



[AI초급] SQLD 자격증 코스 - 챕터 4



매 주차 강의자료 시작에 PDF파일을 올려두었어요!

▼ PDF 파일

[수업 목표]

- 데이터 모델과 성능, 데이터베이스에 대한 전반적인 내용을 학습합니다

[목차]

01. 성능 데이터 모델링과 정규화

02. 정규화 이론

03. 반정규화 개념과 설명



모든 토글을 열고 닫는 단축키

Windows : **Ctrl** + **alt** + **t**

Mac : **⌘** + **~** + **t**

01. 성능 데이터 모델링과 정규화



데이터베이스 성능 향상을 목적으로 하는 작업에 대해 학습합니다

▼ 1) 데이터 모델 성능



성능 데이터 모델링의 정의

'성능 데이터 모델링'이란 데이터베이스 성능 향상을 목적으로 하는 작업을 의미합니다. 학습 단계에서는 데이터베이스의 성능이 미치는 영향력에 대해 생각하기란 쉽지 않은 일입니다. 다루는 데이터의 양도 많지 않고 성능 최적화에 대한 다양한 기법이 온전하게 이해되지 않은 상태이기 때문에 오히려 이렇게 복잡한 작업을 왜 하는지에 대해서 의문을 품는 것이 당연할 수 있습니다. 그렇다면 왜 우리는 데이터베이스를 다룰 때 성능에 신경 써야 하는 걸까요?

사실 데이터베이스는 우리가 생각하는 것보다 훨씬 더 많은 요청을 처리합니다. 아무렇지 않게 날리는 쿼리문 하나가 내부적으로는 수많은 과정을 거쳐야 하는 경우도 있습니다. 어쩌다가 딱 한 번 수행되는 구문이라면 그 순간에만 조금 비효율적인 결과에 대한 비용을 지불하면 되지만 매우 빠른 템포로 자주 발생하는 쿼리문이 비효율적이면 DB 전체의 성능에 영향을 줄 수 있습니다.

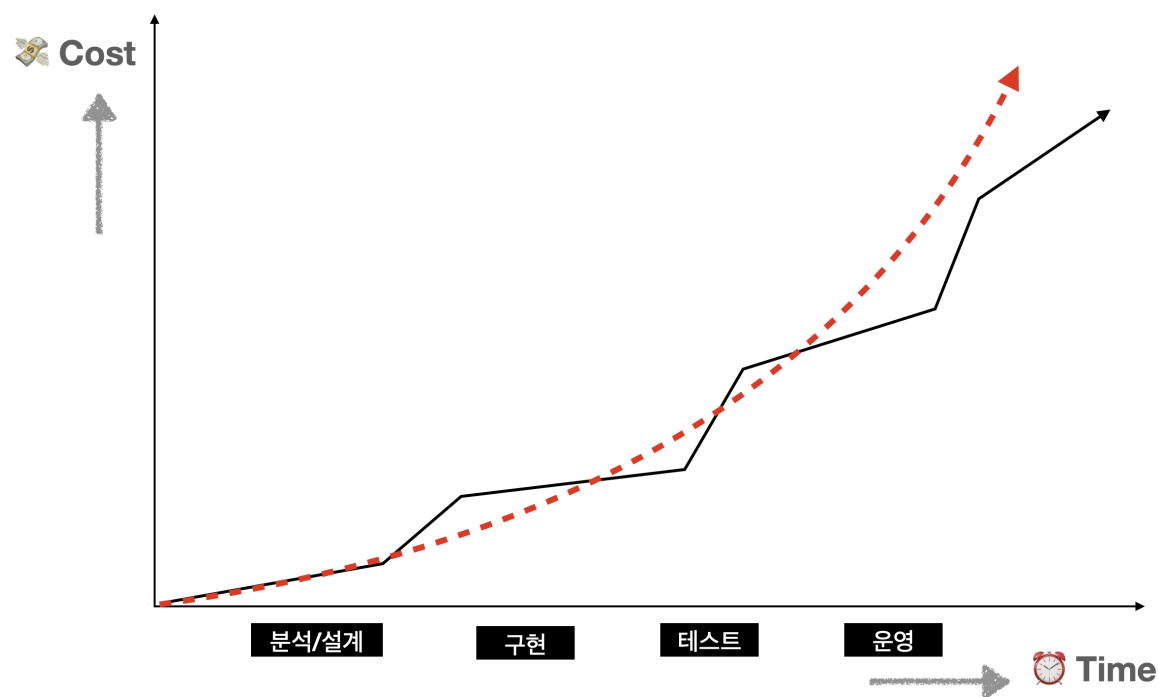
하지만 이것만으로는 해결되지 않는 문제들이 있습니다. 테이블 설계 자체가 잘못되어 쿼리문을 원하는 방식으로 날리지 못할 수도 있고 테이블 내부에 원하는 정보가 없어 불가피하게 다른 테이블에서 정보를 가져와야 하는 경우도 존재합니다. 이처럼 성능 향상을 저해하는 요소는 매우 다양하며 처음부터 성능 향상을 목적으로 진행하는 것이 성능 데이터 모델링입니다. 성능 향상이 목적이기 때문에 데이터 모델링 때부터 정규화, 반정규화, 테이블 통합, 데이터 분할, 조인구조, PK, FK 등 여러 가지 성능과 관련된 사항이 데이터 모델링에 반영될 수 있도록 하는 것으로 정의할 수 있습니다.



