FAQ Base intégrée et requêtes SQL directes

Version 1.1 du 17.10.2006

Réalisé avec : **OOo 2.0.4**

Plate-forme / Os : **Toutes**

Distribué par le projet fr.OpenOffice.org

Sommaire

1 Présentation

Ce document est destiné aux utilisateurs d'OpenOffice.org 2 désirant exécuter des requêtes SQL en mode direct, sans utiliser les aides apportées par le mode ébauche de requêtes. De plus, il est concentré sur l'exploitation d'une base intégrée Hsqldb, même si certaines informations peuvent être valables avec d'autres systèmes de base de données.

2 Comment exécuter des requêtes SQL directement sous Base ?

Base permet d'exécuter des requêtes SQL directement, sans interprétation par Base. Pour utiliser ce mode, il faut cliquer sur l'icône «Requêtes » de la colonne « Base de données », puis menu Insérer > Requête (mode SQL). Une fenêtre d'édition de requête s'ouvre à partir de laquelle vous pourrez saisir, exécuter et sauvegarder des requêtes SQL.

Activez l'option SQL direct par le menu Éditer > Exécuter directement l'instruction SQL. Dans ce mode, vous ne pouvez plus utiliser le mode Ébauche, mais vous êtes sûr que OOo ne modifie pas votre requête et l'envoie directement à Hsqldb.

Pour exécuter la requête saisie, sélectionnez le menu Éditer > Exécuter la requête. La zone d'édition des requêtes se scinde alors en deux pour afficher le résultat de la requête.

3 Comment saisir une requête SQL ?

Les noms de tables et de champs peuvent être saisis sans guillemet ou bien entourés de guillemets doubles :

```
SELECT NOM FROM CLIENTS
```

est valide , ainsi que

```
SELECT "NOM" FROM "CLIENTS"
```

Néanmoins, les guillemets doubles sont obligatoires lorsque les noms comportent des caractères spéciaux ou accentués, ou des espaces. J'utiliserai donc les guillemets doubles dans ce document.

Si vous utilisez les guillemets doubles, les espaces ne sont pas obligatoires :

```
SELECT"NOM"FROM"CLIENTS"
```

Le point virgule en fin de requête n'est pas nécessaire.

Les nombres passés en paramètre sous forme littérale ne sont encadrés d'aucun guillemet :

```
SELECT * FROM "CLIENTS" WHERE "ID_CLIENT" = 1
```

ou bien (pour les nombres décimaux, c'est le point (.) qui sert de séparateur) :

```
SELECT * FROM "PRODUITS" WHERE "PX_UNITAIRE" = 7.65
```

Les chaînes de caractères passées en paramètre sous forme littérale sont encadrées de guillemets simples :

```
SELECT "ID_CLIENT" FROM "CLIENTS" WHERE "NOM" = 'gautier'
```

Les dates passées en paramètre sous forme littérale sont encadrées de guillemets simples et doivent être au format AAAA-MM-JJ :

```
SELECT "ID_CLIENT" FROM "CLIENTS" WHERE "DATE_CREA" < '2006-02-01'
```

Les heures passées en paramètre sous forme littérale sont encadrées de guillemets simples et doivent être au format HH:MM:SS.
Les booléens passés en paramètre sous forme littérale sont de la forme 0 pour FALSE ou autre entier pour TRUE :

```
SELECT * FROM "COMMANDES" WHERE "VALID_LIVRAISON" = 1
```

Vous pouvez également utiliser TRUE ou FALSE :

```
SELECT * FROM "COMMANDES" WHERE "VALID_LIVRAISON" = TRUE
```

4 Au secours, OOo modifie mes requêtes !

Vous avez saisi une requête dans la fenêtre requête SQL :

```
SELECT "NOM" FROM "CLIENTS"
```

Vous sauvegardez, vous fermez, et lorsque vous voulez éditer cette requête, c'est l'éditeur en mode ébauche qui s'ouvre. Si vous désactivez le mode ébauche, votre requête se présente ainsi:

```
SELECT "NOM" FROM "CLIENTS" "CLIENTS"
```

Le nom de la table est désormais répété deux fois. C'est probablement que vous avez désactivé l'option SQL direct lors de la sauvegarde précédente, et OOo a retravaillé la requête. La requête ainsi modifiée s'exécute normalement, mais la lecture n'est pas facilitée.

5 Quelles sont les commandes SQL supportées en mode requête SQL direct ?

Il semble que la seule commande SQL supportée dans le mode requête SQL direct soit la commande SELECT telle qu'elle est définie dans la documentation Hsqldb (disponible en ligne à l'adresse http://hsqldb.org/web/hsqlDocsFrame.htm).

