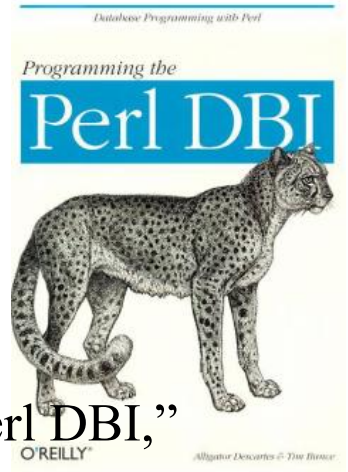
Perl语言入门与提高
Lesson 16

周晓方
courses@xfzhou.homeftp.org

Perl DBI

- 安装DBI和相关模块
  - 要用到和所安装Perl对应的C编译器
- SQL简介
- dbish
- DBI和CSV文件
- DBI和foxpro(dBase)文件
- DBI和MySql
- DBI的proxy
  - 参考：
    - Alligator Descartes & Tim Bunce, "Programming Perl DBI," O'Reilly 2000
    - 王冠编《SQL查询语言及应用》科学出版社1999
    - 《SQL入门经典》4th ed



安装DBI、DBD

- DBI模块的版本和perl的版本
  - DBI-1.616要求Perl5.8.1或以上, v1.53/1.50在Perl5.6.1安装报Warning
  - DBI-1.47可以在Perl5.6.1上安装
- DBD::ODBC
  - V1.1.4 and above (v1.29 now)→perl 5.8.1 or above
  - V1.1.3→work with perl 5.6.1
- DBD::CSV(0.31)→perl 5.8.1 (v0.22→perl 5.6.1)
  - 预装Test-CSV_XS(0.82)
  - 预装SQL::Statement
    - V1.26 and above (v1.33 now)→Perl 5.8.0
    - V1.15→Work with Perl 5.6.1

dbish, DBI proxy

- DBI::Shell(v11.95)
  - 预装IO::Tee(v0.64)
  - 预装Text::Reform(v1.20)

- DBI::Proxy和DBI::ProxyServer(包含在DBI内,一起安装)
  - 预装RPC::PlClient和RPC::PlServer (包含在PIRPC中,v0.2020)
  - 预装Net::Daemon(v0.48)

DBI 压缩/加解密

- 加解密的模块
  - Crypt::DES(v2.05), Crypt::Blowfish(v2.12), Crypt::Blowfish_PP(v1.12), Crypt::CAST5(v0.05), Crypt::CAST5_PP(v1.04), Crypt::Rijndael_PP(v0.05), Crypt::Rijndael(v1.09), Crypt::CBC(v2.30), ExtUtils::MakeMaker(v6.56), Test::Manifest(v1.23)等
  - Crypt::IDEA(v1.08)要手工改一下，才能在activeperl5.6/vc++6下跑通:
    - idea.h的<inttypes.h>一行替换成：

```
typedef unsigned short u_int16_t;
```

    - _idea.c的#include <wininet.h>一行替换成以下三行:

```
typedef int int32_t;
#include <windows.h>
u_int16_t PASCAL FAR htons (u_int16_t hostshort);
```

草莓Perl(v5.8.8)上的安装

- 更新以下模块
  - Test::Simple(v0.98)
  - PathTools(v3.33)
  - Params::Util

yr2017 http://search.cpan.org/dist/PathTools/

- 安装以下模块
  - Clone

- 留意安装顺序，最新的DBI、DBD、Crypt等模块都可以顺利安装
  - DBD::XBase v1.00以上都需要Perl5.10.0，可以找古老的DBD::XBasev0.241

