PROC SQL Fundamentals

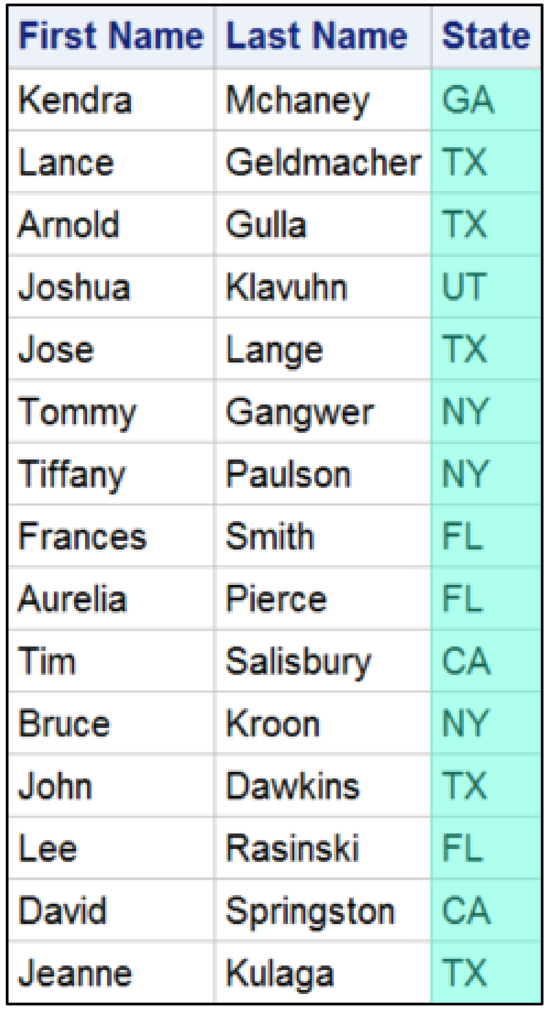
1. Generating Simple Reports

2. Summarizing and Grouping Data

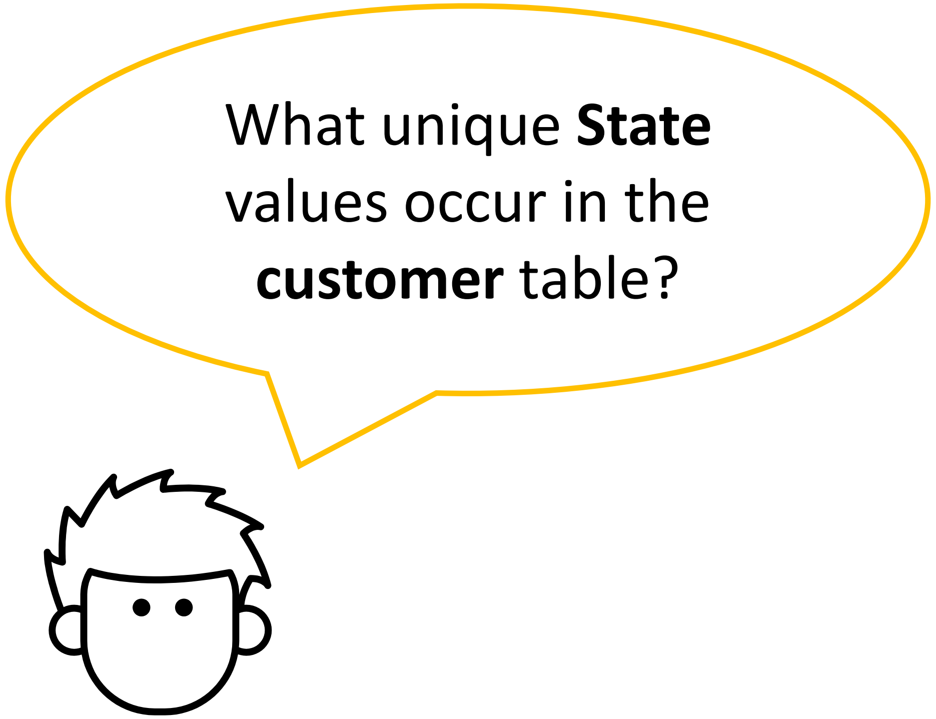
3. Creating and Managing Tables

Eliminating Duplicate Rows

customer



First Name	Last Name	State
Kendra	Mchaney	GA
Lance	Geldmacher	TX
Arnold	Gulla	TX
Joshua	Klavuhn	UT
Jose	Lange	TX
Tommy	Gangwer	NY
Tiffany	Paulson	NY
Frances	Smith	FL
Aurelia	Pierce	FL
Tim	Salisbury	CA
Bruce	Kroon	NY
John	Dawkins	TX
Lee	Rasinski	FL
David	Springston	CA
Jeanne	Kulaga	TX