6 Comment exécuter d'autres commandes SQL que SELECT ?

Pour exécuter d'autres commandes SQL supportées par Hsqldb, il faut utiliser un autre outil SQL, accessible dans le menu principal de Base en sélectionnant Outils > SQL ....
Il est alors possible de saisir d'autres commandes SQL.
Je n'ai pas testé cet outil de manière poussée mais vous pouvez créer une table par ce biais.
Saisissez l'instruction suivante :

```
CREATE TABLE CLIENTS (ID_CLIENT INTEGER IDENTITY, NOM VARCHAR(50),
DATE_CREA DATE)
```

et cliquez sur Exécuter. Un message vous confirme que l'instruction a bien été exécutée. Mais si vous affichez les tables, la table CLIENTS n'apparaît pas . Il faut ensuite, toujours avec les tables affichées, sélectionner le menu Afficher > Actualiser les tables pour que la table ainsi créée apparaisse.
Pour utiliser des caractères en minuscules ou spéciaux, il faut encadrer les chaînes littérales de guillemets doubles :

```
CREATE TABLE "Clients" ("id_client" INTEGER IDENTITY, "Nom" VARCHAR(50),
"Date_créa" DATE)
```

La commande INSERT semble aussi fonctionner. Il faut donc tester avant d'utiliser cet outil.

7 Quelles sont les fonctions et procédures intégrées supportées ?

Là encore, c'est la documentation Hsqldb qui peut vous renseigner. Voir le chapitre «Built-in Functions and Stored Procedures».
Pour les fonctions d'agrégation statistique, sont supportées :

- AVG (Moyenne)
- COUNT (Nombre)
- MAX (Maximum)
- MIN (Minimum)
- SUM (Somme)

8 Les fonctions intégrées retournent des valeurs NULL

Si vous n'attribuez pas d'alias à une fonction :

```
SELECT SUM("QTE_PROD") FROM "LIGNE_COMMANDE" WHERE "REF_PROD" = 1
```

le résultat apparaît vide. Par contre :

```
SELECT SUM("QTE_PROD") AS "Nombre de produits vendus" FROM "LIGNE_COMMANDE"
WHERE "REF_PROD" = 1
```

fonctionne bien. Il faut donc systématiquement attribuer un alias aux fonctions et procédures.

9 Les regroupements renvoient des messages d'erreur

La requête :

```
SELECT "REF_COM", SUM("QTE_PROD"*"PX_UNITAIRE") AS "Total commande" FROM
"LIGNE_COMMANDE", "PRODUITS" WHERE "REF_PROD" = "ID_PRODUIT" GROUP BY
"REF_COM"
```

fonctionne correctement. Mais si on rajoute un champ dans la requête :

```
SELECT "REF_COM", "REF_PROD", SUM("QTE_PROD"*"PX_UNITAIRE") AS "Total
commande" FROM "LIGNE_COMMANDE", "PRODUITS" WHERE "REF_PROD" = "ID_PRODUIT"
GROUP BY "REF_COM"
```

on a droit à un beau message d'erreur évoquant les fonctions d'agrégation.

« La présence de la clause GROUP BY est nécessaire dès que la clause de sélection, ou le filtre WHERE, ou encore les jointures comportent simultanément des calculs d'agrégation et la présence de colonnes de table hors de calculs d'agrégation.

De plus, toutes les colonnes représentées hors des calculs d'agrégation doivent figurer dans la clause GROUP BY » (SQLpro dans http://sql.developpez.com/sqlaz/ensembles/)

Il faut donc saisir les requêtes ainsi :

```
SELECT "REF_COM", "REF_PROD", SUM("QTE_PROD"*"PX_UNITAIRE") AS "Total
commande" FROM "LIGNE_COMMANDE", "PRODUITS" WHERE "REF_PROD" = "ID_PRODUIT"
GROUP BY "REF_COM", "REF_PROD"
```

10 Annexes

10.1 Création de la table servant aux exemples de requêtes

Vous pouvez créer et remplir la base de données qui sert de support aux exemples de requêtes en suivant les instructions suivantes:

Créez un nouveau document Base et attribuez lui un nom de votre choix.

Avec Outils > SQL, exécutez successivement les commandes suivantes (voir ici) :

```
CREATE TABLE PRODUITS (ID_PRODUIT INTEGER IDENTITY, NOM_PROD VARCHAR(50),
PX_UNITAIRE DECIMAL)

INSERT INTO PRODUITS (NOM_PROD, PX_UNITAIRE) VALUES ('marteau', 5.00)

INSERT INTO PRODUITS (NOM_PROD, PX_UNITAIRE) VALUES ('tournevis', 4.10)

INSERT INTO PRODUITS (NOM_PROD, PX_UNITAIRE) VALUES ('scie', 7.65)

INSERT INTO PRODUITS (NOM_PROD, PX_UNITAIRE) VALUES ('rabot', 8.50)

CREATE TABLE CLIENTS (ID_CLIENT INTEGER IDENTITY, NOM VARCHAR(50),
DATE_CREA DATE)

INSERT INTO CLIENTS (NOM, DATE_CREA) VALUES ('dupond', '2006-01-23')

INSERT INTO CLIENTS (NOM, DATE_CREA) VALUES ('legrand', '2005-03-10')

INSERT INTO CLIENTS (NOM, DATE_CREA) VALUES ('gautier', '2006-02-28')
```

```
INSERT INTO CLIENTS (NOM, DATE_CREA) VALUES ('leroy', '2006-02-01')

CREATE TABLE COMMANDES (ID_COM INTEGER IDENTITY, REF_CLIENT INTEGER,
DATE_COM DATE, VALID_LIVRAISON BOOLEAN)

INSERT INTO COMMANDES (REF_CLIENT, DATE_COM, VALID_LIVRAISON) VALUES (1,
'2006-03-01', TRUE)

INSERT INTO COMMANDES (REF_CLIENT, DATE_COM, VALID_LIVRAISON) VALUES (3,
'2006-03-10', FALSE)

CREATE TABLE LIGNE_COMMANDE (ID_LIGNE_COM INTEGER IDENTITY, REF_COM
INTEGER, REF_PROD INTEGER, QTE_PROD INTEGER)

INSERT INTO LIGNE_COMMANDE (REF_COM, REF_PROD, QTE_PROD) VALUES (0, 1, 2)

INSERT INTO LIGNE_COMMANDE (REF_COM, REF_PROD, QTE_PROD) VALUES (1, 0, 1)

INSERT INTO LIGNE_COMMANDE (REF_COM, REF_PROD, QTE_PROD) VALUES (1, 2, 2)
```

