AI/ML

RDB, RAG는 정말 끝났나?

방동걸 (주)웨이스 대표이사
2026년 7월 25일

그래프 데이터베이스와 GraphRAG가 화제가 되면서 부쩍 자주 듣는 질문이 두 가지 있습니다. "이제 관계형 데이터베이스(RDB)는 끝난 거냐"와 "그래프 RAG라는 게 결국 SQL로도 다 되는 거 아니냐"입니다.
결론부터 말하면 둘 다 아닙니다. RDB는 끝나지 않았고, 그래프와 SQL은 같은 일을 하는 도구가 아닙니다. 셋은 서로를 밀어내는 게 아니라 각자 잘하는 자리를 나눠 갖는 방향으로 갑니다. 제조 AI 운영체제 VEXPLOR를 만드는 웨이스가 구축 현장 경험을 바탕으로 정리한 답입니다.

하나씩 짚어보겠습니다.



셋 중 무엇도 죽지 않았습니다 — 각자 잘하는 일이 다를 뿐입니다.

1. RDB — 안정적이지만, 관계를 따라가면 무너진다

관계형 데이터베이스는 데이터를 행과 열의 테이블로 저장하고, 테이블 사이의 관계를 정의해 관리합니다. 수십 년간 기업 시스템의 중심이었던 데는 이유가 있습니다. 트랜잭션 정합성(ACID), 검증된 안정성, 성숙한 도구 생태계 — 정확히 맞아야 하는 원장(≣ ≣)에선 여전히 RDB가 정답입니다.

한계는 "데이터를 저장할 때"가 아니라 "관계를 따라갈 때" 드러납니다.

간단한 추천 문제를 봅시다. 사용자 테이블, 상품 테이블, 그리고 둘의 다대다(N:M) 관계를 잇는 중간 테이블(구매 이력)이 있습니다. 홍길동(연필·볼펜·공책 구매)과 J(연필·볼펜·공책·샤프 구매)가 있을 때 "홍길동에게 무엇을 추천할까"를 풀려면, 논리적으로는 이렇습니다. 홍길동이 산 물건을 찾고 → 같은 물건을 산 다른 사용자를 찾고 → 그 사용자가 산 다른 물건을 찾고 → 홍길동이 이미 산 것을 뺀다.

SQL로 됩니다. 다만 이 "관계를 두세 단계 따라가는" 일이 RDB에서는 테이블을 반복해서 조인하고 서브쿼리를 겹치는 형태가 됩니다. 그리고 관계의 깊이가 깊어질수록 비용이 급격히 커집니다. 이유는 구조적입니다.

- **조인은 곱으로 커진다.** 한 단계 관계를 이을 때마다 중간 결과 집합이 곱해집니다. 두 단계, 세 단계로 들어가면 스캔해야 할 조합이 기하급수로 늘어납니다.
- **재귀에 약하다.** "친구의 친구의 친구" 같은 가변 깊이 탐색은 재귀 쿼리(CTE)로 흉내 낼 수 있지만, 깊이가 늘면 쿼리도 실행계획도 빠르게 무거워집니다.
- **쿼리가 의도를 숨긴다.** 정작 표현하고 싶은 건 "관계를 따라가라"인데, 결과물은 여러 겹의 조인과 서브쿼리라 사람이 읽고 고치기 어렵습니다.

즉 RDB의 문제는 능력이 아니라 **관계 탐색이라는 작업과 테이블·조인이라는 표현 방식이 서로 안 맞는다**는 점입니다.

---

2. 그래프 DB와 온톨로지 — 관계 자체가 데이터다

그래프 데이터베이스는 데이터를 점(노드, 엔티티)과 선(엣지, 관계)으로 저장합니다. 중간 테이블 같은 우회 없이 "사람" 노드와 "상품" 노드를 "구매했다"라는 선으로 직접 잇습니다.

