

ЗАНЯТИЕ 13

ОБЪЕДИНЕНИЕ ДАТАФРЕЙМОВ

КОНСТАНТИН БАШЕВОЙ

Яндекс

Habr

@kpi_maker



Объединение таблиц это всегда весело

- Чем merge отличается от join и их типы в pandas
- Дубликаты как верный спутник объединений
- Оптимизация хранения данных с помощью join

PANDAS

И БОЛЬШИЕ ФАЙЛЫ



Количество уникальных ID

user_id

1

1

1

1

2

2

3

9

Данные отсортированы

Количество уникальных ID

user_id

1

1

1

1

2

2

3

9

- Читаем файл построчно
- Текущее VS прошлое значение
- Смена значения = пользователь

Количество уникальных ID

user_id

1

1

1

1

2

2

3

9

- Читаем файл построчно
- Текущее VS прошлое значение
- Смена значения = пользователь
- В памяти 3 числа
- Размер файла не имеет значения

СКВОЗНАЯ АНАЛИТИКА

Склеить лог визитов и лог покупок

	user_id	source
0	11	ad
1	22	yandex
2	55	email
3	11	google
4	77	ad

	user_id	category
0	11	Спорт
1	22	Авто
2	55	Дача
3	11	Дети
4	99	Авто

Склеить лог визитов и лог покупок

	user_id	source
0	11	ad
1	22	yandex
2	55	email
3	11	google
4	77	ad

	user_id	source	category
0	22	yandex	Авто
1	55	email	Дача

+ еще user_id = 11 и 77

	user_id	category
0	11	Спорт
1	22	Авто
2	55	Дача
3	11	Дети
4	99	Авто

—

ПЕРВЫЕ ПРОБЛЕМЫ

Нет однозначного соответствия

	user_id	source
0	11	ad
1	22	yandex
2	55	email
3	11	google
4	77	ad

	user_id	category
0	11	Спорт
1	22	Авто
2	55	Дача
3	11	Дети
4	99	Авто

Сумма визитов и покупок для user_id

```
visits_grouped = visits.groupby('user_id').count()  
visits_grouped.rename(columns={'source': 'visits'}, inplace=True)  
visits_grouped
```

visits	
user_id	
11	2
22	1
55	1
77	1

Сумма визитов и покупок для user_id

```
purchases_pivot = purchases.pivot_table(index='user_id', columns='category', values='user_id',  
                                         aggfunc='size', fill_value=0)  
purchases_pivot
```

category	Авто	Дача	Спорт
user_id			
11	0	0	2
22	1	0	0
55	0	1	0
99	1	0	0

Сумма визитов и покупок для user_id

```
purchases_pivot = purchases.pivot_table(index='user_id', columns='category', values='user_id',  
                                         aggfunc='size', fill_value=0)  
purchases_pivot
```

category	Авто	Дача	Спорт
user_id			
11	0	0	2
22	1	0	0
55	0	1	0
99	1	0	0

С count не работает

ТИПЫ ОБЪЕДИНЕНИЙ В PANDAS

Join – по индексу, merge – по столбцам

visits	
user_id	
11	2
22	1
55	1
77	1

category	Авто	Дача	Спорт
user_id			
11	0	0	2
22	1	0	0
55	0	1	0
99	1	0	0

user_id сейчас индекс – используем join

В таком варианте merge

```
visits.groupby('user_id').count().reset_index()
```

	user_id	source
0	11	2
1	22	1
2	55	1
3	77	1

user_id сейчас столбец

В таком варианте merge

```
purchases_pivot.reset_index()
```

category	user_id	Авто	Дача	Спорт
0	11	0	0	2
1	22	1	0	0
2	55	0	1	0
3	99	1	0	0

user_id сейчас столбец

Все параметры по умолчанию

```
visits_grouped.join(purchases_pivot)
```

	visits	Авто	Дача	Спорт
user_id				
11	2	0.0	0.0	2.0
22	1	1.0	0.0	0.0
55	1	0.0	1.0	0.0
77	1	NaN	NaN	NaN

—

LEFT И RIGHT JOIN

LEFT JOIN

Каждой строчке левой таблицы ищет соответствие
в правой

visits

user_id	
11	2
22	1
55	1
77	1

category Авто Дача Спорт

user_id				
11	0	0	2	
22	1	0	0	
55	0	1	0	
99	1	0	0	

LEFT JOIN

Каждой строчке левой таблицы ищет соответствие в правой

visits

user_id	
11	2
22	1
55	1
77	1

category Авто Дача Спорт

user_id			
11	0	0	2
22	1	0	0
55	0	1	0
99	1	0	0

правая таблица

порядок важен!

