


sqld 시험 정리

<input checked="" type="checkbox"/> 체크 표시됨	<input type="checkbox"/>
 태그	

1과목

모델링의 개념

- 현실 세계의 비즈니스 프로세스와 데이터 요구 사항을 추상적이고 구조화된 형태로 표현하는 과정
- 데이터베이스의 구조와 관계를 정의하며, 이를 통해 데이터의 저장, 조작, 관리 방법을 명확하게 정의

모델링의 특징

1. 단순화

- 현실을 단순화하여 핵심 요소에 집중하고 불필요한 세부 사항을 제거
- 단순화를 통해 복잡한 현실 세계를 이해하고 표현하기 쉬워짐

2. 추상화

- 현실세계를 일정한 형식에 맞추어 간략하게 대략적으로 표현하는 과정
- 다양한 현상을 일정한 양식인 표기법에 따라 표현

3. 명확화

- 대상에 대한 애매모호함을 최대한 제거하고 정확하게 현상을 기술하는 과정
- 명확화를 통해 모델을 이해하는 이들의 의사소통을 원활히 함

데이터 모델링 유의점

1. 중복

- 한 테이블 또는 여러 테이블에 같은 정보를 저장하지 않도록 설계

2. 비유연성

- 사소한 업무 변화에 대해서도 잦은 모델 변경이 되지 않도록 주의
- 데이터 정의를 프로세스와 분리

3. 비일관성

- 데이터베이스 내의 정보가 모순되거나 상반된 내용을 갖는 상태를 의미
- 데이터간 상호연관 관계를 명확히 정의
- 데이터 품질 관리 필요
- 데이터의 중복이 없더라도 비일관성은 발생할 수 있음

데이터 모델링 3가지 요소

- 대상: 업무가 관리하고자 하는 대상
- 속성: 대상들이 갖는 속성
- 관계: 대상들 간의 관계

데이터 모델링의 3 단계

1. 개념적 모델링

- 업무 중심적이고 포괄적인 수준의 모델링
- 추상화 수준이 가장 높음
- 업무를 분석 뒤 업무의 핵심 엔터티를 추출하는 단계
- 도출된 핵심 엔터티들과의 관계들을 표현하기 위해 ERD 작성

2. 논리적 모델링

- 개념적 모델링의 결과를 토대로 세부속성, 식별자, 관계 등을 표현하는 단계
- 데이터 구조를 정의하기 때문에 비슷한 업무나 프로젝트에서 동일한 형태의 데이터 사용 시 재사용 가능
- 동일한 논리적 모델을 사용하는 경우 쿼리도 재사용 가능
- 데이터 정규화 수행
- 재사용성이 높은 논리적 모델은 유지보수가 용이해짐

3. 물리적 모델링

- 논리 모델링이 끝나면 이를 직접 물리적으로 생성하는 과정
- 데이터베이스 성능, 디스크 저장구조, 하드웨어의 보안성, 가용성 등을 고려
- 가장 구체적인 데이터 모델링
- 추상화 수준은 가장 낮음(가장 구체적인 모델링이므로)

데이터 모델의 표기법(ERD: Entity Relationship Diagram)

- 엔터티와 엔터티 간의 관계를 시각적으로 표현한 다이어그램

- 1976년 피터 첸이 만든 표기법, 데이터 모델링 표준으로 사용

ERD 작성 절차(6단계)

1. 엔터티를 도출한 후 그린다
2. 엔터티 배치
3. 엔터티 간의 관계를 설정
4. 관계명을 서술
5. 관계의 참여도 기술
6. 관계의 필수 여부를 확인

스키마의 3 단계 구조

- **스키마: 데이터베이스의 구조와 제약 조건에 관한 전반적인 명세를 기술한 메타데이터의 집합**

1. 외부 스키마

- 사용자가 보는 관점에서 데이터베이스 스키마를 정의
- 사용자나 응용 프로그램이 필요한 데이터를 정의(View: 사용자가 접근하는 대상)

2. 개념 스키마

- 사용자 관점의 데이터베이스 스키마를 통합하여 데이터베이스의 전체 논리적 구조를 정의
- 전체 데이터베이스의 개체, 속성, 관계, 데이터 타입 등을 정의

