



Data is Beautiful: How to visually communicate patterns using SAS Enterprise Guide

December 2010



Data is the new oil!



Importance of Visualizing Data

- Often given very vague concepts in order to facilitate an analysis. Asking the right questions is very important!
- It is our responsibility to be able to build a story, and make connections within the data that are not immediately apparent
- Being able to communicate ideas visually enables users:
 - Quickly understand complex concepts
 - Easily identify patterns
 - More descriptive and elegant, than describing data verbally or through text.

- With the massive amounts of data available, often meaningful connections and patterns are difficult to establish
- Proper context to the data must be provided, as it is easy to be confused with irrelevant figures
- Example:
Comparing the cash intake of branches across the country. It does not mean much on its own since there are various branch sizes, and area demographics that help define what type of business a branch conducts. In this case, further data or information must be related back to branch to put the data in **context!**

Business problem: Track very large cash activity for Client A. Visually we can trend the daily activity using graphs to spot any outlier or unusual behaviour.

What are some of the ways that this can be done?

- Bar chart
 - Line graph
 - Pie chart
- } **All univariate solutions**

Graphing bivariate or multivariate data communicates much more information at once!



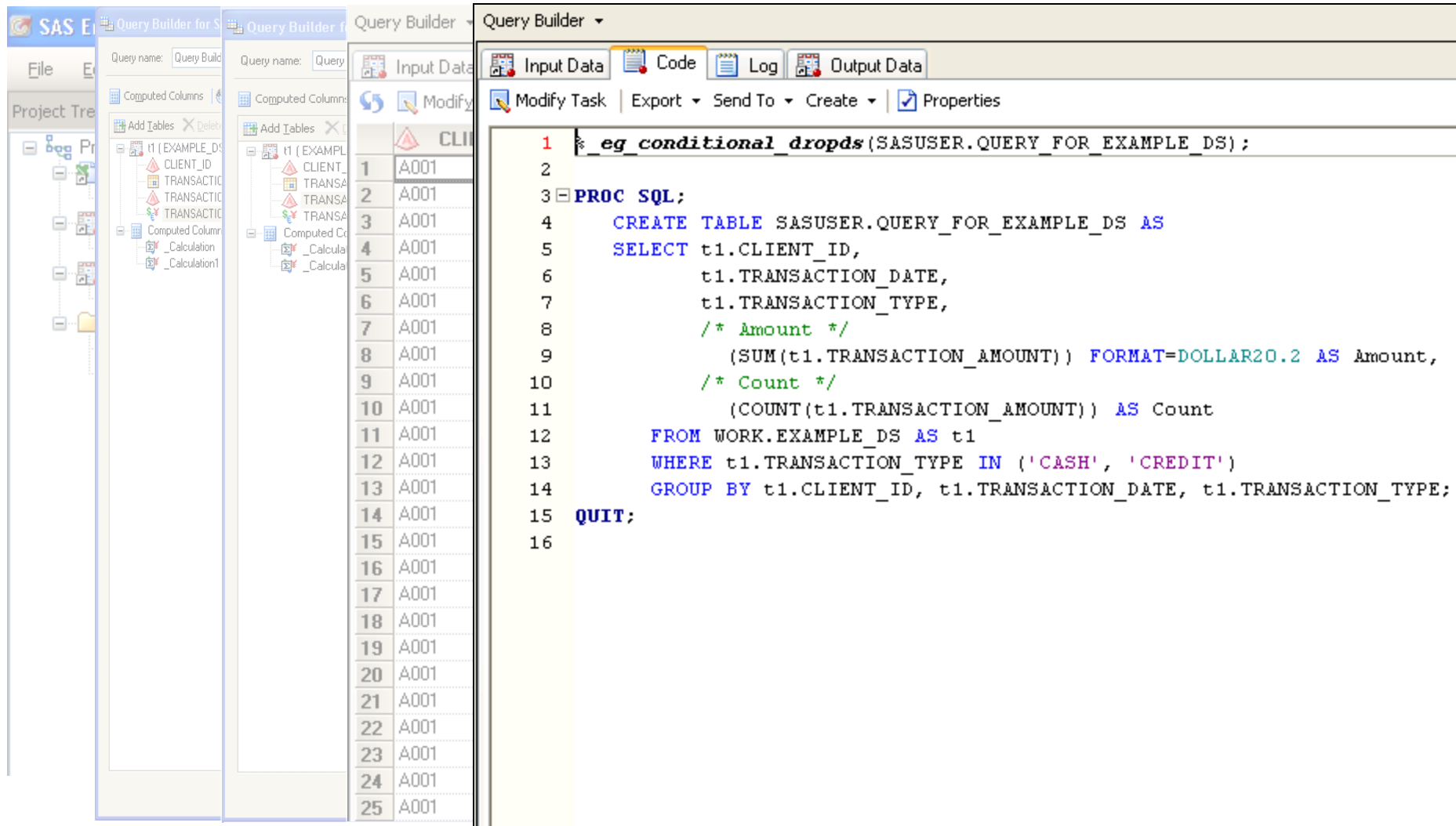
Using the Built-in EG Functionality

Data the looks like the following:

CLIENT_ID	TRANSACTION_DATE	TRANSACTION_TYPE	TRANSACTION_AMOUNT
A001	1-Aug-10	CASH	\$32.00
A001	2-Aug-10	CASH	\$48.00
A001	2-Aug-10	CHEQUE	\$32.00
A001	2-Aug-10	CASH	\$15.00
A001	2-Aug-10	CASH	\$39.00
A001	3-Aug-10	CASH	\$17.00
A001	4-Aug-10	CASH	\$29.00
A001	4-Aug-10	CASH	\$23.00
A001	5-Aug-10	CASH	\$46.00
A001	6-Aug-10	CASH	\$15.00
A001	6-Aug-10	CASH	\$14.00
A001	7-Aug-10	CASH	\$17.00
A001	8-Aug-10	CASH	\$15.00
A001	9-Aug-10	CASH	\$24.00

The first step after importing into EG is to do some data transformations, specifically summarizing the data to a daily level, and transposing the data so that there is one record per transaction date.

Rolling up data to daily level



The screenshot displays the SAS Query Builder interface. The main window shows a SQL query designed to roll up transaction data to a daily level. The query is as follows:

