SQL + Pandas Intro

Raguvir Kunani

Data 100

February 7, 2020

These slides are posted at tinyurl.com/raguvirData100

SQL Overview

What is SQL?

SQL is a language used to interact with relational databases (tables). It is a declarative language, meaning you tell the computer *what* you want as output and not *how* to compute the output.

Why do we use SQL?

The way `pandas` stores tables is not efficient for large amounts of data. SQL allows us to get information from databases that handle large amounts of data well.

SQL Syntax

The main element of SQL is the `SELECT` statement.

```
SELECT [DISTINCT] <column expression list>
FROM <relation>
[WHERE <predicate>]
[GROUP BY <column list>]
[HAVING <predicate>]
[ORDER BY <column list>]
[LIMIT <number>]
```

Note: The parts in square brackets [ ] are optional.

I write SQL queries in this order!

SQL Order of Evaluation

Unfortunately, the SQL syntax does not match the order of evaluation. Here is the actual order of evaluation:

1. `FROM` : which table(s) are we considering?
2. `WHERE` : selects rows based on a predicate
3. `GROUP BY` : forms groups
4. `HAVING` : filters groups (*only use with GROUP BY*)
5. `SELECT` : chooses which column(s) we want in the output
6. `ORDER BY` : orders the output
7. `LIMIT` : controls how many rows to output

SQL Example

Professor	Course	Term
Gonzalez	Data 100	Spring
DeNero	CS61A	Fall
Hug	CS61B	Spring
Hug	Data 100	Fall
Garcia	CS61A	Spring
Adhikari	Data 8	Spring
Hilfinger	CS61B	Fall
Wagner	Data 8	Fall

Joins

Often times, we want to combine information from multiple tables. To do this, we must use a **join**.

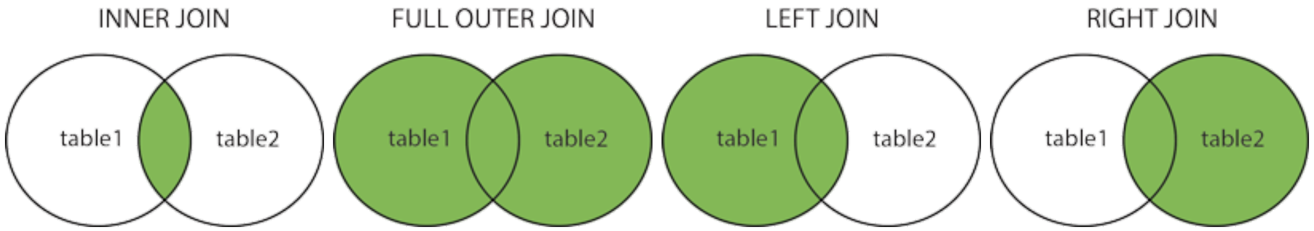
Any time you join tables, you need to make sure that the tables are somehow related. Otherwise, you have no way of correlating the information contained in the tables.

To specify how two tables are related, we use a **join condition** (looks something like `students.course_id = courses.course_id`)

See end of note for a joins example

Types of Joins

There are 4 types of joins:



In practice, you'll mostly use the inner join and the left join, but you should know all of these joins.

Joins in SQL

type of join

join condition

Doing a join in SQL follows almost directly from the names of the joins:

- `FROM table1 INNER JOIN table2 ON table1.a = table1.b`
- `FROM table1 OUTER JOIN table2 ON table1.a = table1.b`
- `FROM table1 LEFT JOIN table2 ON table1.a = table1.b`
- `FROM table1 RIGHT JOIN table2 ON table1.a = table1.b`

Common pitfall: Forming a Cartesian product.

```
FROM table1, table2 WHERE table1.a = table2.b
```

Cartesian product = all possible combinations of rows from table1 and table2

Final Thoughts on SQL

- SQL is really, really useful (especially for recruiting). It's worth learning well.
- If you're having trouble with SQL syntax, it might be helpful to say what you want in English and then convert to SQL
- The textbook was really helpful for me when I was learning SQL for this class

Pandas Overview

What is pandas?

pandas a Python library that optimizes operations on tables. It's different from SQL because it is not as declarative.

Why is pandas useful?

pandas makes it a lot easier to do ~~ machine learning ~~ .

Pandas Basics

pandas represents a tables with a **DataFrame**. The DataFrame is an object, meaning it has methods you can use. All of pandas is using these methods to get the output you want.

A column within a DataFrame is called a Series, and a Series also has methods.

