
# Enterprise Multi-Database Natural Language Query Engine

Version 1.0

Prepared for

myOnsite Healthcare, LLC.

# Document Control

| Rev. No. | Description of Change | Effective Date |
|---|---|---|
| 1.0 | Initial Release | 11st Aug 2025 |

## Authored By

| Name | Role | Signature | Date |
|---|---|---|---|
| Het | Team Lead | | 11st Aug 2025 |

## Reviewed and approved By

| Name | Role | Signature | Date |
|---|---|---|---|
| | | | |

# Enterprise Multi-Database Natural Language Query Engine

## Advanced Text-to-SQL System with Dynamic Schema Adaptation

---

## Project Overview

Build a production-grade, multi-database natural language query engine that can understand complex business questions, generate optimized SQL across different database systems, handle real-time schema evolution, implement advanced security controls, and provide enterprise-level governance and observability.

**Time Allocation:** 5 hours
**Complexity Level:** Principal Engineer Challenge
**Focus Areas:** Advanced NL-to-SQL, multi-database orchestration, real-time schema adaptation, enterprise security

---

## System Architecture Requirements

You're building an enterprise query engine that must:

- Support **multiple database systems** (PostgreSQL, MySQL, SQLite, MongoDB) with dialect-specific optimizations
- Handle **complex natural language queries** including temporal reasoning, multi-table joins, and business logic inference

- Implement **real-time schema introspection** with automatic adaptation to structural changes
- Provide **advanced security controls** including SQL injection prevention, role-based access, and query governance
- Support **concurrent query execution** with intelligent caching and performance optimization
- Include **comprehensive audit logging** and explainable AI for query reasoning
- Handle **ambiguous queries** with clarification mechanisms and confidence scoring

---

# Database Environment

## Multi-Database Setup

You'll work with **4 interconnected databases**:

1. **PostgreSQL** (Primary OLTP)
    - **Tables:** customers, orders, products, inventory, suppliers, categories
    - **Complex features:** Foreign keys, indexes, triggers, stored procedures, views
    - **Data volume:** ~100K customer records, ~500K order records
    - **Advanced structures:** JSON columns, arrays, custom types, partitioned tables
2. **MySQL** (Analytics/Reporting)
    - **Tables:** sales_analytics, customer_segments, product_performance, regional_data
    - **Features:** Window functions, CTEs, materialized views
    - **Data volume:** ~1M aggregated records with time series data

- **Complex queries:** Multi-level aggregations, rolling averages, cohort analysis

3. **SQLite** (Configuration/Metadata)
    - **Tables:** system_config, user_preferences, query_history, schema_versions
    - **Features:** Full-text search, JSON support, temporary tables
    - **Dynamic schema:** Tables and columns change frequently
4. **MongoDB** (Document Store)
    - **Collections:** user_profiles, product_catalogs, activity_logs, recommendations
    - **Features:** Complex nested documents, aggregation pipelines, geospatial queries
    - **Challenge:** Convert natural language to MongoDB aggregation syntax

## Advanced Query Dataset

**25 progressively complex natural language queries:**

**Tier 1 - Basic (5 queries):**

- "Show me all customers from California"
- "What's the total revenue this month?"

**Tier 2 - Multi-table Joins (8 queries):**

- "Which customers have placed orders worth more than $1000 in the last 6 months but haven't ordered in the past 30 days?"
- "Show me the top 5 product categories by revenue growth compared to last quarter"

**Tier 3 - Temporal & Analytics (7 queries):**

- "Calculate the 3-month rolling average of monthly recurring revenue by customer segment"
- "Find customers who exhibited churn behavior patterns similar to our top 10% revenue customers"

**Tier 4 - Cross-Database Complex (5 queries):**

- "Compare PostgreSQL order patterns with MongoDB user activity to identify customers likely to upgrade their subscription tier within 60 days"
- "Generate a cohort analysis of customer lifetime value using data from all databases, segmented by acquisition channel and geographic region"

---

# Advanced Technical Requirements

## Multi-Database Query Orchestration

- **Database abstraction layer** supporting PostgreSQL, MySQL, SQLite, and MongoDB
- **Dialect-specific SQL generation** with database-specific optimizations
- **Cross-database join capabilities** using temporary staging and federated queries
- **Connection pooling and failover** with automatic load balancing
- **Transaction management** across multiple database connections
- **Query result federation** and cross-database data correlation