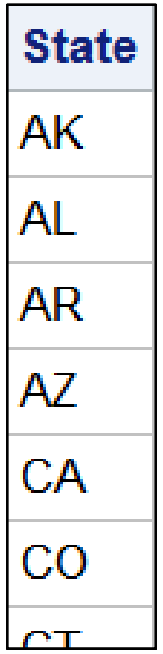

What unique **State** values occur in the **customer** table?

Distinct Keyword

PROC SQL;
SELECT DISTINCT *col-name <,col-name>*
    FROM *input-table*
QUIT;

The **DISTINCT** keyword applies to all columns in the SELECT list.

```
proc sql;
select distinct State
    from sq.customer;
quit;
```



State
AK
AL
AR
AZ
CA
CO
CT

2.07 Activity

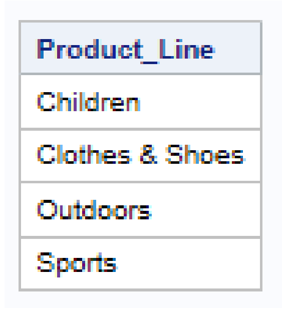
- Import the Products table and perform the following tasks to eliminate duplicate values in a table. (Note, imported Products table will be temporarily saved into WORK folder.)

1. Write a proc sql step to SELECT * FROM Products. Run the query, view the results.
2. Delete * and write **distinct Product_Line** in the SELECT clause. ORDER BY **Product_Line.** Run the query. What does this query show?
3. Add the **Product_Category** column in the SELECT clause after the **Product_Line** column. Run the query. What does this query show?

