Data Cleaning and Validation Processes in the Scraping Pipeline

1. khan_scraper.py: Data Extraction and Normalization

- Extracts content from Khan Academy pages using Selenium and BeautifulSoup.
- Normalizes URLs (e.g., converting embed or shortened YouTube links to canonical watch URLs).
- Captures images with computed dimensions (width × height) ensuring a minimum size of 1.
- Filters links using ignore-lists.
- Deduplicates extracted images, videos, and documents before writing JSON.
- Truncates text fields (titles, alt text, descriptions) to fit database constraints.
- Ensures `ResourceID` uniqueness across runs by tracking all previously used IDs.

2. data_cleaner.py: Deep Cleaning and Canonicalization

- Enforces presence of canonical tables (Resource, Note, pdf, Image, Video, Website).
- Deduplicates images by URL, selecting the largest version based on computed size.
- Generates missing Resource entries for images and ensures corresponding Note rows for alt text.
- Normalizes YouTube URLs and Website links.
- Truncates fields such as `Topic` and Note bodies to database-safe lengths.
- Ensures all sizes are positive integers and attempts to reuse existing ResourceIDs when requested.
- Guarantees structural consistency prior to validation.

3. validate.py: Schema-Level Integrity Checking

- Verifies that all top-level sections exist.
- Confirms all `ResourceID` values are integers and unique.
- Ensures ISO-format dates for `Date` and `DateFor`.
- Validates all columns for type, size, and domain constraints:
  - Author ≤ 50 chars.
  - Topic ≤ 25 chars.
  - Keywords ≤ 25 chars.
  - Rating numeric and between 0.0–9.9.
  - Format one of: Note, Video, Website, Pdf, Image.
- Checks foreign-key consistency across tables (e.g., each Note must reference an existing Resource).
- Validates URLs with regex and ensures image/video durations, sizes, and bodies meet database rules.

Summary

Together, these components form a robust scraping pipeline: the scraper gathers and normalizes raw data, the cleaner restructures and canonicalizes that data for database compatibility, and the validator enforces strict schema-level correctness to guarantee that only clean, consistent, and compliant data enters the final system.