MySQL avec Mac OS X

MySQL est un moteur de bases de données relationnelles (SGBDR système de gestion de bases de données relationnelles) basé sur le langage SQL. Il s'agit d'un langage normalisé de requêtes pour bases de données (Strutured Query Language) créé dans les années 70 par E. F. Codd d'abord pour IBM puis repris par Oracle.

1) Installation du moteur MySQL

Télécharger le fichier "mysql-standard-4.1.4-gamma-apple-darwin7.5.0-powerpc.dmg" sur le site "http://dev.mysql.com/downloads/mysql/4.1.html" pour Mac OS X 10.3 et plus.

Quelques manipulations avec le terminal sont nécessaires si une version de MySQL est déjà lancée:

```
$ /usr/local/mysql/bin/mysqladmin -u root -p shutdown
<votremotdepasse>
```

Lancer l'installation du moteur de bases de données contenu dans "mysql-standard-4.1.4-gamma-apple-darwin7.5.0-powerpc.pkg".

Puis lancer l'installation de l'élément de démarrage automatique du moteur à l'initialisation du Mac contenu dans "MySQLStartupItem.pkg".

Le moteur s'installe dans le répertoire /usr/local/mysql.
L'élément de démarrage s'installe dans le répertoire /Library/StartupItems/MySQLCOM.
Si par la suite vous ne voulez plus du démarrage automatique vous devrez modifier la variable "MYSQLCOM" de "YES" à "NO" dans le fichier /etc/hostconfig.

Les messages d'erreur seront personnalisés en français en créant le fichier de configuration adéquat :

```
$ sudo cat > /etc/my.cnf
[mysqld]
set-variable = language=french
^D
```

Ensuite quelques manipulations avec le terminal sont nécessaires pour lancer le serveur si cela ne l'a pas été de façon automatique :

```
$ sudo /usr/local/mysql/bin/mysqld_safe &
$ /usr/local/mysql/bin/mysqladmin -u root password
<votremotdepasse>
```

De cette façon le contrôle sera assuré par le moteur dont l'administrateur "root" est protégé par un mot de passe, à ne pas confondre avec l'utilisateur "root" du Mac.

2) Utilisation avec le terminal

Lancer l'interpréteur de commande interactif :
(La version réellement affichée peut être différente de celle présentée ici)

```
$ /usr/local/mysql/bin/mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 4.0.18-standard
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

Afficher la liste des commandes avec :

```
mysql> help;
For the complete MySQL Manual online visit:
   http://www.mysql.com/documentation
For info on technical support from MySQL developers visit:
   http://www.mysql.com/support
For info on MySQL books, utilities, consultants, etc. visit:
   http://www.mysql.com/portal
List of all MySQL commands:
   (Commands must appear first on line and end with ';')
help    (\h)    Display this help.
?       (\?)    Synonym for `help'.
clear   (\c)    Clear command.
connect (\r)    Reconnect to the server. Optional arguments are
db and host.
edit    (\e)    Edit command with $EDITOR.
exit    (\q)    Exit mysql. Same as quit.
...
quit    (\q)    Quit mysql.
```

```
rehash  (\#)    Rebuild completion hash.
source  (\.)    Execute a SQL script file. Takes a file name as
an argument.
status  (\s)    Get status information from the server.
system  (\!)    Execute a system shell command.
tee     (\T)    Set outfile [to_outfile]. Append everything into
given outfile.
use     (\u)    Use another database. Takes database name as
argument.
Connection id: 1  (Can be used with mysqladmin kill)
```

Pour créer une base de données nous devons nous connecter avec l'utilisateur "root" (au sens de MySQL) :

```
$ /usr/local/mysql/bin/mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.0.18-standard
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> create database bibli1;
Query OK, 1 row affected (0.14 sec)
mysql> show databases;
+--------------+
| Database     |
+--------------+
| bibli1       |
| mysql        |
| test         |
+--------------+
4 rows in set (0.00 sec)
```

Nous lui affectons les droits pour nous :
(Remplacer "toto" par votre nom d'utilisateur MySQL.)

