**SOLUCIÓN TECNOLÓGICA PARA LA OPTIMIZACIÓN EN LA GESTIÓN DE RECURSOS HÍDRICOS EN EL PERÚ APLICANDO BUSINESS INTELLIGENCE**

**Documento de Modelado del Largue Language Model**

Versión 1.0

**Control de Versiones**

**Lima, 13 de abril del 2025**

**ÍNDICE**

[**1. Introducción 3**](#_heading=h.v3rb3a59kn4g)

[**2. Objetivo 3**](#_heading=h.g82k2sxuay3k)

[**3. Conceptualización General del LLM 4**](#_heading=h.g82k2sxuay3k)

[**4. Vectorización de la Data 4**](#_heading=h.m3b1eqaje8ix)

[**5. Almacenamiento de vectores semánticos 5**](#_heading=h.8r1swqd6on5i)

[**6. Construcción del RAG (Retrieval-Augmented Generation) 6**](#_heading=h.rcdj9nuuujvu)

[**7. Generación de SQL con LLM y Validación 7**](#_heading=h.bfvclorotb5d)

[**8. Seguridad y Buenas Prácticas 9**](#_heading=h.1t3n01d7tdhy)

# **Introducción**

Este documento establece el funcionamiento detallado de los procesos utilizados para el entrenamiento de los Largue Language Model, incluyendo el proceso de vectorizado de la información, almacenamiento de los vectores semánticos, construcción del RAG (Retrieval-Augmented Generation, entre otros)

# **Objetivo**

Describir de manera precisa y clara los pasos seguidos en el entrenamiento del programa, de manera que este pueda servir para futuros proyectos, así como una mejor comprensión del funcionamiento interno, facilitando así la implementación de mejoras o más funcionalidades.

# **Conceptualización General del LLM** Este proyecto implementa un sistema de preguntas y respuestas con dos funciones principales:

* Conversación tipo RAG (Retrieval-Augmented Generation): Integra recuperación semántica + LLM para responder a consultas generales sobre datos SQL.
* Generación y Ejecución de Consultas SQL: Convierte preguntas en lenguaje natural en SQL, con validación de esquema y ejecución en una base de datos SQL Server.

Los LLM utilizados son:

* OllamaLLM con el modelo deepseek-r1:8b
* Groq API con llama3-70b-8192 para limpieza/validación avanzada de SQL

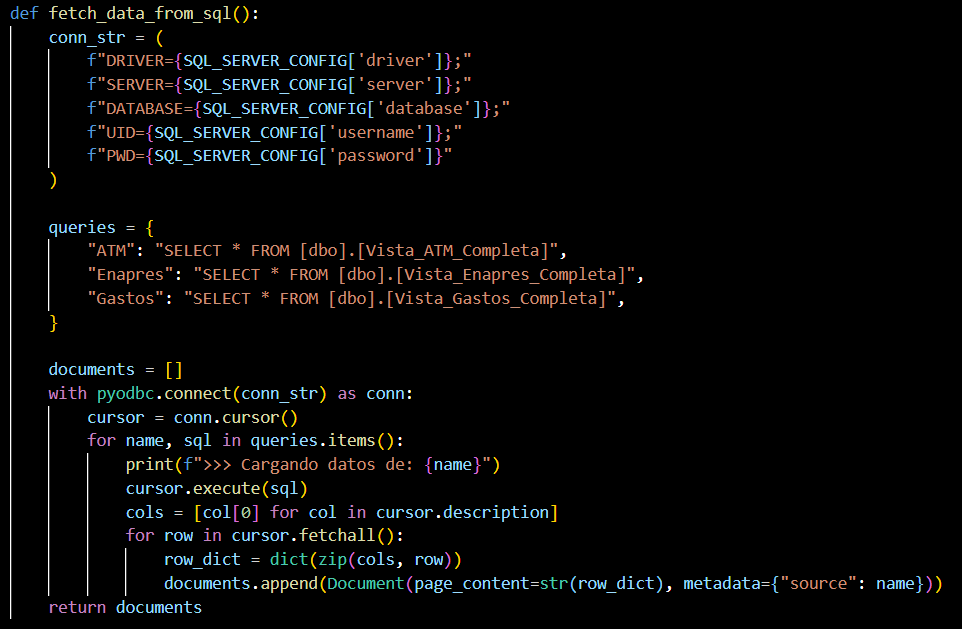
# **Vectorización de la Data**

El sistema convierte documentos obtenidos de vistas SQL en vectores semánticos usando el modelo all-MiniLM-L6-v2 de HuggingFace.  
Flujo:

