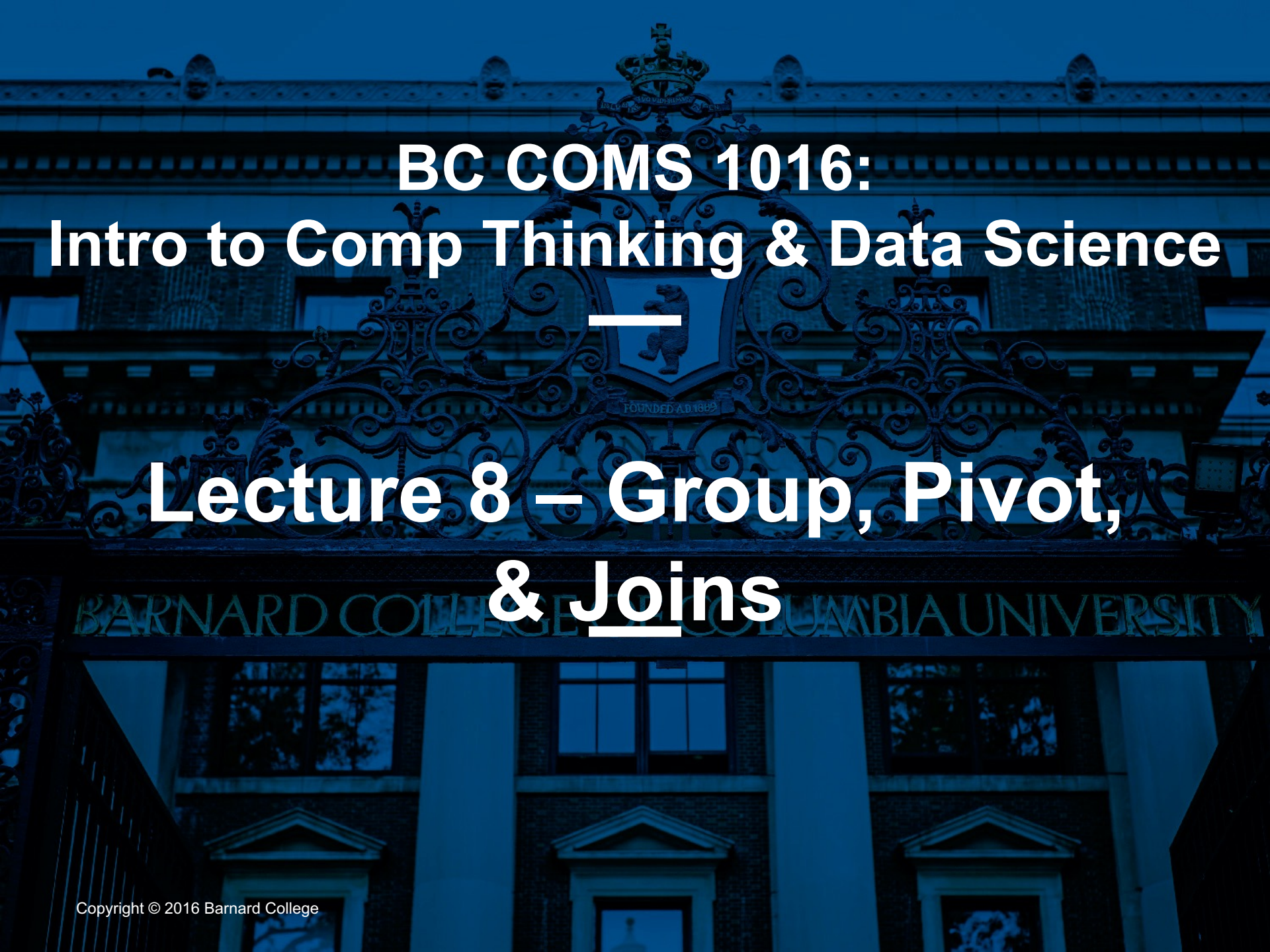


BC COMS 1016: Intro to Comp Thinking & Data Science

—

Lecture 8 – Group, Pivot, & Joins

Announcements

- HW03 - Functions, Histograms, and Groups
  - Due Monday (02/21)
- Checkpoint/Project 1:
  - Will come out this week, will have 2 weeks
  - HW03 and HW04 are short
- This week's lab:
  - Either cancelling or splitting it into 2