성능 데이터 모델링 수행시점

성능 향상을 위한 비용을 프로젝트 수행 중보단 사전에 철저하게 설계된 상태로 도입할 수록 비용이 적게 듭니다. 특히, 분석 및 설계 단계에서 데이터 모델에 성능을 고려한 데이터 모델링을 수행할 경우 성능 저하로 인해 발생하는 재업무(Rework) 비용을 최소화할 수 있습니다.

분석/설계 단계에서 데이터 모델은 대충하고 성능이 저하되는 SQL 문장을 고치고 부족한 하드웨어 용량을 늘리는 등의 작업은 추가적인 비용을 들여 성능 개선을 위한 작업을 하는 것입니다. 이러한 댄질 식의 성능 이슈 대응은 데이터의 증가 속도가 빠를 수록 기하급수적으로 비용이 증가하는 문제점이 있습니다.



따라서 데이터베이스 분석 및 설계 단계에서 처리 성능을 향상 시키기 위한 준비를 많이 해야 합니다. 어떠한 트랜잭션이 비즈니스 로직의 핵심인지 파악하고 사용자 업무 처리에 중요성을 보이는지 분석 해야합니다. 취약점이라고 생각되는 부분에 트랜잭션을 발생시켜 실제 성능을 테스트 해보는 것도 필요한 작업입니다.

* 트랜잭션이란? 데이터베이스의 상태를 변화시키기 위해 수행하는 작업 단위

☑ 성능 데이터 모델링의 고려사항

(1) 정규화를 정확하게 수행

- 데이터를 주요 관심사별로 분산 시킬 수 있기 때문에 이 자체로 성능 향상의 효과가 있습니다.
- 정규화를 통해 중복된 데이터가 쌓이는 것을 막을 수 있습니다.

(2) 데이터베이스 용량 산정 수행

- 어떤 테이블(엔티티)에 데이터가 집중되는지 파악 가능합니다.
- 필요한 경우 테이블 분리와 조인을 통한 데이터 수집이 필요합니다.

(3) 데이터베이스에서 발생하는 트랜잭션의 유형을 파악

- CRUD 매트릭스 혹은 시퀀스 다이어그램을 보면 파악하기에 용이합니다.
- 데이터 조회에 필요한 조인 관계 등을 파악할 수 있게 됩니다.

(4) 데이터베이스의 용량과 트랜잭션 유형에 따라 반정규화 수행

- 테이블, 속성, 관계 등에 대해서 포괄적인 반정규화를 통해 성능을 조정해야 합니다.

(5) 이력 모델, PK / FK, 슈퍼 타입 / 서브 타입의 조정

- 성능이 우수한 순서대로 칼럼의 순서를 조정해야 합니다.

(6) 성능 관점에서 데이터 모델 검증

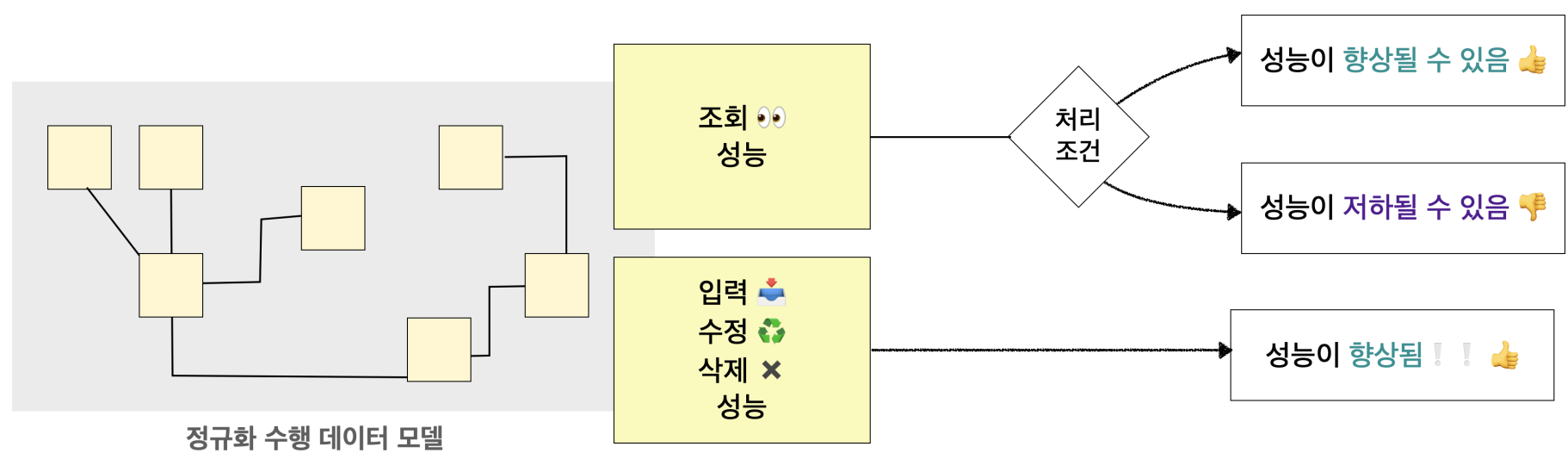
- 항상 성능 최적화를 위해서 데이터 모델을 검증합니다.
- 데이터 모델이 괜찮은 형태로 구조화 되어있다고 하더라도 성능 최적화를 위한 선택을 위해 끊임없이 고민합니다.

