



## **Tecnicatura Universitaria en Inteligencia Artificial**

### **Procesamiento del Lenguaje Natural**

#### **Trabajo Práctico N°2 - RAG y Agente ReAct**

#### **Docentes:**

Geary, Alan

Masón, Juan Pablo

Ferrucci, Constantino

Solberger, Dolores

#### **Alumna:**

Sumiacher, Julia

S-57932

# Índice

## 1. RAG

- 1.1. Breve introducción
- 1.2. Resumen
- 1.3. Desarrollo de la base de datos vectorial
- 1.4. Desarrollo de la base de datos de grafo
- 1.5. Desarrollo de la base de datos tabular
- 1.6. Desarrollo de queries dinámicas
- 1.7. Desarrollo de los clasificadores
- 1.8. Desarrollo de chatbot para usuario
- 1.9. Conclusiones
- 1.10. Herramientas utilizadas

## 2. Agente ReAct

- 2.1. Resumen
- 2.2. Desarrollo del agente
- 2.3. Conclusiones
- 2.4. Herramientas utilizadas

# 1. Recuperación Aumentada con Generación (RAG)

## 1.1. Breve Introducción

7 Wonders es un juego de mesa creado por Antoine Bauza en 2010 y publicado originalmente por Repos Production (parte de Asmodee Group). Tres mazos de cartas con imágenes de civilizaciones históricas, conflictos armados y actividad comercial se utilizan en el juego de dibujo de cartas 7 Wonders. Es un juego de estrategia y desarrollo de civilizaciones en el que los jugadores asumen el papel de líderes de antiguas ciudades estado. A lo largo de tres eras, cada jugador debe tomar decisiones clave para construir su ciudad, desarrollar su maravilla y acumular la mayor cantidad de puntos de victoria. La mecánica principal del juego se basa en la selección de cartas mediante un sistema de *drafting*, en el que cada jugador elige una carta de su mano y pasa el resto a su vecino, lo que obliga a pensar no solo en la propia estrategia, sino también en la de los demás. El juego recibió el éxito de la crítica tras su lanzamiento y ganó numerosos premios, incluido el premio inaugural Kennerspiel des Jahres connoisseurs en 2011.

## 1.2. Resumen

Esta sección detalla el desarrollo de un chatbot experto en el juego de mesa 7 Wonders, utilizando un sistema de Recuperación Aumentada con Generación (RAG) implementado para gestionar y extraer información proveniente de varias fuentes: un PDF con reglas, foros de usuarios, información estructurada, y estadísticas de distintas bases de datos (vectorial, de grafo y tabular). Para lograr esto, se han implementado diversas estrategias como embeddings semánticos, queries dinámicas y clasificadores basados en aprendizaje automático. Además, se ha desarrollado un chatbot que integra estas bases de datos para responder preguntas en lenguaje natural de manera eficiente y precisa.

## 1.3. Desarrollo de la base de datos vectorial

Para el desarrollo de la base de datos vectorial, se extrae información de tres fuentes diferentes.

### a. PDF

Primero se definen funciones para extraer y limpiar texto, cruciales para luego convertirlos en embeddings. Entre ellas encontramos:

- *limpiar\_texto*: Limpia caracteres no deseados, reduce espacios, etc.
- *extract\_text\_with\_pypdf2*: Extrae texto de un PDF usando PyPDF2 y luego lo limpia
- *split\_into\_sections\_auto*: Dada una lista de títulos en el PDF, extrae fragmentos de texto para cada sección.
- *process\_text\_to\_vectors*: Toma una lista de segmentos de texto y genera embeddings usando un modelo con `method = encode`.

Luego, se extrae el texto del manual de reglas “7 Wonders Cities”. Para esto, almacenamos en la variable *indices\_manual* la lista de títulos esperados, como 'Contenido', 'En resumen', 'Elementos del juego', etc. Posteriormente nos valemos de *split\_into\_sections\_auto* para dividir el texto del PDF en secciones basadas en esos títulos.

### b. Foros (BoardGameGeek y MisutMeeple)

Se utilizan funciones para obtener contenido desde la página dinámica haciendo uso de la librería Selenium :

- *extraer\_comentarios*: Para BoardGameGeek.
- *extraer\_todo\_el\_texto*: Para MisutMeeple.

Se utiliza *replace()* y *strip()* para reemplazar saltos de línea y eliminar espacios respectivamente.

De esta forma, se construye un corpus de texto con secciones del PDF (reglas y contenidos), comentarios del foro de BoardGameGeek y la reseña de Misut Meeple. Luego, para manejar fragmentos de texto de longitud adecuada, se define:

- *chunker*: Utiliza *RecursiveCharacterTextSplitter()* para dividir el texto en bloques (chunks) de tamaño máximo 400 caracteres, con overlap de 30 para no perder contexto entre bloques. Estos “chunks” se agrupan en listas: *chunk\_reglas* para el PDF y *chunk\_opiniones* para BoardGameGeek/Misut Meeple.

Posteriormente, se carga un modelo de embeddings, en este caso *Universal Sentence Encoder* multilenguaje, y se inicializa un cliente de *ChromaDB* con persistencia. Para llenar la colección *vec\_db* a partir de texto chunked crea:

- *fill\_db*: Itera sobre cada documento, obtiene embedding, y lo agrega a ChromaDB.

Con esto, la base de datos vectorial queda poblada con embeddings para cada fragmento.

