

Agente Tutor en Bases de Datos: Integración de LLM y Grafos de Conocimiento para una Orientación Efectiva en Prácticas de Álgebra Relacional y SQL

Ricardo Di Pasquale, Agustín Filippe, Agustín Bolívar, Marco Fernandez,
Natacha Soledad Represa, and Carlos Marcelo Benítez

Pontificia Universidad Católica Argentina, Facultad de Ingeniería y Ciencias Agrarias,
Laboratorio de Ciencias de Datos e Inteligencia Artificial, Buenos Aires, Argentina
rdipasquale@uca.edu.ar - agustinfilippe@uca.edu.ar -
agustinbolivar@uca.edu.ar - marcofernandez@uca.edu.ar -
natacharepresa@uca.edu.ar - carlosbenitez@uca.edu.ar

Abstract. El presente trabajo expone el desarrollo de un agente tutor basado en modelos grandes de lenguaje (LLM) y grafos de conocimiento (KG) con implementación en bases de datos orientadas a grafos, dedicado a asistir a estudiantes de carreras de ingeniería en la materia de Bases de Datos, pudiendo, en el futuro ser aumentado para prácticas de otras materias. Particularmente, este trabajo se enfoca en la resolución de ejercicios de álgebra relacional y SQL, proponiendo una solución educativa que fomenta la autonomía del alumno sin proporcionar respuestas directas, sino acompañamiento para la resolución adecuada del problema ayudando al alumno a recalibrar su razonamiento para llegar a una solución compatible con la respuesta al ejercicio.

La propuesta se fundamenta en la integración de dos componentes tecnológicos esenciales. Por un lado, el flujo agéntico basado en LLM se encarga de procesar el lenguaje natural, capturando la esencia de la pregunta y generando respuestas orientativas, así como implementar el flujo de razonamiento basado en el conocimiento almacenado en el KG. Por otro lado, el KG almacena el conocimiento teórico derivado del contenido curricular de la asignatura y las directrices pedagógicas definidas por el cuerpo docente. Esta estructura (ontología y taxonomía) garantiza que la asistencia brindada sea coherente con los objetivos académicos y el estilo particular del curso, evitando conocimientos generales del LLM que puedan contradecir los lineamientos del programa.

Una de las características distintivas del agente tutor es su capacidad para interactuar de forma dinámica y personalizada con cada estudiante. Mediante el análisis del lenguaje y del contexto de la consulta, el tutor identifica áreas de dificultad y ofrece pistas y sugerencias que orientan al alumno en la resolución del ejercicio. Este proceso de retroalimentación se diseña para reforzar los avances del estudiante y corregir errores constructivamente, sin revelar la respuesta final de forma explícita. Además, el tutor es capaz de vincular la práctica con la teoría, permitiendo que el alumno acceda al detalle teórico específico cuando sea necesario.

El modelo incorpora la posibilidad de ser ajustado y personalizado según las necesidades del curso. Los docentes pueden definir y actualizar los patrones y objetivos de la práctica, permitiendo que el agente tutor actúe como refuerzo del estilo del curso. Esta adaptabilidad asegura una mayor congruencia entre la asistencia automatizada y el contenido curricular, facilitando la incorporación de nuevas directrices pedagógicas.

La ontología del KG está orientada al refuerzo y al feedback constante, permitiendo tanto el trabajo en particular en ejercicios específicos, como la abstracción razonada en temáticas específicas, o en conjuntos de ejercicios, favoreciendo la capacidad de generalización del agente.

En resumen, este trabajo representa un avance en la aplicación de tecnologías emergentes en el ámbito educativo, combinando inteligencia artificial y estructuración del conocimiento para crear un agente tutor que refuerza el proceso de aprendizaje. La sinergia entre el flujo agéntico basado en LLMs y el KG da lugar a una herramienta innovadora que potencia el desarrollo de habilidades críticas y analíticas en los estudiantes.

