

# L'interpréteur de script SQL pour Microsoft Access (v4)

## ***SQL-Interpreter-FR (v4).mdb***

### 0. Préambule

Ce document décrit les versions 2, 3 et 4 de l'interpréteur de scripts SQL développé pour Access de Microsoft. Cet interpréteur est capable d'exécuter des scripts constitués d'une séquence de requêtes create, alter, drop, insert, delete et update stockées dans un fichier texte.

La version 4 (SQL-Interpreter-FR (v4).mdb) est disponible depuis 2008. Elle offre notamment les nouvelles fonctionnalités suivantes :

- ◆ variables de script
- ◆ requêtes et instructions paramétrables via les variables de script
- ◆ scripts interactifs via les instructions ask et display
- ◆ instructions delimiter, set, ask et compute de gestion et d'assignation de valeurs aux variables de script
- ◆ branchements (goto, étiquettes), alternatives (if)

On consultera le document The SQL-Script Language.rtf pour une description complète de la version 4 du langage de script.

### 1. Introduction

*SQL-Interpreter-FR.mdb* est une application qui comble une lacune importante d'Access. Microsoft Access permet en effet,

- ◆ de construire et sauvegarder des requêtes isolées (onglet Requetes d'une base de données),
- ◆ de développer en Visual Basic (VB) des programmes de complexité quelconque, qui incluent des requêtes SQL.

La première possibilité est très pratique, mais présente trois défauts qui en limitent fortement l'utilité :

1. les requêtes sont stockées dans la base de données et ne peuvent être réutilisées pour d'autres bases de données (elle peuvent cependant être *importées*);
2. les requêtes doivent être introduites manuellement et ne peuvent être produites par d'autres moyens, tels qu'une génération automatique par exemple,
3. plus grave, on ne peut inclure dans le texte d'une requête qu'une seule instruction SQL.

La seconde possibilité (programme VB) exige une connaissance technique qui n'est pas à la portée de l'utilisateur ordinaire et est clairement réservée aux programmeurs. VB est en effet un langage complet relativement complexe, particulièrement en ce qui concerne l'accès à la base de données.

Entre ces deux extrêmes, il n'existe aucune autre possibilité d'exécuter facilement une suite de requêtes SQL sur une base de données. On observe pourtant que la plupart des SGBD (tels qu'Oracle, InterBase ou Firebird) permettent d'exécuter de manière séquentielle des instructions SQL rangées dans un fichier, ce qu'on appelle un *script*. C'est ce que propose l'application *SQL-Interpreter.mdb*, et en particulier la version française *SQL-Interpreter-FR.mdb*

## 2. Qu'est-ce que *SQL-Interpreter.mdb* ?

Il s'agit d'une base de données Access (ou plus exactement d'une *application* Access) **vide**, à l'exception d'un formulaire nommé *Execute SQL script*, et de quelques fonctions qui lui sont associées et que nous pouvons ignorer. En particulier, cette application ne contient initialement **aucune table ni aucune requête**.

Le formulaire, nommé "**Exécuter un script SQL**" est très simple et permet d'effectuer deux opérations :

- ◆ sélectionner un script SQL (Locate script ou **Rechercher script**),
- ◆ exécuter le script ainsi sélectionné (Execute script ou **Exécuter script**).

## 3. Qu'est-ce qu'un script SQL Access ?

Examinons d'abord le fichier CLICOM-Creer.sql, qui est un bon exemple de script élémentaire. Nous l'ouvrons à l'aide d'un traitement de texte quelconque, tel que Bloc-note (Notepad) par exemple. On y trouve une suite d'instructions SQL dont l'objectif est assez évident : créer une petite base de données de quatre tables et y insérer quelques données. Le script comporte les instructions de création des tables CLIENT, PRODUIT, COMMANDE et DETAIL, puis les instructions d'insertion des données dans chacune de ces tables. On observe que certaines requêtes sont écrites en plusieurs fragments consécutifs. On repère également des lignes blanches et des commentaires préfixés par "--".

