

Unit 2-3 Exercises

1. Creating Accumulating Totals

The data set **orion.order_fact** contains information about orders for several years, sorted by **Order_Date**. Each observation represents one order, and **Total_Retail_Price** contains the sales value for the order.

Partial Listing of **orion.order_fact** (617 Total Observations, 12 Total Variables)

Order_ID	Order_Date	Total_Retail_Price
1230058123	11JAN2003	\$16.50
1230080101	15JAN2003	\$247.50
1230106883	20JAN2003	\$28.30
1230147441	28JAN2003	\$32.00
1230315085	27FEB2003	\$63.60
1230333319	02MAR2003	\$234.60

- a. Orion Star would like to examine growth in sales during the date range of 01Nov2004 to 14Dec2004.
- Open file **p203e01**. It creates and prints a data set named **work.mid_q4** from **orion.order_fact**. The DATA step uses the following WHERE statement to read only the observations in the specified date range.

```
where '01nov2004'd <= Order_Date <= '14dec2004'd;
```

- Modify the program to create an accumulating total, **Sales2Dte**, to display the sales-to-date total.
 - Also create an accumulating variable, **Num_Orders**, indicating how many orders-to-date that total represents. Each observation counts as one order.
- b. Modify the PROC PRINT step to show your results.
- Display **Sales2Dte** with a DOLLAR10.2 format.
 - Show only the columns **Order_ID**, **Order_Date**, **Total_Retail_Price**, **Sales2Dte**, and **Num_Orders**.

PROC PRINT Output



Orders from 01Nov2004 through 14Dec2004					
Obs	Order_ID	Order_Date	Total_Retail_Price	Sales2Dte	Num_Orders
1	1234033037	01NOV2004	\$53.70	\$53.70	1
2	1234092222	10NOV2004	\$7.20	\$60.90	2
3	1234133789	17NOV2004	\$328.30	\$389.20	3
4	1234186330	25NOV2004	\$200.10	\$589.30	4
5	1234198497	26NOV2004	\$39.00	\$628.30	5
6	1234235150	02DEC2004	\$369.00	\$997.30	6
7	1234247283	03DEC2004	\$187.20	\$1,184.50	7
8	1234255111	04DEC2004	\$257.40	\$1,441.90	8
9	1234279341	08DEC2004	\$116.70	\$1,558.60	9
10	1234301319	11DEC2004	\$105.60	\$1,664.20	10

2. Creating Accumulating Totals with Conditional Logic

The data set **orion.order_fact** contains a group of orders across several years, sorted by **Order_Date**.

Partial Listing of **orion.order_fact** (617 Total Observations, 12 Total Variables)

Order_ID	Order_ Type	Order_ Date	Quantity
1230058123	1	11JAN2003	1
1230080101	2	15JAN2003	1
1230106883	2	20JAN2003	1
1230147441	1	28JAN2003	2
1230315085	1	27FEB2003	3
1230333319	2	02MAR2003	1

- a. Orion Star would like to analyze 2005 data by creating accumulating totals for the number of items sold from retail, catalog, and Internet channels.
 - The value of **Order_Type** indicates whether the sale was retail (=1), catalog (=2), or Internet (=3).
 - Create a data set named **work.typetotals** with accumulating totals for **TotalRetail**, **TotalCatalog**, and **TotalInternet**, as described above.
 -  The variable **Quantity** contains the number of items sold for each order.
 - For testing your program in this step, read only the first 10 observations that satisfy the WHERE statement.
 -  Remember to process only those rows where **Order_Date** occurs in 2005.
- b. Continue testing your program by printing the results from step a. Print all the variables and check to make sure that the program is correctly calculating values for the accumulating totals.

PROC PRINT Output

Obs	Customer_ID	Employee_ID	Street_ID	Order_ Date	Delivery_ Date	Order_ ID	Order_ Type	Product_ID
1	195	120150	1600101663	02JAN2005	02JAN2005	1234437760	1	230100600028
2	36	99999999	9260128237	11JAN2005	14JAN2005	1234534069	3	240800100026
3	183	120121	1600100760	12JAN2005	12JAN2005	1234537441	1	240100200001
4	16	99999999	3940105865	12JAN2005	14JAN2005	1234538390	2	220200300015
5	16	99999999	3940105865	17JAN2005	19JAN2005	1234588648	2	230100500101
6	16	99999999	3940105865	17JAN2005	19JAN2005	1234588648	2	230100600024
7	16	99999999	3940105865	24JAN2005	26JAN2005	1234659163	2	230100700008
8	63	99999999	9260125492	24JAN2005	25JAN2005	1234665265	2	240100100063
9	171	99999999	1600101555	29JAN2005	02FEB2005	1234709803	3	220100100304
10	183	120179	1600100760	31JAN2005	31JAN2005	1234727966	1	240700400004

