

Utilisation de sqlite3

Sqlite3 est l'utilitaire en ligne de commande fourni avec la bibliothèque du système de gestion de base de données (SGBD) SQLite. Il permet d'interagir sur la ligne de commande (en tapant du texte) avec un fichier contenant une base de données SQLite (ou une base de données en mémoire). Sqlite3 existe sur toutes les plateformes et se télécharge sur la page <http://www.sqlite.org/download.html> (sous le nom `sqlite-shell-XX.zip`).

Utilisation sous Windows

Le programme n'a pas besoin d'être installé au sens classique sous Windows, il suffit de décompresser le fichier téléchargé ce qui crée l'exécutable `sqlite3.exe`. Placez le dans un répertoire facile à retrouver, par exemple dans vos documents (`C:\Users\nom-utilisateur\Documents`).

Pour être capable d'ouvrir un fichier de base de données SQLite avec `sqlite3`, il faut d'abord lancer une ligne de commande Windows. Il y a deux choix, `cmd.exe` et `powershell`, le second étant plus agréable à utiliser, mais disponible seulement à partir de Windows 7. Lancez `powershell`.

Il est important de comprendre à ce stade que la ligne de commande est toujours "dans" un répertoire de votre ordinateur (le nom est toujours indiqué avant le `>`). Les programmes que vous exécutez et les fichiers auxquels vous accédez peuvent être désignés de deux façons: de façon relative à la position de votre ligne de commande ou de façon absolue. De façon relative, le chemin se compose à partir du répertoire dans lequel la ligne de commande se trouve, par exemple `Desktop` ou `Documents` si la ligne de commande est dans `C:\Users\nom-utilisateur\` (pour référencer un répertoire parent, par exemple dans `C:\Users\` si la ligne de commande est dans `C:\Users\nom-utilisateur\` on utilise `..`). Sinon, le chemin est absolu à partir d'un des disques de votre ordinateurs, comme `C:\Users\nom-utilisateur\Documents`.

Pour exécuter `sqlite3` sur un fichier `tmp.sqlite` dans le même répertoire, par exemple dans `C:\Users\nom-utilisateur\Documents` (le répertoire "Mes Documents"), suivez les étapes suivantes (en général, `powershell` démarre dans votre répertoire personnel, soit `C:\Users\nom-utilisateur`):

```
> cd Documents
> sqlite3.exe tmp.sqlite
```

Alternativement, un exemple de chemin relatif est

```
> Documents\sqlite3.exe Document\tmp.sqlite et un exemple de chemin absolu est
> C:\Users\nom-utilisateur\Documents\sqlite3.exe C:\Users\nom-utilisateur\Documents\tmp.sqlite. Ces commandes sont équivalentes et créeront le fichier tmp.sqlite s'il n'existe pas.
```

La commande `cd nom-répertoire` où `nom-répertoire` peut être un chemin relatif ou absolu permet de changer le répertoire de la ligne de commande. Pour éviter de taper les noms complets des répertoires, vous pouvez utiliser la complétion automatique en utilisant la touche tabulation (en tapant par exemple les premières lettres du répertoire après `cd`, puis tabulation, la ligne de commande

complètera le nom du répertoire s'il n'y a pas d'ambiguïté ou proposera l'une après l'autre toutes les correspondances possible).

Utilisation de sqlite3

Sqlite3 accepte toutes les commandes SQL (variante SQLite). Il ne faut pas oublier de les terminer par `;`. De plus, il existe un ensemble de commandes de l'utilitaire qui commence toutes par `.`, telles que `.help` qui liste ces commandes. La commande `.table` liste les tables existantes dans la base de données, la commande `.schema nom-table` retourne la commande SQL qui a créé la table.

En particulier, ces données permettent d'importer et d'exporter des données. Pour importer les données d'un fichier csv (bien formaté, avec un nombre de colonnes constant):

```
sqlite> .separator ";"
sqlite> .import fichier.csv nom-table
```

La première ligne indique le séparateur utilisé dans le fichier csv, la seconde importe les données du fichier `fichier.csv` dans la `nom-table` créée au préalable et dont le schéma est compatible avec les données dans le fichier. Ces commandes fonctionnent avec les fichiers de carte à puce de la STM. Il est possible de faire les opérations d'importation ci-dessous sans spécifier de fichier sqlite en lançant sqlite3: dans ce cas, la base de données sera créée en mémoire, ce qui vous causera des problèmes si votre ordinateur n'a pas assez de mémoire pour charger toutes les données (plus de 4 Go).

Les résultats des requêtes SQL sont affichées sur l'écran de la ligne de commande (ajouter `limit N` à la fin de vos requêtes SQL pour limiter le retour à N lignes au maximum et obtenir des résultats plus rapides lorsque vous testez vos requêtes). Pour exporter les données, il faut indiquer à sqlite3 d'écrire le résultats de vos requêtes dans un fichier plutôt que sur la ligne de commande:

```
sqlite> .output resultat.csv
sqlite> .mode tabs
sqlite> requête sql;
```

La première ligne indique le nom du fichier dans lequel écrire (`resultat.csv`) et la seconde le format du fichier (le mode `csv` existe, mais le mode `tabs` retourne un fichier texte où les cases du tableau sont séparées par des tabulations, format qui peut ensuite être lu par Tanagra).

Enfin, il n'est pas nécessaire de sauver votre base de données: le fichier de la base est mis à jour après chaque requête effectuée. Quittez sqlite3 par `.exit`.

Fonctions pour les données de temps et heure

Il existe plusieurs fonctions dans SQLite pour générer ou extraire des informations sur les champs de type `timestamp` (ou `date` ou `time`), en particulier `strftime`. La fonction permet d'extraire par exemple

- l'heure: `strftime("%H", date)`
- le jour (un caractère "0" pour lundi, "1" pour mardi, etc.): `strftime("%w", date)`

De même que les champs `timestamp` sont en fait enregistrés comme du texte, `strftime` retourne du texte. Supposons que notre table `mesures` à un champ `horaire`, voici quelques exemples de requêtes :

```
sqlite> SELECT * FROM mesures WHERE strftime("%H", horaire)="8"
```

La requête retourne tous les champs de la table pour lesquels le champ `horaire` est entre 8h00 et 8h59.

```
sqlite> SELECT * FROM mesures WHERE strftime("%w", horaire)>=5
```

La requête retourne tous les champs de la table pour lesquels le champ `horaire` correspond 0 un jour de fin de semaine.

La référence sur ces fonctions se trouve à la page http://www.sqlite.org/lang_datefunc.html.

Recommandation

Si une table contient beaucoup de données, je vous conseille de toujours travailler avec un plus petit ensemble de données (en sélectionnant un sous-ensemble dans une table temporaire) avant de lancer les calculs sur l'ensemble.