Your 2 GB CSV is a 200 MB Parquet.

D8A.ACADEMY/RESOURCES/PARQUET-CSV-JSON

THE THREE

csv	rows of text, no types
json	nested docs, keys repeat
parquet	columnar, typed, compressed

SIZE

csv 2.1 GB	parquet: ~200 MB
json biggest	keys repeat every row
why	← per-column compression bites.

SPEED

columnar reads	load 3 cols, skip 47
predicate pushdown	skips row groups
csv scan	parse every byte, always

TYPES

csv "00123"	zip code or number? guess
parquet schema	types in the file
dates in csv	← strings until parsed.

WHEN EACH WINS

csv	humans, Excel, exchange
json	APIs, events, real nesting
parquet	lakes, analytics

GOTCHAS

delimiter roulette	commas in quotes
not appendable	write new files
not readable	← keep a csv sample.

WORTH MEMORIZING

01 The conversion is one line

Same dataframe, three files: only one of them remembers the types.

```
df = pd.read_csv("sales_2026.csv")   # 2.1 GB

df.to_parquet("sales_2026.parquet")  # 205 MB

# and reads pick their columns
pd.read_parquet("sales_2026.parquet",
                columns=["date", "amount"])
```

→ The read_parquet call touches two column chunks and ignores the rest of the file. The equivalent read_csv parses all 50 columns to give you 2.

02 Trap: the CSV round-trip

What a CSV quietly does to your types on the way through.

COLUMN IN	AFTER CSV ROUND-TRIP	IN PARQUET
zip "00123"	123, an int now	"00123", string
2026-09-25	a string, parse again	date32, stays a date
NULL vs ""	both become empty	null survives

→ Interview phrasing: CSV is a serialization of strings, not of data. Every consumer re-guesses the schema, and each guesses differently.