PROC SUMMARY

(Almost) Everything you need to know about PROC SUMMARY

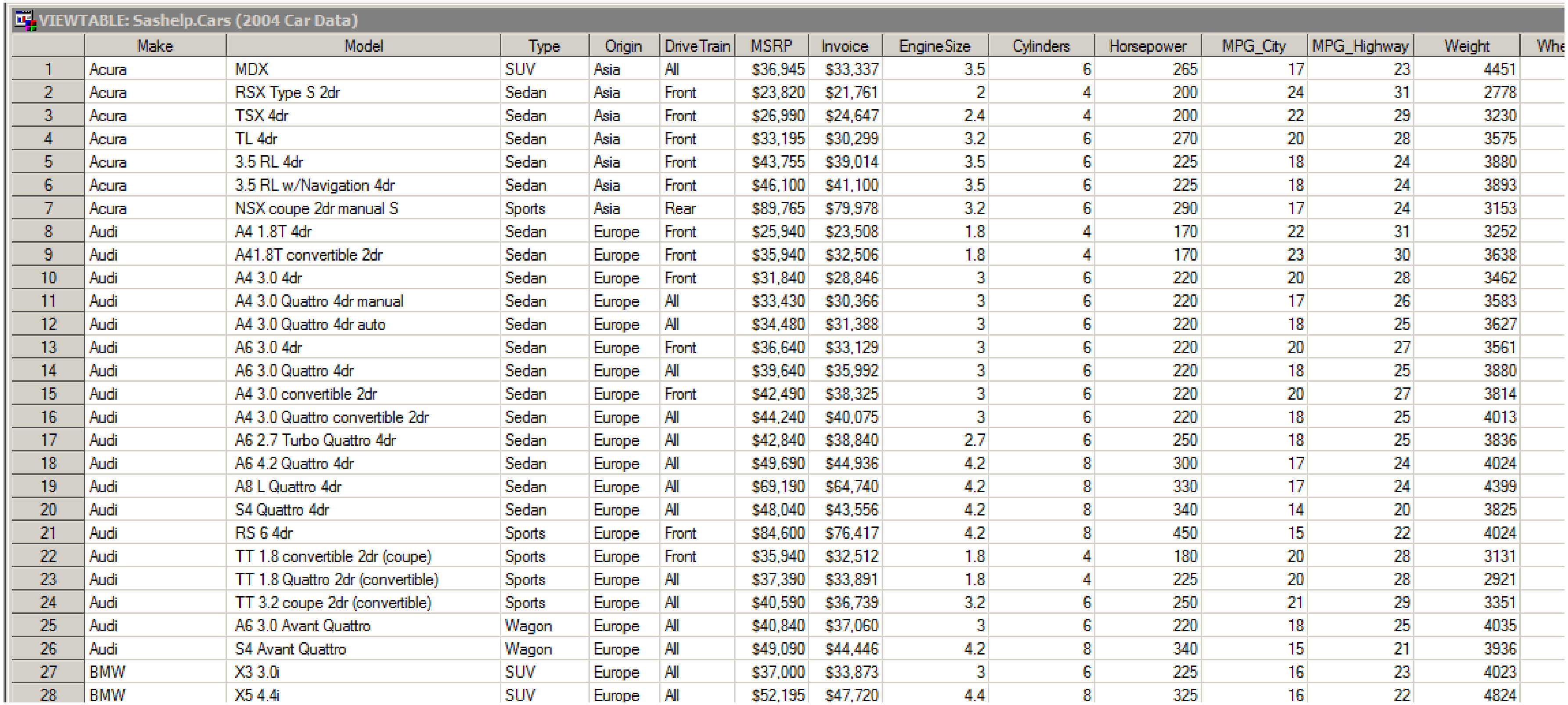
M&T Bank

PROC SUMMARY Overview

- Useful for summarizing data overall and/or by categories
- Approximately 99% overlap with PROC MEANS
  - Default output from PROC MEANS is a printed table
  - Default output from PROC SUMMARY is a SAS Data Set
  - These defaults can be over-ridden
- Can be faster than doing similar calculations in PROC SQL
- Computes many useful statistics including:
  - Mean
  - Sum
  - Standard Deviation
  - N
  - N Missing
  - Median
  - Quartiles and Percentiles
  - Minimum
  - Maximum
  - Range
  - Median (or other percentiles)
  - Many other statistics (see SAS Help)

PROC SUMMARY Overview

- Data to be used in most examples is SASHELP.CARS (partial view), which you all have access to (no LIBNAME statement needed)



VIEWTABLE: Sashelp.Cars (2004 Car Data)

	Make	Model	Type	Origin	DriveTrain	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Whe
1	Acura	MDX	SUV	Asia	All	$36,945	$33,337	3.5	6	265	17	23	4451	
2	Acura	RSX Type S 2dr	Sedan	Asia	Front	$23,820	$21,761	2	4	200	24	31	2778	
3	Acura	TSX 4dr	Sedan	Asia	Front	$26,990	$24,647	2.4	4	200	22	29	3230	
4	Acura	TL 4dr	Sedan	Asia	Front	$33,195	$30,299	3.2	6	270	20	28	3575	
5	Acura	3.5 RL 4dr	Sedan	Asia	Front	$43,755	$39,014	3.5	6	225	18	24	3880	
6	Acura	3.5 RL w/Navigation 4dr	Sedan	Asia	Front	$46,100	$41,100	3.5	6	225	18	24	3893	
7	Acura	NSX coupe 2dr manual S	Sports	Asia	Rear	$89,765	$79,978	3.2	6	290	17	24	3153	
8	Audi	A4 1.8T 4dr	Sedan	Europe	Front	$25,940	$23,508	1.8	4	170	22	31	3252	
9	Audi	A41.8T convertible 2dr	Sedan	Europe	Front	$35,940	$32,506	1.8	4	170	23	30	3638	
10	Audi	A4 3.0 4dr	Sedan	Europe	Front	$31,840	$28,846	3	6	220	20	28	3462	
11	Audi	A4 3.0 Quattro 4dr manual	Sedan	Europe	All	$33,430	$30,366	3	6	220	17	26	3583	
12	Audi	A4 3.0 Quattro 4dr auto	Sedan	Europe	All	$34,480	$31,388	3	6	220	18	25	3627	
13	Audi	A6 3.0 4dr	Sedan	Europe	Front	$36,640	$33,129	3	6	220	20	27	3561	
14	Audi	A6 3.0 Quattro 4dr	Sedan	Europe	All	$39,640	$35,992	3	6	220	18	25	3880	
15	Audi	A4 3.0 convertible 2dr	Sedan	Europe	Front	$42,490	$38,325	3	6	220	20	27	3814	
16	Audi	A4 3.0 Quattro convertible 2dr	Sedan	Europe	All	$44,240	$40,075	3	6	220	18	25	4013	
17	Audi	A6 2.7 Turbo Quattro 4dr	Sedan	Europe	All	$42,840	$38,840	2.7	6	250	18	25	3836	
18	Audi	A6 4.2 Quattro 4dr	Sedan	Europe	All	$49,690	$44,936	4.2	8	300	17	24	4024	
19	Audi	A8 L Quattro 4dr	Sedan	Europe	All	$69,190	$64,740	4.2	8	330	17	24	4399	
20	Audi	S4 Quattro 4dr	Sedan	Europe	All	$48,040	$43,556	4.2	8	340	14	20	3825	
21	Audi	RS 6 4dr	Sports	Europe	Front	$84,600	$76,417	4.2	8	450	15	22	4024	
22	Audi	TT 1.8 convertible 2dr (coupe)	Sports	Europe	Front	$35,940	$32,512	1.8	4	180	20	28	3131	
23	Audi	TT 1.8 Quattro 2dr (convertible)	Sports	Europe	All	$37,390	$33,891	1.8	4	225	20	28	2921	
24	Audi	TT 3.2 coupe 2dr (convertible)	Sports	Europe	All	$40,590	$36,739	3.2	6	250	21	29	3351	
25	Audi	A6 3.0 Avant Quattro	Wagon	Europe	All	$40,840	$37,060	3	6	220	18	25	4035	
26	Audi	S4 Avant Quattro	Wagon	Europe	All	$49,090	$44,446	4.2	8	340	15	21	3936	
27	BMW	X3 3.0i	SUV	Europe	All	$37,000	$33,873	3	6	225	16	23	4023	
28	BMW	X5 4.4i	SUV	Europe	All	$52,195	$47,720	4.4	8	325	16	22	4824	

PROC SUMMARY Overview

Example 1: Compute average MSRP by Origin

```
proc summary data=sashelp.cars;
    class origin;
    var msrp;
    output out=summary_out mean=;
run;
```

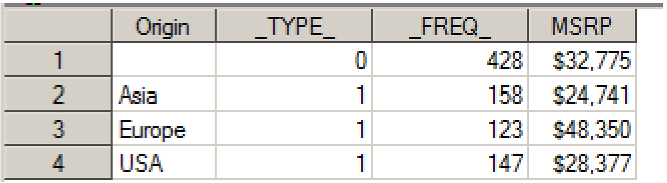
Things to note:

- The CLASS statement accepts the name of the variable(s) used for categories; can be either a numeric or character variable; CLASS is optional but usually present; data does not have to be pre-sorted by CLASS variables. PROC SUMMARY figures out how many distinct values of each CLASS variable are present (for ORIGIN, there are 3).
- The VAR statement specifies the name of the variable(s) to compute statistics from; must be a numeric variable; required
- The OUTPUT statement requires OUT= followed by the name of the output data set; and at least one statistic. If there is no text after MEAN= then the mean gets the same variable name as the VAR variable(s). The OUTPUT statement is required. The OUT= is optional but recommended; the statistic is also optional but recommended.

PROC SUMMARY Overview

Example 1: Compute average MSRP by Origin

Result of executing this code:



	Origin	_TYPE_	_FREQ_	MSRP
1		0	428	$32,775
2	Asia	1	158	$24,741
3	Europe	1	123	$48,350
4	USA	1	147	$28,377

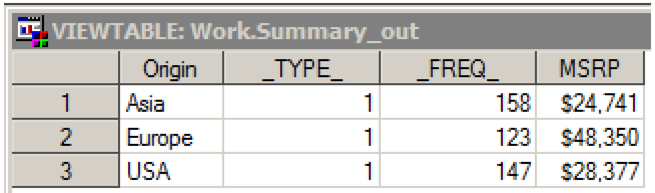
Things to note:

- Each value of the CLASS variable gets a row, showing the value of the CLASS variable, _TYPE_ (to be explained later), _FREQ_ (number of records found) and the mean value of the VAR variable.
- You also get a row where the CLASS variable is missing and _TYPE_=0, this is the average over all data (regardless of CLASS).

