perplexity

Ferramentas Gratuitas em Python e Soluções com IA para Limpeza e Otimização de Conteúdo para Bancos de Dados

Recomendação Principal:
Utilizar uma combinação de bibliotecas especializadas conforme o tipo de dado, apoiadas por frameworks de alto nível e agentes LLM (p. ex. CleanAgent) para automatizar roteiros de limpeza e padronização de colunas com mínima codificação.

1. Dados Tabulares

Para dados estruturados em linhas e colunas (CSV, Excel, SQL):

Biblioteca	Funcionalidade Principal	Licença
pandas	Leitura, limpeza de valores faltantes, normalização, junções, filtros, tipo `DataFrame`[1].	BSD
pyjanitor	API fluente sobre pandas para operações comuns (remover duplicatas, renomear colunas, preencher nulos)[2].	BSD
Dask	Processamento paralelo de DataFrames muito grandes, compatível com pandas.	BSD
Dataprep.Clean[1]	Padronização de colunas com chamadas declarativas (datas, valores monetários, texto)[3].	Apache 2
CleanAgent[1]	Agente LLM que integra Dataprep.Clean para automação “hands-free” da padronização de colunas[3].	Apache 2

[1] Componentes do projeto *CleanAgent* que abstraem lógica de padronização em uma só linha de código[3].

2. Textos e Dados Semiestruturados

Para extração e normalização de texto livre (logs, campos livres, HTML):

Biblioteca	Funcionalidade Principal	Licença
spaCy	Tokenização, lematização, reconhecimento de entidades (NER)	MIT
NLTK	Pré-processamento clássico (stopwords, stemming, tokenização)	Apache 2
RapidFuzz / FuzzyWuzzy	Similaridade de strings e correção aproximada de texto (fuzzy matching)[4].	MIT
OpenRefine	Ferramenta GUI para limpeza interativa; exportável em JSON/Python para reprodutibilidade[5].	BSD

Biblioteca	Funcionalidade Principal	Licença
badgers	Geração de déficits de qualidade (outliers, drift) para teste de pipelines textuais [6].	Apache 2

3. PDFs e Documentos Digitalizados

Para extração de texto e tabelas de PDFs:

Ferramenta	Funcionalidade Principal	Licença
pdfplumber	Extração de texto e tabelas, detecção de layouts complexos	MIT
PyPDF2 / pypdf	Leitura e manipulação de páginas, metadados	BSD
Camelot / tabula-py	Extração de tabelas de PDFs como DataFrames pandas	MIT
Tesseract + pytesseract	OCR para conversão de PDFs escaneados em texto bruto	Apache 2

4. Web Scraping

Para raspagem e limpeza de conteúdo HTML/JSON:

Biblioteca	Funcionalidade Principal	Licença
BeautifulSoup	Parser de HTML/XML; navegação na árvore DOM, limpeza de tags e extração de texto	MIT
Scrapy	Framework completo de scraping assíncrono, pipelines de limpeza e exportação de dados	BSD
requests	Requisições HTTP simples para obtenção de páginas e APIs	Apache 2

5. Imagens, Áudio e Outros Dados Não-Tabulares

Para pré-processamento de imagens e arquivos multimídia:

Biblioteca	Funcionalidade Principal	Licença
OpenCV	Remoção de ruído, redimensionamento, equalização, segmentação	BSD
scikit-image	Filtros, transformação geométrica, detecção de bordas	BSD
Pillow	Leitura, corte e conversão de formatos básicos	PIL
librosa	Análise de áudio, extração de features, limpeza de ruído	ISC
pytube / youtube-dl	Download e conversão de streams de vídeo para extração posterior	MIT

6. Otimização e Automação com IA

Frameworks que combinam LLMs e bibliotecas de limpeza para gerar pipelines automaticamente:

- **CleanAgent**: Integra Dataprep.Clean e agentes LLM para gerar código de padronização de colunas a partir de requisitos em linguagem natural, eliminando grande parte da codificação manual[3].
- **RetClean**: Propõe uso de ChatGPT para sugerir correções em células de tabelas, com fallback em repositórios locais de conhecimento para dados não públicos[7].
- **AlphaClean**: Busca pipeline óptimo de limpeza por hiper-parametrização estruturada; foca em tabular, mas metodologia extensível a outros tipos[8].

7. Fluxo de Trabalho RecomENDADO

1. **Ingestão**
   - Scrapy/requests para web scraping ou `pandas.read_*` para fontes estruturadas.
2. **Limpeza Inicial**
   - pandas + pyjanitor para remoção de nulos, formatação de tipos simples.
3. **Padronização Avançada**
   - Dataprep.Clean (CleanAgent) para normalizar datas, moedas, identificadores em uma chamada.
4. **Dados Não-Tabulares**
   - PDF: pdfplumber → DataFrame → pandas
   - Imagem/Áudio: OpenCV/librosa → features → pandas
   - Texto livre: spaCy/NLTK → tokens/entidades → vetorização
5. **Validação**
   - badgers para gerar casos de teste (outliers, drift) e garantir robustez[6].
6. **Carga no Banco**
   - SQLAlchemy ou `pandas.to_sql`; ajuste de índices e estilos dependendo do SGBD.
7. **Automação Contínua**
   - Orquestração com Airflow ou Prefect, integrando scripts e agentes LLM para atualização periódica.

Esse ecossistema modular permite tratar **todos os tipos de dados** (tabular, texto, PDF, web, multimídia) de forma integrada, gratuita e automatizável, alcançando limpeza e otimização adequada antes da carga em bancos de dados.

⁂

1. https://www.ijeat.org/wp-content/uploads/papers/v9i4/D9057049420.pdf
2. http://conference.scipy.org/proceedings/scipy2019/pdfs/eric_ma.pdf

3. https://arxiv.org/abs/2403.08291
4. https://ijeedu.com/index.php/ijeedu/article/view/188
5. https://pmc.ncbi.nlm.nih.gov/articles/PMC3738091/
6. https://arxiv.org/pdf/2307.04468.pdf
7. https://arxiv.org/pdf/2303.16909.pdf
8. https://arxiv.org/pdf/1904.11827.pdf