3. 내부 스키마

- 데이터가 물리적으로 어떻게 저장되는지를 정의
- 데이터의 저장 구조, 컬럼, 인덱스 등을 정의함

3 단계 스키마의 독립성

- **독립성: 물리적, 논리적 구조를 변경하더라도 사용자가 사용하는 응용 프로그램에 영향을 주지 말아야 함**
- 1) 논리적 독립성: 논리적 데이터 구조가 변경되어도(개념 스키마 변경) 응용 프로그램에 영향을 주지 않는 특성
- 2) 물리적 독립성: 물리적 구조가 변경되어도(내부 스키마 변경) 개념/외부 스키마에 영향을 주지 않는 특성

엔터티(Entity)의 개념

- 현실 세계에서 독립적으로 식별 가능한 객체나 사물을 나타냄
- 엔터티는 업무상 분석해야 하는 대상으로 이루어진 집합
- 인스턴스는 엔터티의 특정한 속성 값들로 구성되며, 엔터티의 개념을 현실에서 구체적으로 나타낸 것

ex) 엔터티와 속성, 인스턴스 등의 관계

- **엔터티(Entity):** 학생
- **속성(Attribute):** 학번, 이름, 학과 등.
- **식별자(Identifier):** 학번 (고유한 학번으로 각 학생을 식별)
- **인스턴스:** 특정 학생의 데이터

-학번:2021001 / 이름: 홍길동 / 학과: 컴퓨터 공학

엔터티(Entity)의 특징

1. 유일한 식별자에 의해 식별 가능
 - 인스턴스가 식별자에 의해 한 개씩만 존재하는 지 검증 필요
 - 유일한 식별자는 그 엔터티의 인스턴스만의 고유 이름
 - ex) 이름은 동명이인이 있을 수 있으므로 사번, 학번 등이 고유식별자
2. 해당 업무에 필요하고 관리하고자 하는 정보
 - 설계하는 업무의 시스템 구축에 필요한 정보여야 함

ex) 학교 시스템 구축 시 학생 정보 필요, 다른 업무엔 학생 정보 불필요.

3. 인스턴스들의 집합
 - 영속적으로 존재하는 2 개 이상의 인스턴스의 집합
 - 인스턴스가 한 개 밖에 없는 엔터티는 집합이 아니므로 성립이 안됨.
4. 엔터티는 반드시 속성을 가짐
 - 각 엔터티는 2 개 이상의 속성을 가짐
 - 하나의 인스턴스는 각각의 속성들에 대한 1 개의 속성 값만을 가짐

ex) 학생 엔터티에서 한 학생의 데이터(인스턴스)의 이름(속성) 정보에는 반드시 한 값만 저장됨

5. 엔터티는 업무 프로세스에 의해 이용
 - 업무적으로 필요해 선정했거나 실제 사용되지 않으면 잘못 설계된 것

- 모델링 시 발견하기 어려운 경우 데이터 모델 검증이나 상관 모델링 시 단위 프로세스 교차점검으로 문제 도출
 - 누락된 프로세스의 경우 추후 해당 프로세스 추가
 - 반대로 사용되지 않는 고립 엔터티는 제거 필요
6. 다른 엔터티와 최소 1 개 이상의 관계 성립
- 엔터티는 업무적 연관성을 갖고 다른 엔터티와 연관의 의미를 가짐
 - 관계가 없는 엔터티 도출은 부적절한 엔터티이거나 적절한 관계를 찾지 못한 것

엔터티의 분류

1) 유형과 무형에 따른 분류

1. 유형엔터티

- 물리적 형태가 있음(실체가 있는 대상)
- 안정적이며 지속적으로 활용되는 엔터티
- 업무로부터 구분하기가 가장 용이한 엔터티

ex) 사원, 물품, 감사 등

2. 개념엔터티

- 물리적인 형태 없음
- 관리해야 할 개념적 정보로부터 구분되는 엔터티

ex) 조직, 보험상품 등

3. 사건엔터티

- 업무를 수행에 따라 발생하는 엔터티
- 발생량이 많고 각종 통계자료에 이용

ex) 주문, 청구, 미납 등

2) 발생 시점에 따른 분류

1. 기본엔터티

- 그 업무에 원래 존재하는 정보
- 다른 엔터티와 관계에 의해 생성되지 않고 독립적으로 생성
- 타 엔터티의 부모 역할을 하는 엔터티
- 다른 엔터티로부터 주식별자를 상속받지 않고 자신의 고유한 주식별자를 가짐

