Top 10 SAS Efficiency Hacks

MWSUG

October 2024

Charu Shankar

SAS Education

Top 10 SAS Efficiency Hacks

Charu Shankar, SAS® Institute

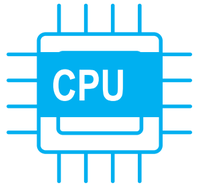
With a background in computer systems management. SAS Instructor Charu Shankar engages with logic, visuals, and analogies to spark critical thinking since 2007.

Charu curates and delivers unique content on SAS, SQL, Viya, etc. to support users in the adoption of SAS software.

When not coding, Charu teaches yoga and loves to explore Canadian trails with her husky Miko.

Agenda

Introduction to Efficiencies



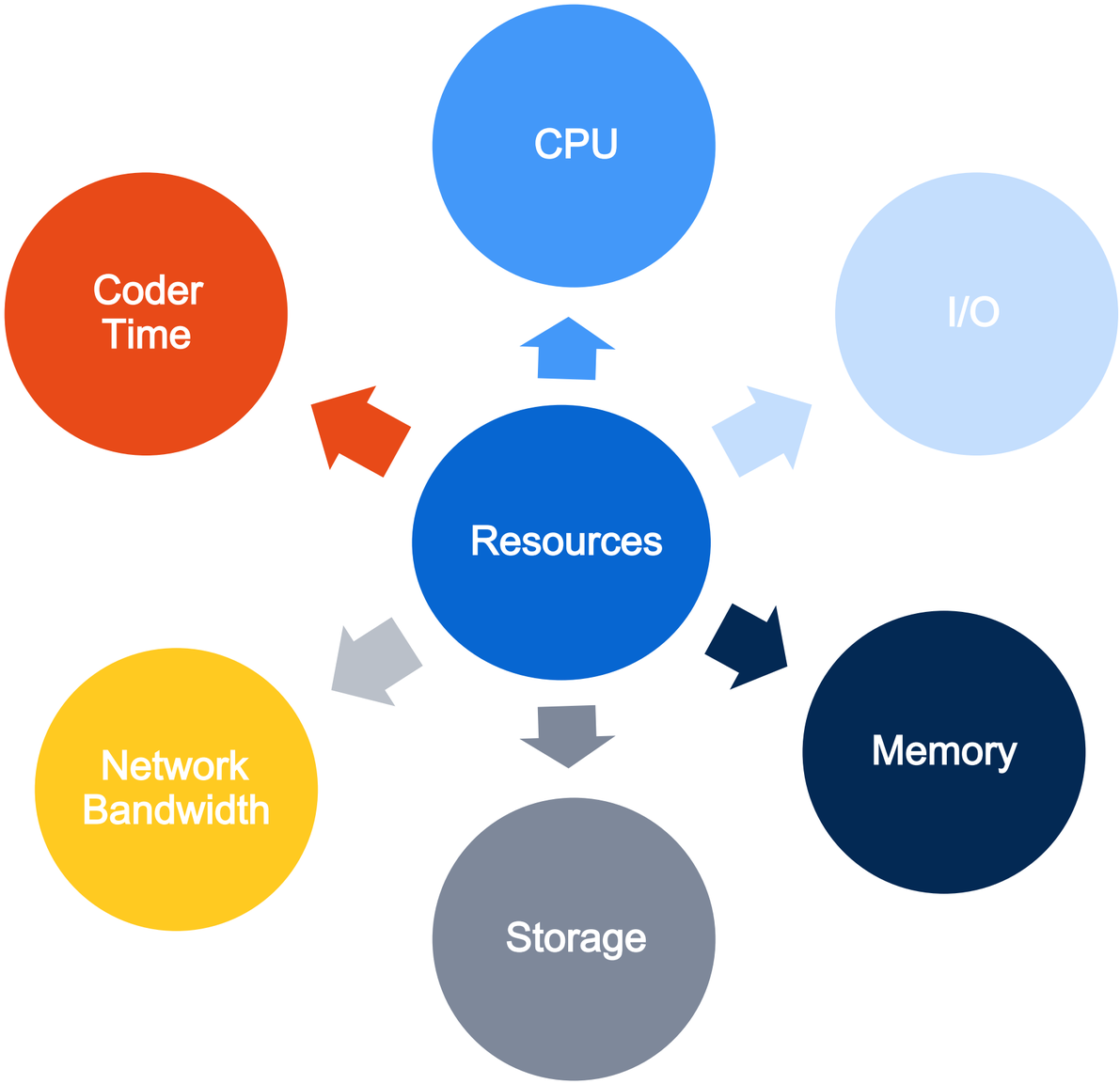
CPU Efficiency Hacks

I/O Efficiency Hacks

Space Efficiency Hacks

Handy Links

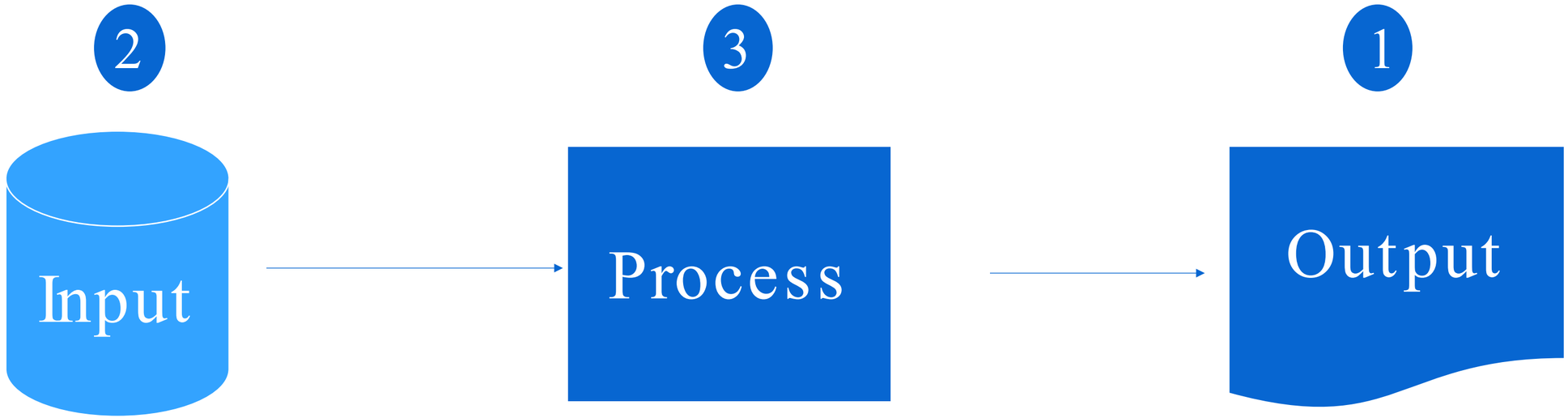
Introduction to Efficiencies



Some Definitions

Resource	Definition
CPU	Measure of time that the central processing unit takes to perform requested tasks like calculations, reading/writing data, conditional, iterative logic, etc.
I/O	Measure of the read-write operations performed when data and programs are moved from a storage device to memory(input) or from memory to storage or a display device(output)
Memory	Size of the work area required to hold executable program modules, data, buffers.
Storage	Amount of space required to store data on a disk or tape.
Coder time	Amount of time required for the programmer to write and maintain the program.
Network bandwidth	Amount of data that can pass through a network interface over time. This time can be minimized by performing as much of the subsetting and summarizing as possible on the data host before transferring the results to the loal computer. Network bandwidth is heavily dependent on network loads.