Keywords: AI, LLM, knowledge graph, graphRAG, agentic workflow

1 Introducción

El uso extendido de los modelos grandes de lenguaje (LLM) en el ámbito de la educación universitaria supone a la vez una oportunidad y un desafío tanto para alumnos como docentes. Patrones como el oponente socrático [Liu et al., 2024] son utilizados para ayudar a los alumnos a mejorar la defensa de sus trabajos mediante el entrenamiento con un agente capaz de desafiar sus producciones escritas.

En el ámbito de la práctica, el uso de herramientas de inteligencia artificial (IA) ha estado más orientado a la resolución de problemas, en términos de buscar la respuesta correcta, o entender alguna solución provista por el docente.

En este trabajo estamos interesados en la creación de un agente capaz de acompañar como tutor la ejecución de ejercicios prácticos por parte del alumno. Este acompañamiento se orienta a comprender las dificultades que presenta el alumno, y tratar de encaminarlo hacia la solución apoyándose en la teoría y en recomendaciones. Nunca el agente provee la respuesta correcta, sino que evalúa constantemente cuál es la diferencia en el razonamiento del alumno que lo hace alejarse de la respuesta, y mediante consejos intenta re-encaminar al alumno en el proceso de resolver adecuadamente el ejercicio.

A continuación se presenta una interacción ilustrativa con el agente. El diálogo se inicia a partir de un ejercicio catalogado (ingestado en el KG).

Contexto: El ejercicio 1.d de la práctica 2 pide construir una sentencia de álgebra relacional que devuelva *"Nombre de los clientes que no han comprado nada"*. El esquema utilizado para toda la sección 1 de la práctica 2 es el siguiente:

CLIENTES (*NroCliente* (*clave*), *Nombre*, *Dirección*, *Teléfono*, *Ciudad*)

PRODUCTO(*Cod Producto* (*clave*), *Descripción*, *Precio*)

VENTA(*Cod Producto*, *NroCliente*, *Cantidad*, *Id Venta* (*clave*))

Alumno: Necesito ayuda con el ejercicio 1.d de la práctica 2. No sé cómo plantear el left join de manera que descalifique a los clientes que no tengan una venta asociada.

Agente: De acuerdo! Veamos... ¿Te parece que la operación central adecuada para el ejercicio es una junta?

Alumno: Si ¿Por qué no? Si puedo hacer left join desde CLIENTES a VENTAS quizás puedo evaluar donde hay nulos en la parte derecha, pero no me sale fácil.

Agente: Razonemos un poco ¿No hay alguna operación más adecuada para tipificar al ejercicio? Recuerda que pide encontrar un conjunto de CLIENTES que no hayan comprado nada. Quizás si pensamos CLIENTES como el conjunto universo de CLIENTES podríamos identificar muy fácil el conjunto de CLIENTES que compraron algo juntándolo con las VENTAS ¿Verdad?

Alumno: OK. Entonces puedo hacer una diferencia de conjuntos entre el Universo y los que compraron algo.

Agente: Excelente! ¿Cómo quedaría eso?

Alumno: $(CLIENTES) - (CLIENTES \bowtie_{CLIENTES.NroCliente=VENTA.NroCliente} VENTA)$

Agente: El intento es bueno, pero fíjate que el formato de los dos conjuntos es distinto, no podemos aplicar una diferencia de conjunto a dos relaciones que tienen un esquema distinto.

Alumno: $\Pi_{Nombre}(CLIENTES) - \Pi_{Nombre}(CLIENTES \bowtie_{CLIENTES.NroCliente=VENTA.NroCliente} VENTA)$

Agente: En ningún lado se enunció la unicidad del atributo *Nombre* ¿Qué inconvenientes puede traer proyectar por Nombre al hacer la diferencia de conjuntos?

Alumno: Ah, si hay clientes con mismo nombre pero distinto NroCliente se generarían inconsistencias en las respuesta.

Agente: Exacto!

Alumno: Ok, entonces solo proyecto en la parte derecha los atributos de la relación CLIENTE y ahí debería estar correcto, verdad?

Agente: Perfecto! Bien hecho.

