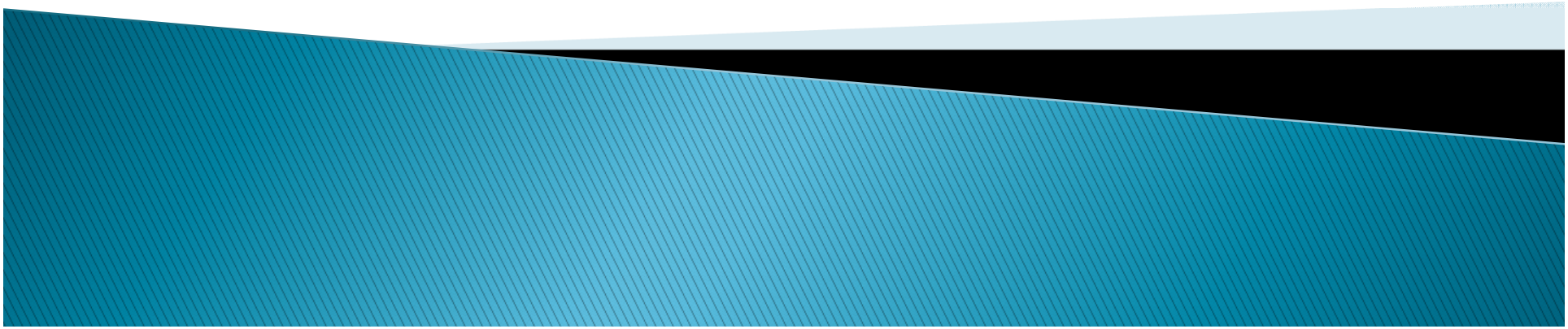
SAS® Perl Regular Expressions

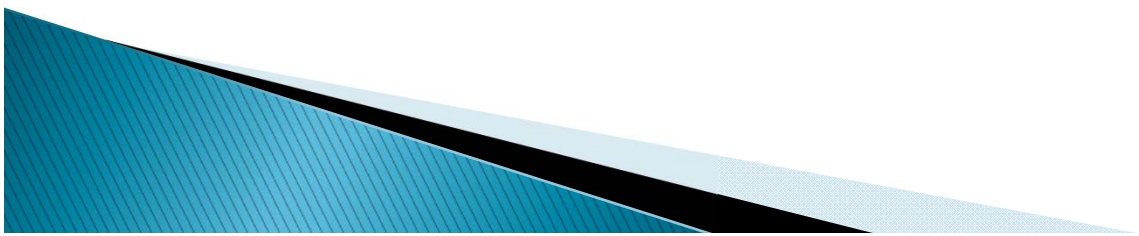
Mike Atkinson
Acko Systems Consulting Inc

May 10, 2016



Perl regular expressions (in SAS)

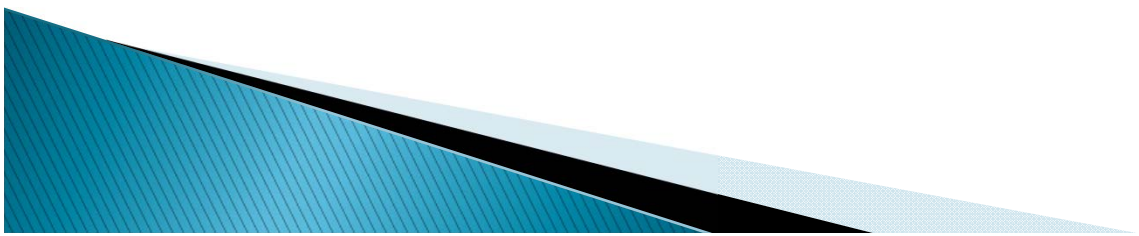
- Are basically another programming language
- Designed to work with text strings, performing pattern matching
- Can validate the format of various types of strings
- Can use pattern matching to modify strings, such as changing the case (upper or lower)



