



**THE
POWER
TO KNOW[®]**

A Sampler of What's New in Base SAS 9.2

Jason.Secosky@sas.com

Caramel, Nut, or Truffle?

- DATA Step
 - INPUT with String Delimiter
 - Current Input Data Set Name
 - Faster IN Operator
 - Calling Java
- PROC SQL
 - Faster SELECT DISTINCT
 - Faster COUNT DISTINCT
- PROC SORT Linguistic Ordering
- Encrypted Macro Storage

String Delimiter

- "Top 10" SASware Ballot Item
- Difficult to parse multi-character delimiters
- INFILE DLMSTR=
- Works like DLM=

String Delimiter

```
data _null_;  
  infile mydata dlmstr='--' dsd;  
  input x y name $ addr:$16.;  
  put x= y= name= addr=;  
datalines;  
1.25--2.25--"Name"--"12 Main St"  
;  
  
x=1.25 y=2.25 name=Name addr=12 Main St
```

SET with INDSNAME=

- "Top 10" SASware Ballot Item
- Place current data set name into a variable
- SET with INDSNAME = *var*

SET with INDSNAME=

```
data results;
```

```
    set gas_price_option
```

```
        gas_rbid_option
```

```
        coal_price_forward
```

```
        coal_rbid_forward
```

```
        indsname = cur_dataset;
```

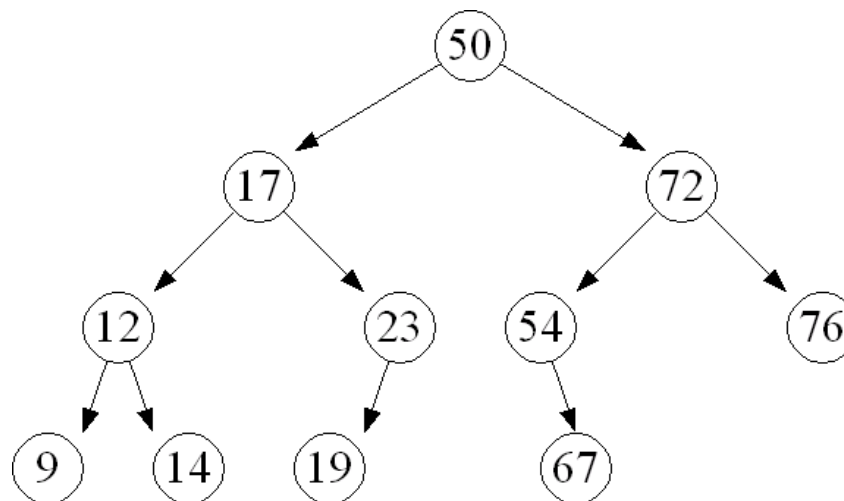
```
    . . .
```

```
run;
```

DATA Step IN Operator

- IN operator used simple array search
- Now, IN searches a binary tree

9	12	14	17	19	23	50	54	67	72	76
---	----	----	----	----	----	----	----	----	----	----



DATA Step IN Operator

```
data _null_;  
  set customers;  
  if state in ('NC', 'SC', 'GA',  
              'TN', 'FL');  
  
run;
```



DATA Step JavaObj

- "Compute" methods written in Java
- Call from DATA Step with JavaObj
- Methods like Java Native Interface (JNI)
- Uses dot syntax
- Production in SAS 9.2
- Experimental in SAS 9.1.3

DATA Step JavaObj

```
data _null_;  
  declare javaobj jo("myclass");  
  jo.callDoubleMethod("compute",  
                      x, y, z, res);  
  . . .
```



PROC SQL

- Faster: SELECT DISTINCT
- Faster: COUNT(DISTINCT var)
- Hash Join Performance
- Row Counts from DBMSes

SELECT DISTINCT

```
proc sql;  
    create table distinct_values as  
    select distinct month, year  
    from customer_info;  
quit;
```



COUNT DISTINCT

```
proc sql;  
    create table count_distinct as  
    select count(distinct customer_id)  
    from customer_info;  
quit;
```



Linguistic Sorting

Obs	Binary	Dictionary	Linguistic
1	MacCarty	macabre	macabre
2	Macintosh	MacCarty	MacCarty
3	McCoy	McCoy	mace
4	macabre	mace	macintosh
5	mace	macintosh	Macintosh
6	macintosh	Macintosh	mammoth
7	mammoth	mammoth	McCoy

Linguistic Sorting

```
proc sort data=dictionary  
          out=linguistic  
          sortseq=linguistic;  
  
  by y;  
  
run;
```

Encrypted Macro Storage

```
%macro mymac(x, y) / store secure;  
...  
%mend;
```