Premiers pas avec SQLfast

Niveau Expert : Apprentissage et utilisation du langage SQLfast

1 décembre 2020

Objectif

Ce tutoriel montre comment utiliser l'outil SQLfast pour l'apprentissage et la programmation dans le langage SQLfast. Il décrit également certaines fonctions de base de l'interface SQLfast.

SQLfast offre deux niveaux d'utilisation :

- **Basic** - Apprentissage et utilisation du langage SQL
- **Expert** - Apprentissage et utilisation du langage SQLfast

... qu'il est recommandé de parcourir dans cet ordre.

Le présent document est consacré à l'utilisation de SQLfast au niveau **Expert**. Il est également disponible sous la forme du document d'aide *Getting started* du niveau **Expert**, accessible via le menu **Help** de la fenêtre principale

Différences avec le niveau Basic

Alors que le niveau **Basic**, consacré à la familiarisation avec les concepts des bases de données et à l'apprentissage et à l'utilisation légère du langage SQL, offrait un jeu limité de fonctions, le niveau **Expert** permet un accès beaucoup plus large aux fonctions du moteur SQLfast.

En particulier, il favorise le développement et l'utilisation de scripts, c'est à dire de programmes et procédures rédigés dans le langage SQLfast. Un script SQLfast peut être constitué d'une ou plusieurs requêtes SQL ou, plus généralement, d'une suite d'instructions SQLfast.

Le niveau **Expert** se différencie du précédent principalement par les aspects suivants :

- La présence de menus offrant une plus large gamme de fonctions.
- La gestion des scripts.
- Trois modes d'exécution d'un script.
- L'accès à toutes les fonctions du langage SQLfast, notamment les instructions de création, d'ouverture et de fermeture de bases de données.
- Le contrôle personnalisé des transactions par les scripts.
- Les fenêtres de schéma et de données sont indépendantes de la base de données ouverte par un script.

En contrepartie, certains fonctionalités du niveau **Basic** ne sont plus disponibles :

- Les boutons d'accès à une base de données (**Create DB**, **Open DB**, **Restore DB**, **Close DB**) sont remplacés par des instructions SQLfast ou des fonctions du menu **Database**.
- La notion de *base de données courante* n'existe plus. Tout script peut ouvrir et fermer une ou plusieurs bases de données (une à la fois).
- Les fenêtres de données ne sont plus mises à jour automatiquement suite à l'exécution de requêtes SQL de modification mais doivent être rafraîchies explicitement.

Avertissement

Tant le logiciel que le(s) langage(s) SQLfast sont en évolution constante. Il est donc possible que certaines descriptions et certains documents soient relatifs à des versions plus anciennes. Ils seront actualisés dès que possible. Il est recommandé de vérifier régulièrement l'état d'avancement de ces fonctions et de leur documentation.

Le logiciel référencé dans ce tutoriel est la version 64 bits sous Windows, exécutable sous toutes les versions standard de Windows disponibles depuis janvier 2015. Elle fonctionne également sous Linux via Wine. Dans ce dernier cas, quelques ajustements dans le fichier d'initialisation "**SQLfast.ini**" peuvent s'avérer nécessaires, dont certains *paths* par défaut et le codage des fichiers de texte (par exemple UTF-8 au lieu du standard Windows cp1252).

Il va sans dire que les conséquences de l'utilisation des outils, des langages et des documents relatifs à SQLfast sont de la totale responsabilité des utilisateurs. L'utilisation de ces produits n'engage en aucune manière la responsabilité des auteurs de ces derniers ni celle de l'université de Namur.

Contact : jean-luc.hainaut@unamur.be

Table des matières

1. Installation

Le logiciel SQLfast aura été installé selon les instructions du document SQLfast_Installation.pdf disponible à l'adresse https://projects.info.unamur.be/~dbm/mediawiki/index.php/DUNOD2015_SQLfast.

La figure 1 montre l'état du répertoire SQLfast, installé à la racine du disque **D:**. Il contient divers répertoires et fichiers dont seuls nous intéressent pour débuter :

– **SQLfast.exe** : le logiciel SQLfast. Pour simplifier son utilisation, il peut être utile d'en créer un raccourci sur le bureau.