3 Questions To Ask Before Diving Into Data Work

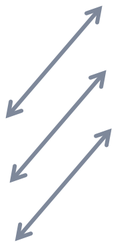


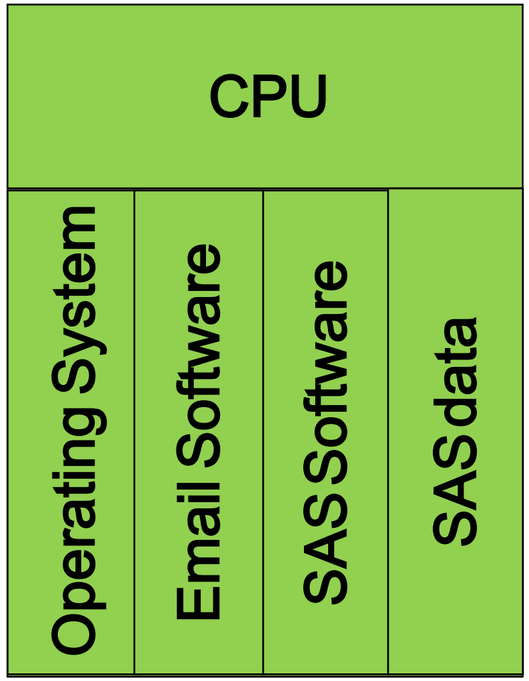
Computer Processing

You @ your terminal

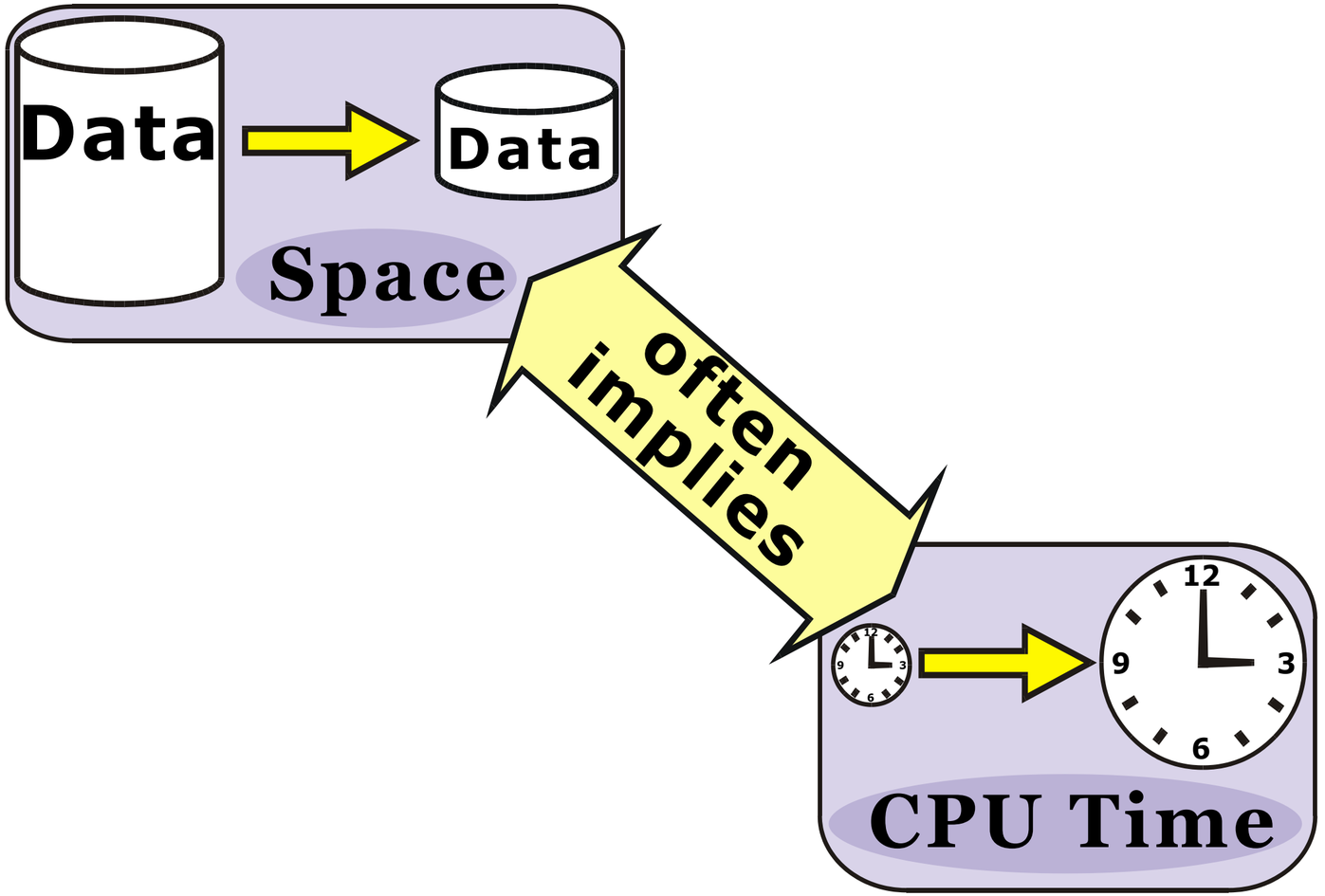
Applications

O/S
Email
SAS

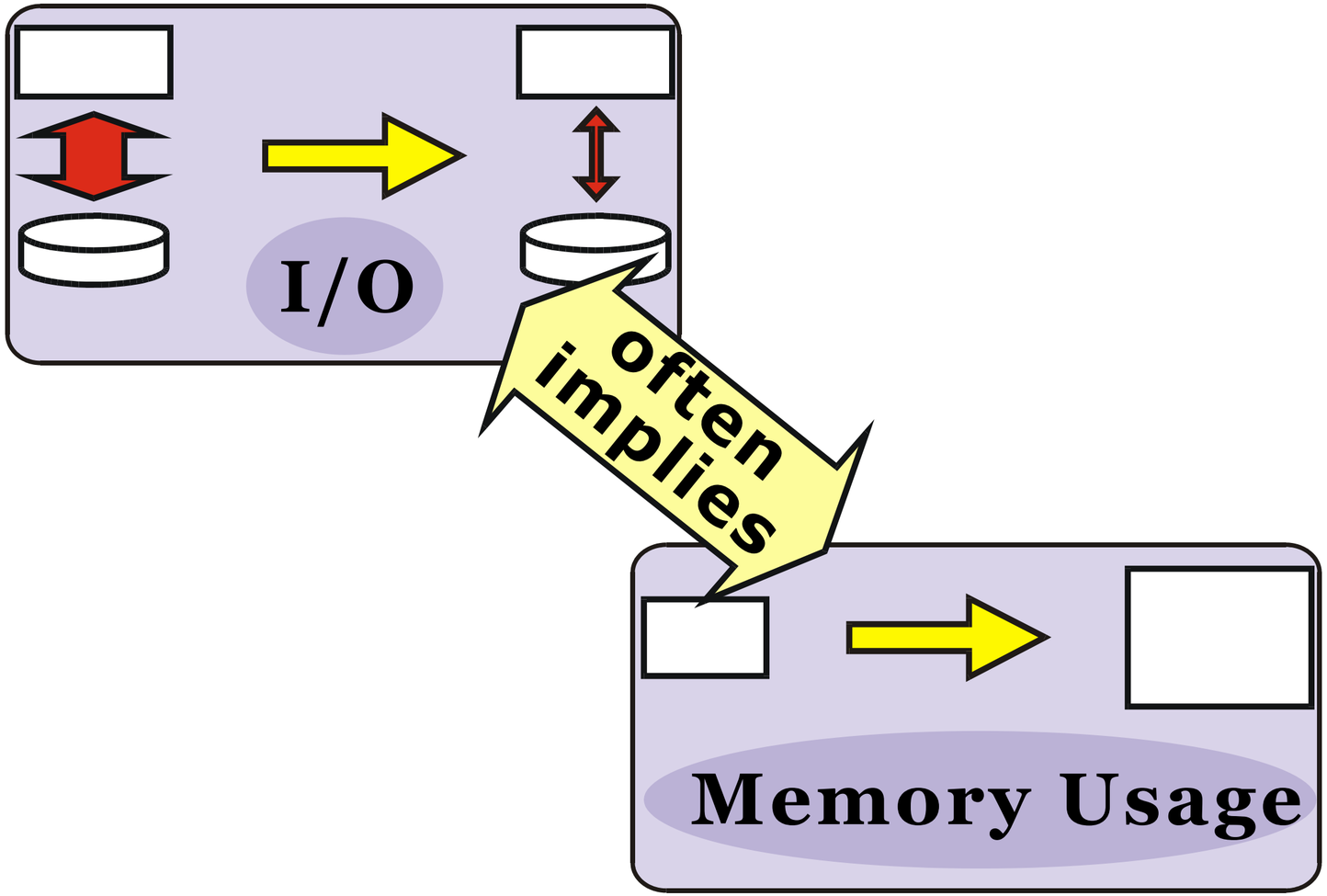




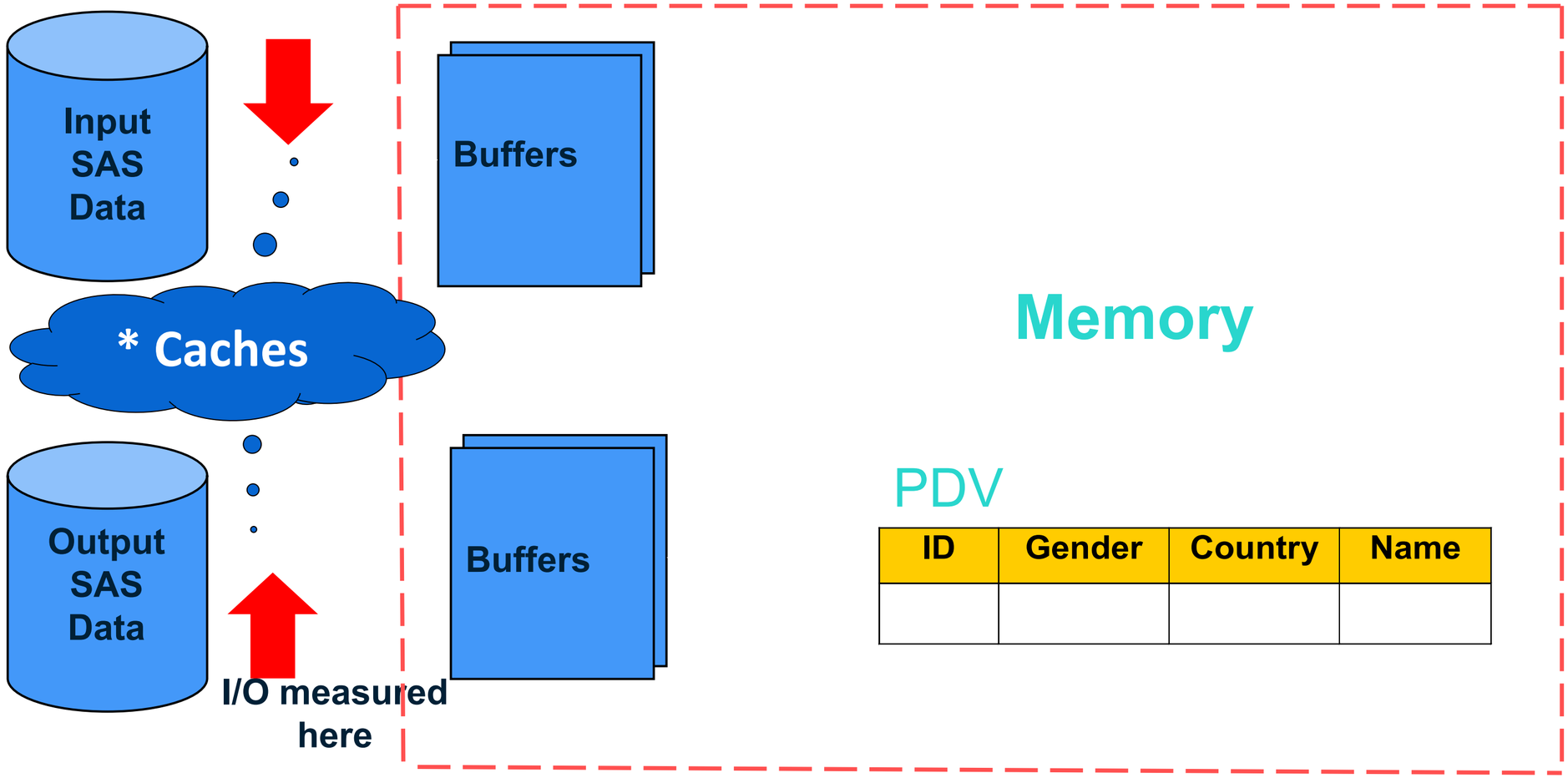
Understanding Efficiency Trade-offs



Understanding Efficiency Trade-offs



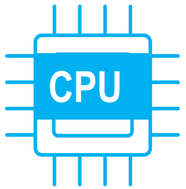
Where Is I/O Measured?



