Mémo TECFA

version 0.9

Patrick Jermann

TECFA,
Faculté de Psychologie et des Sciences de l'Education,
Université de Genève

9 route de Drize, CH-1227 Carouge
Tél.: +41 22 70 9694

courrier électronique: Patrick Jermann@tecfa.unige.ch
MOO: tecfamoo.unige.ch 7777 (aka Colin)
WWW: http://tecfa.unige.ch/tecfa-people/jermann.html

1. Administration du serveur mySQL

mySQL est un **serveur** de bases de données, c'est à dire que plusieurs bases de données sont gérées par un seul et même programme. Toute machine UNIX peut héberger un serveur mySQL.

Une base de donnée est composée d'une ou de plusieurs **tables** qui contiennent des enregistrements. Ces enregistrements sont composés de **champs**.

Ainsi, à TECFA nous avons deux serveurs. Le permier réside sur *tecfasun1.unige.ch* et sert de base de données principale. Le second réside sur *tecfasun2.unige.ch* et sert de base de données tampon. Les nouvelles soumissions des utilisateurs y sont stockées en attendant l'approbation d'un administrateur pour figurer dans la base de données principale.

1-1 Lancer le serveur

Si le serveur est déjà actif, le système répond 'A mysqld process already exists'. Les connexions au serveur passent par la porte 3306.

1. Se logger en tant que *root* sur la machine qui fait serveur.
2. Exécuter la commande: /comm/soft/bin/safe_mysqld --log &.
   L'option --log permet d'enregistrer les messages d'erreur ainsi que toutes les requêtes qui sont faites au serveur. Le fichier log /comm2/soft/mysql/var/tecfasun1.log est créé automatiquement s'il n'existe pas encore.

1-2 Arrêter le serveur

1. Se logger en tant que *root* sur la machine qui fait serveur (tecfasun1.unige.ch).
2. Exécuter la commande: /comm/soft/bin/mysqladmin shutdown

(Syntaxe **mysqladmin** : 4-1)

1-3 Créer une base de données

1. Se connecter sur tecfasun1 comme *root*
2. Utiliser la commande mysqladmin pour créer la base de données. La même commande sert à effacer une base de données (Cf. 1-4) et à arrêter le serveur msql (Cf.1-2).

Exemple:

1. **mysqladmin** create colin_test

(Syntaxe **mysqladmin** : 4-1)

1-4 Effacer une base de données

Attention, effacer une base de données efface toutes les informations que vous y aviez mises. Il n'y a aucun moyen pour récupérer les données !

Exemple:

1. **mysqladmin** drop colin_test

(Syntaxe **mysqladmin** : 4-1)

1-5 Backup d'une base de données

Pour chaque base de données, mySQL crée un répertoire dans /comm2/soft/mysql/var/. Chaque table d'une base de données est stockée dans trois fichiers qui portent les extensions <nom_table>.frm <nom_table>.ISD et <nom_table>.ISM. Normalement ces fichiers sont backupés avec le reste du système.

mysqldump exporte la structure d'une ou de plusieurs tables et les données qu'elles contiennent sous la forme de requêtes SQL. Cette commande est particulièrement adaptée aux sauvegardes destinées à être réutilisées par mySQL ou par un autre système qui comprend SQL (mSQL, ACCESS, ORACLE, etc.).

Utilisée telle quelle, cette commande imprime à l'écran le contenu d'une base de données ou d'une table. Pour récupérer ce résultat nous utilisons une **redirection** qui permet d'écrire l'output d'une commande dans un fichier. La redirection s'exprime avec le signe >.

Exemples:

1. **mysqldump** studio > /home/jermann/imprime/studio-all.backup
   Crée un fichier studio.backup qui contient les définitions des tables de la base de données studio ainsi que celles des enregistrements
2. **mysqldump** -d studio > /home/jermann/imprime/studio-tables.backup
   Ne sauve que les définitions des tables
3. **mysqldump** -tv studio > /home/jermann/imprime/studio-records.backup
   Ne sauve que les enregistrements et imprime des messages sur l'avancement du backup.

(Syntaxe **mysqldump** : 4-2)

1-6 Restoration d'un backup

Après avoir utilisé msqldump pour sauvegarder nos données il suffit de soumettre le fichier de backup (qui est en fait un fichier de commandes SQL) à la commande mysql. (Voir aussi "Soumettre le fichier de commande à mSQL" à la page 9)

Exemple:

1. **mysql** studio < studio.backup

(Syntaxe **mysql** : 4-3)

1-7 Gestion des permissons

A FAIRE ... Dur dur ...

2. Opérations de base

2-1 Lister les bases de données, les tables et les champs

La commande mysqlshow permet lister les bases de données qui résident sur une machine. Le paramètre -h permet de spécifier le nom de la machine qui nous intéresse. Dans l'exemple ci-dessous, l'utilisateur est connecté à la machine *tecfasun2* mais liste les bases de données sur la machine *tecfasun1*.

(Syntaxe **mysqlshow** : 4-4)

Exemple:

1. **mysqlshow** -h tecfasun1
   L'utilisateur est connecté à la machine tecfasun2 mais liste les bases de données sur la machine tecfasun1.

```
+-----------+
| Databases |
+-----------+
| mysql     |
| pnr33     |
| studio    |
| tecfa     |
| test      |
+-----------+
```

Lorsqu'on ajoute en argument le nom d'une base de données, mysqlshow affiche le nom des tables qui composent celle-ci. Dans l'exemple ci-dessous, la base de données *studio* contient quatre tables.

Exemple:

1. **mysqlshow** -h tecfasun1 studio

```
Database: studio
+---------+
| Tables  |
+---------+
| auth    |
| phase_1 |
| phase_2 |
| phase_3 |
+---------+
```

De même, en ajoutant le nom d'une table à la commande, mysqlshow liste la définition des champs de la table. Chaque champ possède un nom (Field), un type et une longueur (Type). En plus, trois marqueurs permettent de spécifier si la valeur d'un champ peut être vide (Not Null) s'il s'agit d'une clé primaire (Key) et finalement, il est possible de définir une valeur par défaut (Default). La dernière colonne (Extra) indique si certains champs sont auto-incrémentés.

Exemple:

1. **mysqlshow** -h tecfasun1 studio phase_1
   Nous voyons la liste des champs pour la table phase_1. Six types de champs apparaissent. Les types char, varchar et text définissent des valeurs textuelles. Il n'est pas possible d'effectuer des recherches par mot-clé sur un champ de type text. Par contre, il est possible d'y stocker un nombre illimité de caractères. Inversément, il est possible d'effectuer une recherche par mots-clé sur le contenu d'un champ char ou varchar. Par contre, il n'est pas possible d'y stocker un nombre de caractères supérieur à 255.

Ainsi, dans notre exemple, les recherches par mot-clé ne pourront porter que sur les champs suivants: authors, parents, title, thumb_url, format et outil_realisation.

Les types de champs date et time permettent de stocker des dates (par exemple, 01-Feb-1998) et des heures (par exemple, 19:45:00)

Le type int permet de stocker des nombres entiers. Le champ id est un entier et en plus sert de clé de recherche primaire (id ne peut avoir deux fois la même valeur)

```
Database: studio  Table: phase_1  Rows: 1
+--------------------------+--------------+------+-----+---------+------+
| Field                    | Type         | Null | Key | Default | Extra|
+--------------------------+--------------+------+-----+---------+------+
| id                     | int(11)     |     | PRI | 0    |auto_increment|
| when_date                | date         | YES  |     |         |      |
| when_time                | time         | YES  |     |         |      |
| authors                  | varchar(25)  |      |     |         |      |
| parents                  | varchar(25)  | YES  |     |         |      |
| title                    | varchar(80)  |      |     |         |      |
| description              | text         |      |     |         |      |
| thumb_url                | varchar(255) |      |     |         |      |
| public_cible             | text         | YES  |     |         |      |
| couts_benefices          | text         | YES  |     |         |      |
| couts_benefices_comments | text         | YES  |     |         |      |
| exists                   | char(3)      | YES  |     |         |      |
| format                   | varchar(255) | YES  |     |         |      |
| outil_realisation        | varchar(255) | YES  |     |         |      |
+--------------------------+--------------+------+-----+---------+------+
```

(Syntaxe **mysqlshow** : 4-4)

2-2 L'interface en-ligne à mySQL

L'interface en ligne permet de soumettre des requêtes SQL au serveur qui nous renvoie alors le résultat de la requête. La commande mysql sert à ouvrir une connexion vers le serveur mySQL. Les arguments de la commande sont le nom de la machine sur laquelle le serveur mySQL est installé et le nom de la base de données à laquelle on veut se connecter.

