Journal of Korea Multimedia Society Vol. 22, No. 9, September 2019(pp. 1036-1045)
https://doi.org/10.9717/kmms.2019.22.9.1036

관계형데이터를 이용한 그래프 데이터베이스의 모델별 구조 분석과 쿼리 성능 비교 연구

배석민[†], 김진형[††], 유재민[†††], 양성열[††††], 정재진[†††††]

Structural Analysis and Performance Test of Graph Databases using Relational Data

Bae Suk Min[†], Kim Jin Hyung[††], Yoo Jae Min[†††], Yang Seong Ryul[††††], Jung Jai Jin[†††††]

ABSTRACT

Relational databases have a notion of normalization, in which the model for storing data is standardized according to the organization's business processes or data operations. However, the graph database is relatively early in this standardization and has a high degree of freedom in modeling. Therefore various models can be created with the same data, depending on the database designers. The essences of the graph database are two aspects. First, the graph database allows accessing relationships between the objects semantically. Second, it makes relationships between entities as important as individual data. Thus increasing the degree of freedom in modeling and providing the modeling developers with a more creative system. This paper introduces different graph models with test data. It compares the query performances by the results of response speeds to the query executions per graph model to find out how the efficiency of each model can be maximized.

Key words: Graph Database, Graph Model Structure, Performance Test

1. 서 론

관계형 데이터베이스는 정규화라는 개념이 있어 데이터를 저장하는 모델이 조직의 업무 프로세스 혹은 데이터 운용 목적에 따라 데이터베이스 모델의 구조가 보편적으로 표준화되어 있다. 하지만, 그래프 데이터베이스는 상대적으로 이런 표준화가 초기에 있고 모델링의 자유도가 높기 때문에 기획하는 사람에 따라 같은 데이터로도 다양한 모델을 생성할 수 있다.

그래프 데이터베이스는 정점과 간선의 집합으로 이루어지며, 경로 탐색 중심으로 데이터를 조직한다. 그래프는 개체를 정점으로 나타내며 이러한 개체들 간의 관계를 실제 세계와 직접적으로 관련된 방식으로 나타낸다. 그래프 데이터베이스는 개체간의 연결 관계에도 직접적인 의미를 부여함으로써 개체 간 연

※ Corresponding Author : Jung Jai Jin, Address: (16890) 152, Jukjeon-ro, Suji-gu, Yongin-si, Gyeonggi-do, Republic of Korea, TEL : +82-31-8005-2497, E-mail : dothan@dankook.ac.kr
Receipt date : Aug. 13, 2019, Revision date : Sep. 16, 2019
Approval date : Sep. 19, 2019
[†] Dept. of Future ICT Convergence Engineering, Graduate School, Dankook University
(E-mail : 72160340@dankook.ac.kr)
[††] Dept. of Future ICT Convergence Engineering, Graduate School, Dankook University
(E-mail : jinhyung.kim91@gmail.com)
[†††] Dept. of Future ICT Convergence Engineering, Graduate School, Dankook University
(E-mail : 72151704@dankook.ac.kr)
[††††] Dept. of Future ICT Convergence Engineering, Graduate School, Dankook University
(E-mail : 72171408@dankook.ac.kr)
[†††††] Dept. of Applied Computer Engineering, Dankook University

결 탐색에서 논리적 구조를 반영할 수 있다. 이는 기존 관계형 데이터베이스에서 사용하는 테이블 간의 단순 연결이 아닌, 연결 관계에도 라벨과 데이터를 동시에 저장함으로써 개체 간의 관계에도 세부적인 의미를 부여할 수 있다. 이러한 범용적 표현 구조는 다양한 시나리오를 모델링 할 수 있게 해준다. 데이터 설계자마다 다른 접근의 그래프 모델링을 한다. 본 논문에서 소개하는 그래프 모델은 관계형 데이터를 가장 단순하게 변환하여 그래프 모델에 적용한 모델, 관계형 데이터를 업무적 관점에서 그래프 모델로 재구성한 모델, 그래프 모델을 RDF 모델처럼 정규화 하여 분할한 모델이다[1,2].

그래프 모델링을 처음 접하는 데이터 설계자는 기존 모델링이 다양한 만큼 오히려 어떤 모델을 사용해야 할지 혼돈이 생길 수 있다. 본 논문에서는 다수의 그래프 모델의 쿼리 성능을 비교하고 결과를 제공함으로써 그래프 모델에 따른 쿼리 실행에 대한 응답 속도의 차이를 통해 각 모델의 효율성이 얼마나 극대화 되었는지를 알아보고자 한다.

성능 측정에 쓰인 데이터 세트는 Northwind로 데이터베이스에서 가장 많이 사용되는 샘플 데이터 세트를 각각의 모델에 맞게 마이그레이션 하였다. 실험은 정확성을 위하여 모든 쿼리를 20번씩 측정하여 평균값을 산출하였다. 이 연구의 결과는 그래프 모델링 기법에 효율성을 높이고 운영 목적에 적합한 모델링을 선택하는 것에 안내를 마련하고자 한다.