▼ 2) 정규화를 통한 성능 향상 전략

정규화를 수행한다는 것은 데이터에 대한 중복성을 제거하여 성능을 향상시키는 것을 의미합니다. 이러한 작업을 통해 데이터의 중복된 속성을 제거할 수 있으며 특정한 칼럼으로 분산되어 있는 데이터의 의미를 하나로 집약 시킬 수 있기 때문에 테이블의 칼럼 수가 줄어들어 데이터 용량을 줄일 수 있습니다.

예를 들면, 학생 테이블에서 학번이 '123456'인 학생의 이름이 '김민재'라는 사람을 조회한다고 가정 해봅시다. 이때 학번을 통해 검색하거나 학생의 이름을 통해 검색을 할 텐데, 해당 정보가 여러 테이블의 칼럼에 속해있다면 어떨까요? 원하는 데이터를 찾기 위해 불필요하게 여러 번 조회해야 합니다. 학생 이름 칼럼 및 학번 칼럼은 학생 테이블에만 존재할 때 가장 효과적입니다. 이처럼 조회하고자 하는 데이터가 여러 테이블에 분산되어 있지 않으면 조회 과정에서의 불필요한 연산을 줄일 수 있기 때문에 성능 향상에 기여할 수 있습니다.

정규화된 테이블에서 데이터를 조회할 때 성능이 향상될 수도 혹은 아닐 수도 있습니다. 어떤 방식으로 처리되느냐에 따라서 성능이 달라지게 됩니다. 불필요한 조회 조건 등이 많아지게 되어 테이블 간 조인 작업 등이 빈번하게 나타나는 경우는 정규화 이전의 구조에서 조회 성능이 좋아질 수도 있습니다. 반면에 데이터를 삽입하거나 수정, 삭제할 때는 중복된 요소를 줄였기 때문에 성능이 향상됩니다.



• 정규화 한 줄 요약

데이터의 일관성을 유지하고 데이터의 중복을 방지하며 데이터의 유연성을 유지하기 위해 데이터를 분해하는 과정

▼ 3) 정규화 용어 및 이점

☑ 정규화 용어

용어	설명
정규화 (Normalization)	DBMS 테이블의 삽입, 삭제, 수정 과정에서의 이상(Anomaly) 현상의 발생을 최소화하기 위해 작은 단위의 테이블로 나뉘는 과정입니다.
정규형 (NF: Normal Form)	정규화된 결과물에 의해 도출된 데이터 모델이 갖춰야 할 특성을 만족하는 '정규화된 결과물'을 의미합니다.
함수적 종속성 (FD: Functional Dependency)	- 테이블의 특정한 칼럼 값(A)을 알고 있으면 다른 칼럼 값(B)을 알 수 있다고 가정할 때, 칼럼 B는 칼럼 A에 함수 종속성을 갖는다고 표현합니다. - 예를 들어, 학번을 통해 학생 이름을 알 수 있다고 하면 '학생 이름은 학번에 함수적 종속성을 갖는다'고 표현할 수 있습니다.
결정자 (Determinant)	- 함수적 종속성에서 학번은 학생 이름을 결정짓는 요소기 때문에 '결정자'라고 표현합니다.
다치 종속 (MVD: MultiValued Dependency)	- 결정자 칼럼 A에 의해 칼럼 B의 값을 다수 알 수 있을 때, 칼럼 B는 칼럼 A에 다치종속 되었다고 표현합니다.

☑ 정규화의 이점

• 데이터의 유연성

- 종속성이 강한 데이터를 분리하여 독립된 개념으로 정의하기 때문에 높은 응집도와 낮은 결합도 원칙에 충실해집니다.

* 응집도 : 요소들이 서로 관련되어 있는 정도 (높을수록 품질이 좋다)

* 결합도 : 요소들 간의 상호 의존하는 정도 (결합도가 높으면 시스템 구현 및 유지보수가 어렵다)

- 데이터의 재활용성

- 정규화를 통해 데이터의 개념이 조금 더 세분화될 수 있고 그 결과로 개념에 대한 재활용 가능성이 증가합니다.

- 데이터의 중복 최소화

- 정규화는 식별자가 아닌 속성이 한 번만 포함되기 때문에 데이터의 중복이 최소화됩니다.