安装MySql和DBD::mysql

- MySql
  - 选择一：下载安装MySql(包含.h/.lib)；选择二：
  - 下载安装XAMPP(含Apache/MySql等,不含.h/.lib)
  - 可参考Lynda.com的MySql Essential Training入门
- DBD::mysql(需要MySql的.h/.lib和本地编译器)
  - 最新版4.019，无法在AP5.6/VC6上编译
  - 早期版本2.900x在AP5.6/VC6上也没能编译通过
- DBD::mysqlPP(v0.04, 纯Perl版本,功能弱一些)
  - 预安装Net::MySQL(v0.09)
- 如果Perl程序无法连接MySql服务器,请检查MySql的网络安全设置,并尝试卸载win上的ipv6

```
ipv6 -uninstall
```

SQL简介—连接和断开

- 用户要先连接到SQL服务器，才能开始一个SQL会话；会话结束后要断开连接

connect user@database

…可能需要输入密码

… … …

…会话，用户输入各种SQL命令，或执行SQL程序，得到结果

… … …

disconnect

SQL简介—创建和删除数据库

- 用户登陆数据库服务器后,可管理多个数据库
  - 每个数据库包含一个或多个表格
  - 每个表格预先规定了表头(域和数据类型),称为域或列,每行数据称为记录或行

People | Rank | List | City

Prime Key	Name (job title)	Age	Nickname	
001	Gyacomo Guilissoni Founder & CEO	34	Peldi	☑
002	Guido Jack Guilizzoni	4	The Guids	☐
003	Marco Botti Tuttofare	31		☑
004	Mariah Mclachlan Better Half	35	Patata	☑

Database ➔ several Tables

Table ➔ rows & columns

- 创建/罗列/选取/删除数据库的SQL命令

CREATE DATABASE *数据库名称*

SHOW DATABASES

USE *数据库名称*

DROP DATABASE *数据库名称* *drop是危险的命令*

SQL简介—创建表格

- 考虑下面的关系型数据库表格的结构
  - 4个字段和类型，分别是
    - 编号ID，整数，int
    - 名字NAME，字符串，char (20)
    - 性别GENDA，boolean
    - 生日DOB，date
- 创建表格

```
create table list    (id      int,
                      name    char(20),
                      genda   boolean,
                      dob     date)
```

字段名称

数据类型

表格的名称

- 删除表格

```
drop table list
```

drop是危险的SQL命令

SQL简介—数据类型

- 定长字符串 char (*n*)
- 大对象类型 varchar(*n*)
- 数值 numeric(*n*) numeric(*p*,*s*)
  - 小数 decimal(*p*, *s*)　　*p*是有效位数，*s*是小数位数
  - 整数 integer
  - 浮点数 float(*p*, *s*)　　real(*s*) doule percision(*p*, *s*)
- 日期和时间 date　time　datetime　timestamp
- NULL(或”一对单引号，中间没有东西)
- 布尔值 boolean，取值范围是true/false/null
- 枚举型enum(有限个取值中选一个),集合型set(多选)
- 自定义
- 注意：不是所有的类型都被特定数据库支持,数据类型的具体名称也可能各自不同

SQL简介—插入和删除记录

- 插入记录

```
insert into list values( "10",
                         "Weby",
                         "0",
                         "19881009")
```

表格名称

根据所用SQL数据库的手册，确定单双引号以及数字、日期类型的具体格式，例如’October 9, 1988’

- 删除记录

```
delete from list
```

删除全部记录!!

```
delete from list where  id > 5
```

删除满足条件的记录

SQL简介—数据库查询

- SELECT是最强大的SQL命令
- select *[unique]*
  field_list 或 *
  from *table_name*
  *[*where *search_condition]*
  *[*order by *sort_condition]*

表格名称

可以规定记录返回的次序

select * from *list* **where** *id* < *5* **order** by name

要显示的字段和表达式列表,*表示全部字段

where子句给出过滤条件, 否则返回全部记录

SQL简介—更新记录

- update *table_name* set *column*1 = '*value*'
  [, *column2* = '*val2*', *col3* = '*val3*'…]
  [where *condition*];

update *list* set *name* = '!!'
where *id* in (select *id* from *rank* where *perl* > 90)

在rank中找到perl字段大于90的所有记录，对应id在list表的name字段改成'!! '

