< 이상(Anomaly)과 함수적 종속 >

- 실제로 사용되는 데이터를 오류 없이 무결성과 독립성 그리고 일관성을 유지하면서 상호 융통성 있는 관계 형성을 통해 어떻게 효율적으로 처리하느냐이다.
- 우리가 다시 살펴보아야 하는 것은 데이터베이스를 설계하는 과정에 대한 내용.
- 데이터베이스는 개념적 설계를 거쳐 논리적 설계, 물리적 설계 단계로 진행하면서 대량의 데이터를 여러 사용자가 사용함에 있어 정확하고 효율적인 처리가 이루어지도록 설계가 이루어져야 함.
- 논리적 설계 단계에서 무결성 등을 유지하기 위해 이상(Anomaly)과 종속에 대한 문제를 살펴보아야 함.

1. 이상(Anomaly)

- 이상(Anomaly) 현상 - 데이터베이스의 논리적 설계 시 하나의 릴레이션에 많은 속성들이 존재하여 데이터의 중복과 종속으로 인해 발생되는 문제점을 말함. 릴레이션을 처리하는데 여러 가지 문제를 초래하게 됨.
- 이상의 종류 - 삭제 이상, 삽입 이상, 갱신 이상

① 삭제 이상(Delete Anomaly)

- 관계 데이터베이스에서 삭제는 튜플 단위로 이루어짐. 삭제 이상은 테이블에서 하나의 자료를 삭제하고자 하는 경우 그 자료가 포함된 튜플이 삭제됨으로 인해 **원하지 않은 자료까지 함께 삭제**가 이루어져 발생하는 문제점을 말함.
- 다음은 '고객번호', '제품번호', '제품명', '단가', '주문량' 속성으로 구성된 [고객주문] 테이블

고객번호	제품번호	제품명	단가	주문량
A012	S-321	SD메모리	25,000	2
A012	M-789	메모리	28,000	1
A023	K-002	키보드	5,000	1
A123	K-012	헤드셋	10,000	2
A134	M-123	마우스	6,000	4
A134	S-321	SD메모리	25,000	2
A321	K-012	헤드셋	10,000	1
A567	M-123	마우스	6,000	2
A789	M-123	마우스	6,000	3
A789	S-567	스캐너	100,000	1

- [고객주문] 테이블은 한 명의 고객이 여러 제품을 주문하고, 하나의 제품을 여러 고객이 주문하고 있는 상황을 나타내는 테이블
- 따라서 [고객주문] 테이블에서는 하나의 속성만으로는 튜플들을 식별할 수 없고, '고객번호'와 '제품번호'가 조합된 합성키 (고객번호, 제품번호)가 기본키가 됨.
- 이 중에서 고객번호 'A789'가 주문한 제품 중 스캐너 주문을 취소한다면, [고객주문] 테이블에서 스캐너를 삭제해야 함.
- 삭제는 튜플 단위로 이루어지기 때문에 스캐너가 포함된 튜플이 삭제가 됨. 이때 스캐너가 포함된 튜플 전체가 삭제되며, 스캐너에 대한 가격 정보도 함께 삭제되어 스캐너에 대한 가격을 알 수 없게 됨. [고객주문] 테이블에서는 스캐너를 주문한 경우가 더 이상 없기 때문.
- 이와 같이 하나의 자료만 삭제하고 싶지만 그렇지 못하고 그 자료가 포함된 튜플 전체가 삭제됨으로 인해 **원하지 않는 정보가 손실되는 문제점을 삭제 이상**이라고 함.

② 삽입 이상(Insertion Anomaly)

- 관계 데이터베이스에서 삽입 역시 튜플 단위로 이루어짐. 이때 삽입하는 과정에서 **원하지 않는 자료가 삽입**된다든지 또는 삽입하는데 **자료가 부족해 삽입이 되지 않아** 발생하는 문제점을 삽입 이상이라고 함.
- 위 [고객주문] 테이블에서 새로운 제품을 판매하기 위해 새로운 제품에 대한 정보로 '제품번호', '제품명', '단가'를 삽입하려고 함.
- 그러나 [고객주문] 테이블에서는 (고객번호, 제품번호)로 조합된 합성키가 기본키이기 때문에 '고객번호'가 없다면 삽입할 수 없음. 기본키에는 NULL이 올 수 없기 때문.
- 따라서 [고객주문] 테이블에 새로운 제품 정보를 삽입하기 위해서는 고객이 주문을 해서 '고객번호'

를 알기 전까지는 새로운 제품에 대한 정보를 삽입할 수 없는 현상이 발생하게 됨.

- 이와 같이 삽입 작업을 수행하는 경우 **원하지 않게 삽입이 되지 않는 현상**을 삽입 이상이라고 함.

③ 갱신 이상(Update Anomaly)

- 관계 데이터베이스의 자료를 갱신하는 과정에서 정확하지 않거나 일부의 튜플만 갱신됨으로 인해 **정보가 모호해지거나 일관성이 없어져** 정확한 정보의 파악이 안 되는 현상을 말함.
- 위 [고객주문] 테이블에서 마우스의 단가를 5,000으로 변경하려고 함. [고객주문] 테이블에서 마우스가 포함된 튜플은 3개임.
- 그런데 마우스가 포함된 3개의 튜플을 모두 변경하지 않고 일부만 변경한다면 마우스의 단가를 파악할 때 5,000인지 6,000원이지 알 수 없게 됨.
- 이와 같이 자료의 갱신에서 잘못된 작업으로 인해 **정보의 일관성이 없어져 정확한 정보를 파악하지 못하는 현상**을 갱신 이상이라고 함.