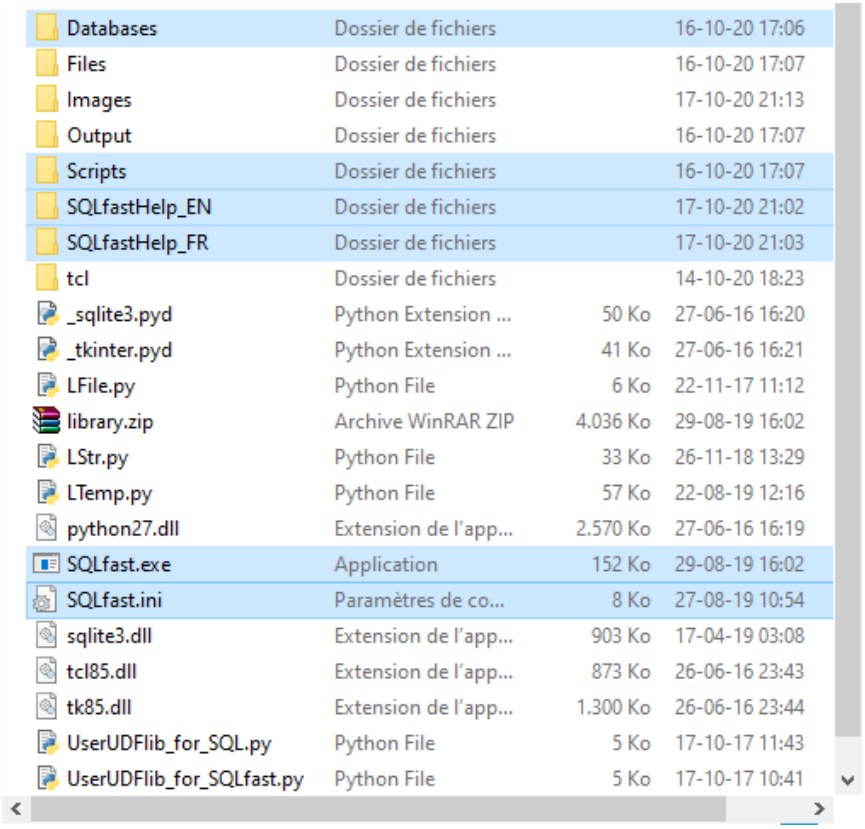

Figure 1 - Le contenu du répertoire SQLfast

– **Databases** : le répertoire par défaut dans lequel seront stockées les bases de données que nous créerons et utiliserons. Deux bases de données sont déjà disponibles et prêtes à l'emploi : CLICOM.db et ORDERS.db (anglais).
– **Scripts** : le répertoire par défaut dans lequel seront stockés les scripts SQLfast.
– **Ouput** et **Files** : les répertoires dans lesquels seront stockés par défaut les fichiers produits et utilisés par les scripts SQLfast.

- **SQLfastHelp_FR** : le répertoire par défaut dans lequel sont stockées les versions françaises des fichiers d'aide SQLfast (extension "**.help**") ainsi que des tutoriels (extension "**.tuto**").
- **SQLfast.ini** : le fichier d'initialisation des paramètres de SQLfast. Il contient les valeurs par défaut de ces paramètres. Ces valeurs peuvent être modifiées selon les besoins, soit dans ce fichier, soit, dynamiquement, via certaines instructions du lange SQLfast.

Les autres fichiers et répertoires sont utiles ou nécessaires au fonctionnement de SQLfast mais ne nous intéressent pas pour l'instant.

2. Démarrage de SQLfast

Double-cliquer sur **SQLfast.exe** dans le répertoire **SQLfast** ou sur son raccourci. Les deux fenêtres principales de l'interface graphique de SQLfast apparaissent (figure 2) :

- **fenêtre principale** : C'est à partir de cette fenêtre que nous allons effectuer les principales manipulations. Elle comporte aussi un espace dans lequel nous allons écrire les requêtes SQL et les scripts SQLfast à exécuter, la *zone de script*.
- **fenêtre de sortie** : (ou SQLfast output Window) C'est dans cette fenêtre que vont s'afficher les résultats de l'exécution des scripts SQLfast. On y trouvera aussi divers messages d'information sur les conditions d'exécution des requêtes.

On vérifie que l'interface est en mode "Expert" (bouton en haut à droite).

3. La fenêtre principale

De haut en bas : la *barre de menu*, l'identification du *répertoire de script* et le *nom du script courant*, les *boutons de fonction* et la *zone de script*.

Le menu offre principalement des fonctions de gestion globale des sessions, des scripts et des bases de données. Il propose également une collection plus complète de documents d'aide. A droite de la barre de menu, les boutons de *sélection du niveau*.

Deux champs d'identification des scripts :

- **Script directory** : répertoire par défaut des scripts; plus tard, répertoire du script courant;
- **Script** : nom local du script courant.

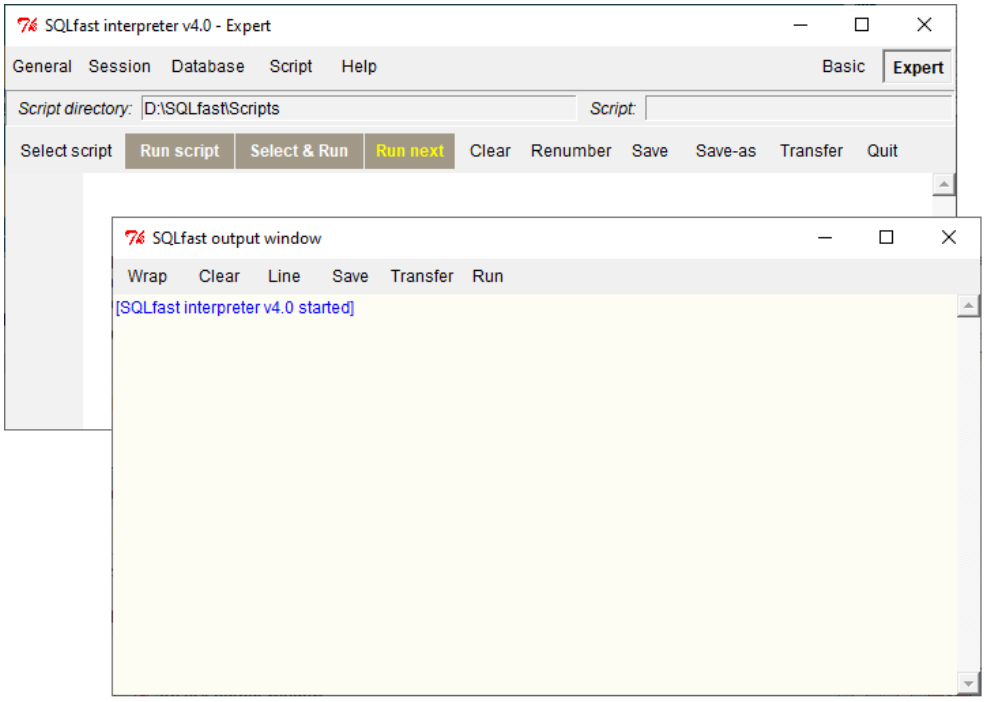

Figure 2 - Les fenêtres de l'interface SQLfast au niveau Expert

Les boutons permettent un accès rapide à des fonctions courantes :

- **Select script** : sélectionner un script et le charger dans la zone de script; ce script devient le script courant;
- **Run script** : exécuter le script courant présent dans la zone de script;
- **Select & Run** : sélectionner un script et l'exécuter; le script courant n'est pas modifié;
- **Run next** : exécution incrémentale des instructions de la zone de script; équivalent au bouton "**Run**" du niveau **Basic**;
- **Clear** : effacer le script courant;
- **Renumber** : renuméroter les lignes du script courant;
- **Save** : recopier sur disque le script courant;
- **Save as** : sauver le script courant dans un nouveau fichier **.sql**;
- **Transfer** : copier le script courant dans la fenêtre de sortie;
- **Quit**

Les fonctions de la fenêtre principale sont décrites de manière plus complète dans le document d'aide SQLfast environment, disponible via le menu **Help** : **Help > SQLfast references > SQLfast environment**.

