

# **PROJETO DE PESQUISA**

**Tadeu Gomes Teixeira**

Agente Inteligente Baseado em LLM para Consulta em  
Linguagem Natural de Dados da ANTAQ

## **1. IDENTIFICAÇÃO DO PROJETO**

**Título:** Agente Inteligente Baseado em LLM para Consulta em Linguagem Natural de Dados da ANTAQ

**Área de Conhecimento:** Inteligência Artificial

**Subárea:** Processamento de Linguagem Natural / Engenharia de Prompt / Agentes de IA

**Linha de Pesquisa:** Sistemas de Informação Inteligentes e Interfaces Conversacionais

## **2. INTRODUÇÃO E CONTEXTUALIZAÇÃO**

### **2.1 Contextualização**

A Agência Nacional de Transportes Aquaviários (ANTAQ) constitui o órgão regulador do setor de transporte aquaviário no Brasil, desempenhando papel fundamental na coleta, processamento e disponibilização de dados estatísticos sobre a movimentação de cargas nos portos brasileiros. Estes dados assumem importância estratégica em múltiplas dimensões da gestão portuária e do planejamento logístico nacional.

Primeiramente, constituem subsídio essencial para a formulação de políticas públicas de transporte e logística, permitindo que decisores governamentais fundamentem suas escolhas em evidências empíricas sobre padrões de movimentação de cargas, tendências de comércio exterior e dinâmicas regionais do setor portuário. Igualmente relevante, tais informações servem de base para o planejamento estratégico do setor portuário, subsidiando decisões sobre investimentos em infraestrutura, expansão de terminais e modernização de instalações. As análises de competitividade e eficiência portuária também dependem criticamente destes dados, viabilizando comparações entre diferentes portos, identificação de gargalos operacionais e avaliação de desempenho ao longo do tempo. Por fim, a comunidade acadêmica utiliza extensivamente estas estatísticas em pesquisas sobre comércio exterior e logística, produzindo conhecimento científico que retroalimenta tanto a formulação de políticas quanto a gestão empresarial do setor.

No entanto, apesar de sua relevância estratégica, o acesso efetivo a estas informações encontra-se limitado por barreiras técnicas significativas que restringem sua utilização a um grupo relativamente restrito de especialistas. A primeira dessas barreiras consiste na necessidade de conhecimento técnico da linguagem SQL (*Structured Query Language*), requisito que exclui imediatamente usuários sem formação em tecnologia da informação ou áreas correlatas. O aproveitamento pleno dessas informações frequentemente requer o uso de ferramentas especializadas de *Business Intelligence*, software complexos que demandam treinamento específico e investimento financeiro considerável. Este conjunto de barreiras técnicas acumuladas cria, na prática, uma assimetria informacional que contradiz o princípio de transparência inerente aos dados públicos governamentais.

## 2.2 Problema de Pesquisa

A barreira técnica para acesso aos dados da ANTAQ limita sua utilização por:

- Decisores políticos que não possuem conhecimento técnico em SQL.
- Pesquisadores de áreas não técnicas (economia, relações internacionais, logística).
- Empresários do setor que necessitam de informações rápidas para tomada de decisão.
- Jornalistas que precisam acessar dados para embasar reportagens.

Diante disso, o problema central de pesquisa é: *Como tornar acessíveis dados governamentais complexos para usuários não técnicos através de interfaces em linguagem natural?*

## 2.3 Justificativa

A justificativa para este projeto de pesquisa fundamenta-se em três dimensões complementares que convergem para demonstrar sua pertinência tanto no âmbito social quanto científico e econômico.

Do ponto de vista social, a pesquisa contribui diretamente para a democratização do acesso a dados públicos governamentais, um imperativo em sociedades democráticas contemporâneas. Atualmente, a barreira técnica imposta pela necessidade de conhecimento em SQL e ferramentas de *Business Intelligence* cria uma divisão informacional que privilegia especialistas técnicos em detrimento de outros atores sociais igualmente legítimos no acesso a informações públicas. Ao desenvolver uma interface em linguagem natural, o projeto remove essa barreira, permitindo que decisores políticos, jornalistas, pesquisadores de áreas não técnicas e cidadãos em geral possam exercer plenamente seu direito ao acesso à informação. Esta democratização fortalece os princípios de transparência e *accountability* no setor de transportes, possibilitando que a sociedade civil acompanhe, fiscalize e participe de discussões informadas sobre políticas públicas portuárias. Ademais, ao facilitar o acesso a evidências empíricas, o sistema promove uma cultura de tomada de decisões baseada em dados, essencial para políticas públicas efetivas e responsáveis.