PROC SUMMARY Overview

Example 1 Modified: Compute average MSRP by Origin, but suppose you don’t want the first row with the overall average

```
proc summary data=sashelp.cars nway;
    class origin;
    var msrp;
    output out=summary_out mean=;
run;
```



VIEWTABLE: Work.Summary_out

	Origin	_TYPE_	_FREQ_	MSRP
1	Asia	1	158	$24,741
2	Europe	1	123	$48,350
3	USA	1	147	$28,377

Notes: the NWAY option eliminates the overall average row. It also has other effects that do not show up in this simple example, these are described later.

PROC SUMMARY Overview

Example 2: Compute average, minimum and maximum of MSRP and Horsepower by Origin

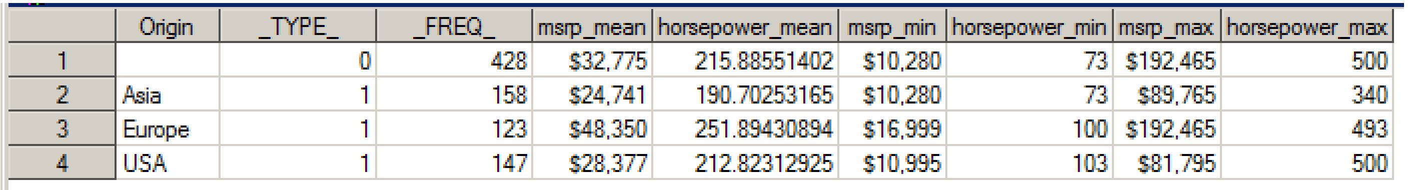
```
proc summary data=sashelp.cars;
    class origin;
    var msrp horsepower;
    output out=summary_out mean=msrp_mean horsepower_mean
        min=msrp_min horsepower_min max=msrp_max horsepower_max;
run;
```

Things to note:

- The OUTPUT statement is required; OUT= is used to specify the name of the output data set; and at least one statement is requested. If you have *n* VAR variables, you must specify *n* variable names after each statistic (use any legal SAS variable name).
    - Except that you can leave the variable names after one statistic blank and the VAR variable names would be used in the output data set for that statistic.
    - Or use `output out=summary_out mean= min= max= / autoname;`

PROC SUMMARY Overview

Example 2: Compute average, minimum and maximum of MSRP and Horsepower by Origin



	Origin	_TYPE_	_FREQ_	msrp_mean	horsepower_mean	msrp_min	horsepower_min	msrp_max	horsepower_max
1		0	428	$32,775	215.88551402	$10,280	73	$192,465	500
2	Asia	1	158	$24,741	190.70253165	$10,280	73	$89,765	340
3	Europe	1	123	$48,350	251.89430894	$16,999	100	$192,465	493
4	USA	1	147	$28,377	212.82312925	$10,995	103	$81,795	500

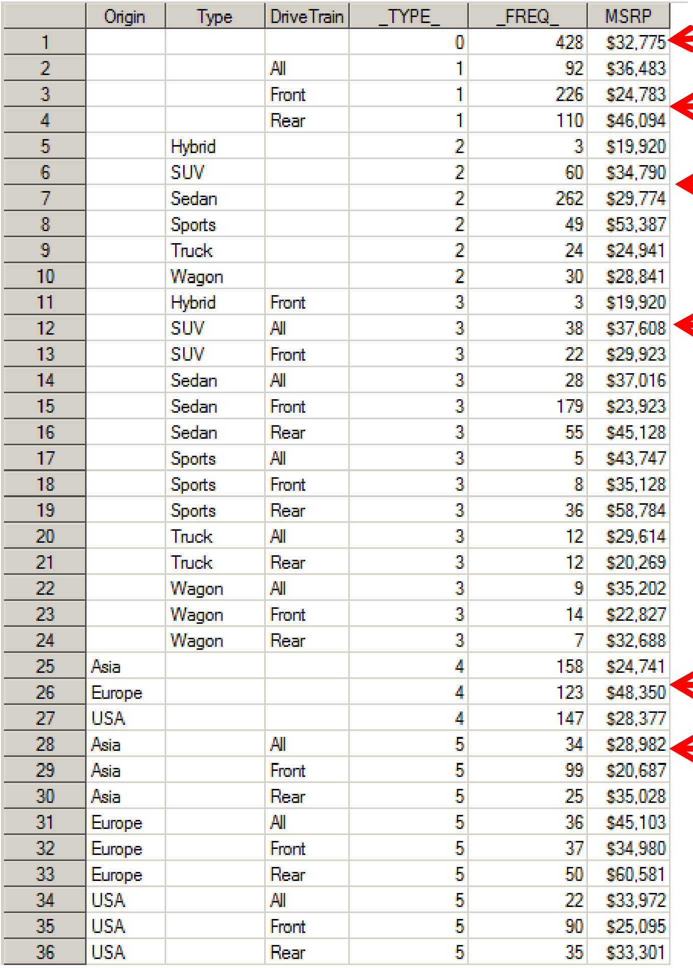
PROC SUMMARY Overview

Example 3: Compute average using multiple variables in the CLASS Statement

```
proc summary data=sashelp.cars;
    class origin type drivetrain;
    var msrp;
    output out=summary_out mean=;
run;
```

PROC SUMMARY Overview

Example 3: Compute average using multiple variables in the CLASS Statement



	Origin	Type	DriveTrain	_TYPE_	_FREQ_	MSRP
1				0	428	$32,775
2			All	1	92	$36,483
3			Front	1	226	$24,783
4			Rear	1	110	$46,094
5		Hybrid		2	3	$19,920
6		SUV		2	60	$34,790
7		Sedan		2	262	$29,774
8		Sports		2	49	$53,387
9		Truck		2	24	$24,941
10		Wagon		2	30	$28,841
11		Hybrid	Front	3	3	$19,920
12		SUV	All	3	38	$37,608
13		SUV	Front	3	22	$29,923
14		Sedan	All	3	28	$37,016
15		Sedan	Front	3	179	$23,923
16		Sedan	Rear	3	55	$45,128
17		Sports	All	3	5	$43,747
18		Sports	Front	3	8	$35,128
19		Sports	Rear	3	36	$58,784
20		Truck	All	3	12	$29,614
21		Truck	Rear	3	12	$20,269
22		Wagon	All	3	9	$35,202
23		Wagon	Front	3	14	$22,827
24		Wagon	Rear	3	7	$32,688
25	Asia			4	158	$24,741
26	Europe			4	123	$48,350
27	USA			4	147	$28,377
28	Asia		All	5	34	$28,982
29	Asia		Front	5	99	$20,687
30	Asia		Rear	5	25	$35,028
31	Europe		All	5	36	$45,103
32	Europe		Front	5	37	$34,980
33	Europe		Rear	5	50	$60,581
34	USA		All	5	22	$33,972
35	USA		Front	5	90	$25,095
36	USA		Rear	5	35	$33,301

Overall average (_TYPE_=0)

Average by Drive Train (_TYPE_=1)

Average by Type (_TYPE_=2)

Average by combination of Type and Drivetrain (_TYPE_=3)

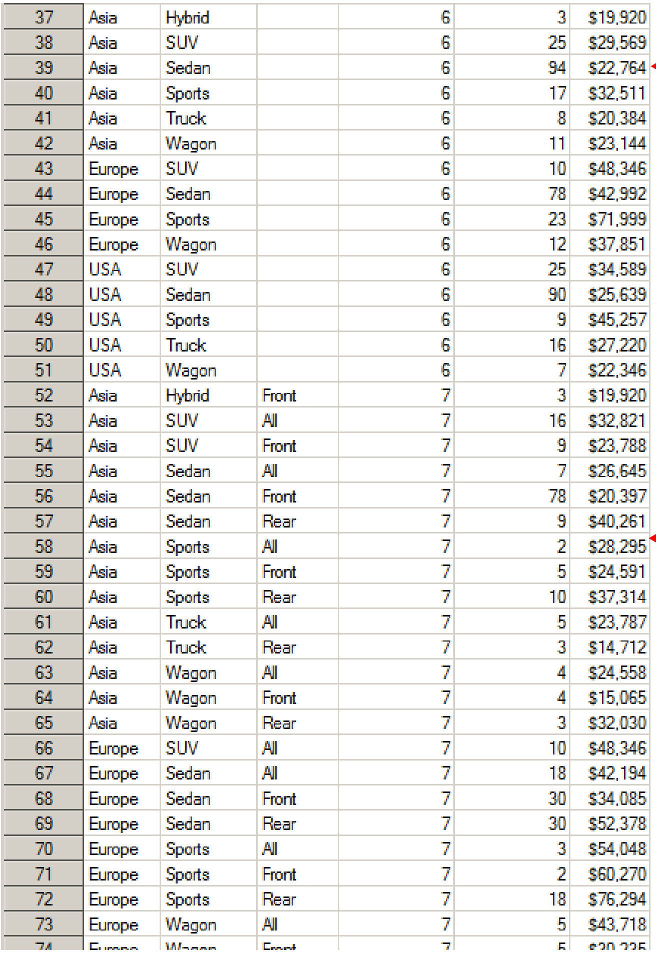
Average by Origin (_TYPE_=4)

Average by combination of Origin and Drivetrain (_TYPE_=5)

