Databases and Perl

Using Perl to talk to databases

Why Program Databases?

Maxim 13.1

If at all possible, avoid the use of database driver ``enhancements''

Preparing Perl

```
$ man DBI
$ man DBD::mysql
```

```
$ find `perl -Te 'print "@INC"' ` -name '*.pm' -print | grep 'DBI.pm'
$ find `perl -Te 'print "@INC"' ` -name '*.pm' -print | grep 'mysql.pm'
```

```
$ locate DBI.pm
$ locate mysql.pm
```

Checking the DBI installation

```
#! /usr/bin/perl -w

# check_drivers - check which drivers are installed with DBI.

use strict;

use DBI;

my @drivers = DBI->available_drivers;

foreach my $driver ( @drivers )
{
    print "Driver: $driver installed.\n";
}
```

Programming Databases With DBI

```
#! /usr/bin/perl -w

# show_tables - list the tables within the MER database.
# Uses "DBI::dump_results" to display results.

use strict;
use DBI qw( :utils );

use constant DATABASE => "DBI:mysql:MER";
use constant DB_USER => "bbp";
use constant DB_PASS => "passwordhere";

my $dbh = DBI->connect( DATABASE, DB_USER, DB_PASS )
    or die "Connect failed: ", $DBI::errstr, ".\n";

my $sql = "show tables";
my $sth = $dbh->prepare( $sql );
$sth->execute;
print dump_results( $sth ), "\n";
$sth->finish;
$dbh->disconnect;
```

Maxim 13.2

Be sure to adhere to any established naming conventions within a programming community

Results from show_tables ...

```
$ chmod +x show_tables

$ ./show_tables


'citations'
'crossrefs'
'dnas'
'proteins'
4 rows
4
```

Maxim 13.3

Avoid littering programs with username/password combinations

Developing A Database Utility Module

```
package DbUtilsMER;

# DbUtilsMER.pm - the database utilities module from "Bioinformatics,
# Biocomputing and Perl".
#
# Version 0.01: module created to hold MERconnectDB.

require Exporter;

our @ISA = qw( Exporter );

our @EXPORT = qw( MERconnectDB );
our @EXPORT_OK = qw();
our %EXPORT_TAGS = ();

our $VERSION = 0.01;
```

Developing A Database Utility Module, cont.

```
use constant DATABASE => "DBI:mysql:MER";
use constant DB_USER => "bbp";
use constant DB_PASS => "passwordhere";

sub MERconnectDB {
    #
    # Given: nothing.
    # Return: a "connected" database handle to the MER database or
    # if no connection is possible, return "undef".
    #

    return DBI->connect( DATABASE, DB_USER, DB_PASS );
}

1;
```

Using the Database Utility Module

```
use lib "$ENV{'HOME'}/bbp/";
use DbUtilsMER;

my $dbh = MERconnectDB
    or die "Connect failed: ", $DBI::errstr, ".\n";

use constant DATABASE => "DBI:mysql:MER";
use constant DB_USER => "bbp";
use constant DB_PASS => "passwordhere";

my $dbh = DBI->connect( DATABASE, DB_USER, DB_PASS )
    or die "Connect failed: ", $DBI::errstr, ".\n";
```

Improving upon dump_results

```
#! /usr/bin/perl -w

# show_tables2 - list the tables within the MER database.
# Uses "fetchrow_array" to display results.

use strict;

use DBI;

use lib "$ENV{'HOME'}/bbp/";
use DbUtilsMER;

my $dbh = MERconnectDB
    or die "Connect failed: ", $DBI::errstr, ".\n";

my $sql = "show tables";

my $sth = $dbh->prepare( $sql );
```

Improving upon dump_results, cont.

```
$sth->execute;

print "The MER database contains the following tables:\n\n";

while ( my @row = $sth->fetchrow_array )
{
    foreach my $column_value ( @row )
    {
        print "\t$column_value\n";
    }
}

$sth->finish;

$dbh->disconnect;
```

Results from show_tables2 ...

```
The MER database contains the following tables:

citations
crossrefs
dnas
proteins
```

Customizing Output

```
There are 22 cross references in the database.

The protein P04336 is cross referenced with J01730.
The protein P08332 is cross referenced with J01730.
The protein P08654 is cross referenced with M15049.
The protein P20102 is cross referenced with X03405.
The protein Q52107 is cross referenced with AF213017.
    .
    .
The protein P03830 is cross referenced with J01730.
The protein Q52109 is cross referenced with AF213017.
The protein P20102 is cross referenced with J01730.
The protein Q52106 is cross referenced with AF213017.
The protein P04337 is cross referenced with K03089.
The protein P08664 is cross referenced with M15049.
```

Customizing Output, cont.

```
#! /usr/bin/perl -w

# which_crossrefs - nicely displayed list of protein->dna
# cross references.

use strict;

use DBI;

use lib "$ENV{'HOME'}/bbp/";
use DbUtilsMER;

my $dbh = MERconnectDB
    or die "Connect failed: ", $DBI::errstr, ".\n";

my $sql = "select * from crossrefs";

my $sth = $dbh->prepare( $sql );
```