### 1.3. Desarrollo de la base de datos de grafo

Mediante Selenium, se visitan páginas de BoardGameGeek para la extracción de datos de diseñadores, artistas, publicadores, etc. Para esto se crean:

- *encontrar\_url*: Extrae todas las URLs relevantes de un tipo específico (diseñadores, artistas, editores, etc) desde la página de créditos en el sitio web BoardGameGeek.
- *encontrar\_datos*: Para cada tipo (designer, artist, publisher, related\_to, etc) se extraen nombres, premios, juegos adicionales diseñados, etc.

A partir de esto, se obtienen: *dict\_desingers*, *dict\_artist*, *dict\_publishers*, *dict\_related\_to*, *dict\_alternative\_names* y *dict\_awards*. Se utilizan las distintas entidades para la construcción de un grafo orientado *nx.DiGraph()* con *NetworkX*.

En primer lugar se añade el nodo principal "7 Wonders", y luego se lo relaciona con otros nodos de tipo diseñador, artista, publicador, premios, etc. utilizando *G.add\_edge()*. Por último, se visualiza con *Plotly* (trazos de nodos, aristas y layout).

Posteriormente, para serializar en formato RDF se usa *rdflib* y se lo guarda en un archivo *graph\_db.rdf*.

### 1.4. Desarrollo de la base de datos tabular

Para la extracción de información estadística se define:

- *obtener\_estadisticas\_y\_valores*: Recoge estadísticas y valores de una página web utilizando Selenium, extrayendo información relevante sobre estadísticas de juegos y jugadores.

Se capturan valores: "Avg. Rating", "No. of Ratings", etc. Se guardan en un DataFrame *df\_valores* y se exporta a CSV.

## 1.5. Desarrollo de queries dinámicas

Se desarrollan funciones para realizar consultas a las bases de datos.

### a. Búsqueda en la base vectorial: híbrida + ReRank

- *keyword\_score*: Calcula un puntaje de coincidencia de palabras clave basado en la cantidad de ocurrencias de cada término de la consulta en el texto.
- *search\_vectorial*: Calcula embedding para la query, consulta la base vectorial con *n\_results* y reajusta el ranking mezclando la similitud semántica con un factor de *keyword\_score*. Finalmente, retorna un listado de tuplas (documento, *puntaje\_final*).

### b. Búsqueda en la base de grafo: queries dinámicas

- *search\_graph*: Genera y ejecuta dinámicamente una consulta *SPARQL* a partir del prompt, recuperando únicamente los elementos relacionados con el término buscado. Luego, devuelve una respuesta formateada en lenguaje natural.

### c. Búsqueda en la base tabular: queries dinámicas

- *search\_tabular*: Identifica la entidad (por ejemplo, “calificación promedio”) dentro del prompt. Busca en *df\_valores* la fila correspondiente a la “Entidad”. Finalmente, retorna su valor numérico.

## 1.6. Desarrollo de los clasificadores

El clasificador es el encargado de analizar el prompt ingresado por el usuario y determinar a cuál de los tres recursos disponibles enviar la consulta para obtener un contexto. Luego, la consulta enviada al LLM se conforma de prompt + contexto. Se diseñan dos versiones de clasificadores:

### a. Clasificador basado en un modelo entrenado y embeddings

Se genera un dataset de queries etiquetadas que apuntan a tres clases, con 50 ejemplos por cada una de ellas:

- 0: base vectorial (reglas, reseñas).
- 1: base de grafo (diseñador, artista, premios).
- 2: base tabular (estadísticas, calificaciones).

Se divide en train y test. Luego se calculan embeddings con un *SentenceTransformer*, y se entrena un modelo *LogisticRegression*.

Finalmente, se imprime el reporte de clasificación y se define *classifier\_relog()* para usarlo en producción.

#### b. Clasificador basado en LLM

Se define:

- *classifier\_llm*: Envía un mensaje al modelo LLM (en este caso, “Qwen/Qwen2.5-72B-Instruct”) indicándole que clasifique la consulta con una etiqueta: relations, statistics, content, reviews o rules. Dependiendo de la etiqueta, se asigna un valor numérico (1 = grafo, 2 = tabular, 0 = vectorial).

## 1.8. Desarrollo de chatbot para usuario

Se integra todo el proceso en la siguiente función:

- *chatbot*: Muestra un mensaje de bienvenida, espera que el usuario ingrese una pregunta, detecta el idioma usando *langdetect* y lo traduce al inglés si no está ya en inglés con *translate\_prompt*, pasa la consulta al clasificador (sea LLM o regresión logística), busca en la base de datos correspondiente con *search\_graph*, *search\_tabular* o *search\_vectorial*, usa un LLM (*InferenceClient*) para generar la respuesta, pero incluye en el prompt el “context” (el resultado de la búsqueda), traduce la respuesta de vuelta al idioma original del usuario (*translate\_response*). Finalmente, muestra el resultado en pantalla.

## 1.9. Conclusiones

Se logró un flujo de RAG que integra múltiples fuentes: vectorial (texto de foros/PDF), grafo (relaciones del juego), y tabla (estadísticas del juego). El sistema está diseñado para adaptarse a consultas de distinta naturaleza (sobre autores, puntuaciones, reglas, reseñas, etc.). Se emplea un enfoque multilingüe con traducción automática según el idioma del usuario. La clasificación de intenciones permite dirigir la consulta a la base de datos adecuada.

Dentro de las dificultades presentadas encontramos múltiples inconvenientes con el entorno de Colab, teniendo que reiniciar el entorno periódicamente para solucionar problemas, además de incompatibilidades de versiones entre distintas librerías. Por otro

lado, el web scrapping fue un proceso de prueba y error, en el que fueron necesarios numerosos ajustes en el código para lograr un resultado aceptable.