2 Construcción del Grafo de Conocimiento KG

El primer paso en la construcción del KG consistió en recopilar los apuntes de la materia. Esto es importante dado que se busca dar un soporte teórico a la gestión de las prácticas. Por otro lado, el agente es capaz de resolver preguntas teóricas utilizando el conocimiento almacenado, pero ese es algo que va más allá del alcance propuesto en el presente trabajo. La metodología empleada en este trabajo para construir el subgrafo de conocimiento teórico se explica en [Karan Chellani, 2025] donde se presenta una metodología para extraer el conocimiento teórico, modelarlo es un grafo almacenado en una base de datos (orientada a grafos) Neo4J y optimizar su explotación. En la Fig.1 se ilustra el esquema del

grafo de conocimientos resultante de ingestar los apuntes de la teoría de la materia. De esta manera, el conocimiento queda modelado en el KG, de la misma manera que el texto se incluye también en nodos tipo *chunk* asociado a los nodos del KG, posibilitando las consultas híbridas [Alex Thomas, 2024] tal como se verá más adelante.

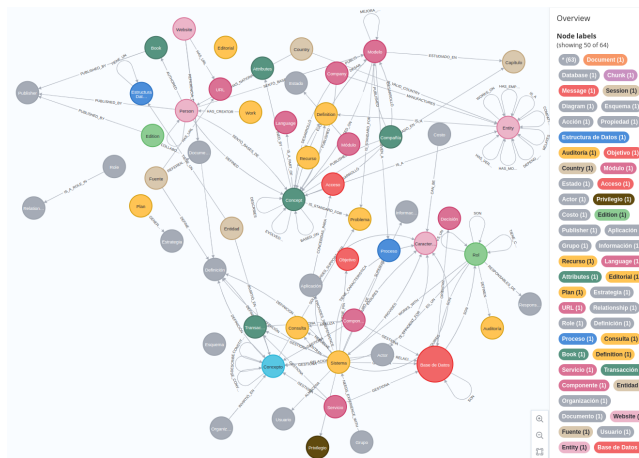


Fig. 1. Grafo de Conocimiento - Subgrafo obtenido al procesar los apuntes de la teoría.

Para la ingesta de las prácticas, se solicitó a los docentes a cargo modelar cada práctica contemplando una cabecera y un detalle. El esquema de datos de la cabecera es el siguiente: { *PracticaNumero: Int; PracticaTitulo: String; Descripcion: String; Objetivos: String; TeoriaRelacionada: list[Apunte]; tips: list[String]* }. El esquema para el detalle: { *PracticaNumero: Int; Tipo: Tipo; Ejercicio: String; Enunciado: String; Respuesta: String; tips: list[String]* }. Cabe aclarar que el atributo tipo en el detalle apunta a saber si el punto se trata de un ejercicio, o de un enunciado. En los tips, que en realidad pueden ir asociados a cualquier elemento de la práctica y de los contenidos teóricos, se pretende abstraer el enfoque de los docentes en la práctica. De forma ilustrativa, en el presente caso, para los docentes de la práctica es muy importante tanto en álgebra relacional como en SQL, que los alumnos sean capaces de diferenciar la semántica de las operaciones. Por ejemplo, si un enunciado establece claramente que se espera operar una diferencia de conjuntos, se espera que los alumnos apliquen la diferencia de conjuntos tanto a nivel de álgebra relacional, como de SQL (*not in, not exists, minus, except*). Los docentes saben que la respuesta podría modelarse como un subconjunto del producto cartesiano, pero quieren hacer énfasis en la utilización adecuada de los operadores, por lo que pueden reforzar mediante tips a nivel de teoría y a nivel de práctica global estos principios.