2. 함수적 종속(Functional Dependency)

- 이상(Anomaly)과 함께 관계 데이터베이스에서 고려해야 할 것 중에 하나가 종속.
- 종속 – 어떤 릴레이션에서 속성 A, B가 있을 때 임의 튜플에서 A의 값에 따라 B의 값이 달라진다면 B는 A에 함수적으로 종속되었다고 하고, 기호로는 A → B로 표기.
- B가 A에 종속되어 A 값을 알면 B 값을 알 수 있을 때 A를 '결정자'라고 하고, B를 '종속자'라고 함.
- 종속의 종류 – 완전 함수 종속, 부분 함수 종속, 이행적 함수 종속
- 다음은 '학번', '성명', '수강과목', '학년'으로 구성된 [학생] 테이블

학번	성명	수강과목	학년
990111	김철수	정보통신	1
981010	이철준	컴퓨터	3
990223	박태인	데이터베이스	1
972020	김길동	운영체제	2
981533	오준석	산업공학	3
961533	최길동	컴퓨터	4
962111	이철준	데이터베이스	4

- [학생] 테이블에서 기본키인 '학번'을 알면 그 학생의 성명, 수강과목, 학년을 알 수 있음. 이때 '성명', '수강과목', '학년'은 '학번'에 종속되었다고 함.
- 표기는 다음과 같음.
  ①학번 → 성명 ②학번 →수강과목 ③학번 → 학년

① 완전 함수 종속과 부분 함수 종속

- 완전 함수 종속(Full Functional Dependency) – 릴레이션에서 한 속성이 **오직 기본키에만 종속**이 되는 경우
- 부분 함수 종속(Partial Functional Dependency) – 릴레이션에서 한 속성이 **기본키가 아닌 다른 속성에 종속**이 되거나 또는 **기본키가 2개 이상 합성키(복합키)로 구성된 경우 이 중 일부 속성에 종속**이 되는 경우
- 위 [학생] 테이블에서 '성명', '수강과목', '학년'은 기본키인 '학번'을 알아야 알 수 있으므로 '성명', '수강과목', '학년'은 '학번'에 완전 함수 종속되었다고 함.
- 다음은 '고객번호', '제품번호', '제품명', '단가', '주문량' 속성으로 구성된 [고객주문] 테이블

고객번호	제품번호	제품명	단가	주문량
A012	S-321	SD메모리	25,000	2
A012	M-789	메모리	28,000	1
A023	K-002	키보드	5,000	1
A123	K-012	헤드셋	10,000	2
A134	M-123	마우스	6,000	4
A134	S-321	SD메모리	25,000	2
A321	K-012	헤드셋	10,000	1
A567	M-123	마우스	6,000	2
A789	M-123	마우스	6,000	3
A789	S-567	스캐너	100,000	1

- 위 [고객주문] 테이블에서는 '고객번호'와 '제품번호'가 조합된 (고객번호, 제품번호)가 기본키임. [고객주문] 테이블에서 '주문량' 속성은 기본키인 '고객번호'와 '제품번호'를 모두 알아야 구분할 수 있음. 이런 경우 '주문량' 속성은 기본키에 완전 함수 종속되었다고 함.
- 표기는 다음과 같음. **(고객번호, 제품번호) → 주문량**
- 반면, '제품명'은 기본키인 '고객번호'와 '제품번호'를 모두 알아도 값을 구분할 수 있지만 기본키의 일부인 '제품번호'만 알아도 '제품명'을 알 수 있다. 이와 같은 경우 '제품명'은 기본키에 부분 함수 종속되었다고 함.
- 표기는 다음과 같음. **제품번호 → 제품명**
- [고객주문] 테이블의 함수 종속 관계를 다이어그램을 이용하여 표현하면 다음과 같음.

- 위 다이어그램에서 '주문량'은 '고객번호'와 '제품번호'의 조합에 종속됨을 나타냄. 반면, '제품명'은 '제품번호' 한 가지에만 종속됨을 나타냄.

② 이행적 함수 종속(Transitive Functional Dependency)

• 릴레이션에서 A, B, C 3가지 속성 간의 종속이 A → B, B → C 일 때, A → C 가 성립이 되는 경우. 즉 **A를 알면 B를 알 수 있고, B를 알면 C를 알 수 있을 때**, A를 알면 C를 알 수 있는 경우

• 다음은 '제품번호', '제품명', '단가' 속성으로 구성된 [제품] 테이블

제품번호	제품명	단가
S-321	SD메모리	25,000
M-789	메모리	28,000
K-002	키보드	5,000
K-012	헤드셋	10,000
M-123	마우스	6,000
S-567	스캐너	100,000

- [제품] 테이블에서는 '제품번호'를 알면 '제품명'을 알 수 있음. 또 '제품명'을 알면 '단가'를 알 수 있음. 결국 '제품번호'를 알면 '단가'를 알 수 있음. 이와 같은 경우를 이행적 함수 종속이라고 함.
- [제품] 테이블의 종속 관계는 다음과 같음.