4. La fenêtre de sortie

Les fonctions de la fenêtre de sortie sont identiques à celles du niveau **Basic**. Les boutons **Transfer** et **Run** jouent le même rôle que dans la fenêtre d'aide : copier dans l'espace de script ou exécuter les lignes sélectionnées. Cette action prend son sens lorsque le script exécuté a généré des requêtes SQL ou des scripts SQLfast.

5. Interroger une base de données

La distribution standard inclut déjà, dans le répertoires **Databases**, une base de données, **CLICOM.db**, contenant quelques données.

Pour la consulter, nous devons d'abord l'ouvrir à l'aide d'une commande SQLfast. Nous écrivons donc dans la fenêtre principale l'instruction suivante :

```
openDB CLICOM.db;
```

Vient ensuite la requête de consultation :

```
select * from CLIENT where LOCALITE = 'Poitiers';
```

Nous fermons enfin la base de données :

```
closeDB;
```

La fenêtre principale contient désormais un script SQLfast complet (figure 3) :

```
openDB CLICOM.db;
  select * from CLIENT where LOCALITE = 'Poitiers';
closeDB;
```

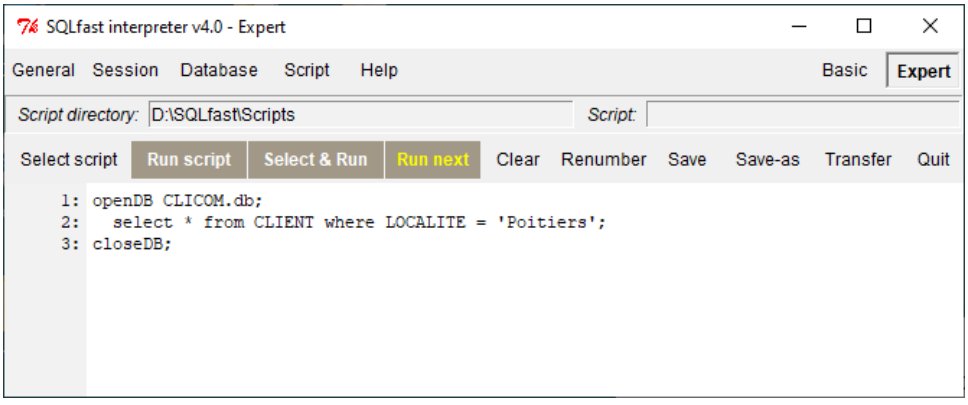

Figure 3 - Script SQLfast de consultation de la table CLIENT de la base de données CLICOM.db

On exécute ce petit script en cliquant sur le bouton **Run script**. Le résultat apparaît dans la fenêtre de sortie (figure 4).

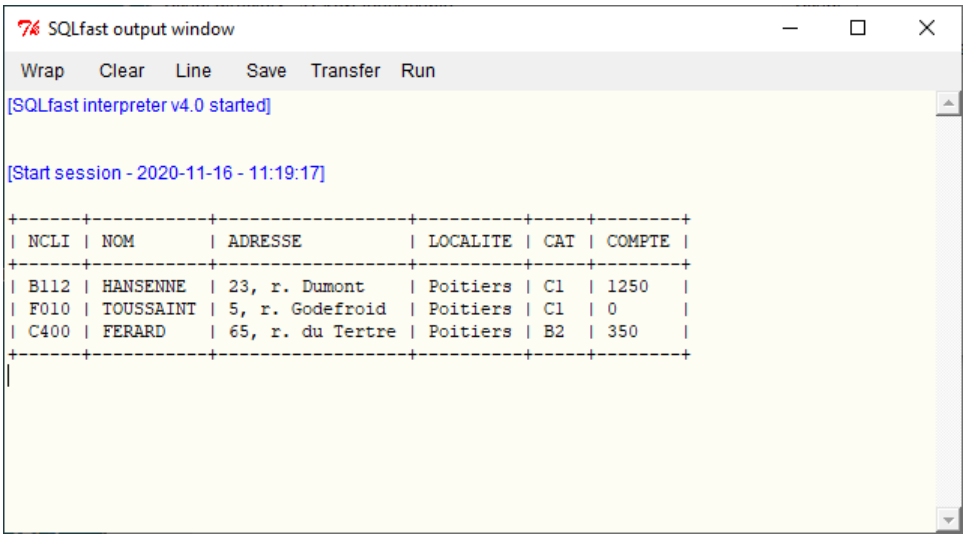

Figure 4 - Résultat de la consultation de la table CLIENT

Commentaires

Le contenu actuel de la fenêtre principale s'appelle le *script local*. Il peut être introduit manuellement, comme nous l'avons fait, chargé à partir d'un fichier de script qu'on sélectionne à l'aide du bouton **Select script** (il devient le script courant, existant sous la forme d'un fichier), ou encore transféré à partir d'un tutoriel.

Le bouton **Run script** commande l'exécution du script local. Nous explorerons plus tard les deux autres boutons d'exécution : **Select & Run** et **Run next**.

Le bouton **Renum** assigne des numéros aux lignes que nous avons introduites. Cette information sera utile pour identifier les erreurs qui seront détectées lors de l'exécution d'un script.