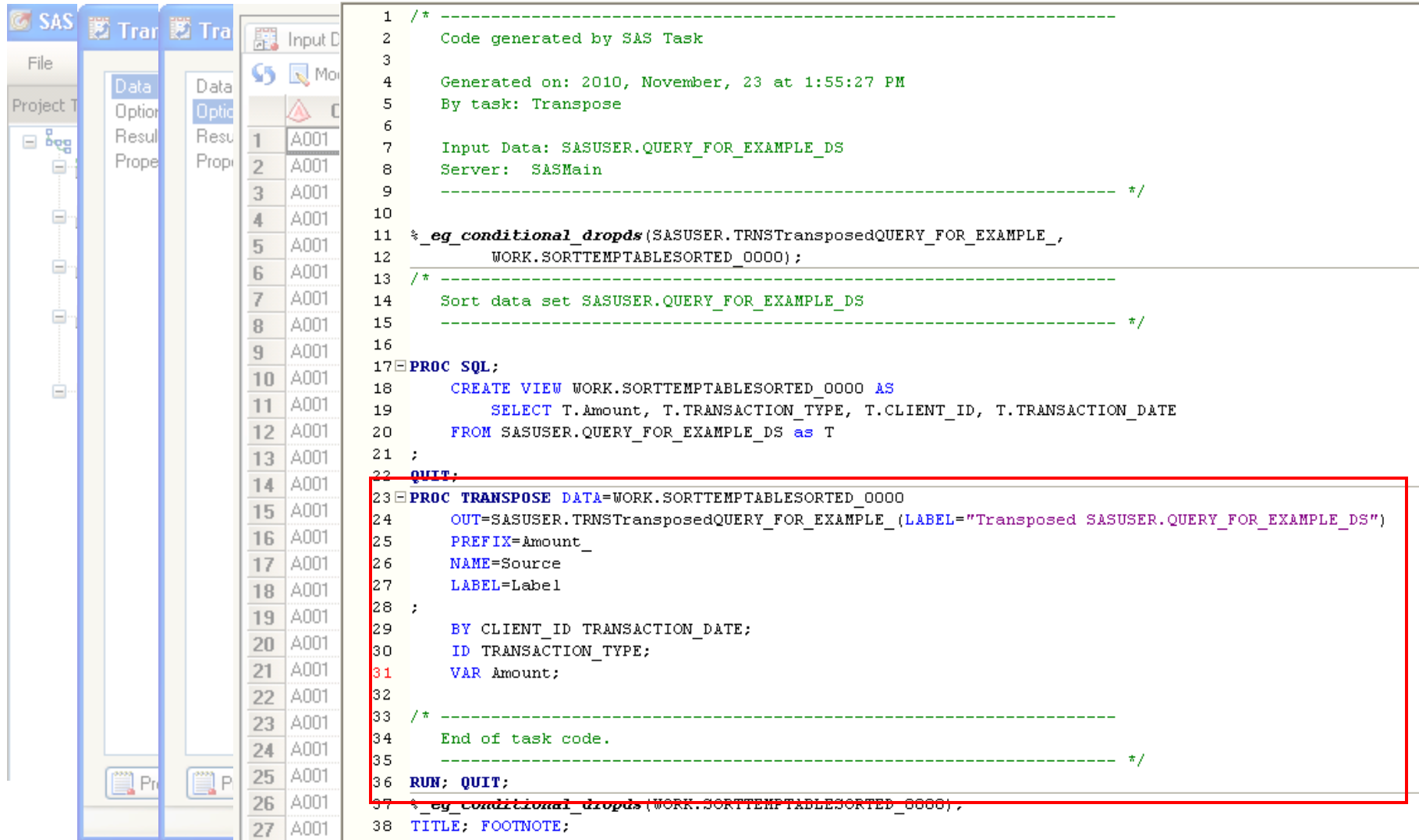
```

1  %eg_conditional_drops(SASUSER.QUERY_FOR_EXAMPLE_DS);
2
3  PROC SQL;
4      CREATE TABLE SASUSER.QUERY_FOR_EXAMPLE_DS AS
5      SELECT t1.CLIENT_ID,
6             t1.TRANSACTION_DATE,
7             t1.TRANSACTION_TYPE,
8             /* Amount */
9             (SUM(t1.TRANSACTION_AMOUNT)) FORMAT=DOLLAR20.2 AS Amount,
10            /* Count */
11            (COUNT(t1.TRANSACTION_AMOUNT)) AS Count
12      FROM WORK.EXAMPLE_DS AS t1
13     WHERE t1.TRANSACTION_TYPE IN ('CASH', 'CREDIT')
14     GROUP BY t1.CLIENT_ID, t1.TRANSACTION_DATE, t1.TRANSACTION_TYPE;
15  QUIT;
16

```

The interface also shows a table of results with 25 rows, all having a CLIENT_ID of 'A001'. The table structure is as follows:

Row	CLIENT_ID
1	A001
2	A001
3	A001
4	A001
5	A001
6	A001
7	A001
8	A001
9	A001
10	A001
11	A001
12	A001
13	A001
14	A001
15	A001
16	A001
17	A001
18	A001
19	A001
20	A001
21	A001
22	A001
23	A001
24	A001
25	A001