ex) 사원, 부서, 고객, 상품 등

2. 중심엔터티

- 기본엔터티로부터 발생되고 그 업무에서 중심적인 역할
- 많은 데이터가 발생되고 다른 엔터티와의 관계를 통해 많은 행위 엔터티를 생성

ex) 계약, 사고, 청구, 주문, 매출 등

3. 행위엔터티

- 2개 이상의 부모엔터티로부터 발생
- 자주 내용이 바뀌거나 데이터 양이 증가
- 분석 초기 단계보다는 상세 설계 단계나 프로세스와 상관모델링을 진행하면서 도출

ex) 주문(고객과 상품 엔터티로부터 발생하므로 행위엔터티이기도 함), 사원변경이력, 이력 등

엔터티의 명명

1. 현업에서 사용하는 용어 사용
2. 가능하면 약자 사용은 자제
3. 단수 명사 사용
4. 모든 엔터티에서 유일하게 이름 부여
5. 엔터티 생성 의미대로 이름 부여

엔터티와 인스턴스 표기법

- 엔터티는 사각형으로 표현, 속성은 조금씩 다름

속성의 개념

-속성은 업무에서 필요로 하는 고유한 성질, 특징을 의미(관찰 대상) → 컬럼으로 표현할 수 있는 단위!

-업무상 인스턴스로 관리하고자 하는 더 이상 분리되지 않는 최소의 데이터 단위

-인스턴스의 구성 요소

ex) 학생 엔터티에 이름, 학번, 학과번호 등이 속성이 될 수 있음

엔터티, 인스턴스, 속성, 속성값의 관계

- 한 개의 엔터티는 2개 이상의 인스턴스의 집합이어야 한다(하나의 테이블은 두 개 이상의 행을 가짐)

- 한 개의 엔터티는 2개 이상의 속성을 갖는다(하나의 테이블은 두 개 이상의 컬럼으로 구성됨)
- 한 개의 속성은 1개의 속성 값을 갖는다(각 컬럼의 값은 하나씩만 삽입 가능)
- 속성은 엔터티에 속한 엔터티에 대한 자세하고 구체적인 정보를 나타냄. **각 속성은 구체적인 값을 가짐**

속성의 특징

- 반드시 해당 업무에서 필요하고 관리하고자 하는 정보여야 한다.
- 정해진 주식별자에 함수적 종속성을 가져야 한다.
- 하나의 속성은 한 개의 값만을 가진다.
- 하나의 속성에 여러 개의 값이 있는 다중 값일 경우 별도의 엔터티를 이용하여 분리한다.
- 하나의 인스턴스는 속성마다 반드시 하나의 속성 값을 가진다.

⇒ 각 속성이 하나의 값을 갖고 있음을 의미(속성의 원자성)

원자성이란

- 데이터모델에서 각 엔터티의 인스턴스가 해당 속성에 대해 단일하고 명확한 값을 가지는 것을 의미

함수적 종속성

- 한 속성의 값이 다른 속성의 값에 종속적인 관계를 갖는 특징을 말함
- 즉, 어떤 속성 A 의 값에 의해 다른 속성 B도 유일하게 결정된다면, B는 A에 함수적으로 종속됐다 하고,
- 이를 수식으로 나타내면 $A \rightarrow B$ 라고 표현함

1) 완전 함수적 종속

- 특정 컬럼이 기본키에 대해 완전히 종속될 때를 말함
- PK를 구성하는 컬럼이 2개 이상일 경우 PK 값 모두에 의한 종속관계를 나타낼 때 완전 함수 종속성 만족

ex) (주문번호 + 제품번호) 에 의해 수량 컬럼의 값이 결정됨

2) 부분 함수적 종속

- 기본키 전체가 아니라, 기본키 일부에 대해 종속될 때를 말함

ex) 수강기록 테이블에서 학생번호와 과목이 PK라고 가정할 때, 과목에 의해서도 교수가 결정되면 부분 함수적 종속 관계!