Two useful functions

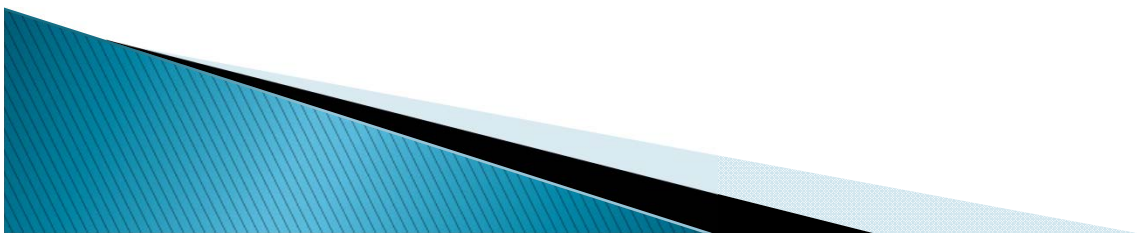
- prxmatch
- prxchange

  - prxparse
  - prxposn
  - prxnext
  - prxdebug
  - prxfree



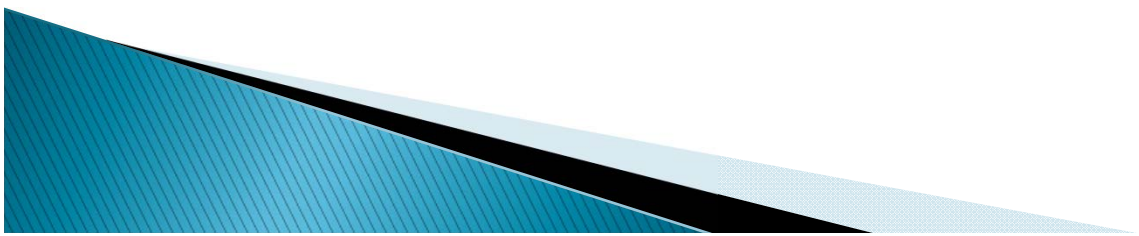
Metacharacters

- \d Matches a digit (between 0 and 9)
- \w Matches a word character (letter, number, or underscore)
- \s Matches a space
- \S Matches a non-space character
- ^ (or \A) Match beginning of string
- $ (or \Z) Match end of string
- \b Matches the boundary between a word and a non-word position
- …and others



Brackets

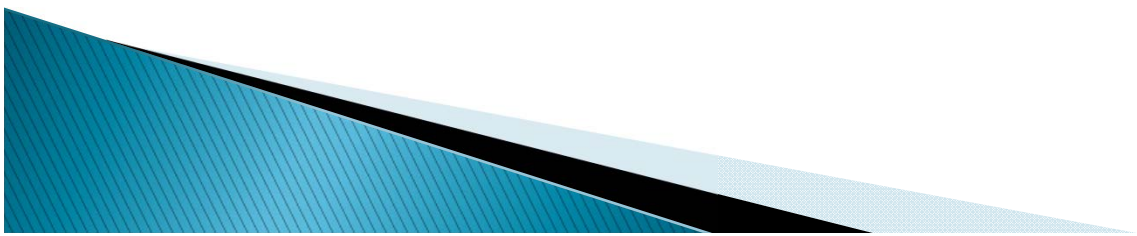
- []
  - Match any of a set of characters
  - Can specify a range, such as [a-z]
  - Or, don't match any of the characters, e.g, [^cdef]
- {}
  - Indicate number of matches, examples next slide
- ()
  - Group characters into a item
  - Perform "capture" of the matched text
  - Can also perform zero-width matching (with ?)



Number of times to match

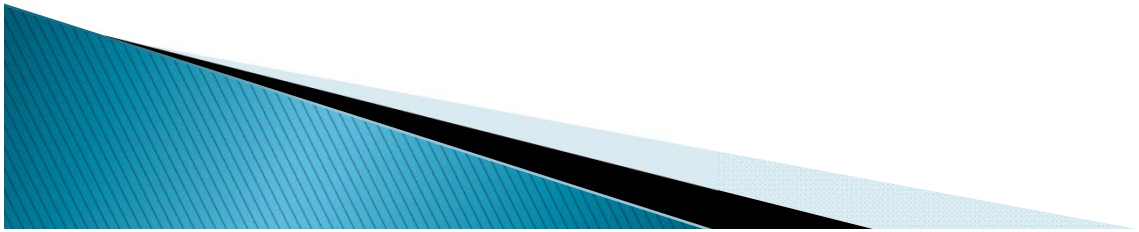
- * Match zero, one, or more occurrences
- + Match one or more
- {n} Match exactly n (where n is a number)
- {n,m} Match between n and m occurrences

- E.g. A* matches ‘A’, ‘AA’, ‘AAA’, and even nothing (‘’)
- A{3} only matches ‘AAA’



Vertical Bar for OR

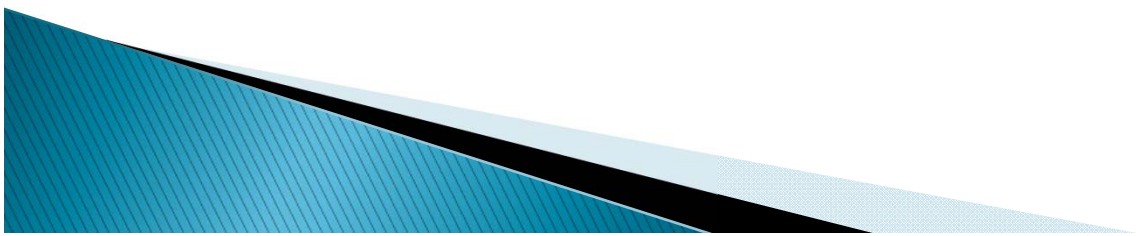
- () with |
  - works like an Or condition
- E.g.
  - (Jim|Bob|Steven) matches any one of the three names