Na dimensão acadêmica, a pesquisa posiciona-se na fronteira do conhecimento em múltiplas frentes. Primeiro, contribui para o estado da arte em *Text-to-SQL*, um problema fundamental em processamento de linguagem natural que, apesar dos avanços recentes com *Large Language Models*, permanece não resolvido para cenários empresariais reais. A aplicação de técnicas de *Retrieval-Augmented Generation* em um domínio específico e altamente especializado como os dados da ANTAQ representa avanço significativo, pois testa a capacidade de RAG em lidar com *schemas* complexos, terminologia técnica de domínio e requisitos de validação metodológica oficial. Adicionalmente, a pesquisa sobre engenharia de *prompt* para agentes de IA em contextos onde precisão e confiabilidade são críticas expande o conhecimento sobre como estruturar *prompts* que não apenas gerem SQL sintaticamente correto, mas também semanticamente alinhado com requisitos de negócio complexos. A integração de arquiteturas agênticas com RAG e

decomposição de tarefas em domínio governamental brasileiro constitui contribuição original inexplorada na literatura internacional.

Economicamente, o projeto apresenta potencial de impacto significativo através de três mecanismos principais. Primeiro, a redução de custos de treinamento em SQL e ferramentas de *Business Intelligence* representa economia direta para organizações públicas e privadas do setor portuário, que atualmente investem recursos substanciais na capacitação técnica de profissionais. Segundo, a agilidade proporcionada na obtenção de *insights* para decisões empresariais pode traduzir-se em vantagem competitiva para empresas do setor, permitindo respostas mais rápidas a mudanças de mercado, identificação oportuna de oportunidades logísticas e otimização de operações baseada em análise de dados históricos. Terceiro, ao facilitar o acesso a informações estratégicas sobre eficiência portuária, movimentação de cargas e padrões de comércio exterior, o sistema contribui para a melhoria sistêmica da eficiência do setor portuário brasileiro, com potenciais ganhos em produtividade, redução de custos logísticos e fortalecimento da competitividade internacional dos portos nacionais. Estes benefícios econômicos transcendem ganhos individuais de organizações para impactar positivamente a economia como um todo, dado o papel estratégico do setor portuário no comércio exterior brasileiro.

### **3. OBJETIVOS**

#### **3.1 Objetivo Geral**

Desenvolver e validar um agente de inteligência artificial capaz de interpretar consultas em linguagem natural sobre dados de transporte aquaviário e convertê-las automaticamente em *queries* SQL executáveis no *BigQuery*, retornando resultados formatados e *insights* acionáveis.

#### **3.2 Objetivos Específicos**

##### **1. Levantamento de Requisitos**

- Criar um banco de dados com *schemas* no *BigQuery* a partir dos dados da ANTAQ;
- Catalogar os tipos de consultas mais frequentes aos dados;

##### **2. Desenvolvimento do Sistema**

- Implementar arquitetura baseada em *LangGraph* para orquestração do agente;
- Criar módulo de metadados para descrição semântica do esquema;
- Desenvolver sistema RAG para recuperação de exemplos relevantes;
- Implementar camada de segurança e validação de SQL;

##### **3. Validação e Avaliação**

- Realizar testes com usuários reais de diversos perfis.

## 4. REFERENCIAL TEÓRICO

A tradução automática de linguagem natural para SQL representa uma das fronteiras mais promissoras da inteligência artificial aplicada a bancos de dados. Este referencial teórico examina a evolução do campo *Text-to-SQL* desde os sistemas pioneiros dos anos 1970 até as arquiteturas agênticas contemporâneas baseadas em *Large Language Models*, identificando que a convergência entre decomposição de tarefas, *Retrieval-Augmented Generation* e orquestração multi-agente constitui o paradigma mais efetivo para sistemas de próxima geração.