  제품번호 → 제품명　　제품명 → 단가　　제품번호 → 단가

< 정규화 >

1. 정규화(Normalization)

• 논리적 설계 단계에서 발생할 수 있는 종속으로 인한 이상(Anomaly) 현상의 문제점을 해결하기 위해, 속성들 간의 종속 관계를 분석하여 여러 개의 릴레이션으로 분해하는 과정.

• 정규화되는 과정. 정규화의 종류로는 제1정규형, 제2정규형, 제3정규형, BCNF, 제4정규형, 제5정규형

2. 정규형의 종류

① 제1정규형(1NF : First Normal Form)

• 한 릴레이션을 구성하는 모든 도메인이 원자값만으로 구성되도록 하는 정규형

• 다음은 '회원번호', '연락처', '수강과목', '수강료'로 구성된 [회원] 테이블

회원번호	성명	연락처	수강과목	수강료
10010	박순신	123-4567	POP글씨	40,000
			지점토공예	40,000
20020	이감찬	234-1122	펜글씨	30,000
20030	김길동	321-4321	지점토공예	40,000
			기타	50,000

- 위 [회원] 테이블에서 '박순신'과 '김길동' 회원은 1명의 회원이 여러 과목을 수강하고 있음.
- 그런데 '박순신' 회원과 '김길동' 회원에 대한 중복이 되는 속성값 '회원번호', '성명', '연락처'에 해당하는 튜플을 하나로 합쳐서 나타내고 있음.
- 데이터베이스에서는 검색, 삽입, 삭제 등 여러 가지 작업이 튜플 단위로 이루어지기 때문에 '박순신' 회원과 '김길동' 회원과 같이 튜플을 하나로 합쳐서 표현하면 원활하게 수행되지 못함.
- 따라서 위 [회원] 테이블이 각각의 튜플로 구성되도록 회원 정보를 나타내는 [회원] 테이블과 수강과목에 대한 정보를 나타내는 [강좌] 테이블로 분해하면 다음과 같음.

회원(<u>회원번호</u>, 성명, 연락처)
강좌(<u>수강과목</u>, 수강료)

회원

<u>회원번호</u>	성명	연락처
10010	박순신	123-4567
20020	이감찬	234-1122
20030	김길동	321-4321

강좌

<u>수강과목</u>	수강료
POP글씨	40,000
지점토공예	40,000
펜글씨	30,000
지점토공예	40,000
기타	50,000

- 이와 같이 모든 도메인이 각각의 튜플로 구성되도록 즉, 원자값만으로 구성되도록 분해하는 과정을 제1정규형이라고 함.

② 제2정규형(2NF : Second Normal Form)

• 제1정규형을 만족하면서 릴레이션을 구성하는 모든 속성이 기본키에 완전 함수 종속이 되도록 분해하는 과정.

• 즉, 제2정규형에서는 릴레이션에 존재하는 부분 함수 종속을 제거하고 모든 속성이 기본키에 완전 함수 종속이 되도록 함.

• 다음은 '고객번호', '제품번호', '제품명', '주문량'의 속성을 가진 [고객주문] 테이블

<u>고객번호</u>	<u>제품번호</u>	제품명	주문량
A012	S-321	SD메모리	2
A012	M-789	메모리	1
A023	K-002	키보드	1
A123	K-012	헤드셋	2
A134	M-123	마우스	4
A134	S-321	SD메모리	2
A321	K-012	헤드셋	1
A567	M-123	마우스	2
A789	M-123	마우스	3
A789	S-567	스캐너	1

- [고객주문] 테이블에서 '고객번호'와 '제품번호'가 조합된 합성키(복합키)가 기본키가 됨.
- [고객주문] 테이블의 종속 관계를 살펴보면 '주문량' 속성값은 '고객번호'와 '제품번호' 모두 알아야 구분할 수 있으므로 기본키인 (고객번호, 제품번호)에 완전 함수 종속됨.

(고객번호, 제품번호) → 주문량

- 반면 '제품명' 속성값은 기본키 (고객번호, 제품번호)의 일부인 '제품번호'만 알아도 구분할 수 있으므로 부분 함수 종속 관계에 있음.

제품번호 → 제품명

- 따라서 이와 같이 부분 함수 종속 관계가 있는 테이블을 기본키에 완전 함수 종속이 되도록 분해하면 다음과 같이 분해할 수 있음.

주문량(고객번호, 제품번호, 주문량)
제품(제품번호, 제품명)

주문량

고객번호	제품번호	주문량
A012	S-321	2
A012	M-789	1
A023	K-002	1
A123	K-012	2
A134	M-123	4
A134	S-321	2
A321	K-012	1
A567	M-123	2
A789	M-123	3
A789	S-567	1

제품

제품번호	제품명
S-321	SD메모리
M-789	메모리
K-002	키보드
K-012	헤드셋
M-123	마우스
S-321	SD메모리
K-012	헤드셋
M-123	마우스
M-123	마우스
S-567	스캐너

- 이와 같이 [고객주문] 테이블을 [주문량] 테이블과 [제품] 테이블로 분해하면 [주문량] 테이블에서 '주문량'은 기본키인 (고객번호, 제품번호)에 완전 함수 종속이 되고, [제품] 테이블에서 '제품명'은 기본키인 '제품번호'에 완전 함수 종속이 되어 제2정규형을 만족하게 됨.