Continued on next slide

PROC SUMMARY Overview

Example 3: Compute average using multiple variables in the CLASS Statement



37	Asia	Hybrid		6	3	$19,920
38	Asia	SUV		6	25	$29,569
39	Asia	Sedan		6	94	$22,764
40	Asia	Sports		6	17	$32,511
41	Asia	Truck		6	8	$20,384
42	Asia	Wagon		6	11	$23,144
43	Europe	SUV		6	10	$48,346
44	Europe	Sedan		6	78	$42,992
45	Europe	Sports		6	23	$71,999
46	Europe	Wagon		6	12	$37,851
47	USA	SUV		6	25	$34,589
48	USA	Sedan		6	90	$25,639
49	USA	Sports		6	9	$45,257
50	USA	Truck		6	16	$27,220
51	USA	Wagon		6	7	$22,346
52	Asia	Hybrid	Front	7	3	$19,920
53	Asia	SUV	All	7	16	$32,821
54	Asia	SUV	Front	7	9	$23,788
55	Asia	Sedan	All	7	7	$26,645
56	Asia	Sedan	Front	7	78	$20,397
57	Asia	Sedan	Rear	7	9	$40,261
58	Asia	Sports	All	7	2	$28,295
59	Asia	Sports	Front	7	5	$24,591
60	Asia	Sports	Rear	7	10	$37,314
61	Asia	Truck	All	7	5	$23,787
62	Asia	Truck	Rear	7	3	$14,712
63	Asia	Wagon	All	7	4	$24,558
64	Asia	Wagon	Front	7	4	$15,065
65	Asia	Wagon	Rear	7	3	$32,030
66	Europe	SUV	All	7	10	$48,346
67	Europe	Sedan	All	7	18	$42,194
68	Europe	Sedan	Front	7	30	$34,085
69	Europe	Sedan	Rear	7	30	$52,378
70	Europe	Sports	All	7	3	$54,048
71	Europe	Sports	Front	7	2	$60,270
72	Europe	Sports	Rear	7	18	$76,294
73	Europe	Wagon	All	7	5	$43,718
74	Europe	Wagon	Front	7	5	$30,225

Average of the combination of Origin and Type (_TYPE_=6)

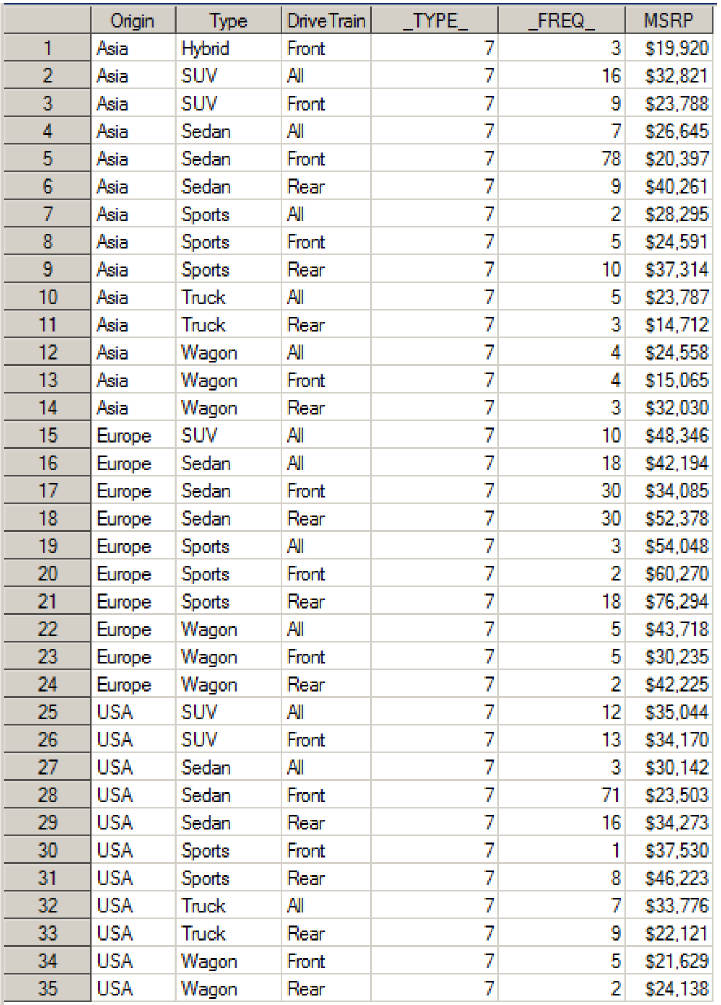
Average of the combination of all three CLASS variables (_TYPE_=7)

PROC SUMMARY Overview

Example 3 MODIFICATION 1: Compute average using multiple variables in the CLASS Statement, but you only want combinations of all three CLASS variables

```
proc summary data=sashelp.cars nway;
    class origin type drivetrain;
    var msrp;
    output out=summary_out mean=;
run;
```

Notes: You only get the _TYPE_=7 results (*i.e.* the combination of all three CLASS variables). NWAY gives you the combination of all CLASS variables and nothing else.

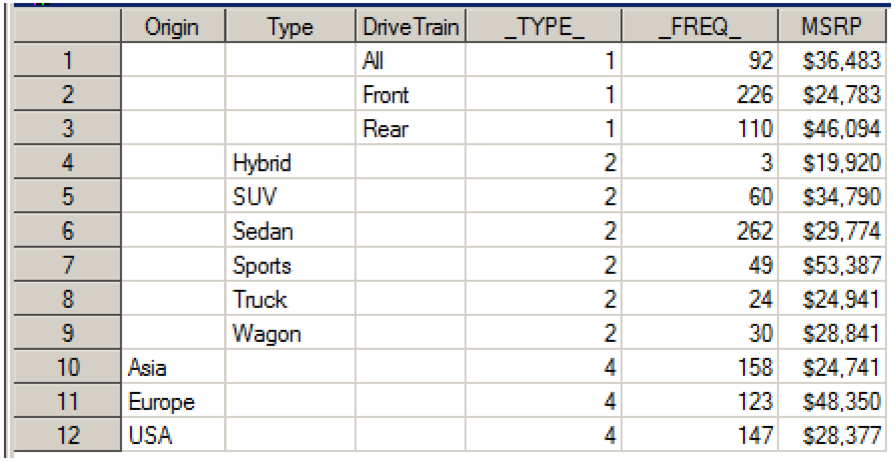


	Origin	Type	DriveTrain	_TYPE_	_FREQ_	MSRP
1	Asia	Hybrid	Front	7	3	$19,920
2	Asia	SUV	All	7	16	$32,821
3	Asia	SUV	Front	7	9	$23,788
4	Asia	Sedan	All	7	7	$26,645
5	Asia	Sedan	Front	7	78	$20,397
6	Asia	Sedan	Rear	7	9	$40,261
7	Asia	Sports	All	7	2	$28,295
8	Asia	Sports	Front	7	5	$24,591
9	Asia	Sports	Rear	7	10	$37,314
10	Asia	Truck	All	7	5	$23,787
11	Asia	Truck	Rear	7	3	$14,712
12	Asia	Wagon	All	7	4	$24,558
13	Asia	Wagon	Front	7	4	$15,065
14	Asia	Wagon	Rear	7	3	$32,030
15	Europe	SUV	All	7	10	$48,346
16	Europe	Sedan	All	7	18	$42,194
17	Europe	Sedan	Front	7	30	$34,085
18	Europe	Sedan	Rear	7	30	$52,378
19	Europe	Sports	All	7	3	$54,048
20	Europe	Sports	Front	7	2	$60,270
21	Europe	Sports	Rear	7	18	$76,294
22	Europe	Wagon	All	7	5	$43,718
23	Europe	Wagon	Front	7	5	$30,235
24	Europe	Wagon	Rear	7	2	$42,225
25	USA	SUV	All	7	12	$35,044
26	USA	SUV	Front	7	13	$34,170
27	USA	Sedan	All	7	3	$30,142
28	USA	Sedan	Front	7	71	$23,503
29	USA	Sedan	Rear	7	16	$34,273
30	USA	Sports	Front	7	1	$37,530
31	USA	Sports	Rear	7	8	$46,223
32	USA	Truck	All	7	7	$33,776
33	USA	Truck	Rear	7	9	$22,121
34	USA	Wagon	Front	7	5	$21,629
35	USA	Wagon	Rear	7	2	$24,138

PROC SUMMARY Overview

Example 3 MODIFICATION 2: You only want results for each CLASS variable by itself (and not all those combinations of variables). Use the WAYS command.

```
proc summary data=sashelp.car;
    class origin type drivetrain;
    ways 1;
    var msrp;
    output out=summary_out mean=;
run;
```

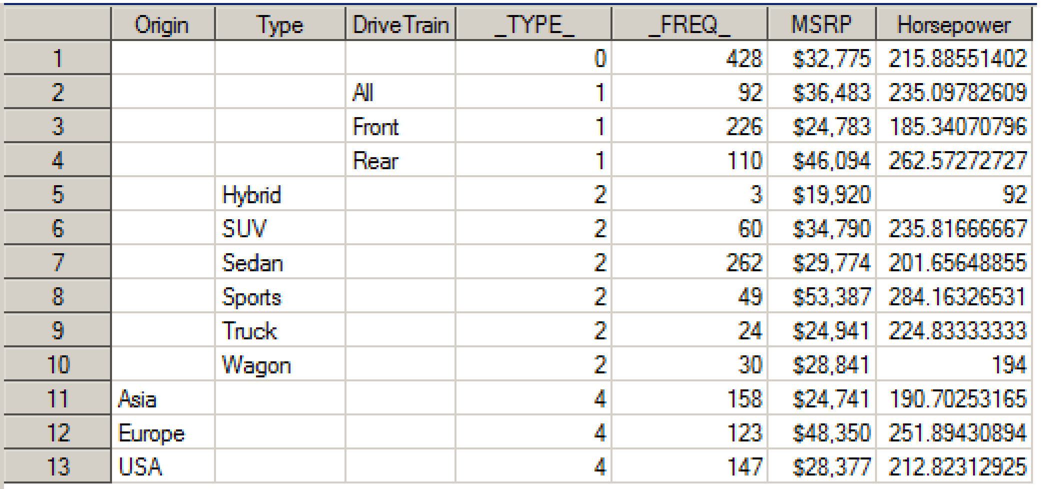


	Origin	Type	DriveTrain	_TYPE_	_FREQ_	MSRP
1			All	1	92	$36,483
2			Front	1	226	$24,783
3			Rear	1	110	$46,094
4		Hybrid		2	3	$19,920
5		SUV		2	60	$34,790
6		Sedan		2	262	$29,774
7		Sports		2	49	$53,387
8		Truck		2	24	$24,941
9		Wagon		2	30	$28,841
10	Asia			4	158	$24,741
11	Europe			4	123	$48,350
12	USA			4	147	$28,377