- **속성의 분류**

1) 속성의 특성에 따른 분류

1. 기본 속성

- 업무로부터 추출된 모든 속성
- 엔터티에 가장 일반적으로 많이 존재하는 속성

ex) 원금, 예치기간 등

2. 설계 속성

- 기본 속성 외에 업무를 규칙화하기 위해 새로 만들어지거나 기본 속성을 변형하여 만들어지는 속성

ex) 상품코드, 지점코드, 예금분류 등

3. 파생 속성

- 다른 속성에 의해 만들어지는 속성
- 일반적으로 계산된 값들이 해당
- 데이터 정합성을 유지하기 위해 가급적 적게 정의하는 것이 좋은

ex) 합계, 평균, 이자 등

2) 엔터티 구성방식에 따른 분류

1. PK(primary Key, 기본키)

- 인스턴스를 식별할 수 있는 속성

2. FK(Foreign Key, 외래키)

- 다른 엔터티와의 관계에서 포함된 속성

3. 일반 속성

- 엔터티에 포함되어 있고 PK/FK에 포함되지 않는 속성

3) 분해 여부에 따른 속성

1. 단일 속성

- 하나의 의미로 구성된 경우

ex) 회원 ID, 이름 등

2. 복합 속성

- 여러개의 의미로 구성된 경우

ex) 주소(시, 구, 동 등으로 분해 가능) 등

3. 다중값 속성

- 속성에 여러 개의 값을 가질 수 있는 경우
- 다중값 속성은 엔터티로 분해

ex) 상품 리스트 등

속성의 명명규칙

1. 해당 업무에서 사용하는 이름을 부여
2. 서술식 속성명은 사용하지 않음
3. 약어의 사용은 가급적 제한
4. 전체 데이터 모델에서 유일한 명칭

도메인(Domain)

- 도메인은 각 속성이 가질 수 있는 값의 범위를 의미함
- 엔터티 내에서 속성에 대한 데이터 타입과 크기, 제약사항을 지정하는 것이다

관계(Relationship)의 개념

- 관계는 엔터티간의 연관성을 나타낸 개념
- 관계를 정의할 때는 인스턴스(각 행 데이터)간의 논리적인 연관성을 파악하여 정의
- 엔터티를 어떻게 정의하느냐에 따라 변경되기도 함

관계의 종류

1) 존재적 관계

- 한 엔터티의 존재가 다른 엔터티의 존재에 영향을 미치는 관계
- 엔터티 간의 연관된 상태를 의미

ex) 부서 엔터티가 삭제되면 사원 엔터티의 존재에 영향을 미침

2) 행위적 관계

- 엔터티 간의 어떤 행위가 있는 것을 의미

ex) 고객 엔터티의 행동에 의해 주문 엔터티가 발생

***ERD에서는 존재관계와 행위관계를 구분하지 않는다.**

관계의 구성

1. 관계명
2. 차수(Cardinality)
3. 선택성(Optionality)

관계의 차수(Cardinality)

- 한 엔터티의 레코드(인스턴스)가 다른 엔터티의 레코드(인스턴스)와 어떻게 연결되는지를 나타내는 표현
- 주로 1:1, N:N, N:M 등으로 표현

1) 1대1 관계

- **완전 1대1 관계**
- 하나의 엔터티에 관계되는 엔터티가 반드시 하나로 존재하는 경우

ex) 사원은 반드시 소속 부서가 있어야 함

- **선택적 1대1 관계**
- 하나의 엔터티에 관계되는 엔터티가 하나이거나 없을 수 있는 경우

ex) 사원은 하나의 소속 부서가 있거나 아직 발령전이면 없을 수 있음

2) 1대 N 관계

- 엔터티에 하나의 행에 다른 엔터티의 값이 여러 개 있는 관계

ex) 고객은 여러 개 계좌를 소유할 수 있음.

