정규화란?

데이터베이스 정규화는 **데이터베이스 내의 데이터 구조를 조직화하고 최적화하는 과정**이다.
정규화를 하는 목적은 **데이터 중복을 제거하고, 효율성을 향상시키며, 데이터 무결성을 보장하기 위함**이다.

데이터 정규화에는 다음과 같이 여러 단계가 있다.

제1 정규화 (1NF)
제2 정규화 (2NF)
제3 정규화 (3NF)

이상 종류란 정규화를 거치지 않은 데이터에서 발생할 수 있는 현상을 말한다.

- 삽입 이상 (Insertion Anomaly) : 데이터 삽입 시 의도와 다른 값들도 삽입
- 삭제 이상 (Delete Anomaly) : 데이터 삭제 시 의도와 다른 값들도 연쇄 삭제
- 갱신 이상 (Update Anomaly) : 속성값 갱신 시 일부 튜플만 갱신되어 모순 발생

제1 정규화 (1NF)

데이터베이스의 **각 컬럼이 원자 값(하나의 값)을 가지도록 하는 과정**이다.
즉, **각 컬럼은 하나의 데이터만 저장하며, 다중 값을 갖지 않는다.**

제1 정규화 적용 전

예를 들어 아래와 같이 **"수강신청 테이블"**이 존재한다고 해보자.

수강과목ID	수강과목	수강자
1	AI	홍길동
2	JAVA	이순신, 홍길동
3	Java Script	유관순

지금 이 테이블에서 JAVA 과목은 이순신, 홍길동 여러 수강자를 갖고 있기 때문에 제1 정규화를 만족하지 못한다.
이런식으로 테이블이 짜여 있다면 **다음과 같은 문제가 발생**할 수 있다.

- 데이터의 중복 발생이 가능하다.
- 데이터 무결성이 유지되지 않는다.
- 데이터의 수정, 삭제, 삽입 연산에 이상 문제가 있을 수 있다.
- 데이터의 구조가 불분명하다.

발생할 수 있는 이상현상

▶ 갱신 이상
홍길동이 JAVA 과목을 Python로 바꿀 경우 이순신이 듣고 있는 JAVA 과목도 바뀌어 버린다.
(UPDATE 수강신청 SET 수강과목 = 'Python' WHERE 수강과목 = 'JAVA' AND 수강자 = '홍길동';)

▶ 삭제 이상
홍길동이 JAVA 과목 수강을 취소할 경우 같이 듣고 있는 이순신의 수강 정보도 삭제된다.
(DELETE FROM 수강신청 WHERE 수강과목 = 'JAVA' AND 수강자 = '홍길동';)

제1 정규화 적용 후

다음은 각 컬럼은 하나의 데이터만 저장하는 **제1 정규화를 적용 시킨 테이블**이다.
이러한 제1 정규화가 적용된 테이블을 **"제1 정규형 테이블"**이라고 한다.

수강과목ID	수강과목	수강자
1	AI	홍길동
2	JAVA	이순신
3	Java Script	유관순
2	JAVA	홍길동

제1 정규화를 적용 시킨후 전과 비교하면 **다음과 같은 장점**을 얻을 수 있다.

- 데이터의 중복이 크게 줄어들어 공간적 효율성 향상
- 데이터 무결성 유지에 도움
- 데이터의 수정, 삭제, 삽입 연산에 이상 문제 방지
- 데이터 구조가 단순해지고 명확해진다.

제2 정규화 (2NF)

제1 정규화를 완료한 테이블에서, 부분 함수 종속성을 제거하는 과정이다.
모든 기본 키의 부분집합에 의존하는 컬럼들을 분리해 새로운 테이블을 만들고 그 사이의 관계를 설정한다.
쉽게 설명하면, **현재 테이블의 주제와 관련 없는 컬럼을 다른 테이블로 빼는 작업**이다.

제2 정규화 적용 전

다음 테이블은 **제1 정규화가 완료**된 **"회원 주문 테이블"**이다.
이 테이블을 보면 **회원이름과 회원주소는 회원ID에 종속**되어 있으며, **상품명과, 가격은 상품 코드에 종속**되어 있다.

회원ID	회원이름	회원주소	상품코드	상품명	가격
101	홍길동	서울시	S100	소파	200
102	이순신	부산시	B100	침대	300
103	유관순	대전시	S100	소파	200
101	홍길동	서울시	S100	의자	150

이 테이블에서는 **다음과 같은 문제가 발생**할 수 있다.

- 한 테이블에서 회원 정보, 상품 정보, 구매 내역이 모두 포함되어 있어 데이터 중복이 발생하고 무결성이 유지되지 않는다.