Una vez inicializado el KG, se comienza a trabajar con el flujo agéntico que se detallará en la siguiente sección. El objetivo es completar recomendaciones y

tipos de manera que el producto responda de la mejor manera posible. El esquema relacionado con la práctica puede visualizarse en la Fig.2. Es muy importante comprender la relevancia de los nodos tipo *tip* y *EjemploInteraccion*, así como de las relaciones de los mismos y su peso (propiedad *weight*) a la hora de la toma de decisiones en el flujo agéntico. La potencialidad de generalizar conocimiento pedagógico del flujo agéntico depende de la evolución de estos componentes. El *feedback* de los usuarios y de los docentes se refleja en el incremento (o decremento) del peso de las relaciones; así como de la incorporación de interacciones exitosas al KG a través de los nodos tipo *EjemploInteraccion*. El ajuste adecuado (a lo largo del tiempo) de estos parámetros, permite al flujo agéntico poder obtener los refuerzos (*few shots*) ideales para que el LLM pueda referir conductas generalizables a través de ejemplos.

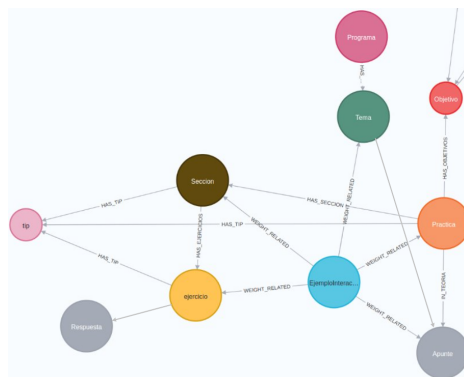


Fig. 2. Grafo de Conocimiento - Subgrafo obtenido al procesar las prácticas

3 Construcción del Flujo Agéntico (Agentic Workflow)

El flujo multi-agéntico ha sido modelado en el framework *langgraph* (Wang and Duan, 2024) tal como puede observarse en la Fig.3. Este modelo permite la persistencia del estado del *chat*, pudiendo retomar la conversación y agregar contexto en cada interacción. El elemento central de esta implementación consiste en la posibilidad de explotar el KG de manera dinámica a través de la técnica de *hybrid retrieval* [Alex Thomas, 2024] que permite navegar el grafo de conocimiento realizando consultas que a la vez filtren nodos y arcos del esquema del grafo, así como, explotando los *embeddings* de texto almacenados en nodos y arcos (con algoritmos como *cosine similarity* entre otros).

El agente *IntentEvaluation* se encarga de comprender la consulta (y el contexto) del alumno, de manera de poder decidir si hacer un pedido de aclaración (o un rechazo de los términos) o seguir adelante. El *Strategy Supervisor* se encarga de rutear el flujo, y mediar en el ecosistema de agentes dentro de su alcance.

El *Gap Analyzer* es capaz de medir la distancia entre las respuestas correctas y el intento del alumno. El *Theory Builder* es capaz de concatenar razonamientos teóricos en función del flujo del ejercicio y la teoría almacenada en el KG. El *Pedagogy Builder* es propiamente el tutor que una vez que se tiene el contexto teórico y el análisis del gap, sugiere un siguiente paso en términos pedagógicos. El *Judge* es un agente capaz de evaluar la eficacia de la acción propuesta, pudiendo sugerir ajustes e iterar el flujo. Finalmente el agente *writer* es capaz de volcar en texto con estilo pedagógico la respuesta al alumno.