2. 이 론

2.1 관계형 데이터베이스의 정규화

관계형 데이터베이스의 정규화 목적은 데이터베이스 내에 이상이 있는 관계를 재구성함으로써 데이터가 불일치되는 위험을 최소화하여 바람직한 구조로 만들어 가는데 있다. 제1 정규형에서 제5 정규형을 거치면서 데이터베이스 내의 단일 테이블은 수많은 작은 테이블로 분할되어 어떤 구조로 데이터를 저장하는가에 대한 구조화를 갖춘다.

정규화는 데이터베이스의 데이터를 구성하는 프로세스이다. 이 프로세스에는 중복성 및 일치하지 않는 종속성을 제거하여 데이터베이스의 유연성을 높이는 동시에 데이터를 보호하도록 설계된 규칙에 따라 테이블을 만들고 해당 테이블 간의 관계를 설정하는 작업이 포함된다. 데이터가 중복되면 디스크 공간이 낭비되며 유지 관리상의 문제가 발생한다. 또한 중복된 정보로 인해 갱신 이상이 발생하게 되며, 여러 위치에 있는 데이터를 변경해야 하는 경우에는 모든 위치에서 데이터를 정확히 동일한 방식으로 변경해야 한다. 이러한 문제들을 해결하기 위해 정규화를 한다[3].

2.2 그래프 데이터베이스 시스템

그래프 데이터베이스는 정점과 간선의 집합으로 이뤄진다. 보다 쉽게 표현하면 정점들의 연결하는 간선은 정점들 간의 관계를 표현한다. 그래프는 개체를 정점으로 나타내며 이러한 개체들 간의 관계를 실제 세계와 직접적으로 관련된 방식으로 나타낸다. 이러한 범용적 표현 구조는 다양한 시나리오를 모델링 할 수 있게 해준다[4].

그래프 데이터베이스의 그래프는 특정 엣지의 유형 또는 전체 그래프를 전반을 통하여 트래버스 될 수 있다. 그래프 데이터베이스에서 노드 간의 관계는 쿼리 시간에는 포함되지 않지만 데이터베이스에서 유지되기 때문에 조인 또는 관계를 트래버스하는 속도가 매우 빠르다. 그래프 데이터베이스는 데이터 간의 관계를 만들고 이러한 관계를 신속하게 쿼리해야 하는 소셜 네트워킹, 추천 엔진, 이상 탐지 등의 사용 사례에 유용하다[5].

2.3 선행 연구 – Arango의 성능 측정

Arango의 테스트에서는 소셜 네트워크 사이트에 일반적으로 사용되는 데이터 집합을 사용자 프로필과 관계를 사용했다. 가능한 모든 데이터베이스 작업을 측정하지는 않았으며, 그보다는 소셜 네트워크에 적합한 쿼리에 중점을 두었다. 예를 들어, 단일 읽기 및 쓰기 프로필을 수행하고, 연령 분포에 대한 개요를 얻기 위해 집계를 계산하거나, 이웃의 이웃 관계를 요청하거나, 최단 경로 등의 쿼리를 사용하였다. 이 쿼리는 내부적으로 사용하는 데이터 모델과 상관없이 테스트를 거친 모든 데이터베이스에 대해 실행되었으며 결과적으로 특수 솔루션과 다중 모델 데이터베이스 간에 성능을 비교할 수 있었다[6].

테스트 데이터베이스는 ArangoDB, MongoDB, Neo4j이며, 테스트 데이터는 슬로바키아의 소셜 네트워크 스냅 샷이며 Stanford University SNAP 컬

렉션에서 제공한다. 해당 데이터는 사람들을 묘사하는 1632803 개의 정점과 그들의 관계를 묘사하는 30622622개의 간선을 포함한다. 테스트를 위해 워크로드를 처음부터 5회 실행하여 결과를 평균화 하였다. Fig. 1은 테스트 결과이다[6].

2.4 Hongcheng Huang의 성능 측정

Hongcheng Huang는 그래프 데이터베이스인 Neo4j를 통해 쿼리 성능 측정을 하였다. 그래프 모델 기반의 Neo4j 데이터베이스는 구현 시 일반적인 관계형 데이터베이스와 다르다. Neo4j는 세 가지 쿼리 방법을 제공하며 데이터 크기, 쿼리 복잡성, 쿼리 번호 등 여러 차원에서 쿼리 성능을 평가했다. 결과는 여러 시나리오에서 성능에 분명한 차이가 있음을 보여준다[7].