2.07 Activity – Correct Answer

2. What does this query show? **It displays the unique values of *product lines* in the Product_Line column.**

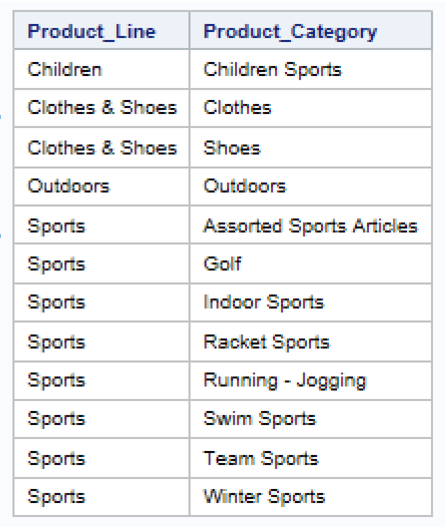
```
select distinct Product_Line
```



Product_Line
Children
Clothes & Shoes
Outdoors
Sports

3. What does this query show? **It displays unique combinations of Product_Line and Product_Category values.**

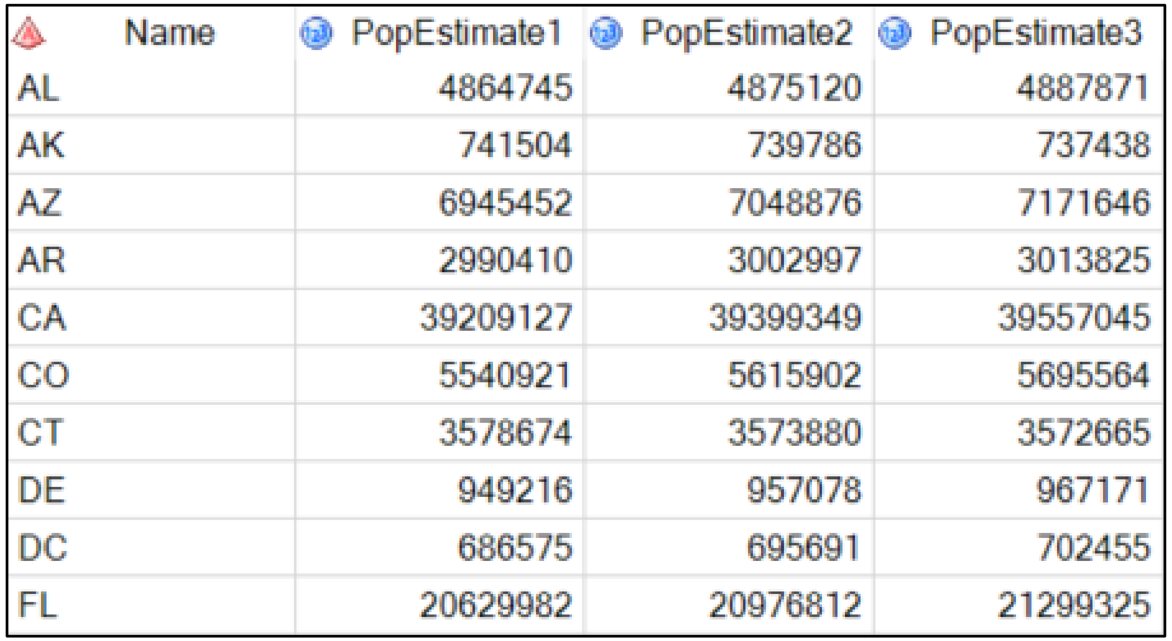
```
select distinct Product_Line, Product_Category
```



Product_Line	Product_Category
Children	Children Sports
Clothes & Shoes	Clothes
Clothes & Shoes	Shoes
Outdoors	Outdoors
Sports	Assorted Sports Articles
Sports	Golf
Sports	Indoor Sports
Sports	Racket Sports
Sports	Running - Jogging
Sports	Swim Sports
Sports	Team Sports
Sports	Winter Sports

Summarizing Data

statepopulation



Name	PopEstimate1	PopEstimate2	PopEstimate3
AL	4864745	4875120	4887871
AK	741504	739786	737438
AZ	6945452	7048876	7171646
AR	2990410	3002997	3013825
CA	39209127	39399349	39557045
CO	5540921	5615902	5695564
CT	3578674	3573880	3572665
DE	949216	957078	967171
DC	686575	695691	702455
FL	20629982	20976812	21299325

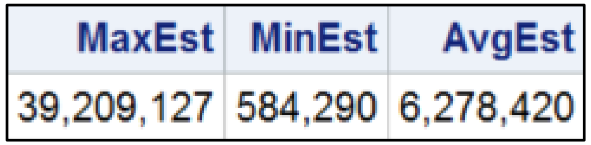
summarize data

Summary Functions: Down a Column

ANSI

SELECT *summary function(column)*;

```
proc sql;
select max(PopEstimate1) as MaxEst format=comma16.,
       min(PopEstimate1) as MinEst format=comma16.,
       avg(PopEstimate1) as AvgEst format=comma16.
    from sq.statepopulation;
quit;
```

