Analytics Jumpstart

Joining Dataframes

Nashville Software School

For today

- More pandas
  - Merging vs Concatenating
  - Aggregating

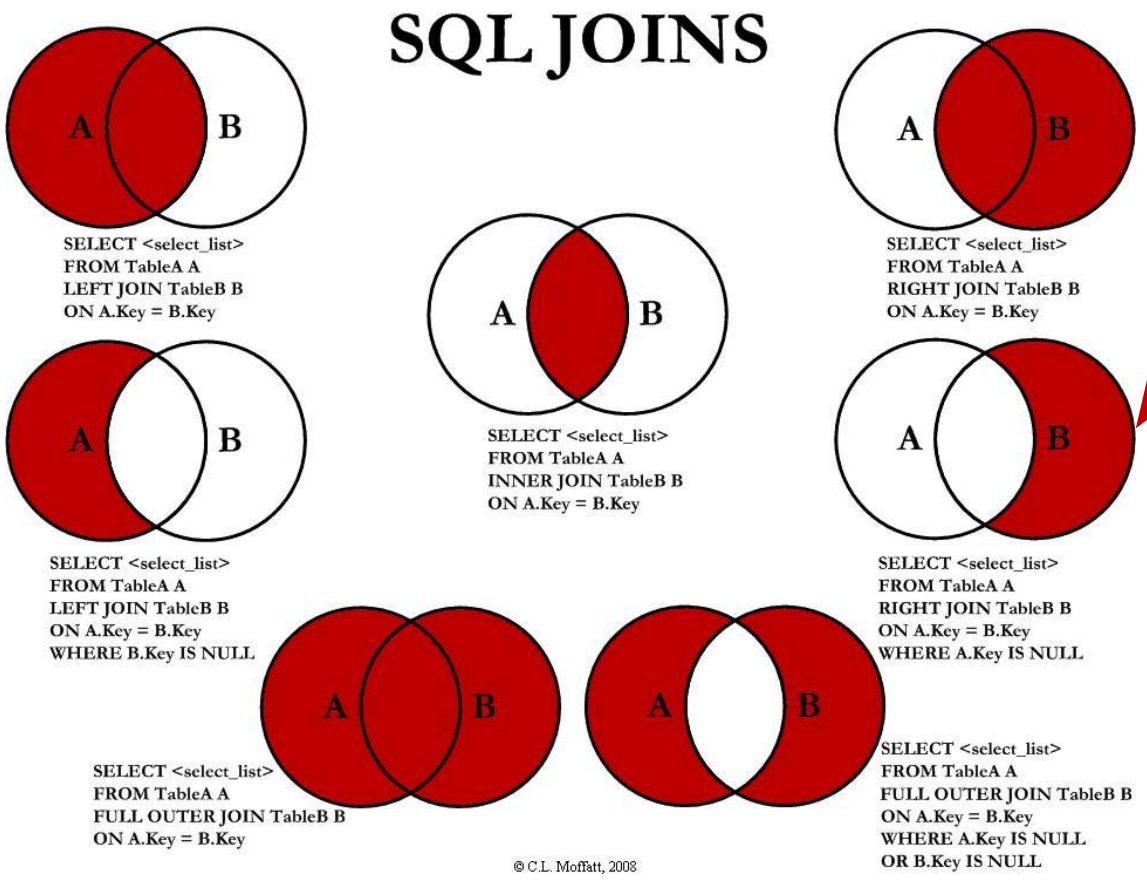
- **df.groupby(col) –** groups the DataFrame by the specified column
- **df.groupby(col).size()** – groups by a column and gets the size of each group
- **df.groupby([col_a, col_b]).agg(func) –** groups the DataFrame by col_a and col_b, then performs the specified aggregation function on each group
- **df.reset_index()** – useful for resetting the index after aggregation (moves the aggregation column from the row index to a column and uses zero-based row indexing)
- **df_1.append(df_2)** – stacks two DataFrames on top of each other. Does not pay attention to column numbers or names
- **pd.concat([df_1, df_2])** – combines list of DataFrames vertically or horizontally. Tries to align along concatenation axis
- **pd..merge(df_1, df_2, on, how)** – horizontally combine two DataFrames using column contents and following defined merging approaches.

