RAG: Mejorando los LLMs con Conocimiento Externo

¿Qué es RAG?

Retrieval-Augmented Generation (RAG) es un patrón de arquitectura que combina un modelo de lenguaje pre-entrenado con un sistema de recuperación de información.

- **Objetivo**: Dar al LLM acceso a conocimiento externo y actualizado para que pueda generar respuestas más precisas y fundamentadas, reduciendo las "alucinaciones".
- **Componentes**: Un componente de **recuperación (Retrieval)** y un componente de **generación (Generation)**.

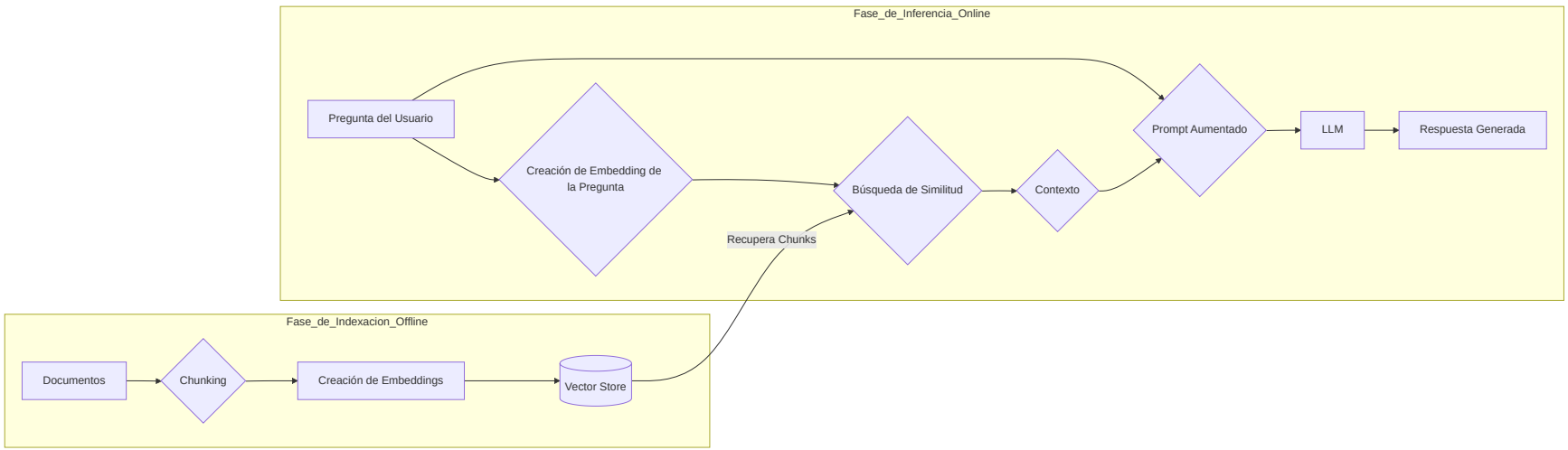
¿Por qué necesitamos RAG?

Los LLMs tienen limitaciones:

- **Conocimiento Estático**: Su conocimiento se limita a los datos con los que fueron entrenados y no se actualiza.
- **Alucinaciones**: Pueden inventar hechos o detalles que no son correctos.
- **Falta de Transparencia**: Es difícil saber de dónde proviene la información de sus respuestas.

RAG mitiga estos problemas.

Flujo de Funcionamiento de RAG



Pasos del Proceso RAG

1. **Indexación**: Se procesa una base de conocimiento (documentos, PDFs, etc.), se divide en trozos (chunks), se generan embeddings para cada chunk y se almacenan en una base de datos vectorial.

2. **Recuperación**: Cuando un usuario hace una pregunta, se genera un embedding de esa pregunta.

3. **Búsqueda**: Se utiliza el embedding de la pregunta para buscar los chunks más similares (semánticamente relevantes) en la base de datos vectorial.