```
mysql> grant all privileges on bibli1.* to toto@localhost;
Query OK, 0 rows affected (0.19 sec)
mysql> exit
Bye
```

Nous pouvons repasser en mode utilisateur :

```
$ /usr/local/mysql/bin/mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9 to server version: 4.0.18-standard
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> show databases;
+--------------+
| Database     |
+--------------+
| bibli1       |
| test         |
+--------------+
3 rows in set (0.00 sec)
mysql> use bibli1;
Database changed
mysql> show tables;
Empty set (0.00 sec)
mysql>
```

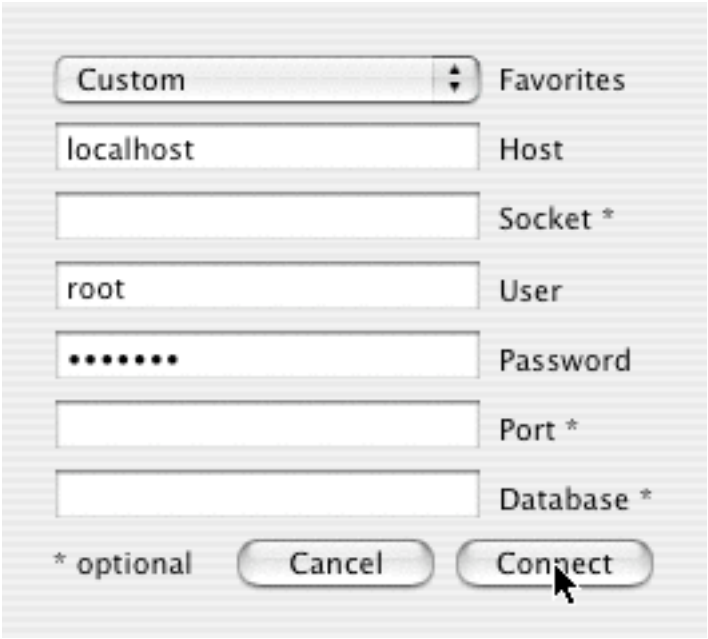
Voilà, la base de données est prête à recevoir les commandes pour créer la structure et saisir les données.

3) Installation et utilisation du logiciel CocoaMySQL

Télécharger le fichier "CocoaMySQL-v0.5.dmg.gz" sur le site "http://cocoamysql.sourceforge.netl".

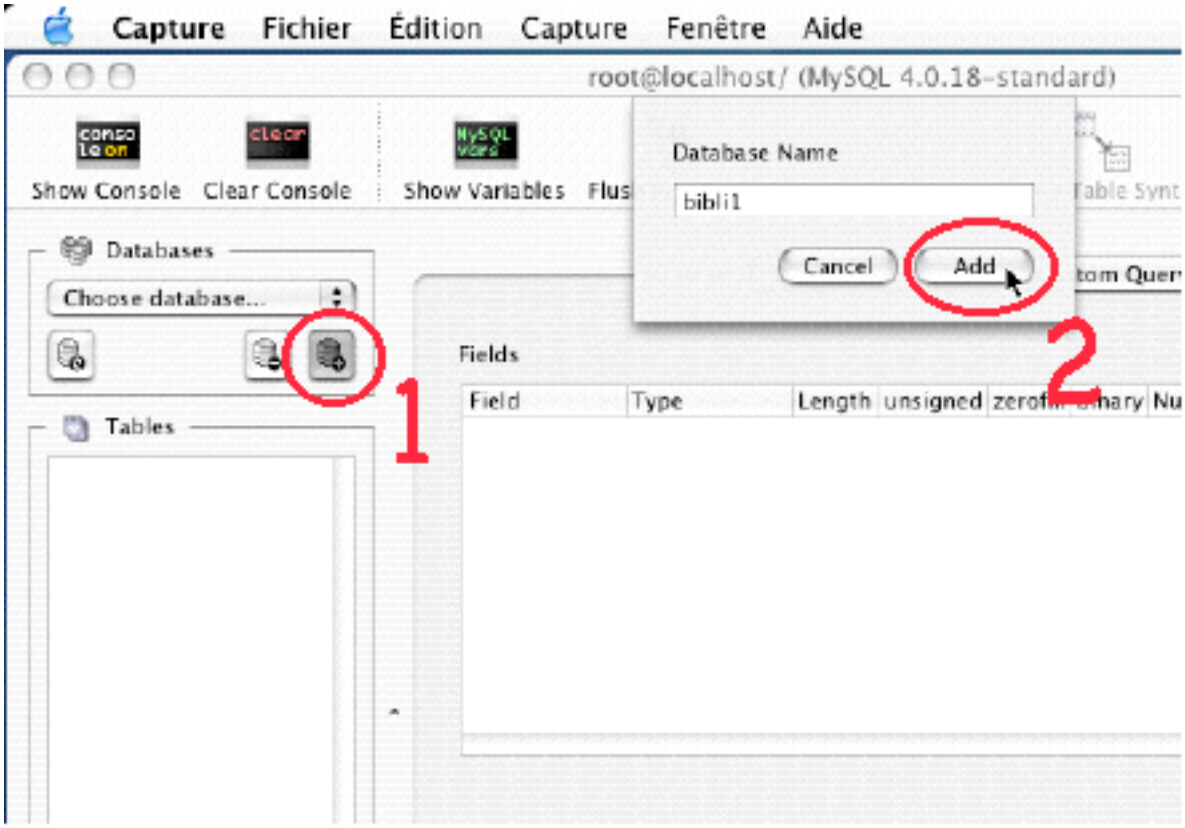
Ouvrer dans le Finder CocoaMySQL-v0.5.dmg.gz, l'image disque s'affiche sur le bureau. Copier le logiciel dans le dossier "Applications" et lancer le.

Pour créer une base de données nous devons nous connecter avec l'utilisateur "root" (au sens de MySQL). Dans la fenêtre de connexion entrer l'utilisateur "root" sur "localhost" avec son mot de passe :



Note : le cryptage des mots de passe de MySQL 4.1.x est incompatible avec CocoaMySQL 0.5.

La création se réalise en suivant les étapes 1 et 2 :
(La version réellement affichée peut être différente de celle présentée ici)