Hello!!! I am your virtual assistant specialized in the board 7 Wonders.  
You can ask in your language.

How can I help you? To exit type: - salir  
Enter your question:¿Cuáles son las reglas del juego?  
Las reglas básicas de 7 Wonders incluyen:

1. **\*\*Preparación del Juego\*\***:
  - Elija una Civilization Board para cada jugador y coloque una Wonder Card a su lado.
  - Prepare los mazos de cartas para cada Era (I, II, III) según el número de jugadores.
  - Reparte 7 monedas a cada jugador.
2. **\*\*Turnos del Juego\*\***:
  - Cada Era consta de 6 rondas.
  - En cada ronda, cada jugador elige una carta de su mano y la juega simultáneamente.
  - Las cartas pueden ser construidas (pagando el costo), construyendo una etapa de tu maravilla, o descartada para obtener monedas.
3. **\*\*Tipos de Cartas\*\***:
  - **\*\*Raw Materials\*\***: Proporcionan recursos básicos.
  - **\*\*Manufactured Goods\*\***: Proporcionan recursos refinados.
  - **\*\*Commerce\*\***: Generan monedas.
  - **\*\*Science\*\***: Proporcionan símbolos de ciencia.
  - **\*\*Military\*\***: Fortalecen el ejército.
  - **\*\*Civilian Structures\*\***: Proporcionan puntos de victoria y beneficios.
  - **\*\*Guilds\*\***: Proporcionan puntos de victoria basados en la configuración del juego.
4. **\*\*Resolución de Conflictos\*\***:
  - Al final de cada Era, resuelve conflictos militares. Los jugadores con más fuerza militar ganan y reciben puntos de victoria.
5. **\*\*Final del Juego\*\***:
  - Tras la tercera Era, se calculan los puntos de victoria totales.
  - Se suman puntos de victoria de construcciones, etapas de maravillas, monedas, conflictos militares, y ciencia.
  - El jugador con más puntos de victoria gana.

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:¿Quiénes publicaron 7 Wonders?  
7 Wonders fue publicado por Repos Production y distribuido en varios países por diferentes compañías, como Asmodee en Estados Unidos.

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:¿Cuántos jugadores se registraron en el juego?  
8 jugadores

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:Menciona una estrategia para ganar 7 Wonders  
Construir una civilización comercial fuerte, enfocándose en la producción y comercio de recursos para ganar monedas y comprar recursos de tus vecinos.

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:¿Quién diseñó 7 Wonders?  
Antoine Bauza diseñó 7 Wonders.

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:Dime las ventajas y desventajas de 7 Wonders  
Ventajas:

- Juego rápido y fácil de aprender.
- Buena interacción entre los jugadores.
- Varias estrategias posibles.

Desventajas:

- Puede ser complicado para principiantes.
- La suerte en la elección de cartas puede influir.
- Repetitivo si se juega mucho.

¿En qué puedo ayudarte? Para salir escribe: - salir  
Ingrese su pregunta:Dame tu opinión de 7 Wonders  
7 Wonders es un excelente juego de mesa que combina estrategia, rápidas decisiones y un atractivo tema histórico. Es fácil de aprender pero ofrece profundidad suficiente para mantener interesados a jugadores experimentados. La mecánica de draft de cartas es muy ingeniosa y asegura partidas dinámicas y diferentes cada vez. Recomendado para quienes disfrutan de juegos de estrategia ligeros y sociales.



## 1.10. Herramientas utilizadas

*Python*. Este es el lenguaje de programación que se utiliza para orquestar toda la solución. Se encarga de coordinar los llamados a bibliotecas de extracción de datos, procesar texto, entrenar clasificadores, manejar la comunicación con modelos de lenguaje, etc. <https://www.python.org/>

*Colab*. Para ejecutar el código y organizar los experimentos, se emplea un entorno de cuadernos interactivos. Estos cuadernos permiten combinar texto, código y resultados en el mismo documento. <https://colab.research.google.com/>

*Selenium*. Es una librería que permite automatizar la interacción con navegadores web. En el proyecto se emplea para extraer contenido dinámico de foros o sitios que cargan datos con JavaScript. Se utiliza para abrir páginas web de foros, esperar a que aparezcan elementos en pantalla y extraer el texto de los comentarios y reseñas. <https://www.selenium.dev/>

*Langdetect*. Es una librería que detecta de forma automática el idioma de un texto. Utiliza un algoritmo entrenado sobre distintos corpus. En el proyecto fue utilizado para determinar si la pregunta del usuario está en español, inglés u otro idioma y para realizar traducciones condicionales (si no está en inglés, se traduce). <https://pypi.org/project/langdetect/>

*TensorFlow & TensorFlow Text*. Son frameworks para la construcción, entrenamiento e inferencia de modelos de deep learning. En este caso, se emplean principalmente para cargar Universal Sentence Encoder Multilingual. <https://www.tensorflow.org/>

*PyPDF2*. Es una librería para leer y manipular documentos PDF. En el proyecto se usa para extraer texto de un reglamento de juego en PDF. <https://pypi.org/project/PyPDF2/>