1. Se extraen los datos de las vistas [Vista\_ATM\_Completa], [Vista\_Enapres\_Completa], [Vista\_Gastos\_Completa].
2. Cada fila es convertida a texto (str(row\_dict)) y envuelta como un objeto Document con metadatos.
3. Estos documentos son vectorizados mediante el modelo de embeddings.

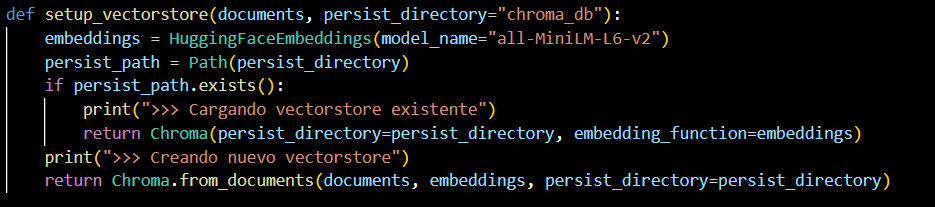
**Figura 1.**

*Código pertinente de etapas 1 y 2 del flujo señalado*

** Elaboración propia.

**Figura 2.**

*Código pertinente de la etapa 3 del flujo señalado*

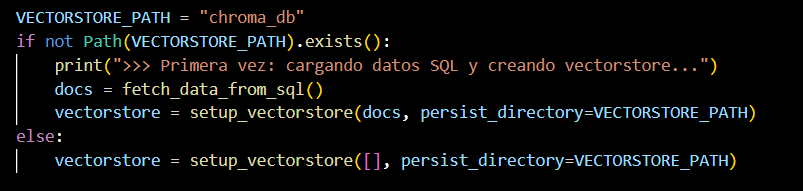
**

Elaboración propia.

# **Almacenamiento de vectores semánticos**

Se utiliza un almacenamiento persistente a través de Chroma como motor de vectorstore, con persistencia en disco (chroma\_db). Esto permite no tener que re-vectorizar los documentos en cada reinicio.  
**Figura 3.**

*Flujo de inicialización para el almacenamiento persistente usando ChromaDB*

**

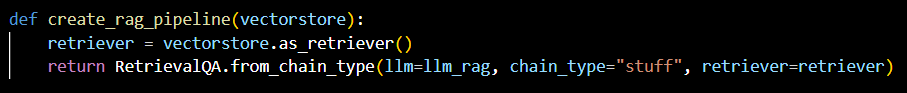
Elaboración propia.

# **Construcción del RAG (Retrieval-Augmented Generation)** El pipeline RAG se construye con:

* Retriever: usa Chroma para recuperar documentos similares al prompt.
* LLM: deepseek-r1:8b para generar respuestas con los documentos recuperados.

**Figura 4.**

*Código que define el uso del Retriever*

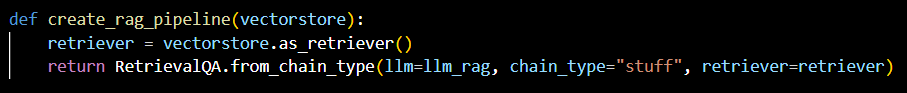
**

Elaboración propia.

A su vez se hace uso del Endpoint /ask para la consulta al RAG

**Figura 5.**

*Código que define el Endpoint a utilizar en base a la información vectorial consultada*

**

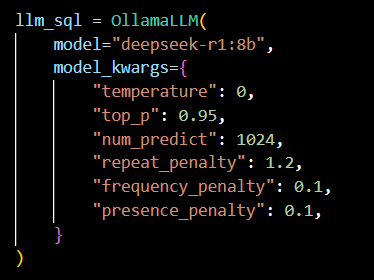
Elaboración propia.