### **Plus précisément ...**

Un script est un fichier de texte, à créer avec un logiciel tel que Bloc-note (Notepad) ou Wordpad de Windows ou encore emacs. On donnera à ce fichier l'extension ".sql".

Un fichier de script contient un nombre quelconque de composants des types suivants,

- ◆ instructions SQL,
- ◆ lignes blanches,
- ◆ commentaires.

Une **instruction SQL** est du type create, drop, alter, insert, delete et update. L'instruction select n'est pas acceptée : si une extraction est nécessaire, on l'exprimera sous la forme d'une requête Access (onglet Requête) ou on créera dans un script une table dans laquelle on rangera le résultat de l'extraction par une instruction insert into ... select ... .

Une instruction SQL peut être écrite en une ou plusieurs lignes consécutives. Une instruction est terminée lorsqu'on rencontre :

- ♦ le caractère ";"
- ♦ ou une ligne blanche,
- ♦ ou la fin du fichier.

On ne peut donc trouver de lignes blanches ni de commentaires au milieu d'une instruction. Chaque fragment (ligne) d'une instruction peut être précédé ou suivi d'espaces. Ceux-ci sont ignorés. Avant l'exécution d'une instruction, celle-ci est reconstituée à partir de ses fragments comme suit : les espaces précédant et suivant les fragments sont éliminés, les fragments sont ensuite concaténés (*recollés*) mais avec insertion d'un espace intercalaire. Il faut donc être attentif lorsqu'on fragmente une longue requête en plusieurs morceaux. On évitera en particulier d'effectuer une coupure au milieu d'une constante ou d'un mot SQL.

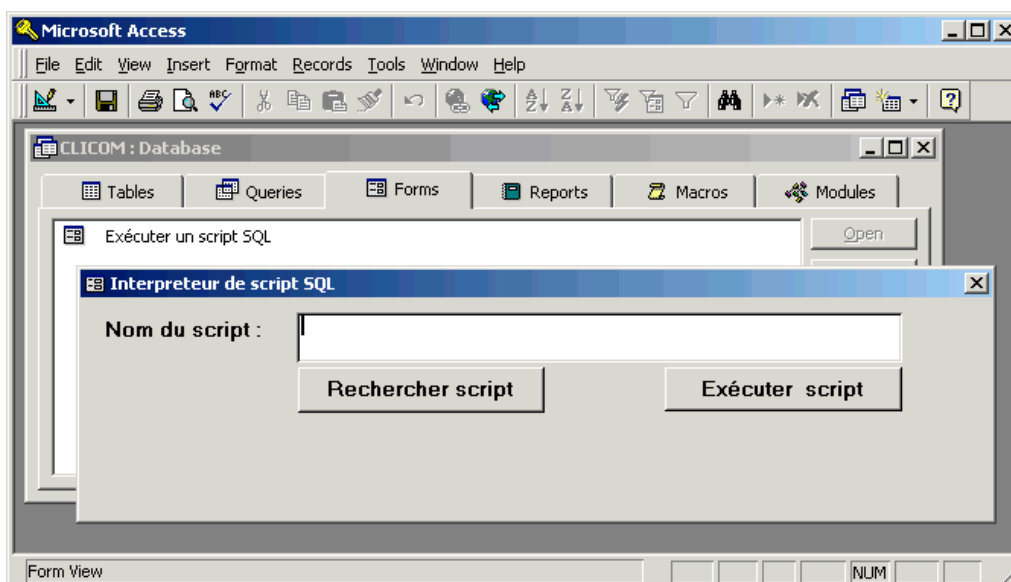
Une **ligne blanche** est vide ou contient seulement des caractères espace.

Une **ligne de commentaire** est préfixée par les caractères "--". Elle est ignorée par l'interpréteur.

#### 4. Comment utiliser *SQL-Interpreter.mdb* ?

Illustrons l'utilisation de l'interpréteur par une manipulation simple. Nous supposons que les fichiers *SQL-Interpreter.mdb* et *CLICOM-Creer.sql* sont placés dans le répertoire courant E:\Access\Scripts.

