Conexión a los datos con LangChain

José Orlando Maldonado Bautista

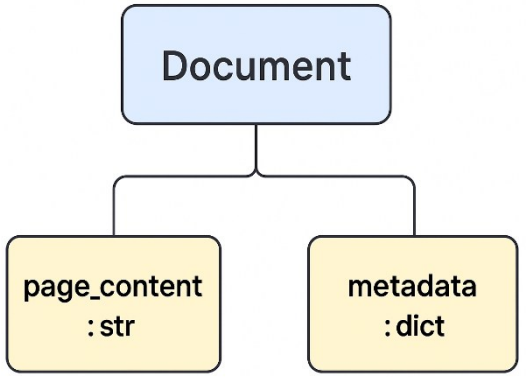
Cargadores de Documentos en LangChain

LangChain ofrece una amplia variedad de **Document Loaders** integrados para convertir diferentes fuentes en objetos estandarizados. Todos los loaders comparten un método común `loader.load()` para obtener los documentos y convertirlos en objetos tipo `Document`.

El objeto `Document` es una estructura fundamental en LangChain que representa una unidad de texto junto con sus metadatos asociados:

- ✓ **Contenido principal (page_content)**: Texto o información que se desea procesar o analizar.
- ✓ **Metadatos (metadata)**: Información adicional contextual que puede incluir el origen del documento, número de página, etiquetas, etc.

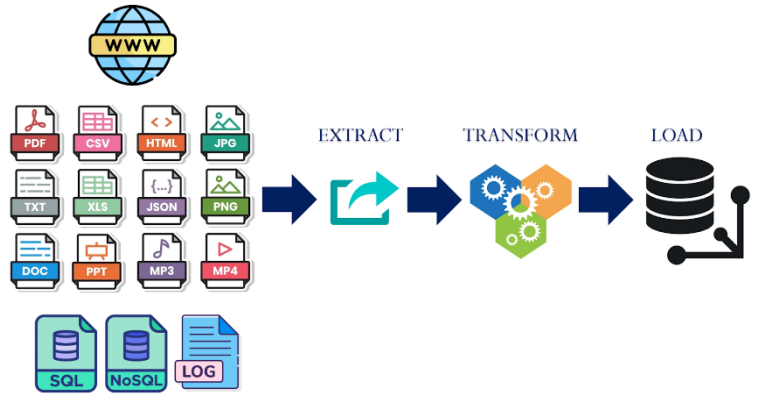
Este diseño facilita la manipulación y el seguimiento de fragmentos de texto en flujos de trabajo con modelos de lenguaje, permitiendo almacenar tanto el contenido como datos relevantes que mejoran la gestión y el análisis.



Cargadores de Documentos en LangChain

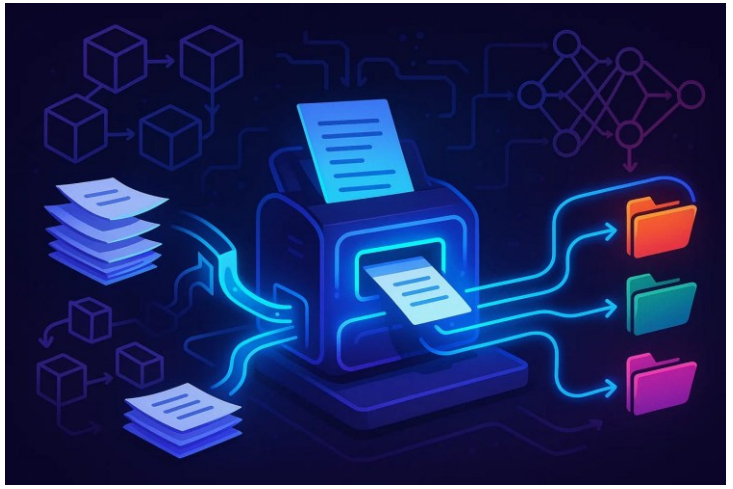
Existen *loaders* específicos según el tipo de fuente:

- **Web:** HTML mediante urllib y BeautifulSoup, (WebBaseLoader, BSHTMLLoader) o usando la biblioteca Unstructured.
- **PDF:** Varias opciones como PyPDFLoader, UnstructuredPDFLoader, PDFPlumber, PyMuPDF, PDFMiner, entre otros.
- **Archivos comunes:** (CSVLoader), JSON (JSONLoader), Markdown (UnstructuredMarkdownLoader), texto plano (TextLoader), entre otros.
- **Servicios en la nube:** Soporte para AWS S3, PyPDFLoader, UnstructuredPDFLoader, PDFPlumber, PyMuPDF, PDFMiner, entre otros.
- **Plataformas y redes sociales:** Incluye loaders para Reddit, Twitter, WhatsApp, Slack, GitHub, Trello, Notion, Figma, entre otros.



Transformadores de Documentos en LangChain

- Son utilidades para **preprocesar, dividir, filtrar o resumir documentos** (Document).
- Su objetivo es **optimizar el texto** antes de usarlo en embeddings, búsqueda semántica o RAG.
- **Principales funciones:**
  - ✓ **División**: cortar documentos largos en *chunks* manejables.
  - ✓ **Filtrado**: eliminar redundancias o información irrelevante.
  - ✓ **Compresión**: reducir volumen de texto manteniendo lo esencial.
  - ✓ **Resumen**: condensar múltiples documentos en versiones más cortas.
- Permiten **mayor eficiencia, relevancia y precisión** en sistemas de IA generativa.



Tipos de Document Transformers

Categoría	Clase / Ejemplo	Uso principal
Text Splitters	CharacterTextSplitter, RecursiveCharacterTextSplitter, TokenTextSplitter, MarkdownTextSplitter, HTMLHeaderTextSplitter, PythonCodeTextSplitter	Dividir documentos en chunks manejables (texto, código, HTML, LaTeX, etc.).
Metadata & Filters	EmbeddingsRedundantFilter, EmbeddingsClusteringFilter, EmbeddingsFilter, LLMChainFilter	Filtrar redundancias o seleccionar los documentos más relevantes.
Document Compressors	DocumentCompressorPipeline, LLMChainExtractor	Reducir el volumen de texto manteniendo la información clave.
Summarization Chains	StuffDocumentsChain, MapReduceDocumentsChain, RefineDocumentsChain	Resumir uno o varios documentos en diferentes niveles de detalle.
Custom Transformers	BaseDocumentTransformer	Crear transformadores propios (limpieza, normalización, extracción personalizada).

Formando nuevas generaciones con sello de excelencia comprometidos con la transformación social de las regiones y un país en paz