Clase 027 — Pandas: concat, merge, join

Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 3 §§ 3.7–3.8 Combining Datasets: Concat/Merge. · Duración estimada: 90 min.

Clase 027 — Pandas: concat, merge, join

> *Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 3 §§ 3.7–3.8 Combining Datasets: Concat/Merge. · Duración estimada: 90 min.*

Objetivo

Que el alumno junte datasets correctamente: concat (apilado simple), merge (SQL-style joins) y join (atajo por index). El error más común es usar el join equivocado y obtener duplicados o filas perdidas — saber qué tipo (inner/left/right/outer) evita semanas de bugs.

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Apilar DataFrames con pd.concat por filas (axis=0) o columnas (axis=1).
2. Hacer joins SQL-style con pd.merge: inner, left, right, outer, cross.
3. Diagnosticar duplicados generados por merge con validate='one_to_one' | 'many_to_one' | ….
4. Joinear por index con df1.join(df2) (atajo para merge por index).
5. Usar indicator=True para saber qué filas vienen de cada lado del merge.

Temas

#	Tema	Por qué importa
1	concat axis=0 (filas) vs axis=1 (columnas)	Apilado simple con alineación de index.
2	merge how='inner'/'left'/'right'/'outer'	Los 4 tipos de join SQL.
3	on vs left_on/right_on	Cuando los nombres de columna difieren.
4	validate para evitar duplicación	1:1, 1:m, m:1, m:m.
5	indicator=True para auditar	Columna _merge con left_only/right_only/bo
6	df.join por index	Atajo idiomático.

Definiciones y características

concat

: Apila DataFrames por filas (axis=0, default) o columnas (axis=1). Alinea por el otro eje. No requiere key; es apilamiento puro.

merge (SQL-style join)

: Combina dos DataFrames por una key común. how='inner'/'left'/'right'/'outer'/'cross' controla qué filas se conservan.

INNER JOIN

: Solo filas presentes en AMBOS lados de la key. Si la key no matchea, se descarta. Default de merge.

LEFT JOIN

: Todas las filas del lado izquierdo (df1). Si no hay match en el derecho, columnas derechas quedan NaN. Útil

para enriquecer datos sin perder ninguno.

OUTER JOIN

: Todas las filas de ambos lados; NaN donde no hay match. Vista "unión". Útil para auditoría.

validate

: Parámetro de merge que valida la cardinalidad esperada: 'one_to_one', 'one_to_many', 'many_to_one', 'many_to_many'. Si los datos no la cumplen, lanza error → evita duplicados ocultos.

indicator=True

: Añade columna _merge con 'left_only'/'right_only'/'both'. Útil para auditar qué tipo de match tuvo cada fila.

Dataset / recursos

Sintético: tabla de clientes + tabla de órdenes (relación 1:N).

Ejercicios

1. Concat por filas. 3 DataFrames mensuales con mismas columnas → uno anual. ignore_index=True.

2. Inner join. Clientes + órdenes por cliente_id. Verifica que solo aparecen clientes con al menos 1 orden.

3. Left join. Clientes + órdenes, conservando clientes sin órdenes (NaN en cols de orden).

4. Detectar duplicados. Provoca un merge muchos-a-muchos no intencional. Usa validate='one_to_many' para que falle si hay duplicación oculta.

5. indicator=True. Auditar cuántas filas son left_only / right_only / both.

Homework verificable

Notebook con clientes (10) + órdenes (25): (a) 4 tipos de join con _merge indicator; (b) tabla con conteo de cada tipo; (c) detección de relación con validate; (d) join por index con df.join.

Criterio de aceptación: Counts de cada join coherentes (inner ≤ left ≤ outer). validate lanza excepción si la relación esperada falla.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Tras merge tengo más filas que el DataFram	Relación uno-a-muchos no esperada. Fix: va
merge produce KeyError en la key	Tipos distintos: int vs str aunque el valo
Columnas se renombran con _x/_y tras merge	Ambos DataFrames tenían cols con el mismo
pd.concat([df1, df2]) da columnas extra co	Los dos tenían columnas distintas (pandas
concat ignora mi ignore_index=True y queda	Si tus DFs tienen index distintos, sin ign

Preguntas frecuentes

¿merge o join?

merge es la API rica (por columnas, control total). df1.join(df2) es atajo cuando ambos tienen index alineado a la key. Mismo motor por dentro.

¿Cuándo concat vs merge?

concat: apilas datos con la misma estructura (mes 1, mes 2, mes 3 → año). merge: combinas datasets diferentes que comparten una key (clientes + órdenes).

¿validate siempre?

Sí — cuesta nada y atrapa el bug "silenciosamente generé el doble de filas". Recomendado en todo merge de producción.

¿Cómo merge por múltiples columnas?

merge(df1, df2, on=['a', 'b']) o left_on=['a','b'], right_on=['x','y']. La key compuesta es lista de strings.

¿Merge es lento con datasets grandes?

Con N=1M ya empieza a notarse. Acelera: setea index a la key antes (set_index('key').join(...)) o usa DuckDB (SELECT ... JOIN) — frecuentemente más rápido.

Referencias

- VanderPlas, cap. 3 §§ 3.7-3.8.
- pandas Merge user guide

Siguiente clase

Clase 028 — Pandas: groupby (split-apply-combine)

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
import numpy as np
import pandas as pd
```

Archivos complementarios

- notebook.ipynb