(Syntaxe **mysql** : 4-3)

Exemple:

1. **mysql** -h tecfasun1 studio

```
Welcome to the MySQL monitor.  Commands end with ; or „.
Your MySQL connection id is 24 to server version: 3.21.24-gamma-log

Type 'help' for help.

mysql>
```

On peut dès lors utiliser des commandes SQL pour effectuer des requêtes et utiliser les codes suivants:

quit ferme la connexion
; ou \g envoie la requête

2-2.1 Exemple de requête SELECT avec l'interface en ligne

Une description complète de la syntaxe des commandes SQL implémentées dans mySQL est accessible à l'URL: http://tecfa.unige.ch/guides/mysql/ref-3.21.22-beta/manual.html#Syntax

Dans le premier exemple, nous sélectionnons les champs `id, when_date, when_time, title et authors` de la table `phase_3`. Notez le ; à la fin de la ligne qui indique la fin de la requête. Nous n'avons pas spécifié de contraintes sur le type d'enregistrement que nous souhaitons afficher, c'est pourquoi tous les 5 enregistrements sont retournés par mSQL.

```
mysql> SELECT id, when_date, when_time, title, authors FROM phase_3 ;
+----+------------+-----------+-------------------+---------+
| id | when_date  | when_time | title             | authors |
+----+------------+-----------+-------------------+---------+
|  1 | 0000-00-00 | 19:45:00  | Super Cow         | 1       |
|  2 | 0000-00-00 | 19:55:00  | Beautiful affiche | 1:2     |
|  3 | 0000-00-00 | 12:45:00  | Educational Cow   | 2       |
|  4 | 0000-00-00 | 12:45:00  | Little hamster    | 2:1     |
|  5 | 0000-00-00 | 09:25:00  | Barre TECFA       | 2       |
+----+------------+-----------+-------------------+---------+
5 rows in set (0.01 sec)
```

Si nous voulons connaître les numéros des objets qui ont été créés par l'auteur numéro '1' et qui contiennent le mot 'cow' dans le titre, les mots-clé WHERE, LIKE et AND permettent de spécifier cette contrainte. L'opérateur LIKE permet de soumettre une expression régulière à mSQL. Les % sont des jokers et remplacent n'importe quel caractère.

```
mysql> SELECT id, when_date, when_time, title, authors FROM phase_3
    -> WHERE authors='1' AND title LIKE '%cow%'
    -> ;
+----+------------+-----------+-----------+---------+
| id | when_date  | when_time | title     | authors |
+----+------------+-----------+-----------+---------+
|  1 | 0000-00-00 | 19:45:00  | Super Cow | 1       |
+----+------------+-----------+-----------+---------+
1 row in set (0.00 sec)
```

Voici la requête pour connaître le titre et la description de l'objet numéro 5. Notez que le chiffre 2 n'est pas entre guillemets parce que c'est un entier.

```
mysql> SELECT title, description FROM phase_3 WHERE id=2;
+-------------------+-------------+
| title             | description |
+-------------------+-------------+
| Beautiful affiche |             |
+-------------------+-------------+
1 row in set (0.00 sec)
```

2-3 Créer des tables

Mécanisme de redirection

Pour des opérations comme la création d'une nouvelle table, une requête SQL peut occuper plusieurs lignes. Il est malaisé de les saisir une à une à l'invite `mySQL>`. C'est pourquoi il est nécessaire d'utiliser un **fichier de commande**, qui consiste en une ou plusieurs requêtes SQL. Celui-ci est ensuite soumis en une fois au serveur en utilisant le mécanisme de **redirection**.

Le signe < est utilisé pour 'donner à manger' à une commande. Dans l'exemple fictif qui suit, la commande `mouline` va prendre comme entrée le fichier de commande `mes_commandes`:

```
mouline < mes_commandes
```

Le mécanisme de redirection est également utilisé pour récupérer le résultat d'une commande dans un fichier. La redirection s'exprime avec le signe >. L'exemple fictif ci-dessous sauve le résultat de la commande `mouline` dans un fichier `resultats.txt`. Voir "Backup d'une base de données" à la page 4 pour un exemple concret.

```
mouline > resultats.txt
```

On crée des tables en écrivant une requête CREATE TABLE dans un fichier de commande et en appelant mysql avec ce fichier en argument. La base de données pour laquelle on veut créer une table doit exister! Si ce n'est pas le cas, voir "Créer une base de données" à la page 3.

2-3.1 Fichier de commande

Voici un exemple de fichier de commandes qui sert à créer une table. (http://agora.unige.ch/mysql/table-exemple.mysql). Un éditeur de texte minimal suffit pour composer un tel fichier (Notepad, Simple Text). Les lignes précédées de # sont des commentaires et ne produisent aucun effet. Chaque champ est défini en donnant d'abord son nom puis son type.

```
# MySQL dump 4.0
#
# Host: tecfa    Database: test
#--------------------------------------------------------

#
# Table structure for table 'phase_3'
#

DROP TABLE phase_3;
CREATE TABLE phase_3 (
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  when_date DATE,
  when_time TIME,
  authors VARCHAR(25) NOT NULL,
  parents VARCHAR(25),
  title VARCHAR(80) NOT NULL,
  description TEXT NOT NULL,
  thumb_url VARCHAR(255) NOT NULL
) ;
```

Notez que la première commande du fichier est '`DROP TABLE phase_3`'. Cette commande efface la table `phase_3` si elle existe déjà. L'exécution de la commande DROP provoque une erreur si la table visée n'existe pas. Dans ce cas il suffit de mettre cette ligne en commentaire (ajouter un # devant).

Les types de champ possibles sont décrits en détail dans le manuel de référence : http://tecfa.unige.ch/guides/mysql/ref-3.21.22-beta/manual.html#Column types. Voici néanmoins les types les plus utilisés:

Table 1: Types de champs

Nom	Description	Taille
CHAR(M) [binary]	Une chaîne de caractères de taille fixe qui utilise des espaces pour remplir le champ jusqu'à concurrence de M caractères. La taille M est comprise entre 1 et 255 caractères. Tous les espaces supplémentaires sont enlevés lorsqu'on récupère la valeur. Le tri et la comparaison de valeurs se fait indifféremment de la casse a moins que le mot-clé 'binary' soit spécifié.	M
VARCHAR(M) [binary]	Une chaîne de caractères de taille variable qui est stockée avec sa longueur (L+1). La taille M maximale est comprise entre 1 et 255 caractères. Le tri et la comparaison de valeurs se fait indifféremment de la casse a moins que le mot-clé 'binary' soit spécifié.	L+1
TEXT and BLOB	Un TEXT/BLOB avec une longueur maximale de 65535 caractères. Ces champs ne peuvent pas servir de clé d'index.	L+2