## Advanced Natural Language Processing

- **Multi-intent query parsing** handling compound questions and sub-queries

- **Business context awareness** understanding domain-specific terminology and relationships
- **Temporal reasoning** interpreting relative dates, time ranges, and business cycles
- **Ambiguity resolution** with confidence scoring and clarification requests
- **Query expansion** automatically adding relevant filters and joins based on business rules
- **Conversational context** maintaining query history and follow-up question handling

## Dynamic Schema Intelligence

- **Real-time schema discovery** across all database systems with caching
- **Schema relationship mapping** automatically detecting foreign key relationships and business logic
- **Change detection and adaptation** handling schema modifications without service interruption
- **Version management** tracking schema changes and maintaining query compatibility
- **Intelligent column mapping** handling renamed/moved columns with semantic similarity matching
- **Constraint awareness** understanding business rules encoded in database constraints

## Enterprise Security Framework

- **Multi-layered SQL injection prevention** with parameterized queries and input validation
- **Role-based access control** with fine-grained table and column permissions
- **Query governance** with approval workflows for potentially expensive operations

- **Data masking and anonymization** for sensitive information in query results
- **Audit trail generation** logging all queries, results, and user interactions
- **Rate limiting and quota management** preventing resource abuse

## Advanced Query Generation

- **Multi-step reasoning** breaking complex questions into executable query sequences
- **Optimization hint generation** suggesting indexes and query plan improvements
- **Cost estimation** predicting query execution time and resource usage
- **Alternative query suggestions** providing multiple approaches for ambiguous requests
- **Business logic integration** incorporating calculated fields and derived metrics
- **Error recovery and self-healing** automatically fixing common SQL syntax issues

## Real-Time Schema Evolution Handling

During execution, your system will face **multiple schema changes**:

**Phase 1 (Hour 2):** Basic structural changes

- Rename `customers.email` to `customers.email_address`
- Add new table `customer_segments` with foreign key relationships
- Modify `orders` table to add `discount_applied` column

**Phase 2 (Hour 3):** Complex relationship changes

- Split `products` table into `products` and `product_variants`
- Add multi-table view `customer_order_summary`
- Introduce partitioning on `orders` table by date range

**Phase 3 (Hour 4):** Advanced schema evolution

- Add MongoDB integration with `user_behavior` collection
- Introduce PostgreSQL array columns for tags and categories
- Add stored procedures for complex business logic calculations
- Implement row-level security policies

**Your system must:**

- **Detect changes automatically** without manual intervention
- **Adapt queries in real-time** maintaining correctness and performance
- **Provide graceful degradation** when schemas are temporarily incompatible
- **Maintain query history compatibility** ensuring previous queries still work
- **Update cached schema information** without service interruption

---

# Extreme Implementation Challenges

## Multi-Database Query Federation

- **Cross-database joins:** Query data from PostgreSQL customers table and MongoDB user behavior collection simultaneously
- **Data type harmonization:** Convert between SQL and NoSQL data structures seamlessly
- **Transaction coordination:** Maintain ACID properties across multiple database systems
- **Performance optimization:** Minimize cross-database data transfer and optimize execution plans

## Advanced Natural Language Understanding

- **Multi-intent parsing:** "Show me revenue trends AND identify underperforming products AND suggest optimization strategies"
- **Contextual ambiguity resolution:** Handle pronouns, temporal references, and business domain terminology
- **Conversational query chaining:** Support follow-up questions that reference previous query results
- **Voice input processing:** Accept and process natural language queries via voice API integration

## Real-Time Schema Evolution

- **Live migration handling:** Continue serving queries while schema changes are applied
- **Backward compatibility:** Ensure existing queries continue to work during and after schema changes
- **Semantic column matching:** Automatically map renamed columns based on data content and usage patterns
- **Relationship inference:** Detect new foreign key relationships and business logic automatically

## Enterprise Security & Governance

- **Dynamic data masking:** Apply different masking rules based on user roles and query context
- **Query approval workflows:** Route potentially expensive or sensitive queries through approval processes
- **Compliance reporting:** Generate audit reports for SOX, GDPR, and industry-specific regulations
- **Zero-trust architecture:** Verify permissions for every query component and data access