How to ask for help

- Explain what you are trying to do
- Give a minimal example
  - Someone else should be able to replicate the problem easily
  - Shouldn't require any data/information that only you have
- Explain what you **think** should happen
- Explain what you get instead (copy / paste or screenshot if you can)
- Explain what else you've tried

Taken from Jordan Boyd Graber

Zoom poll 1

Applying Functions to Columns

The apply method creates an array by calling a function on every element in input column(s)

- First argument: Function to apply
- Other arguments: The input column(s)

```
table_name.apply(function_name, 'column_label')
```

Zoom poll 2

Grouping by One Column

The group method aggregates all rows with the same value for a column into a single row in the resulting table.

- First argument: Which column to group by
- Second argument: (Optional) How to combine values

- len — number of grouped values (default)
- list — list of all grouped values
- sum — total of all grouped values

Lists as Generic Sequences

A list is a sequence of values (just like an array), but the values can all have different types

```
[2+3, 'four', Table().with_column('K', [3, 4])]
```

- Lists can be used to create table rows.
- If you create a table column from a list, it will be converted to an array automatically

Grouping by Multiple Columns

The group method can also aggregate all rows that share the combination of values in multiple columns

- First argument: A list of which columns to group by
- Second argument: (Optional) How to combine values

Pivot Tables

- Cross-classifies according to two categorical variables
- Produces a grid of counts or aggregated values
- Two required arguments:
  - First: variable that forms column labels of grid
  - Second: variable that forms row labels of grid
- Two optional arguments (include **both** or **neither**)
- values='column_label_to_aggregate'
- collect=function_to_aggregate_with

Pivot Example

```
sky = Table.read_table('skyscrapers_v2.csv')
```

Pivot Example

sky = Table.read_table('skyscrapers_v2.csv')

name	material	city	height	age
One World Trade Center	mixed/composite	New York City	541.3	6
Willis Tower	steel	Chicago	442.14	46
432 Park Avenue	concrete	New York City	425.5	5

Pivot Example

sky = Table.read_table('skyscrapers_v2.csv')

name	material	city	height	age
One World Trade Center	mixed/composite	New York City	541.3	6
Willis Tower	steel	Chicago	442.14	46
432 Park Avenue	concrete	New York City	425.5	5

sky.pivot('material', 'city')

Pivot description

Tbl.pivot(col1, col2)

1. **string**: name of column whose unique values will make up columns of pivot table
2. **string**: name of column whose unique values will make up rows of pivot table

Pivot description

Tbl.pivot(col1, col2)

1. **string**: name of column whose unique values will make up columns of pivot table
2. **string**: name of column whose unique values will make up rows of pivot table

sky.pivot('material', 'city')

Pivot description

sky.pivot('material', 'city')

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago			
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago			
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago			1
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago			1
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1		1
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1		2
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta			
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1		3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta		1	
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1		3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta		1	
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1		3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
[illegible]	[illegible]	[illegible]
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta		1	
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1	1	3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
[illegible]	[illegible]	[illegible]
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta		1	
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	1	1	3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
[illegible]	[illegible]	[illegible]
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot description

sky.pivot('material', 'city')

city	concrete	mixed/composite	steel
Atlanta		1	
Austin			
Baltimore			
Boston			
Charlotte			
Chicago	2	1	3
Cincinnati			
Cleveland			
Columbus			

name	material	city
Willis Tower	steel	Chicago
Trump International Hotel & Tower	concrete	Chicago
Aon Center	steel	Chicago
John Hancock Center	steel	Chicago
Bank of America Plaza	mixed/composite	Atlanta
U.S. Bank Tower	steel	Los Angeles
The Franklin - North Tower	mixed/composite	Chicago
JPMorgan Chase Tower	mixed/composite	Houston
Two Prudential Plaza	concrete	Chicago
Wells Fargo Plaza	steel	Houston