Table 1: Types de champs

Nom	Description	Taille
INT[(D)] [UNSIGNED] [ZEROFILL]	Un entier normal. Intervalle signé: -2147483648 à 2147483647. Intervalle non-signé: 0 à 4294967295.	4
DATE	Un type permettant de stocker des dates. Utilise la syntaxe "YYYY-MM-DD", mais peut être mis a jour par un nombre ou une chaîne de caractères. Comprend au moins les syntaxes suivantes: 'YY-MM-DD', 'YYYY-MM-DD', 'YYMMDD', 'YYMM', 'YY'. Intervalle de 0000-00-00 à 9999-12-31.	4
TIME	Un type permettant de stocker des heures.Utilise la syntaxe "HH:MM:SS", mais peut être mis à jour avec un nombre ou une chaîne de caractères. Comprend au moins les syntaxes suivantes: 'HH:MM:DD, 'HHMMDD', 'HHMM', 'HH'.	3
ENUM('value','value2',...)	Une chaîne de caractères qui peut avoir une valeur parmi un ensemble de valeurs spécifiques. Par exemple la colonne test ENUM("one","two", "three") peut avoir une des valeurs suivantes: • "one" • "two" • "three"	1 ou 2
SET('value','value2',...)	Une châine de caractères qui peut avoir une ou plusieurs valeurs d'un ensemble de valeurs possibles. Par exemple, la colonne test SET("one","two") peut avoir une des valeurs suivantes: • "" • "one" • "two" • "one,two"	1-8

2-3.2 Soumettre le fichier de commande à mSQL

Pour ne pas abîmer les bases de données installées sur nos machines, une base de donnée nommée 'test' a été créée sur la machine tecfasun1. Utilisez la pour faire vos essais.

Une fois le fichier de commande terminé, lancer la commande mysql avec une redirection. On donne en argument, la machine sur laquelle se trouve le serveur mSQL (-h tecfasun1), le nom de la base de données (test) et le fichier contenant les commandes SQL:

Exemple

1. mysql -h tecfasun1 test < /comm/tecfa/www/swiss-edu/mysql/table-exemple.mysql
   Crée une table nommée phase_3 dans la base de données test sur la machine tecfasun1. Les commandes SQL qui définissent la table sont dans le fichier table-exemple.mysql.

Remarque:

Si vous obtenez des messages, d'erreur c'est que votre fichier de commande comporte une faute de syntaxe, ou que la base de données n'existe pas, ou encore que vous n'avez pas la permission de modifier cette base de données.

2-4 Effacer des tables

L’effacement d’une table est plus simple que sa création et consiste en une requête DROP. Attention, avant d’effacer les informations contenues dans la table il est conseillé de faire un backup des données sensibles. (Voir “Backup d’une base de données” à la page 4).

Exemple:

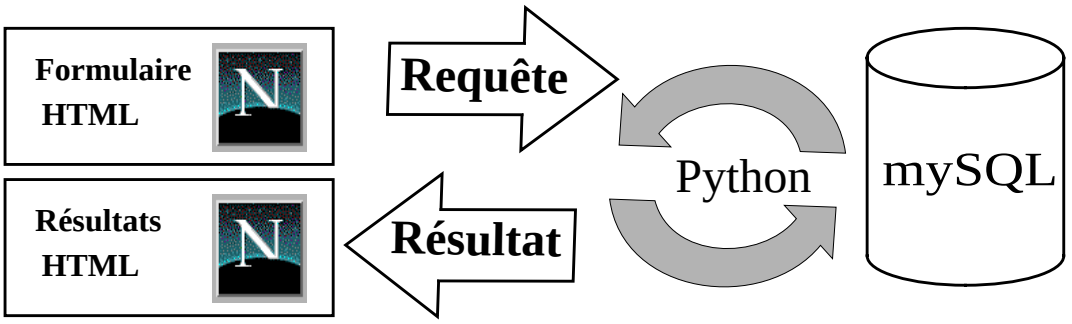
```
mysql> DROP TABLE phase_3 ;
Query OK, 0 rows affected (0.12 sec)
```

Effaçe la table ‘phase_3’ dans la base de données ‘test’ sur la machine ‘tecfasun1’. Toutes les données sont perdues.

3. Interface WWW à mySQL

L'interface WWW à mySQL repose sur un ensemble de scripts Python qui sont capables de transformer le contenu d'un formulaire HTML en une requête SQL, de la soumettre à mSQL et de formatter le résultat en HTML.

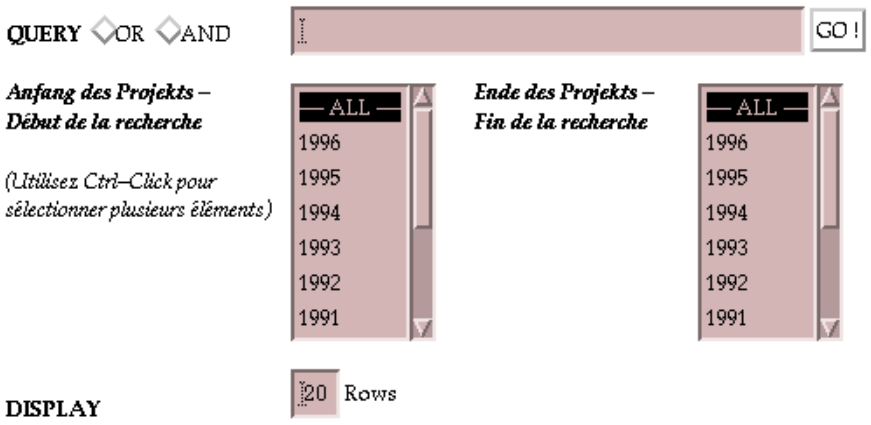
La majorité des interfaces WWW à nos bases de données repose sur deux scripts. Un premier script soumet une requête et retourne une liste d'enregistrements. Le second permet d'afficher en entier les informations contenues dans un seul enregistrement.



3-1 Formulaire HTML

Nous prenons comme exemple le formulaire qui permet d'interroger la base de données 'Information sur la recherche éducationnelle' qui se trouve à l'adresse suivante: http://agora.unige.ch/csre/info/msql/index.html.

Voici le formulaire tel qu'il apparaît dans Netscape. L'utilisateur peut entrer des mots-clé, choisir des items dans des listes à choix multiple et spécifier le nombre d'enregistrements qu'il souhaite voir sur l'écran.



Voici le code HTML (les tags relatifs à la mise en forme du formulaire dans un tableau ont été enlevés pour plus de lisibilité).

Les **champs obligatoires** dans les formulaires sont décrits ci-dessous. Soit l'utilisateur peut les définir lui-même, soit vous les spécifiez en les définissant avec un tag caché `<input type=hidden name=nom_du_champ value=votre_valeur>`. (Les champs `maxrow` et `output_type` sont des champs cachés dans l'exemple ci-dessous):

- **action** indique l'URL du script qui va traiter le formulaire.
- **boolean** peut prendre soit la valeur " OR " soit " AND " (espaces!). Ce paramètre spécifie le connecteur logique qui relie les mots-clé spécifiés dans le champ query.
- **query** contient les mots-clé entrés par l'utilisateur.
- maxrow défini le nombre d'enregistrements affichés sur un écran de résultats. Si le nombre d'enregistrements retournés par mSQL est supérieur à ce nombre, un bouton 'Show next records' permet de voir les maxrow enregistrements suivants.
- **output_type** peut prendre soit la valeur LIST soit la valeur TABULAR. Lorsque LIST est sélectionné, les champs des enregistrements sont affichés les uns après les autres. Lorsque TABULAR est sélectionné, les champs des enregistrements sont affichés dans une table HTML.

Les champs `vonChoice` et `bisChoice` sont spécifiques à notre exemple. Ils font référence à des listes à choix multiples. Voir la section "Choix multiples HTML" à la page 13.

