Journal of the Korean Society of Surveying, Geodesy, Photogrammetry and Cartography
Vol. 40, No. 5, 429-437, 2022
https://doi.org/10.7848/ksgpc.2022.40.5.429

ISSN 1598-4850(Print)
ISSN 2288-260X(Online)
Original article

속성 그래프 및 GraphQL을 활용한 지식기반 공간 쿼리 시스템 설계

Design of Knowledge-based Spatial Querying System Using Labeled Property Graph and GraphQL

장한메[1] · 김동현[2] · 유기윤[3]
Jang, Hanme · Kim, Dong Hyeon · Yu, Kiyun

Abstract

Recently, the demand for a QA (Question Answering) system for human-machine communication has increased. Among the QA systems, a closed domain QA system that can handle spatial-related questions is called GeoQA. In this study, a new type of graph database, LPG (Labeled Property Graph) was used to overcome the limitations of the RDF (Resource Description Framework) based database, which was mainly used in the GeoQA field. In addition, GraphQL (Graph Query Language), an API-type query language, is introduced to address the fact that the LPG query language is not standardized and the GeoQA system may depend on specific products. In this study, database was built so that answers could be retrieved when spatial-related questions were entered. Each data was obtained from the national spatial information portal and local data open service. The spatial relationships between each spatial objects were calculated in advance and stored in edge form. The user's questions were first converted to GraphQL through FOL (First Order Logic) format and delivered to the database through the GraphQL server. The LPG used in the experiment is Neo4j, the graph database that currently has the highest market share, and some of the built-in functions and QGIS were used for spatial calculations. As a result of building the system, it was confirmed that the user's question could be transformed, processed through the Apollo GraphQL server, and an appropriate answer could be obtained from the database.

Keywords : GeoQA, Knowledge Graph, GraphQL, Labeled Property Graph, Neo4j

초 록

최근 사람과 기계의 소통을 위해 QA (Question Answering) 시스템에 대한 요구가 증가하였다. QA 시스템 중 공간에 관련된 질문을 처리할 수 있는 폐쇄 도메인 QA 시스템을 GeoQA라 하는데 본 연구는 GeoQA 분야에서 주로 사용되던 RDF (Resource Description Framework)기반의 데이터베이스가 데이터 입출력 및 변형에 한계를 보인다는 점을 극복하기 위해 최근 주목받고 있는 새로운 형태의 그래프 데이터베이스인 LPG (Labeled Property Graph)를 사용하였다. 또한, LPG 쿼리(query)언어가 표준화되지 않아 GeoQA 시스템이 특정 제품에 의존할 수 있다는 점 때문에 API 형태의 쿼리 언어인 GraphQL (Graph Query Language)을 도입하여 다양한 LPG를 사용할 방안을 제시하였다. 본 연구에서는 공간 관련 질문이 입력되었을 때 답변을 검색할 수 있도록 대한민국 중심의 별도 데이터베이스를 구축하였는데 각 데이터는 국가공간정보포털 및 지방행정 인허가데이터개방 서비스에서 취득하였으며 각 공간 객체 간 공간적 관계는 미리 계산되어 그래프의 엣지(edge) 형태로 입력되었다. 사용자의 질문은 먼저 FOL (First Order Logic)형태를 거쳐 최종적으로 GraphQL로 변환되며 GraphQL 서버를 통해 데이터베이스에 전달되었다. 실험에 사용한 LPG로는 현재 가장 높은 점유율을 보이는 그래프 데이터베이스인 Neo4j를 선택하였고 내장 함수와 QGIS 일부가 공간 연산에 사용되었다. 시스템 구축 결과 사용자의 질문을 변환, Apollo GraphQL 서버를 통해 처리하고 데이터베이스로부터 적합한 답변을 얻을 수 있음을 확인하였다.

핵심어 : 공간QA, 지식 그래프, GraphQL, 속성 그래프, Neo4j

Received 2022. 09. 26, Revised 2022. 10. 17, Accepted 2022. 10. 30
1) Member, Dept. of Civil and Envionmental Engineering, Seoul National University(E-mail: janghaniel@snu.ac.kr)
2) Dept. of Civil and Envionmental Engineering, Seoul National University(E-mail: kaikim98@snu.ac.kr)
3) Corresponding Author, Member, Dept. of Civil and Envionmental Engineering, Seoul National University (E-mail: kiyun@snu.ac.kr)

1. 서 론

GIS 시스템이 점점 고도화되면서 일반 사용자가 공간정보 사업자로부터 생산된 데이터를 단순히 사용하기보다 시스템과 밀접하게 상호작용하는 것이 가능해졌다. 실제로 최근 사용자의 위치를 기반으로 한 검색 서비스나 도시 공간 내에서 사물들의 상대적 관계, 날씨 정보 등을 제공하는 서비스에 대한 요구가 증가하였으며 이를 위해 온라인상에서 원하는 지리정보를 검색, 획득한 뒤 사용자의 질문에 응답할 수 있는 시스템이 다수 개발되었다(Lee, 2012). 사용자와 컴퓨터 사이의 상호이해과정을 QA (Question Answering)라 하는데 QA는 정보탐색, 자연어처리, 컴퓨터 언어학에 걸친 융합 학문으로 최근 인공지능 및 빅데이터 분야의 발전과 함께 가장 주목받는 연구 분야 중 하나이다.

