LinuxFocus article number 226
http://linuxfocus.org

MySQL et Perl, le mariage de raison

par Georges Tarbouriech
<georges.t/at/linuxfocus.org>

L´auteur:

Georges est un vieil utilisateur d'Unix. Il apprécie tous ces produits qui ont contribué à répandre les solutions basées sur le logiciel libre dans le domaine professionnel.

Traduit en Français par:
Georges Tarbouriech
<georges.t/at/linuxfocus.org>

Résumé:

MySQL et Perl font partie du paysage depuis déjà longtemps. Ils sont toujours largement utilisés même si la "mode" est en train de changer. Cet article parle de ces deux produits utilisés conjointement soit sur Internet, soit sur votre réseau local. L'exemple proposé concerne les systèmes Unix, libres ou non, même s'il peut être adapté à d'autres "systèmes" largement répandus.
Ce qu'est cet article : un petit tour d'horizon de ce que l'on peut faire avec ce couple, en insistant sur la facilité d'utilisation, la rapidité, la fiabilité, la sécurité...
Ce que cet article n'est pas : ni un tutoriel MySQL ou Perl, ni une présentation approfondie de MySQL ou Perl.
Par conséquent, nous y verrons MySQL au travail en liaison avec Perl, sans oublier "qu'il y a plus d'une façon de le faire".

Quid de ce couple sympathique ?

MySQL est un Système de Gestion de Bases de Données Relationnelles (SGBDR) disponible sur http://www.mysql.com. Il est publié sous licence GNU GPL gratuitement en fonction de l'usage pour lequel il est prévu. Vérifiez la politique de licence sur le site de MySQL. Il fonctionne en tant que client ou serveur sur de nombreuses plate-formes. D'autres SGBDR libres existent et nous ne ferons aucune comparaison puisque le choix de MySQL pour cet article est purement arbitraire. De même nous ne comparerons pas avec les "gros" produits commerciaux tels qu'Informix, Oracle, Sybase... Précisons toutefois que MySQL est probablement l'un des SGBD les plus utilisés sur Internet. Pour cet article nous avons choisi (toujours arbitrairement) la version 3.23.36. Au moment d'écrire ces lignes, la dernière version stable est la 3.23.46 et la version expérimentale est la tant attendue version 4.0. Tout ceci peut être téléchargé sous forme de code source à compiler ou sous forme de paquetage.
Pour utiliser MySQL avec Perl, vous aurez besoin d'autre chose : les modules DBI de Perl. Vous devez au moins télécharger DBI, Msql-Mysql-modules, Data-Dumper et Data-ShowTable.

Nous ne parlerons pas de leur installation puisqu'elle est évidente et que les distributions vous fournissent tout ce que vous avez besoin de savoir.
Perl signifie Practical Extraction and Report Language. Au tout début il était destiné à la manipulation de document (analyse, extraction...) mais il est rapidement devenu beaucoup plus que ça. Vous pouvez pratiquement tout faire avec Perl, depuis les tâches d'administration aux scripts cgi, en passant par de véritables applications et bien sûr, l'interfaçage de bases de données.
Perl fait partie de nombreuses (sinon toutes) distributions Unix, qu'elles soient libres ou non. La version stable actuelle est la 5.6.1 et l'expérimentale est la 5.7.2 au moment d'écrire ces lignes. Pour cet article nous utiliserons la bonne vieille 5.005_03. Si Perl n'est pas installé sur votre machine (comment est-ce possible ?), vous pouvez l'obtenir sur http://www.perl.com. Perl propose des tonnes de modules pour pratiquement tout. Vous pouvez les obtenir à partir de la section CPAN de ce même site : une mine d'or !
Dernier point et non le moindre, pour travailler avec les deux outils réunis, surprise : il vous faut un serveur web ! Apache semble le bon choix puisqu'il fait partie de nombreuses distributions Unix. Si vous ne l'avez pas (où avez-vous trouvé votre distribution ?), il est disponible à http://www.apache.org.