Exemple:

```
<form name="queryForm" action="http://tecfa.unige.ch/cgi-bin/agora/info-
ed-96/info-ed-query-result.py" method=post>

<strong>QUERY</strong>
<input type=radio name=boolean value=" OR ">OR
<input type=radio name=boolean value=" AND ">AND

<input type=text size=40 name=query>
<input type=button value="GO !" onClick="makeQuery(this.form)">

<strong><i>Anfang des Projekts - <br>Début de la recherche </i></
strong><p><i>(Utilisez Ctrl-Click pour <br>sélectionner plusieurs
éléments)</i>

<select name=vonChoice multiple size=7>
<option selected>-- ALL --
<option>1996
<option>1995
etc..
<option>1991
<option>1990
<option>1989
</select>

<strong><i>Ende des Projekts - <br>Fin de la recherche</i></strong>

<select name=bisChoice multiple size=7>
<option selected>-- ALL --
<option>1996
<option>1995
<option>1994
etc...
<option>1990
<option>1989
</select>

<strong>DISPLAY</strong>
<input type=text name=maxrow size=2 value=20> Rows
<input type=hidden name=output_type value="LIST">
</form>
```

3-2 Choix multiples HTML

Lorsqu'un champ de la base de données contient des champs qui prennent un ensemble fini de valeurs, il est possible d'utiliser une liste à choix multiples pour effectuer des recherches.

Dans notre exemple, les champs `von` et `bis` contiennent les dates du début et de la fin d'une recherche: 1989, 1990, etc... La liste à choix multiples consiste en un tag SELECT et une série d'entrées spécifiées à l'aide du tag OPTION.

```
<select name=vonChoice multiple size=7>
<option selected>-- ALL --
<option>1996
<option>1995
<option>1994
<option>1993
<option>1992
<option>1991
<option>1990
<option>1989
</select>
```

Pour que Python puisse interpréter le choix de l'utilisateur il faut respecter **deux contraintes**:

- le nom du champ est composé de deux parties: **`<nom_du_champ>Choice`**, dans notre exemple: **`vonChoice`**. <nom_du_champ> correspond au nom du champ en question dans la base de données mySQL.
- La première option doit s'appeller '**`-- ALL --`**' (avec un espace après -- et un espace après ALL). Elle est sélectionnée par défaut. Lorsque ALL est sélectionné le champ en question (von) est ignoré dans la requête. Ce n'est que lorsque une des options restantes est sélectionnées qu'elle est prise en compte,

3-3 Configuration de base d'un script Python

La configuration des scripts Python est simple et permet d'influer sur l'interface WWW et sur le mécanisme de recherche. ATTENTION, faites une **copie de sauvegarde** du script avant de le modifier.

Les scripts utilisés par l'interface WWW se trouvent dans les sous-répertoires de `/comm2/soft/httpd/cgi-bin/new-agora`. Les scripts qui servent d'interface aux différentes bases de données sont semblables et sont reconnaissables par les suffixes `query-result` et `full-display.`

Par exemple, dans le répertoire `/comm2/soft/httpd/cgi-bin/new-agora/info-ed-96/` se trouvent deux scripts:

- `info-ed-query-result.py*` transforme le contenu du formulaire HTML, soumet la requête à mSQL et renvoie une liste d'enregistrements.
- `info-ed-full-display.py*` affiche en entier un enregistrement

Notation

En Python, les chaînes de caractère sont entourées de guillemets : “ ou ‘

```
dbname = “csre-info”
```

Pour écrire une chaîne de caractères sur plusieurs lignes on utilise les triple guillemets:

```
header = “”” Ceci est le
titre de mon
document
”””
```

Les listes sont entourées de parenthèses carrées et leurs éléments sont séparés par des virgules:

```
multi_choice= [“SpracheChoice”,”vonChoice”,”bisChoice”]
```

3-3.1 Script ‘query-result’

Cette section présente les paramètres qui règlent le fonctionnement du script. Le script qui correspond à l’exemple se trouve ici: `/comm2/soft/httpd/cgi-bin/agora/info-ed-96/info-ed-query-result.py`. La zone des paramètres se trouve au début du fichier et se termine par le signe:

```
... paramètres ....
                                    #
                                    ##
                                    # #
####################################  #
#                                     #
# No further changes below this line ...
#                                     #
####################################  #
                                    # #
                                    ##
                                    #
```

Paramètres de base

Au début du fichier, une série de commentaires (signalés par #) donne quelques indications que nous complétons ci-après.

- *host*: indique sur quelle machine réside la base de données qui est intérrogée par le script.
- *dbname*: indique le nom de la base de données que le script interroge.
- *table*: indique le nom de la table qui est concernée par le script
- *start_page*: l’URL du formulaire qui appelle ce script
- *addition_page*: l’URL du formulaire qui permet d’ajouter un enregistrement
- *full_script*: l’URL du script qui affiche un enregistrement au complet
- *query_script*: l’URL de ce script
- *temp_path*: le répertoire qui héberge les fichiers temporaires produits par le script
- *logfile*: le chemin et le nom du fichier qui contient une trace des requêtes soumises à ce script

host=’tecfa.unige.ch’
dbname = “csre-info”
table = “info_ed”
start_page = “http://agora.unige.ch/csre/info/msql/index.html”
addition_page=”http://agora.unige.ch/csre/info/msql/info_ed-addition-form.html”
full_script = “http://tecfa.unige.ch/cgi-bin/new-agora/info-ed-96/info-ed-full-display.py”
query_script = “http://tecfa.unige.ch/cgi-bin/new-agora/info-ed-96/info-ed-query-result.py”

temp_path= ‘/temp/msql.temp/’
logfile=’/temp/msql.temp/info-ed-log’

Paramètres pour la requête

Ces paramètres permettent de régler l'étendue de la recherche et les champs à afficher en résultat.

- *select_list*: donne la liste des champs qui seront présents dans les résultats du script. Le premier champ doit toujours être `_rowid.`
- *where_list*: donne la liste des champs qui sont examinés en cas de recherche par mots-clé
- *multi_choice*: énumère les listes à choix multiple (cf. 3-2 "Choix multiples HTML")
- *unique_list*: liste des champs dont la valeur ne doit apparaître qu'une seule fois dans les résultats. (Utile lorsqu'une information est redondante à l'intérieur des données).

select_list = ["id","Titel","Titel_ubersetzt","Deskriptoren","Descripteurs"]
where_fields = ["Titel","Titel_ubersetzt", "Deskriptoren","Descripteurs"]
multi_choice= ["vonChoice","bisChoice"]
unique_list=[]

Choix multiples

Ces paramètres sont les plus difficiles à fixer. Nous avons vus le principe de la liste à choix multiples dans la section 3-2 "Choix multiples HTML".

Imaginons que nous avons un champ 'langue' dans notre base de données. Lors de la saisie, nous avons donné des codes aux langues. Ainsi, une valeur 1 signifie 'allemand', une valeur 2 signifie 'français' et une valeur 3 signifie 'italien'. Nous ne pouvons pas demander à l'utilisateur de choisir entre trois numéros (il ne sait pas quelle est leur signification) !

Pour remédier à ce problème nous définissons des dictionnaires qui font correspondre une valeur entrée par l'utilisateur à une valeur à utiliser pour la recherche dans la base de données. Le script va faire correspondre la valeur choisie par l'utilisateur (français) dans le formulaire et la valeur qu'il doit rechercher dans la base de données (2).