QA 시스템을 구축하기 위해서는 먼저 광범위한 데이터베이스가 필요한데 최근에는 다양한 지식이 연결된 형태의 지식 그래프가 구축되어 정보탐색 및 추론이 가능해졌다. 실제로 인터넷에 존재하는 방대한 정보를 체계적으로 조회하기 위해 시맨틱웹과 LOD (Linked Open Data) 온톨로지가 구축되었다. 최신의 QA 시스템은 KB (Knowledge Base)를 사용한 그래프 탐색 방법과 추론방법으로 구분되는데 전자는 사용자의 질문이 기계에 입력되었을 때 Freebase, YAGO2 등 방대한 그래프 데이터베이스를 이용하여 체계적으로 데이터를 탐색한 뒤 답을 도출하는 것을 의미한다. 반면, 추론방법은 객체들을 그래프 형태로 저장하고 딥러닝 기법을 적용하여 각 노드 및 노드 사이의 관계를 벡터공간에 임베딩하여 결과를 도출하는 방법이며 현재 웹에 존재하는 데이터는 완전하지 않기 때문에 두 방법은 상호호환적인 관계가 있다.

KB는 일반적으로 W3C표준 프레임워크인 RDF (Resource Description Framework)가 사용되는데 RDF는 웹에 존재하는 온톨로지를 Subject, Predicate, Object로 표현된다. 만약 QA 시스템에서 사람에 의해 질문이 입력되면 기계는 이 질문을 데이터베이스가 이해할 수 있는 언어인 SPARQL로 변환하여 데이터를 탐색, 반환하게 된다.

한편, 공간과 관련된 질문에 대한 답변을 생성하는 분야를 GeoQA라 하는데 현재 개방형 공간정보 컨소시엄(OGC: Open GeoSpatial Consortium)에서는 RDF로 구성된 데이터베이스에서 공간과 관련된 속성을 조회, 연산하기 위한 언어인 GeoSPARQL을 제안, 채택하고 있다. GeoQA 시스템을 구현하기 위한 대표적인 KB로는 WorldKG, YAGO2geo 등이 있는데 방대한 데이터베이스의 크기에도 불구하고 공간정보의 기본요소인 점, 선, 면을 속성으로 가지고 있는 객체의 수가 부족하며 각 객체 간 관계들이 명확하게 정의되어 있지 않으므로 공간정보와 관련된 질문에 정확하게 답하기는 매우 어렵다(Mai *et al*., 2021). 따라서 시스템의 성능을 높이기 위해서는 기존의 그래프 데이터에 공공데이터나 사용자가 요구하는 데이터 및 공간정보 토폴로지 등을 추가하여 완성도 높은 KB를 제작할 필요가 있다.

현재 GeoQA 시스템은 보편적으로 RDF형태의 LoD를 저장한 데이터베이스에 자연어로부터 변환한 SPARQL을 호출하도록 설계되어 있는데, 기존에 사용되는 그래프 데이터베이스의 노드 및 엣지(edge), 속성 등은 이미 제작되어 온라인에 저장된 객체이므로 데이터베이스에 새로운 데이터 및 공간정보 객체 간 토폴로지를 저장하기에는 적합하지 않다. 따라서 본 연구는 필요한 데이터를 유연하게 추가할 수 있는 그래프 데이터베이스인 LPG (Labeled Property Graph)를 도입하여 기존의 공간정보 KB에 새로운 공간정보를 추가한 데이터베이스를 구축, GeoQA시스템을 구현하고자 한다.

LPG를 GeoQA 시스템에 적용하기 위해서는 SPARQL 중심이었던 쿼리(query)를 LPG에 적합한 형태로 변형해야 하는데 LPG는 상대적으로 최근에 주목받고 있는 그래프 데이터베이스로서 NeptuneDB는 Gremlin, ArangoDB는 AQL, Neo4j는 Cypher 등 개발사마다 서로 다른 쿼리 언어를 사용하고 있다. 따라서 LPG를 사용하기 위해서는 QA 시스템에 입력된 자연어 질문이 쿼리 언어로 변환될 때 각 데이터베이스의 종류에 의존할 수밖에 없고 자연어처리, 쿼리 생성 등의 부분에서 LPG마다 새로운 방법론이 개발되어야 한다는 단점이 있다. 따라서 본 연구에서는 LPG를 이용하여 데이터베이스를 구축하는 것뿐만 아니라 유연하게 데이터를 처리할 수 있는 LPG의 장점을 유지하면서 특정 데이터베이스 쿼리 언어에 의존하지 않도록 GraphQL API를 시스템에 도입하는 프레임워크도 함께 제안하였다.