不是所有的Sql实现都支持上面这条较复杂的Sql命令

- 如果不加where 条件，指定表格中**所有记录**的相关字段都会被修改

dbish——操练SQL命令的入口

- 直接进入dbish并菜单选择数据库接口

dbish ↵

```
Available DBI drivers:
  1: dbi:CSV
  2: dbi:XBase
  .. .. .. ..
  9: dbi:mysqlPP
Enter driver name or number, or full 'dbi:...:...'
   DSN:
```

- 从命令行直接选择数据库接口类型并进入dbish：

dbish *dbi:接口:参数 用户 密码*

dbish dbi:CSV:. ↵

dbish dbi:XBase:. ↵

dbish dbi:mysqlPP:database=test;host=localhost try 333 ↵

试用一下 dbish dbi:XBase:.

- 创建表格
- 插入数据
- 查询

```
1.  create table city (
2.  id int not null auto_increment primary key ,
3.  name varchar(30),
4.  pop int);
5.  insert into city values(1, "shanghai", 1854);
6.  insert into city values(2, "beijin", 1567);
7.  insert into city values(3, "chongqing", 1205);
8.  insert into city values(4, "wuhan", 1024);
9.  insert into city values(5, "tianjin", 957);
10. select * from city where pop < 1200;
11. ID,NAME,POP
12. 4,'wuhan',1024
13. 5,'tianjin',957
14. [2 rows of 3 fields returned]
15. @dbi:XBase:.> /quit
```

DBD-XBase尚不支持这些属性

dbish

- dbish本身的命令是/或;开头的，例如
  - 退出dbish　　/quit　或　　/exit
  - 简单的帮助　　/help
  - 有哪些drivers　　/drivers
  - 当前driver的数据类型　　/type_info
  - 连接其他drivers　　/connect
    /connect dbi:CSV:.
- dbish里面的sql命令是/或;结尾的，例如
  - 表格列表　　select * from *my_table* ;

Perl DBI的用法

- use DBI;程序开头列DBI，perl会自己找DBD
- 有两种句柄，db handle和statement handle，
- 程序中一般用$dbh，$sth来表示

- DBI->connect成功连接后，返回$dbh

- $dbh->prepare('一个SQL命令')返回$sth
- $sth->execute()执行SQL命令
- 每个SQL命令在Perl DBI中都是先'准备'再'执行'的
  - 非SQL select命令也可以用$dbh->do('命令')，返回影响的行数

- 最后$dbh->disconnect断开DBI和数据库服务器的连接

Perl访问CSV格式数据库

- 准备一个csv文件，csv就是逗号分割的文本文件，第一行相当于表头(域的名字)，后面行是数据，例如从wiki找来一些数据，做这样一个文本文件，并存放在./csv/wiki.csv内(.是当前目录)

Rank,Country,Area_km2,Area_sqmi,%_of_Total,Notes
1,Russia,17098242.00 ,6601668.00 ,11.50%,The largest country in the world.
2,Canada,9984670.00 ,3855100.00 ,6.70%,The largest country in North …
3,United States,9629091.00 ,3717813.00 ,6.50%,Includes only states ..
… …

- 保存好后可以先尝试用dbish去访问一下

```
dbish  dbi:CSV:f_dir=csv ↵
/table_info ↵
select * from wiki.csv; ↵
select Country,Area_km2 from wiki.csv where Country
  like 'A%'; ↵
/quit ↵
```

'A%'表示A开头的字符串，这是SQL的"正则表达式"

- CSV的SQL驱动来自SQL::Statement，有限支持SQL，参考 http://search.cpan.org/~rehsack/SQL-Statement-1.33/lib/SQL/Statement/Syntax.pod