③ 제3정규형(3NF : Third Normal Form)

- 제2정규형을 만족하면서 릴레이션을 구성하는 속성들 간에 이행적 함수 종속 관계를 분해하여 비이행적 함수 종속이 되도록 하는 과정
- 다음은 '학번', '전공', '담당교수' 속성으로 구성된 [수강] 테이블

학번	전공	담당교수
0001	컴퓨터	김선수
0002	기계	박길동
0003	토목	이찬성
0004	컴퓨터	김선수
0005	기계	박길동

- [수강] 테이블의 종속 관계를 살펴보면 다음과 같음.
- '학번'을 알면 그 학생의 '전공'을 알 수 있음. 즉, '전공'은 '학번'에 종속되어 있음. 또한 '전공'을 알면 '담당교수'를 알 수 있음. 즉, '담당교수'는 '전공'에 종속되어 있음. 결국 '학번'을 알면 '전공' 속성값을 알 수 있고, '담당교수' 속성값도 알 수 있게 됨. 즉, '학번'과 '담당교수' 속성 간에 이행적 함수 종속 관계가 있는 것.
- 따라서 이행적 함수 종속 관계가 있는 [수강] 테이블을 분해하면 다음과 같이 분해할 수 있음.

학생(학번, 전공)
교수(전공, 담당교수)

학생

학번	전공
0001	컴퓨터
0002	기계
0003	토목
0004	컴퓨터
0005	기계

교수

전공	담당교수
컴퓨터	김선수
기계	박길동
토목	이찬성

- 이와 같이 [수강] 테이블을 [학생] 테이블과 [교수] 테이블로 분해하면 각 테이블이 기본키에 완전 함수 종속 관계로 유지되면서 이행적 함수 종속 관계도 해결되어 제3정규형을 만족하게 됨.

④ 보이스-코드 정규형(BCNF : Boyce-Codd Normal Form)

• 제3정규형을 만족하면서 릴레이션에서 모든 결정자가 후보키가 되도록 하는 과정

※ 후보키 : 릴레이션에서 각 튜플을 유일하게 식별할 수 있는 속성이나 속성의 집합

• 다음은 '회원번호', '수강과목', '강사' 속성으로 구성된 [등록] 테이블

회원번호	수강과목	강사
10010	POP글씨	최수지
10010	서예	김선수
20020	기타	이영춘
20030	네일아트	이태선
20030	POP글씨	최수지
30010	서예	박길동
30010	POP글씨	김정미

- [등록] 테이블에서 한 명의 회원이 여러 과목을 수강할 수 있으므로 '회원번호' 하나의 속성으로는 후보키가 될 수 없음.
- [등록] 테이블에서 후보키가 될 수 있는 것은 합성키(복합키)로 (회원번호, 수강과목) 또는 (회원번호, 강사)임. (수강과목, 강사)는 후보키가 될 수 없음. 수강과목이 'POP글씨'이고, 강사가 '최수지'인 경우는 회원번호가'10010'과 '20030' 두 가지로 서로 튜플을 식별할 수 없기 때문.
- 후보키 (회원번호, 수강과목)과 (회원번호, 강사) 중 기본키를 (회원번호, 수강과목)으로 지정하면 다음과 같은 종속 관계가 성립됨.

(회원번호, 수강과목) → 강사

- 또한 [등록] 테이블에서는 한 과목을 여러 명의 강사가 강의할 수 있음을 알 수 있음. 'POP글씨'를 강사 '최수지'와 '김정미'가 강의하고 있고, '서예'는 '김선수'와 '박길동' 강사가 강의하고 있음. 따라서 수강과목과 강사 간의 종속 관계는 강사를 알면 수강과목을 알 수 있는 형태. 수강과목과 강사 간에는 다음과 같은 종속 관계가 성립됨.

강사 → 수강과목

- 이때 '강사' 속성은 후보키가 아님. 후보키가 아님에도 수강과목을 결정하는 결정자 역할을 하고 있는 것. 이와 같이 결정자가 후보키가 아닌 경우 분해하는 과정을 BCNF라고 함. 위 [등록] 테이블은 다음과 같이 분해할 수 있음.

회원등록(<u>회원번호</u>, 강사) 강사(<u>강사</u>, 수강과목)

- [회원등록] 테이블의 '강사' 속성을 외래키로 지정하여 [강사] 테이블로 참조함.

회원등록

회원번호	강사
10010	최수지
10010	김선수
20020	이영춘
20030	이태선
20030	최수지
30010	박길동
30010	김정미

강사

강사	수강과목
최수지	POP글씨
김선수	서예
이영춘	기타
이태선	네일아트
박길동	서예
김정미	POP글씨

- 이와 같이 [등록] 테이블을 [회원등록] 테이블과 [강사] 테이블로 분해하면 모든 결정자가 후보키가 되어 BCNF를 만족하게 됨.

⑤ 제4정규형(4NF : Fourth Normal Form)

• 제4정규형은 릴레이션에서 다치 종속(MVD : Multivalued Dependency) 관계가 성립되는 경우 분해하는 정규형

- 함수 종속은 'A → B'인 경우 A의 속성값은 B의 속성값 하나를 결정하게 됨.