구축한 데이터베이스 및 QA 시스템의 유효함을 확인하기 위한 연구의 내용으로 2장에서는 QA시스템 및 그래프 데이터베이스 쿼리에 관한 내용을 다룬다. 3장에서는 공간에 관련된 질문을 선별, 분석하고 현재 LPG 중 가장 점유율이 높은 Neo4j를 사용하여 풍부한 온톨로지 및 객체를 가진 데이터베이스를 구축하는 방법을 제시한다. 또 사용자의 질문을 GraphQL로 변환, 데이터베이스로부터 질문의 답을 얻는 파이프라인 및 실험결과를 확인할 수 있다. 마지막으로 4장에서는 본 연구를 통한 결론과 시사점을 도출한다.

2. 이론적 배경

2.1 GeoQA

QA 시스템이란 기계가 사람의 자연어를 이해하고 적합한 답변을 주는 시스템을 의미하며 다양한 분야의 질문에 답하는 오픈 도메인 QA와 특정한 분야(법률, 의료, 관광 등)에만 국한하여 전문적인 대답을 제공하는 폐쇄 도메인 QA로 구분된다. 일반적으로 폐쇄 도메인 QA는 더 제한적이고 전문적인 질문과 데이터를 취급하기 때문에 답변의 정확도가 더 높다고 알려져 있다. QA 시스템은 구현 방법에 따라 문서나 정형화된 데이터로 구성된 데이터베이스에서 정보를 찾고 답변하는 IRQA (Information Retrieval QA) 방법과 KB 또는 KG (Knowledge Graph)로부터 데이터를 탐색하는 KBQA로 구분된다(Fig. 1). KBQA는 지식 트리플을 구성하는 노드와 엣지를 이용하여 데이터를 탐색하는 방법론인데 간혹 KB의 데이터가 완전하지 않더라도 노드, 엣지를 GNN (Graph Neural Network)이나 분산 표현 방법으로 임베딩하여 데이터를 보완할 수 있어 QA 분야에서 현재 널리 사용된다(Chen *et al*., 2020;Lan *et al*., 2021).

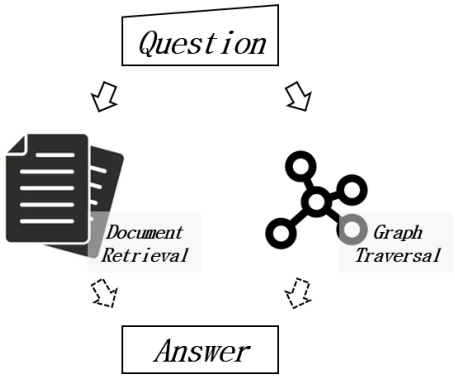

Fig. 1. QA system

GeoQA는 폐쇄 도메인 QA 중 하나로서 공간정보와 관련된 질문에 답할 수 있도록 고안된 QA 시스템이다. 예를 들어, “춘천시와 인접한 도시는?”이라는 질문에 대해 일반적인 검색엔진은 답변을 제공하지 못한다. 또한, 검색엔진은“마포역에서 가장 가까운 이발소는?”이라는 질문이 입력되었을 때 블로그나 인스타그램 등에 사용자가 게시한 정보를 이용하여 답변을 생성하므로 실제 정답과는 차이가 있다. 즉 GeoQA 시스템에서는 사용자의 질문을 이해하고 정보를 탐색, 반환하는 과정 이외에도 객체들의 공간적 관계를 분석하는 과정이 요구된다. 예를 들어, Hamzei *et al*.(2022)는 영국과 인근 지역에 대해 200여 개의 문장으로 이루어진 공간 관련 질문에 답할 수 있는 GeoSPARQL 생성 방법을 제시하였는데 생성된 GeoSPARQL에는 NEAR, CLOSEST_TO, CROSS 등 객체의 공간적 특성을 판단할 수 있는 함수들이 포함되었다.

반면, Regalia *et al*.(2019)에 의하면 실제 환경에서 공간 연산은 정확도를 담보하기 어렵고 컴퓨팅 자원의 소모가 많아 즉각적인 답을 생성하는 것이 어려우므로 많은 GeoQA 시스템에서는 필요한 객체 간 공간 연산을 미리 수행하여 KB를 제작한 뒤 그래프 탐색만으로 답변을 생성하는 방법을 채택하고 있다. 실제로 Nam and Kim(2015)은 공간 객체 간 포함 관계를 미리 정의하고 상대적 방위를 계산하여 미국의 행정구역간 공간적 관계를 파악할 수 있는 시스템을 제작하였다. Wang .(2019)은 중국 난징 지역의 자연환경 및 도시 변화에 대한 시공간 KB를 제작하였으며 Liu *et al*.(2021)는 도시에 존재하는 POI (Point-of-Interest) 및 행정구역간의 관계를 미리 데이터베이스에 저장하고 SPARQL을 이용하여 데이터를 조회하는 시스템을 구현한 바 있다.