Customizing Output, cont.

```
$sth->execute;

print "There are ", $sth->rows, " cross references in the database.\n\n";

while ( my @row = $sth->fetchrow_array )
{
    print "The protein $row[0] is cross referenced with $row[1].\n";
}

$sth->finish;

$dbh->disconnect;
```

Alternatives to fetchrow_array

```
while ( my ( $protein, $dna ) = $sth->fetchrow_array )
{
    print "The protein $protein is cross referenced with $dna.\n";
}
```

```
while ( my $row = $sth->fetchrow_hashref )
{
    print "The protein $row->{ ac_protein } is cross referenced ";
    print "with $row->{ ac_dna }.\n";
}
```

Maxim 13.4

Use fetchrow_hashref to guard against changes to the structure of a database table

Customizing Input

```
Provide a protein accession number to cross reference ('quit' to end): p03377
Not found: there is no cross reference for that protein in the database.

Provide a protein accession number to cross reference ('quit' to end): p04337
Found: P04337 is cross referenced with J01730.

Provide a protein accession number to cross reference ('quit' to end): q52109
Found: Q52109 is cross referenced with AF213017.

Provide a protein accession number to cross reference ('quit' to end): x6587
Not found: there is no cross reference for that protein in the database.

Provide a protein accession number to cross reference ('quit' to end): quit
```

Customizing Input

```
#! /usr/bin/perl -w

# specific_crossref - allow for the "interactive" checking
# of crossrefs from the command-line.
# Keep going until the user enters "quit".

use strict;

use DBI;

use lib "$ENV{'HOME'}/bbp/";
use DbUtilsMER;

use constant TRUE => 1;

my $dbh = MERconnectDB
    or die "Connect failed: ", $DBI::errstr, ".\n";

my $sql = qq/ select ac_dna from crossrefs where ac_protein = ? /;

my $sth = $dbh->prepare( $sql );
```

Customizing Input, cont.

```
while ( TRUE )
{
    print "\nProvide a protein accession number to cross ";
    print "reference ('quit' to end): ";

    my $protein2find = <>;

    chomp $protein2find;

    $protein2find = uc $protein2find;

    if ( $protein2find eq 'QUIT' )
    {
        last;
    }
```

Customizing Input, cont.

```
    $sth->execute( $protein2find );

    my $dna = $sth->fetchrow_array;

    $sth->finish;

    if ( !$dna )
    {
        print "Not found: there is no cross reference for that protein ";
        print "in the database.\n";
    }
    else
    {
        print "Found: $protein2find is cross referenced with $dna.\n";
    }
}

$dbh->disconnect;
```

Extending SQL

```
#! /usr/bin/perl -w

# The 'db_match_embl' program - check a sequence against each EMBL
# database entry stored in the dnas
# table within the MER database.

use strict;

use DBI;

use lib "$ENV{'HOME'}/bbp/";
use DbUtilsMER;

use constant TRUE => 1;
use constant FALSE => 0;

my $dbh = MERconnectDB
    or die "Connect failed: ", $DBI::errstr, ".\n";

my $sql = qq/ select accession_number, sequence_data,
                         sequence_length from dnas /;

my $sth = $dbh->prepare( $sql );
```

Extending SQL, cont.

```
while ( TRUE )
{
    my $sequence_found = FALSE;

    print "Please enter a sequence to check ('quit' to end): ";

    my $to_check = <>;

    chomp( $to_check );
    $to_check = lc $to_check;

    if ( $to_check =~ /^quit$/ )
    {
        last;
    }
    $sth->execute;
    while ( my ( $ac, $sequence, $sequence_length ) = $sth-
   >fetchrow_array )
    {
        $sequence =~ s/\s*//g;
```

Extending SQL, cont.

```
            if ( $sequence =~ /$to_check/ )
            {
                $sequence_found = TRUE;

                print "The EMBL entry in the database: ",
                      $ac, " contains: $to_check.\n";
                print "[Lengths: ", length $sequence,
                      "/$sequence_length]\n\n";
             }
         }

        if ( !$sequence_found )
        {
            print "No match found in database for: $to_check.\n\n";
        }

        $sth->finish;
    }

    $dbh->disconnect;
```

Results from db_match_embl ...

```
Please enter a sequence to check ('quit' to end): aattgc
The EMBL entry in the database: AF213017 contains: aattgc.
[Lengths: 6838/6838]

Please enter a sequence to check ('quit' to end): aatttc
The EMBL entry in the database: AF213017 contains: aatttc.
[Lengths: 6838/6838]

The EMBL entry in the database: J01730 contains: aatttc.
[Lengths: 5747/5747]

Please enter a sequence to check ('quit' to end): accttaaatttgtacgtg
No match found in database for: accttaaatttgtacgtg.

Please enter a sequence to check ('quit' to end): quit
```

Where To From Here