여기에 얹히는 개념이 **온톨로지**입니다. 사물과 개념 사이의 관계를 컴퓨터가 사람처럼 이해하도록 명시적으로 그린 지식 지도입니다. 그래프 DB는 이 온톨로지를 담아내는 대표적인 방법입니다. "이건 설비, 저건 공정, 이 설비는 이 공정에 속하고, 이 불량은 이 공정에서 난다" 같은 의미 구조를 데이터 자체에 새겨 넣는 셈입니다.

장점은 두 가지로 요약됩니다.

- **관계가 한눈에 보인다.** 중간 테이블을 헤아리지 않아도 노드와 선만 보면 구조가 드러납니다.
- **탐색이 곧 답이다.** 추천이라면 홍길동 노드 → 구매한 상품 → 그 상품을 산 다른 사람 → 그 사람이 산 다른 상품으로 **선을 따라가기만** 하면 됩니다. 조인을 겹칠 필요가 없습니다.

이게 가능한 기술적 이유는 그래프 DB가 **인접 노드를 직접 참조(index-free adjacency)**하기 때문입니다. RDB가 관계를 이을 때마다 인덱스를 뒤져 매칭하는 것과 달리, 그래프 DB는 한 노드에서

연결된 노드로 바로 건너뜁니다. 그래서 탐색 비용이 전체 데이터 크기가 아니라 **실제로 지나가는 관계의 수**에 비례합니다. 관계가 깊고 복잡할수록 격차가 벌어집니다.

---

3. 벡터 DB — 의미로 찾지만, 관계와 근거는 약하다

한 축이 더 있습니다. 벡터 데이터베이스입니다. 문장·문서를 임베딩(숫자 벡터)으로 바꿔 저장하고, 의미가 비슷한 것을 유사도로 찾아옵니다. LLM에 관련 문맥을 넣어주는 RAG(검색 증강 생성)의 기본기가 여기서 나옵니다.

벡터 검색은 "비슷한 걸 찾는" 데 강하지만, "정확히 어떻게 연결됐는지"와 "왜 그렇게 답했는지(근거)"에는 약합니다. 관련은 있어 보이는데 사실관계가 틀린, 이른바 환각이 벡터 RAG에서 자주 나오는 것도 이 때문입니다.



미래는 배척이 아니라 보완입니다.

---

4. GraphRAG — 답을 지식 지도에 묶는다

그래서 나온 흐름이 GraphRAG입니다. 벡터로 "비슷한 걸" 데려오는 데 그치지 않고, **지식 그래프(온톨로지)의 관계 구조를 함께 근거로 넣어** LLM이 답하게 합니다.

효과는 분명합니다.

- **다단계 추론이 된다.** "A와 연결된 B, 그 B가 속한 C"처럼 관계를 따라가는 질문에, 그래프가 실제 경로를 짚어 문맥으로 제공합니다.
- **환각이 줄고 근거가 남는다.** 답을 흩어진 문서 조각이 아니라 명시적인 관계 그래프에 묶으면, "왜 이렇게 답했는지"의 경로가 남습니다. LLM이 지어내는 여지가 줄어듭니다.
- **전체 구조를 요약할 수 있다.** 그래프를 주제 단위(커뮤니티)로 묶어 요약해두면, 개별 문서 검색으로는 답하기 힘든 "이 데이터 전체에서 무슨 일이 일어나고 있나" 같은 넓은 질문에도 답할 수 있습니다.

정리하면 GraphRAG는 확률적으로 답을 만드는 LLM에게 **관계라는 뼈대**를 쥐여주는 방식입니다.

---

5. 그래서 미래는 — 배척이 아니라 보완(Hybrid)

여기서 처음의 질문으로 돌아갑니다. "그래프 RAG, SQL로도 되는 거 아니냐." 논리적으로는 상당 부분 됩니다. 하지만 관계 탐색을 조인으로 흉내 내는 순간 쿼리가 감당 못 할 만큼 복잡해진다는 게 앞서 본 그대로입니다. 반대로 "그럼 RDB 버리고 다 그래프로 가면 되냐"도 현실과 다릅니다.