2.2 LPG

LPG(Neo4j, 2017)는 객체를 라벨(label) 및 속성(property)으로 표현하는 그래프 데이터베이스를 의미한다(Fig. 2). LPG의 라벨은 객체가 속한 집단을 구분하는 데 사용되며 모든 객체는 0개 이상의 라벨을 이용하여 표현된다. 속성은 키-값 형태로 관리되며 LPG의 라벨 및 속성을 이용하면 효율적으로 데이터의 저장 및 검색을 수행할 수 있다. 관계형 데이터베이스와는 다르게 LPG는 엄격한 스키마를 요구하지 않으므로 같은 라벨을 가진 객체라도 서로 다른 키-값 쌍을 가질 수 있어 자유로운 데이터 표현이 가능하다. 또한, 객체 간 관계를 데이터베이스 내에서 그대로 표현할 수 있으므로 조인(join) 연산을 수행할 필요가 없어 데이터 간 복잡한 관계를 탐색하는 데 효율적이다.

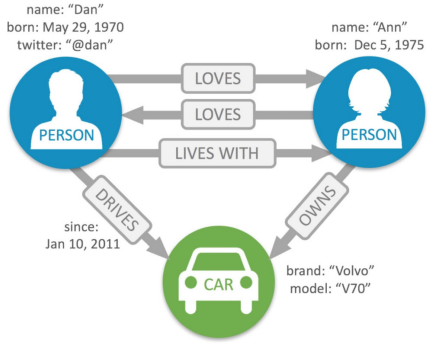

Fig. 2. Labeled property graph

2.3 GraphQL

GraphQL은 별도의 서버를 통해 API 방식으로 데이터베이스에 접근할 수 있는 쿼리 언어이다. GraphQL은 그래프 데이터베이스 이외에도 대부분 데이터베이스와 호환되며 Restful API 방식이 아닌 Query/Mutation 방식을 사용한다. Query를 이용하면 데이터베이스 내에 있는 데이터를 조회 가능하며 Mutation은 데이터를 생성, 수정, 삭제하는 기능을 제공한다. Fig. 3에서 볼 수 있듯이 Query는 매우 간단한 객체 이름과 속성들로 구성되어 있으며 GraphQL을 이용하여 실제 데이터베이스에 접근할 때는 별도의 스키마 파일이 필요하다.

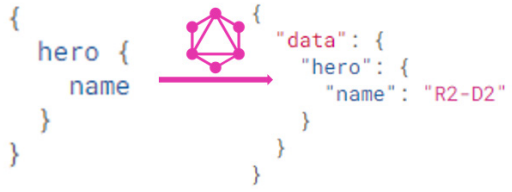

Fig. 3. GraphQL query example

3. GeoQA 파이프라인 구현 및 실험결과

이 장에서는 공간 객체를 미리 LPG에 저장하고 공간 관련 질문을 GraphQL로 변환한 뒤 저장된 정보를 탐색, 조회할 수 있는 GeoQA 파이프라인 및 결과를 제시하였다(Fig. 4). 먼저 GeoQA 시스템에 입력될 수 있는 공간 관련 질문 목록을 작성하였으며 의존성 분석 방법으로 질문을 분석하였다. 또 분석 결과를 이용해 LPG에 적재할 공간 객체들의 관계를 미리 정의하였으며 질문을 정해진 규칙에 따라 FOL 및 GraphQL로 변환하였다. 마지막으로 구축된 LPG를 이용하여 GraphQL 서버에 필요한 스키마 파일을 자동 생성하였고 쿼리를 호출하여 결과를 확인하였다.

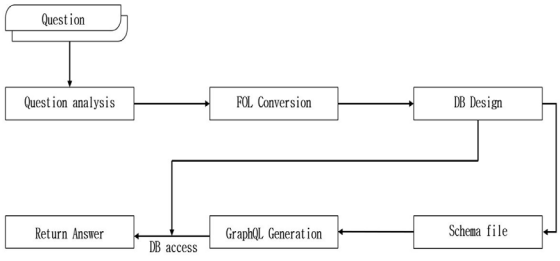

Fig. 4. GeoQA pipeline

3.1 공간 쿼리

3.1.1 공간 관련 질문 목록 구축

GeoQA는 폐쇄 도메인 QA로, 일반적인 QA 시스템과 비교하면 전문적인 대답을 제공할 수 있지만 처리할 수 있는 질문의 범위도 제한적이다. 공간 관련 질문이란 공간 내에서 식별 가능한 객체의 특성이나 객체 사이의 공간적 관계를 질문하는 것인데 GeoQA 분야에서는 크게 두 가지 질문 목록을 채택하고 있다. 먼저 Hamzei *et al.*(2019)가 분석한 바 있는 MS-MARCO의 공간 관련 질문은 주로 도시와 POI에 대한 질문으로 구성되어 있으며 KRR&A TEAM(2018)이 제작한 GeoQuestion201은 자연지형, 공간 객체 및 도시 등 다양한 질문으로 구성되어 있다. 이 중 GeoQuestion201은 Punjani *et al.*(2018)의 연구와 Hamzei *et al.*(2022)의 연구에서 사용된 바 있으며 MS-MARCO의 공간 관련 질문보다 더 다양한 공간 객체를 다루고 있어 본 연구에서도 GeoQuestion201의 질문을 참조하였다. Table 1에 GeoQuestion201에 있는 각 문장의 장소 및 행정구역을 대한민국의 지명으로 치환한 결과를 나타내었다.