6. Sauver le script local (bouton "Save-as")

Nous allons sauver le petit script que nous venons de rédiger sous la forme d'un fichier de script, afin de le réutiliser ultérieurement.

Nous cliquons sur le bouton **Save-as**. Une boîte de sauvetage apparaît, montrant le contenu du répertoire **Scripts** (figure 5). On nomme le fichier **Consulter-CLIENT**. Un nouveau fichier **Consulter-CLIENT.sql** apparaît dans ce répertoire.

7. Sélection d'un fichier de script (bouton "Select script")

Commençons par effacer le script local en cliquant sur le bouton **Clear**. La fenêtre est désormais vide.
En cliquant sur le bouton **Select script**, nous ouvrons une boîte de sélection de fichier, qui, par défaut, montre le contenu du répertoire Scripts. Nous sélectionnons le fichier Consulter-CLIENT.sql dernièrement créé.
Le contenu de ce fichier apparaît dans la zone de script de la fenêtre principale et devient le script local. Il peut alors être exécuté (**Run script**), modifié, sauvé (**Save** ou **Save as**) et supprimé (**Clear**). Cette suppression ne concerne évidemment pas le fichier de script d'origine.

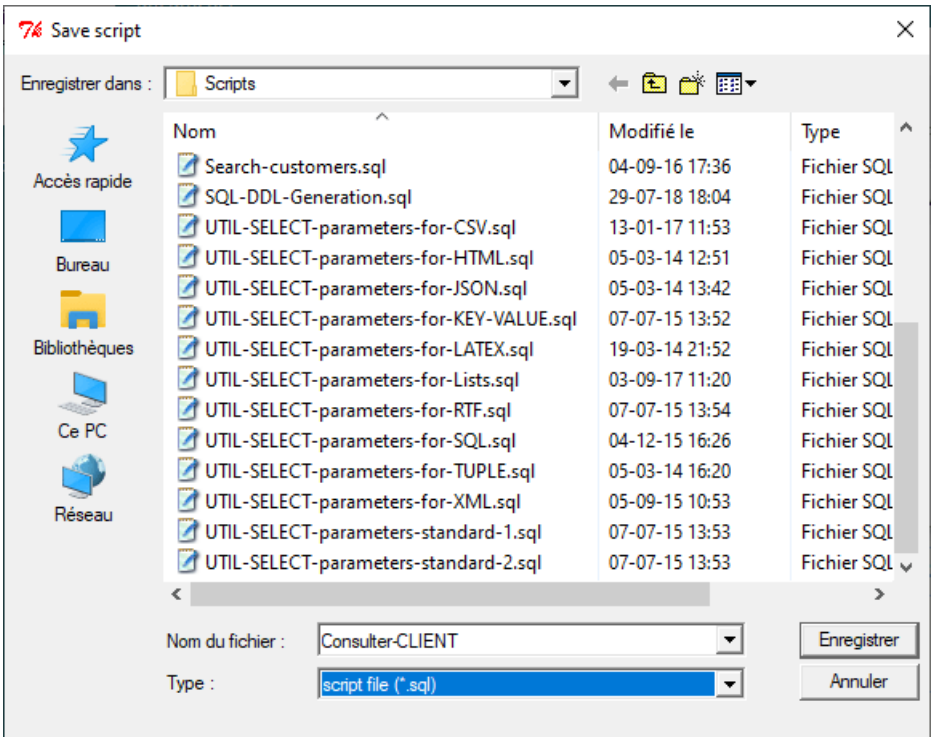

Figure 5 - Création d'un fichier de script à partir du script local.

8. Sélection et exécution directe d'un script (bouton "Select & Run")

Ce mode d'exécution permet, en une seule opération, de sélectionner un fichier de script et de l'exécuter. Il s'obtient par le bouton **Select & Run**. Chose importante : cette opération ne modifie pas le script local.

Pour tester cette fonction, chargeons notre script Consulter-Client.sql par le bouton **Select script**. Si nous l'exécutons, nous obtenons le résultat de la figure 4.
A présent, exécutons, via le bouton **Select & Run**, un des scripts "UTIL-SELECT-parameters-...sql" qui apparaissent dans la figure 5. Choisissons par exemple

```
UTIL-SELECT-parameters-for-CSV.sql
```

Rien ne semble s'être produit. Vidons la fenêtre de sortie (bouton **Clear**) puis exécutons à nouveau le script local (**Run script**). Nous obtenons dans la fenêtre de sortie un résultat tout différent :

```
NCLI;NOM;ADRESSE;LOCALITE;CAT;COMPTE
"B112";"HANSENNE";"23, r. Dumont";"Poitiers";"C1";"1250"
"F010";"TOUSSAINT";"5, r. Godefroid";"Poitiers";"C1";"0"
"C400";"FERARD";"65, r. du Tertre";"Poitiers";"B2";"350"
```

On aura peut-être reconnu un des formats d'échange standard **csv** (*comma-separated values*) familier des utilisateurs de feuilles de calcul (Excel ou Calc)..

Sauvons ce résultat (bouton **Save** de la fenêtre de sortie) dans un fichier que nous nommons Poitiers.csv et double-cliquons sur ce dernier : nous obtenons un tableau Excel contenant les données produites par la requête !

Si le résultat n'est pas satisfaisant, on modifiera (dans le fichier SQLfast.ini ou directement dans le script courant) certains paramètres qui spécifient la présence ou l'absence de la ligne qui donne le nom des colonnes (csv-header) ou le séparateur de valeurs : virgule, point-virgule ou autre (csv-separator).

Il est possible d'automatiser l'ensemble de cette séquence mais ceci est une autre histoire, qui sera abordée dans le tutoriel **Help > Survival guide**.

Les autres scripts UTIL-SELECT-parameters... vont nous donner toute une gamme de format variés : HTML, JSON, LateX, RTF, XML, etc., qu'on peut explorer sans risque !

