Query Generation

July 5, 2025

Outline

What is Query Generation?

Core Definition

Query Generation is a **reverse process** to instruct LLMs to generate specific query types given context samples for comprehensive system testing.

Key Objectives:

- Evaluate system's ability to retrieve relevant contexts
- Generate accurate responses across diverse scenarios
- Stress-test capabilities: reasoning, recall, synthesis, generalization
- Ensure comprehensive coverage of system functionality

Why It Matters

Enables targeted stress-testing and scenario-based evaluation of system capabilities under realistic and diverse user inputs.

Query Generation Benefits

Comprehensive Assessment

- Tests retrieval and generation accuracy
- Covers all topics systematically
- Enables scenario-based testing
- Identifies specific weaknesses

Diverse Coverage

- Fact extraction to analytical reasoning
- Multiple linguistic formats
- Noise and ambiguity handling
- Structured data interpretation

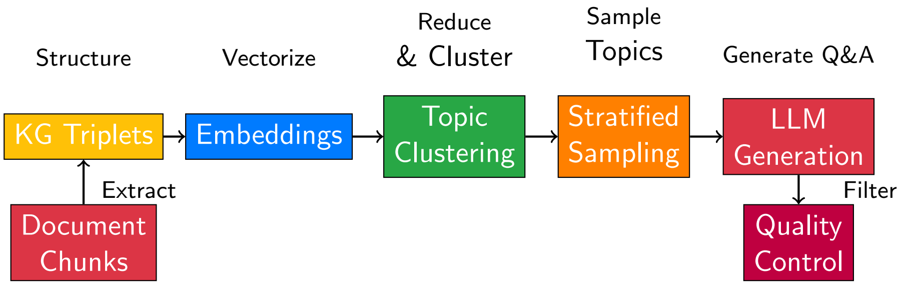
Automated Scale

- LLM-powered generation
- Stratified sampling approach
- Quality control mechanisms
- Consistent output formats

Robust Evaluation

- Consistency across variations
- Error tolerance testing
- Edge case identification
- Performance evaluation of specific use case

Embedding-Driven Workflow



Process Steps:

1. **Structuring**: Convert document chunk to triplets
2. **Vectorization**: Convert triplets to embeddings
3. **Clustering**: Group similar content into topics using dimensionality reduction
4. **Stratified Sampling**: Ensure representative coverage across all topics
5. **Generation**: Use LLMs to create diverse query types from samples
6. **Quality Control**: Filter and validate generated Q&A pairs

Topic Stratification and Sampling

Stratification Strategy

Topics (clusters of similar contents) define stratification to guarantee comprehensive test coverage.

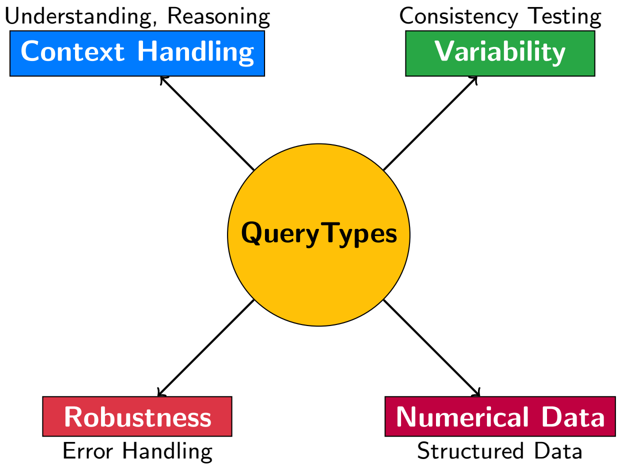
Sampling Approaches:

- **Random Samples**: Basic sampling from each topic cluster
- **Twinning Samples**: Guarantee distribution replication with statistical properties preservation

Coverage Guarantee: Embedding-based stratification ensures that test generation covers the full semantic space of the document corpus systematically

- Samples from each topic as contexts
- Ensures no semantic domain is left untested
- Maintains proportional representation
- Enables targeted evaluation of specific areas

Four Main Query Categories



Each category tests different aspects of system capabilities with specific objectives and evaluation criteria.

Context Handling Queries

Objective

Test system's ability to retrieve right contexts and to understand, recall, reason, synthesize, and generalize.

Query Subtypes:

- **Fact Extraction**: Extract specific facts without additional reasoning
- **Definition & Explanation**: Test ability to define or explain terms
- **Procedural**: Understanding of processes or sequences
- **Analytical Reasoning**: Logical analysis and synthesis
- **Multi-Hop Reasoning**: Integration across multiple context parts

Example System Prompt

Fact Extraction: "Given the following context, generate a question that extracts a specific fact or detail without requiring additional reasoning. Ensure the question is concise and unambiguous."

Variability Queries

Objective

Test whether system generates consistent responses to similar inputs and manages variability across different contexts and prompts.

Query Subtypes:

- **Paraphrased Queries**: Similar queries phrased differently
- **Repeated Queries**: Same query multiple times for consistency
- **Rephrased Follow-Up**: Different phrasings of follow-up questions
- **Multi-Round**: Sequential related queries building on each other
- **Fine-Grained Detail**: Focus on minor details for variability testing

Example System Prompt

Paraphrased: "Given the following context, generate multiple paraphrased versions of a query asking for the same information in different ways."

Robustness Queries

Objective

Test system ability to handle ambiguity, errors, noise, domain-specific intricacies, and stress-test scenarios.