Obs	Quantity	Total_Retail_ Price	CostPrice_ Per_Unit	Discount	Total Retail	Total Catalog	Total Internet
1	2	\$193.40	\$48.45	.	2	0	0
2	4	\$525.20	\$58.55	.	2	0	4
3	1	\$16.00	\$6.35	.	3	0	4
4	1	\$115.00	\$52.40	.	3	1	4
5	1	\$138.70	\$62.50	.	3	2	4
6	1	\$76.10	\$38.15	.	3	3	4
7	1	\$504.20	\$245.80	.	3	4	4
8	2	\$48.40	\$9.75	.	3	6	4
9	2	\$122.60	\$30.75	.	3	6	6
10	1	\$13.20	\$5.95	.	4	6	6

c. When the results from steps **a** and **b** are correct, do the following:

- Modify the program to read all observations satisfying the WHERE statement.
- Keep only the variables **Order_Date**, **Order_ID**, **TotalRetail**, **TotalCatalog**, and **TotalInternet**.
- Print your results with an appropriate title.

Partial PROC PRINT Output (90 Total Observations)

2005 Accumulating Totals for Each Type of Order						
Obs	Order_ Date	Order_ID	Total Retail	Total Catalog	Total Internet	
1	02JAN2005	1234437760	2	0	0	
2	11JAN2005	1234534069	2	0	4	
3	12JAN2005	1234537441	3	0	4	
4	12JAN2005	1234538390	3	1	4	
5	17JAN2005	1234588648	3	2	4	
6	17JAN2005	1234588648	3	3	4	
7	24JAN2005	1234659163	3	4	4	
8	24JAN2005	1234665265	3	6	4	
9	29JAN2005	1234709803	3	6	6	
10	31JAN2005	1234727966	4	6	6	
11	16FEB2005	1234891576	4	6	7	
12	16FEB2005	1234897732	5	6	7	
13	22FEB2005	1234958242	5	7	7	

3. Creating Accumulating Totals by Month

The data set **orion.order_fact** contains a group of orders across several years, sorted by **Order_Date**.

Partial Listing of **orion.order_fact** (617 Total Observations, 12 Total Variables)

Order_ID	Order_Date	Total_Retail_Price
1230058123	11JAN2003	\$16.50
1230080101	15JAN2003	\$247.50
1230106883	20JAN2003	\$28.30
1230147441	28JAN2003	\$32.00
1230315085	27FEB2003	\$63.60
1230333319	02MAR2003	\$234.60

Orion Star would like to generate the following report showing all orders in 2007 along with an accumulating total.

- The accumulating total should reset to zero at the start of each new month.
- Remember to process only those rows where **Order_Date** occurs in 2007.

Partial PROC PRINT Output

Accumulating Totals by Month in 2007				
Obs	Order_Date	Order_ID	Total_Retail_Price	MonthSales
1	02JAN2007	1241054779	\$195.60	\$195.60
2	03JAN2007	1241063739	\$160.80	\$356.40
3	04JAN2007	1241066216	\$306.20	\$662.60
4	06JAN2007	1241086052	\$37.80	\$700.40
5	13JAN2007	1241147641	\$362.60	\$1,063.00
6	23JAN2007	1241235281	\$72.60	\$1,135.60
7	24JAN2007	1241244297	\$258.20	\$1,393.80
8	24JAN2007	1241244297	\$81.20	\$1,475.00
9	24JAN2007	1241244297	\$358.20	\$1,833.20
10	25JAN2007	1241263172	\$102.40	\$1,935.60
11	25JAN2007	1241263172	\$113.20	\$2,048.80
12	28JAN2007	1241286432	\$174.40	\$2,223.20
13	29JAN2007	1241298131	\$37.40	\$2,260.60
14	05FEB2007	1241359997	\$117.60	\$117.60
15	07FEB2007	1241371145	\$656.60	\$774.20
16	07FEB2007	1241371145	\$129.00	\$903.20

4. Summarizing Data Using the DATA Step

The data set **orion.order_summary** contains information about sales in a particular year for each customer, separated by month. For a given customer, there might be some months that he did not place an order.