Table 1. Example sentences in GeoQuestion201

Sentences from GeoQuestion201	Korean Sentences
Which restaurants are near Edinburgh Castle?	서울시청 근처에 있는 식당을 알려주세요
Which is the largest lake in England?	한국에서 가장 큰 호수는?
Which universities are in England?	서초구에 있는 대학교 이름은?

3.1.2 문장 분석

본 연구에서는 GeoQA를 구현하기 위해 객체들 사이의 공간적 관계를 미리 정의하고 LPG 내에 구축하는 방법을 채택하였다. 따라서 질문 유형을 객체 간 공간적 관계에 따라 분류하고 각 유형에 대응할 수 있도록 키워드를 분석하였다.

Table 2. FOL conversion with spatial keywords

Korean Sentences	Spatial Relationship Extraction from FOL
천안에 인접한 도시는?	x0: PLACE(천안) ∧ CITIES(x0) ∧ BORDER(x0, 천안)
인천에 있는 가장 큰 아파트는?	BIGGEST(x0): PLACE(인천) ∧ APARTMENT(x0) ∧ IN(x0, 인천)
용산구에 있는 음식점은?	x0: PLACE(용산구) ∧ RESTAURANTS(x0) ∧ IN(x0, 용산구)

Table 3. Spatial Relationship sorted by frequency

Spatial Relationship	Frequency	Spatial feature	Selected
IN	108	Polygon ↔ Point	○
NEAR / CLOSE	26	Point ↔ Point	○
LARGEST / LONGEST / BIGGEST / HIGHEST	20	Property(order by)	○
BORDER + EAST / NORTH / WEST / SOUTH	20	Polygon ↔ Polygon	○
COUNT	16	Property(count)	×
CROSS / CROSSES	15	Polygon ↔ Polygon	×

우선 각 문장에 대해 의존성 분석을 수행하여 주요 객체 및 공간 관계를 추출하였고 이를 이용하여 FOL 구조를 작성하였다(Table 2).

공간 관계의 빈도수에 따라 분석을 진행한 결과는 Table 3에 나타내었다. 우선 특정 POI가 행정구역 내에 속하는 것을 의미하는 IN 키워드가 가장 높은 빈도수를 나타내었으며 다음으로는 서로 가까운 POI 사이에 사용할 수 있는 NEAR, CLOSE가 뒤를 이었다. 또한, 공간 연산은 아니지만, POI 간의 속성을 비교하고자 하는 질문도 있었다. 본 논문에서는 공간 연산에 필요한 데이터의 유형에 따라 대표적인 연산을 선택하여 프레임워크를 구현하였다. 예를 들어 IN 연산은 점(point)과 폴리곤(polygon)이 수행되는 대표적인 연산이므로 선택되었다. 반면, 행정구역간 경계를 나타내는 BORDER와 행정구역과 하천의 관계를 나타내는 CROSS는 폴리곤 간 연산에 해당하므로 더 높은 빈도를 나타낸 BORDER를 채택하였다. 마찬가지로 속성을 조회하는 COUNT도 제외하였다.

3.2 데이터베이스 구축

본 논문에서는 현재 가장 점유율이 높은 LPG인 Neo4j(4.0.7 Enterprise Edition)에 대한민국의 행정구역 및 POI 정보 및 공간적 관계를 미리 계산하여 질문에 대답할 수 있는 시스템을 구축하였다. 3.1절에서 결정한 공간 키워드는 각각 BORDER, IN, NEAR/CLOSE이며 Neo4j 데이터베이스에서는 노드 사이의 관계(relationship)로 표현된다. 사용된 데이터 및 관계 구축에 대한 구체적인 내용은 다음과 같다.