강사

강사	수강과목
최수지	POP글씨
김선수	서예
이영춘	기타
이태선	네일아트

- [강사] 테이블에서 '강사 → 수강과목' 종속 관계가 있음. 따라서 '강사' 속성의 값은 '수강과목' 속성 하나의 값과 대응되어 '수강과목' 하나하나를 식별할 수 있는 것.
- 다치 종속(MVD)은 함수 종속과는 달리 하나의 속성값이 대응되는 속성의 집합을 결정하는 종속 관계를 말하며, 릴레이션의 속성이 3개 이상일 때 존재함.
- 다치 종속의 표기는 다음과 같음.

A ↠ B : A의 속성값은 B의 속성값의 집합을 결정하게 됨.

• 다음은 '과목명', '강사', '교재' 속성으로 구성된 [강좌] 테이블

과목명	강사	교재
POP글씨	최수지	POP-1
POP글씨	최수지	POP-2
POP글씨	김정미	POP-1
POP글씨	김정미	POP-2
서예	박길동	서예-1
서예	박길동	서예-2

- 위 [강좌] 테이블은 함수 종속 관계가 성립되지 않음. 그런데 [강좌] 테이블에서는 '과목명'을 알면 그 과목과 관련된 강사들이 누구인지 강사의 집합을 알 수 있으며, '과목명'을 알면 그 과목과 관련된 교재의 집합을 알 수 있음. 즉, POP글씨와 관련된 강사는 (최수지, 김정미)임. 또, POP글씨와 관련된 교재는 (POP-1, POP-2)임.
- 이와 같이 하나의 속성값과 여러 개의 속성값이 종속된 관계를 다치 종속(MVD)이라고 함. [강좌] 테이블은 다음과 같은 다치 종속(MVD)이 성립됨.

과목명 ↠ 강사 과목명 ↠ 교재

- 이와 같은 다치 종속(MVD) 관계의 테이블을 분해하는 것이 제4정규형임.
- [강좌] 테이블은 다음과 같이 분해할 수 있음.

강사(과목명, 강사) 교재(과목명, 교재)

강사

과목명	강사
POP글씨	최수지
POP글씨	김정미
서예	박길동

강사

과목명	교재
POP글씨	POP-1
POP글씨	POP-2
서예	서예-1
서예	서예-2

⑥ 제5정규형(5NF : Fifth Normal Form)

- 제5정규형은 릴레이션에 존재하는 조인 종속이 후보키를 통해서만 성립이 되도록 하는 정규형
- 조인 종속 - 원래의 릴레이션을 분해한 뒤 자연 조인한 결과가 원래의 릴레이션과 같은 결과가 나오는 종속성을 말함.

제1정규형	모든 도메인이 원자값이 되도록 분해
↓	
제2정규형	부분 함수 종속 관계 제거
↓	
제3정규형	이행적 함수 종속 관계 제거
↓	
BCNF	후보키가 아닌 결정자 관계 제거
↓	
제4정규형	다치 종속 관계 제거
↓	
제5정규형	후보키를 통하지 않은 조인 종속 관계 제거

< 트랜잭션(Transaction) >

1. 트랜잭션(Transaction)

- **데이터베이스 내에서** 한꺼번에 모두 수행되어야 할 연산들의 집합으로 하나의 작업 처리를 위한 **논리적 작업 단위**
- 트랜잭션 내의 연산은 한꺼번에 완료되어야 하며 그렇지 못한 경우 모두 취소되어야 함.

2. 트랜잭션의 성질

① 원자성(Automicity)

트랜잭션의 가장 기본적인 특성으로 트랜잭션 내의 연산은 반드시 **모두 수행**되어야 하며 그렇지 않은 경우 모두 수행되지 않아야 함.

② 일관성(Consistency)

트래잭션이 정상적으로 **완료된 후 언제나 일관성** 있는 데이터베이스 상태가 되어야 하며, 결과에 모순이 생겨서는 안 됨.

사용자가 물건을 구매한 후 10점의 포인트를 적립카드에 적립하여 100점이 되었다면 그 결과는 어디에서 점수를 조회하더라도 동일한 결과가 나와야 한다. 본사에서 포인트 점수를 조회하면 100점인데 일반 매장에서 조회해서 100점이 아니면 일관성이 결여된 것이다.

③ 격리성(Isolation)

하나의 트랜잭션이 수행 중에는 다른 트랜잭션이 접근할 수 없으며 **각각의 트랜잭션은 독립적**이어야 함. 따라서 독립성이라고도 함.

동일한 회사의 적립카드를 이용하는 사용자 A와 사용자 B가 동시에 적립카드에 적립하더라도 사용자 A에게 적립하는 처리과정과 B에게 적립하는 처리과정은 서로 구별되어 정확하게 처리되어야 한다.

④ 영속성(Durability)

트랜잭션이 성공적으로 완료된 후 결과는 지속적으로 유지되어야 함. 따라서 지속성이라고도 함.

사용자가 적립카드에 100점까지 적립했다면 그 결과는 이후 적립이 이루어지기 전까지 계속 유지되어야 한다.

3. 트랜잭션 연산

- 트랜잭션 연산에는 COMMIT과 ROLLBACK이 있으며 하나의 트랜잭션은 COMMIT이나 ROLLBACK이 되어야 함.

