데이터 일관성 수준별 트랜잭션·합의 알고리즘 추천

데이터 일관성 수준에 따른 트랜잭션 및 합의 알고리즘 추천은 다음과 같이 구분할 수 있습니다.

데이터 일관성 수준	트랜잭션 형식 및 알고리즘	특징 및 추천 상황
강한 일관성 (Strong Consistency)	2PC (2-Phase Commit), 분산 트랜잭션	모든 참여 노드가 같은 값을 보고, 완전한 원자성을 보장. 은행, 금융 거래 등 정확한 데이터가 필수인 시스템에 적합. 하지만 상대적으로 성능 저하와 지연 있음. [1] [2]
내결함성 있는 합의 (Consensus)	Paxos, Raft, Viewstamped Replication	분산 시스템에서 노드 장애가 있어도 정상적으로 합의를 이루어 데이터 동기화 보장. 노드 간 상태 일관성 유지에 강점. 복잡한 분산 서비스나 마이크로서비스 환경에 적합. [1] [3]
최종 일관성 (Eventual Consistency)	CRDT, Vector Clocks, Gossip Protocol	데이터 복제 노드가 최종적으로 일관된 상태에 도달함을 보장. 실시간 일관성이 필요 없고, 가용성과 성능을 우선시하는 시스템에 적합. CDN, 소셜 미디어 등에 활용됨. [4] [5]
인과 일관성 (Causal Consistency)	Lamport Timestamps, Hybrid Logical Clocks	단말기간 인과관계가 유지되는 일관성. 사용자 경험 중시 시스템에서 활용, 분산 로깅, 협업 도구에 적합. [5]

추가 상세 설명

- 2PC는 분산 트랜잭션에서 원자성을 유지하지만, 중간 코디네이터 고장 시 블로킹 가능성 있음. 실시간 성능보다는 안정성이 중요할 때 선택.[1]
- Paxos·Raft는 다수결 기반의 합의 프로토콜로 장애 허용성과 데이터 일관성을 균형있게 유지, Microservices와 분산 데이터 저장소 등에서 광범위 활용.[3] [1]
- CRDT, Vector Clock 같은 기법은 네트워크 분할 시에도 데이터 충돌 해결이 가능하며 네트워크 회복 시 자연스럽게 데이터 일관성을 회복해주는 특성 있음.[5] [4]
- 인과적 일관성은 전체 순서 보장보다는 관련 이벤트 간 인과 관계를 보장하는 수준으로 클라우드 기반 협업툴이나 분산 로그에 적합함.[5]

이처럼 애플리케이션 요구사항에 따라 데이터 정확성과 성능, 내결함성을 고려해 적합한 트랜잭션 및 합의 알고리즘을 선택하는 것이 중요합니다.

⁂

1. https://loosie.tistory.com/890
2. https://f-lab.kr/insight/transaction-and-data-integrity-20250114
3. https://catsbi.oopy.io/79c4ab28-2130-4132-90b2-10076372a573
4. https://www.geeksforgeeks.org/system-design/eventual-consistency-in-distributive-systems-learn-system-design/

5. https://pompitzz.github.io/blog/ComputerScience/DesigningDataIntensiveApplication/ch9.html
6. https://twentytwentyone.tistory.com/475
7. https://contents.premium.naver.com/yonseident/ysdent/contents/231231101444697cp
8. https://notavoid.tistory.com/68
9. https://velog.io/@kimtjsdlf/분산-트랜잭션과-합의-내결함성
10. https://www.borntodare.me/94dff6e7-5d2c-4f9c-92a7-7f98418dd1e7
11. https://blockchain.kisa.or.kr/file/2024_블록체인_솔루션_편람.pdf