*ChromaDB*. Es un motor de base de datos vectorial que facilita el almacenamiento y consulta de embeddings. Con ello, se pueden realizar búsquedas semánticas y organizar grandes volúmenes de vectores de forma eficiente. En este proyecto se lo utiliza para guardar cada fragmento (“chunk”) de texto junto con sus embeddings, permitir la búsqueda por similitud de embeddings cuando llega una consulta del usuario y combinar la similitud semántica con un factor de palabras clave. <https://docs.trychroma.com/>

*Deep-translator*. Es una librería Python que ofrece una interfaz sencilla para distintos servicios de traducción, incluyendo Google Translate. Permite traducir texto de forma automática entre distintos idiomas. Entre los usos principales en este proyecto encontramos traducir prompts y respuestas entre inglés y español, según se requiera, e integrar la función GoogleTranslator para la detección y traducción. <https://pypi.org/project/deep-translator/>

*Rdflib*. Permite trabajar con modelos de grafos RDF en Python: creación, manipulación y consultas SPARQL. Se utiliza para modelar la información en forma de tripletas

(sujeto–predicado–objeto). Entre los usos principales en este proyecto encontramos construir un grafo RDF, serializarlo a un archivo .rdf y consultar con SPARQL para recuperar información (diseñadores, premios, nombres alternativos) <https://rdflib.readthedocs.io/>

*SpaCy*. Es un framework avanzado de NLP. Aunque en el código se descarga “es\_core\_news\_sm” (el modelo de español), su uso principal puede variar según las necesidades de tokenización, lematización, etc. <https://spacy.io/>

*NLTK*. Es un conjunto de librerías y recursos para procesar lenguaje natural en Python. Se usa para tokenización, manejo de stopwords, etc. <https://www.nltk.org/>

*Scikit-learn*. Se emplea en el proyecto para entrenar clasificadores, como la regresión logística que decide si la consulta del usuario va dirigida a la base vectorial, al grafo, o a la base tabular. <https://scikit-learn.org/>

*Sentence-transformers*. Es una biblioteca que facilita el uso de modelos tipo BERT/SBERT para producir embeddings de oraciones y medir similitud semántica. En el proyecto, se usa para obtener representaciones vectoriales de las consultas de usuario y del corpus de entrenamiento (para el clasificador logístico). <https://www.sbert.net/>

## 2. Agente ReAct

### 1.1. Resumen

ReAct es un enfoque para integrar razonamiento lógico y uso de herramientas en agentes basados en modelos de lenguaje (LLMs). Su nombre proviene de la combinación de "Reasoning" (razonamiento) y "Acting" (acción). Está diseñado para permitir que un agente resuelva problemas de manera más efectiva al alternar entre el razonamiento explícito y la ejecución de acciones, como el uso de herramientas externas.

El agente puede dividir tareas complicadas en subtareas manejables, en lugar de intentar resolver todo de una vez. Se trata de un proceso iterativo que culmina cuando encuentra una respuesta adecuada o por límite de tiempo, si no la encuentra.

Configuramos un LLM (en este ejemplo, Ollama con el modelo "llama3.2:latest") que funciona como "cerebro" del agente.

En conclusión, el agente recibe consultas, decide razonando qué herramienta usar y con qué argumentos, observa la salida y produce una respuesta.

### 2.2. Desarrollo del agente

Inicialmente, se importan componentes de *llama\_index* y se definen las funciones (herramientas) con *FunctionTool*. Luego, se crea una instancia de *Ollama* para el LLM.

Definimos tres herramientas principales:

- *get\_info\_graph\_db*: para consultar la base de datos de grafo.
- *get\_info\_tabular\_db*: para la base tabular.
- *get\_info\_vector\_db*: para la base vectorial.

Dentro del *system\_prompt* se le indica al agente:

- Que debe responder exclusivamente con la información proveniente de las herramientas.

- Que identifique la mejor herramienta según la pregunta (sobre grafo, tabular o vectorial).
- El agente estructura su respuesta generando Thought, Action, Action Input, Observation y Final Answer.

El proceso comienza cuando ingresa una pregunta del usuario y desencadena la siguiente secuencia de tareas:

- *Query*: pregunta del usuario
- *Thought*: el agente evalúa la pregunta del usuario y decide qué pasos debe seguir, utilizando su conocimiento general y las herramientas disponibles. Esto ayuda a que el sistema sea transparente y a que podamos entender por qué decide tomar una acción específica.
- *Action*: El agente toma medidas concretas basadas en su razonamiento y decide usar una herramienta para buscar la información.
- *Action Input*: El agente especifica el parámetro de entrada que pasará a la herramienta.
- *Observation*: La herramienta devuelve un resultado. Este resultado se envía de vuelta al agente para que lo procese. Esta es una respuesta parcial y debe ser validada o utilizada para construir la respuesta final.

Finalmente, se define:

- *chat\_con\_agente*: Solicita al usuario la pregunta, llama a *agent.chat(query)* e imprime la respuesta o maneja errores.

## 2.3. Conclusiones

El agente ReAct se apoya en tres “tools” que encapsulan el acceso a cada tipo de base de datos, implementa un flujo de razonamiento, decide cuál herramienta es apropiada y, tras obtener el resultado, elabora la respuesta final. Permite una integración flexible y escalable de más herramientas a futuro.

El resultado de llamar a las herramientas, es clave para construir la respuesta del usuario. Un error en la llamada o recuperar resultados que no tienen que ver con la consulta del usuario, pueden hacer perder el hilo de la conversación y no llegar al resultado esperado, incluso seguir iterando de forma indefinida. Es por eso que suele ponerse un límite de iteraciones para evitar estas situaciones.