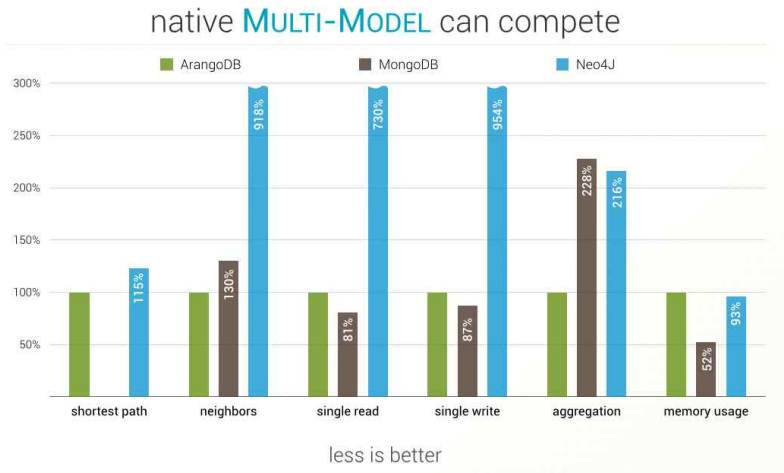
Neo4j의 쿼리 메서드에는 Java Core API, Traverser Frame Work 및 Cypher 쿼리 언어가 포함된다. Neo4j의 쿼리 시간은 100 개의 객체 데이터 세트에서 MySQL보다 2~5배, 500 개의 객체 데이터 세트에서 15~30 배 더 낮았다. 그리고 그래프 모델이 관계형 모델보다 우수함을 보여준다. 또한, Neo4j가 독립형 모드보다 임베디드 모드에서 실행될 때 쿼리 성능이 높음을 보여준다. 이 논문 마지막에서는 데이터 크기, 관계의 복잡성 및 쿼리 안정도 등 다양한 쿼리 방법의 성능을 평가하고 분석했다. Fig. 2는 테스트 결과이다[7].

3. 테스트 모델

3.1 Neo4j의 그래프 모델

그래프 데이터 모델링은 사용자가 노드와 속성 및 레이블이 있는 관계로 연결된 그래프로 임의의 도메인을 설명하는 프로세스이다. Fig. 3의 그래프 데이터 모델은 Cypher 쿼리 형식의 질문에 응답하고 그래프 데이터베이스의 데이터 구조를 구성하여 비즈니스 및 기술적 문제를 해결하도록 설계되었다.[8]

그래프 데이터 모델링에서는 관계형 데이터 모델



(a)

	shortest path	neighbors	single read	single write	aggregation	memory usage
ArangoDB	100%	100%	100%	100%	100%	100%
MongoDB	n. / a.	130.1%	80.9%	86.8%	228.4%	51.9%
Neo4j	115.0%	918.5%	730.1%	954.1%	215.6%	92.8%

(b)

Fig. 1. Arango's performance test. (a) Result chart. (b) Result table.

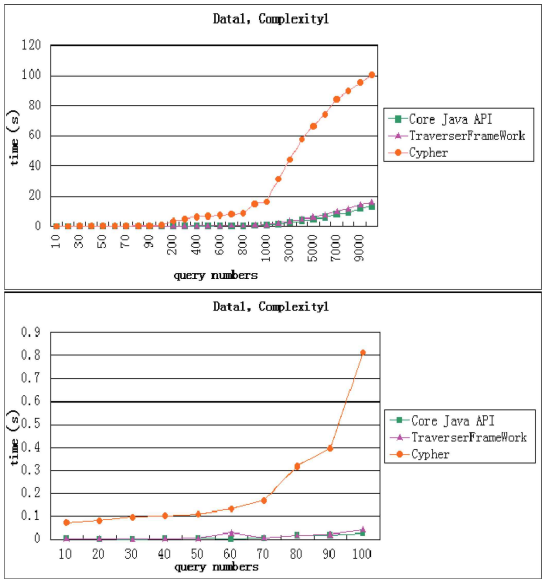


Fig. 2. Hongchaen Huang's performance test.

링과 유사한 프로세스가 존재한다. 그러나 정규화 된 테이블 구조에 맞게 데이터 모델을 수정하는 대신 그래프 데이터 모델은 화이트보드에 그려진 그대로 유지된다. 모든 데이터 모델은 유스 케이스와 사용자가 데이터로 대답해야 하는 질문 유형에 따라 고유하다. 이 때문에 데이터 모델링에 대한 일률적인 방식이 없다. 모범 사례와 신중한 모델링을 사용하면 프로세스와 사용 사례에 도움을 주는 정확한 데이터 모델을 산출하는 데 가장 중요한 결과를 얻을 수 있다고 했다[8].

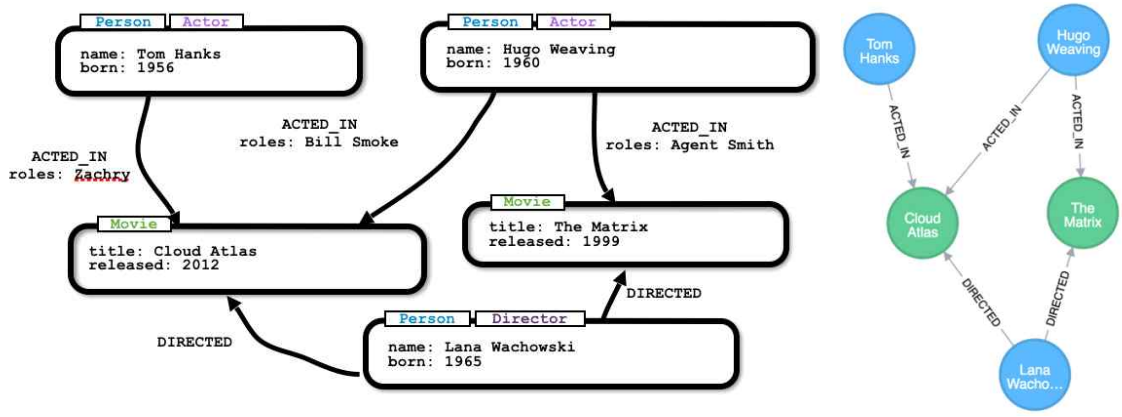
3.2 AgensGraph의 그래프 모델

AgensGraph의 그래프 모델은 테이블을 사용하는 관계형 데이터베이스에서 그래프 데이터베이스로 전환하는 것이다. 관계형 데이터베이스에서 열은 그래프 모델에서 속성을 가진 정점으로, 테이블명은 정점 레이블과 간선 레이블로 하여 Northwind 데이터 세트의 그래프 모델을 Fig. 4로 표현했다[9].