3) M 대 N 관계

- 두 엔터티가 다대다의 연결 관계 가지고 있음
- 이 경우 조인 시 카테시안 곱이 발생하므로 두 엔터티를 연결하는 연결엔터티의 추가로 1대N 관계로 해소할 필요가 있음

ex) 한 학생이 여러 강의를 수강할 수 있고, 한 강의 기준으로 여러 학생이 보유할 수 있음

⇒ 이 두 엔터티의 연결엔터티로는 구매이력 엔터티가 필요함

관계의 페어링

- 엔터티 안에 인스턴스가 개별적으로 관계를 가지는 것
- 관계란 페어링의 집합을 의미함

관계와 차수, 페어링 차이

- 학생과 강의 엔터티는 관계를 가짐
- 한 학생은 여러 강의를 수강할 수 있고, 한 강의도 여러 학생에게 수강될 수 있으므로 M 대 N 관계이며, 이 때 차수는 M:N 가 됨
- 인스턴스의 관계를 보면 "학생 A가 강의 B를 2023년 1학기에 수강했고 성적은 'A+'를 받았다"와 같은 특정한 페어링이 형성
- 이런식으로 관계의 차수는 하나의 엔터티와 다른 엔터티 간의 레코드 연결 방식을 나타내는 반면, 관계 페어링은 두 엔터티 간의 특정 연결을 설명하고 추가 정보를 제공하는 용도로 사용

식별자 개념

- 하나의 엔터티에 구성된 여러 개의 속성 중에 엔터티를 대표할 수 있는 속성을 나타냄
- 하나의 유일한 식별자가 존재해야 함
- 식별자는 논리 모델링에서 사용하는 용어, 물리 모델링에서는 키(key)라고 표현
- ex) 학생 엔터티의 주식별자는 학생번호 속성 → 학생 테이블의 기본키는 학생번호 컬럼

주식별자 특징

1. 유일성: 주식별자에 의해 모든 인스턴스를 유일하게 구분함

ex) 학생 엔터티에서 이름 속성은 동명이인이 발생할 수 있으므로 모든 인스턴스를 완벽하게 구분

2. 최소성: 주식별자를 구성하는 속성은 유일성을 만족하는 최소한의 속성으로 구성
3. 불변성: 주식별자가 한번 특정 엔터티에 지정되면 그 식별자의 값은 변하지 않아야 함
4. 존재성: 주식별자가 지정되면 반드시 값이 존재해야 하며 NULL은 허용 안 됨

식별자 분류

1) 대표성 여부에 따른 식별자의 종류

주식별자

- 유일성과 최소성을 만족하면서 엔터티를 대표하는 식별자
- 엔터티 내에서 각 인스턴스를 유일하게 구분할 수 있는 식별자
- 타 엔터티와 참조관계를 연결할 수 있는 식별자

보조식별자

- 엔터티 내에서 각 인스턴스를 구분할 수 있는 구분자지만, 대표성을 가지지 못해 참조 관계 연결을 할 수 없는 식별자

- 유일성과 최소성은 만족하지만 대표성을 만족하지 못하는 식별자

2) 생성 여부에 따른 식별자의 종류

내부식별자

- 다른 엔터티 참조 없이 엔터티 내부에서 스스로 생성되는 식별자

외부식별자

- 다른 엔터티와 관계로 인하여 만들어지는 식별자(외래키)

3) 속성 수에 따른 식별자 종류

단일식별자

- 하나의 속성으로 구성

복합식별자

- 두개 이상의 속성으로 구성

4) 대체 여부에 따른 식별자의 종류

본질식별자(원조식별자)

- 비즈니스 프로세스에서 만들어지는 식별자

인조식별자

- 인위적으로 만들어지는 식별자
- 자동 증가하는 일련번호 같은 형태

주식별자 도출기준

1) 해당 업무에서 자주 이용되는 속성을 주식별자로 지정한다.