Dentro de las dificultades presentadas encontramos las limitaciones del uso de GPU en Colab, y lo frustrante de realizar las pruebas del agente con CPU, lo cual hizo insuficiente el tiempo para realizar todas las pruebas y modificaciones deseadas. Se llegó a aumentar el tiempo de espera de solicitud de 30 a 500 segundos con CPU para obtener alguna respuesta.

Como consideración para futuros trabajos, se recomienda utilizar distintas cuentas de Google para ejecutar el código (o en su defecto pagar la suscripción a Colab Pro) para poder aprovechar los recursos de la GPU, dado que estos son limitados y realmente hacen la diferencia.

Este trabajo práctico fue un desafío interesante, un proceso de mucho aprendizaje en un tiempo muy acotado.

```
¿Quién diseñó 7 Wonders?
> Running step c68b8aff-4194-4cb2-8475-14e225e1c8ca. Step input: ¿Quién diseñó 7 Wonders?
Thought: The current language of the user is: español. I need to use a tool to help me answer the question.
Action: get_info_vector_db
Action Input: {'consulta': 'Diseñadores de 7 Wonders'}
Observation: [('Portada Entre los muchos galardones acumulados por este titulo, hay que destacar el Kennerspiel des Jahres (Juego del Año para Expertos) dentro de los Spiel des Jahres del 2011, es decir, que tiene suficientes credenciales como para ser un Must Have. ¿Que nos propone 7 Wonders? Pues nada mas y nada menos que ponernos al mando de una de las siete civilizaciones que contruyeron alguna de las siete', 4.8160418131399085), ('Esta expansión de 7 Wonders te da la oportunidad de conocer otro aspecto de las grandes ciudades de la Antigüedad. 27 nuevas cartas que te ofrecen edificios nuevos con efectos hasta ahora nunca antes vistos. 3 Gremios, 6 Líderes y 2 Maravillas completan esta expansión. 7 Wonders Cities permite incluir un 8 jugador y también ofrece una variante de juego por equipos. - 3 -', 4.8107995601400875), ('Saltar al contenido Reseña: 7 Wonders Escrito por iMisut 06/02/2013 21 Comentarios Introducción Hoy toca hablar de uno de los juegos que, desde que lo tengo, mas veces ha salido a la mesa. Se trata, nada mas y nada menos que de 7 Wonders, el aclamado y galardonado juego del diseñador francés Antoine Bauza, con las magnificas ilustraciones del también famoso Miguel Coimbra. Portada Entre los muchos', 4.228326319153709), ('mi experiencia, el número mínimo ideal para jugar a 7 Wonders es 4 jugadores, pues con 4 jugadores ya hay aislamiento, es decir, hay jugadores que no están adyacentes entre si, provocando que haya recursos que no estén accesibles a todos, que es uno de los grandes problemas al que se enfrentan los jugadores en 7 Wonders, que no es otro que planificar que recursos interesa que tengamos nosotros en', 3.311234751710273), ('en las fundas para proteger las cartas), por lo que en mi opinión, es un juego caro, sobre todo teniendo en cuenta eso, que es posible quemarlo bastante rápido y que tarde en volver a salir a la mesa. Pero no deja de ser un señor juego que no debería faltar en ninguna ludoteca. Por todo esto le doy un... Me gusta esto: Cargando... Reseña: 7 Wonders Duel, Pantheon Reseña: 7 Wonders Duel Reseña:', 3.309510327490227)]
> Running step 67223341-297b-4748-8320-f6c17ab9ab91. Step input: None
Thought: El texto proporcionado por el usuario parece ser una reseña de un juego de mesa llamado "7 Wonders". No tengo suficiente información para determinar quién diseñó el juego original, pero puedo intentar buscar más información.
Action: get_info_graph_db
Action Input: {'consulta': 'Diseñadores de 7 Wonders'}
Observation: Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'Los' (at char 0), (line:1, col:1)
> Running step 21c4accd-dc26-46b3-9ec2-09b14468971d. Step input: None
Thought: El usuario proporcionó una reseña de un juego de mesa llamado "7 Wonders", pero no tengo suficiente información para determinar quién diseñó el juego original. Es posible que la reseña esté relacionada con una expansión o variante del juego, en lugar de la versión original.
Thought: Revisaré nuevamente la reseña para ver si puedo encontrar alguna pista sobre el diseñador original del juego.
Action: get_info_graph_db
Action Input: {'consulta': 'Antoine Bauza 7 Wonders'}
Observation: Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'Lo' (at char 0), (line:1, col:1)
> Running step dd2876f8-6a6e-48b0-8ccc-cd7e6f6e3518. Step input: None
Thought: El usuario proporcionó una reseña de un juego de mesa llamado "7 Wonders", pero no tengo suficiente información para determinar quién diseñó el juego original. Es posible que la reseña esté relacionada con una expansión o variante del juego, en lugar de la versión original.
Thought: Revisaré nuevamente la reseña para ver si puedo encontrar alguna pista sobre el diseñador original del juego.
Action: get_info_graph_db
Action Input: {'consulta': 'Diseñadores de 7 Wonders Duel'}
Observation: Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'Los' (at char 0), (line:1, col:1)
> Running step 57ab2bff-394d-45e6-af12-584a27c7bf07. Step input: None
Thought: El usuario proporcionó una reseña de un juego de mesa llamado "7 Wonders", pero no tengo suficiente información para determinar quién diseñó el juego original. Es posible que la reseña esté relacionada con una expansión o variante del juego, en lugar de la versión original.
Thought: Intentaré buscar directamente al diseñador Antoine Bauza.
Action: get_info_graph_db
Action Input: {'consulta': '{"properties": {"consulta": {"title": "Antoine Bauza 7 Wonders"}}}}'}
Observation: Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'Lo' (at char 0), (line:1, col:1)
Error al procesar la consulta: Reached max iterations.
```

¿Cuáles son las reglas del juego?