① COMMIT

- 트랜잭션이 성공적으로 종료된 후 수정된 내용을 지속적으로 유지하기 위한 연산
- 적립카드에 10점의 점수를 적립하는 트랜잭션을 수행해서 정상적으로 종료되는 경우는 다음과 같음.

read(card)	→ 카드 인식
A = A + 10	→ 포인트 입력
recognition	→ 승인
write(A)	→ 포인트 적립
COMMIT	→ 정상 종료

② ROLLBACK

- 트랜잭션이 비정상적으로 수행되었거나 오류가 발생했을 때 수행 작업을 취소하고 이전 상태로 되돌리기 위한 연산
- 적립카드에 10점의 점수를 적립하는 트랜잭션을 수행하는 도중 오류가 발생하는 경우는 다음과 같음.

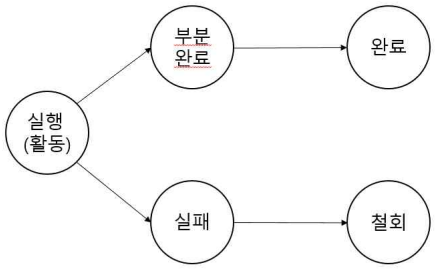
read(card)	→ 카드 인식	↰
A = A + 10	→ 포인트 입력	↑ 작업
오류		↑ 취소
ROLLBACK		

4. 트랜잭션의 상태도

- 트랜잭션이 수행되는 과정은 다음과 같은 상태로 구분할 수 있음.

실행	현재 실행 중인 상태
부분완료	실행을 모두 마치고, 데이터베이스에 결과를 저장하기 직전 상태
완료	트랜잭션의 연산을 정상적으로 마치고, 연산 결과를 데이터베이스에 저장한 상태
실패	트랜잭션 실행 중 오류에 의해 더 이상 진행될 수 없는 상태
철회	트랜잭션 실행이 실패되어 복귀되는 상태

- 부분완료는 실행은 마쳤지만 아직 데이터베이스에 저장이 이루어지지 않은 상태. 따라서 부분완료가 되었다하더라도 실패로 이어질 수 있음.
- 예를 들어 문서를 다 작성했더라도 저장하지 않으면 작성한 문서가 손실이 될 수도 있듯이 트랜잭션이 정상적으로 완료되었다 하더라도 COMMIT 연산을 수행하지 않으면 수행된 결과가 유지될 수 없음.
- 트랜잭션이 비정상적으로 수행되거나 오류가 발생하면 ROLLBACK 연산에 의해 취소됨. 이와 같이 하나의 트랜잭션은 반드시 COMMIT이나 ROLLBACK 되어야 함.



[트랜잭션의 상태도]

< 회복 기법 >

1. 회복(Recovery)
- 여러 가지 요인으로 인해 손상된 데이터베이스를 손상되기 이전의 **정상적인 상태로 복구**시키는 작업
- 회복을 위해 로그(Log)를 이용하게 되는데, 로그(Log)란 트랜잭션이 수행되어 변경되는 데이터베이스의 상황 정보를 기록하는 것으로, 트랜잭션이 수행되기 이전 값과 수행된 이후 값 모두 기록됨. 이와 같은 정보를 담고 있는 파일을 로그 파일(Log File)이라고 함.

① 회복 기법
- **즉시 갱신 기법** : 트랜잭션이 실행(활동) 상태에서 변경되는 내용을 **그때그때 바로 데이터베이스에 적용**하는 기법. 변경되는 모든 내용은 로그(Log)에 기록하여 장애 발생 시 로그(Log)의 내용을 토대로 회복시킴.
- **지연 갱신 기법** : 트랜잭션이 수행되어 부분완료 될 때까지 데이터베이스에 적용하지 않고 **지연시킨 후 부분완료가 되면** 로그(Log)의 내용을 토대로 **데이터베이스에 적용**하는 기법.
- **검사 시점 기법** : 트랜잭션이 실행되는 중간에 **검사 시점(Check Point)을 지정**하여 검사 시점까지 수행 후 완료된 내용을 데이터베이스에 적용하는 기법.
- **그림자 페이징(Shadow Paging) 기법** : 로그(Log)를 사용하지 않고, 데이터베이스를 동일한 크기의 단위인 페이지로 나누어 각 페이지마다 복사하여 그림자 페이지를 보관함. 데이터베이스의 변경되는 내용은 원본 페이지에만 적용하고, 장애가 발생되는 경우 그림자 페이지를 이용해 회복함.

② REDO(재수행)와 UNDO(취소)
- **REDO** : 트랜잭션이 수행되어 COMMIT이 되면 변경된 내용을 데이터베이스에 반영하게 됨. 이때 로그(Log)의 내용을 토대로 재수행하며 변경된 내용을 데이터베이스에 반영하게 됨. 이와 같이 재수행하는 과정.
- **UNDO** : 트랜잭션이 수행되는 도중 오류가 발생하거나 비정상적으로 종료되는 경우 트랜잭션이 시작된 시점으로 되돌아가는 과정.