### 4.1 Dos sistemas pioneiros à era neural

O campo de interfaces de linguagem natural para bancos de dados (NLIDBs) originou-se com sistemas como LUNAR (Woods, 1973), desenvolvido para consultas sobre análises de amostras lunares das missões Apollo utilizando *Augmented Transition Networks*. Como observam Androutsopoulos, Ritchie e Thanisch (1995), estes primeiros sistemas eram construídos tendo um banco de dados específico em mente, não podendo ser facilmente adaptados para diferentes domínios.

O sistema LADDER (Hendrix et al., 1978) representou o primeiro avanço significativo ao introduzir gramáticas semânticas configuráveis para diferentes bancos de dados. A década de 1980 consolidou abordagens baseadas em representações lógicas intermediárias, com sistemas como MASQUE utilizando Prolog para converter perguntas em formas lógicas antes da geração SQL. O framework PRECISE (Popescu et al., 2003) marcou o ápice desta era, combinando tokenização semântica com *lookup* em léxicos e WordNet para alcançar independência de banco de dados sem treinamento.

O ano de 2017 constitui um divisor de águas com a publicação de Seq2SQL (Zhong, Xiong & Socher, 2017), primeira rede neural profunda para tradução de linguagem natural em SQL. Utilizando *policy-based reinforcement learning* com execução de *queries* no *loop* de treinamento, o sistema elevou a *execution accuracy* de 35.9% para 59.4% comparado a modelos seq2seq tradicionais. Simultaneamente, o dataset WikiSQL estabeleceu o primeiro *benchmark* de grande escala com 80.654 pares question-SQL distribuídos em 24.241 tabelas.

O *benchmark* Spider (Yu et al., 2018), descrito pelos autores como o primeiro dataset complexo e *cross-domain* para *semantic parsing* e *text-to-SQL*, introduziu 200 databases em 138 domínios com *queries* envolvendo JOINs, agregações e subqueries aninhadas. A estratificação em níveis de dificuldade (Easy, Medium, Hard, Extra Hard) revelou que modelos falham progressivamente em *queries* compostos.

O framework RAT-SQL (Wang et al., 2020) representou *breakthrough* ao introduzir *relation-aware self-attention* para modelagem explícita de relações entre questão e *schema* como grafo, atingindo 65.6% *exact match accuracy* no Spider com BERT. A arquitetura demonstrou que *schema encoding* e *schema linking* constituem desafios fundamentais que transcendem mero aumento de capacidade do modelo.

### 4.2 A era dos LLMS

O BIRD (Li et al., 2023), apresentado no NeurIPS 2023, adicionou realismo crítico: databases com dados sujos, requisitos de conhecimento externo, e a primeira métrica de eficiência (*Valid Efficiency Score*). Os resultados revelaram *gap* substancial: enquanto humanos atingem 92.96% *execution accuracy*, GPT-4 alcança apenas 54.89%.

O Spider 2.0 (Lei et al., 2024) expõe limitações dramáticas em cenários empresariais reais: databases com 700-3000 colunas, múltiplos dialetos SQL (*BigQuery*, *Snowflake*, *PostgreSQL*), e necessidade de navegação em documentação externa. Os resultados são alarmantes: GPT-4o atinge apenas 10.1%, o1-preview 17.1%, comparados a 86.6% no Spider original. Este *gap* evidencia que o problema permanece fundamentalmente não resolvido para contextos empresariais reais.

A emergência dos *Large Language Models* transformou a abordagem dominante de *supervised fine-tuning* para *in-context learning* (ICL). Como observam Zhu et al. (2024), com o desenvolvimento dos LLMs, o padrão de *text-to-SQL* sofreu mudanças significativas, especialmente em capacidades de compreensão de linguagem natural. Desde 2023, a maioria das soluções *top-performing* nos leaderboards são baseadas em LLMs com *prompting* sofisticado.

Os LLMs demonstraram capacidades emergentes cruciais: schema linking automático, raciocínio composicional para queries com JOINs e subqueries, generalização cross-domain, e self-debugging. O GPT-4 com técnicas avançadas atinge 85.3% *execution accuracy* no Spider test set (DIN-SQL; Pourreza & Rafie, 2023), superando modelos fine-tuned anteriores.