Pivot Tables

- Cross-classifies according to two categorical variables
- Produces a grid of counts or aggregated values
- Two required arguments:
  - First: variable that forms column labels of grid
  - Second: variable that forms row labels of grid
- Two optional arguments (include **both** or **neither**)
- values='column_label_to_aggregate'
- collect=function_to_aggregate_with

Group vs Pivot

Pivot	Group
One combo of grouping variables **per entry**	One combo of grouping variables **per row**
Two grouping variables: columns and rows	**Any number** of grouping variables
Aggregate values of **values column**	Aggregate values of **all other columns** in table
Missing combos **= 0 (or empty string)**	Missing combos **absent**

Joining Two Tables

tblA.join(colA, tblB, colB)

tblA.join(colA, tblB)

Table Review

t.select(column, …) or t.drop(column, …)
t.take([row, …]) or t.exclude([row, …])
t.sort(column, descending=False)
t.where(column, are.condition(…))
t.apply(function, column, …)
t.group(column) or t.group(column, function)
t.group([column, …]) or t.group([column, …], function)
t.pivot(cols, rows) or t.pivot(cols, rows, vals, function)
t.join(column, other_table, other_table_column)

https://coms1016.barnard.edu/python-reference.html

Comparisons

Comparison Operators

Operator	Table predicate
==	are.equal_to
!=	are.not_equal_to
>	are.above
>=	are.above_or_equal_to
<	are.below
<=	are.below_or_equal_to

The result of a comparison expression is a bool value:

True, False

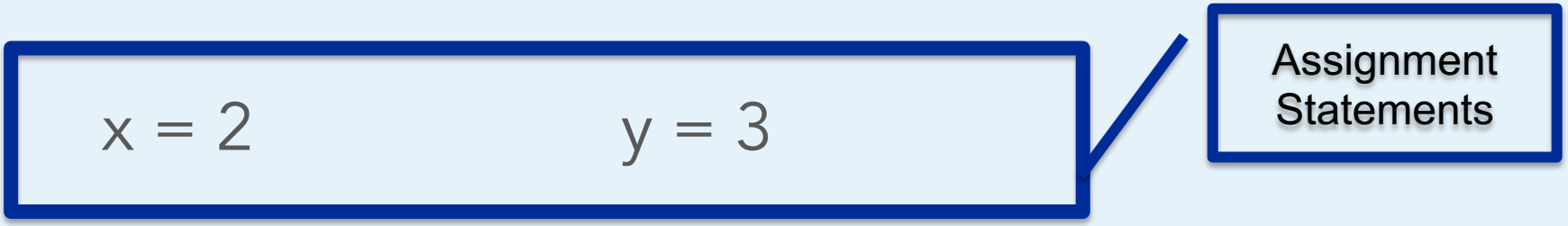
Comparison Operators

The result of a comparison expression is a bool value

x = 2 y = 3

Comparison Operators

The result of a comparison expression is a bool value



Comparison Operators

The result of a comparison expression is a bool value

```
x = 2          y = 3
```