▼ 연습문제

✓ 문제 1

Q. 문제	성능 데이터 모델링에 대한 설명으로 옳지 않은 것은?
A. (1)	항상 성능 최적화를 위해서 데이터 모델을 검증해야 한다
A. (2)	성능 향상을 저해하는 요소는 매우 다양하며 처음부터 성능 향상을 목적으로 진행하는 것이 성능 데이터 모델링이다
A. (3)	성능 향상을 위한 비용을 프로젝트 수행에 있어 사전에 도입할 수록 비용이 들지 않는다
A. (4)	구현/테스트 단계에서 데이터 모델에 성능을 고려한 데이터 모델링을 수행할 경우 성능 저하로 인해 발생하는 재업무(Rework) 비용을 최소화할 수 있다

▼ 정답

(4) 분석/설계 단계에서 데이터 모델에 성능을 고려한 데이터 모델링을 수행할 경우 성능 저하로 인해 발생하는 재업무(Rework) 비용을 최소화할 수 있다

✓ 문제 2

Q. 문제	정규화를 통한 성능 향상 전략으로 옳은 것은?
A. (1)	정규화를 수행하면 조회 성능은 저하되지 않는다
A. (2)	정규화를 수행하면 입력 성능은 좋아지지 않는다
A. (3)	정규화를 수행하면 조회 성능은 향상될 수도 있다
A. (4)	정규화를 수행하면 삭제 성능은 저하될 수 있다

▼ 정답

(3)

✓ 문제 3

Q. 문제	정규화의 이점으로 옳지 않은 것은?
A. (1)	종속성이 강한 데이터를 분리하여 독립된 개념으로 정의하기 때문에 높은 응집도와 낮은 결합도 원칙에 충실해진다
A. (2)	정규화를 통해 데이터의 개념이 조금 더 세분화될 수 있다
A. (3)	정규화는 식별자가 아닌 속성이 다수 포함되기 때문에 데이터의 중복이 최소화된다
A. (4)	데이터의 재활용 가능성이 증가한다

▼ 정답

(3) 정규화는 식별자가 아닌 속성이 한 번만 포함되기 때문에 데이터의 중복이 최소화된다

02. 정규화 이론

✓ 정규화 유형과 함수적 종속성에 대해 학습합니다

▼ 1) 제1정규화



한 속성에 여러 개의 속성이 포함되어 있거나 같은 유형의 속성이 여러 개로 나뉘져있는 경우 해당 속성을 분리

정규화는 함수적 종속성을 근거로 합니다. 함수적 종속성(Functional Dependency)이란 데이터들이 어떤 기준값에 의해 종속되는 현상을 의미합니다. 이때 기준값을 '결정자'라고 하고 종속값을 '종속자'라고 합니다.



종속자는 함수 종속성을 갖게 되며 결정자에 의해 종속자가 결정되는 구조입니다.



이름, 출생지, 주소는 주민등록번호에 대해 함수 종속성을 갖습니다. 여기서 이름, 출생지, 주소는 주민등록번호가 결정하기 때문에 '종속자'라고 볼 수 있고 주민등록번호는 이들을 결정하기 때문에 '결정자'라고 볼 수 있습니다.

그렇다면 아래의 테이블을 통해 제1정규형을 위반한 모습을 살펴볼까요? 연락처 칼럼에 핸드폰 번호와 이메일 주소 두 개의 속성이 포함되어 있습니다.

• 회원정보

아이디	나이	성별	회원구분	연락처
justin	33	남	일반	010-1234-1234, justin@email.kr
ed	45	남	일반	010-2456-1234, ed@email.kr
chelsea	27	여	프리미엄	010-1256-7895, chelsea@email.kr

위에서 발생한 문제를 해결하기 위해서는 아래와 같은 작업이 필요합니다.

• 회원 정보


아이디	나이	성별	회원구분
justin	33	남	일반
ed	45	남	일반
chelsea	27	여	프리미엄

• 회원 연락처

아이디	연락처 구분	연락처
justin	핸드폰	010-1234-1234
justin	이메일	justin@email.kr
ed	핸드폰	010-2456-1234
ed	이메일	ed@email.kr
chelsea	핸드폰	010-1256-7895
chelsea	이메일	chelsea@email.kr

우선 회원 정보 테이블에서 연락처 속성 삭제합니다. 회원 연락처 테이블에서 속성의 경우 연락처를 구분할 수 있는 칼럼과 그에 해당하는 연락처 칼럼을 따로 두어 구분하였습니다. 하나의 칼럼이 여러 개의 속성값을 갖기 때문에 이를 분리하였고 결과적으로 제1정규형을 만족한다고 할 수 있습니다.


▼ 2) 제2정규화

 제 1정규화를 만족시키고 PK가 아닌 모든 칼럼은 PK 전체에 종속

만약 PK에 종속되지 않거나 PK 중에서 일부의 칼럼에만 종속되는 칼럼이 있다면 이는 분리시켜야 합니다.

PK 값은 해당 테이블의 어떤 속성과도 종속관계를 갖는다는 사실을 알고 있습니다. 아래 고객 주문 테이블에서 PK는 고객 아이디와 주문 순번의 복합 속성으로 구성되어 있습니다. 복합 식별자(PK)를 기준으로 놓고 볼 때 '고객 이름'과 '고객 등급' 속성은 '고객 아이디' 속성에는 종속되지만 '주문 아이템'이라는 속성에는 종속되지 않습니다. 고객 이름과 고객 등급의 값을 결정하는 기본키를 구성하는 칼럼은 고객 아이디 하나이고 '주문 순번'은 이에 영향을 주지 않습니다.