What's the only answer to "What's the best way to do this?"

It Depends!

Introduction to Efficiencies



CPU Efficiency Hacks

I/O Efficiency Hacks

Space Efficiency Hacks

Handy Links

1 CPU Hack - Data worker # 1 Rule
Plus - Bonus PROC SQL or SAS?

Poll

What is the Data Worker's #1 Rule?

Quiz

Know Thy Data

CESALES_ANALYSIS

Type:	TABLE
Default Action:	VIEWTABLE
Location:	CHOC.CESAI
Engine:	V9
Rows:	115928
Columns:	11
Created:	04Nov11:10:
Modified:	04Nov11:11:
Description:	

Column Name	Type	Length
customer_id	Text	4
Customer_N...	Text	16
Customer_T...	Text	13
Product_ID	Text	7
Product_Name	Text	26
Category	Text	11
SubCategory	Text	16
Month	Num...	8
Year	Num...	8
Total_Cases	Num...	8
Total_Sales	Num...	8

Ceorder_info

Type:	TABLE
Default Action:	VIEWTABLE %8b.
Location:	Choc.Ceorder_info
Engine:	V9
Rows:	114488
Columns:	8
Created:	02Nov11:23:20:54
Modified:	02Nov11:23:20:54

Column Name	Type	Length
Customer_ID	Text	4
Customer_N...	Text	16
Customer_T...	Text	13
Product_ID	Text	7
Month	Num...	8
Year	Num...	8
Total_Cases	Num...	8
Total_Sales	Num...	8

Choc.Ce_product_info Properties

General | Details | Columns | Indexes | Integrity | Passwords

Ce_product_info

Type:	TABLE
Default Action:	VIEWTABLE %8b."%s
Location:	Choc.Ce_product_info
Engine:	V9
Rows:	40
Columns:	4
Created:	05Nov11:13:13:52
Modified:	05Nov11:13:13:52

Column Name	Type	Length
Product_ID	Text	7
Product_Name	Text	26
Category	Text	11
SubCategory	Text	16

dec04sales - Notepad

File Edit Format View Help

```
Customer_ID,Customer_Name,Customer_Type,Product_ID,Product_Name,Category,SubCategory,Month,Year,Total_Cases,Total_Sale:
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
100,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,73,$569.40
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,40,$360.00
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,40,$360.00
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Choclate,Chocolate Dark,12,2004,40,$360.00
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,40,$360.00
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,40,$360.00
101,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,40,$360.00
102,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,20,$192.00
102,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,20,$192.00
102,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,20,$192.00
102,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,20,$192.00
103,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,22,$211.20
103,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,22,$211.20
103,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,22,$211.20
104,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,11,$118.80
104,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,11,$118.80
105,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,3,$30.60
106,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,10,$102.00
106,All-Mart,BigBoxRetail,101001,SM Dark Choc Bar,Chocolate,Chocolate Dark,12,2004,10,$102.00
```

Know Thy Data - Dictionary Tables

Dictionary tables to gather metadata can be accessed either through PROC SQL or SAS procedures/ data step code.

Which Technique Is Faster?

```
proc print data=sashelp.vcolumn label noobs;
      var libname memname name type length;
      where libname =MWSUG' and   upcase(name) contains 'ID';
run;
NOTE: There were 21 observations read from the data set SASHELP.VCOLUMN.
      WHERE (libname='MWSUG') and UPCASE(name) contains 'ID';
NOTE: PROCEDURE PRINT used (Total process time):
      real time           0.58 seconds
      user cpu time       0.03 seconds
      system cpu time     0.04 seconds
      memory              5254.75k
```

Why did searching the SASHELP library take so much longer?

```
proc sql;
      select libname, memname, name, type, length
            from dictionary.columns
                  where libname =MWSUG' and upcase(name) contains 'ID';
quit;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.04 seconds
      user cpu time       0.03 seconds
      system cpu time     0.00 seconds
      memory              5168.93k
      OS Memory           31340.00k
      Timestamp           07/17/2019 02:11:24 PM
```

2 CPU Hack - Boil Down Data

Technique 1
Subsetting IF Statement at Bottom of Step

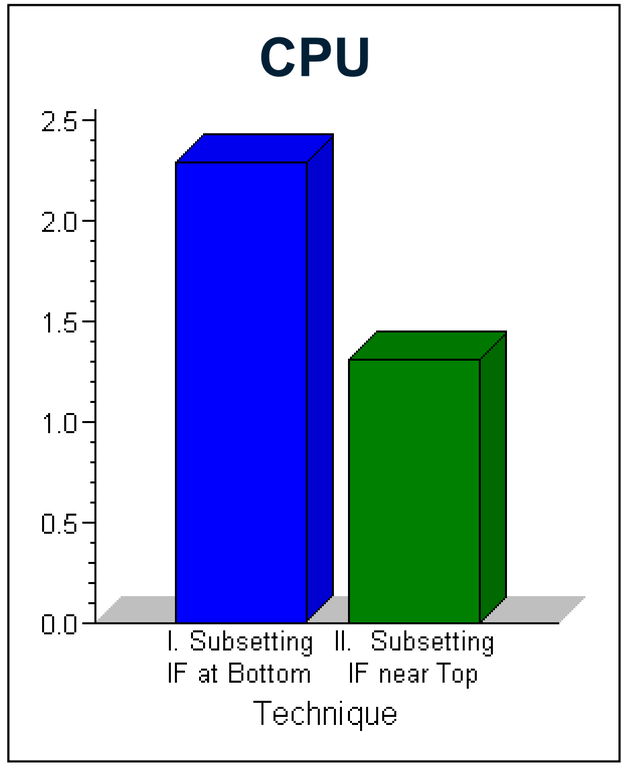
```
data totals;
    set MWSUG.cesales_analysis;
    do month=1 to 12;
        totcase + total_cases;
        totsales + total_sales;
    end;
    cust='******';
    custorder=1;
   if customer_type='Gourmet';
run;
```

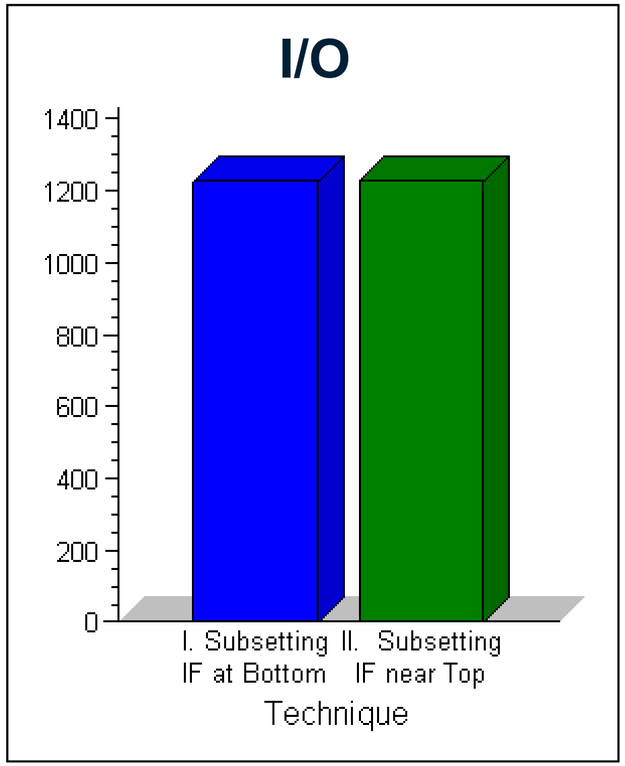
Technique 2
Subsetting IF Statement as High as Possible

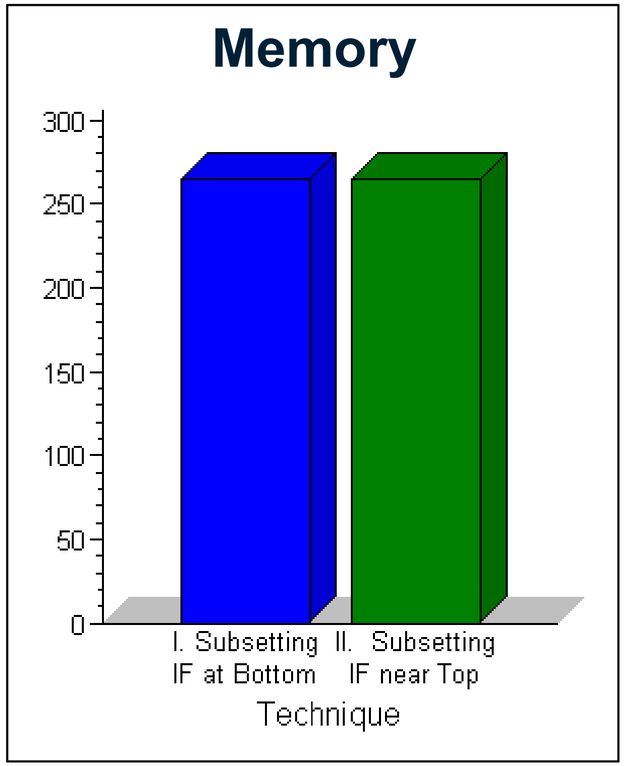
```
data totals;
    set MWSUG.cesales_analysis;
    do month=1 to 12;
        totcase + total_cases;
        totsales + total_sales;
    end;
    if customer_type='Gourmet';
    cust='*****';
    custorder=1;
run;
```

The Result- Comparing Techniques

Technique	CPU	I/O	Memory
I. Subsetting IF at Bottom	2.3	1226.0	265.0
II. Subsetting IF near Top	1.3	1226.0	265.0
Percent Difference	42.8	0.0	0.0



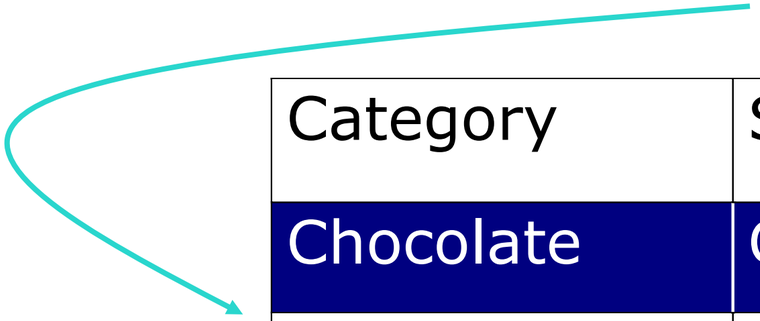




Where or IF – That is the next question?

