3. 로그우선기록(WAL, write ahead logging)

// 로그우선기록(WAL, write ahead logging)

- DBMS는 데이터베이스 **갱신 전**에 로그(log)를 **디스크**의 **로그파일**에 **먼저 기록**한다.
  → 시스템 장애가 발생했을 때 극복하기 위해(트랜잭션 원자성 보장)
- 트랜잭션은 **〈Ti, commit〉** 로그레코드를 안정 저장장치에 출력시켜야만 완료 상태로 들어갈 수 있다.
- **〈Ti, commit〉**를 출력하려며, 먼저 **Ti**와 관련된 모든 로그레코드부터 안정 저장장치에 출력해야 한다.
- 로그가 임시 저장된 주기억장치 블록을 로그버퍼라 한다.
- 로그버퍼와 로그파일에 기록된 로그레코드들의 순서는 같아야 한다.
- 로그버퍼는 주기억장치에서 운영되므로 시스템이 붕괴되면 버퍼 내용을 잃을 수도 있다.

// Steal / No-Steal 정책

Steal	• 트랜잭션이 **완료되기 전**에 갱신된 페이지를 언제든지 디스크에 **기록**할 수 있는 정책 • DBMS의 버퍼관리자가 다른 트랜잭션을 위한 버퍼가 필요로 할 때 적용한다. • 미완료된 트랜잭션이 갱신한 페이지를 훔치는 것처럼 디스크에 기록한다. • 장점: 로그버퍼 공간을 줄일 수 있다.
No-Steal	• 트랜잭션이 완료되기 전까지는 갱신된 페이지를 **버퍼에 유지**하는 정책 • 트랜잭션이 완료되기 전까지는 갱신된 페이지를 디스크에 기록하지 않는다.

- **No-Steal** 정책은 **undo**가 불필요한 매력을 가지고 있다.(디스크에 기록한 것이 없으므로)
- 하지만, **No-Steal** 정책은 매우 큰 메모리 버퍼가 필요하다는 문제점을 가지고 있다.
- **Steal** 정책은 필연적으로 **undo** 복구를 수반한다.
- 이유는, **Steal** 정책은 수정된 페이지가 언제든지 디스크에 기록될 수 있으므로
- 대부분의 DBMS 채택하는 버퍼 관리 정책은 **Steal**이다.

// Force / No-Force 정책

Force	• 트랜잭션이 갱신한 모든 페이지를 트랜잭션 **완료 시점에 즉시 디스크에 반영**하는 정책
No-Force	• 트랜잭션이 **갱신**한 페이지를 트랜잭션 **완료** 시점에 **즉시 디스크에 반영하지 않는** 정책 • 트랜잭션이 **완료**되어 당연히 기록할 것 같지만 DBMS는 **적절한 시점**에 **기록**한다. • 그리고, 주의할 것은 즉시 반영하지 않는다는 것이지 어떤 기록도 없다는 것은 아니다. • 특정 페이지가 여러 트랜잭션에 의해 자주 갱신되는 경우에 적용한다. • 장점 : 디스크 입출력 비용을 줄인다.**(성능 효율 향상)**

- **Force** 정책은 **redo**가 불필요(수정된 페이지가 이미 디스크의 데이터베이스에 반영되어서)
- 하지만, **Force** 정책을 적용해도 데이터베이스 백업 복구는 **redo**가 요구된다.
- **No-Force** 정책은 **redo**가 필요(수정된 페이지가 데이터베이스에 반영되지 않을 수 있기에)
- 대부분의 DBMS 채택하는 버퍼 관리 정책은 **No-Force**이다.

2 http://cafe.daum.net/pass365(홍재연)

// 핵심 정리

Steal	· 트랜잭션이 **완료되기 전**에 갱신된 페이지를 언제든지 디스크에 **기록**할 수 있는 정책 · DBMS의 버퍼관리자가 다른 트랜잭션을 위한 버퍼가 필요로 할 때 적용한다. · 미완료된 트랜잭션이 갱신한 페이지를 훔치는 것처럼 디스크에 기록한다. · 장점: 로그버퍼 공간을 줄일 수 있다.
No-Force	· 트랜잭션이 **갱신**한 페이지를 트랜잭션 **완료** 시점에 **즉시 디스크에 반영하지 않는** 정책 · 트랜잭션이 **완료**되어 당연히 기록할 것 같지만 DBMS는 **적절한 시점**에 **기록**한다. · 그리고, 주의할 것은 **즉시 반영하지 않는다는 것이지 어떤 기록도 없다는 것**은 아니다. · 특정 페이지가 여러 트랜잭션에 의해 자주 갱신되는 경우에 적용한다. · 장점 : 디스크 입출력 비용을 줄인다.**(성능 효율 향상)**

정리	· 대부분의 DBMS가 채택하는 버퍼 관리 정책은 **Steal**과 **No-Force**이다. · 해서, DBMS는 **redo**와 **undo** 복구가 모두 필요하게 된다.

