A trivial Perl module improving Oracle access from Perl

Author: *Stephen Thomas, Senior Systems Analyst,*
University of Adelaide Library

Last Update: *8/08/2006 12:30 pm*

Abstract: *Describes a simple Perl module used to simplify the writing and maintenance of Perl scripts used to extract data from an Oracle database.*

Problem

The Voyager Library system uses Oracle as its database engine, and this makes it possible to provide additional functionality using Perl scripts with embedded SQL statements to query the Oracle database. Over time, we have a developed a number of scripts of this nature, to perform a range of functions, from simple reports to authentication scripts.

In order to access the Oracle data, the Perl scripts use the DBI modules (available from CPAN). These require a number of environment variables, and also a username and password. A typical script will include the following lines:

```
use DBI;

#  Voyager/Oracle variables

$ENV{ORACLE_SID} = "VGER";
$ENV{ORACLE_HOME} = "/oracle/app/oracle/product/9.2.0";

my ($db_name, $db_username, $db_password, $db_host, $db_sid);

$db_host     = 'vtrain.library.adelaide.edu.au';
$db_sid      = "$ENV{ORACLE_SID}";
$db_name     = "xxxxxxdb";
$db_username = "xxxxxxxx";
$db_password = "xxxxxxxx";

        ...

my $dbh = DBI->connect(
                "dbi:Oracle:host=$db_host;sid=$db_sid",
                $db_username,
                $db_password
      ) || die "Could not connect: $DBI::errstr";
```

This is all very well, but poses several problems:

- Every script carries the access details, and in particular username and password, in plain text;

- If access details change, then every script needs to be edited to make those changes.

Solution

The solution to this problem arrived at here is to create a simple Perl module, VGER.pm, to provide the access details to scripts which need them. The module is then included in each script, in place of the actual details. When access details change, we need only edit the module, and all scripts immediately pick the changes. Although the module is still a plain text file on the server, the details are at least shown in only one place.

This has the additional benefit of making it easy to share scripts without sharing your passwords!

The complete module is as follows:

```
package VGER;
require Exporter;
@ISA = Exporter;
@EXPORT = qw(
        $db_host $db_sid $db_name
        $db_username $db_password
        $db_rw_username $db_rw_password
);

$VERSION = '2006.05.05';

# Initialise variables for database access

$ENV{ORACLE_SID} = "VGER";
$ENV{ORACLE_HOME} = "/oracle/app/oracle/product/9.2.0";

# database access details
$db_host        = "voyager-db.services.adelaide.edu.au";
$db_sid         = "VGER";
$db_name        = "xxxxxxdb";
$db_username    = "ro_xxxxxxxxx";
$db_password    = "ro_xxxxxxxxx";
$db_rw_username = "xxxxxxxxxxx";
$db_rw_password = "xxxxxxxxxxx";

1;

__END__
```

The module does two things: first it sets the required ENVIRONMENT variables needed by DBD::Oracle; second it provides variables needed for the DBI->connect function.

The module also provides two usernames, with read-only access and with write permission for scripts that update the database.

Our Perl scripts can then be simplified, with the required code reduced to this:

```
use VGER;
        ...
my $dbh = DBI->connect(
                "dbi:Oracle:host=$db_host;sid=$db_sid",
                $db_username,
                $db_password
            ) || die "Could not connect: $DBI::errstr";
```

References

Include references as relevant.

Endeavor's Voyager System
http://www.endinfosys.com/software/voyager/index.html

DBI
http://search.cpan.org/~timb/DBI-1.51/DBI.pm

DBD::Oracle
http://search.cpan.org/author/PYTHIAN/DBD-Oracle-1.18a/Oracle.pm