A literatura revela *trade-offs* significativos entre as abordagens. *Fine-tuning* oferece maior controle, menor latência e privacidade (processamento local), porém requer investimento computacional substancial e dados de domínio. *Prompting* proporciona flexibilidade e *deployment* rápido, mas com custos de inferência elevados e dependência de APIs externas.

O projeto CodeS (Li et al., 2024, SIGMOD) demonstrou que modelos *open-source* especializados (1B-15B parâmetros) podem aproximar performance de GPT-4 sendo 10-100x menores. CodeS-15B atinge 83.6% no Spider test e aproximadamente 55% no BIRD, representando caminho promissor para democratização de *Text-to-SQL* sem dependência de APIs proprietárias.

Qu et al. (2024) identificaram três classes de *hallucinations* em SQL: schema-based (referência a elementos inexistentes), *logic-based* (SQL sintaticamente correto mas semanticamente incorreto), e *content-based* (valores fabricados). Estas limitações fundamentais motivam a necessidade de arquiteturas que combinem LLMs com mecanismos de validação e correção.

O estudo sistemático de DAIL-SQL (Gao et al., 2024, VLDB) estabeleceu que o formato DDL com CREATE TABLE combinado com exemplos de valores (SELECT 3) oferece melhor *trade-off* entre *accuracy* e eficiência de *tokens*. A pesquisa comparou cinco formatos de representação, demonstrando que *Code Representation* supera alternativas em consistência.

*Schema pruning* emerge como técnica essencial para databases grandes. Abordagens incluem similaridade semântica via *embeddings*, combinação BM25 + Dense Retrieval, e bidirectional schema linking. O PET-SQL (Li et al., 2024) introduz PreSQL-based schema linking, gerando SQL preliminar para identificar entidades mencionadas, reduzindo significativamente o contexto necessário.

A estratégia DAIL Selection representa estado da arte, combinando *question similarity* com *query/skeleton similarity* para seleção de exemplos. O método atinge 86.6% execution accuracy no Spider com *self-consistency*, sendo significativamente mais eficiente em tokens (~1600 por query) que abordagens Full-Information.

A comparação com DIN-SQL (Pourreza & Rafiei, 2023) revela complementaridade: enquanto DAIL-SQL otimiza seleção de exemplos, DIN-SQL foca em decomposição de tarefas. Ambos compartilham *insight* fundamental: LLMs beneficiam-se de estruturação explícita do problema.

Tai et al. (2023, EMNLP) conduziram estudo sistemático de Chain-of-Thought para Text-to-SQL, reportando +5.2 pontos absolutos no Spider dev e +6.5 pontos no Spider Realistic. O framework Divide-and-Prompt (Liu & Tan, 2023) propõe três variantes: Clause-by-Clause (construção incremental), Schema Linking first, e Generate-and-Refine (iterativo). Cada padrão é adequado para diferentes perfis de complexidade de queries.

### 4.3 Retrieval-Augmented Generation

O paper seminal de Lewis et al. (2020, NeurIPS) definiu RAG como paradigma que combina memória paramétrica (conhecimento nos parâmetros) com memória não-paramétrica (índice vetorial externo). A arquitetura original utilizava Dense Passage Retriever (DPR) baseado em BERT como retriever e BART como generator, propondo duas formulações: RAG-Sequence (mesmos documentos para toda sequência) e RAG-Token (documentos diferentes por token).

RAG endereça problemas fundamentais de LLMs: knowledge grounding, redução de hallucinations, atualização de conhecimento sem retreino, e rastreabilidade de fontes. Para Text-to-SQL, estas propriedades são críticas: schemas mudam, terminologia de domínio evolui, e queries históricas validadas constituem conhecimento valioso.

O survey de Gao et al. (2024) propõe taxonomia em quatro estágios evolutivos: Naive RAG (pipeline simples retrieve-then-read), Advanced RAG (técnicas de pré-retrieval como query rewriting, retrieval aprimorado com hybrid search, e pós-retrieval com re-ranking), Modular RAG (componentes intercambiáveis como RAPTOR e RAFT), e Agentic RAG (integração com padrões agênticos como reflection e planning).