Common methods you will use:

- `df.loc[]` and `df.iloc[]`
- `series.loc[]` and `series.iloc[]`
- `df.groupby()`
- `series.value_counts()`

`.loc[]` vs. `.iloc[]`

`.loc[]` and `.iloc[]` are both methods used to select rows (and columns) from a DataFrame

Every DataFrame and Series has an **index**, which you can think of as like a "row ID".

`.loc[]` uses this index to choose which rows to select, whereas `.iloc[]` just uses the *position* to select rows.

Example:

$\texttt{baby\_names.loc[}\underbrace{\texttt{1:4}}_{\text{rows}}\texttt{, }\underbrace{\texttt{:}}_{\text{cols}}\texttt{]}$ vs. `baby_names.iloc[1:4, :]`

Boolean Indexing

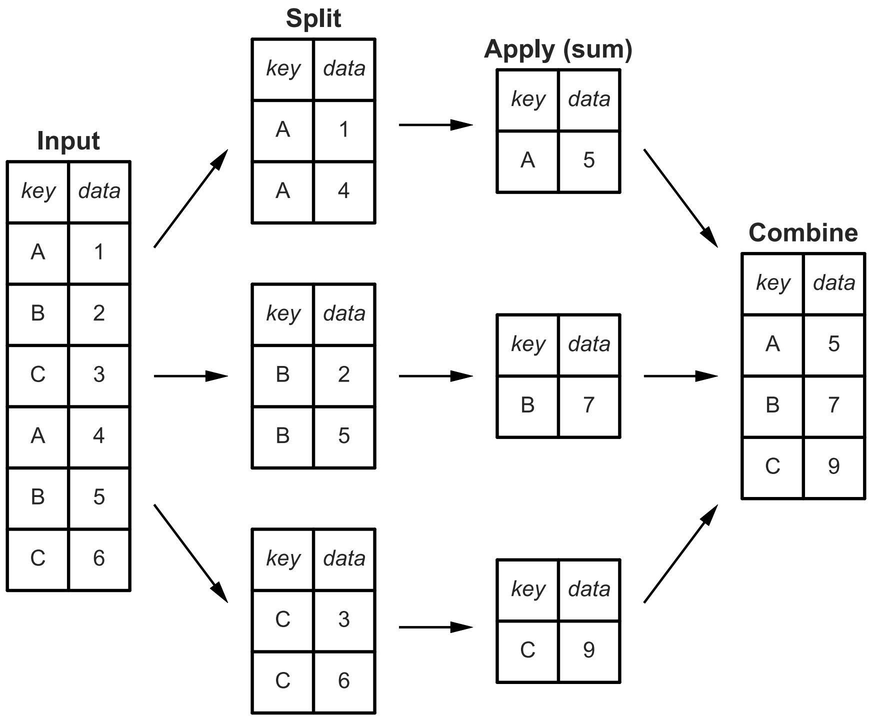
You can use `.loc[]` to filter the rows of a DataFrame based on some condition.

Example: `baby_names.loc[baby_names['State'] == 'CA']`

What does `baby_names['State'] == 'CA'` output?

`groupby()`

The purpose of `groupby()` is to change what each row in a DataFrame represents. Here's a visual of how the process works:



Final Thoughts on pandas

- pandas is one of things that is difficult for someone to explain to you, you just have to keep practicing to learn it
- Always keep track of what the input and output of methods are (e.g. `df.loc[]` takes in a DataFrame and outputs a DataFrame).
- Do things step by step. There are certain pandas questions on homeworks/projects in this course whose solutions are difficult to come up with all at once. Instead of trying to tackle the whole solution at once, try to write code that will get you part-way to the solution and then go from there

Feedback

- If you have any feedback for me (about my teaching, slides, ot anything else), fill out my feedback form!
- If you have any questions, feel free to ask me in person or by email (rkunani@berkeley.edu).

students

name	course_id
A	1
B	2
C	3
D	6
E	7

courses

course_id	course_name
1	Data 100
2	Prob 140
3	CS61A
4	Math 54
5	CS189

Inner Join

A	Data 100
B	Prob 140
C	CS61A

Outer Join

A	Data 100
B	Prob 140
C	CS61A
D	NULL
E	NULL
NULL	Math 54
NULL	CS189

Left Join

A	Data 100
B	Prob 140
C	CS61A
D	NULL
E	NULL

Right Join

A	Data 100
B	Prob 140
C	CS61A
NULL	Math 54
NULL	CS189