Selecting Observations

We want to subset for **category**= “Chocolate“



Category	SubCategory	Product_Name
Chocolate	Chocolate Dark	SM Dark Choc Bar
Gummy	Gummy Sour	Gummy Lions Bag
Hard	Hard Sweet	Butterscotch Disks Bag
Sugar-Free	SF Chocolate	SF Jelly Beans Bag

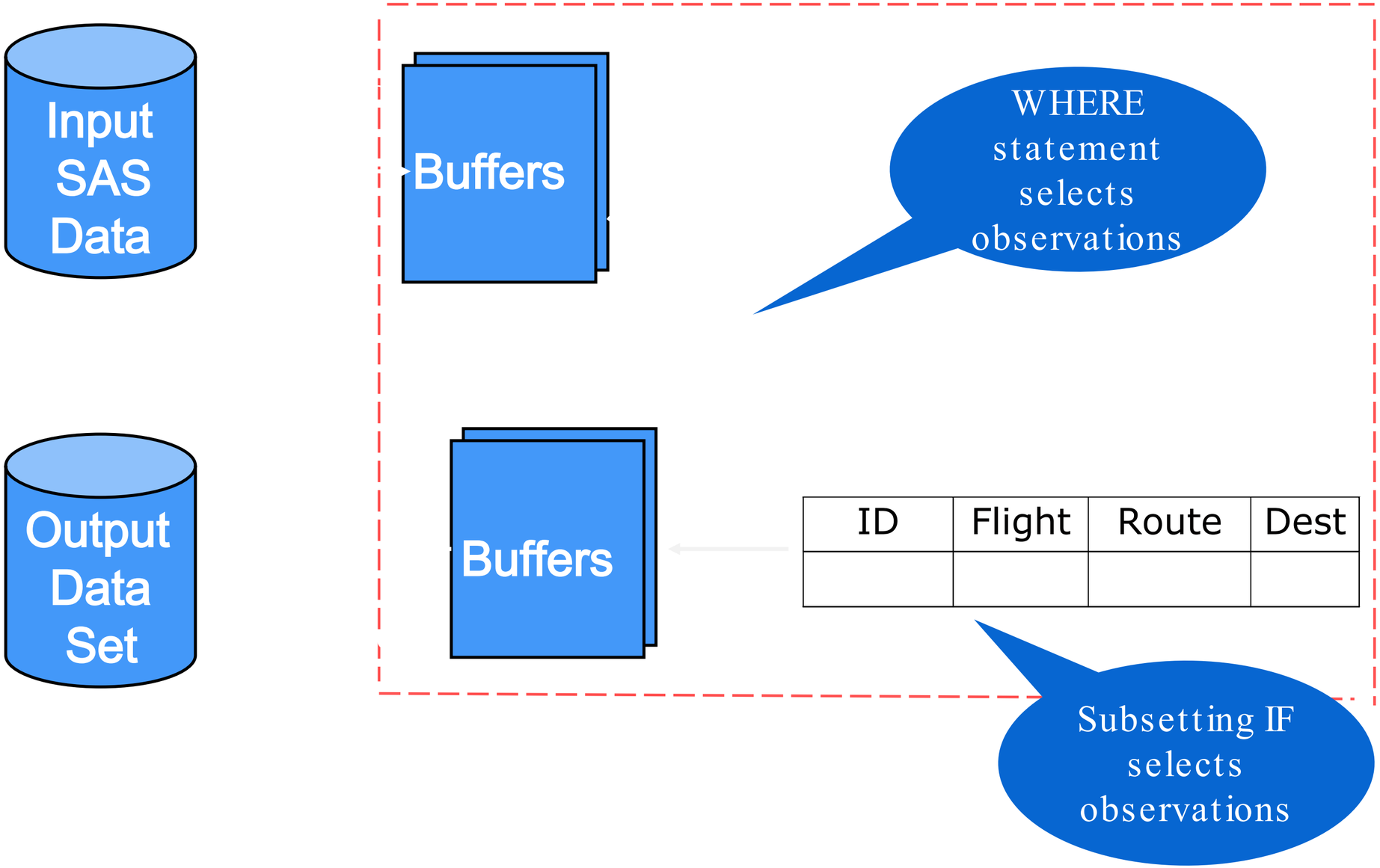
Subsetting IF Or The Where Clause?

Create a subset of the cesales_analysis dataset that contains data for Chocolate.

```
data chocolate;
     set MWSUG.cesales_analysis;
     if category='Chocolate';
Run;
NOTE: There were 115928
observations read from the data set
MWSUG.CESALES_ANALYSIS.
NOTE: The data set WORK.CHOCOLATE
has 50368 observations and 11
variables.
NOTE: DATA statement used (Total
process time):
real time                2.83 seconds
cpu time                 0.04 seconds
```

```
data chocolate;
     set MWSUG.cesales_analysis;
     where category='Chocolate';
Run;
NOTE: There were 50368 observations
read from the data set
MWSUG.CESALES_ANALYSIS.
WHERE category='Chocolate';
NOTE: The data set WORK.CHOCOLATE has
50368 observations and 11 variables.
NOTE: DATA statement used (Total
process time):
real time                2.26 seconds
cpu time                 0.06 seconds
```

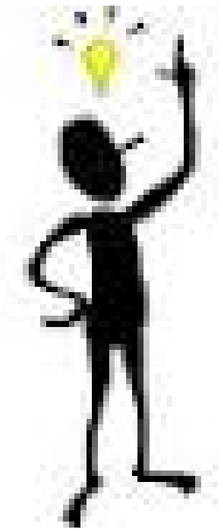
The Subsetting IF and WHERE Statements



Consider- When To Use Which One?

The WHERE clause Or The Subsetting IF
The answer lies in this question - do you want to subset existing or newly created obs?

Use the WHERE to subset by existing variables
Use the IF to subset by new variables



Did you know ? The WHERE clause is the same one used in SQL.

3 CPU Hack – Use Conditional Logic

IF-THEN/ELSE	Executes a SAS statement for observations that meet a specific condition
SELECT	Executes one of several statements or groups of statements

3.1 CPU Hack
Technique 1-Using Parallel IF Statements

Create variable named **var**, based on existing variable **var1**.

```
data _null_;
   length var $ 30;
   retain var2-var50 0 var51-var100 'ABC';
   do x=1 to 10000000;
      var1=10000000*ranuni(x);
      if var1>1000000 then var='Greater than 1,000,000';
       if 500000<=var1<=1000000 then var='Between 500,000 and
1,000,000';
       if 100000<=var1<500000 then var='Between 100,000 and 500,000';
       if 10000<=var1<100000 then var='Between 10,000 and 100,000';
       if 1000<=var1<10000 then var='Between 1,000 and 10,000';
       if var1<1000 then var='Less than 1,000';
   end;
run;
```

3.2 CPU Hack
Technique 2 - Using ELSE-IF Statements

```
data _null_;
   length var $ 30;
   retain var2-var50 0 var51-var100 'ABC';
   do x=1 to 10000000;
      var1=10000000*ranuni(x);
      if var1>1000000 then var='Greater than 1,000,000';
      else if 500000<=var1<=1000000 then var='Between 500,000 and 1,000,000';
      else if 100000<=var1<500000 then var='Between 100,000 and 500,000';
      else if 10000<=var1<100000 then var='Between 10,000 and 100,000';
      else if 1000<=var1<10000 then var='Between 1,000 and 10,000';
      else if var1<1000 then var='Less than 1,000';
   end;
run;
```

3.3 CPU Hack
Technique 3 - Using a SELECT Block

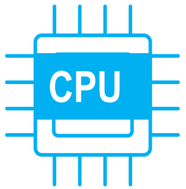
```
data _null_;
     length var $ 30;
     retain var2-var50 0 var51-var100 'ABC';
     do x=1 to 10000000;
          var1=10000000*ranuni(x);
          select;
               when (var1>1000000) var='Greater than 1,000,000';
               when (500000<=var1<=1000000) var='Between 500,000 and
1,000,000';
               when (100000<=var1<500000) var='Between 100,000 and 500,000';
               when (10000<=var1<100000)  var='Between 10,000 and 100,000';
               when (1000<=var1<10000)  var='Between 1,000 and 10,000';
               when (var1<1000 ) var='Less than 1,000';
          end;
     end;
run;
```

The Result-Let's Compare Techniques

Technique	CPU	I/O	Memory
I. Stand Alone IF Statements	15.9	6797.0	280.0
II. ELSE-IF Statements	9.7	6797.0	288.0
III. SELECT/WHEN Block	9.8	6797.0	288.0

The I/O for each technique is the same.

