データベース構築論2

15章「PERL」

中島康彦

§15. 1 perlの概要

シェルスクリプトでは限界がある.
- パターンマッチが貧弱

C言語では大がかりになってしまう.
- 文字列処理のための領域確保が繁雑

文字列を扱える簡便なインタプリタ言語が必要
- Practical Extraction and Report Language
- 見た目(小規模, エレガント, 最少)より実用性重視(使い易さ, 効率, 完全性)
- sed, awk, sh, Cから良いとこ取り

§15. 2 perlのデータ構造

スカラ

```
$days               # 単純なスカラ変数 "days" の値
$days[28]           # 配列 @days の 29 番目の要素の値
$days{'Feb'}        # ハッシュ %days の 'Feb' の値
$#days              # 配列 @days の最後のインデクス値
```

スカラの配列

```
@days               # ($days[0], $days[1],..., $days[n])
@days[3,4,5]        # @days[3..5] と同じ
@days{'a','c'}      # ($days{'a'},$days{'c'}) と同じ
```

スカラの連想配列「ハッシュ」

```
%days               # (key1, val1, key2, val2,...)
```

§15. 2 perlのデータ構造(続き)

数値リテラル

```
12345
12345.67
.23E-10
0xffff              # 16 進数
0377                # 8 進数
4_294_967_296       # 下線は読みやすさのため
```

文字列リテラル

```
$Price = '$100';                    # 置換されない
print "The price is $Price.\n";     # 置換される
```

§15. 3 perlの構文

コメント

文字 "#" から行末まで

複合実行文

```
if (EXPR) BLOCK
if (EXPR) BLOCK else BLOCK
if (EXPR) BLOCK elsif (EXPR) BLOCK ... else BLOCK
LABEL while (EXPR) BLOCK
LABEL while (EXPR) BLOCK continue BLOCK
LABEL for (EXPR; EXPR; EXPR) BLOCK
LABEL foreach VAR (ARRAY) BLOCK
LABEL BLOCK continue BLOCK
```

§15. 4 perlの演算子と優先順位

結合規則と優先順位(高い順)

左結合	項　リスト演算子　(左方向に対して)
左結合	->(アロー演算子)
非結合	++ --(インクリメント／デクリメント)
右結合	**(指数演算子)
右結合	! ~ \(リファレンス) 単項の+ 単項の-
左結合	=~ !~(パターンマッチに拘束)
左結合	* / % x(繰り返し演算子)
左結合	+ - .(文字列を連結)
左結合	<< >>(シフト)
非結合	名前付き単項演算子(括弧が省略可能な関数)
非結合	< > <= >= lt gt le ge(数値比較と文字列比較)
非結合	== != <=> eq ne cmp(数値比較と文字列比較)

§15. 4 perlの演算子と優先順位(続き)

左結合	&(ビット毎の論理積)
左結合	| ^(ビット毎の論理和, 排他論理和)
左結合	&&(最後に評価された値を返す点がCとは異なる)
左結合	||(最後に評価された値を返す点がCとは異なる)
非結合	. .(範囲演算子)
右結合	?:
右結合	= += -= *= などの代入演算子
左結合	, =>(Cと同じコンマ演算子, 目立つコンマ)
非結合	リスト演算子 (右方向に対して)
左結合	not(論理演算 !と同じ)
左結合	and(論理演算 &&と同じ)
左結合	or xor(論理演算 ||と同じ)

§15. 5 perlの正規表現

^	行の最初にマッチする
$	行の終わりにマッチする
.	(改行以外の) すべての文字にマッチする
*	0 回以上にマッチ
+	1 回以上にマッチ
?	1 回または 0 回にマッチ
[]	文字クラス
()	グループ化
|	選択
\	次のメタ文字をクォートする
{n}	ちょうど n 回にマッチ
{n,}	n 回以上にマッチ
{n,m}	n 回以上 m 回以下にマッチ
\w	「単語」の構成文字 (英数字と "_") にマッチ
\W	単語の構成文字以外にマッチ
\s	空白文字にマッチ
\S	空白文字以外にマッチ
\d	数字にマッチ
\D	数字以外にマッチ

