PGVector Vector Store node documentation

# PGVector Vector Store node#

PGVector is an extension of Postgresql. Use this node to interact with the PGVector tables in your Postgresql database. You can insert documents into a vector table, get documents from a vector table, retrieve documents to provide them to a retriever connected to a chain, or connect directly to an agent as a tool.

On this page, you'll find the node parameters for the PGVector node, and links to more resources.

Credentials

You can find authentication information for this node here.

Parameter resolution in sub-nodes

Sub-nodes behave differently to other nodes when processing multiple items using an expression.

Most nodes, including root nodes, take any number of items as input, process these items, and output the results. You can use expressions to refer to input items, and the node resolves the expression for each item in turn. For example, given an input of five name values, the expression {{ $json.name }} resolves to each name in turn.

name

{{ $json.name }}

In sub-nodes, the expression always resolves to the first item. For example, given an input of five name values, the expression {{ $json.name }} always resolves to the first name.

name

{{ $json.name }}

## Node usage patterns#

You can use the PGVector Vector Store node in the following patterns.

### Use as a regular node to insert and retrieve documents#

You can use the PGVector Vector Store as a regular node to insert or get documents. This pattern places the PGVector Vector Store in the regular connection flow without using an agent.

You can see an example of this in scenario 1 of this template (the template uses the Supabase Vector Store, but the pattern is the same).

### Connect directly to an AI agent as a tool#

You can connect the PGVector Vector Store node directly to the tool connector of an AI agent to use vector store as a resource when answering queries.

Here, the connection would be: AI agent (tools connector) -> PGVector Vector Store node.

### Use a retriever to fetch documents#

You can use the Vector Store Retriever node with the PGVector Vector Store node to fetch documents from the PGVector Vector Store node. This is often used with the Question and Answer Chain node to fetch documents from the vector store that match the given chat input.

An example of the connection flow (the linked example uses Pinecone, but the pattern is the same) would be: Question and Answer Chain (Retriever connector) -> Vector Store Retriever (Vector Store connector) -> PGVector Vector Store.

### Use the Vector Store Question Answer Tool to answer questions#

Another pattern uses the Vector Store Question Answer Tool to summarize results and answer questions from the PGVector Vector Store node. Rather than connecting the PGVector Vector Store directly as a tool, this pattern uses a tool specifically designed to summarizes data in the vector store.

The connections flow (the linked example uses the Simple Vector Store, but the pattern is the same) in this case would look like this: AI agent (tools connector) -> Vector Store Question Answer Tool (Vector Store connector) -> Simple Vector store.

## Node parameters#

### Operation Mode#

This Vector Store node has four modes: Get Many, Insert Documents, Retrieve Documents (As Vector Store for Chain/Tool), and Retrieve Documents (As Tool for AI Agent). The mode you select determines the operations you can perform with the node and what inputs and outputs are available.

In this mode, you can retrieve multiple documents from your vector database by providing a prompt. The prompt will be embedded and used for similarity search. The node will return the documents that are most similar to the prompt with their similarity score. This is useful if you want to retrieve a list of similar documents and pass them to an agent as additional context.

Use Insert Documents mode to insert new documents into your vector database.

Use Retrieve Documents (As Vector Store for Chain/Tool) mode with a vector-store retriever to retrieve documents from a vector database and provide them to the retriever connected to a chain. In this mode you must connect the node to a retriever node or root node.

Use Retrieve Documents (As Tool for AI Agent) mode to use the vector store as a tool resource when answering queries. When formulating responses, the agent uses the vector store when the vector store name and description match the question details.

### Get Many parameters#

• Table name: Enter the name of the table you want to query.

• Prompt: Enter your search query.

• Limit: Enter a number to set how many results to retrieve from the vector store. For example, set this to 10 to get the ten best results.

10

### Insert Documents parameters#

• Table name: Enter the name of the table you want to query.

### Retrieve Documents parameters (As Vector Store for Chain/Tool)#

• Table name: Enter the name of the table you want to query.

### Retrieve Documents (As Tool for AI Agent) parameters#

• Name: The name of the vector store.

• Description: Explain to the LLM what this tool does. A good, specific description allows LLMs to produce expected results more often.

• Table Name: Enter the PGVector table to use.

• Limit: Enter how many results to retrieve from the vector store. For example, set this to 10 to get the ten best results.

10

## Node options#

### Collection#

A way to separate datasets in PGVector. This creates a separate table and column to keep track of which collection a vector belongs to.

• Use Collection: Select whether to use a collection (turned on) or not (turned off).

• Collection Name: Enter the name of the collection you want to use.

• Collection Table Name: Enter the name of the table to store collection information in.

### Column Names#

The following options specify the names of the columns to store the vectors and corresponding information in:

• ID Column Name

• Vector Column Name

• Content Column Name

• Metadata Column Name

### Metadata Filter#

Available in Get Many mode. When searching for data, use this to match with metadata associated with the document.

This is an AND query. If you specify more than one metadata filter field, all of them must match.

AND

When inserting data, the metadata is set using the document loader. Refer to Default Data Loader for more information on loading documents.

## Templates and examples#

Browse PGVector Vector Store integration templates, or search all templates

## Related resources#

Refer to LangChain's PGVector documentation for more information about the service.

View n8n's Advanced AI documentation.

## Self-hosted AI Starter Kit#

New to working with AI and using self-hosted n8n? Try n8n's self-hosted AI Starter Kit to get started with a proof-of-concept or demo playground using Ollama, Qdrant, and PostgreSQL.