Self-RAG (Asai et al., 2023, ICLR 2024) introduziu innovation significativa: reflection tokens para auto-avaliação. Quatro tokens especiais controlam o comportamento: [Retrieve] (decide necessidade de retrieval), [ISREL] (avalia relevância), [ISSUP]

(verifica suporte no contexto), e [ISUSE] (avalia utilidade). O sistema supera ChatGPT e Llama2-chat demonstrando que retrieval adaptativo (sob demanda) supera retrieval fixo.

A literatura identifica três aplicações principais de RAG para Text-to-SQL: (1) Retrieval de exemplos similares via embedding de queries históricas; (2) Retrieval de documentação de schema incluindo descrições de tabelas, colunas, relacionamentos e glossários de negócio; (3) Cache semântico de queries para reutilização de pares (pergunta, SQL) validados. Técnicas de entity masking (substituir entidades específicas por placeholders genéricos) melhoram generalização em aproximadamente 7% accuracy.

#### 4.4 Arquiteturas Agênticas

Um agente de IA é definido como sistema autônomo capaz de perceber ambiente, interpretar dados, raciocinar e executar ações para alcançar objetivos sem intervenção humana explícita. A arquitetura típica compreende: módulo de percepção (coleta de informações), módulo de memória (episódica e semântica), módulo de planejamento (decomposição de tarefas), e módulo de ação/ferramentas.

O paradigma ReAct (Yao et al., 2022, ICLR 2023), acrônimo de Reasoning + Acting, estabeleceu framework influente onde reasoning traces e ações intercalam-se: Thought → Action → Observation → Thought. Os benefícios sobre Chain-of-Thought puro incluem grounding em fontes externas, trajetórias interpretáveis, e capacidade de atualização de conhecimento. Resultados reportados mostram superioridade em HotpotQA, Fever, ALFWorld e WebShop.

LangGraph, inspirado por Pregel (Google) e Apache Beam, implementa state machines para agentes através de arquitetura baseada em grafos. Nodes representam estados ou funções (chamadas LLM, tool use), edges conectam nodes (diretas ou condicionais), e state persiste como objeto compartilhado. O diferencial crítico sobre frameworks anteriores é suporte nativo a ciclos, essencial para arquiteturas agênticas onde iteração é fundamental.

Comparado ao LangChain tradicional (paradigma chain-first linear), LangGraph oferece controle low-level sobre fluxo de execução, persistência de estado first-class, e design explícito para workflows complexos. Outros frameworks relevantes incluem AutoGPT (loops autônomos), BabyAGI (arquitetura task-driven), CrewAI (equipes com papéis definidos), e AutoGen (Microsoft, conversational multi-agent).

O framework MAC-SQL (Wang et al., 2024) representa implementação exemplar de arquitetura multi-agente para Text-to-SQL, com três agentes especializados: Selector Agent (comprime databases grandes via schema pruning), Decomposer Agent (quebra questions complexas em sub-problemas com Chain-of-Thought), e Refiner Agent (valida SQL via execução e corrige erros). Resultados demonstram 59.59% execution accuracy no BIRD test com GPT-4, evidenciando que especialização de agentes supera approaches monolíticos.

#### 4.5 Estado da Arte e Trabalhos Recentes

DIN-SQL (Pourreza & Rafiei, 2023, NeurIPS) foi pioneiro em demonstrar que decomposed in-context learning supera métodos fine-tuned. O pipeline de quatro etapas (Schema Linking → Query Classification → SQL Generation → Self-

Correction) atinge 85.3% execution accuracy no Spider test. Análise por dificuldade mostra ganhos progressivos: marginal em Easy, +8% em Medium, +12% em Hard, +15% em Extra Hard, evidenciando que decomposição é particularmente valiosa para queries complexas.

C3 (Dong et al., 2023) propôs framework zero-shot sistemático com três componentes: Clear Prompting (layout otimizado), Calibration with Hints (correção de vieses como uso excessivo de LEFT JOIN), e Consistent Output (self-consistency). Atinge 82.3% execution accuracy no Spider test sem exemplos, demonstrando que prompt engineering sofisticado pode aproximar few-shot methods, reduzindo significativamente custos de inferência.