§15. 5 perlの正規表現(続き)

\b	単語の境界にマッチ
\B	単語の境界以外にマッチ
\A	文字列の最初にのみマッチ
\Z	文字列の最後にのみマッチ
\G	前回の m//g が終わったところにのみマッチ
\t	タブ
\n	改行
\r	復帰
\f	改ページ
\a	アラーム (ベル)
\e	エスケープ
\033	8 進数で表した文字
\x1b	16 進数で表した文字
\c[	コントロール文字
\l	次の文字を小文字にする
\u	次の文字を大文字にする
\L	\E まで小文字にする
\U	\E まで大文字にする
\E	変更の終わり
\Q	\E まで正規表現のメタ文字をクォートする

§15. 6 perlによる加工

/bin/df -ik の出力

```
Filesystem  1K-blocks     Used     Avail Capacity iused    ifree %iused
Mounted on
/dev/da0s1a    516062    70112    404666     15%    3029    61865     5%  /
mfs:27         515598       12    515586      0%      10    64500     0%  /tmp
/dev/da0s1e   4129310  1859550   1939416     49%   73226   424308    15%  /usr
/dev/da0s1f 930509708 55966928 874542780      6%  160623 14409359     1%  /usr0
procfs              4        4         0    100%      69     6095     1%  /proc
```

加工するには

```
デバイス          容量KB       使用量/i            残量/i          率/i   DIR
/dev/da0s1a     516062     70112/3029      404666/61865     15%/5%  /
mfs:27          515598        12/11        515586/64499      0%/0%  /tmp
/dev/da0s1e    4129310   1859550/73226    1939416/424308    49%/15% /usr
/dev/da0s1f 930509708  55966928/160623 874542780/14409359   6%/1%  /usr0
procfs              4          4/71             0/6093     100%/1%  /proc
```

§15. 6 perlによる加工(続き)

```
$TEMP  = "/tmp/df-$$";
$DF    = "/bin/df";

system "$DF -ik @ARGV > $TEMP";

if (!open(INFH, $TEMP)) {
  die "can't open $TEMP";
}

while (<INFH>) {
  if (/^(Filesystem)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)
    \s+(\S+)\s+(Mounted on)$/) {
    printf "デバイス        容量KB    使用量/i       残量/i      率/i   DIR\n";
  }
  elsif (/^(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)\s+(\S+)
    \s+(\S+)\s+(\S+)$/) {
    printf "%-11s%10s%10s/%-6s%10s/%-9s%4s/%-3s %-9s\n",
    $1, $2, $3, $6, $4, $7, $5, $8, $9;
  }
}

close(INFH);
unlink $TEMP;
exit 0;
```

§15. 7 メールに対する自動応答

メールに対して自動的に応答メッセージを返信する例

- MH(Message Handler)に含まれるslocal(special local mail delivery)機能を用いる場合
  - .forwardに以下のように書く

    ```
    "| /usr/local/lib/mh/slocal -user nakashim"
    ```

  - .maildeliveryに以下のように書く

    ```
    Subject ip-        | A "/usr/home/nakashim/Bin/receipt"
    Subject unix1-     | A "/usr/home/nakashim/Bin/receipt"
    Subject unix2-     | A "/usr/home/nakashim/Bin/receipt"
    Subject db1-       | A "/usr/home/nakashim/Bin/receipt"
    Subject db2-       | A "/usr/home/nakashim/Bin/receipt"
    default -          > ? "/var/mail/nakashim"
    ```

- receiptの標準入力にメールが渡される
  - シェルスクリプト／C言語／Perlなど，何で書いても構わない

§15. 7 メールに対する自動応答(つづき)

- perlを用いて，receiptを作成する

  標準入力に対し，MIMEエンコードされている部分を元の文字に戻し，日本語コードを7bit-JISコードに統一する.
  変換結果は，/tmp/receipt0-$$に格納しておく. $$は一意に付けられる番号.

```
#!/usr/local/bin/perl
open(FILE0, "|/usr/local/bin/nkf -m -j>/tmp/receipt0-$$");
while (<>) {
    print FILE0;
}
close FILE0;
```