# **Generación de SQL con LLM y Validación** El proceso consta de 4 etapas principales:

1. Conversión de lenguaje natural a query SQL mediante DeepSeek

**Figura 6.**

*SetUp del modelo de LLM a utilizar así como configuración de parámetros*

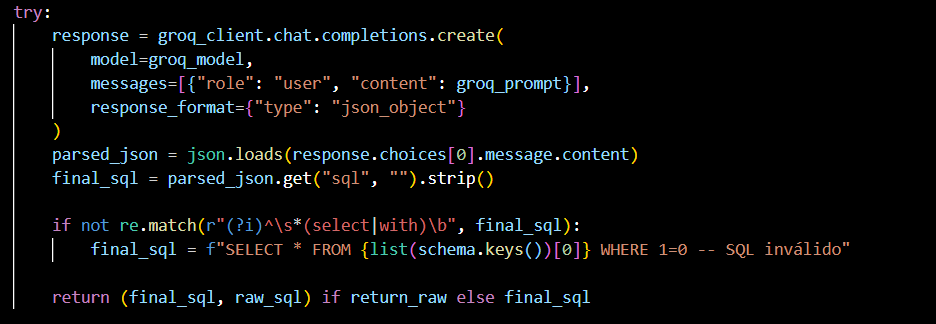
**

Elaboración propia.

1. Validación y limpieza de la respuesta generada mediante Groq llama3

**Figura 7.**

*Código referente a la limpieza de la respuesta provista por DeepSeek de modo que solo quede el query*

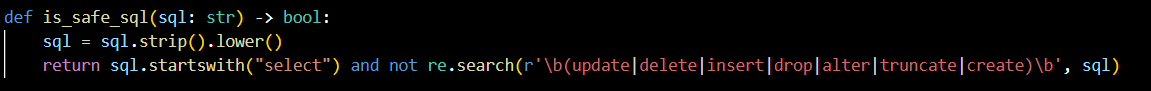
**

Elaboración propia.

1. Validaciones de seguridad que bloquean caracteres peligrosos y solo permiten consultas de tipo SELECT

**Figura 8.**

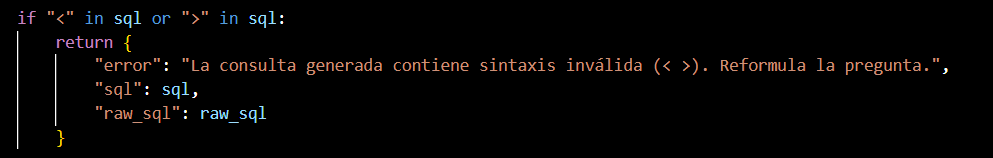
*Código que solo permite las consultas de tipo SELECT*

**

Elaboración propia.

**Figura 9.**

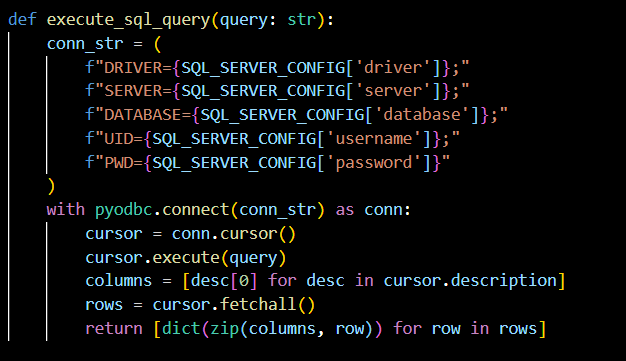
*Código que informa sobre sintaxis inválida en la query*

**

Elaboración propia.

1. Ejecución en SQL Server

**Figura 10.***Código que permite la ejecución del query previamente procesado en la base de datos SQL Server*

** Elaboración propia.

# **Seguridad y Buenas Prácticas**

Con el fin de llevar a cabo las buenas prácticas se llevaron a cabo las siguientes consideraciones:

* Se utiliza .env para variables de conexión a SQL Server.
* Tal como se mostró anteriormente se realiza una validación estricta de las consultas generadas para evitar inyecciones o acciones destructivas
* API estructurada con FastAPI y habilitación CORS para uso web seguro