PROC SUMMARY Overview

Example 3 MODIFICATION 2: You only want results for each CLASS variable by itself (and not all those combinations of variables); plus the overall average; and you want these statistics for MSRP and also for horsepower

```
proc summary data=sashelp.cars;
    class origin type drivetrain;
    ways 0 1;
    var msrp horsepower;
    output out=summary_out mean=;
run;
```



	Origin	Type	DriveTrain	_TYPE_	_FREQ_	MSRP	Horsepower
1				0	428	$32,775	215.88551402
2			All	1	92	$36,483	235.09782609
3			Front	1	226	$24,783	185.34070796
4			Rear	1	110	$46,094	262.57272727
5		Hybrid		2	3	$19,920	92
6		SUV		2	60	$34,790	235.81666667
7		Sedan		2	262	$29,774	201.65648855
8		Sports		2	49	$53,387	284.16326531
9		Truck		2	24	$24,941	224.83333333
10		Wagon		2	30	$28,841	194
11	Asia			4	158	$24,741	190.70253165
12	Europe			4	123	$48,350	251.89430894
13	USA			4	147	$28,377	212.82312925

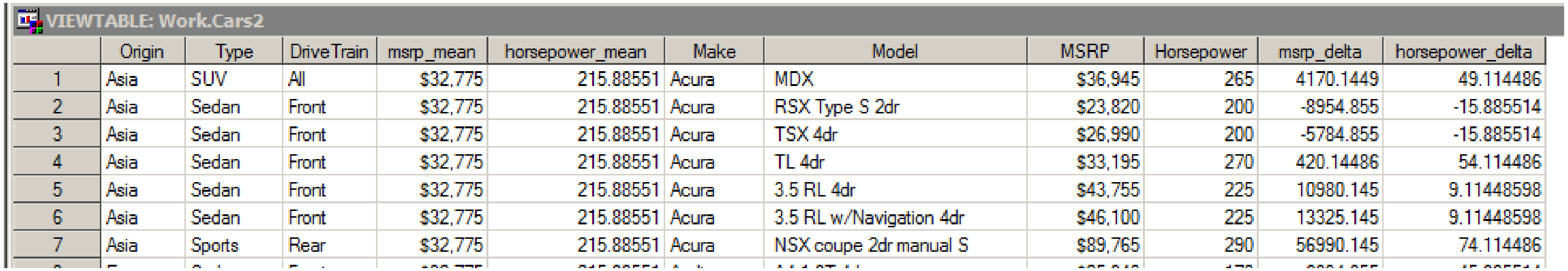
What would the command `ways 2;` produce?

PROC SUMMARY Overview

Example 3 MODIFICATION 3: Now compute a new variable which is the delta from the overall mean of both MSRP and Horsepower. This can’t be done directly in PROC SUMMARY, but it is easily done in a data step.

```
proc summary data=sashelp.cars;
    var msrp horsepower;
    output out=summary_out mean=msrp_mean horsepower_mean;
run;
data cars2;
    if _n_=1 then set summary_out;
    set sashelp.cars;
    msrp_delta=msrp-msrp_mean;
    horsepower_delta=horsepower-horsepower_mean;
run;
```

You can also do this in PROC STDIZE, but that’s a different seminar



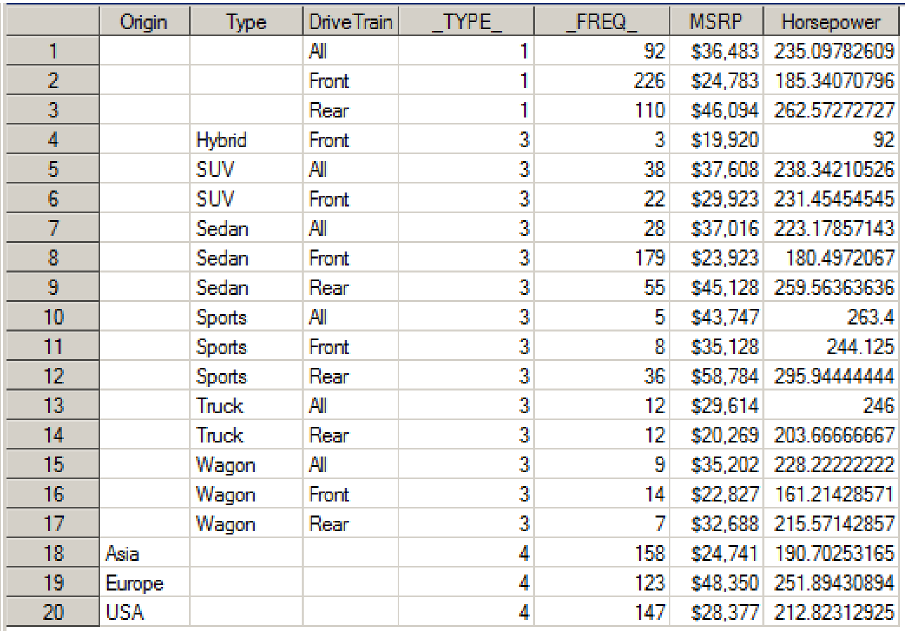
VIEWTABLE: Work.Cars2

	Origin	Type	DriveTrain	msrp_mean	horsepower_mean	Make	Model	MSRP	Horsepower	msrp_delta	horsepower_delta
1	Asia	SUV	All	$32,775	215.88551	Acura	MDX	$36,945	265	4170.1449	49.114486
2	Asia	Sedan	Front	$32,775	215.88551	Acura	RSX Type S 2dr	$23,820	200	-8954.855	-15.885514
3	Asia	Sedan	Front	$32,775	215.88551	Acura	TSX 4dr	$26,990	200	-5784.855	-15.885514
4	Asia	Sedan	Front	$32,775	215.88551	Acura	TL 4dr	$33,195	270	420.14486	54.114486
5	Asia	Sedan	Front	$32,775	215.88551	Acura	3.5 RL 4dr	$43,755	225	10980.145	9.11448598
6	Asia	Sedan	Front	$32,775	215.88551	Acura	3.5 RL w/Navigation 4dr	$46,100	225	13325.145	9.11448598
7	Asia	Sports	Rear	$32,775	215.88551	Acura	NSX coupe 2dr manual S	$89,765	290	56990.145	74.114486

PROC SUMMARY Overview

Example 3 MODIFICATION 4: Suppose you want mean by Origin, by DriveTrain and the combination of Type and DriveTrain. Use the TYPES command instead of the WAYS command.

```
proc summary data=sashelp.cars;
    class origin type drivetrain;
    types origin drivetrain
         type*drivetrain;
    var msrp horsepower;
    output out=summary_out mean=;
run;
```



	Origin	Type	DriveTrain	_TYPE_	_FREQ_	MSRP	Horsepower
1			All	1	92	$36,483	235.09782609
2			Front	1	226	$24,783	185.34070796
3			Rear	1	110	$46,094	262.57272727
4		Hybrid	Front	3	3	$19,920	92
5		SUV	All	3	38	$37,608	238.34210526
6		SUV	Front	3	22	$29,923	231.45454545
7		Sedan	All	3	28	$37,016	223.17857143
8		Sedan	Front	3	179	$23,923	180.4972067
9		Sedan	Rear	3	55	$45,128	259.56363636
10		Sports	All	3	5	$43,747	263.4
11		Sports	Front	3	8	$35,128	244.125
12		Sports	Rear	3	36	$58,784	295.94444444
13		Truck	All	3	12	$29,614	246
14		Truck	Rear	3	12	$20,269	203.66666667
15		Wagon	All	3	9	$35,202	228.22222222
16		Wagon	Front	3	14	$22,827	161.21428571
17		Wagon	Rear	3	7	$32,688	215.57142857
18	Asia			4	158	$24,741	190.70253165
19	Europe			4	123	$48,350	251.89430894
20	USA			4	147	$28,377	212.82312925

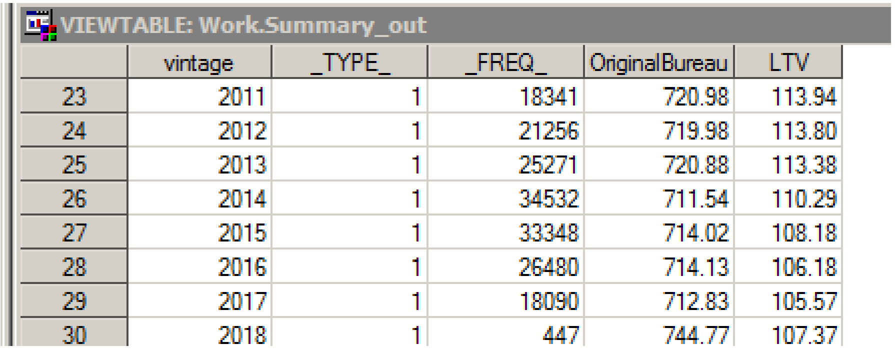
Hint: in the TYPES command, if you want the overall mean, use (for example)

```
types () origin drivetrain type*drivetrain;
```

PROC SUMMARY Overview

Example 4: You want the average of OriginalBureau and LTV, both weighted by LoanBalance, broken down by vintage. These weighted averages should be shown to 2 decimal places.

```
proc summary data=extract nway;
    class vintage;
    var originalbureau ltv;
    weight loanbalance;
    format originalbureau ltv 8.2;
    output out=summary_out mean=;
run;
```