그래프 모델과 관계형 모델의 가장 큰 차이 중 하나는 그래프 모델이 관계형 모델과는 다르게 현실 모델 또는 비즈니스 모델이 있는 그대로 반영한다는 점이다. 관계형 모델은 복잡한 현실 세계의 비즈니스 모델을 가공해 표 형태의 테이블 형식으로 변환한다. 하지만 현실 세계의 비즈니스 모델은 각 객체 간에 관계가 맺어진 형태이고, 유사한 객체끼리의 묶음이 그룹으로 표현되어 있다. 그리고 이는 그래프 모델의 데이터 구현 방식과 매우 유사하다고 했다[9].

3.3 Lance Gutteridge의 그래프 모델

Fig. 5의 양식 노드의 속성은 명명된 값, 즉 필드의 목록이다. 필드에 명명된 값 컬렉션의 목록이 될 수 있는 벡터 유형 필드가 있다. 이들은 하위 양식의 모음으로 간주 될 수 있다. 각 품목에는 제품 코드, 수량 및 가격과 같은 특성이 있다. 이는 양식이 기본적으로 하위 노드이며, 양식 자체로서 기본 노드를 가지며 하나 이상의 첨부된 노드 콜랙션이 있을 수 있음을 의미한다. 이러한 하위 그래프는 유형이 동일하



(a) (b)

Fig. 3. Neo4j's graph model (a)Graph database ERD. (b)Graph model.

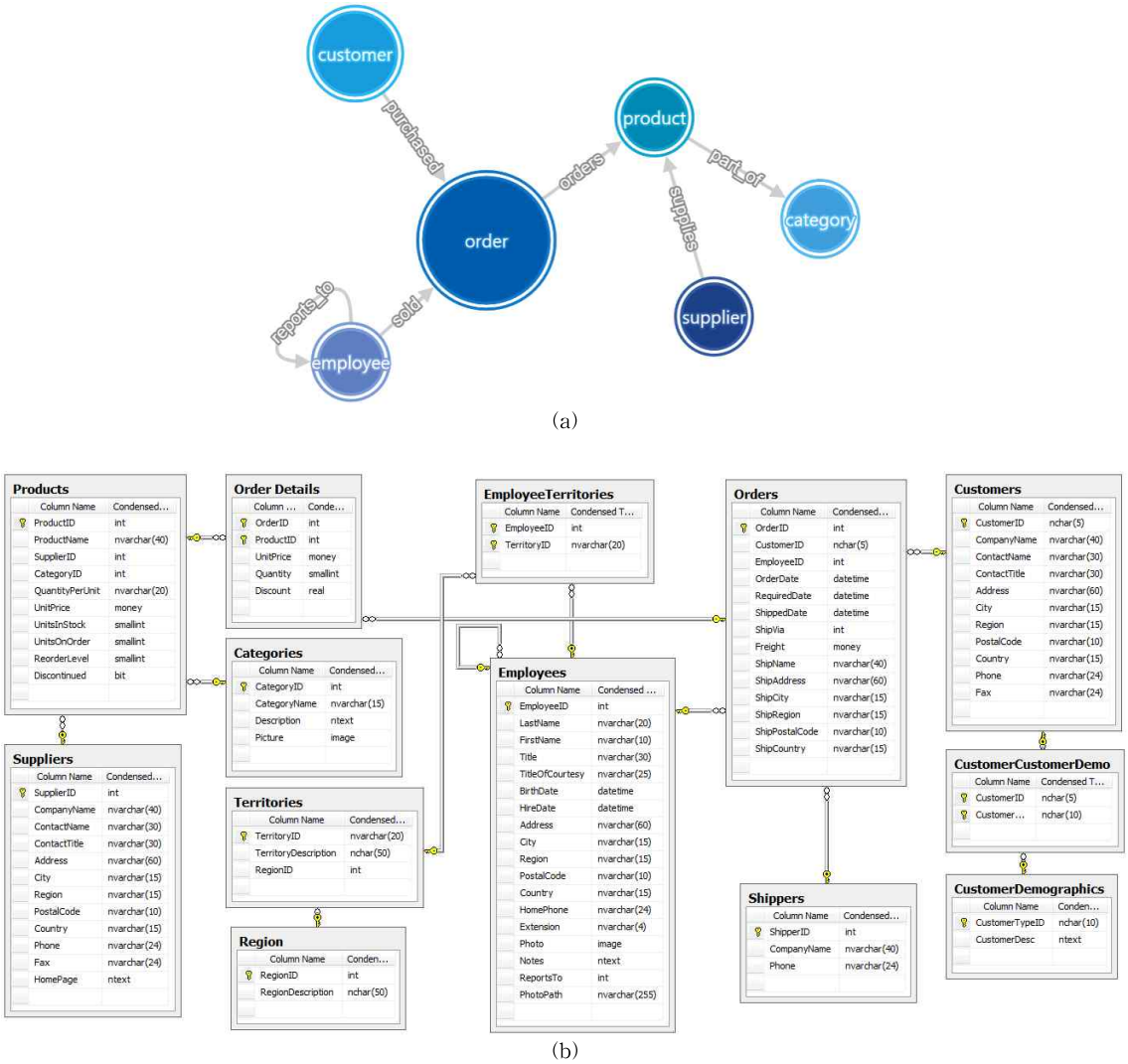


Fig. 4. AgensGraph's graph model (a)Graph model. (b)Relationship database ERD.