Case Insentive

- /*reg-exp*/i 'i' indicates to ignore the case of letters when determining matches

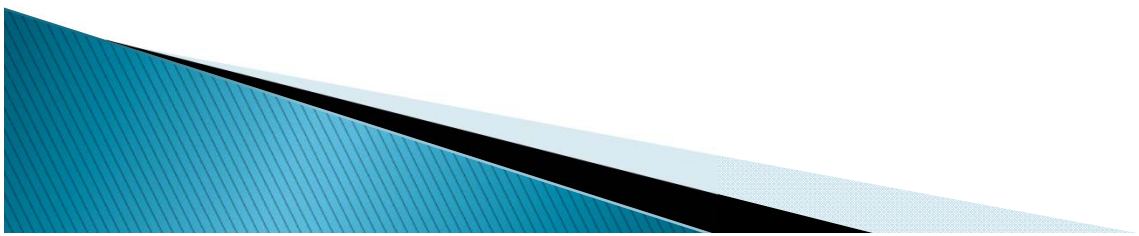
- E.g.
- '/john/i' matches 'JOHN', 'John', and 'john'
- '/john/' only matches 'john'



Matching with prxmatch

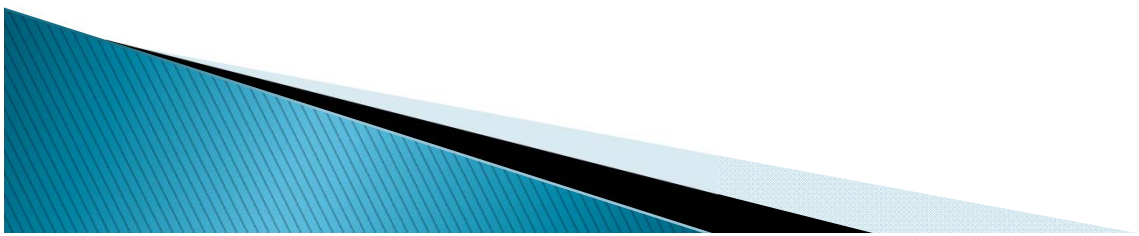
- Basic format
  - prxmatch('/*match*/', input_string)
  - Returns true (if found match) or false
- E.g.
  - If (prxmatch('/(eleph){0,1}ant/i', animal)) then …

Q: will this match "pheasant"?



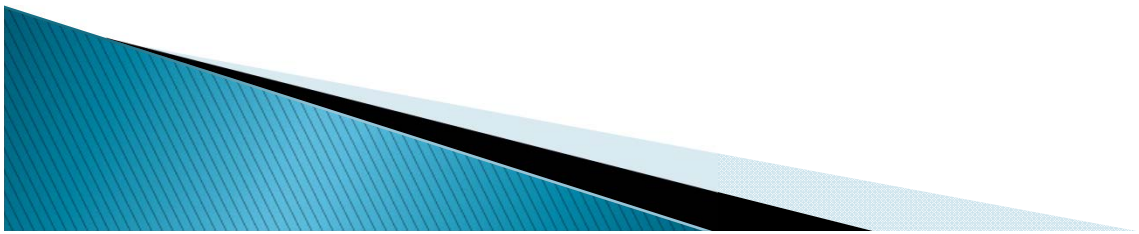
Upper and Lower Case

- Within a substitution, can set the case of the result
  - \u Changes a single character to upper case
  - \l Changes a single character to lower case
  - \U Changes to upper case until \E
  - \L Changes to lower case until \E
  - \E Terminate \U or \L



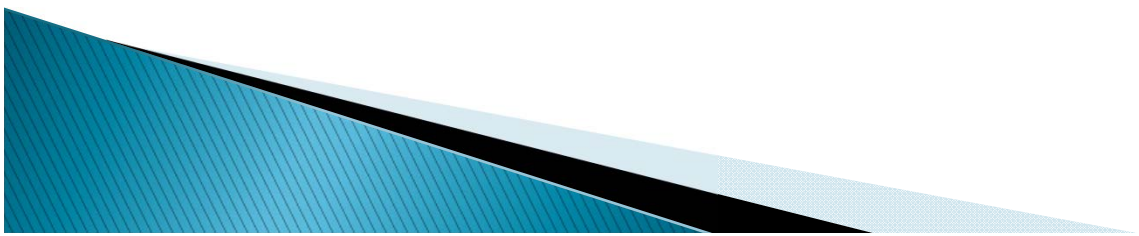
Reference matched text

- \n Reference matched item (that was in brackets) later within a regular expression
- &n Reference matched item within a substitution regular expression