Introduction to Efficiencies



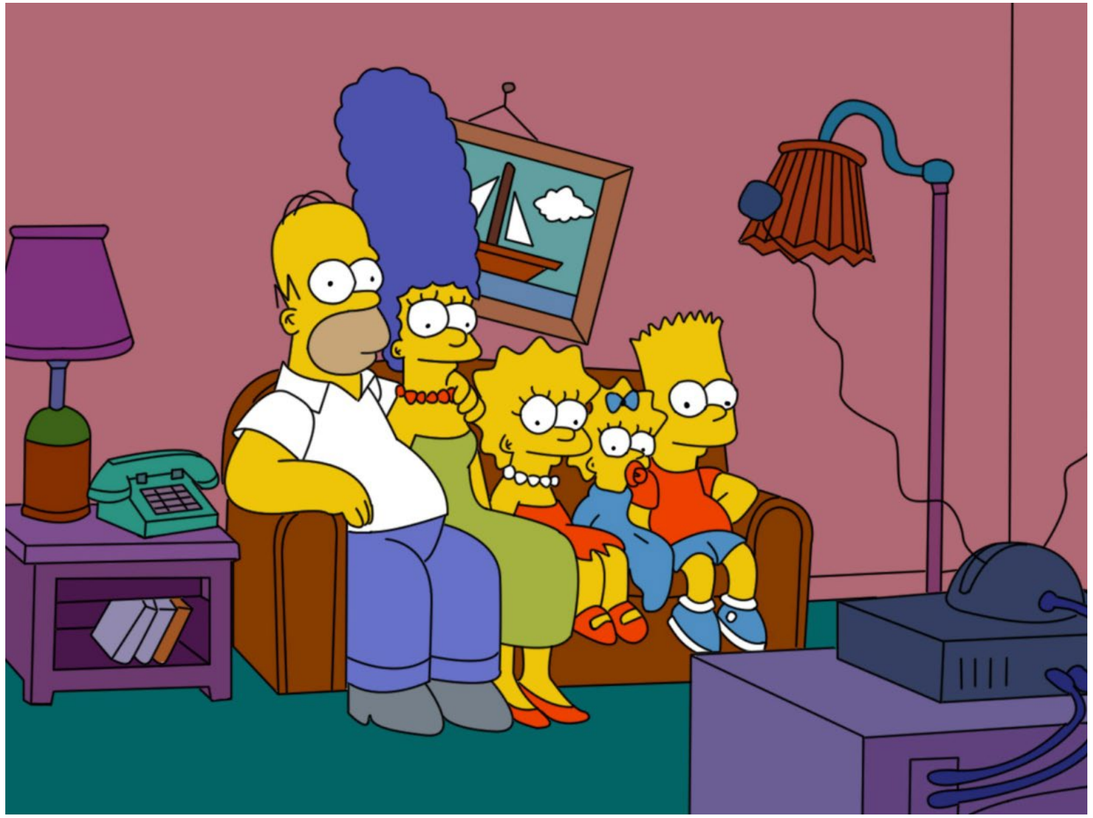
CPU Efficiency Hacks

I/O Efficiency Hacks

Space Efficiency Hacks

Handy Links

4 I/O Hack
Reduce Multiple Passes Of Data



4.1 I/O Hack – Reduce Multiple Passes of Data Technique 1-Multiple Data Steps

```
data chocolate;
     set MWSUG.cesales_analysis;
     if category='Chocolate';
Run;
data gummy;
     set MWSUG.cesales_analysis;
     if category='Gummy';
Run;
data hard;
     set MWSUG.cesales_analysis;
     if category='Hard';
Run;
data sugarfree;
     set MWSUG.cesales_analysis;
     if category='Sugar-Free';
Run;
```

4.2 I/O Hack - Reduce Multiple Passes of Data Technique 2-Multiple Select Statements

```
proc sql;
     create table  chocolate as
          select * from MWSUG.cesales_analysis
               where category='Chocolate';
     create table  gummy as
          select * from MWSUG.cesales_analysis
               where category='Gummy';
     create table  hard as
          select * from MWSUG.cesales_analysis
               where category='Hard';
     create table sugarfree as
          select * from MWSUG.cesales_analysis
               where category='Sugar-Free';
quit;
```

4.3 I/O Hack - Reduce Multiple Passes of Data Technique 3-Single DATA Step

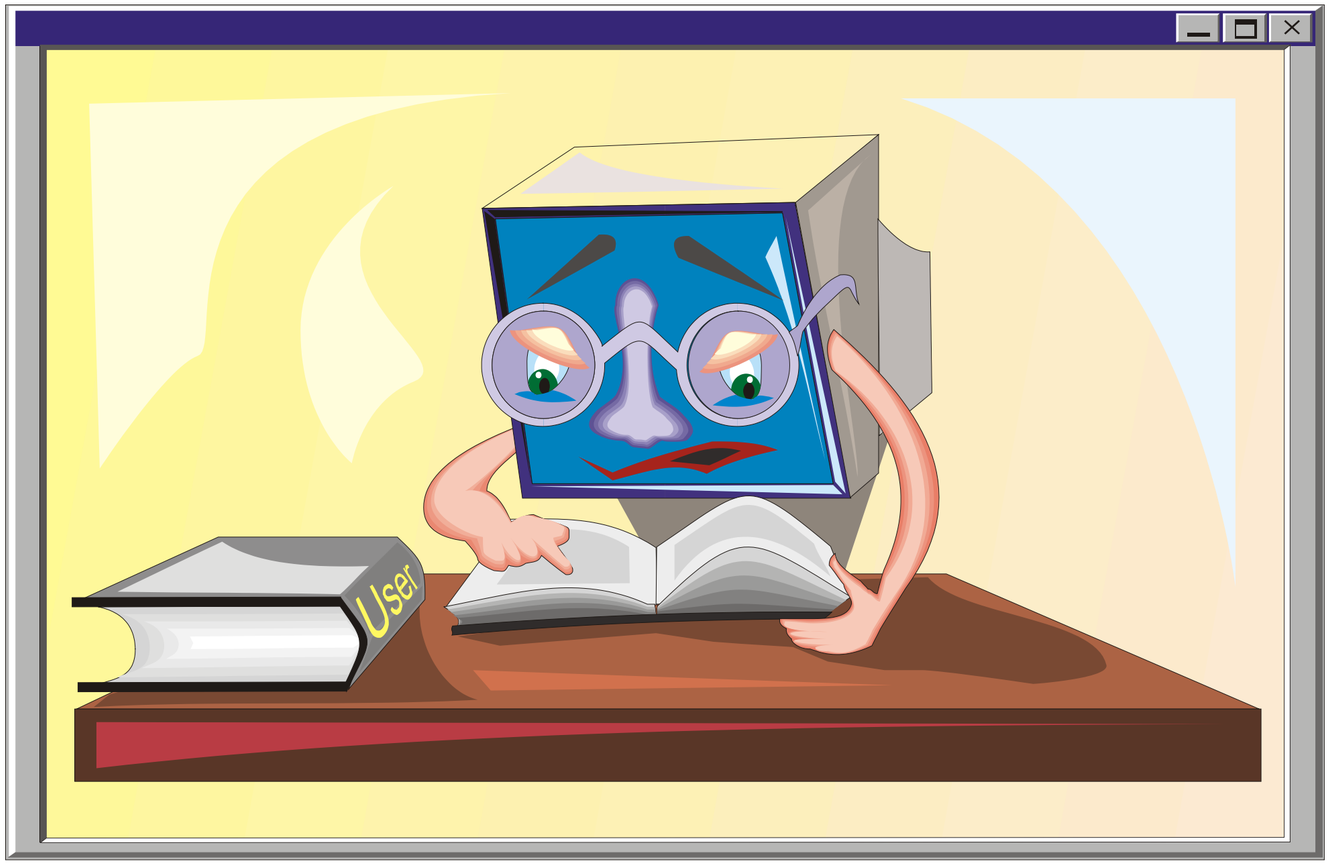
```
data chocolate gummy hard sugarfree;
     set MWSUG.cesales_analysis;
     if category='Chocolate' then
          output chocolate;
     else if category ='Gummy ' then
          output gummy;
     else if category='Hard' then
          output hard;
     else if category='Sugar-Free' then
          output sugarfree;
run;
```

Bonus Tip
PRE-WORK- Order the Conditional processing work in descending frequencies for greater efficiency

```
proc freq data=MWSUG.cesales_analysis order=freq;
     tables category;
run;
```

Techniques affecting CPU and/or IO

If you process fewer variables and observations,
CPU and/or I/O operations can be affected significantly.



5 I/O Hack
Sorting Hat - Algorithm Comparison

5.1 I/O Hack + Bonus CPU Hack

Technique 1 - Subset In Datastep, Then Sort Data

Create a sorted subset of **MWSUG.cesales_analysis** that contains the string Bag.

```
data bag;
     set MWSUG.cesales_analysis;
     where product_name contains 'Bag';
run;

proc sort data=bag;
     by product_name;
run;
```

5.2 I/O Hack + CPU Hack
Technique 2 - Sort & Filter In One Step

```
proc sort data=MWSUG.cesales_analysis out=bag;
     by product_name;
     where product_name contains 'Bag';
run;
```

Q Which one is better?

A Technique 2 as it sorts and filters in the same step thus reducing the number of times data has to be read in and written out..

Did you know ? You can use the WHERE clause in any PROC, it's a powerful way to filter your data as we saw earlier

5.3 I/O Hack + CPU Hack
Technique 3 - Proc Sort Algorithm Options

```
proc sort data=MWSUG.cesales_analysis(obs=5000) out=sortbag
     nodupkey;
     by product_name;
     where product_name contains 'Bag';
run;
proc print;
run;
```

Proc Sort Retains The First Observation Of Each By Variable

Obs	customer_id	Customer_Name	Customer_Type	Product_ID	Product_Name	Category	SubCategory	Month	Year	Total_Cases
1	100	All-Mart	BigBoxRetail	301001	Butterscotch Discs Bag	Hard	Hard Sweet	4	1999	16.00
2	100	All-Mart	BigBoxRetail	301003	Cinnamon Discs Bag	Hard	Hard Sweet	1	1999	12.00
3	100	All-Mart	BigBoxRetail	201002	Gummy Fish Bag	Gummy	Gummy Sweet	1	1999	49.00
4	100	All-Mart	BigBoxRetail	201001	Gummy Lions Bag	Gummy	Gummy Sweet	1	1999	17.00
5	100	All-Mart	BigBoxRetail	202003	Gummy Sour	Gummy	Gummy Sour	1	1999	15.00

5.4 I/O Hack + CPU Hack
Technique 4 - PROC SQL Sorting

```
proc sql number;
     select distinct * from MWSUG.cesales_analysis(obs=5000)
          where product_name contains 'Bag'
               group by product_name
                    order by product_name;
quit;
```