> Running step d4e07dc2-8853-4ac4-bd6a-a16fabf08da4. Step input: ¿Cuáles son las reglas del juego?

Thought: The current language of the user is: español. I need to use a tool to help me answer the question.

Action: get\_info\_vector\_db

Action Input: {'consulta': '¿Cuáles son las reglas del juego?'}

Observation: [({'Los siguientes cambios en las reglas permiten jugar a 8 jugadores. Prepara la partida como para un grupo de 7 jugadores (toda la Era I, la II y la III del juego básico, 9 Gremios y 7 cartas negras en cada mazo de Era). Al comienzo de cada Era, reparte 7 cartas a cada jugador. Cada jugador jugará 6, exactamente como en el juego básico. Nota la inclusión del 8 jugador se ha desarrollado', 1.809148806275946), ('Pero como siempre, la perfección no existe. ¿Cuales son los lunares de este, por otro lado, juego? Como he dicho al principio, en mi opinión no deja de ser un Filler. Uno de los mejores Fillers que he jugado. Pero un Filler a fin de cuentas. Una vez que los jugadores saben jugar, las partidas duran lo que se indica en la caja, en torno a la media hora. Y aunque el juego permita multitud de', 0.9055258309725251), ('por partes, primero explicaré el juego de 3 a 7 jugadores, y, posteriormente, comentaré que modificaciones se realizan para poder jugar 2 jugadores. Conceptos Básicos Hablemos de los conceptos básicos. En primer lugar tenemos las Cartas de, que simbolizan las posibles construcciones que podremos levantar en nuestra civilización. Hay 7 tipos de construcciones, que a la postre darán todas puntos de', 0.8955087297487825), ('semana se juegue a un juego diferente (en plan, si somos 8 los inscritos, cada semana se jugaría a 2 juegos: 4 a uno y los 4 al otro). Y había pensado en que este 7 wonders podría ser uno de los elegidos. ¿Tú qué opinas Iván? ¿Y qué otros juegos ves bien para jugarlos a 4-5? (la idea está por materializar, pero creemos que lo mejor serían partidas de 4). Un saludo!! Responder Miguel Angel', 0.30381708387265544), ('really have a lot of influence on how you approach your own decisions. Short Setup vs. Long Setup - It is a fairly easy game to set up, with the main chore being removing cards from the decks to configure them for your specific player count. But then you can just deal hands and get playing. Easy to Teach vs. Hard to Teach - While 7 Wonders is relatively easy to teach as far as strategy games go,', 0.2936232430185778)]

> Running step 2ba88916-3188-4c8b-a608-af01d8c5cef1. Step input: None

Observation: Error: Could not parse output. Please follow the thought-action-input format. Try again.

> Running step e4b68b11-c70f-4fbd-98fe-25e3812008fb. Step input: None

Thought: The current language of the user is: español. I need to reformat my previous response according to the correct format.

Action: get\_info\_vector\_db

Action Input: {'consulta': '¿Cuáles son las reglas del juego?'}

Observation: [({'Los siguientes cambios en las reglas permiten jugar a 8 jugadores. Prepara la partida como para un grupo de 7 jugadores (toda la Era I, la II y la III del juego básico, 9 Gremios y 7 cartas negras en cada mazo de Era). Al comienzo de cada Era, reparte 7 cartas a cada jugador. Cada jugador jugará 6, exactamente como en el juego básico. Nota la inclusión del 8 jugador se ha desarrollado', 1.809148806275946), ('Pero como siempre, la perfección no existe. ¿Cuales son los lunares de este, por otro lado, juego? Como he dicho al principio, en mi opinión no deja de ser un Filler. Uno de los mejores Fillers que he jugado. Pero un Filler a fin de cuentas. Una vez que los jugadores saben jugar, las partidas duran lo que se indica en la caja, en torno a la media hora. Y aunque el juego permita multitud de', 0.9055258309725251), ('por partes, primero explicaré el juego de 3 a 7 jugadores, y, posteriormente, comentaré que modificaciones se realizan para poder jugar 2 jugadores. Conceptos Básicos Hablemos de los conceptos básicos. En primer lugar tenemos las Cartas de, que simbolizan las posibles construcciones que podremos levantar en nuestra civilización. Hay 7 tipos de construcciones, que a la postre darán todas puntos de', 0.8955087297487825), ('semana se juegue a un juego diferente (en plan, si somos 8 los inscritos, cada semana se jugaría a 2 juegos: 4 a uno y los 4 al otro). Y había pensado en que este 7 wonders podría ser uno de los elegidos. ¿Tú qué opinas Iván? ¿Y qué otros juegos ves bien para jugarlos a 4-5? (la idea está por materializar, pero creemos que lo mejor serían partidas de 4). Un saludo!! Responder Miguel Angel', 0.30381708387265544), ('really have a lot of influence on how you approach your own decisions. Short Setup vs. Long Setup - It is a fairly easy game to set up, with the main chore being removing cards from the decks to configure them for your specific player count. But then you can just deal hands and get playing. Easy to Teach vs. Hard to Teach - While 7 Wonders is relatively easy to teach as far as strategy games go,', 0.2936232430185778)]

> Running step 02edb873-5513-4977-b8aa-9188a8ae0f79. Step input: None

Thought: The output of the tool was not in a format that I could understand, so I need to reformat it.