이러한 경우의 문제점은 특정한 데이터를 갱신하려고 할 때 갱신 이상(Modification Anomaly) 현상이 발생할 가능성이 큼니다. 또한 고객 정보를 모르면 주문 할 수 없는 경우가 생길 수 있습니다.

 **참고 - 갱신 이상(Modification Anomaly)**

반복되는 데이터 중에서 일부를 갱신할 때 데이터가 일치하지 않는 문제를 의미합니다. 특정한 값을 갖는 행을 수정 할 경우에 해당 값을 공유하는 모든 행을 변경해 주지 않으면 업데이트 된 내용이 일치하지 않게 됩니다. 이런 경우를 갱신 이상이 발생했다고 표현합니다.

• 고객주문

고객 아이디(PK)	주문 아이템(PK)	주문 일자	고객 이름	고객 등급
CS001	001	20211014	김민재	일반
CS003	002	20211015	이수민	프리미엄
CS003	002	20211016	이수민	프리미엄
CS003	004	20211017	이수민	프리미엄

→ 만약 수민의 고객등급이 **다이아몬드** 로 오른다면? 🙄

이러한 문제는 아래와 같이 테이블을 분리한 이후에 식별자 속성을 추가하여 하나는 식별자 종속되는 구조를 변경하여 해결 가능합니다.

• 고객주문

고객 아이디(PK)	주문 아이템(PK)	주문 일자
CS001	001	20211014
CS003	002	20211015
CS003	002	20211016
CS003	004	20211017

• 고객

고객 아이디(PK)	고객 이름	고객 등급
CS001	김민재	일반
CS002	이수민	프리미엄

테이블을 분리하여 고객 정보를 위한 식별자를 추가하여 문제를 해결할 수 있습니다. 테이블을 2개로 분리하고 고객 정보는 고객 아이디 식별자에 종속될 수 있도록 구성하여 **복합 식별자의 일부 칼럼에만 종속되는 문제를 해결** 했음을 확인할 수 있습니다. 이렇게 제 2정규화 작

업을 수행할 수 있습니다.

▼ 3) 제3정규화

✅ 제3정규화



제2정규화를 만족시키고 일반 속성 간에도 함수 종속 관계가 존재하지 않아야 함

제3정규화를 만족하려면 일반 속성들 간의 종속 관계가 존재하는 요소를 분리해야 합니다. 아래는 일반 속성에서 종속관계 발생하는 경우에 대한 예시입니다.

- 고객정보

고객 아이디(PK)	고객 이름	직업 코드	직업 이름
ABC123	김민재	JB023	개발자
ABC234	이수민	JB425	기획자
ABC456	오경석	JB666	마케터

고객 정보 테이블에서 PK에 해당하는 고객 아이디를 제외하면 다른 속성 간에는 종속 관계가 발생하면 안 됩니다. 하지만 직업 코드 속성과 직업 이름 속성 간의 관계를 살펴보면 직업 코드에 직업 이름 속성이 종속되는 관계임을 알 수 있습니다. 식별자인 고객 아이디를 제외한 '직업 코드'라는 일반 속성이 다른 속성을 종속하고 있기 때문에 제 3정규화를 위반한 경우라고 볼 수 있습니다.

이를 해결하기 위해서는 직업 정보만을 담고 있는 테이블을 따로 구성하여야 합니다.

- 고객정보

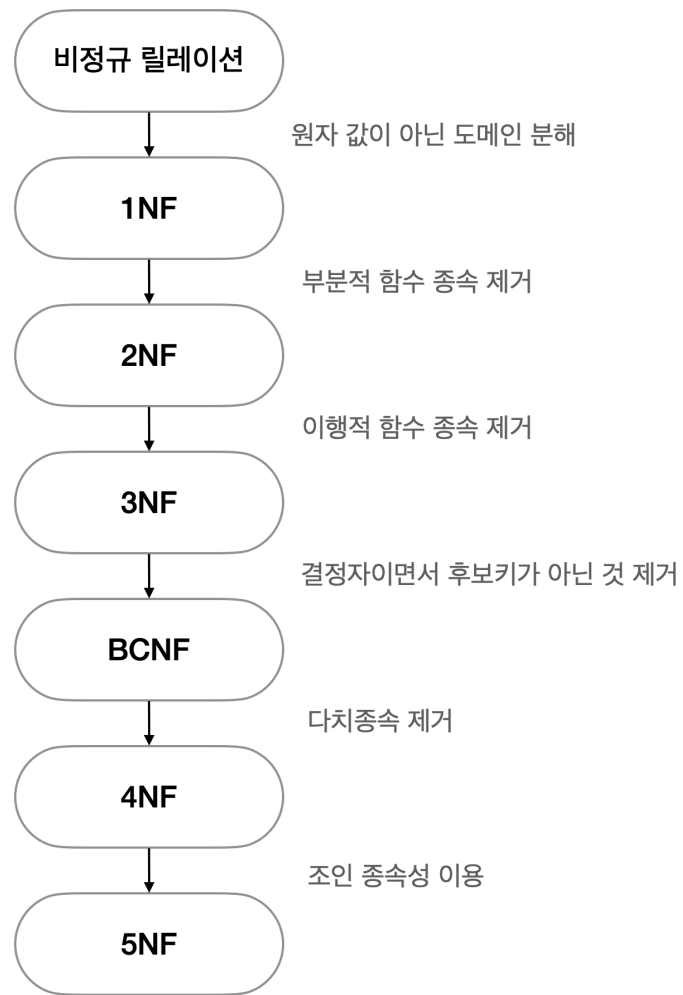
고객 아이디(PK)	고객 이름	직업 코드
ABC123	김민재	JB023
ABC234	이수민	JB425
ABC456	오경석	JB666