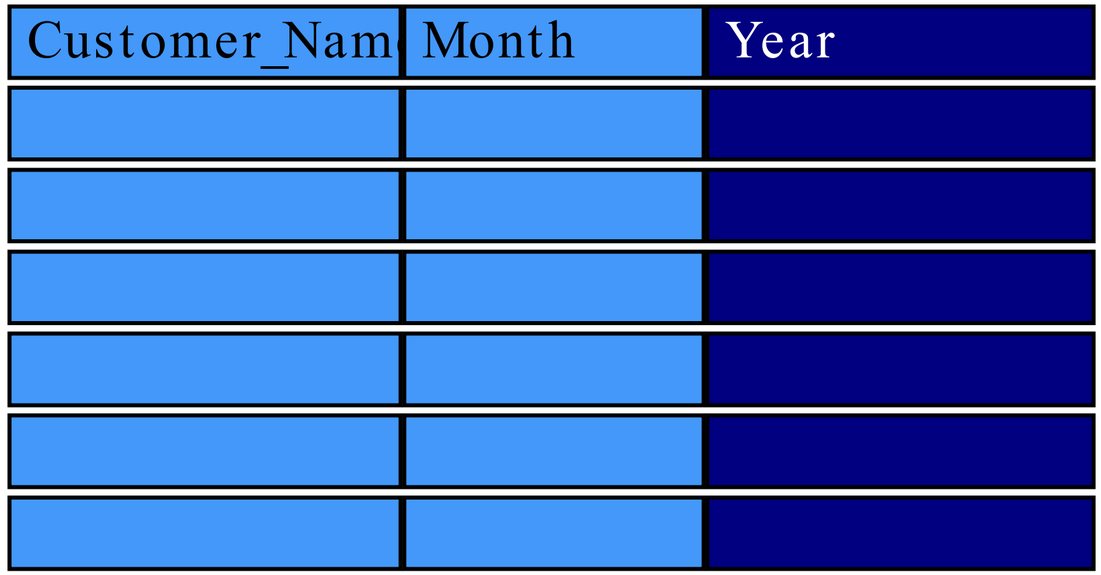
Proc SQL has no ability to retain the first Obs of each by variable

Row	customer_id	Customer_Name	Customer_Type	Product_ID	Product_Name	Category	SubCategory	Month	Year	Total_Cases
1	100	Coal's	BigBoxRetail	301001	Butterscotch Discs Bag	Hard	Hard Sweet	4	2005	16.00
2	100	Candy Unlimited	Internet	301001	Butterscotch Discs Bag	Hard	Hard Sweet	6	2002	21.00
3	100	Coal's	BigBoxRetail	301001	Butterscotch Discs Bag	Hard	Hard Sweet	9	2001	47.00
4	100	Best Candy	Gourmet	301001	Butterscotch Discs Bag	Hard	Hard Sweet	1	2002	23.00
5	100	Candy Unlimited	Internet	301001	Butterscotch	Hard	Hard Sweet	10	2000	30.00

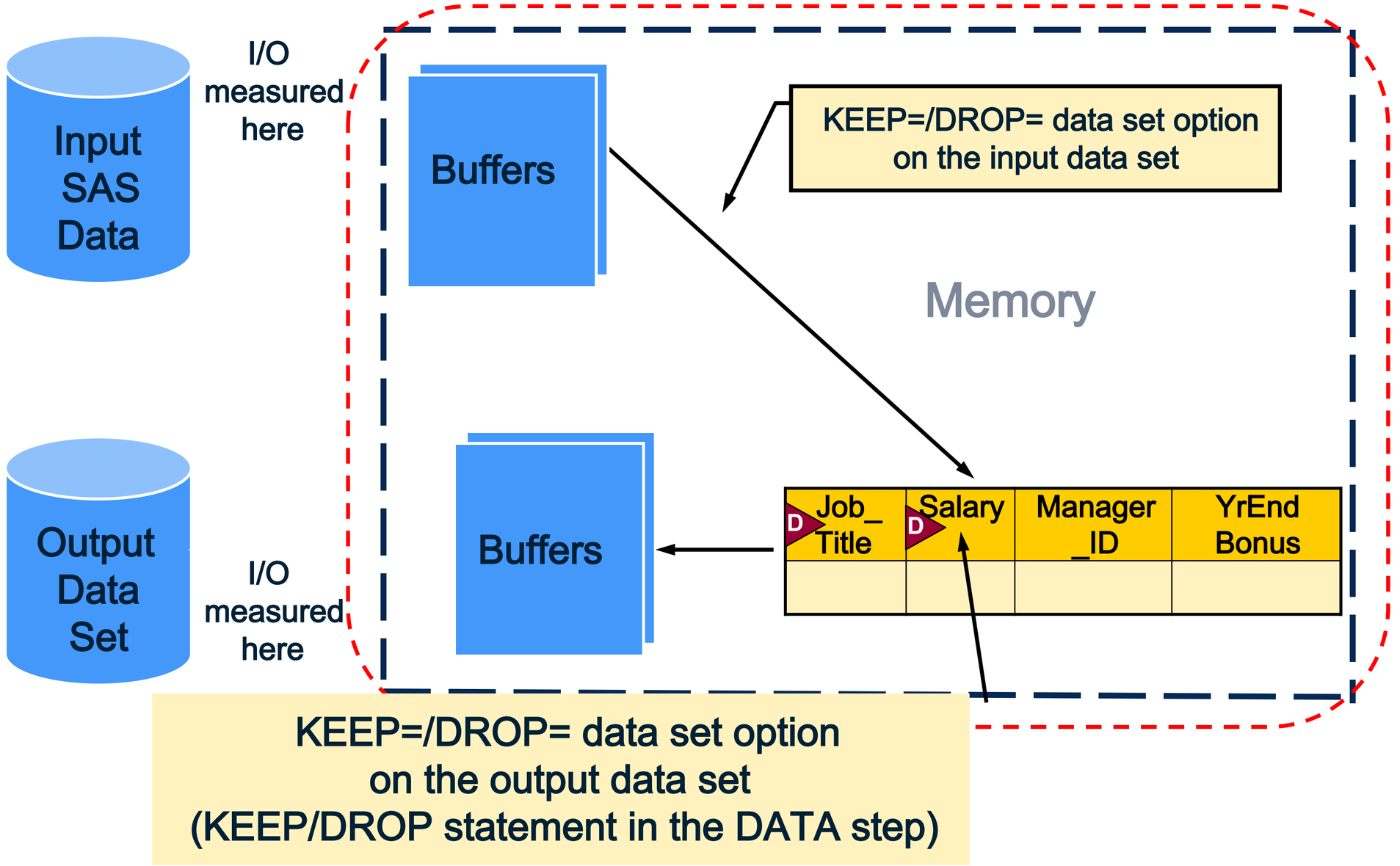
6 I/O Hack
Process Only Necessary Variables

To subset variables, you can use the following:

- DROP and KEEP statements
- DROP= and KEEP= data set options

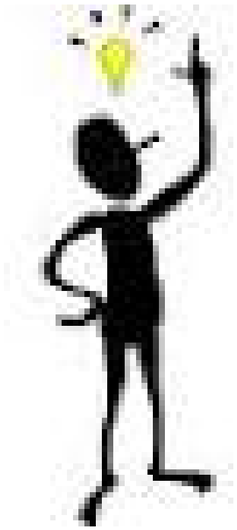


Using KEEP=/ DROP= Options



Consider- Which one is more efficient?

DROP & KEEP statements or
DROP & KEEP options?



Did you know ? DROP & KEEP options & statements are both a great way to reduce # of variables. Which one is better? Options might be better, as the moment of action is pure & clear. You don't have to go scanning code to see what variable went to building the dataset.

6.1 I/O Hack – Process only Necessary Variables Technique 1- Reading and Writing All Variables

```
data MWSUG.yearend;
     set MWSUG.dec04sales;
     extracase=total_cases*2;
run;

proc means data=MWSUG.yearend mean sum;
     where category='Chocolate';
     class customer_type;
     var extracase;
run;
```

6.2 I/O Hack - Process only Necessary Variables Technique 2 -Reading 3 Variables in the DATA step Reading 2 Variables in the PROC step

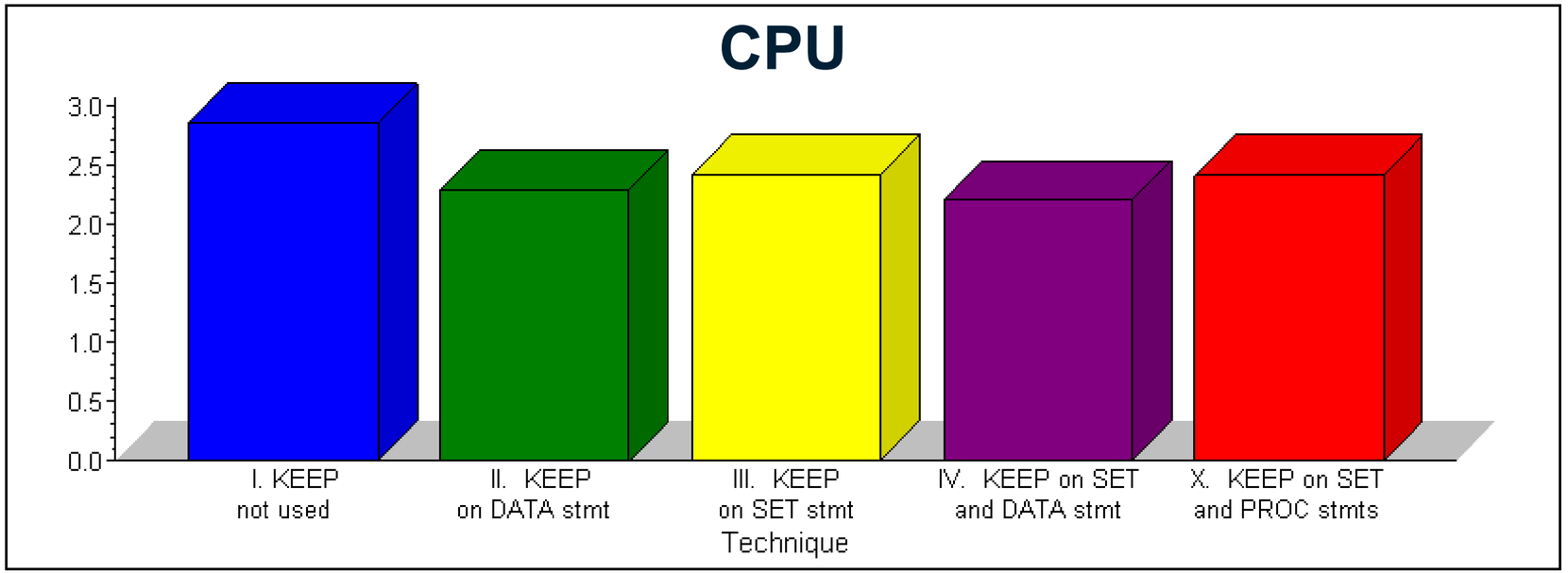
```
data MWSUG.yearend;
     set MWSUG.dec04sales(keep=category total_cases customer_type);
     extracase=total_cases*2;
     where category='Chocolate';
run;

proc means data=MWSUG.yearend(keep=extracase customer_type) mean median;
     class customer_type;
     var extracase;
run;
```

