General Assembly

STUDY GUIDE

SELECTING AND JOINING

Key Terms and Definitions

» **Index:** A DataFrame always has an index. This can be an index of the automatically assigned 1, 2, 3, ... index numbers or an index you assign, such as a datetime column from your DataFrame. An index is the fastest method of identifying a row. Index functions use a dictionary to identify the rows and columns.

» **Useful DataFrame Methods: .dropna()**: Removes null values.

» **Useful DataFrame Methods:** Indexing/Selecting:

- **.iloc**: Works on positions in the index (and therefore only takes integers).
- **.loc**: Works on labels in the index.
- **.ix**: Will usually attempt to behave like **.loc** but reverts to behaving like **.iloc** if it can't find the specified label in the index. This is a more generalized method, but it increases the challenge of writing the Python statement correctly.

» **Combining DataFrames**:

- **.append()**: Works well for adding rows from one **DataFrame** to another, but it only works with rows.
- **.concat()**: Is a more flexible function for combining Pandas objects.
- **.join()**: Is used to perform index JOINs or single-column JOINs.
- **.merge()**: Is more complicated to use than **.join()** but is also more flexible.

» **inplace=True:** Allows you to permanently apply changes to a DataFrame instead of just returning a view of the modified DataFrame to look at temporarily.

Guiding Questions

1. What are the differences and similarities between the **.append()**, **.concat()**, **.merge()**, and **.join()** methods in Pandas?
2. Why is the index important to pay attention to when combining DataFrames?

Additional Resources

1. DataCamp

   » Intermediate Python for Data Science. Check out the "Filtering Pandas DataFrame" content in Section 3, "Logic: Control Flow and Filtering."

   » Merging DataFrames With Pandas. See Section 3, "Merging Data."

2. GA Demo Video: Joining and Merging DataFrames