- 직업정보

직업 코드(PK)	직업 이름
JB023	개발자
JB425	기획자
JB666	마케터

기존에 '고객 정보'라는 하나의 테이블에서 일반 속성 간의 종속성이 발생하는 부분을 분리하여 '직업 정보'라는 다른 테이블로 구성하였습니다. 이제 직업 코드와 직업 이름 속성 간의 관계는 일반 속성의 종속 관계를 나타내지 않고 PK 식별자와 일반 속성의 관계로 변경되었습니다. 이제 제 3정규화를 통해 3정규형을 만족한다고 볼 수 있습니다.

✅ 정규화 유형



정규화 유형을 정리하면 다음과 같습니다.

정규화	설명
제1정규화	- 모든 속성은 반드시 하나의 값을 가져야 합니다. - 한 속성에 여러 속성값을 부여하거나 같은 유형의 속성이 여러 개인 경우 해당 속성을 분리합니다. (속성의 원자성 확보)
제2정규화	주식별자에 완전하게 함수 종속되지 않은 속성을 분리하여 종속 관계를 구성합니다. (부분종속 속성을 분리)
제3정규화	일반 속성간의 함수 종속성이 발생하지 않도록 분리합니다. (이전종속 속성을 분리)
보이스 - 코드 정규화 (BCNF)	결정자 안에 함수 종속을 가진 주식별자 속성을 분리합니다.
제4정규화	다치 종속성(Multi-Valued Dependency)을 제거합니다.
제5정규화	조인 속성(Join Dependency)을 제거합니다.

▼ 연습문제

☑ 문제 1

Q. 문제	아래 보기에 대한 설명으로 옳은 것은? <보기> 한 속성에 여러 개의 속성이 포함되어 있거나 같은 유형의 속성이 여러 개로 나뉘져있는 경우 해당 속성을 분리하는 단계
A. (1)	제1정규화
A. (2)	제2정규화
A. (3)	제3정규화
A. (4)	제4정규화

▼ 정답

(1)

☑ 문제 2

Q. 문제	다음 중 이행적 함수 종속 제거와 관련된 것은?
A. (1)	제1정규화

A. (2)	제2정규화
A. (3)	제3정규화
A. (4)	BCNF

▼ 정답

(3) 제3정규화는 이행적 함수 종속을 제거하는 것이며, 일반 속성들 간의 종속 관계가 존재하는 요소를 분리해야 한다

✓ 문제 3

Q. 문제	아래 보기에 대한 설명으로 옳은 것은? <주식별자에 완전하게 함수 종속되지 않은 속성을 분리하여 종속 관계를 구성한다>
A. (1)	제1정규화
A. (2)	제2정규화
A. (3)	제3정규화
A. (4)	BCNF

▼ 정답

(2)

03. 반정규화 개념과 설명

✓ 반정규화의 개념을 이해하고 성능을 향상시키는 전략에 대해 학습합니다

▼ 1) 반정규화를 통한 성능 향상 전략

✓ 반정규화의 정의

반정규화 (혹은 역정규화)에서 '반'이 갖는 의미는 '절반'이 아닌 '반대'입니다. 정규화를 수행하지 않은 모델을 지칭할 때 사용합니다. 반정규화는 성능 향상에 그 목적이 있습니다. 정규화는 성능 향상을 목적으로 중복된 데이터를 지우는 과정이었습니다. 반정규화는 정규화와 동일하게 '성능 향상'이라는 목적을 갖지만 데이터의 중복을 통해 목적을 달성한다는 측면에서 다른 의미를 지닙니다.

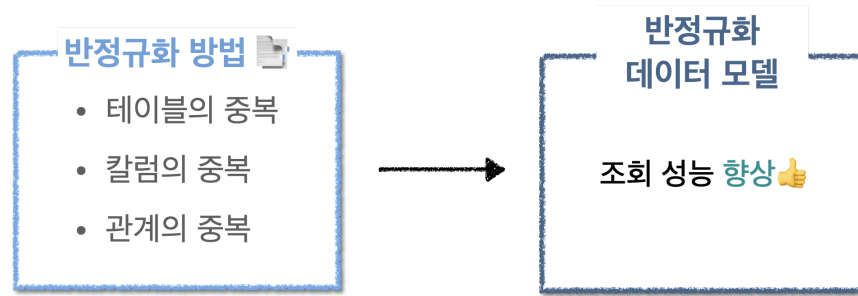
데이터의 정확성과 일관성을 의미하는 '데이터 무결성'의 측면에서 바라보면 반정규화는 있어서는 안 될 일이지만 그럼에도 데이터를 중복하여 반정규화를 적용하는 상황을 아래와 같이 정리해 볼 수 있습니다.