Nous lui affectons les droits pour nous en suivant les étapes 3, 4 et 5 :

Nous pouvons repasser en mode utilisateur. Fermons la fenêtre pour se déconnecter de "root" puis ouvrons en une nouvelle avec "New" du menu "File" :

Voilà, la base de données est prête à recevoir les actions pour créer la structure et saisir les données.
Aidez-vous des messages d'aide qui apparaissent sous le curseur ainsi que du menu "Help".

4) Installation de JDBC et utilisation avec Java

Java Database Connectivity (JDBC) fournit les API de liaison avec les moteurs de base de données, notamment pour les besoins d'applications utilisant Internet ou Intranet. JDBC est basé sur le langage SQL. Chaque pilote spécifique correspondant à un moteur de base de données SQL implémente l'API JDBC. Il peut en exister plusieurs dans l'application. Le choix du pilote est réalisé au moment du chargement du pilote et de la connexion avec le moteur de la base de données. Ce qui fait la puissance de Java. Il y a indépendance de la plate-forme (PC, Mac, Unix), du type d'application (exécutable, applet, servlet) et du moteur de base de données (Oracle, MySQL) grâce à Java et JDBC.

Télécharger le pilote JDBC de MySQL à l'adresse :
http://www.mysql.com/products/connector-j/index.html
Prendre la version correspondante à celle de MySQL.
Le pilote JDBC version 3.1.4 correspond à MySQL version 4.1.4.

Placer l'archive sur votre bureau.

Lancer l'installation sous /usr/local/mysql :