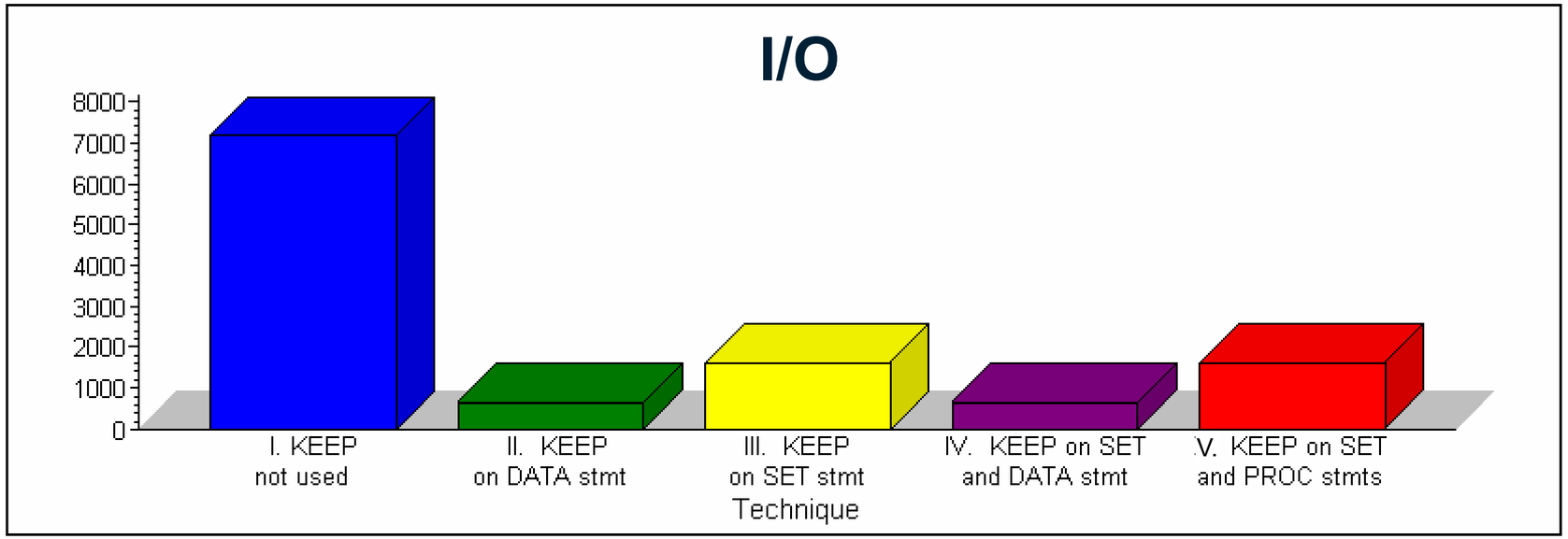
I/O savings results from reducing the number of variables and observations in the output data sets.

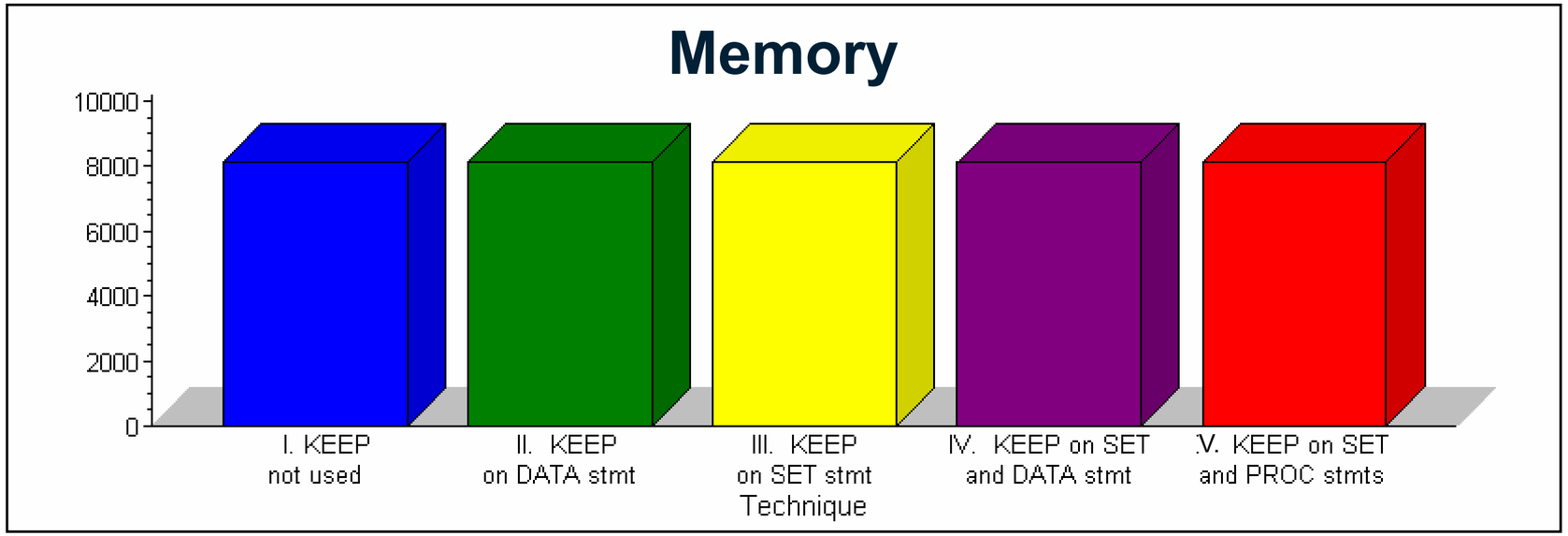
Comparing Techniques

Technique	CPU	I/O	Memory
I. KEEP not used	2.9	7177	8140
II. KEEP on DATA statement	2.3	656	8138
III. KEEP on SET statement	2.4	1625	8138
IV. KEEP on SET and DATA statements	2.2	662	8138
V. KEEP on SET and PROC statements	2.4	1625	8139

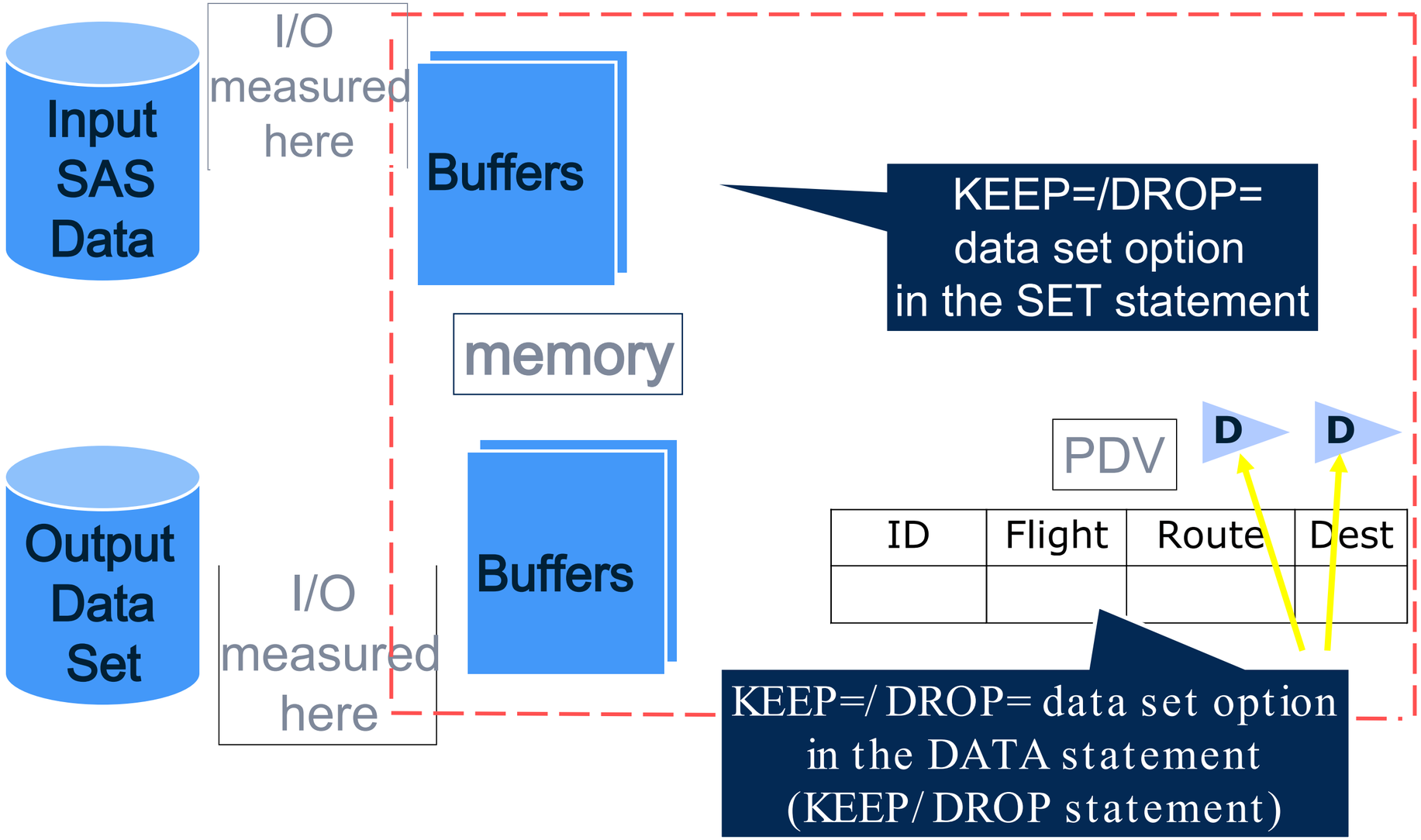


Comparing Techniques





Using the KEEP=/ DROP= Options



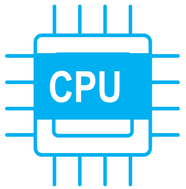
Consider- Which One Is More Efficient?