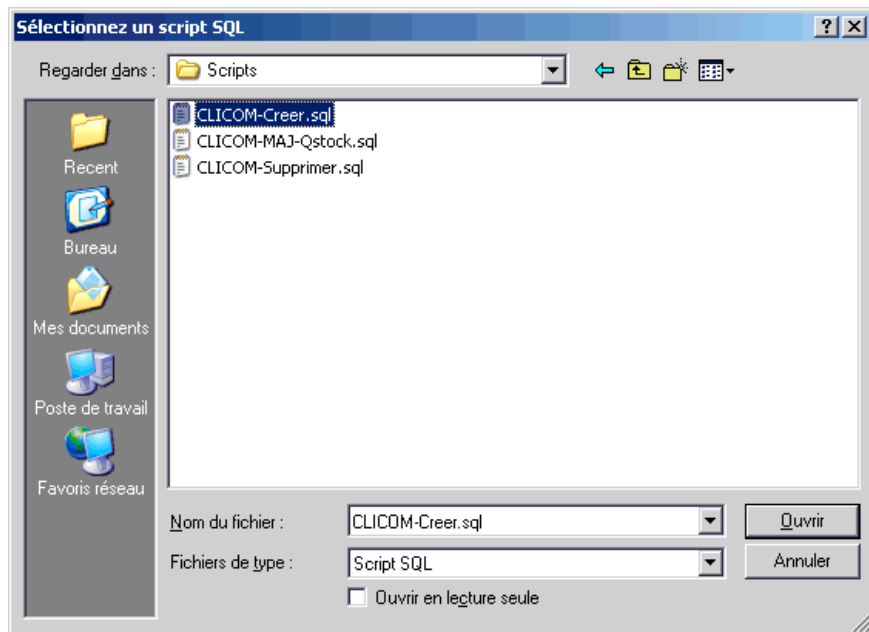
1. Copions l'application *SQL-Interpreter.mdb* sous la forme du fichier *CLICOM.mdb*.
2. Ouvrons l'application *CLICOM.mdb* avec Access; il n'y existe pour l'instant aucune table.
3. Cliquons sur l'onglet Formulaires et ouvrons le formulaire Exécuter un script SQL (Figure 1<sup>1</sup>).



**Figure 1** - Ouverture du formulaire Exécuter un script SQL script

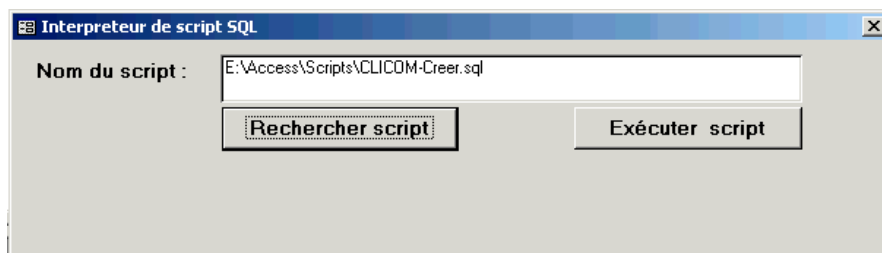
<sup>1</sup> Illustration en Access 97. L'interface est équivalente mais légèrement différente dans les versions ultérieures.

4. Sélectionnons le script à exécuter en cliquant sur le bouton Rechercher script du formulaire. La boîte de dialogue de sélection de fichier s'ouvre dans le répertoire courant (Figure 2).