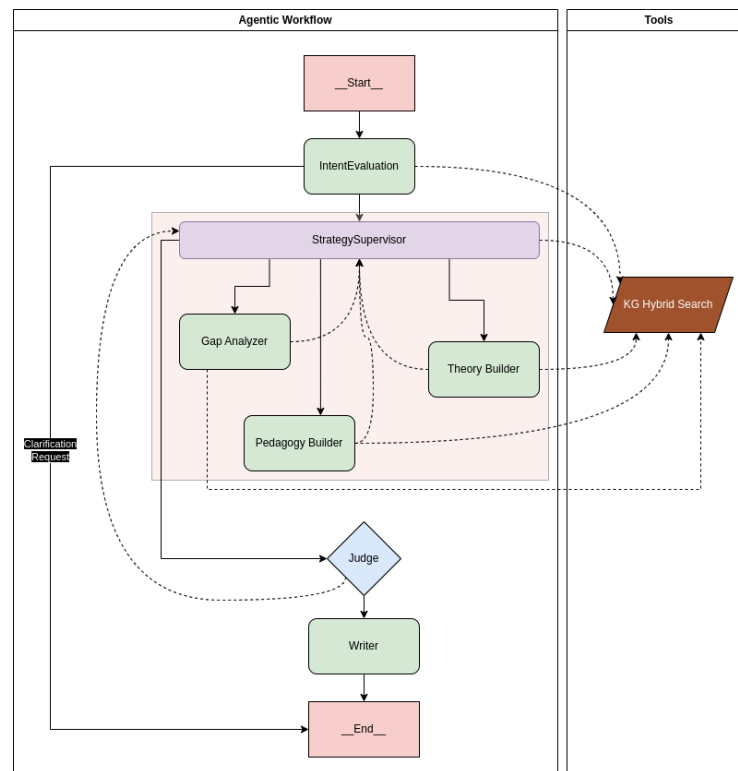


Fig. 3. Flujo multi-agente

4 Conclusión y futuros Desarrollos

En el presente trabajo hemos podido mostrar la implementación de una prueba de concepto exitosa combinando LLM y KG. Desde el punto de vista de las métricas, hemos dado los primeros pasos utilizando dos métricas muy básicas, como la *tasa de abandono* (proporción de alumnos que no siguen adelante la discusión con el agente) y la *tasa de éxito* (proporción de alumnos que son capaces

de llegar a una respuesta correcta de un ejercicio). Otras métricas utilizadas han sido provistas por humanos expertos (docentes) que han calificado la interacción de los agentes, así como métricas subalternas relacionadas al mapeo de entidades relevantes realizadas durante la etapa de evaluación de la intención del alumno (cantidad de entidades detectadas que han podido ser mapeadas vía NER, o cantidad de ejemplos relevantes ubicados en el KG). Creemos que es importante desarrollar mejores métricas para escalar el desarrollo, tanto en el plano *coarsed grain* (donde se inscriben las técnicas actuales) como en el plano *fine grain* que permitirán mayor control unitario de los flujos.

La utilización de un framework como *Langfuse* [Oleh Dubetcky, 2025] ha sido un acierto tanto desde el punto de vista del mantenimiento de una *test-suite* orientada a los agentes, así como desde el punto de vista de observabilidad del LLM. Métricas y Evaluación

Un punto pendiente a la hora de industrializar este producto es la utilización de Guardrails y otros medios para salvaguardar las interacciones de manera sana y preservando la ética que es requerida de un agente de inteligencia artificial en la interacción con alumnos.

References

- Alex Thomas. (2024). Enhancing hybrid retrieval with graph traversal using the graphrag python package. Retrieved March 27, 2025, from <https://neo4j.com/blog/developer/enhancing-hybrid-retrieval-graphrag-python-package/>
- Karan Chellani. (2025). Introduction to the neo4j llm knowledge graph builder. Retrieved March 27, 2025, from <https://neo4j.com/blog/developer/llm-knowledge-graph-builder/>
- Liu, J., Huang, Z., Xiao, T., Sha, J., Wu, J., Liu, Q., Wang, S., & Chen, E. (2024). Socraticlm: Exploring socratic personalized teaching with large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, & C. Zhang (Eds.), *Advances in neural information processing systems* (pp. 85693–85721, Vol. 37). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2024/file/9bae399d1f34b8650351c1bd3692aeae-Paper-Conference.pdf
- Oleh Dubetcky. (2025). Mastering llm observability: A hands-on guide to langfuse and opentelemetry comparison. Retrieved March 27, 2025, from <https://oleg-dubetcky.medium.com/mastering-llm-observability-a-hands-on-guide-to-langfuse-and-opentelemetry-comparison-33f63ce0a636>
- Wang, J., & Duan, Z. (2024). Agent ai with langgraph: A modular framework for enhancing machine translation using large language models. <https://arxiv.org/abs/2412.03801>