CHESS (Talaei et al., 2024) introduz *framework* multi-agente para databases industriais com quatro agentes: Information Retriever, Schema Selector, Candidate Generator, Unit Tester. Resultados no BIRD test atingem 71.10% com high-compute, estabelecendo novo benchmark para cenários realistas e demonstrando a viabilidade de arquiteturas agênticas em contextos empresariais.

A análise transversal dos trabalhos recentes revela padrões convergentes: (1) *Task decomposition* como estratégia dominante; (2) *Schema linking* como primeira etapa crítica; (3) *Self-correction* via execution feedback; (4) *Multi-agent specialization* para databases complexos. Esta convergência sugere que a integração destes componentes representa o caminho mais promissor para sistemas de próxima geração.

## 4.6 Gaps de Pesquisa e Direções Futuras

O campo enfrenta desafios fundamentais não resolvidos: (1) *Schema linking* em escala - databases com milhares de colunas permanecem desafiadores; (2) *Dirty data handling* - gap entre dados limpos de *benchmarks* e dados reais; (3) Ambiguidade linguística - mesma pergunta pode ter múltiplas traduções SQL válidas; (4) Determinismo - natureza probabilística de LLMs introduz inconsistência; (5) Eficiência de custo - métodos SOTA requerem múltiplas chamadas LLM com custo substancial.

O gap entre Spider (~87% SOTA) e Spider 2.0 (~21% SOTA) constitui tanto advertência quanto oportunidade: sistemas atuais funcionam em cenários controlados mas falham em contextos empresariais reais. Pesquisa que endereça esta lacuna possui potencial de impacto significativo tanto acadêmico quanto prático.

A integração de RAG com técnicas agênticas representa fronteira mais promissora. Sistemas que combinam retrieval adaptativo de exemplos e schema, decomposição de tarefas via multi-agent, e self-correction com execution feedback aproximam-se de arquitetura ideal. Para este projeto, isto sugere arquitetura integrando: RAG para retrieval adaptativo, decomposição de tarefas inspirada em DIN-SQL/MAC-SQL, *framework* LangGraph para orquestração *stateful*, e mecanismos de self-correction com execution feedback.

Este projeto posiciona-se na confluência dos avanços mais recentes: utiliza LLMs via *in-context learning*, incorpora técnicas de RAG para retrieval de exemplos e metadados de schema, implementa arquitetura agêntica com LangGraph inspirada em MAC-SQL e CHESS, e aplica *prompt engineering* baseado em DAIL-SQL e DIN-SQL. A contribuição distintiva reside na aplicação deste paradigma integrado a

dados governamentais brasileiros (ANTAQ), domínio ainda não explorado em trabalhos internacionais, com atenção específica a requisitos de metodologia oficial e terminologia de domínio especializada.

## 5. METODOLOGIA

### 5.1 Abordagem Metodológica

O projeto adota uma abordagem *Design Science Research* (Hevner et al., 2004), caracterizada pela construção e avaliação de artefatos tecnológicos para resolver problemas práticos.

### 5.2 Fases da Pesquisa

#### Fase 1: Levantamento e Análise (etapas 1-2)

##### Atividades:

- Construção do BigQuery da ANTAQ
- Entrevistas com usuários dos dados (técnicos e não-técnicos)
- Catalogação de consultas SQL típicas
- Identificação de padrões de análise mais comuns

##### Entregáveis:

- Documentação do esquema de dados
- Catálogo de metadados semânticos
- Taxonomia de tipos de consultas

#### Fase 2: Projeto da Arquitetura (Etapa 3)

A arquitetura do sistema será composta por camadas hierárquicas inspiradas nos frameworks CHESS e MAC-SQL, incluindo:

- Camada de Apresentação: Interface web usando Streamlit
- Camada do Agente: Orquestração com LangGraph (Setup Schema, Retrieve Examples, Generate SQL, Validate SQL, Execute SQL, Format Answer)
- Camada de Serviços: Metadata Helper, RAG/Vector Store, Security Validator
- Camada de Dados: BigQuery - ANTAQ Dataset