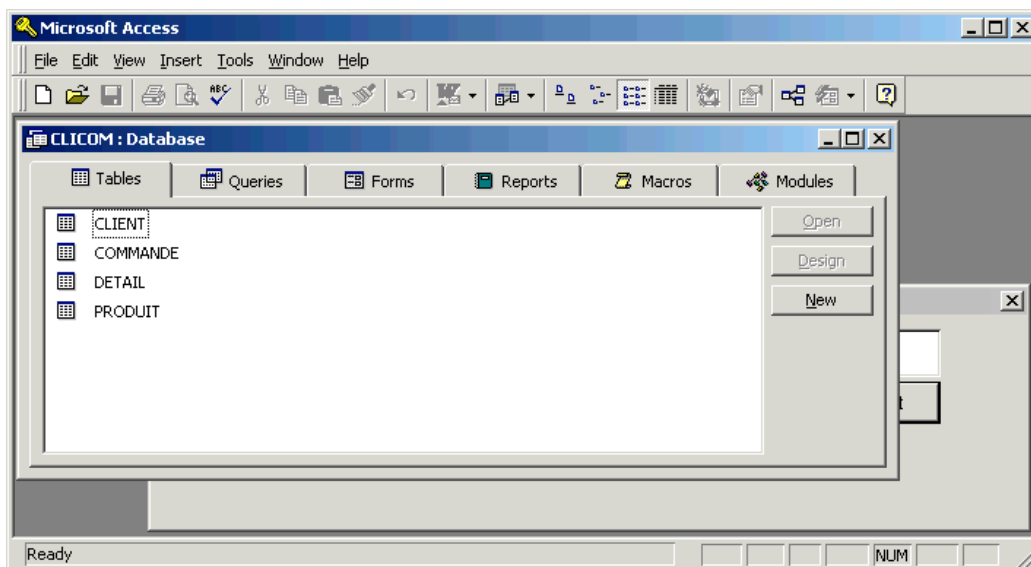
**Figure 2** - Sélection du script à exécuter

5. Nous sélectionnons dans la liste le fichier CLICOM-Creer.sql et nous validons ce choix (bouton Ouvrir). Retour au formulaire, dont le champ Nom du script contient à présent le nom du script sélectionné (Figure 3).



**Figure 3** - Le script est sélectionné et peut être exécuté

6. Il nous reste alors à exécuter ce script en cliquant sur le bouton Exécuter Script du formulaire. En examinant l'onglet Tables, nous observons que les quatre tables ont été créées, et qu'elles contiennent les données désirées (Figure 4).



**Figure 4** - Le script a été exécuté et a créé et garni les quatre tables de la base de données CLICOM

7. Le formulaire de gestion de scripts est toujours ouvert et peut être utilisé pour l'exécution d'autres scripts. Il peut être fermé et réouvert à tout moment.
8. Avant, pendant et après l'utilisation de l'interpréteur via le formulaire, il est possible d'effectuer toutes les manipulations traditionnelles : créer et supprimer des tables, créer, supprimer et utiliser des requêtes, développer et utiliser des états, d'autres formulaires et d'autres procédures VB.

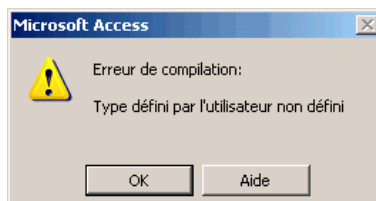
## Observation importante

Si vous développez une base de données ou une application Access pour laquelle vous désirez exécuter des scripts, développez-la dans une copie de *SQL-Interpreter.mdb*.

Si vous désirez exécuter des scripts sur une base de données existante, il est préférable de ne pas toucher à celle-ci, mais d'y accéder à partir d'une copie de l'interpréteur, auquel on aura attaché les tables utiles de la base de données (Fichier/Données externes/Lier les tables).

## 6. Problème potentiel

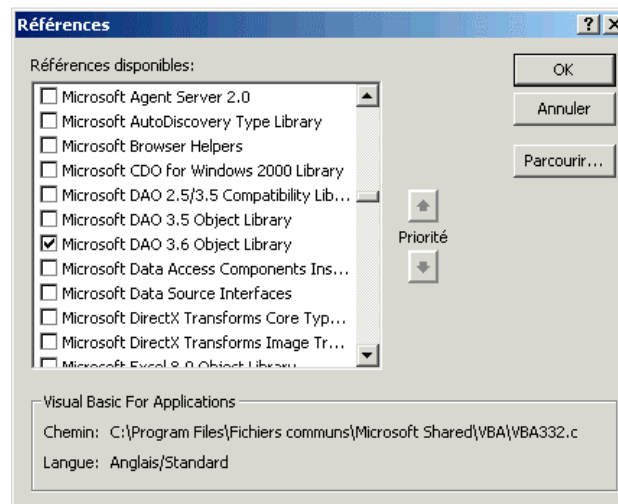
Il est possible qu'un script refuse de s'exécuter lorsqu'on clique sur le bouton Exécuter script. Si le message est le suivant :



... il convient alors de rendre accessible la bibliothèque de fonctions qui permet l'accès à la base de données<sup>2</sup>.