- 1. 데이터를 조회할 때 디스크 I/O량이 많아서 성능이 저하되는 경우입니다.
- 2. 테이블 간 경로가 너무 멀어 조인으로 인한 성능 저하가 예상되는 경우입니다.
- 3. 칼럼을 계산하여 읽을 때 성능이 저하될 것이라고 예상하는 경우입니다.

위와 같은 경우에 성능 향상을 위해서 데이터가 중복되는 것을 감수하고도 반정규화 수행합니다. 어떤 방식으로 반정규화를 수행할 수 있는지 알아보겠습니다.

항목	설명
정규화	데이터의 중복을 최소화 한다
반정규화	데이터의 중복을 허용한다

✓ 반정규화의 적용 방법



✓ 반정규화 절차

1. 반정규화 대상 조사	2. 다른 방법 유도 검토	3. 반정규화 적용
범위 처리 빈도수 조사	뷰(View) 테이블	테이블의 반정규화
대량의 범위 처리 조사	클러스터링 적용	속성의 반정규화
통계성 프로세스 조사	인덱스의 조정	관계의 반정규화
테이블 조인 개수	응용 어플리케이션	

여기서 핵심은 반정규화를 반드시 적용해야 하는 요소가 아니라는 점입니다. 반정규화를 수행할 때는 중복이 많이 발생하기 때문에 이에 영향을 받는 요소들이 많습니다. 따라서 반정규화의 대상을 먼저 조사하고 다른 방법이 있는지를 먼저 찾아보고 적용해야 합니다.

▼ 2) 반정규화의 기법

반정규화를 수행하는 방법은 테이블의 반정규화, 칼럼의 반정규화, 그리고 관계에서의 반정규화 이렇게 3가지가 존재합니다. 이에 대해 하나씩 알아보도록 하겠습니다.

✓ 테이블 반정규화

테이블 반정규화는 테이블 병합, 테이블 분할, 테이블 추가 이렇게 3가지로 나뉘집니다.

• 테이블 병합

- 여러 개의 테이블을 하나로 합치는 과정입니다.
- 테이블을 병합하면 데이터를 중복 저장하게 되지만 조회 과정의 성능 향상을 위해서 수행합니다.

기법	설명
1:1 관계 테이블 병합	- 1:1 관계를 통합하여 성능을 향상시키는 방법입니다. - 2개의 테이블을 하나의 테이블로 병합하여 테이블 간 조인 연산 제거합니다.
1:M 관계 테이블 병합	- 1:M 관계를 통합하여 성능을 향상시키는 방법입니다. - 2개의 테이블을 하나의 테이블로 병합하여 테이블 간 조인 연산 제거합니다.
슈퍼 / 서브 타입 테이블 병합	- 슈퍼/서브 관계를 통합하여 성능을 향상시키는 방법입니다. - 슈퍼/서브 관계를 하나의 테이블로 병합하여 테이블 간 조인 연산 제거합니다.

• 테이블 분할

- 특정 테이블을 여러 개의 테이블로 나누는 것을 의미합니다.

기법	설명
수직 분할	- 칼럼 단위의 테이블을 디스크 입력/출력을 분산하여 처리하기 위해 테이블을 1:1로 분리하여 처리하는 방법입니다. - 예를 들어, 게시글 테이블의 조회수 칼럼은 업데이트가 많이 일어나기 때문에 게시글 조회를 저장하기 위한 테이블을 따로 두고 처리하는 것이 성능 향상에 좋습니다.
수평 분할	- 행 단위로 집중 발생하는 디스크 입/출력 및 데이터 접근 효율을 높여 성능을 향상시키기 위해 행 단위로 테이블을 분리하는 방법입니다. - 하나의 테이블에 있는 값을 기준으로 테이블을 분할합니다. - 예를 들어, 요금 납부 테이블을 년/월/일 테이블로 분할합니다.

• 테이블 추가

- 특정 테이블을 추가하는 것을 의미합니다.

- 데이터의 중복 저장의 비효율이 발생하더라도 조회의 성능을 높이기 위해 사용합니다.

기법	설명
중복 테이블 추가	- 다른 업무이거나 서버가 다른 경우 동일한 테이블 구조를 중복하여 원격 조인을 제거하여 성능을 향상시킵니다.
통계 테이블 추가	- SUM, AVG, COUNT 등을 미리 수행하고 계산하여 조회 시 성능이 향상됩니다.
이력 테이블 추가	- 이력 테이블 중에서 마스터 테이블에 존재하는 레코드를 중복하여 이력 테이블에 적용합니다.
부분 테이블 추가	- 특정 테이블에서 자주 조회하는 칼럼을 파악하여 해당 칼럼을 모아 놓은 별도의 반정규화된 테이블 생성합니다.