- BORDER(행정구역간 인접 정보) : 행정구역정보는 국가공간정보포털에서 획득하였으며 QGIS INTERSECTION 연산을 이용하여 시군구 shape file 내 각 폴리곤의 인접 폴리곤 목록을 생성하였다. 각 광역시 및 특별시의 구는 시, 군과 따로 구분하지 않고 함께 처리하였으며 각 폴리곤은 점 객체로 이루어진 리스트로 저장되었다.
- IN(행정구역과 POI의 포함관계) : 각 행정구역에 속한 POI를 파악하고 Neo4j에 포함관계를 정의하기 위해 행정구역 폴리곤과 POI를 나타내는 점에 대해 INTERSECTION 연산을 수행하였다. 연산에는 Neo4j spatial plugin의 spatial.algo.withInPolygon 함수가 사용되었다.
- NEAR/CLOSE(POI 간 가까운 정도) : 구축된 LPG가 Table 4와 같이 Oh(2006)가 제안한 국내 POI 대분류 7종을 대표할 수 있도록 POI를 선정하였다. 대학교, 아파트, 은행 데이터는 국가공간정보포털에서, 음식점과 호텔 데이터는 지방행정 인허가데이터 개방 서비스에서, 주차장과 병원은 공공데이터 포털에서 각각 내려받았으며 경위도 좌표 및 상호를 속성으로 입력하였다. 본 연구에서는 POI 처리 시간을 고려하여 서울지역에 대해서만 NEAR/CLOSE 연산을 수행하였다.

Table 4. POI categories and representation

POI category	Representation
Transportation	Car park
Living	Restaurant
Travel/Leisure	Hotel
Medical treatment	Hospital
Finance	Bank
Public/Administration	University/College
Company/Group	Apartment

3.3 GraphQL 변환 및 스키마파일 제작

3.3.1 스키마 파일 제작

GraphQL 요청은 변수 및 LPG내에 존재하는 (객체)-[공간관계]→(객체) 구조의 추상화된 형태이며 GraphQL 서버는 GraphQL 스키마 파일을 이용하여 데이터베이스에 직접 접근, 데이터를 조회할 수 있다. 각 LPG에 적합한 스키마 파일은

별도의 라이브러리를 이용하여 자동으로 제작된다(Fig. 5).

```
type Bank {
        restaurantsNear: [Restaurant!]! @relationship(type: "NEAR", direction: IN)
        nearParkings: [Parking!]! @relationship(type: "NEAR", direction: OUT)
        addrCountry: [String]
        addrFloor: [String]
}

type City {
        belongToCities: [City!]! @relationship(type: "belong_to", direction: OUT)
        borderCities: [City!]! @relationship(type: "Border", direction: OUT)
        capital: [String]
        description: [String]
}

type College {
        inCities: [City!]! @relationship(type: "IN", direction: OUT)
        nearRestaurants: [Restaurant!]! @relationship(type: "NEAR", direction: OUT)
        description: [String]
        label: [String]
        point: Point!
        website: [String]
}
```

Fig. 5. GraphQL schema

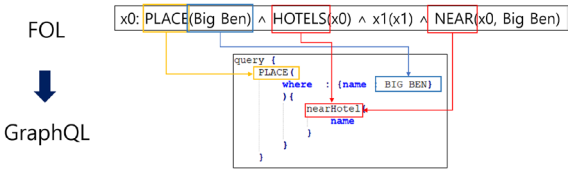

Fig. 6. GraphQL generation from FOL

3.3.2 GraphQL 템플릿 생성 및 GraphQL 생성

먼저 키워드 추출 및 의존성 분석을 통해 사용자의 공간 관련 질문은 FOL로 변환된다. 본 논문에서는 각 FOL에 존재하는 공간적 관계에 관련된 키워드, 공간 객체에 관련된 키워드 등을 이용하여 GraphQL을 제작할 수 있는 템플릿을 제시하였다(Table 5). 각 템플릿에 적합한 객체 및 속성, 공간 관계는 FOL로부터 다시 한번 추출되고 정해진 규칙에 따라 GraphQL 쿼리가 생성되었다(Fig. 6).

3.4 실험결과

3.4.1 데이터베이스 구축 결과

대한민국 지역에 대해 KB를 구축한 결과를 Table 6에 나타내었다. 다양한 포털 및 서비스로부터 내려받은 POI 및 행정구역 데이터가 잘 입력되었음을 확인하였다. 노드 타입은 총 9개로 8개는 앞서 언급한 POI 7종과 행정구역을 의미하고 나머지 노드는 속성정보를 조회하기 위해 각 POI 종류를 담고 있다. 관계 종류는 4가지이며 공간 연산을 수행한 결과 및 POI 종류를 나타낸다. Fig. 7에는 구축한 KB 일부를 시각화하였다.

Table 5. GraphQL templetes

Keywords	Example sentences and templates	variable
IN	"올림피아 호텔은 어떤 도시에 있어?" `query { ${entity} (where : {name : ${name}}){ inCities{ name } } }`	entity : POI type name : POI name
NEAR	"신림 현대아파트에서 가까운 병원은?" `query { ${entity} (where : {name : ${name}}){ near${feature}{ name } } }`	entity : POI type feature : target POI type name : POI name
Property (LARGEST /HIGHEST)	"서울에서 가장 높은 산 이름은?" `query { ${entity} (options : {limit : 1, sort : { ${property} : DESC}}){ name ${property} } }`	entity : POI type property : property(size, height, area, etc.)
BORDER	"경주와 인접한 도시는 뭐야?" `query { cities( where : {name : ${name}}){ borderCities{ name } } }`	name : name of cities