이유는 간단합니다. 이미 수많은 기업이 RDB 위에 방대한 자산을 쌓고 안정적으로 돌리고 있습니다. 이걸 전부 걷어내고 벡터·그래프로 갈아엎는 일은 일어나지 않습니다. 그럴 이유도 없습니다.

현실적인 AX(AI 전환) 아키텍처는 이렇게 갑니다. **기존 RDB는 정합성이 생명인 핵심 원장으로 그대로 두고**, RDB만으로 버거웠던 관계 탐색과 AI 문맥 주입을 **그래프 DB·벡터 DB로 보완**합니다. 역할을 나누면 이렇습니다.

계층	잘하는 일	자리
RDB	트랜잭션·정합성·원장	사실의 원본(system of record)
벡터 DB	의미 유사 검색	비정형 문서·자연어 검색
그래프 DB	관계 탐색·다단계 추론	지식 구조·온톨로지
LLM	확률적 생성·요약	사람과의 접점

견고하고 확정적인 시스템(RDB) 위에, 확률적으로 작동하는 LLM을 얹을 때, 이 보완 구조가 유연하면서도 안정적인 서비스를 만듭니다. 어느 하나가 다른 하나를 대체하는 그림이 아닙니다.

6. 제조 현장에서는 왜 이 이야기가 특히 중요한가

이 구도가 가장 날카롭게 드러나는 곳이 제조입니다. 공장의 데이터는 곧 관계입니다. 어떤 설비가 어떤 공정에 속하고, 그 공정에서 어떤 불량이 나며, 그 불량의 원인이 어느 파라미터와 이어지는가 — 가치 있는 답은 대부분 "관계를 따라가는" 질문에서 나옵니다. 표에 흩어진 값만으로는 안 되고, 비슷한 문서를 데려오는 것만으로도 안 됩니다.

그래서 제조 AI에서 온톨로지와 그래프는 선택이 아니라 토대입니다. 답을 표준화된 로직과 온톨로지에 묶어야 환각 없이 "왜 그런지"까지 말하는 AI가 됩니다. 저희가 제조 특화 Graph RAG를 별도 기술로 확보한 것도 같은 이유입니다. 확률적으로 똑똑한 모델에 관계라는 뼈대를 쥐여주는 것 — 공장을 사람 없이 운영하려면 그 뼈대가 먼저 있어야 합니다.

한 줄 요약. RDB의 SQL로도 관계는 추론할 수 있지만 연산이 감당 못 하게 복잡해집니다. 그래서 온톨로지를 담은 그래프 DB가 나왔고, 벡터 DB는 의미 검색을 더합니다. 이들은 안정적인 RDB를 대체하는 게 아니라, RDB의 복잡도를 낮추고 LLM의 신뢰도를 높이는 방향으로 **서로 보완하며(Hybrid)** 발전합니다. RDB도 RAG도 끝나지 않았습니다. 자리를 나눠 갖는 중입니다.

WACE(웨이스)는 공장이 스스로 판단하고 멈추지 않게 하는 제조 AI 운영체제 VEXPLOR를 만드는 회사다.

원문 blog.vexplor.com/post/rdb-rag-graphrag-hybrid
발행 2026년 7월 25일 · 방동걸 (주)웨이스 대표이사

회사 주식회사 웨이스 (WACE Inc.) · 대표이사 방동걸 · 사업자등록번호 439-87-02852
주소 (18462) 경기 화성시 동탄구 동탄영천로 150, 에이동(A동) 1702-1708호 (영천동, 실리콘앨리동탄)
연락처 전화 070-4730-0629 · contact@wace.me
홈페이지 wace.me · linkedin.com/company/wace-inc

원문 홈페이지 링크드인