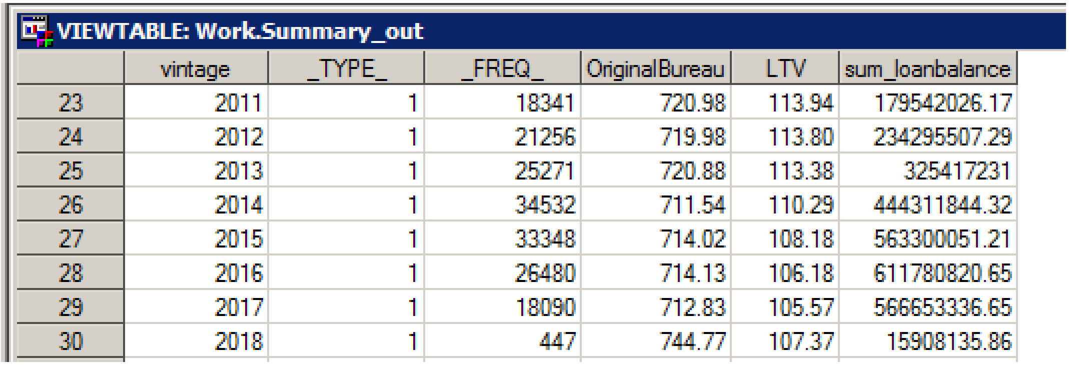
VIEWTABLE: Work.Summary_out

	vintage	_TYPE_	_FREQ_	OriginalBureau	LTV
23	2011	1	18341	720.98	113.94
24	2012	1	21256	719.98	113.80
25	2013	1	25271	720.88	113.38
26	2014	1	34532	711.54	110.29
27	2015	1	33348	714.02	108.18
28	2016	1	26480	714.13	106.18
29	2017	1	18090	712.83	105.57
30	2018	1	447	744.77	107.37

PROC SUMMARY Overview

Example 4 MODIFICATION 1: You want the average of OriginalBureau and LTV, both weighted by LoanBalance, broken down by vintage; and you also want the sum of the values of LoanBalance by Vintage.

```
proc summary data=extract nway;
    class vintage;
    var originalbureau ltv;
    weight loanbalance;
    format originalbureau ltv 8.2;
    output out=summary_out mean= sumwgt=sum_loanbalance;
run;
```



VIEWTABLE: Work.Summary_out

	vintage	_TYPE_	_FREQ_	OriginalBureau	LTV	sum_loanbalance
23	2011	1	18341	720.98	113.94	179542026.17
24	2012	1	21256	719.98	113.80	234295507.29
25	2013	1	25271	720.88	113.38	325417231
26	2014	1	34532	711.54	110.29	444311844.32
27	2015	1	33348	714.02	108.18	563300051.21
28	2016	1	26480	714.13	106.18	611780820.65
29	2017	1	18090	712.83	105.57	566653336.65
30	2018	1	447	744.77	107.37	15908135.86

PROC SUMMARY Overview

Example 4 MODIFICATION 2: You want the average of FICO and LTV, FICO weighted by LoanBalance and LTV unweighted, broken down by vintage.

```
proc summary data=extract nway;
    class vintage;
    var fico/weight=loanbalance;
    var ltv;
    format fico ltv 8.2;
    output out=summary_out mean=;
run;
```

PROC SUMMARY Overview

Example 4 Comment:

A weighted average is easy to do in PROC SQL. Partial code:

```
SUM(a.loanbalance*a.originalbureau)/SUM(a.loanbalance)
```

For a simple example like this, there probably is very little difference in speed doing the analysis in SQL or SUMMARY. If you are computing a lot of statistics in a single SQL call, then it is my experience the PROC SUMMARY is faster. (Applies to all statistics, not just weighted averages)

If you are slicing the data several different ways, this would require several PROC SQL calls, which require SAS to pass through the data several times. But in PROC SUMMARY, you need only one PROC SUMMARY call and then you pass through the data only once.

PROC SUMMARY Overview

Example 4 Comment:

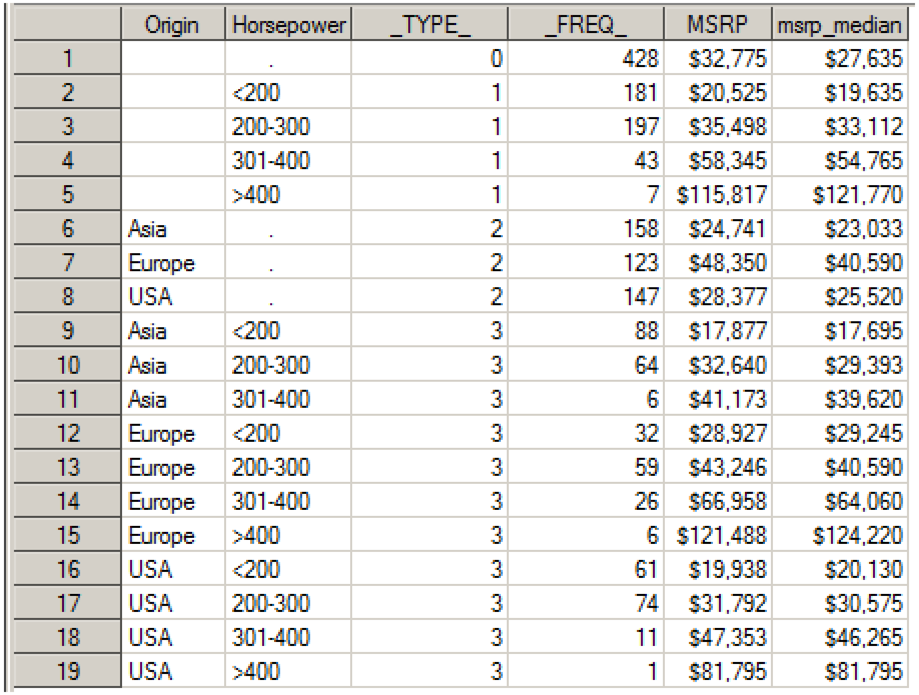
The results between PROC SQL and PROC SUMMARY do not match in the case of a missing value in the numerator of the weighted average. I believe that PROC SUMMARY comes up with the proper result, and PROC SQL is wrong. (Applies to weighted statistics only, not the MEAN function in SQL)

- PROC SUMMARY does not use the record with the missing value in the numerator.
- PROC SQL uses the record with the missing value in the denominator, but not the numerator. So PROC SQL will use a larger denominator, and essentially treats the missing value as a zero in the numerator.

PROC SUMMARY Overview

Example 5: You want an average and median of MSRP by groups of horsepower and by origin. Create a format!

```
proc format;
     value horsef low-199='<200'
        200-300='200-300'
        301-400='301-400'
        401-high='>400';
run;
proc summary data=sashelp.cars;
    class origin horsepower;
    var msrp;
    output out=summary_out mean=
         median=msrp_median;
    format horsepower horsef.;
run;
```



	Origin	Horsepower	_TYPE_	_FREQ_	MSRP	msrp_median
1		.	0	428	$32,775	$27,635
2		<200	1	181	$20,525	$19,635
3		200-300	1	197	$35,498	$33,112
4		301-400	1	43	$58,345	$54,765
5		>400	1	7	$115,817	$121,770
6	Asia	.	2	158	$24,741	$23,033
7	Europe	.	2	123	$48,350	$40,590
8	USA	.	2	147	$28,377	$25,520
9	Asia	<200	3	88	$17,877	$17,695
10	Asia	200-300	3	64	$32,640	$29,393
11	Asia	301-400	3	6	$41,173	$39,620
12	Europe	<200	3	32	$28,927	$29,245
13	Europe	200-300	3	59	$43,246	$40,590
14	Europe	301-400	3	26	$66,958	$64,060
15	Europe	>400	3	6	$121,488	$124,220
16	USA	<200	3	61	$19,938	$20,130
17	USA	200-300	3	74	$31,792	$30,575
18	USA	301-400	3	11	$47,353	$46,265
19	USA	>400	3	1	$81,795	$81,795

Why not something like this:

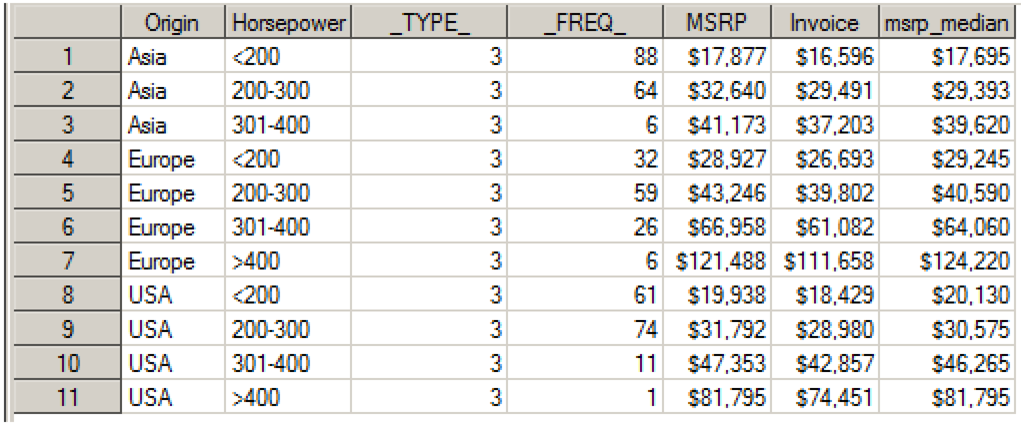
```
if horsepower<200 then hpower='<200’; else ...
```

This will not sort properly; formats sort according to the underlying numeric values.

PROC SUMMARY Overview

Example 5 MODIFIED: You want an average of MSRP and Invoice, and the median of MSRP (but not the median of invoice) by groups of horsepower and origin combined.

```
proc format;
     value horsef low-200='<200'
        201-300='200-300'
        301-400='301-400'
        401-high='>400';
run;
proc summary data=sashelp.cars;
    class origin horsepower;
    var msrp invoice;
    ways 2;
    output out=summary_out mean=
         median(msrp)=msrp_median;
    format horsepower horsef.;
run;
```