L'exemple utilisé

Vous avez sans doute remarqué que LinuxFocus est un magazine multilingue. Ca signifie que lorsque vous êtes un éditeur, vous devez gérer le statut des nouveaux articles, leur traduction. En d'autres termes, qui fait quoi, quand... Actuellement, il y a environ 200 articles, traduits en moyenne en 5 langues. Ce qui fait un millier d'articles (que je suis bon !) et ça augmente tous les mois ! Tout cela doit être archivé, formaté, récapitulé... Comment croyez-vous que cette gestion fonctionne ? Grâce à Perl, bien sûr.
Notre éditeur en chef, Guido Socher, a écrit de nombreux programmes en Perl pour nous faciliter la tâche. Il a également écrit un tutoriel Perl en trois parties ainsi que la présentation d'un livre sur Perl. Voir la partie "Références" à la fin de l'article.
Javi, l'éditeur Espagnol, a écrit un programme pour gérer l'avancement des traductions... en Perl.
Atif, l'un de nos auteurs étoile, vient du royaume de Perl, et c'est pour cela que sa langue maternelle... est Perl. Accessoirement, il a également contribué à MySQL, en améliorant un outil d'administration web. Voir également la partie Références.
Tout cela pour dire que si vous cherchez un paradis Perl... rejoignez LinuxFocus.
Comme je suis l'un des éditeurs Français... et que je suis plutôt paresseux, j'ai crée ma propre base de données LinuxFocus en utilisant, devinez quoi : MySQL et Perl !

Création de la base

Cela suppose que MySQL a été correctement installé, que des utilisateurs ont été crées et ont été protégés par des mots de passe. L'installation n'entre pas dans le cadre de cet article et la documentation d'excellente qualité fournie avec MySQL vous dira tout.
Démarrez le serveur MySQL à l'aide du script *mysql.server*, puisqu'il invoque le démon *safe_mysqld* auquel vous pouvez passer des options. Connectez-vous au serveur en tapant

mysql -h host -u user -p

. Si le serveur est sur votre machine, *-h host* est inutile.
Après avoir tapé le mot de passe, vous êtes connecté au serveur (enfin, vous devriez !). Et vous pouvez maintenant créer votre base de données.
Au prompt mysql, tapez

CREATE DATABASE lf;

Il s'agit de notre exemple (lf pour LinuxFocus) et il est évident que vous donnez le nom de votre choix à votre propre base. Ensuite, garantissez certaines permissions aux utilisateurs autorisés, en supposant que vous ayez le droit de le faire (c'est-à-dire que l'utilisateur connecté posséde des droits d'administrateur). Si vous souhaitez qu'un utilisateur puisse gérer la base, vous pouvez lui en donner les privilèges en tapant

GRANT ALL ON lf.* TO username;

Sélectionnez la base que vous venez de créer en tapant

USE lf

. Créez une table en fonction de vos besoins. Dans notre exemple, nous définissons une table nommée trissue

CREATE TABLE trissue (num INTEGER UNSIGNED, category VARCHAR(25), title VARCHAR(40), author VARCHAR(20), en VARCHAR(20), es VARCHAR(20), fr VARCHAR(20),de VARCHAR(20), nl VARCHAR(20), ru VARCHAR(20), tk VARCHAR(20), issue VARCHAR(20));

. Vérifions qu'elle correspond bien à ce que l'on attend avec :

USE lf
SHOW TABLES;
DESCRIBE trissue;

C'est tout.
Maintenant, il nous faut des données. Pour intégrer des données dans une table vide, le moyen le plus simple consiste à utiliser un fichier texte avec des tabulations en guise de séparateurs. Lorsque votre texte est prêt, tapez :

LOAD DATA LOCAL INFILE "maindb.txt" INTO TABLE trissue;

Si votre fichier texte est correct, la table est maintenant peuplée. Vous pouvez le vérifier par :

SELECT * FROM trissue;

Ceci devrait afficher une longue liste. Vous pouvez dès lors retrouver n'importe quelle donnée à l'aide de requêtes.
Jusque là, ça va.