The screenshot displays the SAS software interface. On the left, a task window shows a list of rows numbered 1 to 27, each with the value 'A001'. The main area shows the SAS code editor with the following code:

```

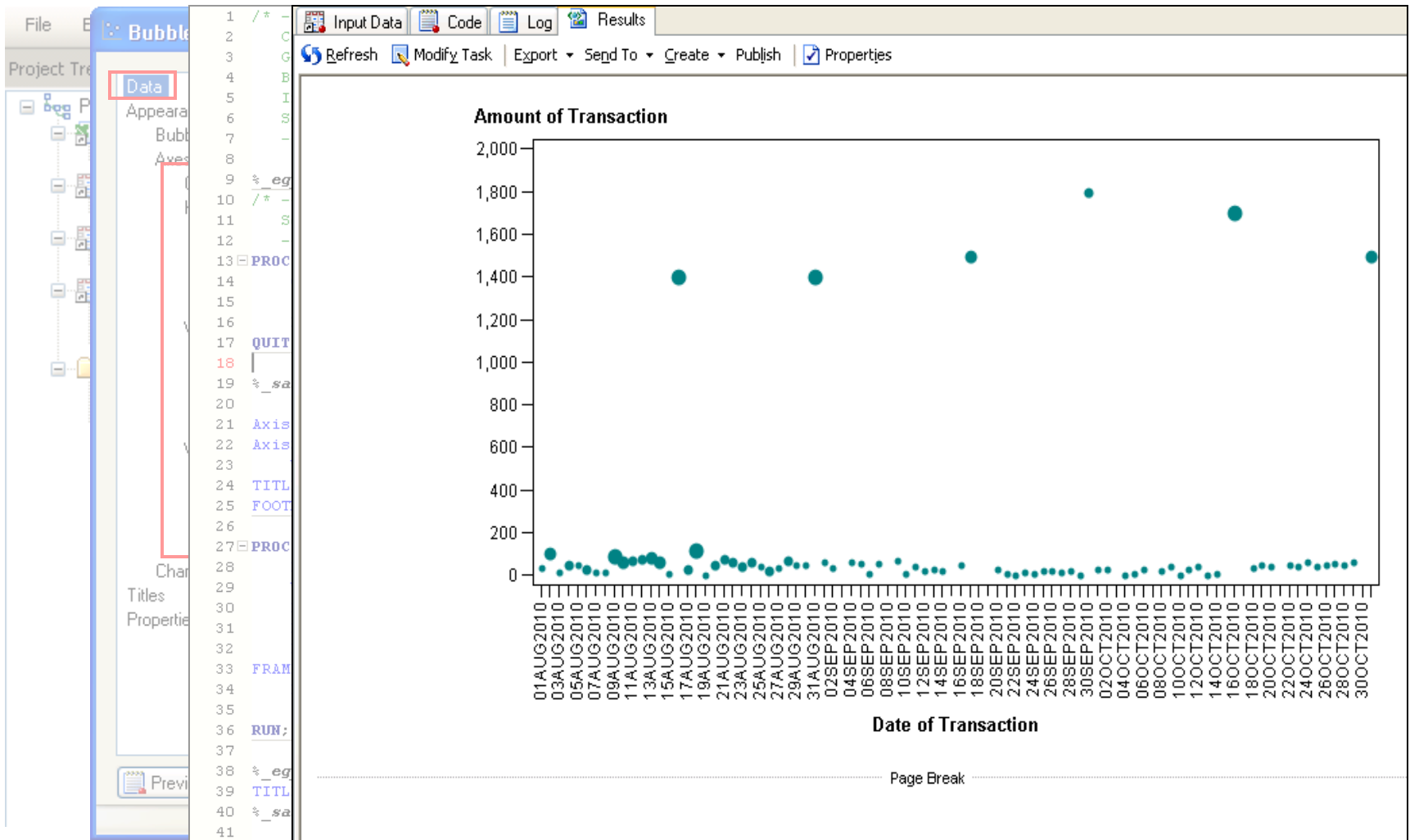
1  /* -----
2     Code generated by SAS Task
3
4     Generated on: 2010, November, 23 at 1:55:27 PM
5     By task: Transpose
6
7     Input Data: SASUSER.QUERY_FOR_EXAMPLE_DS
8     Server: SASMain
9     ----- */
10
11  %_eg_conditional_drops(SASUSER.TRN$TransposedQUERY_FOR_EXAMPLE_,
12                        WORK.SORTTEMPTABLESORTED_0000);
13  /* -----
14     Sort data set SASUSER.QUERY_FOR_EXAMPLE_DS
15     ----- */
16
17  PROC SQL;
18      CREATE VIEW WORK.SORTTEMPTABLESORTED_0000 AS
19          SELECT T.Amount, T.TRANSACTION_TYPE, T.CLIENT_ID, T.TRANSACTION_DATE
20             FROM SASUSER.QUERY_FOR_EXAMPLE_DS as T
21 ;
22  QUIT;
23  PROC TRANSPOSE DATA=WORK.SORTTEMPTABLESORTED_0000
24      OUT=SASUSER.TRN$TransposedQUERY_FOR_EXAMPLE_(LABEL="Transposed SASUSER.QUERY_FOR_EXAMPLE_DS")
25      PREFIX=Amount_
26      NAME=Source
27      LABEL=Label
28 ;
29      BY CLIENT_ID TRANSACTION_DATE;
30      ID TRANSACTION_TYPE;
31      VAR Amount;
32
33  /* -----
34     End of task code.
35     ----- */
36  RUN; QUIT;
37  %_eg_conditional_drops(WORK.SORTTEMPTABLESORTED_0000),
38  TITLE; FOOTNOTE;