4. **Aumento**: Los chunks recuperados se insertan en el prompt junto con la pregunta original.

5. **Generación**: El LLM recibe el prompt "aumentado" y genera una respuesta basada tanto en su conocimiento interno como en el contexto proporcionado.

Ventajas de RAG

- **Respuestas más Precisas y Actualizadas**: Al basarse en una fuente de conocimiento externa, las respuestas son más fiables.
- **Reducción de Alucinaciones**: El modelo se fundamenta en la información recuperada.
- **Transparencia y Trazabilidad**: Se puede citar las fuentes utilizadas para generar la respuesta.
- **Costo-Efectividad**: Es más barato y rápido actualizar una base de datos vectorial que re-entrenar un LLM completo.

Concepto Adicional: Reranking

En un sistema RAG, la fase de recuperación inicial (búsqueda de similitud en la base de datos vectorial) puede devolver documentos que son semánticamente similares a la consulta, pero no necesariamente los más relevantes o útiles para la generación de la respuesta.

El **Reranking** es un paso opcional pero muy efectivo que se inserta entre la recuperación y la generación. Consiste en reordenar los documentos recuperados utilizando un modelo más sofisticado que evalúa la relevancia de cada documento en el contexto de la consulta.

Beneficios del Reranking:

- **Mejora la Calidad de la Respuesta:** Al asegurar que el LLM reciba la información más pertinente.
- **Optimiza el Uso del Contexto:** Permite pasar solo los documentos de mayor calidad al LLM, especialmente útil con ventanas de contexto limitadas.
- **Reduce el Ruido:** Filtra documentos que, aunque semánticamente similares, podrían no ser directamente útiles.

¿Qué es el Fine-tuning de LLMs?

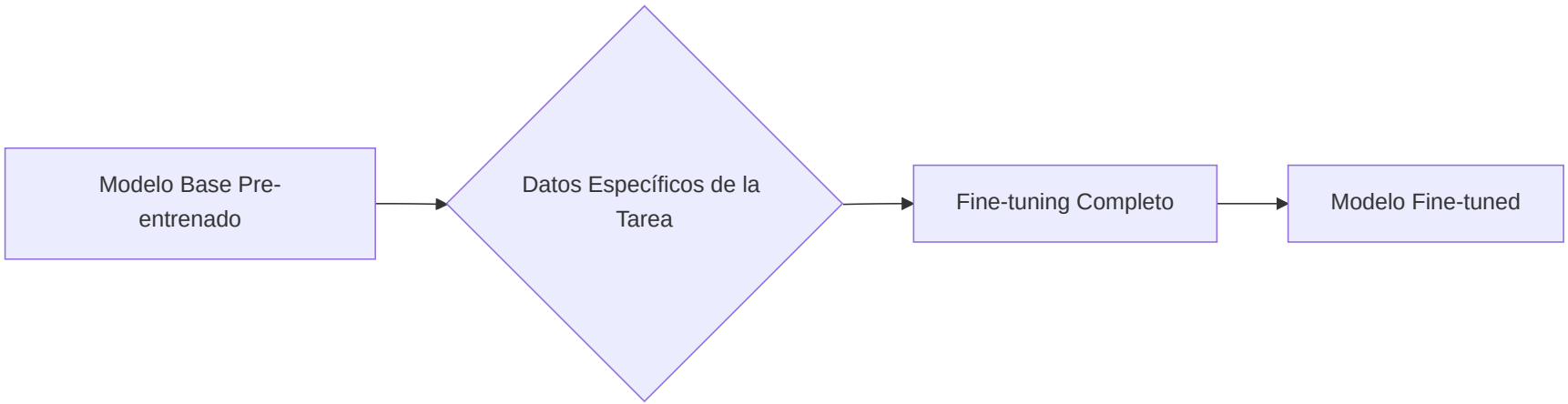
El **Fine-tuning** es el proceso de adaptar un Modelo de Lenguaje Grande (LLM) pre-entrenado a una tarea o conjunto de datos específico. A diferencia del pre-entrenamiento (que se realiza con grandes volúmenes de datos generales), el fine-tuning utiliza un conjunto de datos más pequeño y específico para "especializar" el modelo.