левая таблица

LEFT JOIN

Каждой строчке левой таблицы ищет соответствие
в правой

visits

category Авто Дача Спорт

user_id

user_id

11 2



11 0 0 2

22 1



22 1 0 0

55 1



55 0 1 0

77 1

99 1 0 0

LEFT JOIN

Строчки без пары левой таблицы остаются. Правой – удаляются из результата

visits			category			
user_id			Авто	Дача	Спорт	
11	2	→	11	0	0	2
22	1	→	22	1	0	0
55	1	→	55	0	1	0
77	1	↘	99	1	0	0
						NaN

все строчки левой
таблицы останутся

LEFT JOIN

Строчки без пары левой таблицы остаются. Правой – удаляются из результата

visits		category				
user_id		user_id	Авто	Дача	Спорт	
11	2	11	0	0	2	
22	1	22	1	0	0	
55	1	55	0	1	0	
77	1	✗ 99	1	0	0	

NaN

все строчки левой
таблицы останутся

правой – не факт

LEFT JOIN

Строчки без пары левой таблицы остаются. Правой – удаляются из результата

visits		category			
user_id		user_id	Авто	Дача	Спорт
11	2	11	0	0	2
22	1	22	1	0	0
55	1	55	0	1	0
77	1	99	1	0	0

NaN

```
visits_grouped.join(purchases_pivot,  
                    how='left')
```

visits		Авто	Дача	Спорт
user_id				
11	2	0.0	0.0	2.0
22	1	1.0	0.0	0.0
55	1	0.0	1.0	0.0
77	1	NaN	NaN	NaN

Right join зеркален left join

**Не рекомендуется к использованию без особой
на это необходимости**

Каждой строке правой таблицы ищет
соответствие в левой

visits		category Авто Дача Спорт			
user_id		user_id			
11	2	11	0	0	2
22	1	22	1	0	0
55	1	55	0	1	0
✗ 77	1	99	1	0	0

NaN

```
visits_grouped.join(purchases_pivot,  
                     how='right')
```

visits		Авто Дача Спорт			
user_id					
11	2.0	0	0	2	
22	1.0	1	0	0	
55	1.0	0	1	0	
99	NaN	1	0	0	

—

INNER ∩ OUTER JOIN

INNER JOIN

Оставляет строки, которые есть в обеих таблицах

visits		category Авто Дача Спорт				
user_id		user_id				
11	2	→	11	0	0	2
22	1	→	22	1	0	0
55	1	→	55	0	1	0
✗ 77	1		✗ 99	1	0	0

=

```
visits_grouped.join(purchases_pivot,  
                     how='inner')
```

visits		Авто	Дача	Спорт
user_id				
11	2	0	0	2
22	1	1	0	0
55	1	0	1	0

OUTER JOIN

Оставляет все строки

visits		category			
user_id		user_id	Авто	Дача	Спорт
11	2	11	0	0	2
22	1	22	1	0	0
55	1	55	0	1	0
77	1	99	1	0	0
NaN		NaN			

```
visits_grouped.join(purchases_pivot,  
                    how='outer')
```

	visits	Авто	Дача	Спорт
user_id				
11	2.0	0.0	0.0	2.0
22	1.0	1.0	0.0	0.0
55	1.0	0.0	1.0	0.0
77	1.0	NaN	NaN	NaN
99	NaN	1.0	0.0	0.0

САМОЕ ВЕСЕЛОЕ В JOIN

Что полезно проверять

(исходя из логики задачи)

- После LEFT-join количество строк не изменилось
- Суммы числовых столбцов не изменились
- Суммы в правой таблице тоже неплохо проверить

JOIN И ОПТИМИЗАЦИЯ ХРАНЕНИЯ

Как сэкономить место на диске

- Длинные повторяющиеся столбцы переводим в идентификаторы
- Составляем словари
- Логи - отдельно, словари - отдельно