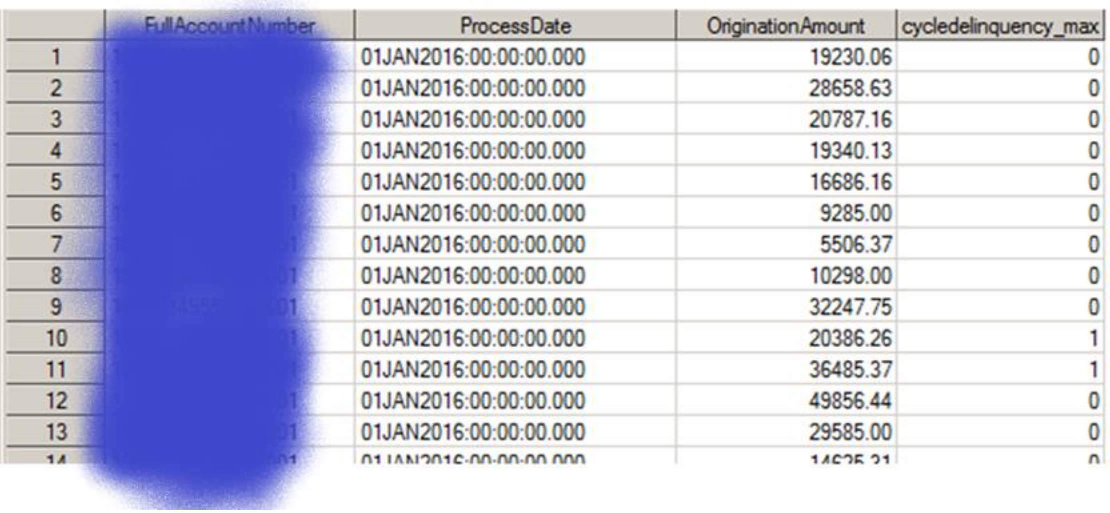


	Origin	Horsepower	_TYPE_	_FREQ_	MSRP	Invoice	msrp_median
1	Asia	<200	3	88	$17,877	$16,596	$17,695
2	Asia	200-300	3	64	$32,640	$29,491	$29,393
3	Asia	301-400	3	6	$41,173	$37,203	$39,620
4	Europe	<200	3	32	$28,927	$26,693	$29,245
5	Europe	200-300	3	59	$43,246	$39,802	$40,590
6	Europe	301-400	3	26	$66,958	$61,082	$64,060
7	Europe	>400	3	6	$121,488	$111,658	$124,220
8	USA	<200	3	61	$19,938	$18,429	$20,130
9	USA	200-300	3	74	$31,792	$28,980	$30,575
10	USA	301-400	3	11	$47,353	$42,857	$46,265
11	USA	>400	3	1	$81,795	$74,451	$81,795

PROC SUMMARY Overview

Example 6: Determine the maximum value of variable CYCLEDELINQUENCY for each loan, keeping identifying information such as PROCESSDATE and ORIGINATIONAMOUNT. Use the ID statement to keep the identifying information in the output.

```
proc summary data=extract nway;
    class fullaccountnumber;
    var cycledelinquency;
    output out=summary_out(drop=_:) max=cycledelinquency_max;
    id processdate originationamount;
run;
```

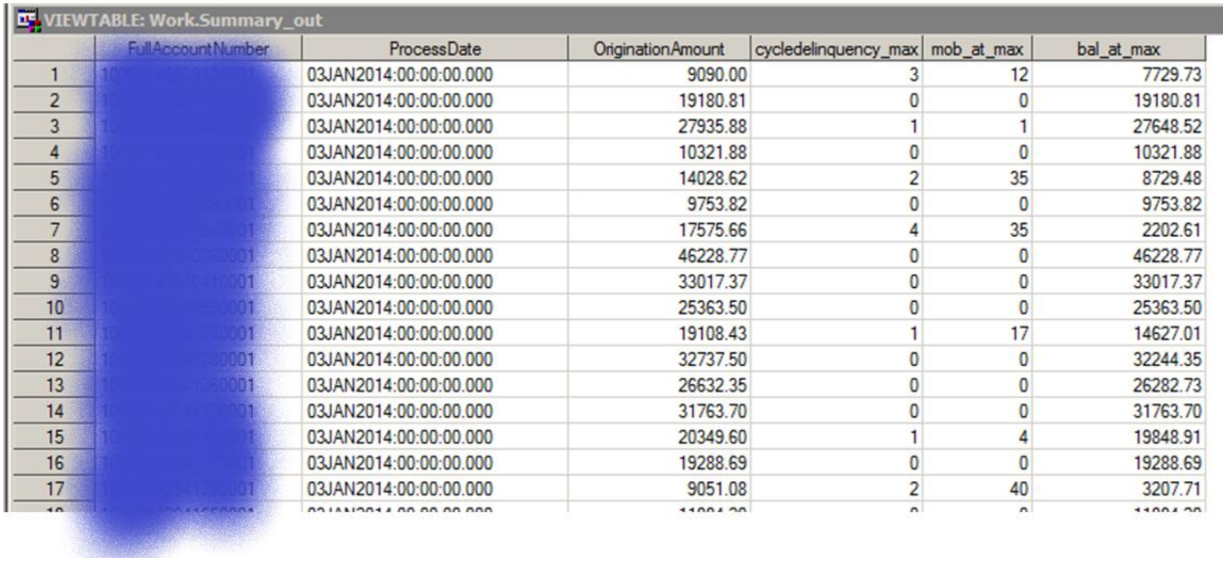


	[redacted]	ProcessDate	OriginationAmount	cycledelinquency_max
1	[redacted]	01JAN2016:00:00:00.000	19230.06	0
2	[redacted]	01JAN2016:00:00:00.000	28658.63	0
3	[redacted]	01JAN2016:00:00:00.000	20787.16	0
4	[redacted]	01JAN2016:00:00:00.000	19340.13	0
5	[redacted]	01JAN2016:00:00:00.000	16686.16	0
6	[redacted]	01JAN2016:00:00:00.000	9285.00	0
7	[redacted]	01JAN2016:00:00:00.000	5506.37	0
8	[redacted]	01JAN2016:00:00:00.000	10298.00	0
9	[redacted]	01JAN2016:00:00:00.000	32247.75	0
10	[redacted]	01JAN2016:00:00:00.000	20386.26	1
11	[redacted]	01JAN2016:00:00:00.000	36485.37	1
12	[redacted]	01JAN2016:00:00:00.000	49856.44	0
13	[redacted]	01JAN2016:00:00:00.000	29585.00	0
14	[redacted]	01JAN2016:00:00:00.000	14625.21	0

PROC SUMMARY Overview

Example 6 MODIFIED: Determine maximum value of CYCLEDELINQUENCY for each loan, keeping identifying information such as PROCESSDATE and ORIGINATIONAMOUNT, and determine the MOB and LOANBALANCE when this maximum CYCLEDELINQUENCY occurred.

```
proc summary data=shaw_extract nway;
    class fullaccountnumber;
    var cycledelinquency;
    output out=summary_out(drop=_:) max=cycledelinquency_max
        maxid(cycledelinquency(mob) cycledelinquency(loanbalance))=
        mob_at_max bal_at_max;
    id processdate originationamount;
run;
```

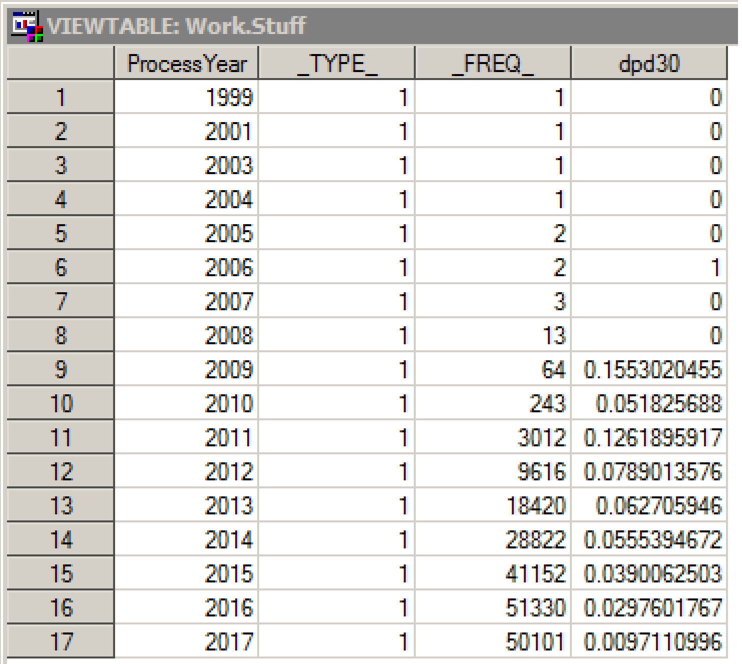

VIEWTABLE: Work.Summary_out

	FullAccountNumber	ProcessDate	OriginationAmount	cycledelinquency_max	mob_at_max	bal_at_max
1	[redacted]	03JAN2014:00:00:00.000	9090.00	3	12	7729.73
2	[redacted]	03JAN2014:00:00:00.000	19180.81	0	0	19180.81
3	[redacted]	03JAN2014:00:00:00.000	27935.88	1	1	27648.52
4	[redacted]	03JAN2014:00:00:00.000	10321.88	0	0	10321.88
5	[redacted]	03JAN2014:00:00:00.000	14028.62	2	35	8729.48
6	[redacted]	03JAN2014:00:00:00.000	9753.82	0	0	9753.82
7	[redacted]	03JAN2014:00:00:00.000	17575.66	4	35	2202.61
8	[redacted]	03JAN2014:00:00:00.000	46228.77	0	0	46228.77
9	[redacted]	03JAN2014:00:00:00.000	33017.37	0	0	33017.37
10	[redacted]	03JAN2014:00:00:00.000	25363.50	0	0	25363.50
11	[redacted]	03JAN2014:00:00:00.000	19108.43	1	17	14627.01
12	[redacted]	03JAN2014:00:00:00.000	32737.50	0	0	32244.35
13	[redacted]	03JAN2014:00:00:00.000	26632.35	0	0	26282.73
14	[redacted]	03JAN2014:00:00:00.000	31763.70	0	0	31763.70
15	[redacted]	03JAN2014:00:00:00.000	20349.60	1	4	19848.91
16	[redacted]	03JAN2014:00:00:00.000	19288.69	0	0	19288.69
17	[redacted]	03JAN2014:00:00:00.000	9051.08	2	40	3207.71

PROC SUMMARY Overview

Example 7: Computing proportions and weighted proportions. If you have a binary (0 or 1) variable, then the mean is the proportion of units that have a 1.

```
proc sql;
    create table extract as select
    case when cycledelinquency>1 then 1 else 0 end
as dpd30,processyear,loanbalance
    from mydatabase;
quit;
proc summary nway data=extract;
    class processyear;
    var dpd30;
    weight loanbalance;
    output out=stuff mean=;
run;
```