On n'oubliera pas, en terminant cette expérience, de remettre les lieux dans l'état où nous les avons trouvés, en exécutant le script du format standard : UTIL-SELECT-parameters-standard-1.sql.

9. Gestion des erreurs de script

Considérons le script ci-dessous, qu'on introduit dans la zone de script (plus simplement, on sélectionne les instructions et on clique sur le bouton "Transfer") :

```
openDB CLICOM.db;
  ask Localite;
  display Clients habitant à $Localite$;
  select NCLI,NOM,ADRESSE from CLIENT
  where  LOCALITE = '$Localite$';
closeDB;
```

Il a pour objectif d'afficher les données des clients d'une localité spécifiée par l'utilisateur via une boîte de dialogue créée par l'instruction **ask**.

Dans cette boîte, l'utilisateur est invité introduire le nom d'une localité. La valeur choisie est rangée dans la variable **Localite**.

Ce choix fait, le script imprime d'abord un titre : 'Clients habitant à ' suivi de la valeur de la variable **Localite**, puis affiche les données des clients de cette localité.

Il n'est pas nécessaire à ce stade de comprendre le détail des instructions utilisées. Ces instructions seront décrites dans le tutoriel **Survival guide** qu'on trouvera dans le menu **Help**.

Cependant, ce script comporte une erreur : le verbe **display** n'existe pas en SQLfast. Le rédacteur a certainement voulu utiliser le terme **write**, ou encore **print**, qui sont les seuls légaux.

Lorsqu'on demande l'exécution du script par le bouton **Run script**, l'erreur est détectée, ce qui conduit à trois réactions, que nous avons d'ailleurs observées au niveau **Basic** :

– apparition d'une boîte de message décrivant l'erreur,
– recopie de cette description dans la fenêtre de sortie,
– coloration en rouge de l'instruction fautive (figure 6).

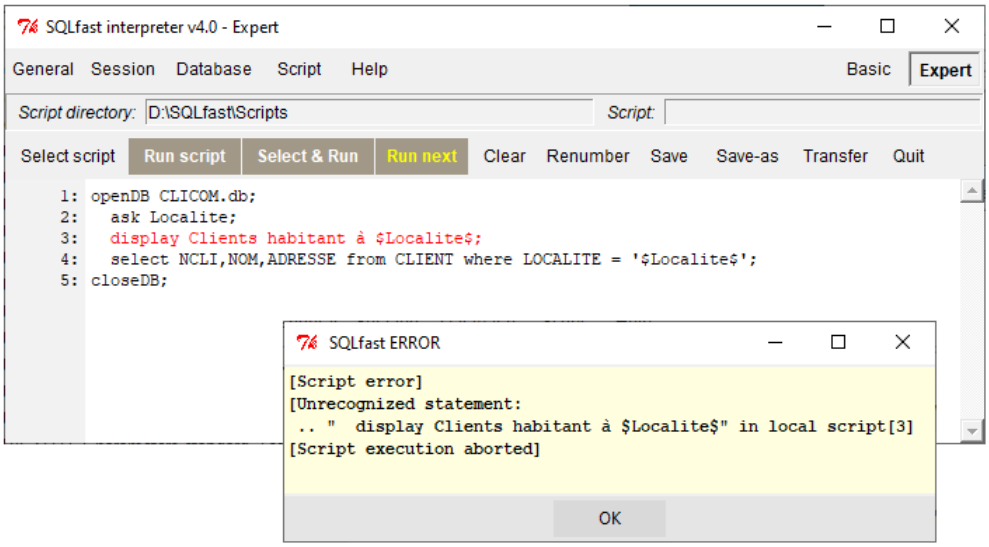

Figure 6 - Erreur détectée lors de l'exécution d'un script local

Le message d'erreur contient cinq informations :

1. la nature de l'erreur : "Unrecognized statement",
2. l'instruction erronée : "display Clients habitant à '$Localite$'",
3. le script où l'erreur est apparue : "local script",
4. le numéro de ligne de l'erreur : "[3]",
5. la réaction : "Script execution aborted".

On corrigera l'erreur et on réexécutera le script pour vérifier sa validité.

10. Script principal et scripts secondaires

Un script peut demander l'exécution d'un autre script. Introduisons (on transférons) dans la zone de script les instructions suivantes :

```
-- Script principal
openDB CLICOM.db;
  execSQL Demander-localite.sql;
  execSQL Rechercher-clients.sql;
closeDB;
```

La première ligne n'est pas une instruction, mais un simple commentaire qui sera ignoré lors de l'exécution.

Ce petit script local ne peut cependant être exécuté. En effet, les deux instructions **execSQL** demandent l'exécution de scripts supposés exister dans le répertoire des scripts, ce qui n'est en principe pas le cas.[1]

Sauvons donc ce script, via le bouton **Save-as**, sous le nom **Principal.sql** et rédigeons les deux scripts manquants. On les qualifiera de *secondaires*, par opposition au premier, qu'on qualifiera de *principal*.

Vidons la zone de script (bouton **Clear**) et introduisons la suite d'instructions suivante :

```
-- Script secondaire: saisir un nom de localité
ask Localite = Nom de la localité[!select distinct LOCALITE
                                  from   CLIENT];
```

... suite que nous sauvons sous le nom **Demander-localite.sql**.

Cette unique instruction (la première ligne est un commentaire) ouvre une boîte de saisie d'un nom de localité, à sélectionner parmi les valeurs extraites de la base de données. Nous reparlerons de cette instruction dans le tutoriel **Survival guide**.