改めて，/tmp/receipt0-$$を読み出し用にOPEN.
続いて，/tmp/receipt1-$$を書き込み用にOPEN.（書き込み時に日本語コードを7bit-JISコードに統一する. ）

```
open(FILE0, "</tmp/receipt0-$$");
open(FILE1, "|/usr/local/bin/nkf -j>/tmp/receipt1-$$");
```

§15. 7 メールに対する自動応答(つづき)

空行を検出するまでがヘッダ部分($parse_headerにて区別する)
chopは行末の改行文字を削除する
$'はパターンマッチした部分以降の文字列を表す

```
$parse_header = 1;
while (<FILE0>) {
   chop;
   if (/^Date:\s*/) {
       if ($parse_header == 1) { $Date = $'; }
       else { print FILE1; print FILE1 "\n"; }
   }
   elsif (/^From:\s*/) {
       if ($parse_header == 1) { $From = $'; }
       else { print FILE1; print FILE1 "\n"; }
   }
   elsif (/^Return-Path:\s*/) {
       if ($parse_header == 1) { $ReturnPath = $'; }
       else { print FILE1; print FILE1 "\n"; }
   }
   elsif (/^Subject:\s*/) {
       if ($parse_header == 1) { $Subject = $'; }
       else { print FILE1; print FILE1 "\n"; }
   }
   elsif (/^To:\s*/) {
       if ($parse_header == 1) { $To = $'; }
       else { print FILE1; print FILE1 "\n"; }
```

§15. 7 メールに対する自動応答(つづき)

```
            elsif (/boundary=/) {
                if ($parse_header == 1) { $boundary = $'; }
                else { print FILE1; print FILE1 "\n"; }
            }
            elsif (/^\s*$/) {
                if ($parse_header == 1) {
                    $parse_header = 0;
/* ヘッダ部分の解析が終了. $parse_headerを0にして，収集した情報から判断する. 次頁にて説明 */
                }
                else { print FILE1; print FILE1 "\n"; }
/* 本文中の空行はそのまま出力 */
            }
            else {
                if ($parse_header == 0){print FILE1;print FILE1 "\n"; }
/* 空行以外は，ヘッダ部分であれば無視. そうでなければ本文なのでそのまま出力 */
            }
        }

        close FILE0;
        close FILE1;
        system "/usr/local/bin/mhmail -s auto-receipt \"$ReturnPath\" < /tmp/receipt1-$$";
        system "/usr/local/bin/mhmail -s auto-receipt \"nakashim\@econ.kyoto-u.ac.jp\" < /tmp/receipt1-$$";
        unlink("/tmp/receipt0-$$");
        unlink("/tmp/receipt1-$$");
        exit 0;
```

§15. 7 メールに対する自動応答(つづき)

返信内容を作成する部分

```
$Subject     = "None"           unless ($Subject);
$Date        = "Unknown"        unless ($Date);
$From        = "Not Specified" unless ($From);
$ReturnPath  = "Not Specified" unless ($ReturnPath);
if ($From !~ /media.kyoto-u.ac.jp/) {
  printf FILE1 ("このレポートの送信者は\"%s\". \n", $From);
  print  FILE1 "メディアセンタのアドレス(*.media.kyoto-u.ac.jp)以外は無効\n";
  print  FILE1 "受け取りは拒否されました. \n";
}
elsif ($Subject =~ /db2-[0-9]{10}/) {
  print  FILE1 "以下のレポート(データベース構築論2)を受信しました. \n";
  print  FILE1 "ただし，現在のところレポート課題はありません. \n";
}
else {
  printf FILE1 ("このレポートの題名は\"%s\". \n", $Subject);
  print  FILE1 "受取りは拒否されました. \n";
}
print  FILE1 "=========================================\n";
printf FILE1 "Subject:    %s\n", $Subject;
printf FILE1 "Date:       %s\n", $Date;
printf FILE1 "From:       %s\n", $From;
printf FILE1 "ReturnPath:%s\n", $ReturnPath;
print  FILE1 "-----------------------------------------\n";
```