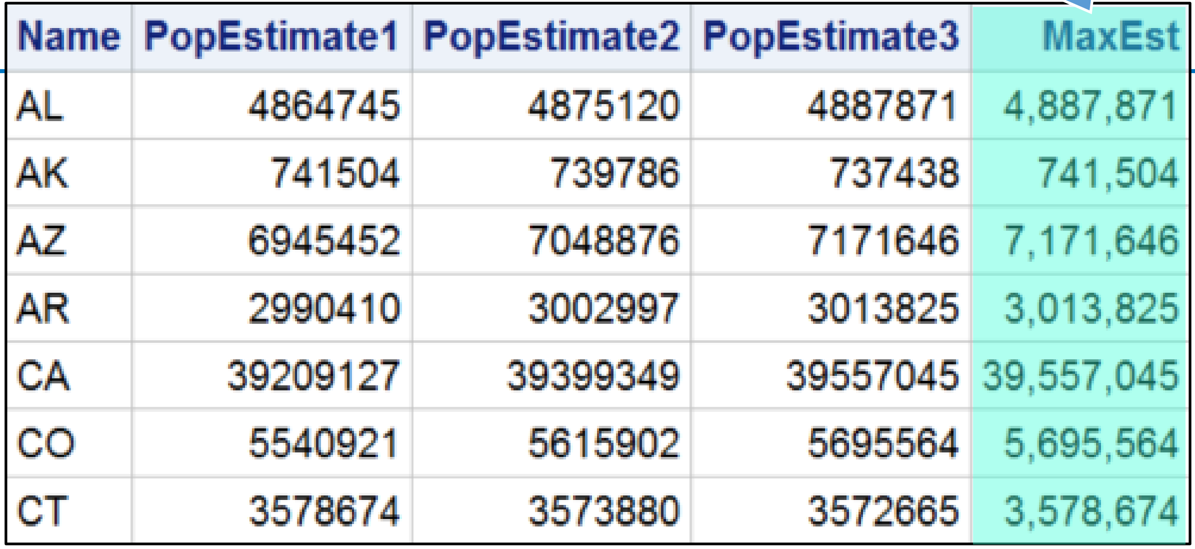


MaxEst	MinEst	AvgEst
39,209,127	584,290	6,278,420

Summary Functions: Across a Row

SELECT *summary function(column1, column-n);*

```
proc sql;
select Name, PopEstimate1, PopEstimate2, PopEstimate3,
       max(PopEstimate1, PopEstimate2, PopEstimate3)
                as MaxEst format=comma16.
    from sq.statepopulation;
quit;
```



Name	PopEstimate1	PopEstimate2	PopEstimate3	MaxEst
AL	4864745	4875120	4887871	4,887,871
AK	741504	739786	737438	741,504
AZ	6945452	7048876	7171646	7,171,646
AR	2990410	3002997	3013825	3,013,825
CA	39209127	39399349	39557045	39,557,045
CO	5540921	5615902	5695564	5,695,564
CT	3578674	3573880	3572665	3,578,674

Commonly Used Summary Functions

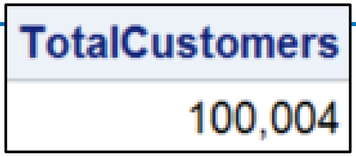
SQL	SAS	Returned Value
AVG	MEAN	Mean (average) value
COUNT	FREQ, N	Number of nonmissing values
MAX	MAX	Largest value
MIN	MIN	Smallest nonmissing value
SUM	SUM	Sum of nonmissing values
	NMISS	Number of missing values
	STD	Standard deviation
	VAR	Variance

Summarizing Data Using the COUNT Function

An asterisk specifies all rows.

SELECT COUNT(*argument*)**;**

```
proc sql;
select count(*) as TotalCustomers format=comma12.
    from sq.customer;
quit;
```



TotalCustomers
100,004

2.08 Activity

- Use the products table and perform the following tasks to summarize a table using the COUNT function:

1. Use **COUNT(*)** to count the number of products in the list.
2. Inside the COUNT function, add the DISTINCT keyword in front of the **Product_Category** column and run the query.
3. In the same select statement, count distinct product lines as well.

2.08 Activity – Correct Answer

1. Use **COUNT(*)** to count the number of products in the list.

```
proc sql;
select count(*) as totalNumProducts
from WORK.products;
quit;
```