Nous n’avons utilisé que MySQL qui nous a permis de tout faire. Alors, que vient faire Perl là-dedans ?

Perl au travail

Perl va nous aider à automatiser les requêtes, à afficher les résultats dans un navigateur web, etc. Encore une fois, ceci implique que les modules Perl ont été correctement installés pour permettre l’utilisation de MySQL en liaison avec Perl.
Nous allons maintenant écrire des scripts Perl utilisés en tant que scripts cgi. Ils vont nous permettre de mêler Perl et HTML pour interroger la base et formater le résultat.
Nous ne traiterons que d’un simple exemple de script, nous autorisant à rechercher tous les articles d’un même auteur. Nous afficherons les numéros des articles, la catégorie, le titre, les noms des traducteurs des différentes langues (uniquement celles des projets établis), et le mois de parution de l’article.
Vous pouvez utiliser ce script comme modèle pour votre usage personnel, mais sachez que cet exemple n’est pas un programme particulièrement sécurisé. Vous pouvez obtenir une version un peu plus commentée =>ici<=.

```
#!/usr/bin/perl -Tw
# D’abord, disons qu’il s’agit d’un script Perl "Tainted".
#
# Ceci est un commentaire
# consultation de la base
#
# Utilisons le module Perl DBI
use DBI;

# en tant que cgi :
use CGI qw(param());

print <<END_of_start;

Content-type: text/html

<html>
<title>LFAuthors main db</title>

<center><TABLE>
<TR VALIGN=TOP>
<TD><form action="/cgi-bin/lf.cgi" method="get">

# Voici le titre du bouton de la page de lancement
<input type="submit" value=" LFAuth ">
</form>
</TD>
</TR>
</TABLE>
```

Nous demandons maintenant au script d’interroger la base

```
<center><H2>Search by author</H2></center>

<form action=\"/cgi-bin/lf.cgi\" method=\"get\">Author name : <input
type=\"text\" size=\"30\" name=\"author\"><input type=\"submit\"
value=\"Search...\"></form></center>


END_of_start


if (param("author") ne ’’) {
$author = param("author");

$autsrch.=’"’;
$autsrch.=$author;
$autsrch.=’"’;

# Nous nous connectons à la base nommée lf en tant qu’utilisateur untel

$dbh = DBI->connect("DBI:mysql:lf","untel",’’);

$sth = $dbh->prepare("
select *
from trissue
where
author = $autsrch
");

$sth->execute;
```

Nous demandons maintenant au script de préparer et d’afficher les résultats de la requête. Soit le champ de recherche est vide et le contenu de la base est affiché en totalité, soit nous recherchons un nom et chaque article correspondant à cet auteur sera affiché. Si vous avez des milliers d’enregistrements dans votre base, je vous déconseille l’affichage de la totalité du contenu !

```
print <<END_suite;

<center>
<TABLE BORDER=>
<tr bgcolor=#A1C4EE>
<th width=60 align=CENTER><font color=#000000> Num </font></th>
<th width=110 align=CENTER><font color=#000000> Category </font></th>
<th width=110 align=CENTER><font color=#000000> Title </font></th>
<th width=110 align=CENTER><font color=#000000> Author </font></th>
<th width=110 align=CENTER><font color=#000000> En </font></th>
<th width=110 align=CENTER><font color=#000000> Es </font></th>
```

```
<th width=110 align=CENTER><font color=#000000> Fr </font></th>
<th width=110 align=CENTER><font color=#000000> De </font></th>
<th width=110 align=CENTER><font color=#000000> Nl </font></th>
<th width=110 align=CENTER><font color=#000000> Ru </font></th>
<th width=110 align=CENTER><font color=#000000> Tk </font></th>
<th width=110 align=CENTER><font color=#000000> Issue </font></th>
</tr>

END_suite

while( ($num,$category,$title,$author,$en,$es,$fr,$de,$nl,$ru,$tk,$issue) =$sth->fetchrow() ) {
print "<tr>";
print "<td width=60 bgcolor=#FFFFE8 align=center> $num</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $category</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $title</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $author</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $en</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $es</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $fr</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $de</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $nl</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $ru</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $tk</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $issue</td>";
print "</tr>";

}
print "</TABLE>";
print "<BR>";
print "<BR>";
print "<br>";


} else {

# Connexion à la base

$dbh = DBI->connect("DBI:mysql:lf","untel",'');


# Recherche

$sth = $dbh->prepare("
select *
from trissue
");

$sth->execute;
```