Le nom du dictionnaire est composé de deux parties: `<nom_du_champ>Dict`. Dans notre exemple : `langueDict`. Voici le dictionnaire:

```
langueDict = {'allemand':'1',
              'français':'2',
              'italien':'3'
             }
```

- Les dictionnaires `vonDict` et `bisDict` correspondent aux éléments de formulaire `vonChoice` et `bisChoice` (cf. 3-2 "Choix multiples HTML"). Or, les valeurs des dates sont stockées sous une forme humainement compréhensible. Ainsi, dans cet exemple les dictionnaires ne servent pas à faire une transformation mais mettent en correspondance des valeurs identiques.
- multi_boolean: règle le connecteur logique lorsque plus d'une valeur est sélectionnée dans une liste à choix multiples. Les valeurs possibles de ce paramètre sont " AND " et " OR ". (Il faut respecter les espaces avant et après).

multi_boolean = " AND "

```
vonDict = {'1996':'1996',
           '1995':'1995',
           '1994':'1994',
           '1993':'1993',
           '1992':'1992',
           '1991':'1991',
           '1990':'1990',
           '1989':'1989',
           '1988':'1988',
           '1987':'1987',
           }
```

```
bisDict = {'1996':'1996',
           '1995':'1995',
           '1994':'1994',
           '1993':'1993',
           '1992':'1992',
```

```
'1991':'1991',
'1990':'1990',
'1989':'1989',
'1988':'1988',
'1987':'1987',
}
```

Contacts

- *bug_report*: l'adresse E-mail de la personne à conteacter en cas de problèmes.
- *contact_address:* l'adresse E-mail de la personne à contacter pour des questions d'ordre général.

bug_report = "Patrick.Jermann@tecfa.unige.ch"
contact_address = "skbf-csre@ping.ch"

Booléens

- *hide_rowid*: peut prendre les valeurs 0 ou 1. Lorsque hide_rowid est égal à 0 le numéro interne des enregistrements est affiché dans les résultats. (En général on laisse hide_rowid à 1).
- *debug*: peut prendre les valeurs 0 ou 1. Lorsque debug est égal à 1 des résultats intermédiaires sont ajoutés aux enregistrements affichés par le script. Ce mode permet de chercher une erreur éventuelle dans le programme en inspectant les variables.
- *allow_empty_query*: peut prendre les valeurs 0 ou 1. Si la valeur est 1, les utilisateurs peuvent lancer une recherche sans avoir spécifié de mot clé. Tous les enregistrements de la table sont alors affichés. Si la valeur est 0, un message d'erreur est présenté à l'utilisateur qui ne spécifie pas de mots-clé dans son formulaire.
- *enable_add*: ajoute un bouton 'Add a record' qui affiche un formulaire de saisie où les utilisateurs peuvent eux-même entrer des enregistrements.

hide_rowid = 1
debug = 0
allow_empty_query = 1
enable_add = 0

Header et Footer HTML

- *page_header*: du code HTML affiché en haut de la page produite par le script
- *page_footer*: du code HTML affiché en bas de la page produite par le script

page_header = """
<table border=0>
<tr>
<td><img alt=\"CSRE logo\" width=76 height=80 border=0 src=\"http://agora.unige.ch/csre/images/t_CSRE_logo.gif\">
<td>

Schweizerische Koorinationsstelle für Bildungsforschung
Centre suisse de coordination pour la recherche en éducation
Centro svizzero di coordinamento della ricerca educativa
Swiss co-ordination center for research in education

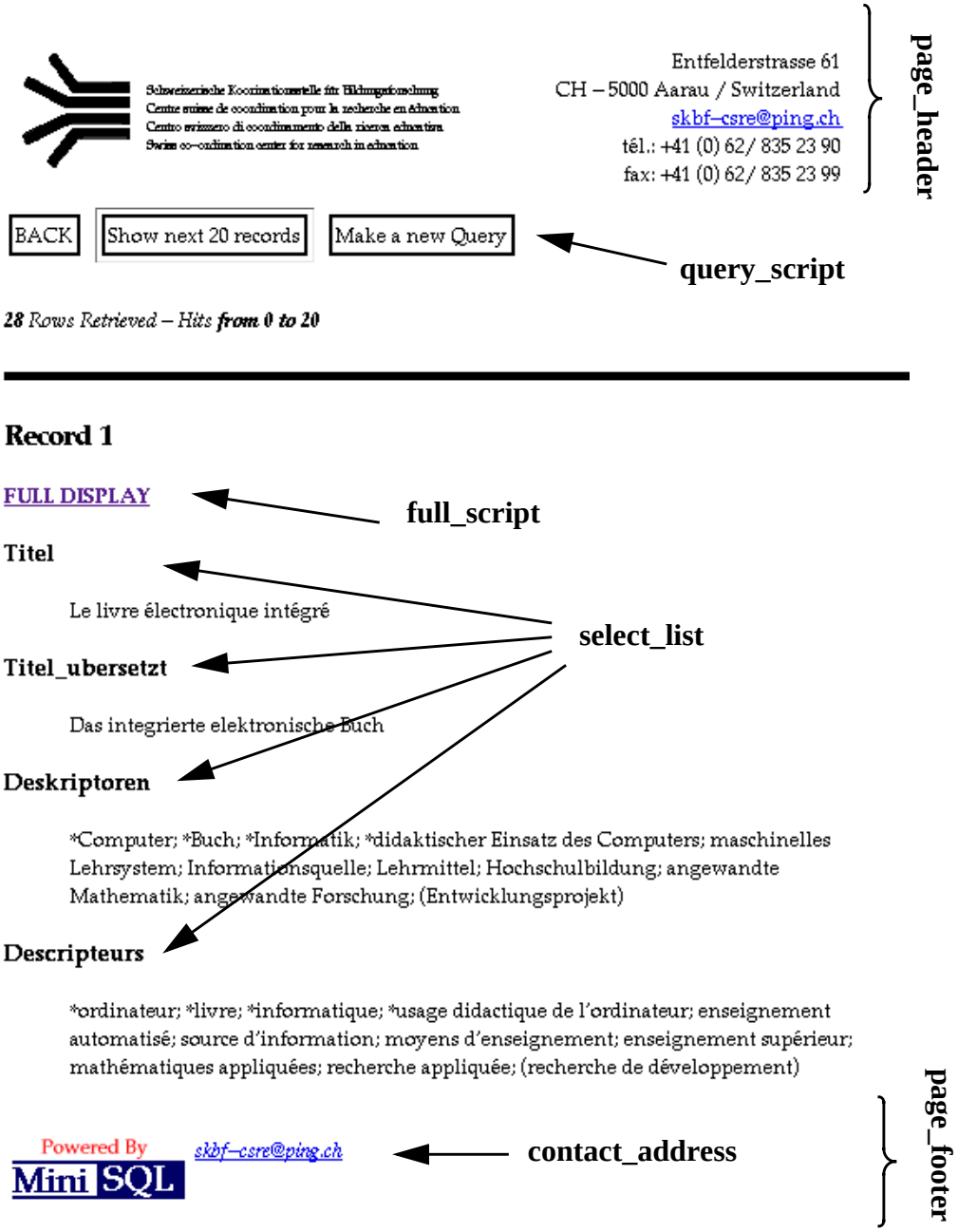
<td width=50>

<td align=right>
Entfelderstrasse 61
CH - 5000 Aarau / Switzerland
skbf-csre@ping.ch
tél.: +41 (0) 62/ 835 23 90
fax: +41 (0) 62/ 835 23 99
</table>
"""

page_footer= """
<p><img src=\"http://agora.unige.ch/images/msql-pwr.gif\" width=110 height=40 border=0 align=left>
<address> %s</address>

</body></html>
“““ % (contact_address,contact_address)

3-3.2 Output du script query-result



3-3.3 Script full-display

Le script full-display affiche en entier un seul enregistrement de la base de données. Le script qui correspond à l'exemple se trouve ici: `/comm2/soft/httpd/cgi-bin/agora/info-ed-96/info-ed-full-display.py`. La modification de ce script permet de changer l'apparence de la page HTML qu'il produit.