#### Fase 3: Implementação do Core (Etapas 4-6)

##### Módulos Principais:

- MetadataHelper: Biblioteca para descrição semântica de schemas inspirada em DAIL-SQL
- RAG System: Recuperação de exemplos similares usando embeddings com técnicas de Self-RAG
- SQL Validator: Validação sintática e semântica das queries geradas
- LangGraph Agent: Máquina de estados para orquestração do fluxo inspirada em DIN-SQL

#### Fase 4: Interface e Integração (Etapa 7)

Desenvolvimento da interface web com Streamlit, incluindo:

- Interface conversacional para entrada de perguntas
- Visualização de resultados em tabelas e gráficos

- Exibição do SQL gerado para transparência
- Sistema de feedback do usuário

### **Fase 5: Testes e Validação (Etapas 8-10)**

#### **Métricas de Avaliação:**

- Execution Accuracy: Porcentagem de queries que retornam resultados corretos
- Valid Efficiency Score: Métrica do BIRD que considera correção e eficiência
- Satisfação do Usuário: Questionários e entrevistas pós-uso
- Tempo de Resposta: Latência média do sistema

### **Fase 6: Análise e Documentação (Etapas 11-12)**

Análise dos resultados, refinamento do sistema, redação de artigos científicos e documentação técnica completa.

## **5.3 Tecnologias e Ferramentas**

#### **Stack Tecnológico:**

- Python 3.11+ (type hints, pattern matching, asyncio)
- LangGraph >= 0.2.0 (orquestração de agentes)
- LangChain >= 0.3.0 (framework de agentes)
- Gemini 1.5 Flash via Vertex AI (LLM)
- BigQuery (banco de dados analítico)
- BigQuery Vector Store (RAG)
- Streamlit 1.28+ (interface web)

#### **Dados Utilizados:**

Dados do Estatístico Aquaviário da ANTAQ

## 6. CRONOGRAMA

Etapas	Atividade	Responsável
1	Levantamento de requisitos e análise de dados	Pesquisador
2	Projeto da arquitetura	Pesquisador + Orientador
3	Implementação MetadataHelper e RAG	Pesquisador
4	Implementação LangGraph e validação SQL	Pesquisador
5	Desenvolvimento interface Streamlit	Pesquisador
6	Testes técnicos e ajustes	Pesquisador
7	Validação com usuários	Pesquisador + Voluntários
8	Análise de resultados e refinamento	Pesquisador + Orientador
9	Documentação e artigos científicos	Pesquisador + Orientador

## 7. RESULTADOS ESPERADOS

O projeto espera gerar produtos técnicos (sistema funcional, interface web, código *open source*), produtos científicos (artigos em periódicos e congressos nacionais/internacionais), e contribuições teóricas (metodologia de engenharia de metadados, arquitetura de agentes especializados para *Text-to-SQL*).

## 8. RISCOS E MITIGAÇÕES

Os principais riscos incluem: dificuldade do LLM em compreender domínio específico (mitigação via few-shot learning + metadados ricos), geração de queries SQL incorretas (mitigação via validador sintático + semântico), performance insuficiente (mitigação via cache + otimizações de prompt), baixa adoção pelos usuários (mitigação via testes com usuários + UX refinada), e custos elevados de API (mitigação via cache + modelo custo-benefício).

## 9. ASPECTOS ÉTICOS

Todos os dados utilizados são públicos (governamentais), não há coleta de dados pessoais, e o sistema será transparente sobre suas limitações. A validação de SQL evitará consultas maliciosas, e haverá limitação de recursos (max rows, timeout) para responsabilidade computacional. A interface será em português brasileiro com suporte a linguagem natural.

## 10. REFERÊNCIAS

ANDROUTSOPoulos, I.; RITCHIE, G. D.; THANISCH, P. **Natural language interfaces to databases: an introduction**. arXiv preprint cmp-lg/9503016, 1995.

ANTAQ. **Anuário estatístico do transporte aquaviário 2023**. Brasília: ANTAQ, 2023.

ANTAQ. **Estatística aquaviária: metodologia oficial**. Brasília: ANTAQ, 2024.