Affichage du résultat

```
print <<SUITE;

<center>
<TABLE BORDER=>
<tr bgcolor=#A1C4EE>
<th width=60 align=CENTER><font color=#000000> Num </font></th>
<th width=110 align=CENTER><font color=#000000> Category </font></th>
<th width=110 align=CENTER><font color=#000000> Title </font></th>
<th width=110 align=CENTER><font color=#000000> Author </font></th>
<th width=110 align=CENTER><font color=#000000> En </font></th>
<th width=110 align=CENTER><font color=#000000> Es </font></th>
<th width=110 align=CENTER><font color=#000000> Fr </font></th>
<th width=110 align=CENTER><font color=#000000> De </font></th>
<th width=110 align=CENTER><font color=#000000> Nl </font></th>
<th width=110 align=CENTER><font color=#000000> Ru </font></th>
<th width=110 align=CENTER><font color=#000000> Tk </font></th>
<th width=110 align=CENTER><font color=#000000> Issue </font></th>
</tr>

SUITE

while( ($num,$category,$title,$author,$en,$es,$fr,$de,$nl,$ru,$tk,$issue) =$sth->fetchrow() ) {
print "<tr>";
print "<td width=60 bgcolor=#FFFFE8 align=center> $num</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $category</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $title</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $author</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $en</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $es</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $fr</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $de</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $nl</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $ru</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $tk</td>";
print "<td width=110 bgcolor=#FFFFE8 align=left> $issue</td>";
print "</tr>";

}
print "</TABLE>";
print "<BR>";

}
print end_html;
$sth->finish;
```

Déconnexion

$dbh->disconnect;

exit;

Voici le résultat obtenu dans un navigateur :