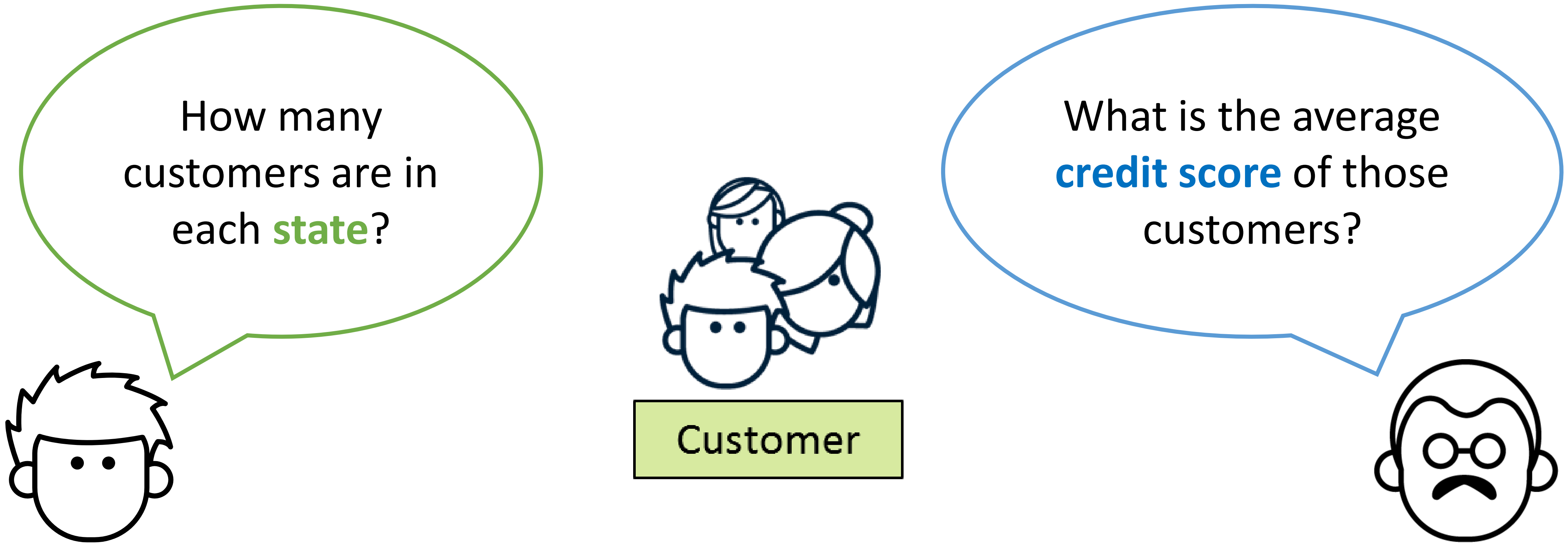
2. Inside the COUNT function, add the DISTINCT keyword in front of the **Product_Category** column and run the query.

```
proc sql;
select count(distinct Product_Category) as totalNumProductCategories,
from WORK.products;
quit;
```

3. In the same select statement, count distinct product lines as well.

```
proc sql;
select count(distinct Product_Category) as totalNumProductCategories,
      count(distinct Product_Line) as totalNumProductLines
from WORK.products;
quit;
```

Grouping Data

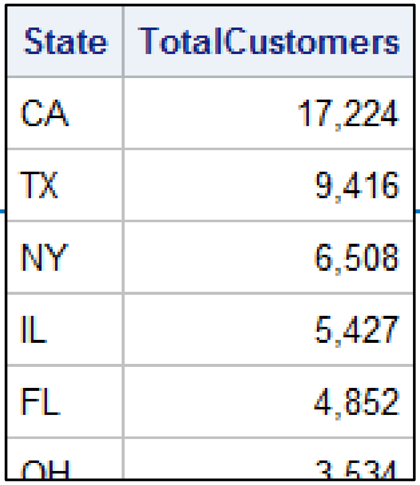

How many customers are in each **state**?
Customer
What is the average **credit score** of those customers?

Grouping Data

The **GROUP BY** clause summarizes groups of data by a specified ***column*** or ***columns***.

SELECT *col-name, summary function(column)*
    FROM *input-table*
    WHERE *expression*
    GROUP BY *col-name <,col-name>*
    ORDER BY *col-name* **DESC***;*

```
select State, count(*) as TotalCustomers format=comma7.
    from sq.customer
    where BankID is not null
    group by State
    order by TotalCustomers desc;
```



State	TotalCustomers
CA	17,224
TX	9,416
NY	6,508
IL	5,427
FL	4,852
OH	3,534

Filtering Grouped Data

The **HAVING** clause instructs PROC SQL how to ***filter*** the data after the data is summarized.