다. 즉, 하위 양식 모음의 각 멤버는 다른 멤버와 속성이 동일하다. 각 양식은 실제로 하위 그래프를 정의했다. 일부 양식에는 상세 선이 없으므로 단일 노드이다[10].

Fig. 5에서 검은 색 화살표 속성은 다른 형식의 식별자일 수 있다. 이것은 비즈니스 애플리케이션에서 고유 ID로 수행된다. 녹색 화살표는 양식의 식별자는 다른 식별자로 구성 할 수 있다. 예를 들어, 각 원장에 별도의 계정과 목표가 있을 수 있으므로 총계정 원장 코드 앞에 지점 ID가 있을 수 있다. 따라서 양식에서 ID로 참조되는 양식으로 연결된다. 빨간색 화살표는 양식 노드와 해당 양식의 세부 라인 목록 간의 관계이다. 각 디테일 라인은 또한 노드이며 폼의 최상위 레벨에 연결된다. 파란색 화살표는 상세 선에서 양식의 상단까지의 연결이다. 디테일 라인은 속성으로 폼 코드를 가질 수 있으므로 다른 폼과 연결될 수 있다[10].

3.4 테스트 데이터 세트(Northwind)

Fig. 6은 Northwind 데이터에 접속한 클라이언트 툴의 화면이다. Northwind는 Microsoft에서 제공하는 샘플 데이터 세트로 가상의 상거래 업체의 비즈니

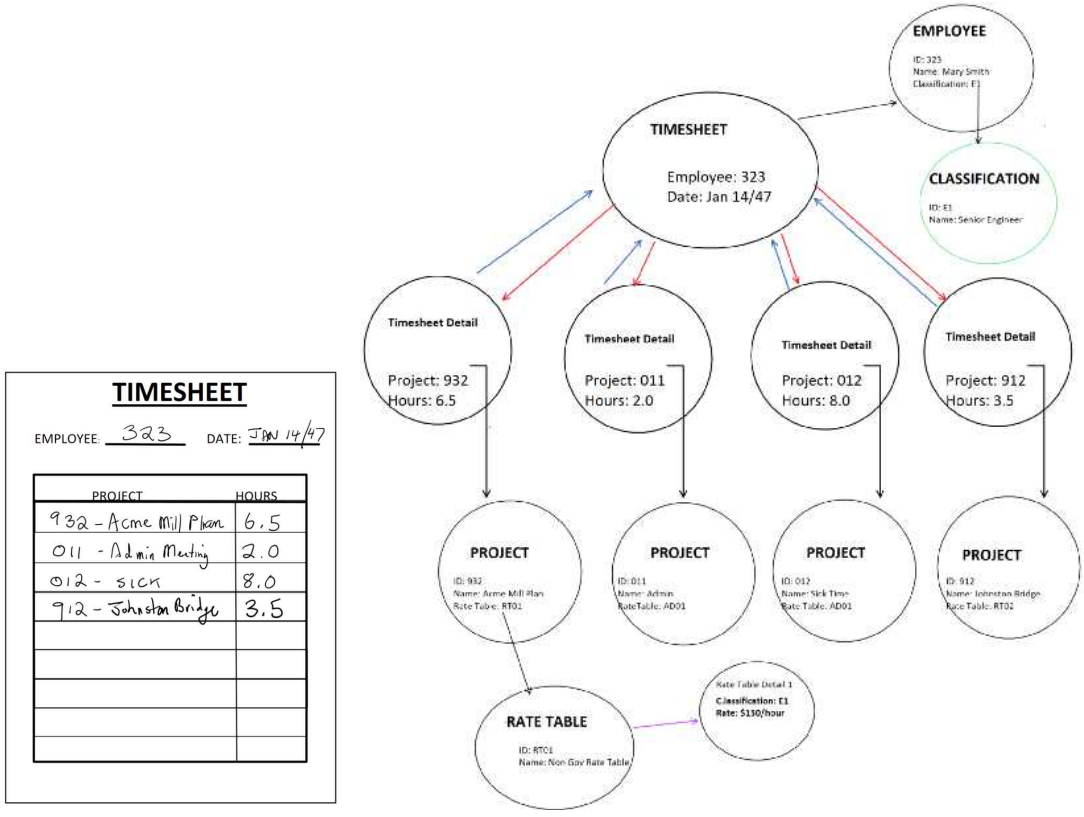


Fig. 5. Lance Gutteridge's graph model (a) Real world data. (b) Graph model.

