**What is Query Routing?**

Query Routing is about giving our RAG app the power of decision-making. Query Routing is a technique that takes the query from the user and uses it to make a decision on the next action to take, from a list of predefined choices.

Query Routing is a module in our Advanced RAG architecture. It is usually found after any query rewriting or guardrails. It analyses the input query and it decides the best tool to use from a list of predefined actions. The actions are usually retrieving context from one or many data sources. It could also decide to use a different index for a data source (like parent-child retrieval). Or it could even decide to search for context on the Internet.

**Example**:

**Normal conversation**: “Hello” or how are you to day?

**Real time question**: “What time is it”

**FAQ**: “How to”, What is?, …

For the above types, we don’t need to push it into retrieval system

**Which are the choices for the Query Router?**

We have to define the choices that the Query Router can take beforehand. We must first implement each of the different strategies, and accompany each one with a nice description. It is very important that the description explains in detail what each strategy does, since this description will be what our router will base its decision on.

The choices a Query Router takes can be the following:

Retrieval from different data sources

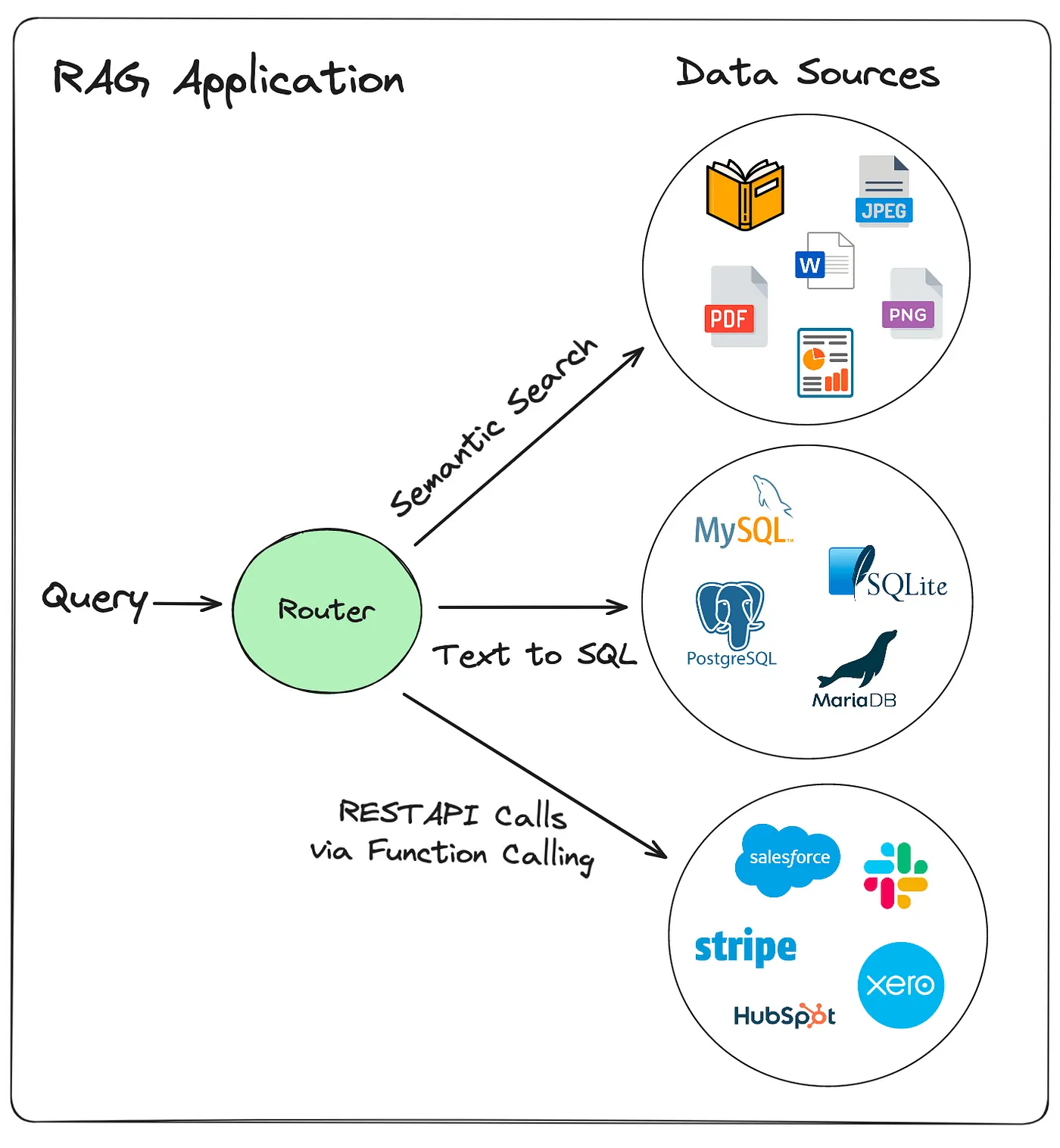
We can catalog multiple data sources that contain information on different topics. We might have a data source that contains information about a product that the user has questions about. And another data source with information about our return policies, etc. Instead of looking for the answers for the user’s questions in all data sources, the query router can decide which data source to use based on the user query and the data source description.

Data sources can be text stored in vector databases, regular databases, graph databases, etc.

Retrieval from different indexes

Query Routers can also choose to use a different index for the same data source.

For example, we could have an index for keyword based search and another for semantic search using vector embeddings. The Query Router can decide which of the two is best for getting the relevant context for answering the question, or maybe use both of them at the same time and combine the contexts from both.

We could also have different indexes for different retrieval strategies. For example, we could have a retrieval strategy based on summaries, or a sentence window retrieval strategy, or a parent-child retrieval strategy. The Query Router can analyze the specificity of the question and decide which strategy is best to use to get the best context

**Semantic Router**

This router type leverages embeddings and similarity searches to select the best route to traverse.

Each route has a set of example queries associated with it, that become embedded and stored as vectors. The incoming query gets embedded also, and a similarity search is done against the other sample queries from the router. The route which belongs to the query with the closest match gets selected.

Reference

[Routing in RAG-Driven Applications | by Sami Maameri | Towards Data Science](https://towardsdatascience.com/routing-in-rag-driven-applications-a685460a7220)

[Build an Advanced RAG App: Query Routing](https://www.ruxu.dev/articles/ai/query-routing/#:~:text=Query%20Routing%20is%20a%20technique%20that%20takes%20the,usually%20found%20after%20any%20query%20rewriting%20or%20guardrails.)

(\*) [Mastering RAG: Advanced Methods to Enhance Retrieval-Augmented Generation | by Samar Singh | Medium](https://medium.com/@samarrana407/mastering-rag-advanced-methods-to-enhance-retrieval-augmented-generation-4b611f6ca99a)