Entregable 2. Pipeline ETL

A continuación, se presenta una descripción detallada del *pipeline* ETL (Extract, Transform, Load) utilizado para obtener, procesar y cargar datos de recetas culinarias provenientes de **AllRecipes** y **Directo al Paladar**, hasta almacenarlos en una base de datos relacional. El flujo se divide en **tres** pasos principales:

1. Extracción

Para la **Extracción** de datos se han utilizado spiders de **Scrapy** que rastrean los sitios web de *allrecipes.com* y *directoalpaladar.com*, leyendo su **sitemap** y guardando la información en archivos `.jsonl`. A continuación, se describen los aspectos clave de cada spider:

1.1. Spider para AllRecipes

- **Tecnología**: Se emplea la clase `SitemapSpider` de Scrapy para recorrer el sitemap principal, que se indica en la variable `sitemap_urls`.
- **Filtrado de URLs**: Solo se procesan aquellos enlaces que contengan `/recipe/`.
- **Estructura de datos**:
  - URL de la receta
  - Título (obtenido a través de selectores XPath)
  - Ingredientes (nombres, cantidades, unidades), localizados mediante selectores CSS
  - Raciones
  - Instrucciones para preparar la receta

El spider escribe la información recopilada en un archivo `allrecipes.jsonl`, evitando duplicados gracias a un conjunto (*set*) que almacena las URLs ya procesadas.

1.2. Spider para Directo al Paladar

- **Tecnología**: También basada en `SitemapSpider`, apuntando a un sitemap que agrupa todas las recetas.
- **Filtrado de URLs**: Se desordena aleatoriamente la lista del sitemap y se descartan las URLs que ya figuran en el archivo de resultados (para evitar duplicados).

- **Estructura de datos**:
  - URL de la receta
  - Título
  - Ingredientes (con nombre, cantidad y unidad)
  - Raciones
  - Instrucciones

La información se almacena en el archivo `dap.jsonl`. Igual que en el spider anterior, se supervisa la lista de URLs procesadas a fin de no sobreescribir recetas ya existentes.

2. Transformación

Una vez que se tienen los datos de las recetas en archivos `.jsonl`, el siguiente paso es **transformarlos** para:

- **Traducir** ingredientes de español a inglés (o viceversa) usando `DeepTranslator`.
- **Obtener** información nutricional de cada ingrediente mediante la API de **Edamam**.
- **Normalizar** valores (por ejemplo, calcular nutrientes por cada 100 g).
- **Enriquecer** la receta con etiquetas de salud (*health labels*).

Uno de los principales problemas encontrados en este paso vendría de la **normalización de valores**, ya que las recetas son redactadas por muchas personas diferentes, cada una usando unidades de medidas diferentes y los nombres de los ingredientes podían variar aún refiriéndose al mismo elemento.

Esto se solucionó gracias al uso de la API de **Edamam**, que incluye un motor **NLP** capaz de detectar ingredientes y unidades de medida a partir de texto, devolviendo siempre un mismo nombre de ingrediente y la posibilidad de generalizar a 100 g.

Las funciones principales para estos propósitos se encuentran en un módulo de soporte (`support_etl.py`).

2.1. Traducción de Ingredientes

Para ajustarse a los requerimientos de la API de Edamam, se traducen los ingredientes al inglés usando la librería `DeepTranslator`. De esta manera, la API reconoce correctamente los ingredientes y devuelve datos nutricionales precisos.

2.2. Obtención de Datos Nutricionales

Se realiza una petición a la API de Edamam, que provee:

- Peso (g)
- Calorías (kcal)
- Proteínas (g)
- Grasas (g)
- Carbohidratos (g)
- Azúcares (g)
- Fibra (g)
- Etiquetas de Salud (por ejemplo, *vegan*, *gluten-free*, *low-carb*, etc.)

2.3. Normalización y Ensamblaje

Antes de cargar los datos en la base de datos, se filtran y normalizan los valores nutricionales para estandarizar las cantidades en 100 g de ingrediente. Al final de este paso, se cuenta con:

- Un listado final de ingredientes con sus nutrientes por 100 g.
- Un dictamen completo de la receta (nombre, URL, nutrientes totales por porción, raciones y etiquetas de salud).

3. Carga

La fase de **Carga** inserta toda la información previamente transformada en **Supabase**, que funciona como una base de datos PostgreSQL a la cual se conecta usando la librería `psycopg2`.

3.1. Inserción de Recetas

Se crea un registro en la tabla `recipes` con:

- Nombre de la receta
- URL
- Tamaño total de la receta en gramos
- Nutrientes totales ajustados por número de raciones
- Cantidad de raciones

3.2. Creación de Ingredientes y Relaciones

Para evitar duplicados, el sistema verifica si un ingrediente ya existe en la tabla `ingredients`. Si no, lo crea registrando sus nutrientes (por 100 g) y crea un id único para este usando `hashlib`. Luego, en la tabla `recipe_ingredients`, se guarda la relación entre cada receta y sus ingredientes, junto a la cantidad usada.

3.3. Etiquetas de Salud

Cada etiqueta de salud se almacena en la tabla `tags`. Si la etiqueta no existe, se inserta; de lo contrario, se reutiliza su registro. Posteriormente, la relación con la receta queda definida en la tabla intermedia `recipe_tags`, lo que permite clasificar las preparaciones según atributos como *vegan, vegetarian, gluten-free*, entre otros.

La estructura de la base de datos es la siguiente:

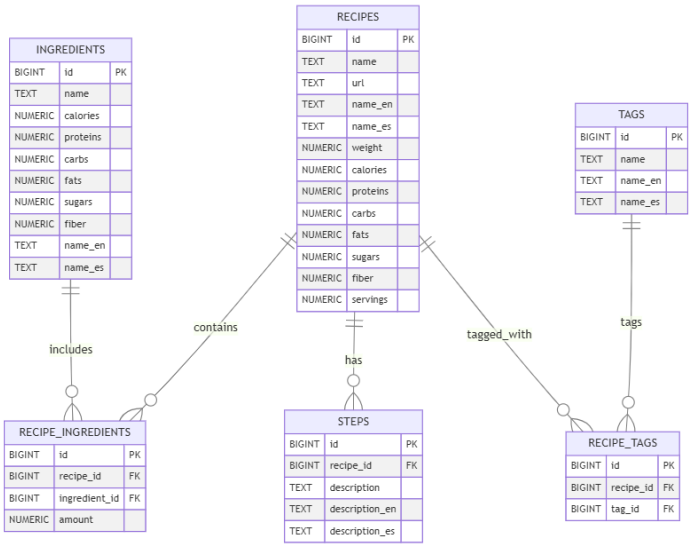

Figure 1: Diagrama de la base de datos.