

Using CGI to List Tables and Rows

Professor Don Colton, BYU Hawaii

March 22, 2004

In this handout, we combine DBI and CGI into the same Perl program. We assume you are familiar with DBI->connect.

Please avoid using CUT and PASTE. Type things in by hand. Think about them as you type. If there is something interesting or confusing, ask about it. You will probably learn more.

1 Main Program

This section presents the main program. It relies on two subroutines to reduce complexity. We read a line of cgi input, then connect to the database and build a list of tables. If a table was previously selected we continue by building a list of its rows. Then we stop.

```
#!/usr/bin/perl -Tw
use DBI;

chomp ( $in = <STDIN> ); # read cgi input
$in = "&$in&"; # add & sentinels

$st = DBI->connect ( "DBI:mysql:DDD:HHHH",
    "UUU", "PPP" );

print "Content-type: text/html\n
<html><head><title>Table Viewer</title>
</head><body>
<h1>Table Viewer</h1>
Please select a table for viewing.
<form method=post action=''\>\n";
tableButtons();

# if table is specified list its contents
if ( $in =~ /&table=([~&]+)&/ ) {
    tableRows($1); }

# end of processing
print "</body></html>\n";
$st->disconnect();
```

2 Table Buttons

The tableButtons subroutine identifies the list of tables that are available and presents them as HTML submit buttons. It could easily be placed in-line in the main program, but is separated to reduce complexity. Note: the space in front of name= is important.

```
sub tableButtons {
    $q1 = $st->prepare ( "show tables" );
    $q1->execute();
    while ( @z = $q1->fetchrow_array() ) {
        print "<input type=submit";
        print " name=table value='$z[0]'\>\n"; }
    $q1->finish();
}
```

3 Table Rows

The tableRows subroutine accepts one table name. Then it does a select to get its contents. It displays the rows and columns as a table in HTML.

```
sub tableRows {
    my ( $table ) = @_; # local variable
    print "<h1>Contents of Table $table</h1>
<table border=1>\n";
    $query = "select * from $table";
    $q2 = $st->prepare ( $query );
    $q2->execute();
    while ( @z = $q2->fetchrow_array() ) {
        print "<tr>";
        foreach $cell ( @z ) {
            print "<td>$cell</td>";
            print "</tr>\n"; }
        $q2->finish();

    print "</table>\n";
}
```

Advanced: "show columns from \$table" gives column names and types. Use <th> instead of <td> to display table header information in HTML.