Ensuite, créons un second script secondaire que nous sauvons sous le nom **Rechercher-clients.sql** et qui est composé des instructions suivantes :

```
-- Script secondaire: afficher les clients d'une localité
execSQL Imprimer-titre.sql;
select NCLI,NOM,ADRESSE
from   CLIENT where LOCALITE = '$Localite$';
```

Ce script fait appel aux services d'un troisième script secondaire, nommé **Imprimer-titre.sql**, que nous créons avec le contenu suivant :

```
-- Script secondaire: afficher un titre
write Clients habitant à $Localite$;
```

1. ... si ces scripts sont déjà présents dans le répertoire Scripts, jouons le jeu honnêtement et supprimons-les d'abord.

Pour nous résumer, nous avons créé quatre scripts SQLfast, qui vont s'exécuter selon la hiérarchie suivante :

```
Principal.sql
     |
     +--> Demander-localite.sql
     |
     +--> Rechercher-clients.sql
               |
               +--> Imprimer-titre.sql
```

Nous pouvons à présent exécuter le script principal, par exemple via le bouton **Select & run**. Le résultat est conforme à notre attente.

11. Erreur dans un script secondaire

A présent, introduisons une erreur dans le script **Imprimer-titre.sql** en remplaçant **`write`** par **`display`**, instruction inconnue en SQLfast :

```
-- Script secondaire: afficher un titre
display Clients habitant à $Localite$;
```

En relançant l'exécution du script principal Principal.sql, l'erreur est détectée et un message identifiant et localisant celle-ci apparaît (figure 7).

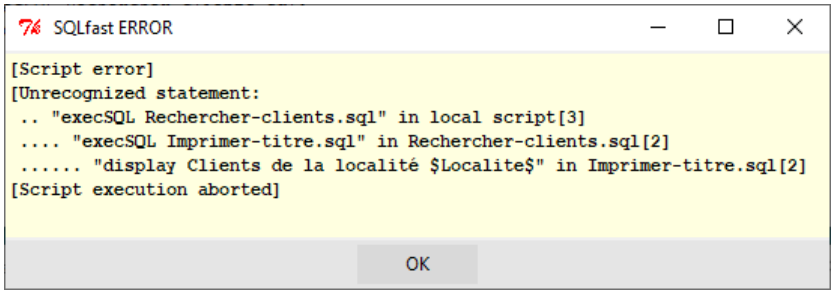

Figure 7 - Description en cascade d'une erreur détectée lors de l'exécution d'un script secondaire

L'identification et la localisation de l'erreur est un peu plus complexe que lorsque celle-ci se produit dans le script local. La description de l'erreur nous informe que (en lisant les lignes *de bas en haut*) :

- l'erreur s'est produite dans le script Imprimer-titre.sql, à la ligne [2], dans l'instruction `print Clients ...`
- ce script avait été appelé par le script Rechercher-clients.sql à la ligne [2], par l'instruction `execSQL Imprimer-titre.sql`
- ce dernier script avait lui-même été appelé par le script Principal.sql à la ligne [4], par l'instruction `execSQL Rechercher-clients.sql`.

Commentaire

Si une erreur se produit avant que les scripts aient eu l'occasion de fermer la base de données, il se peut que celle-ci reste ouverte. Cette éventualité dépend du mode de fermeture automatique défini par le paramètre **autoCloseDB**. Il sera alors prudent d'en forcer la fermeture via le menu : **Database > Close database** pour éviter les erreurs ultérieures du type *DB cannot be opened (a DB is already opened)*.

12. Exécution pas à pas d'un script (bouton "Run next")

Cette fonction est identique à celle du bouton **Run** du niveau **Basic**.

Rappelons donc que, contrairement aux modes **Run script** et **Select & Run**, lorsqu'une série d'instructions a été exécutée en mode **Run next**, en une ou plusieurs fois, la base de données qu'on a ouverte reste ouverte, du moins tant qu'on ne l'a pas explicitement fermée par une instruction `closeDB`. Ceci permet d'ouvrir une fois pour toutes la base de données, puis d'expérimenter différentes requêtes. Lorsque le travail est terminé, on ferme la base de données par une instruction `closeDB` ou via le menu : **Database > Close database**.

Commentaires

– Une conséquence importante de ce principe est que, si après avoir travaillé en mode pas à pas (**Run next**) sans avoir fermé explicitement la base de données, on désire exécuter un script qui lui-même ouvre une base de données (**Select script** puis **Run script** ou **Select & Run**), une erreur se produira. Nous devrons alors fermer la base de données avant de relancer l'exécution du script.
– La séquence des instructions qui apparaît dans la fenêtre principale peut aussi être exécutée dans le mode standard **Run script**, et ce, indépendamment de la couleur des lignes. On peut ainsi mettre au point et exécuter pas à pas un script, puis en redemander l'exécution complète.
– Il existe cependant une petite mais subtile différence avec le mode **Run** du niveau **Basic**. Alors que dans ce dernier, le mode de commit est d'office auto-commit, quel que soit le mode défini dans le fichier d'initialisation **SQLfast.ini**, dans le présent niveau, c'est le mode d'initialisation qui reste d'application.
– Quitter SQLfast (bouton **Quit**) entraîne la fermeture automatique de la base de données qui serait encore ouverte.

13. Un dernier bouton de la fenêtre principale : "Transfer"

Par ce bouton, on copie le contenu de la zone de script (en principe le script local) dans la fenêtre de sortie.

Quel intérêt ? Par exemple préparer un brouillon de document dans lequel on alterne code d'un script et résultat de son exécution, comme illustré à la figure 8.

14. Comment sont gérées les erreurs ?

Nous avons vu que, lorsqu'une erreur se produit, un message apparaît et l'exécution du script se termine. L'instruction fautive, comme elle mérite, est colorée en rouge.

Attention cependant, ce comportement, dit *standard*, peut être modifié par le script lui-même, qui peut demander que SQLfast ignore les erreurs et poursuive l'exécution du script (instruction **`onError`** `continue`). Dans ce cas, il est de la responsabilité du script de gérer correctement ses propres erreurs, en traitant le contenu des variables systèmes **Error**, **SQLdiag**, **FILEdiag**, **WEBdiag** et **EXTENDEDdiag**.