```
$ cd /usr/local/mysql
$ sudo tar xzvf ~/mysql-connector-java-3.1.4-beta.tar.gz
```

La bibliothèque du pilote JDBC est le fichier :
/usr/local/mysql-connector-java-3.1.4-beta/mysql-connector-java-3.1.4-beta-bin.jar

Essayons avec ce programme proposé par Marc Liyanage sur le site www.entropy.ch :

```
// testmysql.java
import java.sql.*;
public class testmysql {
    public static void main(String argv[]) throws Exception {
        // Load the driver class
       try {

Class.forName("com.mysql.jdbc.Driver").newInstance();
        } catch (Exception ex) {
            // handle the error
            System.out.println("Exception: " + ex.getMessage());
        }
        // Try to connect to the DB server.
        // We tell JDBC to use the "mysql" driver
        // and to connect to the "test" database
        // which should always exist in MySQL.
        // We use the username "" and no
        // password to connect. This should always
        // work for the "test" database.
    try {
        Connection conn = DriverManager.getConnection(
            "jdbc:mysql://localhost/test");
        // Set up and run a query that fetches
        // the current date using the "now()" SQL function.
        Statement stmt = conn.createStatement();
        ResultSet rset = stmt.executeQuery("SELECT now();");
        // Iterate through the rows of the result set
        // (obviously only one row in this example) and
        // print each one.
```

```
        while (rset.next()) {
            System.out.println(rset.getString(1));
        }
        rset.close();
        stmt.close();
        conn.close();
        } catch (SQLException ex) {
            // handle any errors
            System.out.println("SQLException: " +
ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " +
ex.getErrorCode());
        }
    }
}
```

Le programme charge dynamiquement le pilote SQL puis se connecte à la base "test" fourni par défaut avec MySQL. Compilons le source Java :

```
$ javac testmysql.java
```

Exécutons le programme avec la bibliothèque JDBC en paramètre :

```
$ java -classpath /usr/local/mysql/mysql-connector-java-3.1.4-
beta/mysql-connector-java-3.1.4-beta-bin.jar:. testmysql
2004-09-10 10:27:45
```

Le programme demande la date et l'heure à MySQL puis l'affiche.

5) Les commandes SQL usuelles

. création d'une table :
create table nom
(col1 type1 [default exp1] [contrainte1],
...,
[contrainte_table]);
. les types : char(n), varchar(2), number, date, int, blob, clob, bfile, long, raw,
. contraintes de colonne : not null, primary key, unique, references table(col), check (cond)
. contraintes de table : primary key, unique, foreign key, references table(col), check (cond)
. suppression d'une table :
drop table nom [cascade constraints};
. modification d'une table :
alter table nom {add|modify}
(col1 type1 [default exp1] [contrainte1]);
alter table nom add [constraint nom] contrainte;
alter table nom drop contrainte;

. création de lignes :
insert into table [(col1, ...)] values (exp1, ...);
. supression de ligne :
delete from table [where condition];
. modification de ligne :
update table set col1=exp1, ... [where condition];

. extraction des lignes avec condition et ordre :
select {col, exp} [as nom] from table [where conditions] [order by col [{asc, desc}]];
cond :
exp1 {=, !=, <>, <, <=, >, >=} exp2
exp1 in (val1, ...)
exp1 between val1 and val2
exp1 like '%_'
not cond
cond1 and cond2
cond1 or cond2
. opérateur in, comparaison modifié par any ou all, exists
. élimination des doublons :
select distinct col from table;
select col from table group by col

. **fonction sur groupes :**
select col, exp from table group by col [having cond];
. **produit cartésien :**
select col1, col2 from table1, table2;
. **jointure :**
select col1, col2 from t1, t2 where t1.col1 = t2.col2;
select col1, col2 from t1 [inner] join t2 on t1.col1 = t2.col2;
select col1, col2 from t1 [{full, left, right} outer] join t2 on t1.col1 = t2.col2;
. **sous-requêtes :**
- derrière from
select a.col1, a.col2, b.col3 from (select col1, col2 from T1) a, (select col3 from T2) b;
- imbriquée
select n.note, n.noel from notes n where n.noel = (select e.noel from eleves e where e.nom = 'dupont');
- corrélées
select noemp, nom, serv from emp e1 where e1.sal > (select avg(e2.sal) from emp e2 where e2.serv = e1.serv);

Le mois prochains nous verrons l'utilisation de MySQL avec PHP.

Pascal Pignard, mars, avril, août-octobre 2004.