Si les tables n'apparaissent pas à l'issue de cette procédure, effectuez un rafraîchissement des tables (Afficher > Actualiser les tables).

10.2 Export de la structure et des données d'une base intégrée

L'instruction SCRIPT permet de créer un fichier texte comportant l'ensemble des éléments nécessaires à la création et au remplissage de la base. Avec Outils > SQL, exécutez la commande suivante :

```
SCRIPT 'Chemin complet vers un repertoire/monScript.script'
```

Sur la base créée pour les exemples, vous obtenez le résultat suivant dans le fichier monScript.script :

```
SET DATABASE COLLATION "French"
CREATE SCHEMA PUBLIC AUTHORIZATION DBA
CREATE CACHED TABLE PRODUITS(ID_PRODUIT INTEGER GENERATED BY DEFAULT AS
IDENTITY(START WITH 0) NOT NULL PRIMARY KEY,NOM_PROD
VARCHAR(50),PX_UNITAIRE DECIMAL)
CREATE CACHED TABLE CLIENTS(ID_CLIENT INTEGER GENERATED BY DEFAULT AS
IDENTITY(START WITH 0) NOT NULL PRIMARY KEY,NOM VARCHAR(50),DATE_CREA DATE)
CREATE CACHED TABLE COMMANDES(ID_COM INTEGER GENERATED BY DEFAULT AS
IDENTITY(START WITH 0) NOT NULL PRIMARY KEY,REF_CLIENT INTEGER,DATE_COM
DATE,VALID_LIVRAISON BOOLEAN)
CREATE CACHED TABLE LIGNE_COMMANDE(ID_LIGNE_COM INTEGER GENERATED BY
DEFAULT AS IDENTITY(START WITH 0) NOT NULL PRIMARY KEY,REF_COM
INTEGER,REF_PROD INTEGER,QTE_PROD INTEGER)
ALTER TABLE PRODUITS ALTER COLUMN ID_PRODUIT RESTART WITH 4
ALTER TABLE CLIENTS ALTER COLUMN ID_CLIENT RESTART WITH 4
ALTER TABLE COMMANDES ALTER COLUMN ID_COM RESTART WITH 2
ALTER TABLE LIGNE_COMMANDE ALTER COLUMN ID_LIGNE_COM RESTART WITH 3
CREATE USER SA PASSWORD ""
GRANT DBA TO SA
SET WRITE_DELAY 60
SET SCHEMA PUBLIC
INSERT INTO PRODUITS VALUES(0,'marteau',5.00)
INSERT INTO PRODUITS VALUES(1,'tournevis',4.10)
INSERT INTO PRODUITS VALUES(2,'scie',7.65)
INSERT INTO PRODUITS VALUES(3,'rabot',8.50)
INSERT INTO CLIENTS VALUES(0,'dupond','2006-01-23')
INSERT INTO CLIENTS VALUES(1,'legrand','2005-03-10')
INSERT INTO CLIENTS VALUES(2,'gautier','2006-02-28')
```

```
INSERT INTO CLIENTS VALUES(3,'leroy','2006-02-01')
INSERT INTO COMMANDES VALUES(0,1,'2006-03-01',TRUE)
INSERT INTO COMMANDES VALUES(1,3,'2006-03-10',FALSE)
INSERT INTO LIGNE_COMMANDE VALUES(0,0,1,2)
INSERT INTO LIGNE_COMMANDE VALUES(1,1,0,1)
INSERT INTO LIGNE_COMMANDE VALUES(2,1,2,2)
```

11 Crédits

Auteur : **Manuel Naudin**

Remerciements : **Fernand COSTA, Chris DRAUX, Tony GALMICHE**

Intégré par : **Tony GALMICHE**

Contacts : **Projet Documentation OpenOffice.org - f.OpenOffice.org**

Traduction :

Historique des modifications**:**

Version	*Date*	*Commentaire*
1.0	19/03/06	Première version
1.1	17/10/06	Rectification d'une coquille dans le chapitre 6

12 Licence