✓ 칼럼 반정규화

기법	설명
중복 칼럼 추가	- 조인 연산으로 인한 성능 저하를 방지하지 위해 중복 칼럼을 추가하여 조인 연산을 수행하지 않도록 합니다. - 인사 테이블과 부서 테이블을 조인하여 부서 테이블에 있는 부서명 칼럼을 조회할 때, 부서명 칼럼을 인사 테이블에도 중복으로 넣어 부서 테이블에 조인하지 않고 조회합니다.
파생 칼럼 추가	- 트랜잭션이 처리되는 시점에 계산하는 값을 미리 계산한 칼럼을 따로 구성합니다. - 예를 들어, 매출 금액 칼럼이 있으면 총 매출 금액을 미리 계산한 칼럼을 추가해 놓습니다.
이력 테이블 칼럼 추가	- 많은 데이터를 처리할 때 불특정한 날에 대한 조회나 최근 값을 조회할 때 나타날 수 있는 성능 저하를 예방하기 위해 이력 테이블에 칼럼 추가합니다. - 예를 들어, 최근에 조회한 값이 있는지를 확인하거나 시작 & 종료 일자 등에 대한 칼럼을 따로 지정합니다.
PK에 의한 칼럼 추가	- 복합의미를 갖는 PK(복합키)를 단일 속성으로 구성한 경우 단일 PK 안에서 특정 값을 별도로 조회하는 경우에 성능 저하가 나타날 수 있습니다. - 이때 이미 PK 안에 데이터가 존재하지만 성능 향상을 위해 일반 속성으로 생성하는 방법이 PK에 의한 칼럼 추가 반정규화입니다. - 예를 들어, 주문번호의 값 구성이 '상품코드 + 주문일자'로 구성된 경우 주문번호 안에 존재하는 주문일자를 일반 속성으로 빼내고 주문일자 칼럼을 인덱스로 설정합니다.
응용 시스템의 오작동을 위한 칼럼 추가	- 비즈니스적으로 의미는 없지만 사용자의 데이터를 처리하다가 잘못된 경우에 원래 값으로 복구를 원할 때 이전 데이터를 임시적으로 중복하여 보관하는 방법입니다.



참고 - 복합키(Composite Key)

기본키로 사용할 수 있는 칼럼이 없을 때 2개 이상의 키를 합쳐 하나의 기본키로 활용하는 것을 의미합니다. 데이터를 대표하는 키가 여러 개의 칼럼으로 구성된 것을 의미합니다. 복합키를 사용하는 경우 SELECT 문을 사용하여 WHERE을 통해 조건을 건 조회를 할 때 어떻게 조회하는지에 따라서 쿼리 성능이 많이 달라집니다.

칼럼의 반정규화는 특정 칼럼을 추가하여 데이터 모델 내에서 중복으로 데이터가 저장됨에도 성능을 향상시킬 수 있는 방법입니다.

중복 칼럼을 추가하여 조인 연산을 제거, 추가 칼럼을 만들어 계산된 값을 미리 만들어 놓고 보여주는 형태로 연산 감소, 이력 테이블에 특정 칼럼을 추가해서 조회 성능 향상, PK 칼럼 내에 특정한 규칙에 의해 데이터가 저장되어 있지만 일반 칼럼으로 만드는 방법, 응용 시스템의 오작동을 위해 칼럼을 추가시키는 방법이 있습니다.

✓ 관계 반정규화

기법	설명
중복 관계 추가	- 데이터를 처리하기 위한 여러 경로를 거쳐 조인이 가능합니다. - 이때 발생할 수 있는 성능 저하를 예방하기 위해 추가적인 관계를 맺는 방법이 관계의 반정규화입니다.

여러 경로에 걸쳐 테이블 조인을 하는 경우 조인 연산 자체를 줄여서 조회 성능을 향상시키는 방법입니다. 만약 $A \rightarrow B \rightarrow C$ 형태로 테이블 조인이 발생한다면 $A \rightarrow C$ 로 줄이는 것이 관계 반정규화의 목적입니다. 관계 반정규화는 데이터 무결성을 위반하더라도 데이터 처리를 할 때 성능을 향상시키기 위해 사용하는 반정규화 기법입니다.

▼ 연습문제

✅ 문제 1

Q. 문제	테이블 반정규화 기법에 대한 것으로 옳지 않은 것은?
A. (1)	중복 테이블 추가
A. (2)	통계 테이블 추가
A. (3)	이력 테이블 칼럼 추가
A. (4)	1:M 관계 테이블 병합

▼ 정답

(3) 이력 테이블 칼럼 추가는 칼럼 반정규화 기법 중 하나이다

✅ 문제 2

Q. 문제	반정규화 절차 순서가 옳은 것은?
A. (1)	반정규화 대상 조사 → 반정규화 적용 → 다른 방법 유도 검토
A. (2)	반정규화 적용 → 다른 방법 유도 검토 → 반정규화 대상 조사
A. (3)	반정규화 적용 → 반정규화 대상 조사 → 다른 방법 유도 검토
A. (4)	반정규화 대상 조사 → 다른 방법 유도 검토 → 반정규화 적용

▼ 정답

(4)

✅ 문제 3

Q. 문제	반정규화 기법 중에서 특정 값의 범위에 따라 분할하는 것은?
A. (1)	수평분할
A. (2)	수직분할
A. (3)	중복 칼럼 추가
A. (4)	테이블 추가

▼ 정답

(1)