• 위에서 강조한 내용을 모르면
• 즉시갱신은 정상적으로 완료된 트랜잭션의 수행 결과가 이미 디스크에 반영되었는데
• 왜? 또 다시 redo를 수행하는지? 이해할 수 없다.
• DBMS가 No-Force 정책을 사용하므로

기출문제 분석

1. 로그버퍼에 대한 설명으로 옳지 않은 것은? [2015년 국가 7급]

① 로그파일은 안정 저장장치(stable storage)에서 운영되며 로그버퍼는 주기억장치에서 운영된다. 따라서 시스템 고장 발생 시 로그버퍼의 내용을 잃을 수 있다.
② 로그레코드 〈Ti, commit〉가 로그파일에 기록되기 전에 로그버퍼내의 Ti와 관련된 모든 로그레코드들은 로그파일에 기록되어야 한다.
③ 데이터베이스 버퍼에 있는 블록을 데이터베이스 파일에 기록하는 것과 로그버퍼에 있는 블록을 로그파일에 기록하는 것은 순서적으로 독립적이다.
④ 로그버퍼에 기록된 로그레코드들의 순서와 로그파일의 이들의 순서는 동일하여야 한다.

♤ **로그우선기록 (WAL)**

• DBMS는 로그를 디스크상의 로그파일에 먼저 기록한다.

정답 : ③

2. 데이터베이스 관리 시스템의 캐시 관리 방식에 대한 설명으로 옳지 않은 것은? [2017년 국가 7급]

① no-steal 방식에서는 회복과정 중 UNDO와 REDO 연산의 수행이 모두 필요하다.
② 갱신된 페이지의 수가 많고 크기가 큰 경우 no-steal 방식에 비해 steal 방식이 필요한 캐시 버퍼 크기를 줄일 수 있다.
③ 완료된(committed) 트랜잭션에서 갱신된 페이지가 다른 트랜잭션들에서도 빈번히 갱신되는 경우, force 방식에 비해 no-force 방식이 디스크로부터 캐시로 그 페이지를 다시 읽는 횟수를 줄일 수 있다.
④ force 방식은 트랜잭션이 완료되기 전에 그 트랜잭션이 갱신한 모든 페이지들을 디스크에 저장하게 한다.

♤ **데이터베이스 관리 시스템의 캐시 관리 방식**

• no-steal 방식에서는 회복과정 중 UNDO와 REDO 연산의 수행이 모두 필요하다.**(×)**
→ no-steal 방식은 자료가 갱신되어도 트랜잭션 종료 전에는 디스크에 기록하지 않는다.
→ 해서, UNDO 과정은 수행할 필요가 없고 REDO 과정만 수행한다.(지연갱신 기법)

정답 : ①

3. 데이터베이스 시스템의 회복에 대한 설명으로 가장 옳지 않은 것은? [2021년 서울 7급]

① 장애로 인해 손상된 데이터베이스를 손상되기 이전의 정상적 상태로 복구하는 것이다.
② SW 오류는 물론 HW 오류까지 대비한다.
③ 검사점(checkpoint) 기법의 목적은 복구 작업에 소요되는 시간을 줄이기 위한 것이다.
④ WAL(Write-Ahead Logging) 기법은 데이터를 갱신한 후에, 로그에 기록을 남기는 방법이다.

♤ **데이터베이스 시스템의 회복 – 로그우선기록(WAL, write ahead logging)**

- DBMS는 데이터베이스 **갱신 전**에 로그(log)를 **디스크**의 **로그파일**에 **먼저 기록**한다.
- 시스템 장애가 발생했을 때 회복하기 위한 것이다.(트랜잭션 원자성 보장)