On consultera à ce sujet le tutoriel **Help > Survival guide**.

La fenêtre de sortie dispose également d'un code de couleur, tout comme la fenêtre principale. Selon la nature des lignes affichées, celles-ci sont écrites en différentes couleurs :

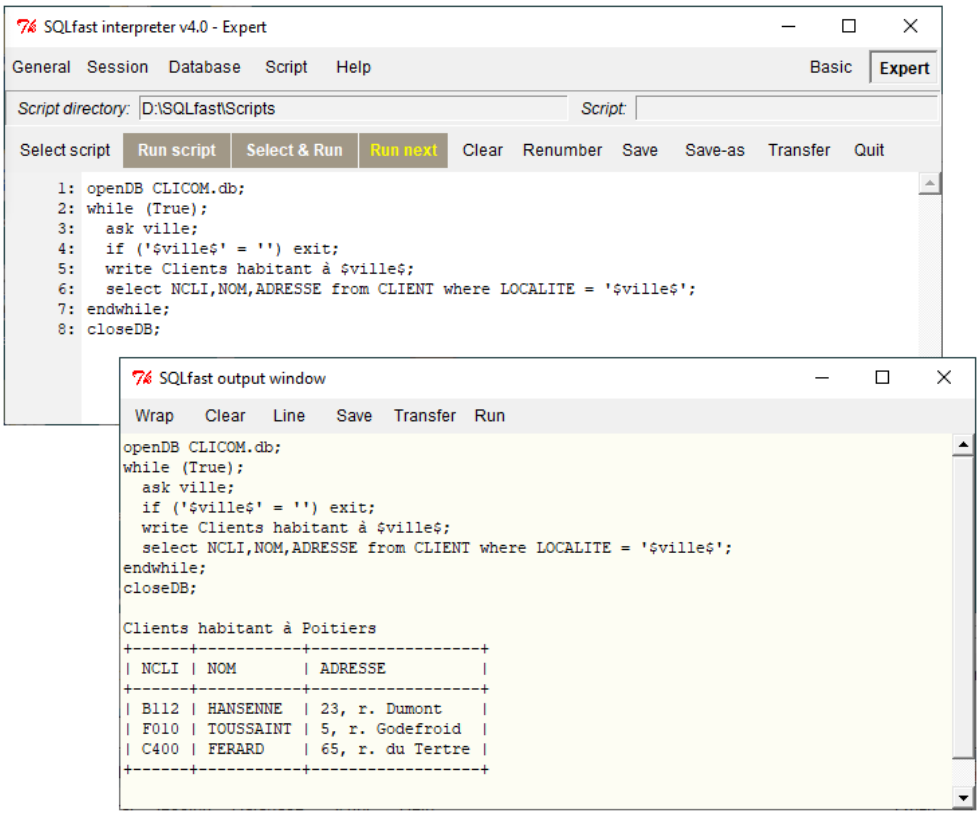

Figure 8 - Transfert du contenu de la fenêtre principale vers la fenêtre de sortie.

- **en noir**, le résultat de l'exécution des requêtes et des scripts.
- **en bleu**, les messages du journal des opérations.
- **en rouge**, les messages décrivant les erreurs critiques survenant lors de l'exécution d'un script SQLfast. L'exécution du script est stoppée.
- **en vert**, les messages d'avertissement n'interrompant pas l'exécution d'un script SQLfast. *Exemples* : tentative de fermeture d'une base de données déjà fermée, commande inconnue dans le fichier source d'un tutoriel, image manquante dans un tutoriel, etc.

15. La fenêtre du schéma

Tout comme dans le niveau **Basic**, il est possible d'ouvrir une fenêtre montrant le schéma d'une base de données sous forme d'un résumé (*Summary*) ou du code SQL-DDL complet (*Full schema*).

Comme il n'existe plus dans le niveau **Expert** de *base de données courante*, la base de données sera sélectionnée via une boîte de sélection de fichier. La fenêtre de schéma est ouverte via le menu : **Database > Show DB schema**.

Si le code SQL-DDL complet est absent, il est recréé automatiquement. Il peut aussi être recréé manuellement via le menu **Database** > **Generate DDL schema**".

16. Les fenêtres de données

Il est aussi possible d'ouvrir des fenêtres de données via le menu : **Database > Show DB data**. Ici encore, il est nécessaire de sélectionner la base de données.

Une différence importante avec le niveau **Basic** : les modifications des données ne sont plus automatiquement propagées vers les fenêtres de données. La boîte de commande (figure 9) comprend cette fois trois boutons :

- **Select tables**, qui permet de choisir les tables à visualiser,
- **Refresh windows**, qui rafraîchit le contenu des fenêtres suite à une modification des données ou de la structure des tables,
- **Close all**, qui permet de fermer toutes les fenêtres de données.

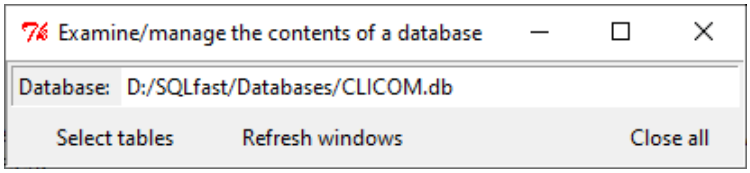

Figure 9 - Boîte de commande des fenêtres de données

17. Comment continuer ?

Le présent document n'est qu'une brève introduction à l'environnement SQLfast au niveau **Expert**. Pour aller plus loin, on suggérera les quatre sources suivantes :

a) Les guides de référence intégrés

Les détails techniques de l'interface du niveau **Expert**, sont disponibles via le menu : **Help > SQLfast references**. On y consultera en particulier :