Num	Category	Title	Author	En	Es	Fr	De	It	Ru	Tk	Issue
53	UNIX Basics	Regular Expressions	G.Socher	G.Socher	F.Gimeno	J.D.Girard	G.Socher	F.Lambrechts	Ksv	unk	July1998
64	UNIX Basics	Finding Files	G.Socher	G.Socher	R.Sotillo	J.Perr	G.Socher	unk	unk	Tsiban	September1998
77	UNIX Basics	File Access Permissions	G.Socher	G.Socher	M.Moreno	J.Perr	G.Socher	E.Wilighagen	Aws	unk	January1999
114	Software Development	Perl I	G.Socher	G.Socher	D.Albilleira	Y.Hirou	K.Socher	F.Lambrechts	unk	unk	September1999
128	Software Development	Perl II	G.Socher	G.Socher	H.Rodriguez	G.Tarbouriech	K.Socher	F.Lambrechts	nt	G.Aky	November1999
134	System Administration	Home Networking	G.Socher	G.Socher	V.Rossetti	J.Peyratout	H.Radke	T.Uijdert	nt	nt	January2000
136	Software Development	Perl part III	G.Socher	G.Socher	J.R.Garcia	G.Tarbouriech	K.Socher	F.Lambrechts	nt	S.Dinc	January2000
148	Software Development	SNIFF+ for Linux	G.Socher	G.Socher	J.Palacios	E.Cappeau	nt	M.Reiners	nt	P.Kerulu	March2000
151	System Administration	Setting up IP-Masquerading	G.Socher	G.Socher	P.Vega	J.Peyratout	M.Blank	T.Uijdert	K.Poukhliakov	K.Ozturk	May2000
165	Hardware	Using Serial Line LCD	G.Socher	G.Socher	J.Nieto	J.Perr	J.Link	nt	K.Poukhliakov	Z.Gul	July2000
186	Hardware	A serial line shutdown button	G.Socher	G.Socher	nt	J.Peyratout	G.Socher	T.Uijdert	K.Poukhliakov	H.Kaya	January2001
192	System Administration	Using different ISPs	G.Socher	G.Socher	R.Velasco	J.Peyratout	G.Socher	nt	K.Poukhliakov	S.Bolucu	March2001
209	Software Development	Bookreview Professional Perl Programmin	G.Socher	G.Socher	A.Pardo	P.Lecoeds	P.Thalmann	A.van Eijkelenborg	K.Poukhliakov	E.Mutlu	July2001
215	System Administration	E-mail over UUCP	G.Socher	G.Socher	R.Hernando	G.Tarbouriech	S.Stein	T.Uijdert	E.Saenko	E.Mutlu	September2001
228	Hardware	Atek Super Mini Optical Mouse	G.Socher	G.Socher	nt	P.Titsch	G.Socher	nt	nt	E.Mutlu	November2001
222	UNIX Basics	Running applications remote with X11	G.Socher	G.Socher	nt	nt	nt	nt	nt	nt	np

C’est tout fini !

Du côté sécurité

De toute évidence, si vous souhaitez proposer un service de base de données sur votre site, vous devez sécuriser l’ensemble. Bien sûr, nous ne fournirons pas de méthode pas à pas pour sécuriser votre site ou votre serveur de bases de données. Toutefois, il est important d’insister sur ce qui est basique.
En bref, lorsque vous proposez des services sur Internet, la première chose à faire consiste à sécuriser votre serveur web. Ceci est très loin du cadre de cet article. Si vous voulez en savoir plus sur le sujet, il existe une énorme documentation. Un bon endroit pour commencer n’est autre que le Linux Documentation Project.
L’étape suivante concerne le serveur de bases de données. Lorsque vous installez un outil tel que MySQL, n’oubliez pas de lire le chapitre sécurité du manuel. Encore une fois, le travail de base concerne les mots de passe des utilisateurs : ne laissez jamais un compte sans mot de passe, particulièrement celui de root (qui devrait être différent de celui de la machine). L’autre point important

concerne les permissions : n’autorisez pas tout à tout le monde. Cela paraît évident... et c’est la raison pour laquelle nombreux sont ceux qui l’oublient !
Pour aller un peu plus loin, pourquoi ne pas "chrooter" la base ? Lisez l’article de Mark "Chrooter tous les services" dans ce numéro. Il y parle d’une autre base mais ce qu’il dit peut être appliqué à MySQL.
Une autre mesure de sécurité concerne la circulation des données. Ce n’est pas une mauvaise idée d’envoyer et de recevoir les données par un canal sécurisé. Vous pouvez lire l’article Par le tunnel pour plus ample information.
Enfin, l’essentiel, la programmation sécurisée est l’une des clés. Perl est un langage extraordinaire mais il est très facile de faire de grosses erreurs de programmation avec lui. Un autre article de LinuxFocus vous dira ce qu’il faut faire, particulièrement avec Perl. Jetez un oeil (et même les deux) là. C’est le dernier article de la série sur la programmation sécurisée et il concerne plus spécialement les scripts cgi. A lire absolument !
Alors, bien sûr, tout ceci suppose que votre système est déjà sécurisé, sans trous de sécurité bien connus, avec tous les derniers correctifs, et avec de nombreux outils de sécurité obligatoires tels qu’un NIDS (Network Intrusion Detection System) comme snort (http://www.snort.org), un pare-feu, des scanners de ports et de sécurité (nmap, nessus), etc.
Si vos moyens vous le permettent, vous pouvez aussi avoir un serveur différent pour chaque service proposé : un serveur web, un serveur de bases de données... et leur miroir pour garantir une haute disponibilité. Ainsi de suite ! Ce n’est jamais fini, puisque la sécurité n’est jamais aboutie. Vous essayez seulement de réduire les risques... et ils sont tous les jours plus gros. Vous voilà avertis.