¿Por qué hacer Fine-tuning?

- **Mejorar el rendimiento** en tareas específicas (ej. clasificación de texto, resumen de documentos legales).
- **Adaptar el modelo** a un dominio o estilo particular (ej. lenguaje médico, tono de marca).
- **Reducir las "alucinaciones"** en contextos específicos.
- **Aprovechar el conocimiento** general del modelo base, añadiendo especialización.

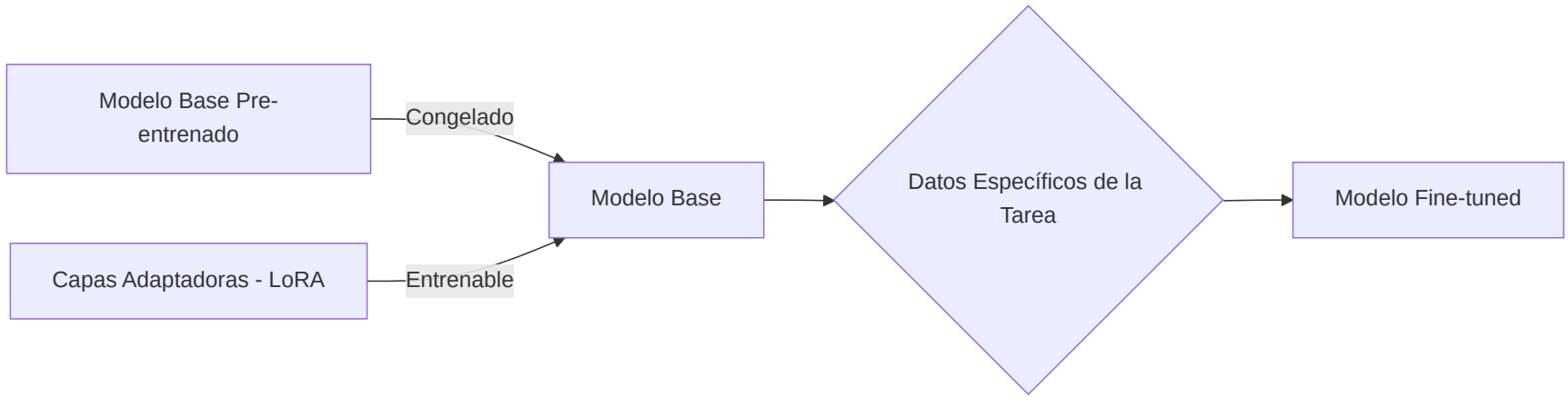
Fine-tuning Completo (Full Fine-tuning)

Se actualizan todos los parámetros del modelo pre-entrenado utilizando el nuevo conjunto de datos.



Fine-tuning Eficiente en Parámetros (PEFT) / LoRA

Solo se actualiza un pequeño subconjunto de parámetros o se añaden pequeñas capas "adaptadoras" (como en LoRA) que se entrenan, mientras que la mayoría del modelo base permanece congelado.



Ventajas y Desventajas del Fine-tuning

- **Rendimiento Mejorado:** Mayor precisión en tareas específicas.
- **Adaptación a Dominio:** Se ajusta a terminología y estilo específicos.
- **Menos Datos:** Requiere menos datos que el pre-entrenamiento.
- **Costo-Eficiencia (PEFT):** Reduce costos computacionales y de almacenamiento.

- **Costo (Full FT):** Alto costo computacional y tiempo.
- **Olvido Catastrófico:** Riesgo de perder conocimiento general (Full FT).
- **Datos de Calidad:** Necesita datos de fine-tuning de alta calidad.
- **Overfitting:** Riesgo si el dataset es muy pequeño.