- **SQLfast environment** : description systématique de l'interface du niveau **Expert** et de ses fonctions
- **SQLfast commands** : une synthèse du langage SQLfast
- **SQLtuto language** : une description du langage de rédaction de tutoriels

b) Les modèles de script (*Survival guide*)

Le tutoriel intitulé **Survival guide** accessible via le menu **Help**, et déjà cité dans ce document, présente une série de modèles de scripts élémentaires correspondant à des problèmes et situations fréquents : créer une base de données, modifier le contenu d'une base de données, utiliser les variables SQLfast, instructions alternatives, scripts à boucles, procédures, formatage des résultats, requêtes complexes, etc.
La plupart des modèles sont prêts à être exécutés.

c) Le manuel SQLfast

On a vu que le support didactique du niveau **Basic**, est constitué, d'une part, de l'ouvrage de référence *Bases de données - Concept, utilisation et développement, 4e édition* publié chez Dunod et d'autre part, d'une suite de tutoriels intégrés reprenant les matériaux et exercices des chapitres 6, 7, 8 et 9.

Le support du niveau **Expert** est constitué pour l'instant du manuel **SQLfast-manual.pdf** comportant 26 chapitres. Chaque chapitre aborde de manière très détaillée à la fois l'utilisation du langage SQLfast et son application à la résolution de problèmes variés. Ils sont disponibles en ligne à l'adresse suivante :

https://staff.info.unamur.be/dbm/Documents/Tutorials/SQLfast/SQLfast-Manual.pdf

d) Les tutoriels intégrés

Le menu **Help** contient également une rubrique **Tutorials**. Ceux-ci donne accès à une série de tutoriels, dérivés chacun d'un chapitre du manuel mentionné ci-dessus.

18. Quelques remarques générales

18.1 a) Comment rédiger un script SQLfast ?

On peut, comme nous l'avons fait, rédiger et sauver de petits scripts à l'aide de la fenêtre principale. Elle offre cependant un éditeur de texte trop rudimentaire pour créer des scripts plus complexes. Deux suggestions pratiques : Wordpad de Windows ou, mieux encore, Notepad++ (https://notepad-plus-plus.org), qu'il est possible de paramétrer pour la syntaxe de SQLfast.

18.2 b) Toutes les fenêtres SQLfast sont éditables

Le contenu de toutes les fenêtres de texte de SQLfast (fenêtre principale, fenêtre de sortie, fenêtre de schéma, fenêtres de données, fenêtre d'aide, etc.) est modifiable et copiable. Les modifications n'ont d'utilité réelle que pour les fenêtres principale et de sortie, dont le contenu peut être sauvé.

18.3 c) Style des fenêtres SQLfast

Conformément au comportement de la bibliothèque graphique *Tkinter*, les fenêtres et autres boîtes de dialogues de SQLfast adoptent la style de la plateforme sur laquelle le logiciel est exécuté. Leur apparence peut donc être légèrement différente de celle des figures de ce document.

18.4 d) Pourquoi les fenêtres de données ne sont-elles pas automatiquement mises à jour ?

(*Cette section est destinée au lecteur averti, curieux des détails techniques relatifs aux interfaces SQLfast.*)

Au niveau **Basic**, la base de données courante est la source unique de données pour les requêtes SQL, pour la fenêtre du schéma et pour les fenêtres de données. Lors de son ouverture, une seule connexion est créée, utilisée les trois processeurs accédant à la base de données : moteur *SQLfast*, affichage du schéma, affichage des données. En outre, dès qu'une requête SQL de modification ou une modification directe dans une fenêtre de données a terminé son exécution, un *commit* implicite est exécuté (mode *autocommit*) de sorte que les données ainsi mises à jour sont visibles immédiatement par les deux autres processeurs.

Dans le mode **Expert**, c'est le script qui décide de la manière dont les transactions sont gérées. Pendant une transaction, les données ne sont pas accessibles par les autres processeurs (en simplifiant), ce qui conduit à des incohérences entre les versions des données. On a donc rendu les trois processeurs indépendants, chacun créant sa propre connexion avec la base de données de son choix. Dans le cas où ils choisissent la même base de données, la synchronisation se fait manuellement, par

une réouverture de la fenêtre du schéma et par le bouton **Refresh windows**. Si la synchronisation ne se fait pas, ou tarde, c'est qu'un script en cours d'exécution a ouvert une transaction mais ne l'a pas encore clôturée par un *commitDB* ou un *abortDB*.

18.5 e) SQLfast peut-il "crasher" ?

(*Cette section est destinée au lecteur averti, curieux des détails techniques relatifs aux interfaces SQLfast*.)

SQLfast est un logiciel développé avec soin et largement testé depuis près de huit ans, mais ... en évolution constante. Il est donc raisonnablement fiable. Néanmoins, oeuvre humaine et donc imparfaite, il contient certainement l'une ou l'autre erreur (plus probablement l'une ET l'autre) non encore identifiées.

Le logiciel SQLfast est constitué de plusieurs modules. Le premier, *SQLfast*, est responsable des interfaces graphique **Basic** *et* **Expert**. Il crée et gère la fenêtre principale, les fenêtres annexes et leurs différentes fonctions. Le module central est constitué du moteur *SQLfastEngine*, chargé de l'exécution des requêtes SQL et des scripts. Il est largement le plus volumineux et le plus complexe.

Si une erreur se produit, on considère deux cas :

– L'erreur survient dans le premier module *SQLfast*. Dans ce cas, le logiciel SQLfast s'arrête après avoir enregistré la description précise de l'erreur dans un fichier nommé SQLfast.exe.log, créé dans le répertoire standard SQLfast.

– L'erreur survient dans le moteur *SQLfastEngine*. Dans ce cas, la description de l'erreur est également enregistrée dans un fichier SQLfast.exe.log mais SQLfast poursuit son exécution.

Que faire en cas d'erreur ? Prendre la peine de me transmettre le fichier SQLfast.exe.log accompagné d'un maximum de détails sur l'incident.