On procède comme suit (suggestion de J. Henrard) :

- dans la fenêtre de la base de données (figure 4), on sélectionne l'onglet Modules;
- on double-clique sur l'objet qui y apparaît (modFileFunctions), ce qui ouvre la fenêtre d'édition Visual Basic;
- dans le menu, on appelle Outil/Références, ce qui affiche une boîte de sélection de bibliothèques (figure 5);



**Figure 5** - Une bibliothèque Microsoft DAO 3.x Object Library doit être sélectionnée

- la bibliothèque Microsoft DAO 3.6 Object Library (ou version ultérieure) doit être cochée.

## 7. Exemples de scripts

### 7.1 Le premier script crée et garni les tables de la base de données CLICOM

```
-- Exemple de script SQL pourACCESS de Microsoft
-- A exécuter via l'interpréteur de script SQL "SQL-Interpreter.mdb"
-- Septembre 2007

-- Création des tables de la base de données CLICOM

create table CLIENT
(NCLI char(8) not null,
NOM char(18) not null,
ADRESSE char(24) not null,
LOCALITE char(20) not null,
CAT char(2),
COMPTE long not null,
constraint PKCLI primary key (NCLI));
```

<sup>2</sup> Par défaut, Access ne permet pas l'accès à une base de données à partir de procédures VB !

```

create table PRODUIT
(NPRO char(10) not null,
LIBELLE char(30) not null,
PRIX long not null,
QSTOCK long not null,
constraint PKPRO primary key (NPRO));

create table COMMANDE
(NCOM char(8) not null,
NCLI char(8) not null,
DATECOM date not null,
constraint PKCOM primary key (NCOM),
constraint FKCOMCLI foreign key (NCLI) references CLIENT);

create table DETAIL
(NCOM char(8) not null,
NPRO char(10) not null,
QCOM integer not null,
constraint PKDET primary key (NCOM,NPRO),
constraint FKDETCOM foreign key (NCOM) references COMMANDE,
constraint FKDETPRO foreign key (NPRO) references PRODUIT);

-- Insertion des données types

insert into CLIENT values
('B112','HANSENNE'      , '23, a. Dumont'      , 'Poitiers' , 'C1',1250);
insert into CLIENT values
('C123','MERCIER'       , '25, r. Lemaitre'    , 'Namur'    , 'C1',-2300);
insert into CLIENT values
('B332','MONTI'         , '112, r. Neuve'      , 'Geneve'   , 'B2',0);
... etc.

insert into PRODUIT values ('CS262','CHEV. SAPIN 200x6x2', 75, 45);
insert into PRODUIT values ('CS264','CHEV. SAPIN 200x6x4', 120,2690);
... etc.

insert into COMMANDE values ('30178','K111','21/12/2005');
insert into COMMANDE values ('30179','C400','22/12/2005');
... etc.

insert into DETAIL values ('30178','CS464',25);
insert into DETAIL values ('30179','PA60',20);
insert into DETAIL values ('30179','CS262',60);
... etc.

```

## 7.2 Le script suivant extrait des données des tables de base et les range dans une table dérivée qu'il crée préalablement.

```

-- Exemple de script SQL pour ACCESS de Microsoft
-- À exécuter via l'interpréteur de script SQL "SQL-Interpreter.mdb"
-- Septembre 2007

create table CLIENT_CAT
(NCLI char(8) not null,
Nom varchar(20) not null);

```

```
insert into CLIENT_CAT
select NCLI,NOM
from   CLIENT
where  CAT is not null;
```