

warning "Atualização de backend (outubro de 2026)" A inferência do GitHub Models foi descontinuada em 30 de julho de 2026. As referências abaixo são mantidas apenas como contexto histórico enquanto este lab é migrado. Use o Lab 013 ? Backends de LLM gratuitos para configurar o Foundry Local ou o Ollama antes de continuar.

Lab 031: pgvector Semantic Search on Azure

Nível: L300 Trilha: RAG Tempo: ~60 min ? Custo: ~\$15/mês (PostgreSQL Flexible Server Burstable B1ms)

warning "Assinatura do Azure necessária" Este lab implanta o Azure PostgreSQL Flexible Server. ? Guia de pré-requisitos

O que Você Vai Aprender

Implantar o Azure PostgreSQL Flexible Server com pgvector habilitado em um clique

Conectar-se com segurança usando SSL e regras de firewall Migrar o pipeline RAG do Lab 022

para o Azure Usar o Azure AI Services para embeddings (ou continuar usando GitHub Models ? gratuito) Expor o banco vetorial como uma ferramenta MCP para agentes

Implantar Infraestrutura

Opção A ? Implantar no Azure (um clique)

Deploy to Azure

Isso implanta: PostgreSQL Flexible Server (Burstable B1ms, ~\$15/mês) Extensão pgvector

habilitada Banco de dados ragdb pré-criado Regra de firewall para serviços do Azure

Parâmetros que você definirá: administratorLoginPassword ? sua senha do banco de dados (anote-a!) location ? escolha uma região próxima a você

Opção B ? Azure CLI (Bicep)

[diagram or code sample]

? Arquivos de Apoio

note "Baixe estes arquivos antes de iniciar o lab" Salve todos os arquivos em uma pasta lab-031/ no seu diretório de trabalho.

Arquivo | Descrição | Download | 001_init.sql | Migração do banco de dados | ? Download | Exercício do Lab

Passo 1: Definir variáveis de ambiente de conexão

Após a implantação, salve seus detalhes de conexão:

[diagram or code sample]

Passo 2: Conectar e verificar o pgvector

[diagram or code sample]

Passo 3: Criar esquema

[diagram or code sample]

Passo 4: Ingerir o conjunto de dados de exemplo

Baixe e ingira os dados de exemplo do OutdoorGear:

[diagram or code sample]

Reutilize o contrato de chunking e embedding do Lab 022 e substitua somente o adaptador em memória pelo Azure Database for PostgreSQL:

[diagram or code sample]

Passo 5: Expor como ferramenta MCP

Encapsule a função de busca como uma ferramenta MCP para que qualquer agente possa consultar sua base de conhecimento:

[diagram or code sample]

Adicione ao .vscode/mcp.json: [diagram or code sample]

Limpeza

[diagram or code sample]

? Verificação de Conhecimento

question "Q1 (Execute o Lab): Após executar 001_init.sql no seu banco de dados, quantas linhas existem na tabela products? Execute SELECT COUNT(*) FROM products; para verificar." Execute a migração e depois consulte a tabela.

??? success "? Revelar Resposta" 7 linhas

A migração insere 7 produtos OutdoorGear: P001 (TrailBlazer Tent 2P), P002 (Summit Dome 4P), P003 (TrailBlazer Solo), P004 (ArcticDown -20°C), P005 (SummerLight +5°C), P006 (Osprey Atmos 65L), P007 (DayHiker 22L). Execute SELECT id, name, category FROM products ORDER BY id; para ver todos.

question "Q2 (Execute o Lab): Que tipo de índice é criado na tabela product_embeddings, e qual coluna ele indexa?"

Abra lab-031/migrations/001_init.sql e encontre a instrução CREATE INDEX.

??? success "? Revelar Resposta" Índice IVFFlat na coluna embedding

A migração cria: CREATE INDEX ON product_embeddings USING ivfflat (embedding vector_cosine_ops) WITH (lists = 100);

IVFFlat (Inverted File with Flat quantization) é um índice de vizinho mais próximo aproximado ? é mais rápido que a busca exata, mas sacrifica uma pequena quantidade de recall. O parâmetro lists = 100 define o número de células de Voronoi para agrupamento. vector_cosine_ops significa que as distâncias são calculadas usando similaridade de cosseno.

question "Q3 (Múltipla Escolha): Você executa SELECT * FROM search_products_by_vector(\$1::vector, 3) e obtém 3 resultados. O que os resultados representam?"

- A) Os 3 produtos inseridos mais recentemente - B) Os 3 produtos com o maior preço - C) Os 3 produtos cujos vetores de embedding são mais similares (menor distância de cosseno) ao vetor de consulta - D) 3 produtos selecionados aleatoriamente do banco de dados

??? success "? Revelar Resposta" Correto: C

A função search_products_by_vector() realiza uma busca de vizinho mais próximo aproximado (ANN) usando o índice IVFFlat. Ela calcula a distância de cosseno entre o vetor de consulta e todos os embeddings de produtos armazenados, e então retorna os k produtos com a menor distância (= maior similaridade semântica). O resultado representa os produtos semanticamente mais relevantes para a consulta do usuário.

Próximos Passos

Row Level Security para agentes multi-tenant: ? Lab 032 ? Row Level Security RAG Agêntico sobre isso: ? Lab 026 ? Agêntico RAG Pattern

Kit de apoio

O build do site gera downloads estáveis em PDF e ZIP. O ZIP contém o guia do lab e todos os arquivos de apoio declarados para este lab.

Baixar PDF gerado Baixar ZIP gerado