발생할 수 있는 이상 현상

▶ 삽입 이상
새로운 상품이 등록된다면 테이블의 구조상 회원 정보와 연관된 컬럼도 채워야 하는 문제가 발생한다.
원하는 정보만 추가 할 수가 없는 것이다.
새로운 책상 상품이 추가된다면 책상 상품만 추가하는 것이 아니라, 회원정보인 회원

ID, 회원 이름과 주소까지 함께 입력해야 한다.
(INSERT INTO 회원주문(회원ID, 회원이름, 회원주소, 상품코드, 상품명, 가격) VALUES(?, '?, ?, ?, '책상', ?);)

▶ 수정 이상

홍길동이 주소를 경기도로 변경하면 홍길동이 있는 연관 행들을 모두 수정해야 한다. 일부 행을 수정하지 못할 경우 데이터의 범주 불일치가 발생한다.
(UPDATE 회원주문 SET 회원주소 = '경기도' WHERE 회원이름 = 홍길동;)

▶ 삭제 이상

의자 상품을 삭제 한다면, 상품의 가격, 상품코드가 사라지고 의자를 구매한 회원의 정보 또한 삭제되는 문제가 발생한다.
(DELETE FROM 회원주문 WHERE 상품명 = '의자';)

제2 정규화 적용 후

제2 정규화를 적용하면 다음과 같이 3개의 테이블로 분리할 수 있다.

회원 테이블

회원ID	회원이름	회원주소
101	홍길동	서울시
102	이순신	부산시
103	유관순	대전시

상품 테이블

상품코드	상품명	가격
S100	소파	200
B100	침대	300
S100	의자	150

구매 테이블

회원ID	상품코드
101	S100
102	B100
103	S100
101	S100

제2 정규화를 적용 시킨 후 전과 비교하면 **다음과 같은 장점**을 얻을 수 있다.

- 각 테이블이 회원 정보, 상품 정보, 구매 정보 등 각각의 역할에 집중하여 관리되고, 중복이 크게 줄어들며 데이터 무결성이 유지된다.
- 전체적인 데이터 관리 및 유지 보수가 향상된다.

제3 정규화 (3NF)

제2 정규화를 완료한 테이블에서, 이행적 함수 종속성을 제거하는 과정이다.
이행적 함수 종속이란 A -> B 종속, B -> C 종속, A -> C 종속인 관계를 말한다.

제3 정규화 적용 전

학생ID	학생이름	강의ID	강의명	교수ID	교수이름
S001	김영희	L001	수학	P001	이은택
S001	김영희	L002	영어	P002	김진희
S002	이철수	L001	수학	P001	이은택
S003	최민수	L003	과학	P003	박현정

위 테이블은 다음과 같이 이행적 함수 종속이 발생한다.
학생이름은 학생ID에 종속, 강의명은 강의 ID에 종속, 교수이름은 교수ID에 종속되어 있다.
그리고, **강의 ID를 통해 교수 ID와 교수 이름 정보에도 간접적으로 종속**되어 있다.
이 경우 **이행적 함수 종속 관계는 다음과 같다.**

이러한 이행적 종속성 때문에 중복 데이터와 데이터 무결성 문제가 발생할 수 있다.

발생할 수 있는 이상 현상

▶ 삽입 이상

새 과목을 추가하려고 할 때, 아직 수강하는 학생이 없으면 해당 과목을 추기하기 어렵다.

왜냐하면 학생ID와 학생이름이 null로 남게 되기 때문이다.

예를 들어, '화학'이라는 새로운 강의를 P004 교수님('최선영')이 맡게 되었다면, 다음과 같은 쿼리를 사용해야 한다.

(INSERT INTO 수강신청목록 (학생ID, 학생이름, 강의ID, 강의명, 교수ID, 교수이름) VALUES (NULL, NULL, 'L004', '화학', 'P004', '최선영');)

하지만 이렇게 입력하면 학생ID와 학생이름이 NULL이 되어, 알맞지 않은 데이터가 저장된다.

▶ 수정 이상

강의를 담당하는 교수님이나 강의명이 변경될 경우, 중복 데이터로 인해 여러 행을 수정해야 하고, 그로 인해 일관성이 깨질 수 있다.

예를들어, 수학 강의명이 고급수학으로 변경되었다면, 모든 연관 행의 데이터를 수정해야 한다.

(UPDATE 수강신청목록 SET 강의명 = '고급수학' WHERE 강의명 = '수학';)

▶ 삭제 이상

특정 강의를 수강한 학생이 삭제되면, 해당 강의와 관련된 정보도 함께 제거 된다.

예를들어 김영희 학생이 수학 강의를 수강취소했다면 수학 강의에 대한 정보와 수학 강의를 담당하는 이은택 교수님에 대한 정보가 손실된다.

(DELETE FROM 수강신청목록 WHERE 학생이름 = '김영희' AND 강의명= '수학';)

제3 정규화 적용 후

학생 정보 테이블

학생ID	학생이름
S001	김영희
S002	이철수
S003	최민수

강의 정보 테이블

강의ID	강의명	교수ID
L001	수학	P001
L002	영어	P002
L003	과학	P003

교수 정보 테이블

교수ID	교수이름
P001	이은택
P002	김진희
P003	박현정

수강 신청 정보 테이블

학생ID	강의ID
S001	L001
S001	L002
S002	L001
S003	L003

제3 정규화를 적용 시킨후 전과 비교하면 **다음과 같은 장점**을 얻을 수 있다.

- 각 테이블은 원자적인 값을 가지게 되고 데이터 중복이 줄어든다.
- 이상 현상을 피할 수 있으며 유지 보수가 훨씬 용이해진다.