```

After merging the data on Client_ID, and Transaction_Date we have a data set that looks something like this:

	CLIENT_ID	TRANSACTION_DATE	Amount_CASH	Amount_CREDIT	Count_CASH	Count_CREDIT
1	A001	01AUG2010	32	.	1	.
2	A001	02AUG2010	102	.	3	.
3	A001	03AUG2010	17	.	1	.
4	A001	04AUG2010	52	.	2	.
5	A001	05AUG2010	46	.	1	.
6	A001	06AUG2010	29	.	2	.
7	A001	07AUG2010	17	.	1	.
8	A001	08AUG2010	15	.	1	.
9	A001	09AUG2010	87	.	4	.
10	A001	10AUG2010	65	.	3	.
11	A001	11AUG2010	71	.	2	.
12	A001	12AUG2010	74	.	2	.
13	A001	13AUG2010	83	.	3	.
14	A001	14AUG2010	62	.	3	.
15	A001	15AUG2010	5	2000	1	1
16	A001	16AUG2010	1400	.	4	.

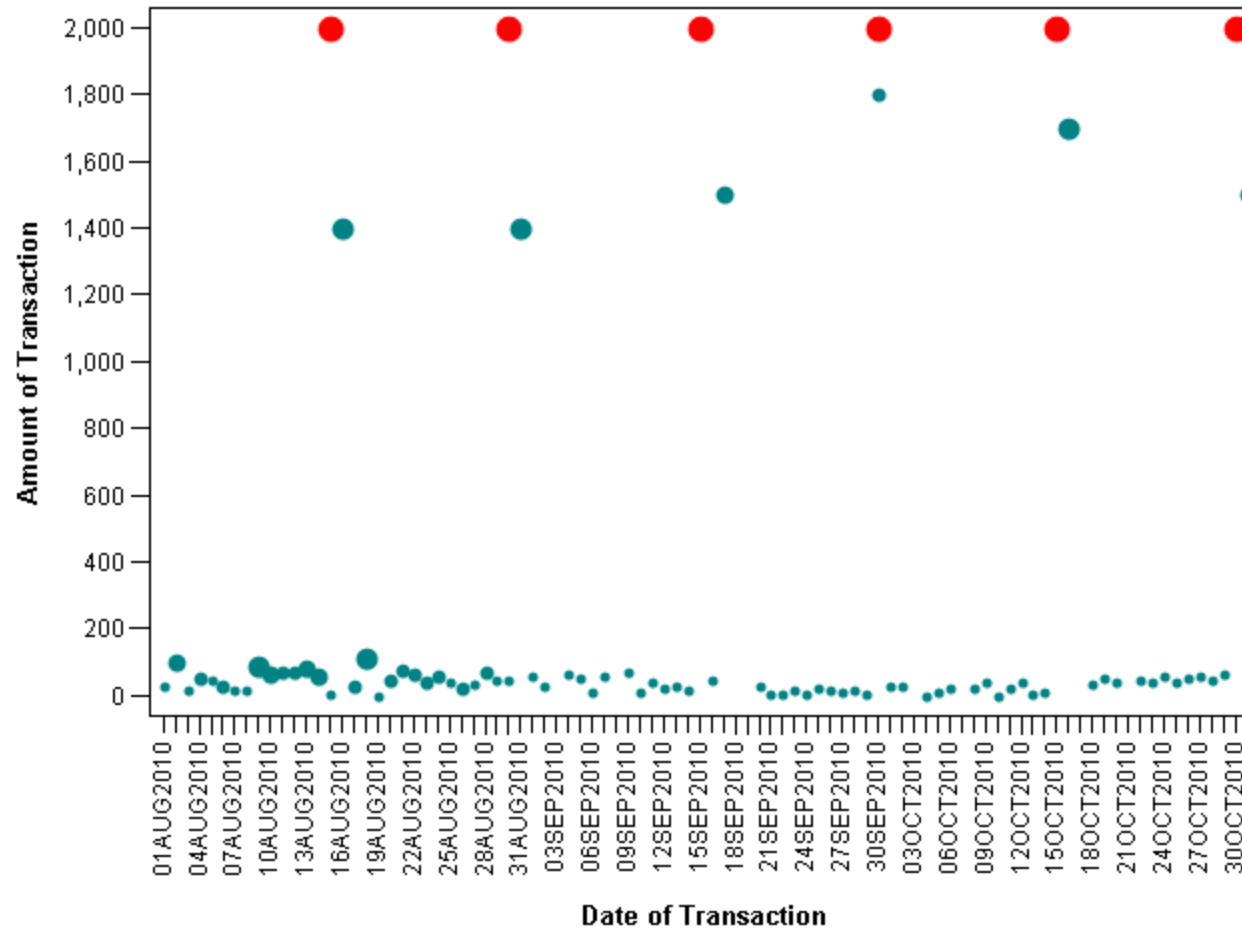
We can use one of the built in graphing tools quickly and easily to build out a graph! Much faster than writing the code to do so!



```
2 PROC SQL;
3     CREATE table WORK.SORTTempTableSorted AS