SELECT *col-name, summary function(column*

FROM *input-table*
WHERE *expression*
GROUP BY *col-name <,col-name>*
HAVING *expression*
ORDER BY *col-name* **DESC**;

```
select State, count(*) as TotalCustomers format=comma7.
    from sq.customer
    where BankID is not null
    group by State
    having TotalCustomers > 6000
    order by TotalCustomers desc;
```

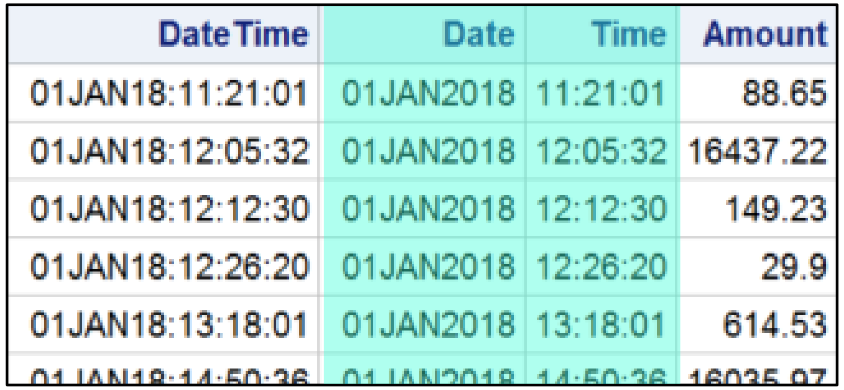
State	TotalCustomers
CA	17,224
TX	9,416
NY	6,508

Extracting Data from a Datetime Value

DATEPART(*datetime-value***)**

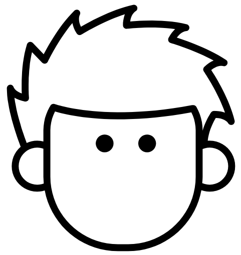
TIMEPART(*datetime-value***)**

```
select DateTime,
       datepart(DateTime) as Date format=date9.,
       timepart(DateTime) as Time format=time.,
       Amount
   from sq.transaction;
```



DateTime	Date	Time	Amount
01JAN18:11:21:01	01JAN2018	11:21:01	88.65
01JAN18:12:05:32	01JAN2018	12:05:32	16437.22
01JAN18:12:12:30	01JAN2018	12:12:30	149.23
01JAN18:12:26:20	01JAN2018	12:26:20	29.9
01JAN18:13:18:01	01JAN2018	13:18:01	614.53
01JAN18:14:50:36	01JAN2018	14:50:36	16035.97

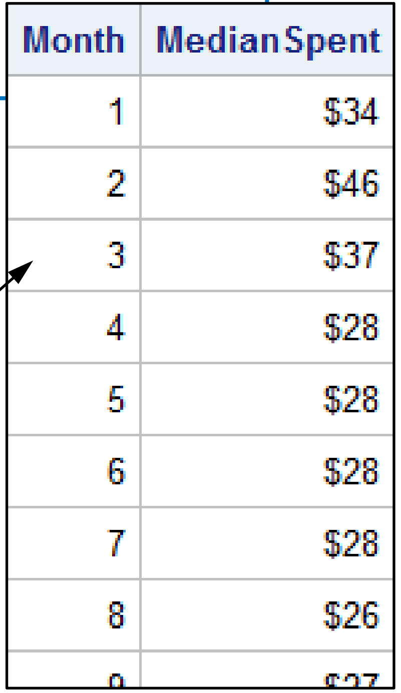
Extract the ***date*** and ***time*** values from the **DateTime** column.



Summarizing Data by Month

```
select month(datepart(DateTime)) as Month,
       Median(Amount) as MedianSpent format=dollar16.
   from sq.transaction
   group by Month;
```

Wrap the DATEPART function inside the MONTH function to extract the numeric month.



Month	MedianSpent
1	$34
2	$46
3	$37
4	$28
5	$28
6	$28
7	$28
8	$26
9	$27

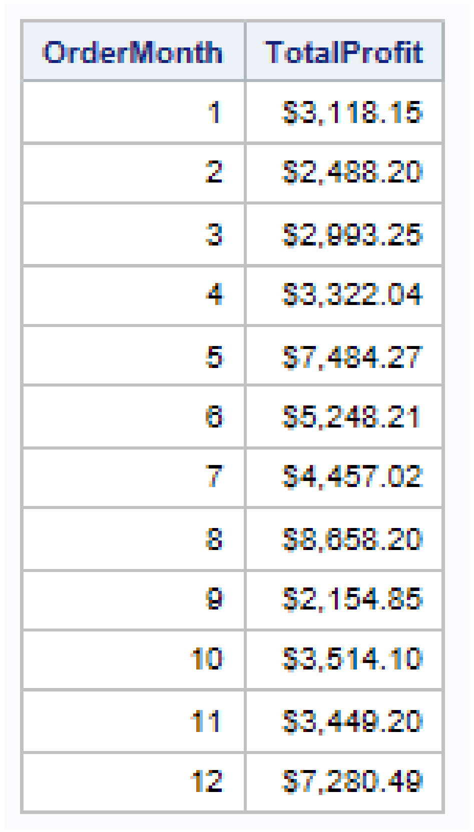
2.09 Activity

- Use Orders dataset and perform the following tasks to summarize data using date functions:

1. Which month has the highest value for total **Profit**?

2.09 Activity – Correct Answer

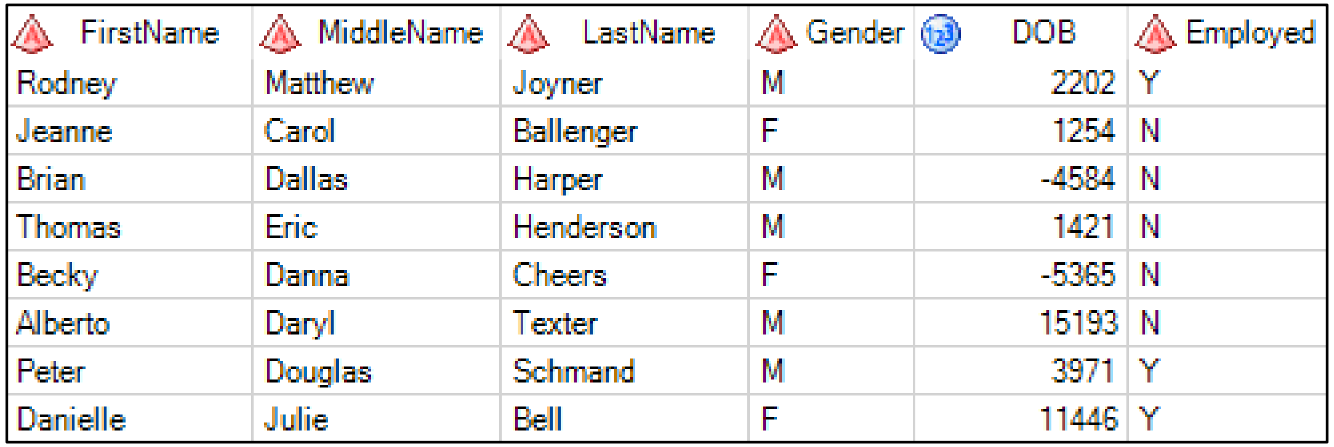
1. Which month has the highest value for total **Profit**? **Month 8, August**



OrderMonth	TotalProfit
1	$3,118.15
2	$2,488.20
3	$2,993.25
4	$3,322.04
5	$7,484.27
6	$5,248.21
7	$4,457.02
8	$8,658.20
9	$2,154.85
10	$3,514.10
11	$3,449.20
12	$7,280.49

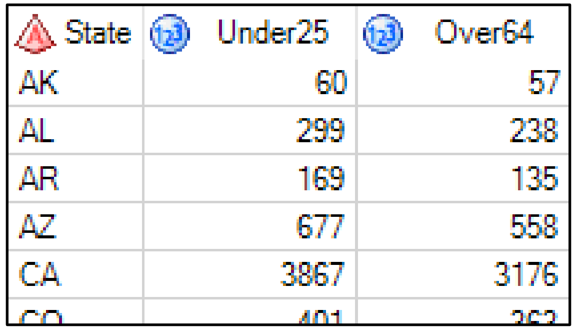
Counting Rows That Meet a Specified Criterion