ASAI, A. et al. **Self-RAG: learning to retrieve, generate, and critique through self-reflection**. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS, 2024. Proceedings... [S.I.]: ICLR, 2024.

BIGQUERY DOCUMENTATION. **BigQuery documentation**. Disponível em: <https://cloud.google.com/bigquery/docs>. Acesso em 12 dez. 2025.

DONG, X. et al. **C3: zero-shot text-to-SQL with ChatGPT**. arXiv:2307.07306, 2023.

GAO, D. et al. **Text-to-SQL empowered by large language models: a benchmark evaluation**. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 2024. Proceedings... [S.I.]: VLDB, 2024.

GAO, Y. et al. **Retrieval-augmented generation for large language models: a survey**. arXiv:2312.10997, 2024.

HEVNER, A. R. et al. **Design science in information systems research**. MIS Quarterly, v. 28, n. 1, p. 75-105, 2004.

KATSOGIANNIS-MEIMARAKIS, G.; KOUTRIKA, G. **A survey on deep learning approaches for text-to-SQL**. The VLDB Journal, v. 32, p. 905-936, 2023.

LANGGRAPH DOCUMENTATION. **LangGraph documentation**. Disponível em: <https://langchain-ai.github.io/langgraph/>. Acesso em 18 dez. 2025.

LEI, F. et al. **Spider 2.0: evaluating language models on real-world enterprise text-to-SQL workflows**. arXiv:2411.07763, 2024.

LEWIS, P. et al. **Retrieval-augmented generation for knowledge-intensive NLP tasks**. In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2020. Proceedings... [S.I.]: NeurIPS, 2020.

LI, H. et al. **CodeS: towards building open-source language models for text-to-SQL**. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 2024. Proceedings... [S.I.]: SIGMOD, 2024.

LI, H. et al. **RESDSQL: decoupling schema linking and skeleton parsing for text-to-SQL**. In: AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE, 2023. Proceedings... [S.I.]: AAAI, 2023.

LI, J. et al. **Can LLM already serve as a database interface? A big bench for large-scale database grounded text-to-SQLs.** In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2023. Proceedings... [S.I.]: NeurIPS, 2023.

POURREZA, M.; RAFIEI, D. **DIN-SQL: decomposed in-context learning of text-to-SQL with self-correction.** In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2023. Proceedings... [S.I.]: NeurIPS, 2023.

QIN, B. et al. **A survey on text-to-SQL parsing: concepts, methods, and future directions.** arXiv:2208.13629, 2022.

SHINN, N. et al. **Reflexion: language agents with verbal reinforcement learning.** In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2023. Proceedings... [S.I.]: NeurIPS, 2023.

TAI, C. Y. et al. **Exploring chain of thought style prompting for text-to-SQL.** In: CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING, 2023. Proceedings... [S.I.]: EMNLP, 2023.

TALAEI, S. et al. **CHESS: contextual harnessing for efficient SQL synthesis.** arXiv:2405.16755, 2024.

WANG, B. et al. **MAC-SQL: a multi-agent collaborative framework for text-to-SQL.** arXiv:2312.11242, 2024.

WANG, B. et al. **RAT-SQL: relation-aware schema encoding and linking for text-to-SQL parsers.** In: ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 2020. Proceedings... [S.I.]: ACL, 2020.

WEI, J. et al. **Chain-of-thought prompting elicits reasoning in large language models.** In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2022. Proceedings... [S.I.]: NeurIPS, 2022.

YAO, S. et al. **ReAct: synergizing reasoning and acting in language models.** In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS, 2023. Proceedings... [S.I.]: ICLR, 2023.

YU, T. et al. **Spider: a large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task.** In: CONFERENCE ON EMPIRICAL METHODS IN NATURAL LANGUAGE PROCESSING, 2018. Proceedings... [S.I.]: EMNLP, 2018.

ZHONG, V.; XIONG, C.; SOCHER, R. **Seq2SQL: generating structured queries from natural language using reinforcement learning.** arXiv:1709.00103, 2017.

ZHU, X. et al. **Large language model enhanced text-to-SQL generation: a survey.** arXiv:2410.06011, 2024.