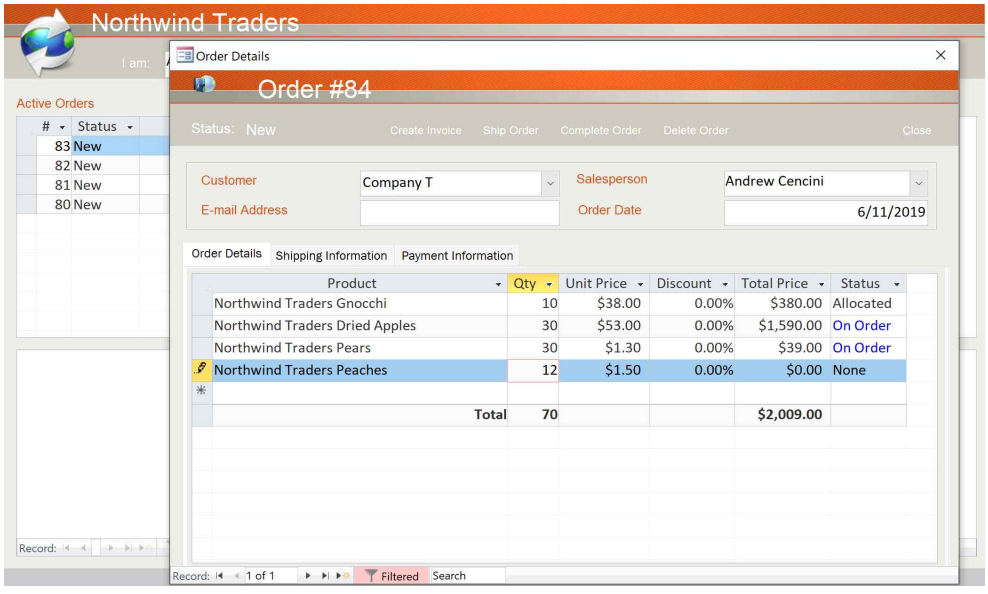


Fig. 6. Northwind data sample.

스를 모델링하였다. 세계적으로 특정 물품을 수출입하는 기업을 가상으로 모델링하여 제작되었다. 따라서 제품의 공급자를 통하여 제품을 공급받고 이를 전 세계의 고객들에게서 주문을 받아 판매하는 데이터가 탑재되어 있다. 매우 간단하고 관련성이 높은 스키마를 가지고 있기 때문에 훌륭한 교육 도구이다. 주문에는 고객과 직원이 있으며 다대일 관계이다. 주문에는 주문 세부 사항, 주문에서 판매된 제품의 품목이 일대다 관계이다[11].

4. 그래프데이터베이스 각 모델별 쿼리 성능

4.1 실험 환경

Table 1은 성능 측정에 사용한 환경 설정 목록이다. Intel i7 8565u 1.8Ghz up to 4.6Ghz의 프로세스와 24GB 메모리 기반의 Windows 10 home 운영체제에 Oracle VirtualBox 6.0 버추얼 머신을 설치하였다. 버추얼 머신 OS로는 Ubuntu 18.04.2 LTS를 설치하였고, 테스트를 진행할 DBMS로는 Agensgraph 2.2를 설치하였다. 쿼리 언어로는 Cypher를 사용하였다.

4.2 성능 측정에 사용할 쿼리

Table 2는 본 논문에서 각 모델별 성능을 측정할 쿼리 목록이다. 모델에 따라 정점, 간선, 속성의 구조가 최대한 다른 형태를 띌 수 있는 쿼리를 사용하였다. 단, Query 3은 모델이 바뀌어도 같은 쿼리를 사용할 수 있는데 이는 성능 측정 시 타 쿼리와 결과를 비교하기 위함이다.

4.3 성능 및 용량 측정 결과

Table 3은 Graph Model 3가지의 성능 및 용량 측정 결과이다. Model A는 관계형 데이터를 가장 단순하게 변환하여 그래프 모델에 적용한 Neo4j의 모델, Model B는 관계형 데이터를 업무적 관점에서 그래프 모델로 재구성한 AgensGraph의 모델이며, Model C는 그래프 모델을 RDF 모델처럼 정규화 하여 분할한 Lance Gutteridge의 모델이다.

각각의 모델은 northwind 데이터 세트를 해당 모델에 맞게 마이그레이션 하여 테스트를 진행하였다. 실험의 정확성을 위해 모든 쿼리는 20번씩 측정하여 평균값을 산출하였으며, 측정 결과 값은 소숫점 3자리에서 반올림하였다.