Table 6. Neo4j Instance statistics

	# of nodes	# of node types	# of relationship	# of relationship types
Neo4j Instance	533,918	9	63,126	4

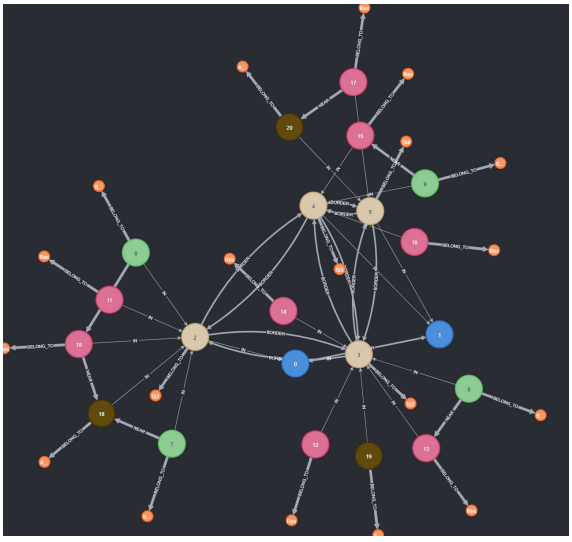

Fig. 7. Neo4j instance

3.4.2 GraphQL 쿼리 결과

본 연구에서는 GraphQL서버로 Apollo를 사용하였으며 Javascript 라이브러리인 @neo4j/introspector을 이용하여 GraphQL 스키마 파일을 생성하였다. 설계된 파이프라인의 동작 여부를 확인하기 위해 Neo4j 데이터베이스에 호출하고 결과를 확인하였다(Table 7). 실험 결과 본 연구에서 정의한 3가지 공간 키워드인 IN, BORDER, NEAR에 관한 GraphQL 생성 후 속성그래프에 쿼리한 결과 POI 및 행정구역 이름이 정확하게 반환됨을 확인하였다. 다만 QGIS의 Intersects 함수를 사용하여 폴리곤 관계를 정의한 BORDER 관계가 일부 객체에서 빠진 것을 확인할 수 있었다. 예를 들어 과천시와 인접한 도시를 조회한 결과 안양시나 의왕시가 반환되지 않았다. 또 Fig. 8.과 같이 서울의 관악구나 부산의 연제구 등 특별시나 광역시의 구가 City로 정의되어 있어 광역자치단체의 행정구역상 위계관계를 나타내지 못하는 한계를 보여 GIS 함수의 처리 결과에 대한 유효성 검사나 행정구역의 세분화 등 데이

Table 7. GraphQL query examples and results

Query Example	Result
`query { hotels(where : {name : "올림피아 리조트"}){ inCities{ name } } }`	`{ "data": { "hotels": [ { "inCities": [ { "name": "평창" }`
`query { cities (options : {limit : 1, sort : {population:DESC}}){ name population } }`	`{ "data": { "cities": [ { "name": "서울", "population": 9509458 } ]`

터베이스 구축방법에 개선이 필요함을 알 수 있었다.

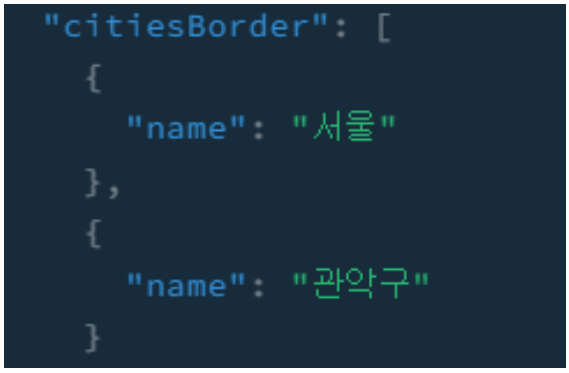

Fig. 8. Border relationship error

4. 요약 및 결론

본 연구는 다양한 사용자들이 독자적으로 구축한 LPG에 대해서도 동작할 수 있도록 GraphQL서버를 도입한 GeoQA 시스템을 제안, 구축하였다. LPG 구축을 위해서는 공간 관련 질문을 분석, 수요가 높은 질문을 추출하고 질문의 유형을 분류하였다. 구축된 LPG에는 객체 간 관계가 미리 정의된 KB가 저장되어 있어 질문이 입력될 때마다 연산을 수행하지 않고 즉각적인 답을 반환할 수 있도록 하였다. KB를 구축한 범위는 대한민국 전역이며 국토정보포털 및 인허가데이터로부터 행정구역 및 POI 데이터를 취득하여 활용하였다. LPG로는 현재 그래프 데이터베이스 시장에서 가장 점유율이 높은 Neo4j를 선택하였으며 데이터 구축을 위해 내장 함수 및 QGIS를 이용하여 공간 연산을 수행하였다. 제안된 GeoQA 시스템은 입력된 공간 관련 질문을 먼저 FOL 구조로 1차 변환하고 미리 정해진 규칙에 따라 최종적으로 GraphQL로 변환한다. 그리고 Apollo GraphQL 서버 및 스키마파일을 이용하여 생성된 GraphQL 쿼리를 데이터베이스에 입력하고 결과를 조회한다. 실험결과 자연어 질문이 입력되었을 때 목적에 부합하는 결과가 반환되는 것을 확인하였다. 본 연구에서 제안한 자연어처리 방법을 통해 생성된 GraphQL 쿼리는 Neo4j 뿐만 아니라 Dgraph, TigerGraph등 대부분의 LPG에서도 동작하므로 표준화된 쿼리 언어가 없는 LPG 진영의 확장성을 높였다는 점에서 차별화된다. 다만, 향후 엄밀한 공간적 연산 방법 및 정확한 개체명 인식기능을 개발하여 완성도 높은 KBQA 시스템을 구현하는 것이 요구되며 GraphQL을 이용한 다양한 활용사례에 관한 실증연구도 필요할 것이다.