Substitution with prxchange

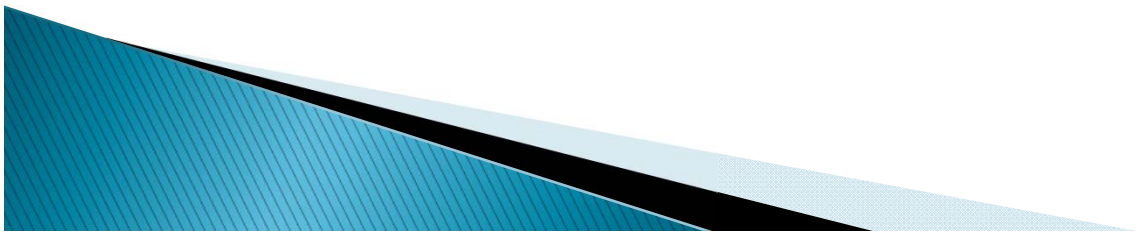
- Basic format:
  - prxchange('s/*match*/*change-to*/', num_times, input_string)
  - Returns modified string as the result
  - Num_times is usually 1 or -1 (for unlimited)
- E.g.
  - dr_name = prxchange('s/\bdr\b/Dr/i', 1, dr_name);



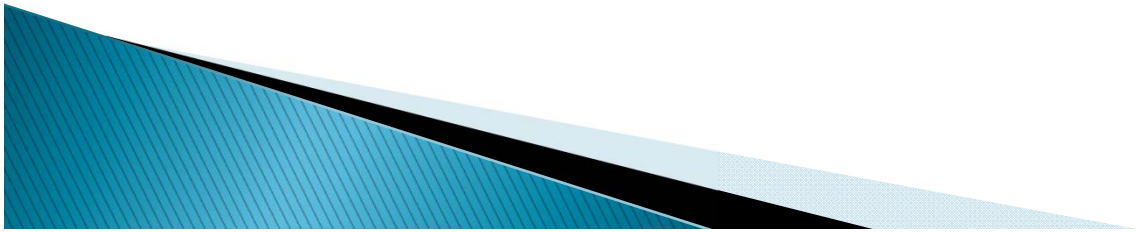
```
* Read some valid and invalid BC PHNs, stored in character format;
* (valid BC PHNs are 10 digits, starting with a '9');

data good_bad_and_ugly_PHNs;
    infile cards4;
    input phn_char $20.;
cards4;
1234567890      does not start with 9
9123456789      GOOD
9191919191      GOOD
91234567890     too long
912345678       too short
9A23456789      contains non-digit
912345 678      contains space
;;;;
run;


* Which values will this have a problem with?;
* What happens for the 11-digit number?;
* Plus, this causes warning messages in the log for the last two;

data convert_phns_1;
    set good_bad_and_ugly_PHNs;
    phn = input(phn_char, 10.);
run;
```

```
* Ensure valid BC phn format (10 digits starting with a '9');
* By checking the format is correct before applying the input function, ;
* we avoid warning messages, and ensure non-BC PHNs are set to missing;

data convert_phns_2;
    set good_bad_and_ugly_PHNs;

    * 10 digits starting with a 9, and no extraneous info on the record;
    if (prxmatch('/\A9\d{9}\s*\Z/', phn_char)) then
        phn = input(phn_char, 10.);
    else
        call missing(phn);
run;
```

Example #1, slide 2 of 2



```
* Demonstrate name formatting;

* Want to put a comma after last name, and capitalise each "Name";
* Note 1: "de Wet" is a last name;
* Note 2: want capitalisation as in MacDonald and McDonald;

data prac_names;
    infile cards4;
    input pracnm $40.;
cards4;
SMITH MARK
JONES SUSAN
DE WET JACOB
MACDONALD ANGUS
MCDONALD MARGARET
;;;;
run;
```



Example #2, slide 1 of 2

```
data prac_names_2;
    set prac_names;

    * make the first character of each word/name upper case, ;
    * remainder lower case;
    pracnm = prxchange('s/\b(\w+)\b/\u\L$1\E/i', -1, pracnm);

    * add a comma after last name & fix de Something;

* this one would not work: missing i for case insensitive matching: ;
* "de" does not match "De", unless 'i' added at end;
* pracnm = prxchange('s/\A(de\s){0,1}(\w+)\b/\L$1\E$2,/', 1, pracnm);

    * Put comma after last name, which is the first "word", ;
    * unless it starts with "de";
    pracnm = prxchange('s/\A(de\s){0,1}(\w+)\b/\L$1\E$2,/i', 1, pracnm);

    * Fix McDonalds;
    pracnm = prxchange('s/\Am(a{0,1})cd/M$1cD/i', 1, pracnm);
run;
```

Example #2, slide 2 of 2