大小写无关用clike，大小写相关用like

Perl DBI访问CSV文件

```
1.  #!/usr/bin/perl -w
2.  use DBI;
3.  #my $dbh = DBI->connect("DBI:CSV:f_dir=csv", '', '');# both works
4.  my $dbh = DBI->connect("DBI:CSV:f_dir=csv",'', '',{f_dir => 'csv'})
5.      or die("Can't connect to DBI:CSV:\n");
6.  # $sth = $dbh->prepare("SELECT Country FROM wiki.csv") or die ...;
7.  $dbh->{csv_tables}->{wiki} = { 'file' => 'wiki.csv'};
8.  my $sth = $dbh->prepare("SELECT Country FROM wiki")
9.      or die "Fail to prepare SELECT.\n";
10. $sth->execute()
11.     or die "Fail to execute.\n";          # 执行select命令
12. my @row;
13. while (@row = $sth->fetchrow_array()) {   # 打印select的结果
14.     print "Row: @row\n";
15. }
16. warn "Data fetching erro $DBI::errstr\n"
17.     if $DBI::err;
18. print '"', $dbh->tables(), '"', "\n";     # 打印一些属性
19. my ($k, $v);
20. while (($k, $v) = each %{$dbh->{csv_tables}{wiki}}) {
21.     print "$k => $v\n";
22. }
23. $dbh->disconnect();
24. 1;
```

Perl DBI访问CSV文件

- 3-5和23行分别是连接、断开
- 设置“属性”可以有两种方式，如3和4行
- CSV和SQL::Statement没有用户和密码
- 7-8行也可以写成6行的样子，前者给wiki.csv取了一个简化的名称
- select命令先prepare(8)再execute(10)，最后用fetchrow_array(13-15)循环来读取数据
- 做其他SQL命令，也是先准备后执行
- 总是检查执行结果，并及时报错

Perl访问dbf文件(DBD::XBase)

- SQL驱动也来自SQL::Statement
- .dbf文件可以用foxpro或excel打开

```
1.  #!/usr/bin/perl -w
2.  use DBI;
3.  unlink "list.dbf", "rank.dbf";    # 准备新建dbf, 最好改成drop命令
4.  my $dbh = DBI->connect("DBI:XBase:.")
5.      or die $DBI::errstr;
6.  my $sth = $dbh->prepare("create table list (
7.      id int, name char(20), male boolean, dob date)")
8.      or die $DBI::errstr;          # 新建表格,列出表头和数据类型
9.  $sth->execute()
10.     or die $DBI::errstr;
11. $sth = $dbh->prepare(             # 插入一行数据
12.     'insert into list values("10", "Weby", "0", "19881009")')
13.     or die $DBI::errstr;          # 留意XBase录入日期的格式
14. $sth->execute()
15.     or die $DBI::errstr;
```

Perl访问dbf文件

```
1.  my ($line, @data);
2.  foreach $line (<DATA>) {
3.      chomp $line;
4.      @data = split /,\s+/, $line;
5.      next unless @data == 4;
6.      $data[-1] =~ s/^(\d+)[-.\/](\d+)[-.\/](\d+)$/sprintf
    '%04d%02d%02d', $1, $2, $3/e;     # XBase的“日期”用yyyymmdd格式
7.      # Note that DBD::XBase accept "YYYYMMDD" for date format.
8.      $sth = $dbh->prepare("insert into list values(" .
9.             join(',', map "\"$_\"", @data) .
10.            ")")
11.            or die $DBI::errstr;
12.     $sth->execute()
13.            or die $DBI::errstr;
14. }
```

```
1.  __END__
2.  11, Job, 1, 1968-12-1
3.  21, Chip, 1, 1966-4-3
4.  3, Lane, 1, 1977-9-15
5.  14, Flora, 0, 1988-5-20
```

Perl访问dbf文件

```
1.  $sth = $dbh->prepare("create table rank (id int, perl int);")
2.      or die $DBI::errstr;
3.  $sth->execute()
4.      or die $DBI::errstr;
5.  #$sth = $dbh->prepare("alter table rank add mcu int")
6.  #$sth = $dbh->prepare("ALTER TABLE rank modify perl mcu;")
7.  #   or die $DBI::errstr;
8.  #$sth->execute()
9.  #   or die $DBI::errstr;

10. foreach (split / /, "11,100 21,98 3,75 14,85 10,91") {
11.     $sth = $dbh->prepare("insert into rank values($_);");
12.     $sth->execute();
13. }
```