정답 : ④

4. DBMS에서 트랜잭션의 동시성제어(concurrency control)와 회복(recovery) 기술에 대한 설명으로 옳지 않은 것은? [2011년 국가 7급]

① 데이터베이스에 대한 즉시갱신은 데이터베이스에 대한 갱신 로그를 저장함으로서 회복에 대비한다. 일반적으로 데이터베이스와 로그의 동시 손상을 피하기 위해 별도의 전용 디스크에 로그를 저장할 수 있다.
② 데이터베이스 갱신에 대한 로그 기록을 유지하는 것은 일반적으로 성능 저하를 가져오므로, 성능을 유지하기 위해 로그보다 데이터베이스 갱신을 먼저 쓰는 것을 WAL(write-ahead logging)이라고 한다.
③ 검사점(checkpoint) 회복 기법은 트랜잭션 수행 중 주기억장치의 로그버퍼와 데이터베이스 버퍼를 디스크에 저장한다.
④ 2-단계 로킹(2-phase locking)은 확장단계(growing phase)와 수축단계(shrinking phase)로 구성된다. 확장단계에서는 트랜잭션이 데이터 항목에 대하여 새로운 로크를 요청할 수 있지만 해제할 수는 없으며, 수축단계에서는 반대로 로크를 해제할 수는 있지만 새로 요청할 수 없다.

♤ **동시성제어와 회복**

- 데이터베이스 갱신에 대한 로그 기록을 유지하는 것은 일반적으로 성능 저하를 가져오므로, 성능을 유지하기 위해 로그보다 데이터베이스 갱신을 먼저 쓰는 것을 WAL이라고 한다.**(×)**
  → 데이터베이스보다 로그를 먼저 쓰는 것을 WAL(write-ahead logging)이라 한다.

정답 : ②

5. 다음 중 데이터베이스 관리 시스템의 트랜잭션 및 버퍼 관리 전략의 하나로써 어느 한 트랜잭션이 갱신된 데이터 블록(block)을 디스크(disk storage)에 반영하는 저장 기능을 완료해야 하는데, 해당 트랜잭션의 커밋(commit) 시점에 디스크에 반영하지 않는 것을 허용하는 정책의 명칭으로 가장 적절한 것은? [2023년 군무 7급]

① 강제 정책(force policy)　② 비강제 정책(no-force policy)
③ 스틸 정책(steal policy)　④ 비스틸 정책(no-steal policy)

♤ **버퍼 관리 전략**

// Steal, No-Steal 정책

Steal	· 트랜잭션이 **완료되기 전**에 갱신된 페이지를 언제든지 디스크에 **기록**할 수 있는 정책 · DBMS의 버퍼관리자가 다른 트랜잭션을 위한 버퍼가 필요로 할 때 적용한다. · 미완료된 트랜잭션이 갱신한 페이지를 훔치는 것처럼 디스크에 기록한다. · 장점: 로그버퍼 공간을 줄일 수 있다.
No-Steal	· 트랜잭션이 완료되기 전까지는 갱신된 페이지를 **버퍼에 유지**하는 정책 · 트랜잭션이 완료되기 전까지는 갱신된 페이지를 디스크에 기록하지 않는다.

· No-Steal 정책은 undo가 불필요한 매력을 가지고 있다.(디스크에 기록한 것이 없으므로)
· 하지만, No-Steal 정책은 매우 큰 메모리 버퍼가 필요하다는 문제점을 가지고 있다.
· Steal 정책은 필연적으로 undo 복구를 수반한다.
· 이유는, Steal 정책은 수정된 페이지가 언제든지 디스크에 기록될 수 있으므로
· 대부분의 DBMS 채택하는 버퍼 관리 정책은 Steal이다.

// Force, No-Force 정책

Force	· 트랜잭션이 갱신한 모든 페이지를 트랜잭션 **완료 시점에 즉시 디스크에 반영**하는 정책
No-Force	· 트랜잭션이 **갱신**한 페이지를 트랜잭션 **완료** 시점에 **즉시 디스크에 반영하지 않는** 정책 · 트랜잭션이 **완료**되어 당연히 기록할 것 같지만 DBMS는 **적절한 시점**에 **기록**한다. · 그리고, 주의할 것은 즉시 반영하지 않는다는 것이지 어떤 기록도 없다는 것은 아니다. · 특정 페이지가 여러 트랜잭션에 의해 자주 갱신되는 경우에 적용한다. · 장점 : 디스크 입출력 비용을 줄인다.**(성능 효율 향상)**

· Force 정책은 redo가 불필요(수정된 페이지가 이미 디스크의 데이터베이스에 반영되어서)
· 하지만, Force 정책을 적용해도 데이터베이스 백업 복구는 redo가 요구된다.
· No-Force 정책은 redo가 필요(수정된 페이지가 데이터베이스에 반영되지 않을 수 있기에)
· 대부분의 DBMS 채택하는 버퍼 관리 정책은 No-Force이다.

정리	· 대부분의 DBMS가 채택하는 버퍼 관리 정책은 Steal과 No-Force이다. · 해서, DBMS는 redo와 undo 복구가 모두 필요하게 된다.

정답 : ②