§15. 8 マークシート集計

マークシート読み取り装置からの入力

- 1枚の内容が1行のテキストとして出力される
- 5択×50項目であれば0～5の数字が50桁並ぶ(区切りはタブ)
- 各項目に複数マークをした場合は0になる
- 3枚読み込んだ場合の例

```
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 …
0 0 1 1 0 0 2 3 0 0 0 0 0 0 0 0 0 0 …
0 0 0 0 0 0 0 0 0 0 1 1 2 5 0 0 0 0 …
```

- 50人の審査対象に，1:合格，2:不合格　をつけるために使う
- 50人の候補者から，数名を選択するために使う

　2つの機能を1つのプログラムで実現する

§15. 8 マークシート集計(つづき)

候補者リスト，使い方の表示，配列初期化

```
#!/usr/local/bin/perl
$maxmem = 5;
@member = ('----  ',
           '中島  ',
           '松井  ',
           '川端  ',
           '佐藤  ',
           '鈴木  ',
);
if ($#ARGV != 1) {
    printf "使い方: %s <type> <ファイル>\n", $PROG;
    printf "         type=0: 可否\n";
    printf "         type=n: n人選出\n";
    exit 1;
}
$type = $ARGV[0];
if (!open(INFH, $ARGV[1])) {
    die "can't open $ARGV[1]";
}
for ($i = 1; $i <= $maxmem; $i++) {
    $sum1[$i] = 0;
    $sum2[$i] = 0;
}
```

§15. 8 マークシート集計(つづき)

入力データを配列に読み込む

```
$pages = 0;
$illegal = 0;
while (<INFH>) {
    chop;
    @line = split(/\t/); /* 1行を読み込み，タブごとに配列lineの各要素に格納 */
    for ($i = 0; $i <= 5; $i++) { $check[$i] = 0; }
    for ($i = 1; $i <= $maxmem; $i++) { $check[$line[$i-1]] ++; }
      /* 0～5の数値のいずれであるかを配列checkに累積していく */
    if ($check[3] + $check[4] + $check[5] > 0) { $illegal++; }
      /* 3～5の数値があった場合，無効票として計上 */
    elsif ($type>0 && ($check[1]>$type || $check[2]>0)) { $illegal++; }
      /* N名選出なのに，1または2のマークがNを超えている場合，無効票として計上 */
    else {
        for ($i = 1; $i <= $maxmem; $i++) {
            $sum1[$i] += ($line[$i-1] == 1)?1:0;
            $sum2[$i] += ($line[$i-1] == 2)?1:0;
        }
      /* 正しいカードであれば集計に加える */
    }
    $pages++;
}
printf "投票総数 %2d\n", $pages;
printf "無効総数 %2d\n", $illegal;
```

§15. 8 マークシート集計(つづき)

可否投票か選出かによって処理を分ける

```
if ($type == 0) { /* 可否投票 */
    for ($i = 1; $i <= 20; $i++) { /* 最大20人に制限(画面サイズの都合) */
        printf "No.%2d  ", $i;
        print (($sum1[$i]>$pages*2/3)?"●":"-"); /* 2/3を超える可にて合格 */
        printf "可%2d ", $sum1[$i];
        printf "否%2d ", $sum2[$i];
        printf "白%2d\n", $pages - $illegal - $sum1[$i] - $sum2[$i];
    }
}
else { /* N人選出 */
    for ($i = 1; $i <= $maxmem; $i++) {
        push(@result, sprintf("%2d %s\n", $sum1[$i], $member[$i]));
          /* 名前を含む文字列として結果を配列resultに格納 */
    }
    @top = reverse sort @result; /* 大きい順に並べ替え */
    for ($i = 0; $i < 20; $i++) {
        print (($i<$type)?"●":"-");
        print $top[$i];
    } /* 当選者に● */
}
close(INFH);
exit 0;
```

今日はここまで