데이터베이스 수행 시 발생되는 장애의 유형
- 트랜잭션 장애 : 하나의 트랜잭션이 수행되는 과정에서 발생하는 오류
- 시스템 장애 : 트랜잭션 장애들로 인해 시스템 상의 문제가 발생하여 트랜잭션이 수행되지 못하는 경우를 말함. 예를 들어 컴퓨터를 사용하는 도중에 컴퓨터가 다운되는 경우가 여기에 속함.
- 미디어 장애 : 하드웨어적으로 하드디스크 등이 손상되는 경우를 말함.

< 객체 지향 데이터베이스(OODB) >

1. 객체 지향 데이터베이스의 등장 배경
- 기존의 관계형 데이터베이스는 문자나 숫자, 날짜, 시간 등과 같은 단순한 형태의 데이터를 저장하는 형태. 그러나 현재는 컴퓨터 기술과 통신이 발달하면서 데이터의 양이나 내용이 과거와 달라지고 있음. 현재는 단순한 형태의 데이터를 포함하여 이미지, 영상, 소리 등 다양한 멀티미디어 정보를 이용하고 있음. 따라서 이와 같은 멀티미디어 정보를 저장 및 관리하고 이용할 수 있도록 등장하게 된 개념이 객체 지향 데이터베이스(OODB : Object-Oriented DataBase)
- 이와 같은 객체 지향 데이터베이스(OODB)를 관리하고 운영하는 데이터베이스 관리 시스템을 객체 지향 데이터베이스 관리시스템(OODBMS : Object-Oriented DataBase Management System)
- 또한 최근에는 기존의 관계 데이터베이스 모델의 한계를 극복하기 위해 새롭게 등장한 객체 지향 데이터베이스(OODB)보다 기존의 관계 데이터베이스와 객체 지향 데이터베이스를 접목한 객체-관계 데이터베이스(ORDB : Object-Relation DataBase)가 등장하게 되었으며, 이를 관리하고 운영하기 위한 시스템을 객체-관계 데이터베이스 관리시스템(ORDBMS : Object-Relation DataBase Management System)

2. 객체(Object)와 객체 지향 기법의 특징

① 객체(Object)

- 유형이나 무형으로 현실 세계에 존재하는 하나하나를 추상화한 것으로 서로 구별되는 개념적인 단위
- 관계 데이터베이스의 개체(Entity)와 유사하나 개체(Entity)의 개념과 자체적으로 처리 기능을 갖는 연산자까지 포함된 하나의 단위 시스템

② 속성(Attribute)

객체의 특성이나 상태를 나타내며, 관계 데이터베이스의 속성과 유사한 개념

③ 메시지(Message)와 메소드(Method)

- 메시지 – 객체에 어떤 처리를 하도록 지시하는 명령
- 메소드 – 메시지에 따라 객체가 실행해야 할 검색, 삽입, 삭제, 변경 등과 같은 구체적인 연산

> '학번', '이름', '점수' 속성으로 구성되어 있는 '학생' 객체에서 "A학생의 점수를 변경하여라"라고 한다면 이와 같은 명령은 메시지이며, 이 명령에 따라 점수를 변경하는 실제 연산을 메소드라고 함.

④ 클래스(Class)

- 유사한 성격과 공통적인 특성을 갖는 객체들의 모임
- 한 클래스 내의 객체들은 유사한 구조를 갖음.
  - 데스크탑과 노트북은 각각 하나의 객체임. 이 두 객체는 하드디스크와 RAM/ROM, 메인보드, 비디오카드, 사운드카드를 가지고 있음. 이와 같이 유사한 성격과 공통적인 특징을 갖는 객체들을 컴퓨터라는 클래스로 만들 수 있음.

⑤ 캡슐화(Encapsulation)

하나의 객체가 문제 해결을 위해 필요한 데이터, 연산, 상수 등의 정보를 하나로 묶음으로써 다른 객체와 정보은폐(정보은닉)가 이루어지도록 하는 것.

> 약의 종류 중에 캡슐에 싸여 있는 약이 있다. 그 약은 어떤 병을 치료하는데 필요한 성분들로만 구성되어 있다. 이와 같이 캡슐화함으로써 다른 성분들과 섞여 치료 효과를 저해하는 것을 방지할 수 있다.

⑥ 상속(계승 : Inheritance)

- 객체 지향 기법의 대표적인 특징으로 클래스의 계층구조에서 상위 클래스의 특징과 정보 등을 하위 클래스에서 그대로 재사용할 수 있는 개념. 따라서 하나의 클래스를 만들 때 상위 클래스의 내용을 재사용함으로써 보다 효율적인 작업이 이루어짐.
- 하나의 클래스로부터 상속받는 것을 단일 상속이라 하며, 여러 개의 클래스로부터 상속받는 것을 다중 상속이라 함.

⑦ 다형성(Polymorphism)

동일한 객체더라도 경우에 따라 다른 의미의 연산으로 사용될 수 있는 개념을 나타내는 것으로 객체 지향 기법에서는 이와 같은 다형성의 특징을 가지고 있음.