customer

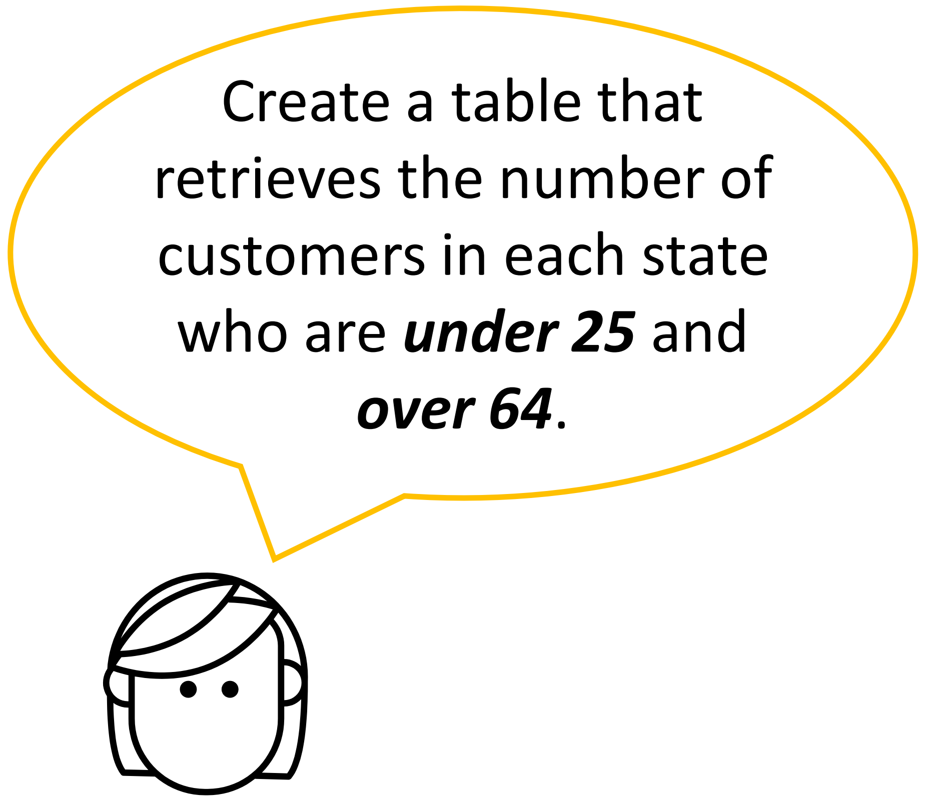


FirstName	MiddleName	LastName	Gender	DOB	Employed
Rodney	Matthew	Joyner	M	2202	Y
Jeanne	Carol	Ballenger	F	1254	N
Brian	Dallas	Harper	M	-4584	N
Thomas	Eric	Henderson	M	1421	N
Becky	Danna	Cheers	F	-5365	N
Alberto	Daryl	Texter	M	15193	N
Peter	Douglas	Schmand	M	3971	Y
Danielle	Julie	Bell	F	11446	Y

customercount



State	Under25	Over64
AK	60	57
AL	299	238
AR	169	135
AZ	677	558
CA	3867	3176
CO	401	262

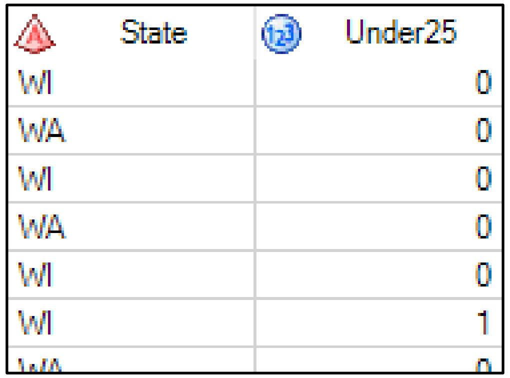

Create a table that retrieves the number of customers in each state who are ***under 25*** and ***over 64***.

Using Boolean Expressions

```
create table CustomerCount as
select State,
       yrdif(DOB,"01JAN2021"d,'age')<25 as Under25
   from sq.customer;
```

If **Age** is less than 25, the value is 1 (true).

If **Age** is greater than 25, the value is 0 (false).



State	Under25
WI	0
WA	0
WI	0
WA	0
WI	0
WI	1

Syntax Summary

SELECT DISTINCT *col-name, col-name*

Eliminate Duplicates

```
SELECT col-name, summary function(column)
       FROM input-table
       GROUP BY col-name;
```

GROUP BY

```
SELECT summary function(column)
SELECT summary function(column1, column2)
```

Summarize Data

WHERE *expression*

Filtering Rows

HAVING *expression*

Filtering Summarized Data

```
COUNT(*)
COUNT(col-name)
COUNT(DISTINCT col-name)
```

COUNT Function