Query Subtypes:

- **Ambiguous Queries**: Multiple plausible interpretations
- **Noisy Queries**: Typographical errors, grammar issues, mixed languages
- **Out-of-Context**: Information not present in context
- **Edge Cases**: Rare or boundary conditions
- **Domain Jargon**: Highly technical or niche terminology

Example System Prompt

Ambiguous: "From the given context, generate a query with ambiguous phrasing or multiple valid interpretations that could confuse the retrieval process."

Numerical, Table and Chart Queries

Objective

Test whether system can accurately extract, interpret, and reason about structured and semi-structured data.

Query Subtypes:

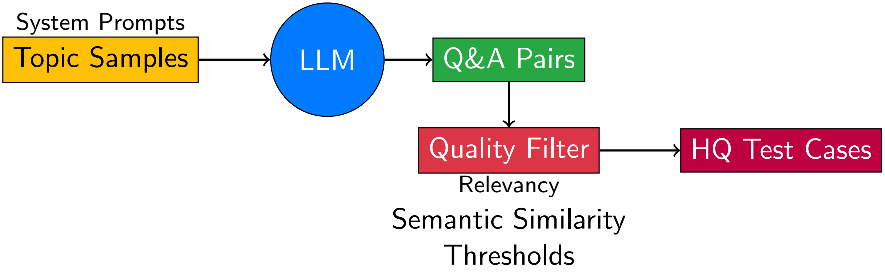
- **Comparative Numerical**: Compare numerical values (max, min, relations)
- **Trend Analysis**: Identify patterns in data over time
- **Table Lookup**: Retrieve specific information from tables
- **Chart Data Retrieval**: Extract specific values from charts
- **Cross-Referencing**: Reason across tables and charts together

Example System Prompt

Trend Analysis: "Generate a query about identifying trends or patterns in the numerical data provided in the context."

Automated Query Generation Workflow

Example Prompt:
"Based on the information of the given chunk, generate 10 Q&A pairs from the text following the designed format."



Process Elements:

- **Structured Prompts**: Specific instructions for each query type
- **Template Formats**: Consistent output formatting
- **Quality Metrics**: Relevancy and semantic similarity evaluation
- **Threshold Filtering**: Remove low-quality generated pairs

Quality Control and Evaluation

Evaluation Metrics:

- **Relevancy**: How well does the question relate to the context?
- **Semantic Similarity**: Do questions and answers align semantically?
- **Coverage**: Are all important topics represented?
- **Diversity**: Do we have sufficient variety in query types?

Quality Control Process:

1. Calculate evaluation metrics for each Q&A pair
2. Set reasonable quality thresholds
3. Filter out low-quality test cases
4. Ensure balanced representation across categories
5. Validate against human-created gold standards

Best Practice

Always implement quality control mechanisms to ensure generated test cases meet evaluation standards and provide meaningful system insights.

Implementation Workflow

Step-by-Step Implementation:

1. **Data Preparation**
    - Chunk documents appropriately
    - Generate embeddings for all triplets
    - Perform clustering and topic identification
2. **Sampling Strategy**
    - Implement stratified sampling across topics and query types
    - Create twinning samples for distribution replication
    - Ensure adequate coverage of all semantic domains
3. **Prompt Engineering**
    - Design category-specific system prompts
    - Create output templates for consistency
4. **Generation and Validation**
    - Generate Q&A pairs using LLMs
    - Apply quality control filters
    - Validate against evaluation metrics

System Testing Framework

Context Handling:

- Understanding
- Recall
- Reasoning
- Synthesis
- Generalization

Robustness:

- Ambiguity handling
- Error tolerance
- Noise resistance
- Domain adaptation

Consistency:

- Response stability
- Format consistency
- Multi-turn coherence
- Paraphrase handling

Data Processing:

- Numerical reasoning
- Table interpretation
- Chart analysis
- Cross-referencing

Comprehensive Coverage: Query generation enables systematic testing of all critical system capabilities through diverse, automatically generated test scenarios.

Best Practices Summary

Key Recommendations:

1. **Systematic Approach**: Use embedding-based stratification for comprehensive coverage
2. **Diverse Categories**: Implement all four query types for complete evaluation
3. **Quality Control**: Always filter generated content using multiple metrics
4. **Template Consistency**: Use structured prompts and output formats
5. **Multi-Step Processing**: Break complex tasks into manageable subtasks
6. **Continuous Improvement**: Regularly update based on evaluation results

Success Factors

Effective query generation requires careful balance of systematic sampling, diverse query types, quality control, and continuous refinement based on evaluation results.

Key Takeaways

1. **Systematic Testing is Essential**
   - Query generation enables comprehensive RAG evaluation
   - Embedding-based stratification ensures complete coverage
   - Four query categories test different system capabilities
2. **Automation Scales Evaluation**
   - LLM-powered generation handles large-scale testing
   - Structured prompts ensure consistent, high-quality outputs
   - Multi-step processes improve generation quality
3. **Quality Control is Critical**
   - Multiple evaluation metrics prevent low-quality tests
   - Template formats ensure output consistency
   - Continuous refinement improves system effectiveness
4. **Comprehensive Coverage Enables Insights**
   - Diverse query types reveal different system strengths/weaknesses
   - Systematic sampling ensures no gaps in evaluation
   - Results guide targeted system improvements