4         SELECT T.TRANSACTION_DATE, T.Amount_CASH, T.Count_CASH, T.Amount_CREDIT, T.Count_CREDIT
5     FROM WORK.ALL as T;
6 QUIT;
7
8 Axis1 STYLE=1 WIDTH=1 ORDER=(0 TO 2000 BY 200) MINOR=NONE LABEL=("Amount of Transaction" ANGLE=90);
9 Axis2 STYLE=1 WIDTH=1 MINOR=NONE ORDER="01aug10"D TO "31oct10"D BY day LABEL=("Date of Transaction")
10     VALUE=(ANGLE=90);
11 Axis3;
12
13 TITLE;
14 FOOTNOTE;|
15 PROC GPLOT DATA = WORK.SORTTempTableSorted;
16     BUBBLE Amount_CASH * TRANSACTION_DATE = Count_CASH /
17         VAXIS=AXIS1
18         HAXIS=AXIS2
19         bsize=14
20         BCOLOR=CX008080
21     FRAME;
22     FORMAT Amount_Cash COMMA6.;
23
24     BUBBLE2 Amount_CREDIT * TRANSACTION_DATE = Count_CREDIT /
25         noaxis
26         VAXIS=axis1
27         bsize = 3
28         BCOLOR=RED
29     FRAME;
30     FORMAT Amount_CREDIT COMMA6.;
31 RUN;
32 QUIT;
33
34 TITLE;
35 FOOTNOTE;
36
```