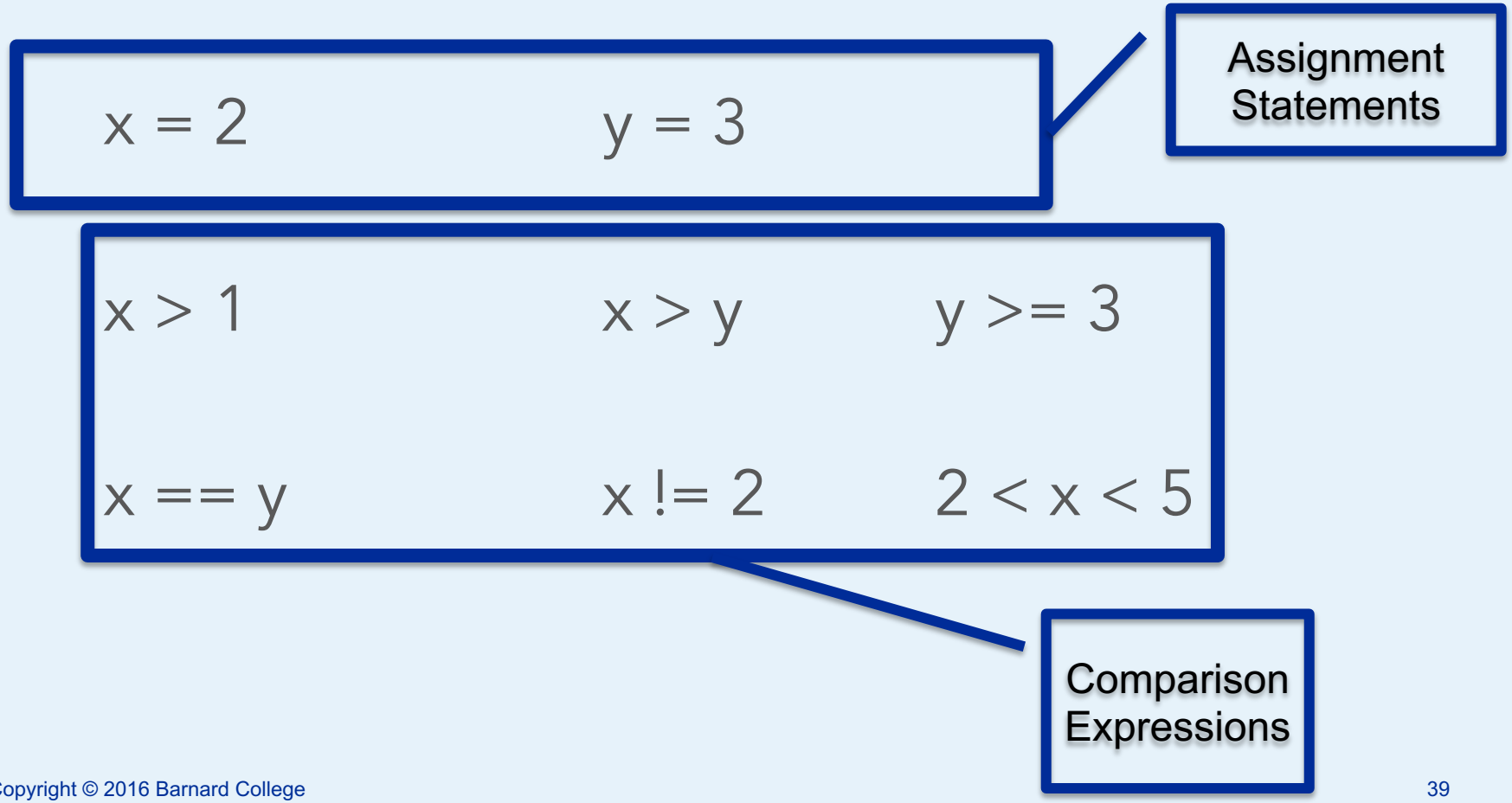
Assignment Statements

```
x > 1          x > y          y >= 3

x == y         x != 2         2 < x < 5
```