감사의 글

본 연구는 국토교통부/국토교통과학기술진흥원의 지원으로 수행되었음(과제번호 RS-2022-00143336).

References

Chen, X., Jia, S., and Xiang, Y. (2020), A review: Knowledge reasoning over knowledge graph, *Expert Systems with Applications*, Vol. 141. https://doi.org/10.1016/j.eswa.2019.112948

Hamzei, E., Li, H., Vasardani, M., Baldwin, T., Winter, S., and Tomko, M. (2019), Place questions and human-generated answers: A data analysis approach, *International Conference on Geographic Information Science,* pp. 3-19. Springer, Cham. https://doi.org/10.1007/978-3-030-14745-7_1

Hamzei, E., Tomko, M., and Winter, S. (2022), Translating place-related questions to GEOSPARQL queries. *In Proceedings of the ACM Web Conference,* pp. 902-911. https://doi.org/10.1145/3485447.3511933

KRR&A TEAM (2018), Athens, *Department of Informatics and Telecommunications, National and Kapodistrian University of Athens,* Athens 157 72 Greece, http://pyravlos-vm5.di.uoa.gr/geoqa-old/benchmarkquestions.html (last date accessed: 10 September 2022).

Lan, Y., He, G., Jiang, J., Jiang, J., Zhao, W. X., and Wen, J. R. (2021), Complex knowledge base question answering: A survey. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence,* pp. 4483-4491. https://doi.org/10.48550/arXiv.2108.06688

Lee J., Joo Y., and Park, S. (2012), Design and implementation of context awareness inference system based on ontology - focusing on tour information guidance smartPhone application, *Journal of the Korean Society for Geospatial Information System*, Vol.20, No.4, pp. 67-75. (in Korean with English abstract) https://doi.org/10.7319/kogsis.2012.20.4.067

Liu, J., Liu, H., Chen, X., Guo, X., Zhao, Q., Li, J., and Liu, J. (2021), A heterogeneous geospatial data retrieval method usingknowledgegraph,*Sustainability*,Vol.13,No.4,pp.1-21.

https://doi.org/10.3390/su13042005

Mai, G., Janowicz, K., Zhu, R., Cai, L., and Lao, N. (2021), Geographic question answering: challenges, uniqueness, classification, and future directions, *AGILE: GIScience Series*, Vol. 2, pp. 1-21. https://doi.org/10.5194/agile-giss-2-8-2021

Nam, Sangha and Kim, Incheol. (2015). Design and implementation of a hybrid spatial reasoning algorithm, *Journal of KIISE*, Vol.42, No.5, pp. 601-608. (in Korean with English abstract) https://doi.org/10.5626/JOK.2015.42.5.601

Neo4j (2017), *Neo4j,* San Mateo, CA 94401, USA, https://neo4j.com/developer/graph-database/ (last date accessed: 10 September 2022).

Oh, S. (2006), *Infra 21 seminar-Korea POI Construction Status and Future Direction*, Report 291, Korea Research Institute for Human Settlements, Sejong, pp. 152-157

Punjani, D., Singh, K., Both, A., Koubarakis, M., Angelidis, I., Bereta, K., and Stamoulis, G. (2018), Template-based question answering over linked geospatial data, *Proceedings of the 12th Workshop on Geographic Information Retrieval*, pp. 1-10. https://doi.org/10.1145/3281354.3281362

Regalia, B., Janowicz, K., and McKenzie, G. (2019). Computing and querying strict, approximate, and metrically refined topological relations in linked geographic data. *Transactions in GIS*, Vol. 23, No. 3, pp. 601-619. https://doi.org/10.1111/tgis.12548

Wang, S., Zhang, X., Ye, P., Du, M., Lu, Y., and Xue, H. (2019). Geographic knowledge graph (GeoKG): a formalized geographic knowledge representation, *ISPRS International Journal of Geo-Information*, Vol. 8, No. 4, pp.184-207. https://doi.org/10.3390/ijgi8040184