Perl访问dbf文件

```
1.  $sth = $dbh->prepare("update list set name = list.name || '!'
    where id < 20; ");
2.  # where id in (select id from rank where perl > 90)");
3.  $sth->execute();

4.  $sth = $dbh->prepare("select * from list ;");
5.  $sth->execute();
6.  my @row;
7.  while (@row = $sth->fetchrow_array()) {
8.      print "Row: @row\n";
9.  }

10. $dbh->disconnect;
11. 1;
```

SQL::Statement不支持

```
Row: 10 Weby! 0 19881009
Row: 11 Job! 1 19681201
Row: 21 Chip 1 19660403
Row: 3 Lane! 1 19770915
Row: 14 Flora! 0 19880520
```

Perl访问mysql

- 安装并运行mysql服务器，创建try用户，密码是333

```
1.  #!/usr/bin/perl -w
2.  use DBI;
3.  my $dbh = DBI->connect(              # connect的参数不同
4.      "DBI:mysqlPP:database=test;host=127.0.0.1",
5.      'try', '333',                    # 用户和密码
6.      {RaiseError => 1},               # 有错就要报错
7.      ) or die $DBI::errstr;           # 后面都和XBase的例子一样..
8.  my $sth = $dbh->prepare("create table list (" .
9.      . "id int, name char(20), male boolean, dob date)")
10.     or die $DBI::errstr;
```

- 最后的update可以包含select，但字符串表达式不同：

```
1.  $sth = $dbh->prepare("update list set name = concat(list.name,
    '!') where id in (select id from rank where perl > 90)");
2.  $sth->execute();
```

➔SQL很古老，有各种方言，会影响移植性

Perl访问mysql

- 删除list和rank表格

```
1.  #!/usr/bin/perl -w
2.  use DBI;
3.  my $dbh = DBI->connect(
4.      "DBI:mysqlPP:database=test;host=127.0.0.1",
5.      'try', '333', {RaiseError => 1})
6.      or die $DBI::errstr;
7.  my $sth = $dbh->prepare("drop table list, rank")
8.      or die $DBI::errstr;
9.  $sth->execute()
10.     or die $DBI::errstr;
11. $dbh->disconnect;
12. 1;
```

- SQL::Statement也支持drop table命令

```
DROP TABLE [IF EXISTS] <table>
```

- SQL::Statement也有内建函数, 并支持自定义函数:

http://search.cpan.org/~rehsack/SQL-Statement-1.33/lib/SQL/Statement/Functions.pm

Perl DBI的proxy

- 服务端运行dbiproxy (--help查看帮助信息)

```
dbiproxy --localport 3333 -debug
```

指定一个端口号

- Perl dbi脚本只需修改connect的参数

请改成运行dbiproxy服务的电脑的域名或IP地址

```
my $dbh = DBI->connect(
"dbi:Proxy:hostname=localhost;port=3333;dsn=" .
"DBI:XBase:.");
```

只要增加这么一段，其他都不用动

```
my $dbh = DBI->connect(
"dbi:Proxy:hostname=localhost;port=3333;dsn=" .
  "DBI:mysqlPP:database=test;host=127.0.0.1",
  'try', '333', {RaiseError => 1});
```

- Topics not covered here: 访问权限、压缩、加密

Perl访问SQLite

- SQLite基于文件的SQL http://sqlite.org/
  - Public Domain. Pure C source code.
  - 数据库为本地文件，不是Client-Server模式
  - 适合用来替代基于fopen的自定义数据库
  - 适合用于教学、临时数据库、嵌入式应用
  - 支持多用户并行读取，但写入只支持单用户
- Perl的DBD::SQLite模块是完整模块
  - 无需再安装任何SQLite的代码
  - dbish dbi:SQLite:my.db ↵

Perl访问ODBC

SQL::Statement的函数