DROP & KEEP options on the DATA statement
Or DROP & KEEP options on the SET statement?

Did you know ? DROP & KEEP options on your input dataset are a great way to reduce # of variables read into the PDV substantially saving your CPU & I/O.

If your new variable's construction depends on an old variable, don't use the DROP/KEEP on the SET statement -otherwise you won't have access to it at all.

Introduction to Efficiencies



CPU Efficiency Hacks

I/O Efficiency Hacks

Space Efficiency Hacks

Handy Links

Quiz

How would you store the data type of the Product_ID column that has this data?

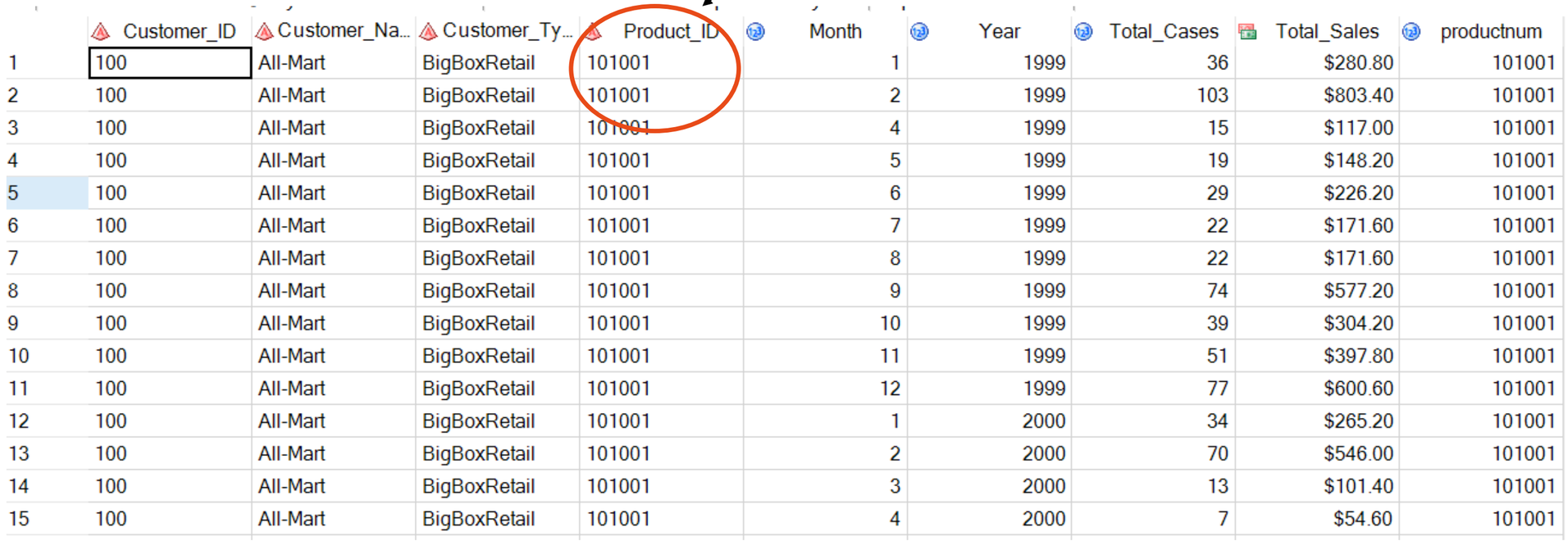
a) Character Data Type.
b) Numeric Data Type.
c) Either, It doesn't Matter.

	Customer_ID	Customer_Na...	Customer_Ty...	Product_ID	Month	Year	Total_Cases	Total_Sales	productnum
1	100	All-Mart	BigBoxRetail	101001	1	1999	36	$280.80	101001
2	100	All-Mart	BigBoxRetail	101001	2	1999	103	$803.40	101001
3	100	All-Mart	BigBoxRetail	101001	4	1999	15	$117.00	101001
4	100	All-Mart	BigBoxRetail	101001	5	1999	19	$148.20	101001
5	100	All-Mart	BigBoxRetail	101001	6	1999	29	$226.20	101001
6	100	All-Mart	BigBoxRetail	101001	7	1999	22	$171.60	101001
7	100	All-Mart	BigBoxRetail	101001	8	1999	22	$171.60	101001
8	100	All-Mart	BigBoxRetail	101001	9	1999	74	$577.20	101001
9	100	All-Mart	BigBoxRetail	101001	10	1999	39	$304.20	101001
10	100	All-Mart	BigBoxRetail	101001	11	1999	51	$397.80	101001
11	100	All-Mart	BigBoxRetail	101001	12	1999	77	$600.60	101001
12	100	All-Mart	BigBoxRetail	101001	1	2000	34	$265.20	101001
13	100	All-Mart	BigBoxRetail	101001	2	2000	70	$546.00	101001
14	100	All-Mart	BigBoxRetail	101001	3	2000	13	$101.40	101001
15	100	All-Mart	BigBoxRetail	101001	4	2000	7	$54.60	101001

Quiz – Correct Answer

How would you store the data type of the Product_ID column that has this data?

a. Character Data Type.
b. Numeric Data Type.
c. Either, It doesn't Matter.
d. It depends



	Customer_ID	Customer_Na...	Customer_Ty...	Product_ID	Month	Year	Total_Cases	Total_Sales	productnum
1	100	All-Mart	BigBoxRetail	101001	1	1999	36	$280.80	101001
2	100	All-Mart	BigBoxRetail	101001	2	1999	103	$803.40	101001
3	100	All-Mart	BigBoxRetail	101001	4	1999	15	$117.00	101001
4	100	All-Mart	BigBoxRetail	101001	5	1999	19	$148.20	101001
5	100	All-Mart	BigBoxRetail	101001	6	1999	29	$226.20	101001
6	100	All-Mart	BigBoxRetail	101001	7	1999	22	$171.60	101001
7	100	All-Mart	BigBoxRetail	101001	8	1999	22	$171.60	101001
8	100	All-Mart	BigBoxRetail	101001	9	1999	74	$577.20	101001
9	100	All-Mart	BigBoxRetail	101001	10	1999	39	$304.20	101001
10	100	All-Mart	BigBoxRetail	101001	11	1999	51	$397.80	101001
11	100	All-Mart	BigBoxRetail	101001	12	1999	77	$600.60	101001
12	100	All-Mart	BigBoxRetail	101001	1	2000	34	$265.20	101001
13	100	All-Mart	BigBoxRetail	101001	2	2000	70	$546.00	101001
14	100	All-Mart	BigBoxRetail	101001	3	2000	13	$101.40	101001
15	100	All-Mart	BigBoxRetail	101001	4	2000	7	$54.60	101001

7 Space Efficiency Hack Storage Considerations

How much space does each of these columns use? Which type is more efficient for saving space

	Character	Numeric
Saving Space	Yes	No

Product_ID	productnum
101001	101001
101001	101001
101001	101001
101001	101001

Space Efficiency Hack
Data Manipulation Considerations

The first 2 characters of Product ID indicate a Tier level. What type should Product_Id be?

	Character	Numeric
Saving Space	Yes	No
Manipulation	Yes	No

Product_ID	productnum
101001	101001
101001	101001
101001	101001
101001	101001

Space Efficiency Hack
Data Calculation Considerations

We would like to see the minimum of Product_ID values. What type should you consider for Product_ID?

Considerations	Character	Numeric
Saving space	Yes	No
Manipulation	Yes	No
Calculation	No	Yes

Product_ID	productnum
101001	101001
101001	101001
101001	101001
101001	101001

10 Space Efficiency Hack
Data Type Conversion

```
/*Saving Space - convert numeric data to
character data using the PUT function*/
data MWSUG.cechardata;
     set MWSUG.cenumdata;
     productchar=put(productnum, $6.);
run;
```

productnum	productchar
101001	101001
101001	101001
101001	101001
101001	101001
101001	101001
101001	101001

Function	What it does
INPUT(*source*, *informat*)	Converts character values to numeric values using a specified informat
PUT(*source*, *format*)	Converts numeric or character values to character values using a specified format

Handy Links

The Power of SAS SQL – SAS YouTube Video

"SAS variable lists". Support.sas.com Website.

Ask The Expert Webinar – Top 5 Handy PROC SQL Tips

SAS Tutorial | Step-by-Step PROC SQL – SAS YouTube Video

"Techniques for Optimizing Memory Usage". Support.sas.com Website.

"Know Thy Data: Techniques for Data Exploration". Pharmasug 2018, Shankar, Charu.

"Exploring DICTIONARY Tables and Views". SAS Users Group International 2005, Lafler, Kirk.

Ask the Expert Webinar - Why choose between SAS data Step & PROC SQL When You Can Have Both

"Put on the SAS®Sorting Hat and Discover Which Sort is Best for You!". PharmaSUG 2019, Hadden, Louise; Shankar, Charu.

Recommended Courses From This Presentation

- SAS®Programming 1: Essentials
- SAS®Programming 2: Data Manipulation Techniques
- SAS®SQL 1: Essentials

Thank You

Charu Shankar
SAS Institute Toronto

EMAIL Charu.shankar@sas.com
BLOG https://blogs.sas.com/content/author/charushankar/
TWITTER CharuYogaCan
LINKEDIN https://www.linkedin.com/in/charushankar/