- 같은 식별자 조건을 만족하더라도 업무적으로 더 많이 사용되는 속성을 주식별자로 지정

2) 명칭이나 내역등과 같은 이름은 피함

- 이름 자체를 주식별자로 사용하는 행위를 피함

3) 속성의 수를 최대한 적게 구성

- 주식별자를 너무 많은 속성으로 구성 시, 조인으로 인한 성능저하 발생 우려
- 일반적으로 7~8개 이상의 주식별자 구성은 새로운 인조식별자를 생성하여 모델을 단순화 시키는 것이 좋음

관계간 엔터티 구분

1) 강한 개체

- 독립적으로 존재할 수 있는 엔터티
- ex) 고객과 계좌 엔터티 중, 고객은 독립적으로 존재할 수 있음

2) 약한 개체

- 독립적으로 존재할 수 없는 엔터티
- ex) 고객과 계좌 엔터티 중, 계좌는 독립적으로 존재할 수 없음(고객으로 파생되는 엔터티)

식별 관계와 비식별관계

1) 식별관계 (Identification Relationship)

- 하나의 엔터티의 기본키를 다른 엔터티가 기본키의 하나로 공유하는 관계
- 식별관계는 ERD에서 실선으로 표시
- ex) 사원과 교육이력 엔터티에서 양쪽 모두 기본키 중 일부가 사원번호임

2) 비식별관계 (Non-Identification Relationship)

- 강한 개체의 기본키를 다른 엔터티의 기본키가 아닌 일반 속성으로 관계를 가지는 것
- 비식별관계는 ERD에서 점선으로 표시
- ex) 부서와 사원의 관계에서 부서의 부서번호(기본키)를 사원 엔터티에서는 일반키로 가짐(사원에서는 사원번호가 기본키)

Key의 종류

- 논리 모델링에서의 식별자가 물리 모델링에서는 Key가 되는데 이를 Key의 특징에 따라 다음과 같이 분류

1. 기본키 - 엔터티를 대표할 수 있는 키
2. 후보키 - 유일성과 최소성을 만족하는 키 / 결국 후보키들 중 하나가 기본키가 되고, 나머지를 대체키라고 부름
3. 슈퍼키 - 유일성은 만족하지만 최소성은 만족하지 않는 키
4. 대체키 - 여러 후보키 중 기본키가 아닌 키
5. 외래키 - 다른 테이블의 기본키를 참조하는 키 / 참조 테이블은 하나 또는 여러 개 가능

정규화

모델링 시 최대한 중복 데이터를 허용하지 않아야 저장공간의 효율적 사용과 업무 프로세스의 성능을 기대할 수 있다. 이러한 중복 데이터를 허용하지 않는 방식으로 테이블을 설계하

는 방식을 정규화라고 한다.

- 최소한의 데이터만을 하나의 엔터티에 넣는식으로 데이터를 분해하는 과정
- 데이터의 일관성, 최소한의 데이터 중복, 최대한의 데이터 유연성 위한 과정
- 데이터의 중복을 제거하고 데이터 모델의 독립성을 확보
- 데이터 이상현상을 줄이기 위한 데이터 베이스 설계 기법
- 엔터티를 상세화는 과정으로 논리 데이터 모델링 수행 시점에서 고려

이상현상(Abnormality)

- 정규화를 하지 않아 발생하는 현상(삽입이상, 갱신이상, 삭제이상)

정규화 단계

1. 제 1 정규화(1NF)

- 테이블이 컬럼이 원자성(한 속성이 하나의 값을 갖는 특성)을 갖도록 테이블을 분해하는 단계
- 쉽게 말해 하나의 행과 컬럼의 값이 반드시 한 값만 입력되도록 행을 분리하는 단계

2. 제 2 정규화(2NF)

- 제 1 정규화를 진행한 테이블에 대해 완전 함수 종속을 만들도록 테이블을 분해
- 완전 함수 종속이란, 기본키를 구성하는 모든 컬럼의 값이 다른 컬럼을 결정짓는 상태
- 기본키의 부분 집합이 다른 컬럼과 1:1 대응 관계를 갖지 않는 상태를 의미
- 즉, PK가 2개 이상일 때 발생하며 PK의 일부와 종속되는 관계가 있다면 분리

3. 제 3 정규화(3NF)

- 제 2 정규화를 진행한 테이블에 대해 이행적 종속을 없애도록 테이블을 분리
- 이행적 종속성이란 $A \rightarrow B$, $B \rightarrow C$ 의 관계가 성립할 때, $A \rightarrow C$ 가 성립되는 것을 말함
- (A,B)와 (B,C)로 분리하는 것이 제 3 정규화