Partial Listing of **orion.order_summary** (101 Total Observations)

Customer_ID	Order_ Month	Sale_Amt
5	5	478.00
5	6	126.80
5	9	52.50
5	12	33.80
10	3	32.60
10	4	250.80
10	5	79.80

- Sort the input data set, **orion.order_summary**, by **Customer_ID**. Use the OUT= option to avoid overwriting the original data set. Name the output data set **work.sumsort**.
- Create a new data set showing a total sales value for each customer.
 - Name the new data set **work.customers**.
 - Name the new variable **Total_Sales**. This variable contains the total of sales across all months for each customer.
- Print your result.
 - Display **Total_Sales** with a DOLLAR11.2 format.
 - Add an appropriate title.

Partial PROC PRINT Output (37 Total Observations)

Total Sales to each Customer		
Obs	Customer_ID	Total_Sales
1	5	\$691.10
2	10	\$3,479.09
3	11	\$78.20
4	12	\$253.20
5	18	\$29.40
6	24	\$358.80
7	27	\$1,093.60
8	31	\$1,777.60

5. Summarizing and Grouping Data Using the DATA Step

The data set **orion.order_qtrsum** contains information about sales in a particular year for each customer, separated by month.

- For a given customer, there might be some months (and quarters) that the customer did not place an order.
- The variable **Order_Qtr** contains the appropriate quarter.

Partial Listing of **orion.order_qtrsum** (101 Total Observations)

Customer_ID	Order_ Qtr	Order_ Month	Sale_Amt
69	4	10	3.2
70187	4	11	8.2
10	2	6	12.2
70079	4	10	14.6
70165	3	7	16.6
92	1	3	16.9
41	3	8	17.6
171	2	4	19.1
41	2	5	19.9
69	3	9	23.5
49	4	12	24.8



The data set is not sorted by **Customer_ID** and **Order_Qtr**.

- Create a data set named **work.qtrcustomers** that summarizes sales based on customer and quarter.
 - The variable **Total_Sales** should contain the total sales for each quarter within each **Customer_ID** value.
 - Create a variable named **Num_Months** that counts the total months within each quarter that the customer had an order.
- Print your results.
 - Display **Total_Sales** with a DOLLAR11.2 format.
 - Add an appropriate title.

Partial PROC PRINT Output (74 Total Observations)

Total Sales to each Customer for each Quarter					
Obs	Customer_ID	Order_ Qtr	Total_Sales	Num_ Months	
1	5	2	\$604.80	2	
2	5	3	\$52.50	1	
3	5	4	\$33.80	1	
4	10	1	\$32.60	1	
5	10	2	\$342.80	3	
6	10	3	\$1,065.79	2	
7	10	4	\$2,037.90	2	
8	11	3	\$78.20	1	
9	12	1	\$117.60	1	

6. Identifying Extreme Values in Each Group of Data

The data set `orion.customer_dim` contains information about Orion Star customers.

Partial Listing of `orion.customer_dim` (77 Total Observations, 11 Total Variables)

Customer_ID	Customer_Name	Customer_Type	Customer_BirthDate
4	James Kvarniq	Orion Club members low activity	27JUN1974
5	Sandrina Stephano	Orion Club Gold members medium activity	09JUL1979
9	Cornelia Krah1	Orion Club Gold members medium activity	27FEB1974
10	Karen Ballinger	Orion Club members high activity	18OCT1984
11	Elke Wallstab	Orion Club members high activity	16AUG1974
12	David Black	Orion Club members medium activity	12APR1969
13	Markus Sepke	Orion Club Gold members low activity	21JUL1988
16	Ulrich Heyde	Internet/Catalog Customers	16JAN1939
17	Jimmie Evans	Orion Club members medium activity	17AUG1954
18	Tonie Asmussen	Orion Club members low activity	02FEB1954

Use First./Last. processing to create the report below. Show data on the oldest and youngest customers for each **Customer_Type**.

- The variable **o_ID** is the **Customer_ID** value of the oldest customer and **y_ID** is the **Customer_ID** value of the youngest customer for each group.
- Create a variable named **agerange** to indicate the spread between these oldest and youngest customers.
- Use **Customer_BirthDate**, rather than **Customer_Age**, for all age determinations because this is more accurate.

PROC PRINT Output

Oldest and Youngest Customers of each Customer Type					
Customer_Type	oldest	youngest	o_ID	y_ID	agerange
Internet/Catalog Customers	08JUL1934	18AUG1969	29	54655	35.1
Orion Club members high activity	28SEP1934	24OCT1986	89	46966	52.1
Orion Club members medium activity	20JAN1934	16SEP1988	70059	2806	54.7
Orion Club Gold members high activity	16JAN1934	25JUL1984	50	39	50.5
Orion Club Gold members low activity	19DEC1969	21JUL1988	70201	13	18.6
Orion Club Gold members medium activity	16MAY1949	09JUL1979	215	5	30.1
Orion Club members low activity	14APR1939	18JUN1984	70108	75	45.2