< 기타 데이터베이스 용어 >

1. 개체(Entity)의 종류

① 독립 개체

데이터베이스 내에서 다른 개체에 종속되지 않고, 그 개체 내에서 모든 검색과 변경 등이 가능한 개체

② 종속 개체

• 데이터베이스의 그 개체 내에서 원하는 검색 등의 연산을 하지 못하고 다른 개체를 참조해야 하는 개체를 말하는 것으로, 다른 개체에 종속되는 개체

교수

번호	교수 이름	학과 번호	직급
1001	오준석	A1	주임
1002	이순신	A2	부주임
1003	홍길동	B1	교수

학과

학과 번호	학과 이름	교수 번호	학생수
A1	컴퓨터	1001	30
A2	정보통신	1002	20
B1	토목	1003	50

- 위 [교수] 테이블에서 '오준석' 교수가 담당하는 학과의 이름이나 학생수는 검색할 수 없고, [교수] 테이블의 '학과번호'를 외래키로 지정해서 [학과] 테이블을 참조해야 원하는 내용을 검색할 수 있음.
- 이와 같은 경우 [교수] 테이블은 '종속 개체'에 해당됨. 반면 [학과] 테이블에서는 자체적으로 원하는 내용을 모두 검색할 수 있으므로 '독립 개체'가 됨.

2. 속성(Attribute)의 종류

① 단순 속성(Simple Attribute)

• 속성의 값을 더 이상 작은 단위로 나눌 수 없는 속성
• 위 [학과] 테이블에서 '학과번호', '학과이름', '학생수' 등은 속성의 값을 더 이상 작은 단위로 나눌 수 없음. 따라서 이와 같은 속성을 단순 속성

② 복합 속성(Composite Attribute)

• 속성의 값을 여러 개의 작은 단위로 나눌 수 있는 속성
• 위 [교수] 테이블이나 [학과] 테이블에서 '교수이름'은 세부적으로 '성'과 '이름'으로 나눌 수 있음. 이와 같이 필요에 따라 속성값을 작은 단위로 나눌 수 있는 속성을 복합 속성

③ 결합 속성(Concatenate Attribute)

• 두 개 이상의 속성값을 합쳐 하나의 속성으로 구성된 속성
• '출생년도'와 '출생월일'로 되어 있는 속성을 합쳐서 '생년월일'로 나타냈다면 '생년월일'은 결합 속성

출생년도	출생월일	→	생년월일

3. 분산 데이터베이스(Distributed DataBase)

• 정보의 양이 급증하고 정보를 사용하는 사용자도 증가함에 따라 처리의 효율과 신속한 서비스를 제공하기 위해 통신 네트워크를 통해 여러 대의 컴퓨터에 데이터를 분산시켜 저장하고 관리하여 사용자의 정보 요청 시 각각 컴퓨터에서 직접 처리 및 제공하도록 구성된 데이터베이스

① 분산 형태

• 수평 분산 : 다량의 정보를 여러 개의 동등한 기능을 가진 컴퓨터(서버)에 저장시켜 운영하는 방식으로, 각각의 서버는 서로 공유할 수 있으며 어느 하나의 서버에 문제가 발생하더라도 데이터베이스 운영에 지장을 주지 않는 형태
• 수직 분산 : 전체를 운영하는 주 서버와 처리를 담당하는 여러 대의 부 서버로 구성하여 운영하는 방식으로, 주 서버의 운영에 따라 관리되므로 관리가 용이하나 주 서버에 장애가 발생할 경우 전체 운영에 지장을 주게 됨.

② 분산 데이터베이스의 장단점

장점	• 자체적인 처리 능력으로 신속한 서비스가 제공됨 • 확장성이 용이함 • 신뢰성과 가용성이 증진됨 • 효율성과 융통성이 있음
단점	• 구축하기 복잡함 • 오류가 증가함 • 구축 및 운영비용이 증가됨

4. 튜닝(Tuning)

- 데이터베이스의 성능 향상과 사용자의 요구에 따라 빠른 검색을 통한 신속한 서비스 제공, 저장 공간의 효율을 향상시키는 등 데이터베이스시스템을 최적화하기 위해 재조정(조율)하는 것.
- 데이터 검색 시 자료가 저장된 블록의 이동과 접근 횟수를 줄일 수 있도록 저장 공간을 조정하여 신속한 검색이 이루어지도록 함.
- SQL 명령어 작성 시 쉽게 이해할 수 있도록 표준화된 형태로 작성함.
- 트랜잭션의 무결성을 유지하면서 정보 공유를 위해 적정한 수준의 Locking 기법을 사용.

5. 트리거(Trigger)

- 참조 관계에 있는 두 테이블에서 하나의 테이블에 삽입, 삭제, 갱신 등의 연산으로 테이블의 내용이 바뀌었을 때 데이터의 일관성과 무결성 유지를 위해 이와 연관된 테이블도 연쇄적으로 변경이 이루어질 수 있도록 하는 것

교수

교수번호	교수이름	학과번호
1001	오준석	A1
1002	이순신	A2
1003	홍길동	B1

학과

학과번호	학과이름	학생수	교수이름
1001	컴퓨터	30	오준석
1002	정보통신	20	이순신
1003	토목	50	홍길동

- 위 두 테이블에서 [교수] 테이블의 '학과번호'를 외래키로 지정하여 [학과] 테이블을 참조
- 이때 [학과] 테이블에 새로운 학과가 신설되어 학과 정보가 삽입되거나 또는 반대로 삭제되거나 변경되는 경우 이와 연관된 [교수] 테이블의 정보도 연쇄적으로 같이 삽입, 삭제, 변경이 이루어져야 데이터의 일관성과 무결성을 유지할 수 있음.