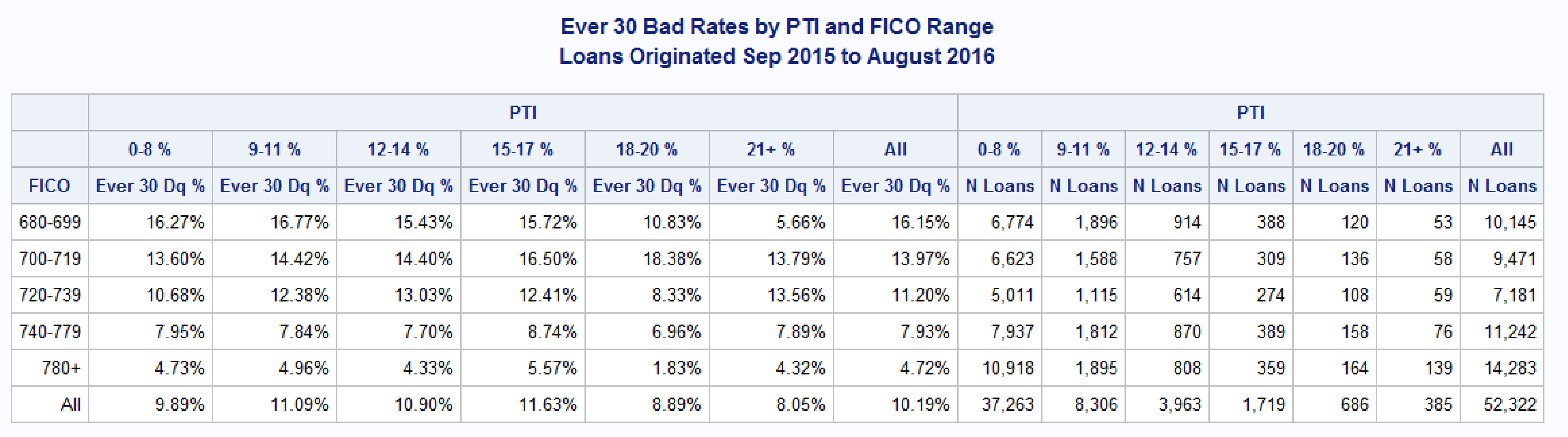

VIEWTABLE: Work.Stuff

	ProcessYear	_TYPE_	_FREQ_	dpd30
1	1999	1	1	0
2	2001	1	1	0
3	2003	1	1	0
4	2004	1	1	0
5	2005	1	2	0
6	2006	1	2	1
7	2007	1	3	0
8	2008	1	13	0
9	2009	1	64	0.1553020455
10	2010	1	243	0.051825688
11	2011	1	3012	0.1261895917
12	2012	1	9616	0.0789013576
13	2013	1	18420	0.062705946
14	2014	1	28822	0.0555394672
15	2015	1	41152	0.0390062503
16	2016	1	51330	0.0297601767
17	2017	1	50101	0.0097110996

PROC SUMMARY Overview

Example 8: Compute Ever 30 % Delinquent by FICO and PTI bands, table of results created by PROC REPORT

First we present the output:



Ever 30 Bad Rates by PTI and FICO Range
Loans Originated Sep 2015 to August 2016

	PTI							PTI						
	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %	All	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %	All
FICO	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans
680-699	16.27%	16.77%	15.43%	15.72%	10.83%	5.66%	16.15%	6,774	1,896	914	388	120	53	10,145
700-719	13.60%	14.42%	14.40%	16.50%	18.38%	13.79%	13.97%	6,623	1,588	757	309	136	58	9,471
720-739	10.68%	12.38%	13.03%	12.41%	8.33%	13.56%	11.20%	5,011	1,115	614	274	108	59	7,181
740-779	7.95%	7.84%	7.70%	8.74%	6.96%	7.89%	7.93%	7,937	1,812	870	389	158	76	11,242
780+	4.73%	4.96%	4.33%	5.57%	1.83%	4.32%	4.72%	10,918	1,895	808	359	164	139	14,283
All	9.89%	11.09%	10.90%	11.63%	8.89%	8.05%	10.19%	37,263	8,306	3,963	1,719	686	385	52,322

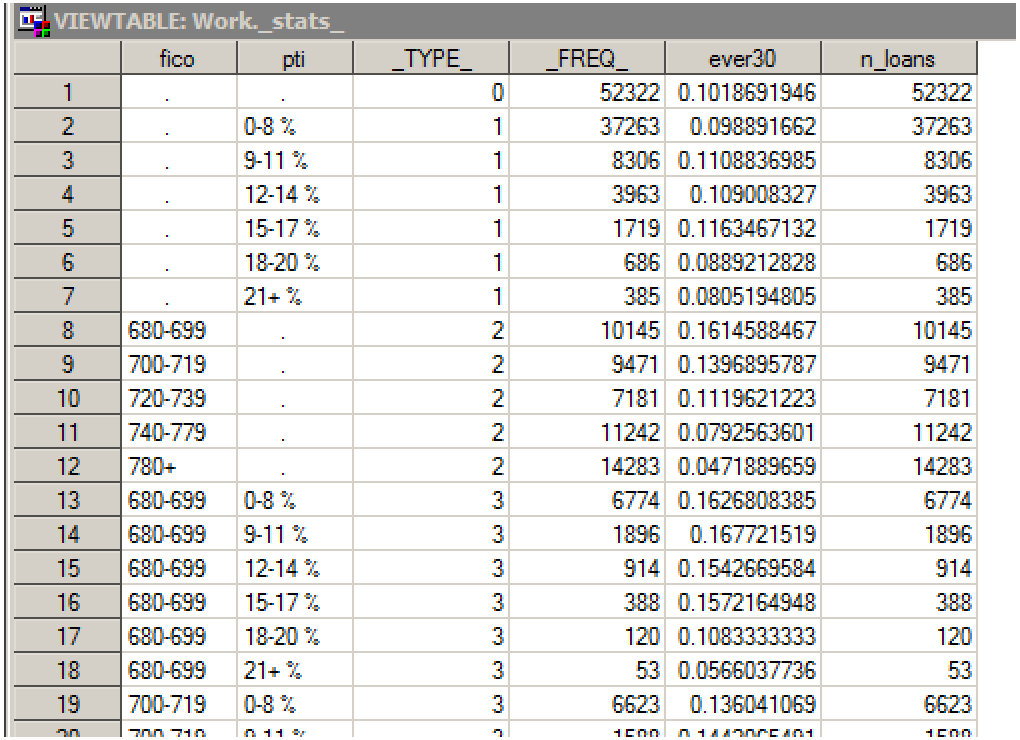
PROC SUMMARY Overview

```
/* Create FICO and PTI formats that define the buckets */
proc format;
    value ficof 680-699='680-699' 700-719='700-719' 720-739='720-739' 740-779='740-779'
        780-9998='780+' 9999='All';
    value ptif 0-8='0-8 %' 9-11='9-11 %' 12-14='12-14 %' 15-17='15-17 %' 18-20='18-20 %'
        21-9998='21+ %' 9999='All';
run;

/* Extract data */
proc sql;
    create table deal_review as select
        a.fullaccountnumber,a.processdate,
        fico format=ficof.,
        pti format=ptif.,
        case when a.cycledelinquency>1 then 1 else 0 end as ever30
    from mydatabase a;
quit;
```

PROC SUMMARY Overview

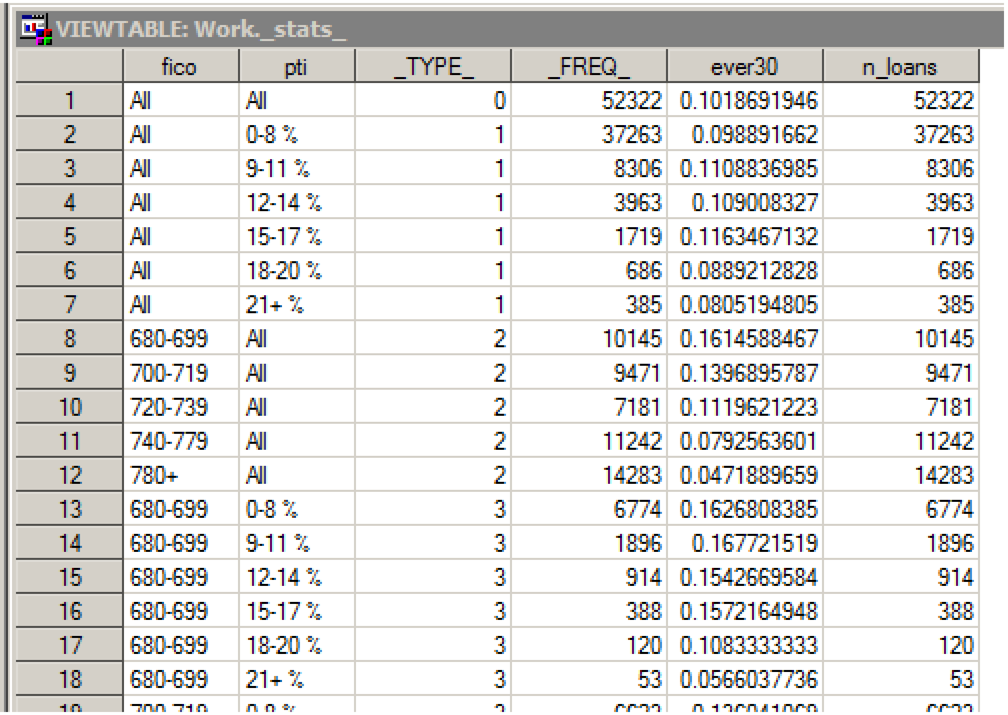
```
/* Compute statistics, since variable ever30 is binary 0/1, the mean is the percent ever
30 delinquent */
proc summary data=deal_review;
    class fico pti;
    var ever30;
    output out=_stats_ mean= n=n_loans;
run;
```



	fico	pti	_TYPE_	_FREQ_	ever30	n_loans
1	.	.	0	52322	0.1018691946	52322
2	.	0-8 %	1	37263	0.098891662	37263
3	.	9-11 %	1	8306	0.1108836985	8306
4	.	12-14 %	1	3963	0.109008327	3963
5	.	15-17 %	1	1719	0.1163467132	1719
6	.	18-20 %	1	686	0.0889212828	686
7	.	21+ %	1	385	0.0805194805	385
8	680-699	.	2	10145	0.1614588467	10145
9	700-719	.	2	9471	0.1396895787	9471
10	720-739	.	2	7181	0.1119621223	7181
11	740-779	.	2	11242	0.0792563601	11242
12	780+	.	2	14283	0.0471889659	14283
13	680-699	0-8 %	3	6774	0.1626808385	6774
14	680-699	9-11 %	3	1896	0.167721519	1896
15	680-699	12-14 %	3	914	0.1542669584	914
16	680-699	15-17 %	3	388	0.1572164948	388
17	680-699	18-20 %	3	120	0.1083333333	120
18	680-699	21+ %	3	53	0.0566037736	53
19	700-719	0-8 %	3	6623	0.136041069	6623
20	700-719	9-11 %	3	1588	0.1442065491	1588