데이터베이스의 용량은 Model A가 가장 적게 사용하였고, 평균 쿼리 처리 성능 또한 Model A가 가장 우수한 것으로 나타났다. 성능만을 고려한다면 그래프 스키마를 최대한 단순화 시킨 Modela A를 선택하

Table 1. Environment configuration

항 목		내 용
Hardware	CPU	Intel i7 8565u 1.8Ghz up to 4.6Ghz
	RAM	24Gb DDR4 2400Mhz
	Storage	512Gb m.2 nvme
Software	OS	Windows 10 home
	Virtual Machine	Oracle VirtualBox 6.0
Virtual Machine	Configuration	CPU : Core 4 / Ram : 8Gb / Storage : 10Gb
	OS	Ubuntu 18.04.2 LTS
	DBMS	Agensgraph 2.2

Table 2. The queries for the performance test

	쿼리 설명
Query 1	USA에 거주중인 고객이 Germany에 있는 업체의 물건을 주문한 수
Query 2	Seattle에 거주하는 직원의 이름과 해당 직원이 주문을 받은 수
Query 3	전체 주문 중 United Package가 배송한 주문
Query 4	Canada에 배송한 전체 주문 수
Query 5	Sydney에 있는 공급업체가 총 납품한 주문 수

Table 3. Performance test results per graph model

	Model A	Model B	Model C
Data Size	10,864.51Kb	10,912.51Kb	14,543.66Kb
Query 1	7.88ms	12.03ms	278.98ms
Query 2	1.22ms	1.43ms	3.62ms
Query 3	8.69ms	6.75ms	6.48ms
Query 4	2.02ms	1.81ms	2.44ms
Query 5	6.24ms	6.77ms	17.98ms
Query Average	5.21ms	5.76ms	61.90ms

는 것이 맞지만, 그래프 모델링 구조상 엣지를 선택하여 다양한 쿼리를 만들어내는 것은 어렵다. 이는 업무에서 개체를 검색 시 다양한 정점과 간선에 매개변수를 이용하여 검색하는 어려울 수 있다는 것이다.

Model B의 경우 Model A에 비해서는 평균 쿼리 처리 성능이 조금 느리지만 Model C에 비해서는 매우 우수하다. 데이터베이스에서 차지하는 용량도 Model A와 크게 차이가 나지 않는다. 그래프 모델링은 관계형 데이터베이스 모델과 비슷하여 실사용에서 그래프 모델링에 익숙하지 않은 데이터 설계자도 손쉽게 이용할 수 있는 장점이 있다.

Model C는 데이터베이스의 용량도 가장 많이 사용하며, 평균 쿼리 처리 성능도 가장 느리다. 하지만, RDF 모델링 스타일의 특성상 데이터 사용자가 원하는 다양한 쿼리를 구현할 수 있으며, 실사용에서 다양한 정점과 간선에 다수의 매개변수를 이용하여 검색하는 것이 가능해 세부적인 분석과 관계를 검색하고자 하는 업무에 최적화되어 있다.

각 Model의 분석 결과는 그래프 모델링에 익숙하지 않은 데이터 설계자에게 중요한 메시지를 준다. 첫째, 그래프 모델링을 세부화 할 경우 평균 쿼리 처리 성능은 느려지지만 다양한 정점과 간선에 다수의 매개변수로 쿼리가 가능하여 복잡한 연결 관계의 분석이 가능하다. 둘째, 그래프 모델링을 단순화 할 경우 평균 쿼리 처리 성능은 빨라지지만 사용 가능한 쿼리의 제한이 생기고 그에 따라 세밀한 분석이 어려워진다. 셋째, 관계형 데이터를 업무적 관점에서 그래프 모델로 재구성할 경우 평균 쿼리 처리 성능도 우수하며, 그래프 모델링에 익숙하지 않은 데이터 설계자도 빠른 학습 곡선으로 쿼리를 이용하는 것이 가능하다. 그래프 모델링은 실제 결과에 필요한 업무를 분석하고 해당 데이터를 그에 맞게 구조화하는 것이 바람직하다.

5. 결 론

그래프 데이터베이스의 본질은 개체 간의 관계를 의미론적으로 접근하여 그 관계 자체를 개체 데이터만큼의 비중을 갖게 함으로써 모델링의 자유도를 높이고 모델링 개발자들에게 좀 더 창의적인 시스템을 개발할 수 있는 바탕을 마련하는 것이다. 모델링을 할 때에 정점에 많은 정보를 담아 스키마를 단순화시키면 칼럼 형태의 모델에 가깝게 구현할 수 있다. 또한, 정점을 정규화하여 스키마의 규모가 커지면 RDF 형태의 모델이 된다. 이와같이 그래프 모델링은 다양한 구조를 자유롭게 제작할 수 있다.

그래프, 키-밸류, 칼럼, 다큐먼트 등의 noSQL은 관계형 데이터베이스의 확장성이나 복잡성의 한계 때문에 등장하였다. XML 기반의 데이터는 관계형 데이터베이스가 적합하지 않을 수 있다.[12] 그래프 데이터베이스는 자료구조의 그래프를 이용한 데이터베이스로 객체간의 관계인 고도로 연결된 데이터를 다루는데 가장 적합하다. 본 논문에서는 그래프 데이터 모델의 다양한 구조를 간선에 의미를 부여하여 레이블로 처리하였다. 더 나아가면 간선에 속성을 부여함으로써 객체간의 중요도나 관계 정보 등을 저장할 수 있다. 경로 사이에 관계 정보를 저장할 수 있다는 것은 중요한 역할을 하는데, 이를 타 noSQL처럼 속성에 저장하여 사용한다면 상대적으로 테이터의 활용도가 떨어질 것이다.

그래프 데이터베이스에서는 각 도메인에서 필요한 쿼리에 따라 모델별로 쿼리 성능이 다르다. 우리는 이에 그래프 모델링은 쿼리 응답 속도로 산출되는 숫자적인 결과물로서만이 아니라 특정한 업무 프로

세스에 대한 의도를 반영하여 기존과는 다른 방식으로 문제를 해결할 수 있다는 점을 역시 상기해야 한다.