Further Customization



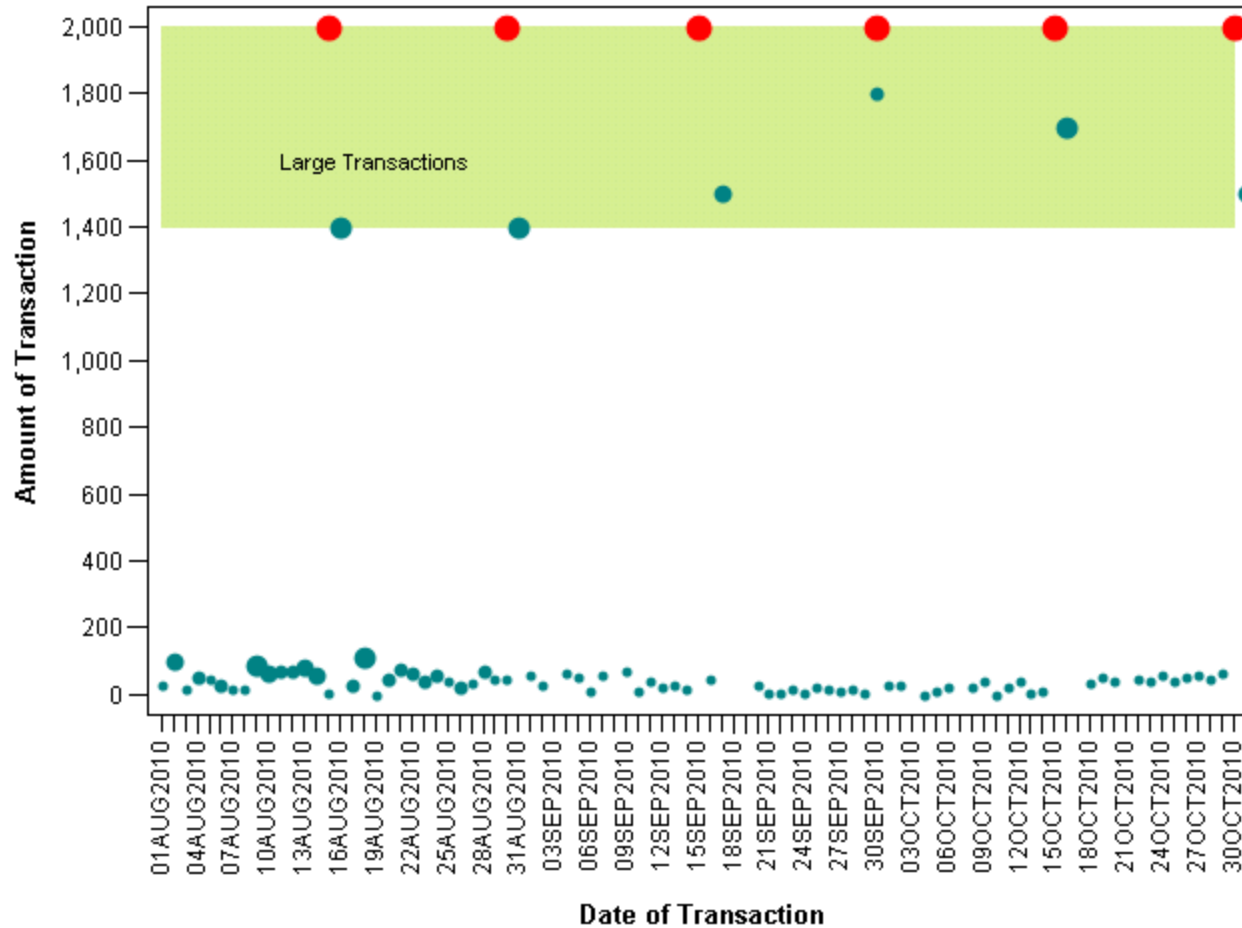


Even More Customization with Annotate!

```
8 Axis1 STYLE=1 WIDTH=1 ORDER=(0 TO 2000 BY 200) MINOR=NONE LABEL=("Amount of Transaction" ANGLE=90);
9 Axis2 STYLE=1 WIDTH=1 MINOR=NONE ORDER="01aug10"D TO "31oct10"D BY day LABEL=("Date of Transaction")
10 VALUE=(ANGLE=90);
11 Axis3;
12
13 TITLE;
14 FOOTNOTE;
15
16 data anno;
17 retain xsys ysys '2';
18 length function $ 8;
19
20 function = 'poly'; x='01AUG2010'd; y=1400; style = 'solid'; color = 'CXD4ED91'; output;
21 function = 'polycont'; x='30OCT2010'd; y=1400; style = 'solid'; color = 'CXD4ED91'; output;
22 function = 'polycont'; x='30OCT2010'd; y=2000; style = 'solid'; color = 'CXD4ED91'; output;
23 function = 'polycont'; x='01AUG2010'd; y=2000; style = 'solid'; color = 'CXD4ED91'; output;
24 function = 'label'; k = '19AUG2010'd; y = 1600; color = 'black'; text = 'Large Transactions'; output;
25 run;
26
27
28 PROC GPLOT DATA = WORK.SORTTempTableSorted anno=anno;
29 BUBBLE Amount_CASH * TRANSACTION_DATE = Count_CASH /
30 VAXIS=AXIS1
31 HAXIS=AXIS2
32 bsize=14
33 BCOLOR=CX008080
34 FRAME;
35 FORMAT Amount_Cash COMMA6.;
36
```



Even More Customization with Annotate!





- SAS Enterprise Guide let's you **quickly and easily** build out complex graphics!
- Not limited to what the menus provide, you can go into the **code and customize!**
- Don't blindly create graphs upon graphs, bring in more data and create the proper story, and assign **context** to your analysis!



Questions?

Contact: meera.das@rbc.com