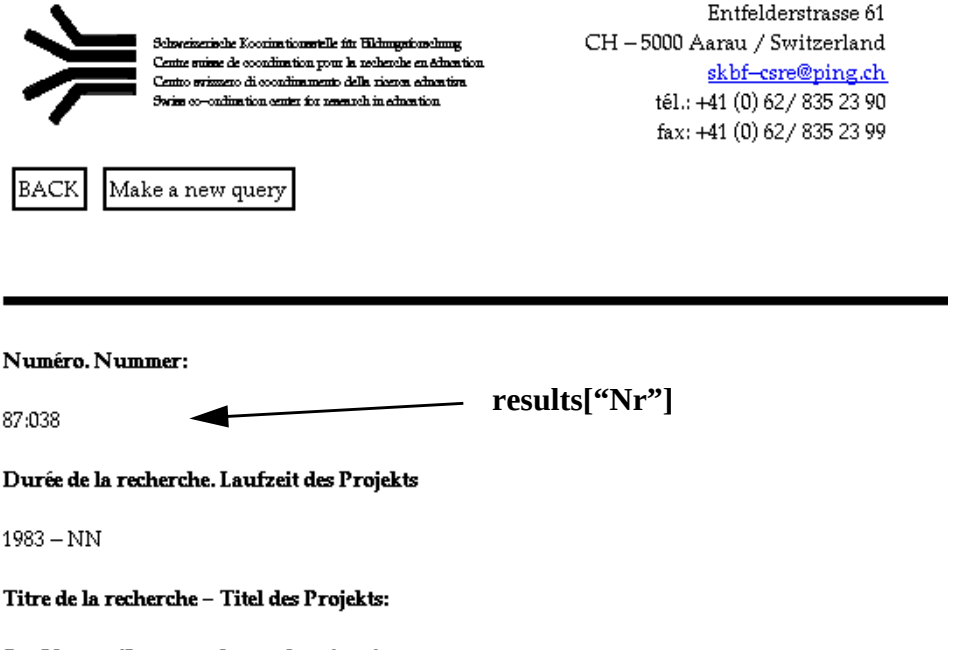
Au milieu du script se trouve une commande print qui produit le corps de la page.

Les %s sont remplacés dans l'ordre par les valeurs spécifiées à la fin de la commande print après le signe “““ %. Ainsi, le premier %s qui apparaît est remplacé par la valeur de `results["Nr"]`, le numéro de la recherche. Pour ajouter un hypothétique champ 'langue' il suffirait d'ajouter un %s dans le print et d'ajouter `results["langue"]` à la fin du print (at-

tention à le mettre à la bonne place).

```
print “““
<H4>Numéro. Nummer:</H4>
%s
<h4>Durée de la recherche. Laufzeit des Projekts</h4>
%s - %s
<h4>Titre de la recherche - Titel des Projekts:</h4>
<h2>%s</h2>
<h2>%s</h2>
<H4>Institution:</H4>
%s
<H4>Chercheurs. Bearbeiter des Projekts</H4>
%s
<H4>Brève description de la recherche:</H4>
%s
<H4>Kurzbeschreibung des Projekts:</H4>
%s
<H4>Descripteurs (EUDISED):</H4>
%s
<H4>Deskriptoren (EUDISED):</H4>
%s
<H4>Publications. Veröffentlichungen:</H4>
%s
<H4>Méthodes de recherche. Methodische Anlage des projekts:</H4>
%s
<H4>Délimitation géographique. Geographischer Raum:</H4>
%s
<H4>Type de recherche. Art des projekts:</H4>
%s
<H4>Mandataire de la recherche. Auftragsgeber des Projekts:</H4>
%s
<H4>Financement de la recherche. Finanzierung:</H4>
%s
““““ %(results[“Nr”],results[“von”],results[“bis”],results[“Titel”],results[“Titel_ubersetzt”],
results[“Institution”],results[“Forscher”],results[“Abstract_frz”],results[“Abstract_dt”],results[“Descripteurs”],results
[“Deskriptoren”],results[“Publikationen”],results[“Meth”],results[“Geo”],results[“Typ”],results[“Auftrag”],results[“F
inanzierung”])
```

3-3.4 Output du script full-display



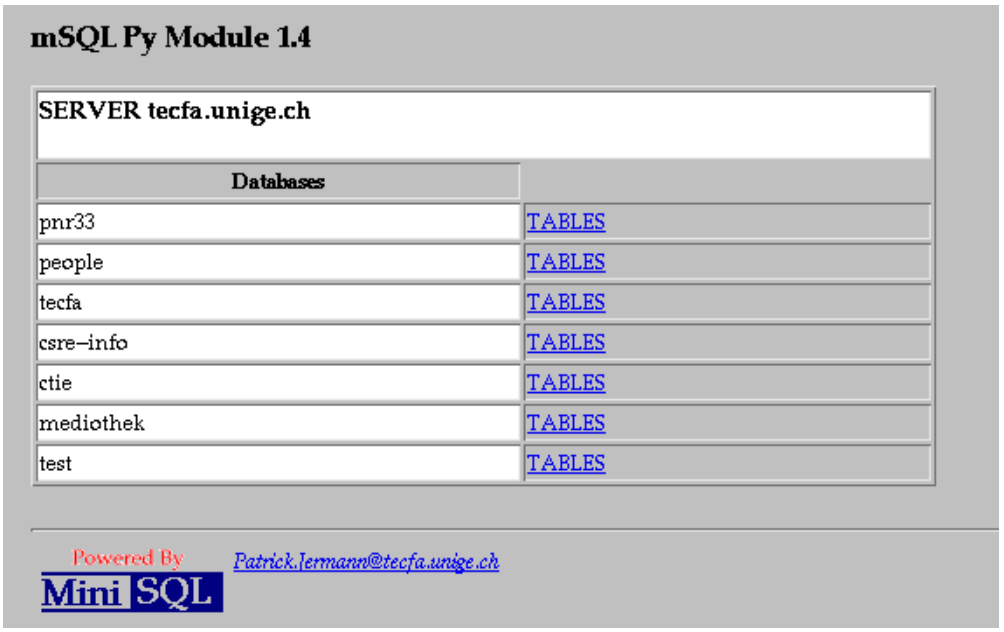
3-4 Interface d'Administration

Il existe une interface 'administrateur' qui permet de manipuler toutes les bases de données installées sur tecfasun1. Il est possible d'ajouter de nouveaux enregistrements, d'en modifier et d'en effacer. Cette page est protégée par un mot de passe.

L'URL pour y accéder: http://tecfa.unige.ch/cgi-bin/mysql/main.py

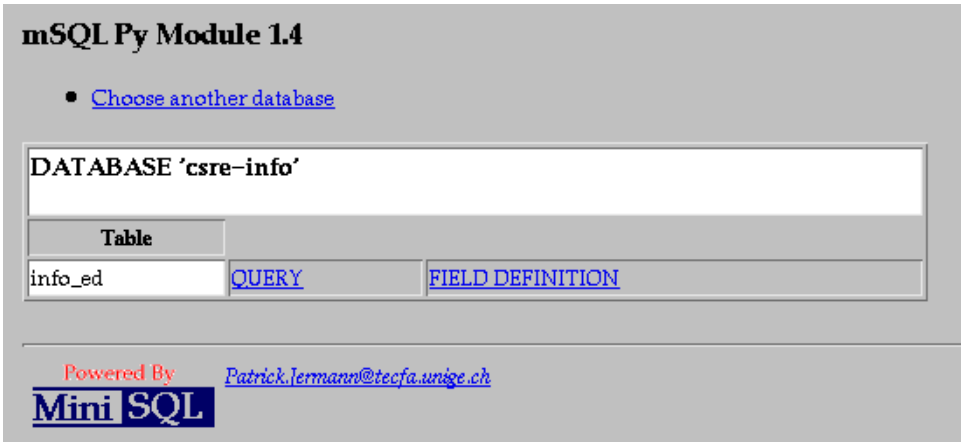
Page d'acceuil

Cette page permet de choisir la base de données qui nous intéresse. Cliquons sur csre-info à titre d'exemple.