Comparison Operators

The result of a comparison expression is a bool value



Combining Comparisons

The result of a comparison expression is a bool value

a = True

b = False

not b

a or b

a and not b

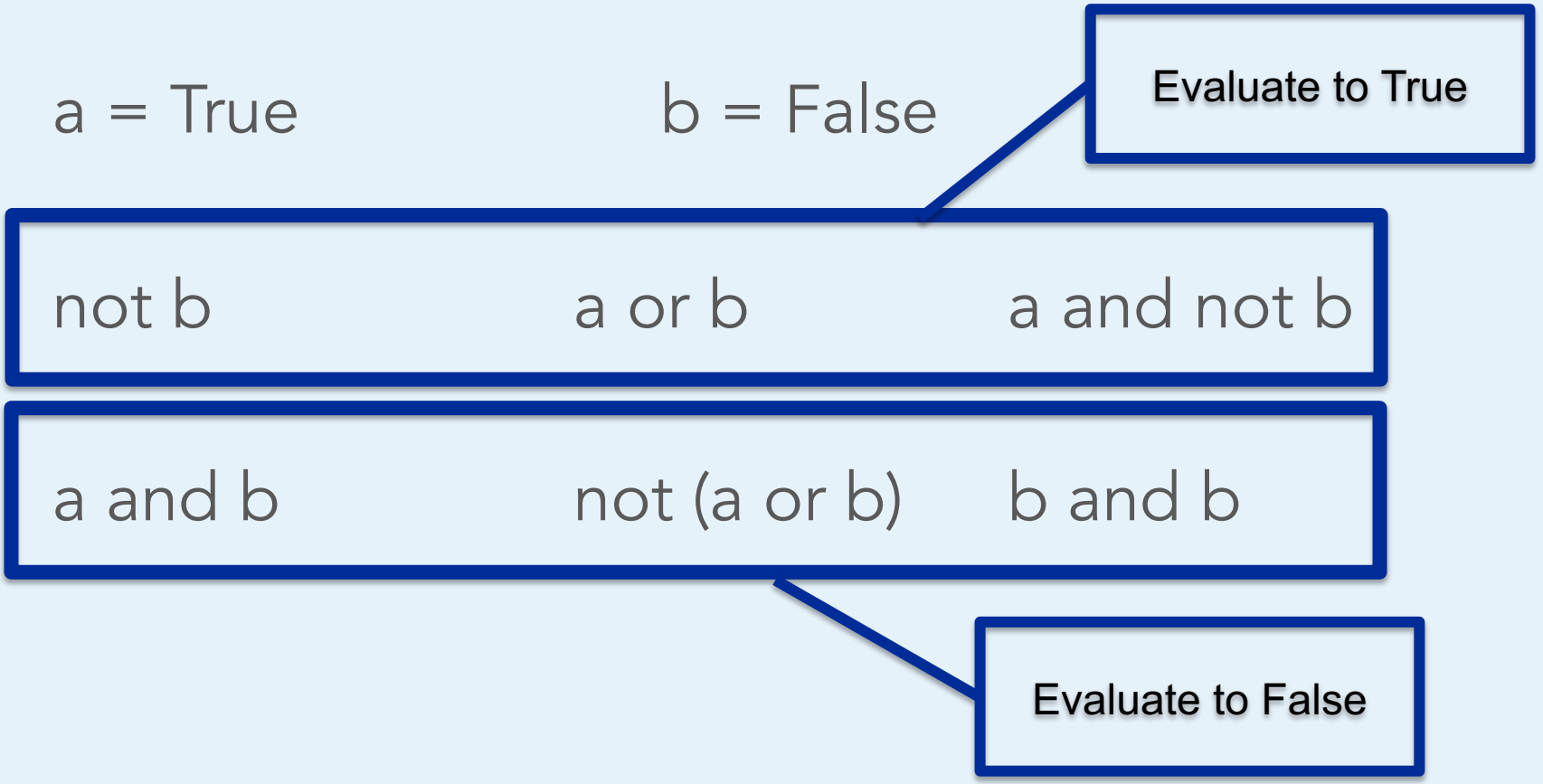
a and b

not (a or b)

b and b

Combining Comparisons

The result of a comparison expression is a bool value



Aggregating Comparisons

Summing an array or list of bool values count the number of True values

```
1 + 0 + 1                    == 2
True + False + True          == 2
sum([1    ,  0    ,  1])    == 2
sum([True, False, True)]     == 2
```

Control Statements

Control Statements

These statements *control* the sequence of computations that are performed

- The keywords if and for begin control statements
- The purpose of if is to define functions that choose different behavior based on their arguments

Random Selection

Random Selection

np.random.choice

- Selects at random
- With replacement
- From an array
- A specific number of times

np.random.choice(some_array, sample_size)

Appending Arrays

A Longer Array

- np.append(array_1, value):
  - new array with value appended to array_1
  - value has to be of the same type as elements of array_1

- np.append(array_1, array_2):
  - new array with array_2 appended to array_1
  - Elements of array_2 have to be of the same type as elements of array_1

Iteration

for **statements**

- for is a keyword that begins a control statement
- The purpose of for is to perform a computation for every element in a list or array