NASHVILLE SOFTWARE SCHOOL

Get Data → Process + Clean Data → Exploratory Data Analysis

Merging two DataFrames:

pd.merge(*<df1>, <df2>, on = <col or list of cols to join on>, how = <join_type>*)



Same concept as a SQL join

NASHVILLE SOFTWARE SCHOOL

DataFrame A

```
df_food.head()
```

	produce_id	produce_name	produce_weight
0	00001	apples	0.5
1	00002	bananas	0.2
2	00003	carrots	0.3

DataFrame B

```
df_sales.head()
```

	produce_id	units_sold
0	00001	12.0
1	00003	NaN
2	00004	2.0

Left Join - Keep A, Match B

```
pd.merge(df_food, df_sales, how = 'left', on = 'produce_id')
```

	produce_id	produce_name	produce_weight	units_sold
0	00001	apples	0.5	12.0
1	00002	bananas	0.2	NaN
2	00003	carrots	0.3	NaN

Right Join - Keep B, Match A

```
pd.merge(df_food, df_sales, how = 'right', on = 'produce_id')
```

	produce_id	produce_name	produce_weight	units_sold
0	00001	apples	0.5	12.0
1	00003	carrots	0.3	NaN
2	00004	NaN	NaN	2.0

Inner Join - Keep Matches

```
pd.merge(df_food, df_sales, how = 'inner', on = 'produce_id')
```

	produce_id	produce_name	produce_weight	units_sold
0	00001	apples	0.5	12.0
1	00003	carrots	0.3	NaN

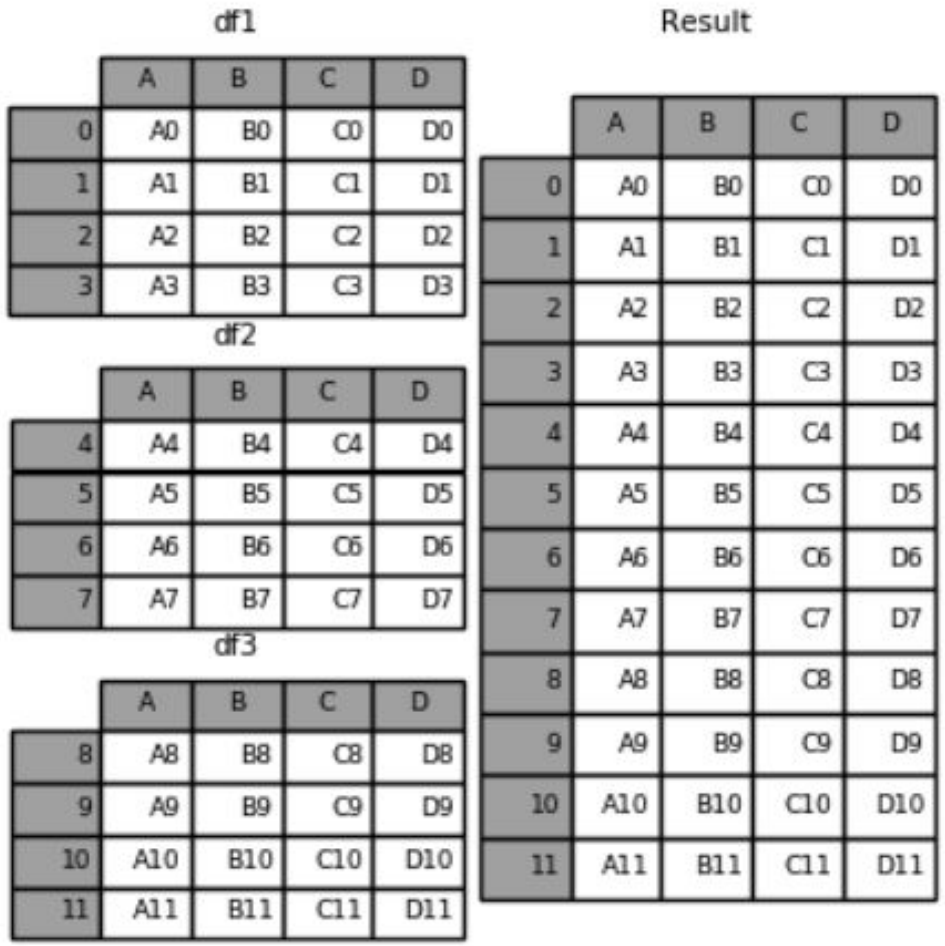
Outer Join - Keep Everything

```
pd.merge(df_food, df_sales, how = 'outer', on = 'produce_id')
```

	produce_id	produce_name	produce_weight	units_sold
0	00001	apples	0.5	12.0
1	00002	bananas	0.2	NaN
2	00003	carrots	0.3	NaN
3	00004	NaN	NaN	2.0

Concatenating two DataFrames:

pd.concat(*[<df1>, <df2>, <df3>]*)



df1

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3

df2

	A	B	C	D
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7

df3

	A	B	C	D
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

Result

	A	B	C	D
0	A0	B0	C0	D0
1	A1	B1	C1	D1
2	A2	B2	C2	D2
3	A3	B3	C3	D3
4	A4	B4	C4	D4
5	A5	B5	C5	D5
6	A6	B6	C6	D6
7	A7	B7	C7	D7
8	A8	B8	C8	D8
9	A9	B9	C9	D9
10	A10	B10	C10	D10
11	A11	B11	C11	D11

- **Same columns**
- **Like pasting them together**

NASHVILLE SOFTWARE SCHOOL

Questions?