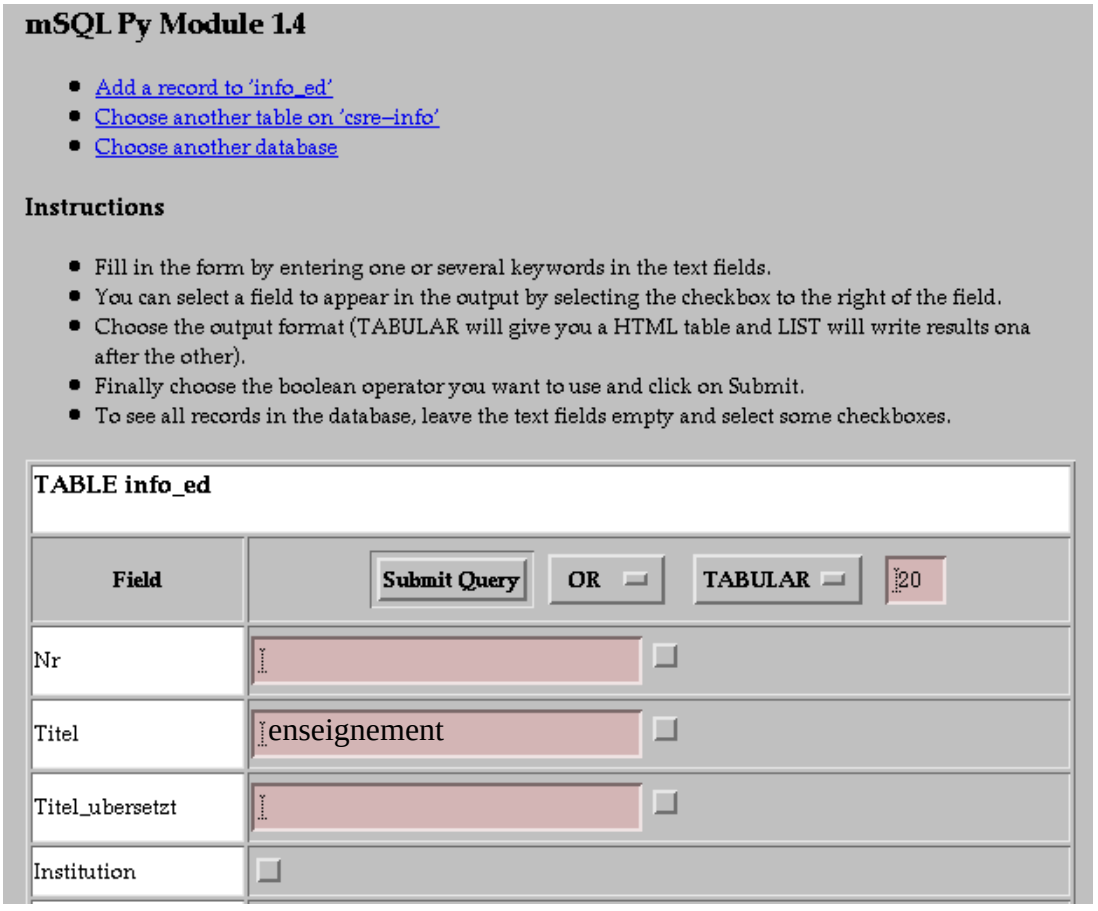
Page 'Table'

Cette page offre deux choix. En cliquant sur FIELD DEFINITION, les définitions des champs de la table sont affichés. (Cf. 2-1 "Lister les bases de données, les tables et les champs"). En cliquant sur QUERY un formulaire de recherche est affiché.



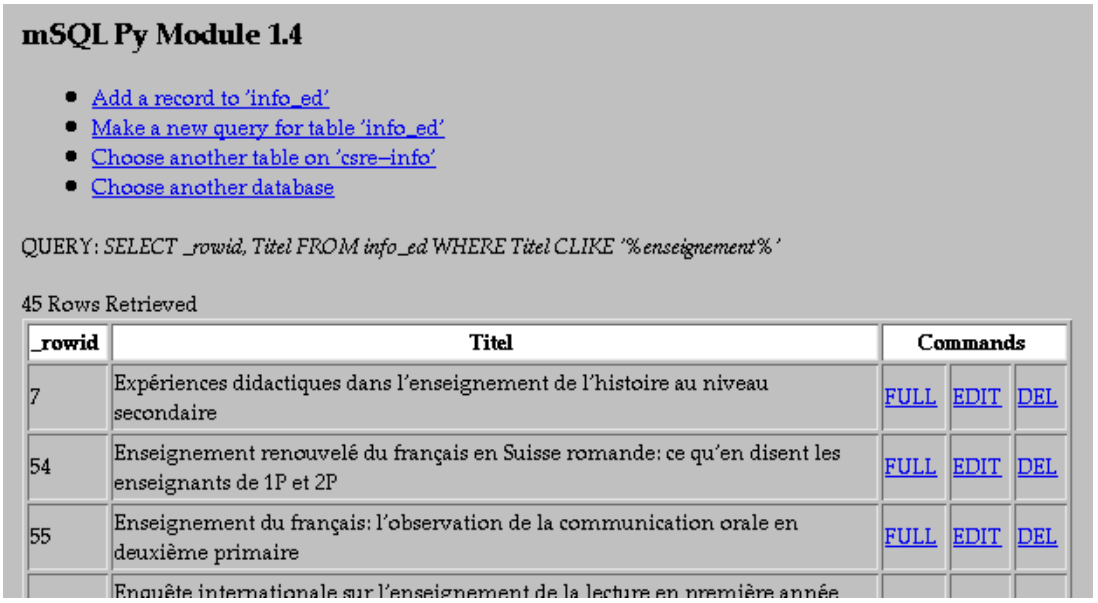
Page 'Query'

Outre un pointeur vers la pages 'Addition' d'un enregistrement présentée plus bas, cette page permet de composer de requêtes précises sur un ou plusieurs champs et de choisir précisément quels champs seront affichés en résultat. En entrant un mot-clé dans une zone de texte, on demande à mSQL de le rechercher dans un seul champ. Les champs sans zone de texte sont de type TEXT (Cf. 2-3 "Créer des tables"). Lorsque les checkbox à droite des zones de texte sont sélectionnés, les champs correspondants figurent dans les résultats. On peut en plus choisir le nombre d'enregistrements à afficher et le connecteur logique à utiliser.



Page 'Query result'

Voici la page qui suit la soumission du formulaire 'Query'. Pour chaque enregistrement qui correspond à la requête, trois commandes sont à disposition: FULL affiche l'enregistrement au comple, EDIT permet de le modifier et DEL l'efface (attention...). L'utilisation de ces pages ne devrait pas poser de problèmes et ne sont par conséquent pas décrites ici. (...probieren geht über studieren ;)



mSQL Py Module 1.4

- Add a record to 'info_ed'
- Make a new query for table 'info_ed'
- Choose another table on 'csre-info'
- Choose another database

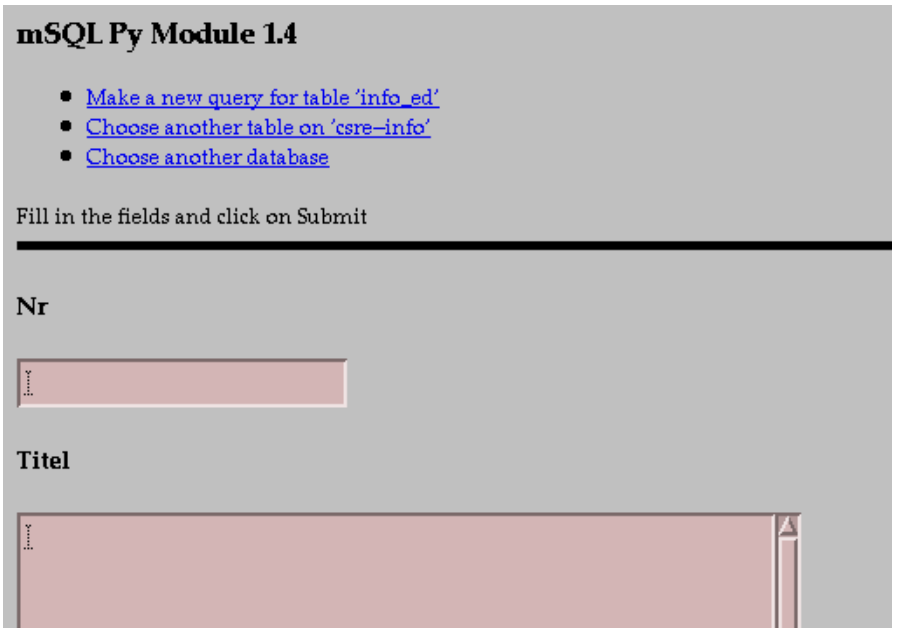
QUERY: *SELECT _rowid, Titel FROM info_ed WHERE Titel CLIKE '%enseignement%'*