## Advanced Performance Optimization

- **Intelligent query caching:** Cache results at multiple levels with invalidation based on data changes

- **Predictive prefetching:** Anticipate follow-up queries and pre-execute common patterns
- **Cost-based routing:** Route queries to optimal database replicas based on current load and costs
- **Real-time performance tuning:** Automatically adjust query execution strategies based on performance metrics

---

# Schema Evolution Stress Test

Your system will face **continuous schema changes** throughout development:

**Hour 1:** Initial setup with stable schemas across all databases **Hour 2: First Evolution Wave**

- PostgreSQL: Rename 3 columns, add 2 new tables with complex foreign key relationships
- MySQL: Partition existing tables, add materialized views
- SQLite: Add full-text search indices, modify constraint definitions
- MongoDB: Restructure document schemas, add new nested field structures

**Hour 3: Complex Relationship Changes**

- Cross-database foreign key simulation through application logic
- Introduction of database-specific advanced features (PostgreSQL arrays, MySQL JSON functions)
- Schema versioning with multiple concurrent versions active

**Hour 4: Enterprise Feature Integration**

- Row-level security policy implementation
- Database-specific stored procedures and functions
- Real-time replication setup with read/write splitting

---

- Advanced indexing strategies (partial indexes, functional indexes)

**Hour 5: Stress Testing & Recovery**

- Simulate database connectivity issues and failover scenarios
- Handle concurrent schema changes from multiple sources
- Test query adaptation under high concurrent load
- Validate data consistency across all database systems

---

# Success Metrics & Expectations

## Functional Requirements (Must Have)

- **Query Accuracy:** >90% correct SQL generation for Tier 1-3 queries, >75% for Tier 4
- **Schema Adaptation:** Automatically handle all schema changes within 30 seconds
- **Multi-Database Support:** Successfully execute queries across all 4 database systems
- **Security Compliance:** Pass all SQL injection tests and access control validations
- **Performance Standards:** <3 seconds for simple queries, <10 seconds for complex cross-database operations

## Advanced Capabilities (Differentiation Factors)

- **Natural Language Sophistication:** Handle ambiguous queries with confidence scoring >0.8
- **Business Intelligence:** Provide actionable insights and alternative query suggestions
- **Explainable AI:** Generate clear explanations for query reasoning and decision-making

- **Enterprise Readiness:** Comprehensive audit logging, compliance reporting, and governance workflows
- **Innovation Factor:** Creative solutions to unique challenges like voice input or predictive querying

### Production-Grade Excellence

- **Monitoring & Observability:** Real-time dashboards with predictive alerting
- **Scalability:** Handle 1000+ concurrent queries without performance degradation
- **Reliability:** 99.9% uptime with graceful degradation during failures
- **Cost Optimization:** Intelligent resource usage with cost tracking and budgeting
- **Security Excellence:** Zero vulnerabilities in penetration testing

---

# Deliverables

### 1. Production System

- **Multi-interface access:** REST API, GraphQL endpoint, WebSocket streaming, and CLI
- **Enterprise dashboard:** Real-time monitoring, query analytics, and administrative controls
- **Mobile-responsive UI:** Web interface for natural language query input and result visualization

### 2. Comprehensive Test Suite

- **Unit tests:** >85% code coverage with edge case handling
- **Integration tests:** End-to-end workflow validation across all database systems
- **Performance benchmarks:** Load testing with scalability analysis

---

- **Security validation:** Penetration testing and vulnerability assessment
- **Schema evolution tests:** Automated testing of all schema change scenarios

## 3. Enterprise Documentation

- **System architecture:** Detailed technical design with decision rationale
- **API documentation:** Interactive documentation with live examples
- **Deployment guide:** Production deployment with high-availability configuration
- **Security documentation:** Security controls, threat model, and compliance mapping
- **Performance tuning:** Optimization strategies and troubleshooting guide

## 4. Business Intelligence Layer

- **Query analytics:** Usage patterns, performance trends, and optimization opportunities
- **Business insights:** Automated detection of data anomalies and business opportunities
- **Cost analysis:** Query cost tracking with optimization recommendations
- **Compliance reporting:** Automated audit reports and regulatory compliance validation