모델 간의 수치적인 결과들은 4장의 Table 3과 같다. 성능만을 고려한다면 그래프 스키마를 최대한 단순화 시킨 Model A을 선택하는 것이 적합하겠지만, 다양한 데이터 구조를 표현하기 위해서는 Model C을 선택하는 것이 적합하다. 또한, 성능과 데이터 구조를 최적화 하면서 빠른 학습 곡선을 기대한다면 Model B를 선택하는 것이 적합하다. 섬세한 결과물을 도출해내야 하는 실사용에서는 성능 상의 한계도 감수해야 한다.

하지만, 그래프 모델을 가장 적합하게 사용하려면 수직적 깊이가 있는 데이터 모델링이 적합하다. 추후 데이터 확장이 용이하고 다양한 매개변수를 이용하여 쿼리가 가능한 반면, 쿼리 속도와는 교환이 생길 수 밖에 없다. 정점을 정규화하여 분리했을 때 단순히 수직적 깊이만 생기는 것이 아닌 온톨로지적 접근을 통해 추론이 가능하다. 온톨로지는 다른 데이터베이스에서 묘사할 수 없는 구조이다. 간선 자체에 의미를 부여함으로써 향후에 추론의 방법론을 도입할 수 있다는 것은, 성능상의 단점을 넘어서는 장점을 가져갈 수 있다.

중요한 점은 그래프 모델링은 시스템의 성능과 시스템의 개발 목적에 적확한 형태로 설계되어야 하며, 여기에 그래프 데이터베이스의 온톨로지적 접근이 보다 섬세하고 논리적인 결과물을 도출하는 데 주효하다.

REFERENCE

[1] I. Robinson, J. Webber, and E. Eifrem, *Graph Databases: New Opportunities for Connected Data*, O'Reilly Media, Sebastopol, CA, 2015.

[2] Database Normalization Basics Explained, https://support.microsoft.com/ko-kr/help/283878/description-of-the-database-normalization-basics (accessed August 1, 2019).

[3] M. Buerli, *The Current State of Graph Databases*, Department of Computer Science Calpoly San Luis Obispo, 2012.

[4] B.S. Min, "Visualization of BlockChain Data Structure based on Graph Database," *Proceedings of Symposium of the Korean Institute of communications and Information Sciences*, pp. 480-482, 2018.

[5] About Graph Database, https://aws. amazon.com/ko/nosql/graph (accessed August 1, 2019).

[6] Native Multi-model Can Compete with Pure Document and Graph Databases, https://www.arangodb.com/2015/06/multi-model-benchmark (accessed August 1, 2019).

[7] H. Huang and Z. Dong, *Research on Architecture and Query Performance based on Distributed Graph Database Neo4j*, IEEE, 10.1109/CECNet.2013.6703387, 2014

[8] Graph Modeling Guidelines, https://neo4j. com/developer/guide-data-modeling (accessed August 1, 2019).

[9] Import, Query, and Modify Graph Data, https://bitnine.net/tutorial/tutorial_kor.html (accessed August 1, 2019).

[10] The Contextual Graph Database of Forms, https://codeburst.io/contextual-database-of-forms-3349364b46d5 (accessed August 1, 2019).

[11] Northwind Traders Relational Data Sample, https://powerapps.microsoft.com/ko-kr/blog/northwind-traders-relational-data-sample (accessed August 1, 2019).

[12] M. Lee, H. Yong, and W. Lee, "A Technique of Converting CXQuery to XQuery for XML Databases" *Journal of Korea Multimedia Society*, Vol. 10, No. 3, pp. 289-302, 2007.

배 석 민

2016년～현재 단국대학교 일반대학원 ICT융합공학 박사수료
2017년～2019년 Graph Blockchain Limited 부사장
관심분야 : 그래프 이론, 프라이빗 블록체인, 머신러닝 등

김 진 형

2016년～현재 단국대학교 미래ICT융합학과 박사과정(수료)
2002년～2015년 한국 아이비엠 근무
2019년～현재 한국 레드햇 상무

관심분야 : 데이타 플랫폼, 블록체인, 공유경제

유 재 민

2017년 단국대학교 ICT융합학과 공학전공(박사과정수료)
2003년～2019년 분당서울대병원 의공팀장
2019년～현재 의료기기연구개발센타 파트장

관심분야 : 의료기기, 원격서비스, ICT융합, 블록체인, 의료정보

양 성 열

2016년～현재 단국대학교 미래ICT융합학과 박사과정
2006년～현재 (주)디엠씨시스 대표이사
2019년～현재 (주)유니트론텍 상무이사

관심분야 : 자율주행. IOT 보안, 블록체인

정 재 진

2005년 동신대학교 디지털콘텐츠학과 교수
2005년～2009년 동의대학교 디지털문화콘텐츠공학과 교수
2009년～현재 단국대학교 SW융합대학 응용컴퓨터공학과 교수
2015년～현재 단국대 일반대학원 미래ICT융합학과 교수
관심분야 : 블록체인 플랫폼, ICT융합서비스, 바이오융합