45 Rows Retrieved

_rowid	Titel	Commands		
7	Expériences didactiques dans l'enseignement de l'histoire au niveau secondaire	FULL	EDIT	DEL
54	Enseignement renouvelé du français en Suisse romande: ce qu'en disent les enseignants de 1P et 2P	FULL	EDIT	DEL
55	Enseignement du français: l'observation de la communication orale en deuxième primaire	FULL	EDIT	DEL
	Enquête internationale sur l'enseignement de la lecture en première année			

Page 'Addition'

La page d'addition est un formulaire qui permet de saisir une nouvelle entrée. Veillez à respecter les types de champs en entrant des données (ne mettez pas de texte dans un champ INT).



mSQL Py Module 1.4

- Make a new query for table 'info_ed'
- Choose another table on 'csre-info'
- Choose another database

Fill in the fields and click on Submit

Nr

Titel

4. Syntaxe

4-1 mysqladmin

```
mysqladmin  Ver 6.6 Distrib 3.21.24-gamma, for sun-solaris2.6 on sparc
TCX Datakonsult AB, by Monty
This software comes with NO WARRANTY: see the file PUBLIC for details.

Administer program for the mysqld demon
Usage: mysqladmin [OPTIONS] command command....

  -#, --debug=...      Output debug log. Often this is 'd:t:o,filename'
  -f, --force           Don't ask for confirmation on drop table.
Continue
                        even if we get an error
  -?, --help            Display this help and exit
  -h, --host=#          Connect to host
  -p, --password[=...]  Password to use when connecting to server
                      If password is not given it's asked from the tty.
  -P  --port=...        Port number to use for connection
  -i, --sleep=sec      Execute commands again and again with a sleep
between
  -S  --socket=...      Socket file to use for connection
  -u, --user=#          User for login if not current user
  -V, --version         Output version information and exit

Where command is a one or more of: (Commands may be shortened)
  create databasename   Create a new database
  drop databasename     Delete a database and all its tables
  kill id,id,...        Kill mysql threads
  processlist           Show list of active threads in server
  reload                Reload grant tables
  refresh               Flush all tables and close and open logfiles
  shutdown              Take server down
  status                Gives a short status message from the server
  variables             Prints variables available
  version               Get version info from server
```

4-2 mysqldump

```
mysqldump  Ver 4.0 Distrib 3.21.24-gamma, for sun-solaris2.6 (sparc)
By Igor Romanenko & Monty & Jani. This software is in public Domain
This software comes with ABSOLUTELY NO WARRANTY

Dumping definition and data mysql database or table
Usage: mysqldump [OPTIONS] database [tables]

  -#, --debug=...      Output debug log. Often this is 'd:t:o,filename'
  -?, --help            Displays this help and exits.
  -c, --compleat-insert Use complete insert statements.
  -F  --flush-logs     Flush logs file in server before starting dump
  -f, --force           Continue even if we get an sql-error.
  -h, --host=...        Connect to host.
  -l, --lock-tables     Lock all tables for read.
  -t, --no-create-info  Don't write table creation info.
  -d, --no-data         No row information.
  -O, --set-variable var=option
                      give a variable an value. --help lists variables
  -p, --password[=...]  Password to use when connecting to server.
                      If password is not given it's asked from the tty.
```

```
  -P, --port=...        Port number to use for connection.
  -q, --quick           Don't buffer query, dump directly to stdout.
  -S, --socket=...      Socket file to use for connection.
  -T, --tab=...       Creates tab separated textfile for each table to
                        given path. (creates .sql and .txt files)
  -u, --user=#          User for login if not current user.
  -v, --verbose         Print info about the various stages.
  -V, --version         Output version information and exit.
```

4-3 mysql

```
mysql  Ver 9.11 Distrib 3.21.24-gamma, for sun-solaris2.6 (sparc)
By TCX Datakonsult AB, by Monty
This software comes with ABSOLUTELY NO WARRANTY.

Usage: mysql [OPTIONS] [database]

  -A, --no-auto-rehash  No automatic rehashing. One has to use 'rehash'
to
                       get table and field compleation. This gives a
quicker
                        start of mysql.
  -B, --batch           print results with a tab as separator, each
row on
                        a new line
  -#, --debug=...      output debug log. Often this is 'd:t:o,filename'
  -T, --debug-info      print some debug info at exit
  -e, --execute=...     execute command and quit.(--batch is implicit)
  -f, --force           continue even if we get an sql error.
  -?, --help            display this help and exit
  -h, --host=...        connect to host
  -n, --unbuffered      flush buffer after each query
  -O, --set-variable var=option
                       give a variable an value. --help lists variables
  -p, --password[=...]  password to use when connecting to server
                       If password is not given it's asked from the tty.
  -P  --port=...        Port number to use for connection
  -q, --quick          don't cache result, print it row by row. This may
                        slow down the server if the output is suspended
  -r, --raw             write fields without conversion. Used with -
-batch
  -s, --silent          be more silent.
  -S  --socket=...      Socket file to use for connection
  -t  --table=...       Output in table format
  -u, --user=#          user for login if not current user
  -v, --verbose         write more (-v -v -v gives the table output
format)
  -V, --version         output version information and exit
  -w, --wait            wait and retry if connection is down
```

4-4 mysqlshow

```
mysqlshow  Ver 6.0 Distrib 3.21.24-gamma, for sun-solaris2.6 (sparc)
TCX Datakonsult AB, by Monty
This software comes with ABSOLUTELY NO WARRANTY

Shows the structure of a mysql database (databases,tables and fields)

Usage: mysqlshow [OPTIONS] [database [table [field]]]

  -#, --debug=...      output debug log. Often this is 'd:t:o,filename'
  -?, --help            display this help and exit
  -h, --host=...        connect to host
```

```
  -k, --keys            show keys for for table
  -p, --password[=...]  password to use when connecting to server
                      If password is not given it's asked from the tty.
  -P  --port=...        Port number to use for connection
  -S  --socket=...      Socket file to use for connection
  -u, --user=#          user for login if not current user
  -V, --version         output version information and exit

If last argument contains a shell wildcard (* or ?) then only what's
matched by the wildcard is shown.
If no database is given then all matching databases are shown.
If no table is given then all matching tables in database are shown
If no field is given then all matching fields and fieldtypes in table
are shown
```

5. Références

URLS

Home Page mySQL: http://www.tcx.se/

Référence mySQL: http://tecfa.unige.ch/guides/mysql/ref-3.21.22-beta/manual_toc.html

Version PDF: http://tecfa.unige.ch/guides/mysql/ref-3.21.22-beta/mysql.pdf

mSQL FAQ: http://tecfa.unige.ch/guides/msql/msql2/faq.html

Home Page Python: http://www.python.org/

Référence Python: http://tecfa.unige.ch/guides/python/python.html

Livres

Python: Mark Lutz. (1996) Programming Python. O'Reilly & Associates.