Action: get\_info\_vector\_db

Action Input: {'consulta': '¿Cuáles son las reglas del juego?'}

Observation: [({'Los siguientes cambios en las reglas permiten jugar a 8 jugadores. Prepara la partida como para un grupo de 7 jugadores (toda la Era I, la II y la III del juego básico, 9 Gremios y 7 cartas negras en cada mazo de Era). Al comienzo de cada Era, reparte 7 cartas a cada jugador. Cada jugador jugará 6, exactamente como en el juego básico. Nota la inclusión del 8 jugador se ha desarrollado', 1.809148806275946), ('Pero como siempre, la perfección no existe. ¿Cuales son los lunares de este, por otro lado, juego? Como he dicho al principio, en mi opinión no deja de ser un Filler. Uno de los mejores Fillers que he jugado. Pero un Filler a fin de cuentas. Una vez que los jugadores saben jugar, las partidas duran lo que se indica en la caja, en torno a la media hora. Y aunque el juego permita multitud de', 0.9055258309725251), ('por partes, primero explicaré el juego de 3 a 7 jugadores, y, posteriormente, comentaré que modificaciones se realizan para poder jugar 2 jugadores. Conceptos Básicos Hablemos de los conceptos básicos. En primer lugar tenemos las Cartas de, que simbolizan las posibles construcciones que podremos levantar en nuestra civilización. Hay 7 tipos de construcciones, que a la postre darán todas puntos de', 0.8955087297487825), ('semana se juegue a un juego diferente (en plan, si somos 8 los inscritos, cada semana se jugaría a 2 juegos: 4 a uno y los 4 al otro). Y había pensado en que este 7 wonders podría ser uno de los elegidos. ¿Tú qué opinas Iván? ¿Y qué otros juegos ves bien para jugarlos a 4-5? (la idea está por materializar, pero creemos que lo mejor serían partidas de 4). Un saludo!! Responder Miguel Angel', 0.30381708387265544), ('really have a lot of influence on how you approach your own decisions. Short Setup vs. Long Setup - It is a fairly easy game to set up, with the main chore being removing cards from the decks to configure them for your specific player count. But then you can just deal hands and get playing. Easy to Teach vs. Hard to Teach - While 7 Wonders is relatively easy to teach as far as strategy games go,', 0.2936232430185778)]

> Running step 66f5fc1a-8897-4565-adb2-329187daf952. Step input: None



Observation: [(('moradas) dan puntos según la condición que indiquen, que normalmente estará relacionada con alguno de los tipos de construcciones disponibles y relacionada con nosotros y nuestros vecinos. Preparación de la Partida En primer lugar, hay que dejar en la caja las cartas que no correspondan al número de jugadores que haya en la partida. A continuación: Se ajusta el numero de cartas necesarios para', 10.756979897187483), ('DEBE descartar una en cada fase de Resolución de Conflictos. - Un jugador con una ficha de Diplomacia está OBLIGADO a jugarla durante la siguiente fase de Resolución de Conflictos (incluso si ese jugador es militarmente más fuerte que los jugadores vecinos). Si solamente hay dos jugadores implicados en una fase de Resolución de Conflictos, se enfrentan entre sí una sola vez, por lo que', 9.25499662669421), ('Pero como siempre, la perfección no existe. ¿Cuales son los lunares de este, por otro lado, juego? Como he dicho al principio, en mi opinión no deja de ser un Filler. Uno de los mejores Fillers que he jugado. Pero un Filler a fin de cuentas. Una vez que los jugadores saben jugar, las partidas duran lo que se indica en la caja, en torno a la media hora. Y aunque el juego permita multitud de', 6.856152065720865), ('P qué ocurre si a través del Mausoleo de Halicar - naso (o de Salomón), un jugador construye una carta de Diplomacia de una Era ya terminada? R El jugador coge igualmente una ficha de Diplo - macia que tendrá que utilizar durante la siguiente fase de Resolución de Conflictos. P Durante una fase de Resolución de Conflictos, si Nerón me da 2 veces 2 monedas por mis victorias, Berenice me da 1 ó', 6.2567394736181665), ('Things to Dislike Falls into Narrow Niche of Complexity - In our experience, 7 Wonders can fall into kind of a weird in-between category where it can have too much complexity for some less experienced gamers, but can also leave some gamers feeling like it isn't complex enough. As long as you have a group that all fall within those bounds they will likely have a great time, but for a game that can', 1.4573149677036503)]

> Running step 15c7d5ff-6d87-4280-b3a5-b93c6a08181e. Step input: None

Thought: The output of the tool contains information about the game's rules and modifications.

Answer: El juego de los Señores del Poder es un juego de estrategia y habilidad que se puede jugar con 2 a 7 jugadores. Cada jugador tiene una ficha de poder que debe utilizar durante la partida para ganar puntos. La partida dura alrededor de la media hora, y el juego permite multitud de estrategias y movimientos.

El juego de los Señores del Poder es un juego de estrategia y habilidad que se puede jugar con 2 a 7 jugadores. Cada jugador tiene una ficha de poder que debe utilizar durante la partida para ganar puntos. La partida dura alrededor de la media hora, y el juego permite multitud de estrategias y movimientos.