PROC SUMMARY Overview

```
/* Assign value 9999 to missing fico or missing pti, so it will be formatted as the word
'All' */
data _stats_;
    set _stats_;
    if missing(fico) then fico=9999;
    if missing(pti) then pti=9999;
run;
```



VIEWTABLE: Work._stats_

	fico	pti	_TYPE_	_FREQ_	ever30	n_loans
1	All	All	0	52322	0.1018691946	52322
2	All	0-8 %	1	37263	0.098891662	37263
3	All	9-11 %	1	8306	0.1108836985	8306
4	All	12-14 %	1	3963	0.109008327	3963
5	All	15-17 %	1	1719	0.1163467132	1719
6	All	18-20 %	1	686	0.0889212828	686
7	All	21+ %	1	385	0.0805194805	385
8	680-699	All	2	10145	0.1614588467	10145
9	700-719	All	2	9471	0.1396895787	9471
10	720-739	All	2	7181	0.1119621223	7181
11	740-779	All	2	11242	0.0792563601	11242
12	780+	All	2	14283	0.0471889659	14283
13	680-699	0-8 %	3	6774	0.1626808385	6774
14	680-699	9-11 %	3	1896	0.167721519	1896
15	680-699	12-14 %	3	914	0.1542669584	914
16	680-699	15-17 %	3	388	0.1572164948	388
17	680-699	18-20 %	3	120	0.1083333333	120
18	680-699	21+ %	3	53	0.0566037736	53
19	700-719	0-8 %	3	6632	[illegible]	6632

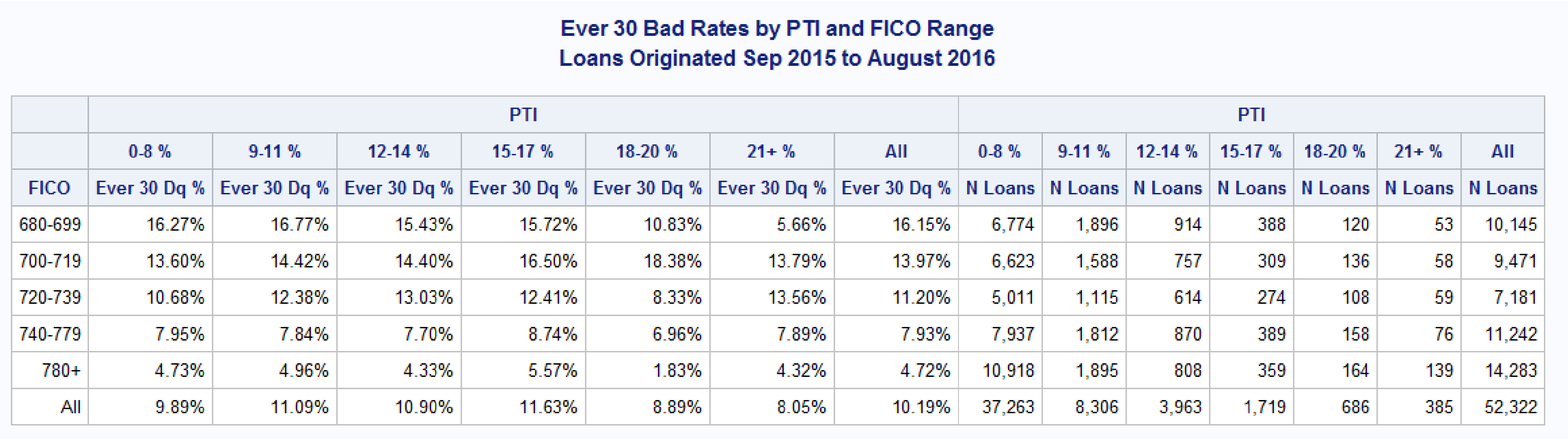
PROC SUMMARY Overview

```
/* Produce report */

title "Ever 30 Bad Rates by PTI and FICO Range";
title2 "Loans Originated Sep 2015 to August 2016";

proc report data=_stats_;
    columns fico pti,(ever30) pti,(n_loans);
    define fico/group "FICO" order=internal;
    define pti/across "PTI" order=internal;
    define ever30/analysis sum "Ever 30 Dq %" format=percent9.2;
    define n_loans/analysis sum "N Loans" format=comma8.0;
run;
```

`order = internal` forces PROC REPORT to order the columns and rows in the proper numerical order



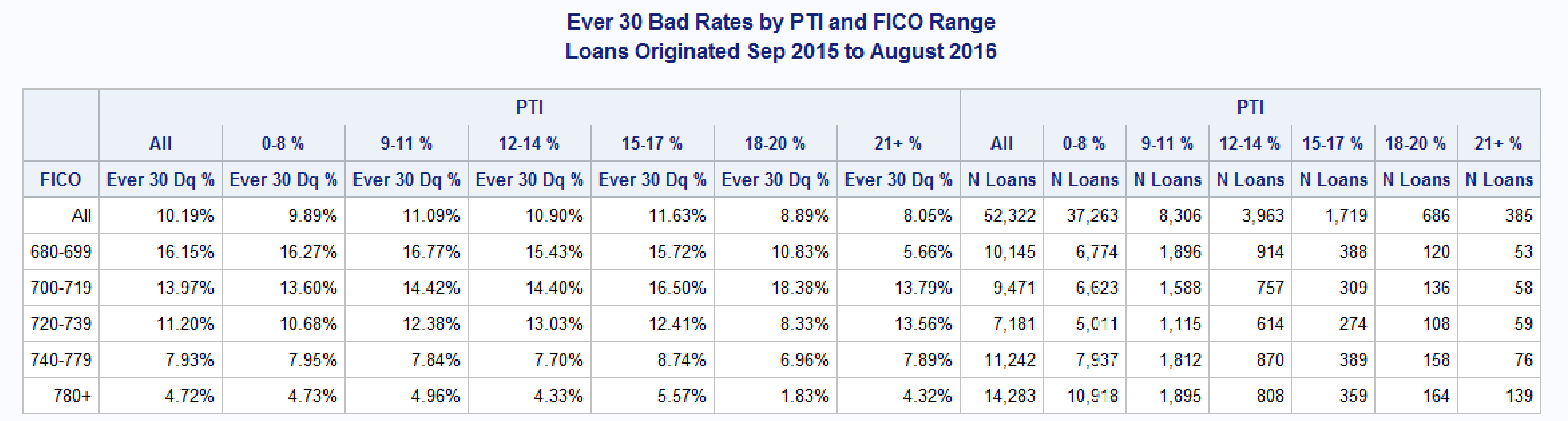
Ever 30 Bad Rates by PTI and FICO Range
Loans Originated Sep 2015 to August 2016

	PTI							PTI						
	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %	All	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %	All
FICO	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans
680-699	16.27%	16.77%	15.43%	15.72%	10.83%	5.66%	16.15%	6,774	1,896	914	388	120	53	10,145
700-719	13.60%	14.42%	14.40%	16.50%	18.38%	13.79%	13.97%	6,623	1,588	757	309	136	58	9,471
720-739	10.68%	12.38%	13.03%	12.41%	8.33%	13.56%	11.20%	5,011	1,115	614	274	108	59	7,181
740-779	7.95%	7.84%	7.70%	8.74%	6.96%	7.89%	7.93%	7,937	1,812	870	389	158	76	11,242
780+	4.73%	4.96%	4.33%	5.57%	1.83%	4.32%	4.72%	10,918	1,895	808	359	164	139	14,283
All	9.89%	11.09%	10.90%	11.63%	8.89%	8.05%	10.19%	37,263	8,306	3,963	1,719	686	385	52,322

PROC SUMMARY Overview

What would happen if we used –9999 instead of 9999??

```
proc format;
    value fico 680-699='680-699' 700-719='700-719' 720-739='720-739' 740-779='740-779‘
        780-9998='780+' -9999='All';
    value ptif 0-8='0-8 %' 9-11='9-11 %' 12-14='12-14 %' 15-17='15-17 %' 18-20='18-20 %'
        21-9998='21+ %' -9999='All';
run;
proc summary data=deal_review;
    class fico pti;
    var ever30;
    output out=_stats_ mean= n=n_loans;
run;
data _stats_;
    set _stats_;
    if missing(fico) then fico=-9999;
    if missing(pti) then pti=-9999;
run;
```



Ever 30 Bad Rates by PTI and FICO Range
Loans Originated Sep 2015 to August 2016

	PTI							PTI						
	All	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %	All	0-8 %	9-11 %	12-14 %	15-17 %	18-20 %	21+ %
FICO	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	Ever 30 Dq %	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans	N Loans
All	10.19%	9.89%	11.09%	10.90%	11.63%	8.89%	8.05%	52,322	37,263	8,306	3,963	1,719	686	385
680-699	16.15%	16.27%	16.77%	15.43%	15.72%	10.83%	5.66%	10,145	6,774	1,896	914	388	120	53
700-719	13.97%	13.60%	14.42%	14.40%	16.50%	18.38%	13.79%	9,471	6,623	1,588	757	309	136	58
720-739	11.20%	10.68%	12.38%	13.03%	12.41%	8.33%	13.56%	7,181	5,011	1,115	614	274	108	59
740-779	7.93%	7.95%	7.84%	7.70%	8.74%	6.96%	7.89%	11,242	7,937	1,812	870	389	158	76
780+	4.72%	4.73%	4.96%	4.33%	5.57%	1.83%	4.32%	14,283	10,918	1,895	808	359	164	139

PROC SUMMARY Overview

- Hashtag
  Feel free to use the following hashtag on social media to indicate to all your friends how cool you are now

#ProcSummaryRulez

Contact: PaigeMiller at
communities.sas.com