Quoi d’autre ?

Comme Il Y a toujours Plus d’Une Façon de Le Faire (TIMTOWDI c’est quand même plus parlant que IYPUFDLF), vous pouvez choisir celle qui vous convient le mieux. Il existe de nombreux SGBDR ainsi que de nombreux langages pour communiquer avec eux. L’idée derrière cet article était de montrer que MySQL et Perl fonctionnent vraiment bien ensemble.
Certes, le choix était entièrement subjectif : j’adore MySQL pour sa relative petite taille, parce qu’il fonctionne sous de nombreux OS, qu’il est rapide, fiable... J’apprécie également beaucoup le travail de l’équipe de MySQL, sans oublier les nombreux contributeurs. Et ce que préfère : ces gens n’ont pas essayé de réinventer la roue. Ils ont conservé la simplicité.
Pour ce qui est de Perl, tout a été dit : que pourrais-je ajouter ? Je crois vraiment qu’il est impossible de travailler sans lui, que vous soyez administrateur réseau, développeur ou que sais-je. La communauté Perl est l’un des sièges du partage des connaissances. Un magazine existe, nommé le Perl Journal, qui est maintenant intégré au magazine SysAdmin, tous les deux numéros. Si vous voulez vous abonner, visitez http://www.samag.com.
Puisque nous parlons de beau travail, voici l’habituelle partie hors-sujet. Vous, lecteurs de LinuxFocus, n’avaient sans doute pas remarqué le petit nombre de personnes investies dans le magazine. Pourtant, vous pouvez le lire dans de nombreuses langues. Avez-vous remarqué que certaines équipes travaillent presque toujours avec une ou deux personnes qui font tout ? Ils sont traducteurs, webmestres, etc. Regardez du côté de l’équipe Russe ou de l’équipe Turque : vous verrez que la plupart des articles sont traduits par Kirill ou Erdal. Jetez un oeil sur les projets en cours de développement, tels que le Portugais ou l’Arabe : même résultat ! J’aimerais les féliciter pour l’énorme travail réalisé. Merci à vous tous : la communauté du logiciel libre peut vous être reconnaissante.
Pardon pour la digression, mais je crois qu’il fallait que ce soit dit.
Pour en revenir au sujet, terminons par quelques mots sur le logiciel libre. Les gens de MySQL et de

Perl méritent un grand merci. Ils nous fournissent des outils extraordinaires, presque toujours gratuitement. Pourtant ces outils sont souvent aussi "bons" que les véritables produits commerciaux (sinon meilleurs), ils sont fréquemment mis à jour, très bien documentés et vous pouvez les utiliser sur la plupart des systèmes Unix. Connaissez-vous un équivalent ailleurs ? J'ai bien peur que non !
Cet article ne vous apprendra sans doute pas grand chose, mais s'il vous donne envie d'essayer ces produits, il n'aura pas été inutile.
Quand je vous dis qu'on vit une époque formidable !

Références

Perl mongers

Le tutorial Perl de Guido :

Perl I
Perl II
Perl III

Revue du livre Programmation Professionnelle de Perl :

Programmation avec Perl

La contribution d'Atif à MySQL.

Une revue de MySQL sur LinuxFocus : vieil article encore d'actualité :

MySQL

Un vieux tutoriel SQL en deux parties sur LinuxFocus :

SQL Partie I
SQL Partie II