Dime una estrategia para ganar en 7 Wonders

> Running step 0125b95e-db6b-4348-a553-14ae7c0899b1. Step input: Dime una estrategia para ganar en 7 Wonders

Thought: The current language of the user is: español. I need to use a tool to help me answer the question.

Action: get\_info\_tabular\_db

Action Input: {'consulta': 'Estrategias de juego en 7 Wonders'}

Observation: Error: invalid syntax (<string>, line 1)

> Running step e5151112-2e91-4272-95b5-44145f0aa7fc. Step input: None

Thought: The input for the tool was not in a valid JSON format. I need to rephrase the Action Input.

Action: get\_info\_tabular\_db

Action Input: {'consulta': 'Estrategias de juego en 7 Wonders'}

Observation: Error: invalid syntax (<string>, line 1)

> Running step 6b9cd5ec-0247-490b-9ce5-960db68a2d6e. Step input: None

Thought: The tool still did not understand the input. I need to rephrase it again.

Action: get\_info\_tabular\_db

Action Input: {'consulta': 'Estrategias de juego en 7 Wonders'}

Observation: Error: invalid syntax (<string>, line 1)

> Running step 5773b5c3-4d45-4225-8c43-ae272234a34c. Step input: None

Thought: The tool is still not working with the input. I need to try a different approach.

Action: get\_info\_graph\_db

Action Input: {'consulta': 'Estrategias de juego en 7 Wonders'}

Observation: Error: Expected {SelectQuery | ConstructQuery | DescribeQuery | AskQuery}, found 'Lo' (at char 0), (line:1, col:1)

> Running step 6a379b63-07e2-4943-8fed-3980b87ef9e2. Step input: None

Thought: The tool is still not working with the input. I need to try a different approach.

Action: get\_info\_vector\_db

Action Input: {'consulta': 'Estrategias de juego en 7 Wonders'}

Observation: [(('porque coge un concepto bastante común en este tipo de juegos, como es el de tener una mano de posibles construcciones e ir levantándolas en nuestra civilización según convenga, con el de que dicho mano vaya rotando por todos los jugadores, pues es aquí donde aparece la gran novedad estratégica que ofrece 7 Wonders. Por otro lado es un juego que «nos aísla» del resto de jugadores. Es decir', 5.446398059896163), ('algo distinto. Si estás decidido a ampliar el 7 Wonders, pues Citie s. Un saludo! Miguel Angel Cabaleiro Fernandez dice: 29/04/2015 a las 08:46 Buenas Iván: Pues yo la verdad es que le tengo muchas ganas a este juego. Gran parte de mis amigos juegan al 7 wonders y próximamente vamos a organizar una liguilla con emparejamientos semanales, con la premisa de que cada semana se juegue a un juego', 5.143475764211233), ('strategic experience with a game length that remains short even with larger player counts. Great Ratio of Depth to Game Length - And it is really the combination of those interesting decisions and quick play time that give 7 Wonders an outstanding ratio of depth to game length. For a larger group that is still looking for a more strategic experience, 7 Wonders gives them great bang for their', 4.853450921263222), ('Watch Extended Final Thoughts Is It For You? If you want your interaction to be a little more direct or prefer strategic planning over tactical decisions, 7 Wonders might not be the best choice. But if you want a strategy game that can support higher player counts while still playing quickly, enjoy figuring out the best card out of a hand of options, and like building up your own tableau of cards', 2.4583376014906584), ('them great bang for their buck. Variety of Strategic Routes - And the game remains interesting after many plays, largely because there is a lot of room for your strategy to play out differently. There are some extreme strategies such as spamming blue point cards, green technologies, or dominating military battles, but there are also a number of hybrid strategies that could be victorious. If you', 0.9430249292514168)]

Error al procesar la consulta: Reached max iterations.

## 2.4. Herramientas adicionales utilizadas

*Ollama*. Es un entorno que facilita la descarga y ejecución de modelos Llama de forma local (o en un servidor), sin necesidad de usar una API en la nube. Permite correr inferencias con modelos grandes y ofrece una interfaz de línea de comandos y endpoints HTTP para interactuar con el modelo. En el proyecto se instala ollama, se descarga un modelo *Llama* con *ollama pull*, y finalmente se configura un objeto *Ollama* en Python para comunicarse con el servidor local de *Ollama*. <http://ollama.com>

*Llama-index*. Es un framework que permite crear estructuras de datos para conectar LLMs con fuentes externas, manejar agentes conversacionales basados en el patrón ReAct (Reason + Act), e implementar herramientas personalizadas que el agente puede invocar. [https://github.com/run-llama/llama\\_index](https://github.com/run-llama/llama_index)

*Function Tools*. Este componente permite envolver funciones de Python en “herramientas” que el agente pueda llamar. Cada herramienta (por ejemplo, *get\_info\_graph\_db*) tiene un “nombre”, una “descripción” y la implementación real de la función. [https://gpt-index.readthedocs.io/en/latest/how\\_to/custom\\_react\\_tools.html](https://gpt-index.readthedocs.io/en/latest/how_to/custom_react_tools.html)

*ReactAgent*. El núcleo del Ejercicio 2, implementa la arquitectura de Reason + Act, en la que el modelo razona en pasos intermedios (Thought), decide una Acción a tomar (por ejemplo, “usar la herramienta X”), se queda esperando la Observación (la salida de dicha herramienta), y con base en la observación, construye la respuesta final al usuario. [https://gpt-index.readthedocs.io/en/latest/guides/advanced\\_usage/react\\_agent.html](https://gpt-index.readthedocs.io/en/latest/guides/advanced_usage/react_agent.html)