bandedessinée" add column "éditeur";
```

9. Mettre à jour une valeur de la table :

```
sqlite> UPDATE "bandedessinée" SET éditeur ='casterman'
WHERE id = 1;
```

10. Modifier le nom d'une table :

```
sqlite> alter table 'bandedessinée' rename to 'bd';
```

Autres commandes

- **.backup ?DB? FILE** : Sauvegarde DB (par défaut "main") vers FILE
- **.bail ON|[OFF]** : Stop après une erreur

- **.echo ON|OFF** : Bascule la commande d'écho ON/OFF
- **.explain ?ON|OFF?** : Bascule le mode de sortie approprié pour EXPLIQUER on or off
Sans argument, on
- **.load FILE ?ENTRY?** : Charger une bibliothèque d'extension
- **.log FILE|off** : Activer ou désactiver la journalisation. FILE peut être stderr/stdout
- **.nullvalue STRING** : Imprime STRING à la place des valeurs NULL
- **.prompt MAIN CONTINUE** : Remplace les prompts standards
- **.read FILENAME** : Exécuter SQL dans FILENAME
- **.restore ?DB? FILE** : Restaurer le contenu de la DB (par défaut "main") à partir de FILE
- **.schema ?TABLE?** : Afficher les déclarations CREATE. Si TABLE est spécifié, ne montrer que les tables correspondant au motif TABLE
- **.stats ON|OFF** : Active ou désactive les stats
- **.timeout MS** : Essayer d'ouvrir les tables verrouillées durant MS millisecondes
- **.trace FILE|off** : afficher chaque instruction SQL au moment de son exécution
- **.vfsname ?AUX?** : afficher le nom de la pile VFS
- **.timer ON|OFF** : activer ou désactiver la minuterie du CPU

Désinstallation

Voir aussi

- (fr) <http://Article>

Basé sur « [Article](#) » par Auteur.

From:

<https://nfrappe.fr/doc-0/> - Documentation du Dr Nicolas Frappé

Permanent link:

<https://nfrappe.fr/doc-0/doku.php?id=logiciel:sql:sqlite:start>

Last update: **2022/08/13 21:57**