관계(Relationship)의 개념

- 엔터티의 인스턴스 사이의 논리적인 연관성
- 엔터티의 정의, 속성 정의 및 관계 정의에 따라서도 다양하게 변할 수 있음
- 관계를 맺는다는 의미는 부모의 식별자를 자식에 상속하고, 상속된 속성을 매핑키(조인 키)로 활용 → 부모, 자식을 연결함

관계의 분류

- 관계는 존재에 의한 관계와 행위에 의한 관계로 분류
- **존재 관계는 엔터티 간의 상태를 의미**
- ex) 사원 엔터티는 부서 엔터티에 소속
- **행위 관계는 엔터티 간의 어떤 행위가 있는 것을 의미**
- ex) 주문은 고객이 주문할 때 발생

조인의 의미

-결국 데이터의 중복을 피하기 위해 테이블은 정규화에 의해 분리된다. 분리되면서 두 테이블은 서로 관계를 맺게 되고, 다시 이 두 테이블의 데이터를 동시에 출력하거나 관계가 있는 테이블을 참조하기 위해서는 데이터를 연결해야 하는데 이 과정을 조인이라고 함

상호배타적 관계

- 두 테이블 중 하나만 가능한 관계를 말함

트랜잭션이란

- **하나의 연속적인 업무 단위를 말함**
- 트랜잭션에 의한 관계는 필수적인 관계 형태를 가짐
- 하나의 트랜잭션에는 여러 SELECT, INSERT, DELETE, UPDATE 등이 포함될 수 있음

ex) 계좌이체를 예를 들면)

A 고객이 B 고객에게 100만원을 이체하려고 한다고 가정하자.

Step1) A 고객의 잔액이 100만원 이상인지 확인

Step2) 이상이면, A 고객 잔액을 -100 UPDATE

Step3) B 고객 잔액에 +100 UPDATE

이 때, 2번과 3번 과정이 동시에 수행되어야 한다. 즉, **모두 성공하거나 모두 취소되어야 함 (All or Nothing)**

→ 이런 특성을 갖는 연속적인 업무 단위를 트랜잭션이라고 한다.

*주의

1. A 고객 잔액 차감과 B 고객 잔액 가산이 서로 독립적으로 발생하면 안됨

→ 각각의 INSERT 문으로 개발되면 안됨

2. 부분 COMMIT 불가

→ 동시 COMMIT 또는 ROLLBACK 처리

필수적, 선택적 관계와 ERD

- 두 엔터티의 관계가 서로 필수적일 때 하나의 트랜잭션을 형성
- 두 엔터티가 서로 독립적 수행이 가능하다면 선택적 관계로 형성

IE 표기법)

- 원을 사용하여 필수적 관계와 선택적 관계를 구분
- 필수적 관계에는 원을 그리지 않는다.
- 선택적 관계에는 관계선 끝에 원을 그린다.

버커표기법)

- 실선과 점선으로 구분
- 필수적 관계는 관계선을 실선으로 표기
- 선택적 관계는 관계선을 점선으로 표기

NULL이란

- DBMS에서 아직 정해지지 않은 값을 의미
- 0과 빈문자열("")과는 다른 개념
- 모델 설계 시 각 컬럼별로 NULL을 허용할 지를 결정 (Nullable Column)

NULL의 특성

1. NULL을 포함한 연산 결과는 항상 NULL

→COMM 컬럼에 공백으로 보이는 것들이 NULL

2. 집계함수는 NULL을 제외한 연산 결과 리턴

***sum, avg, min, max 등의 함수는 항상 null을 무시한다.**

NULL의 ERD 표기법

- IE 표기법에서는 NULL 허용여부를 알 수 없음
- 버커 표기법에서는 속성 앞에 동그라미가 NULL 허용 속성을 의미 ○

식별자 구분(대체 여부에 따른)

1) 본질식별자

- 업무에 의해 만들어지는 식별자(꼭 필요한 식별자)

2) 인조식별자

- 인위적으로 만들어지는 식별자(꼭 필요하지 않지만 관리의 편의성 등의 이유로 인위적으로 만들어지는 식별자)
- 본질식별자가 복잡한 구성을 가질때 인위적으로 생성
- 주로 각 행을 구분하기 위한 기본키로 사용되며 자동으로 증가하는 일련번호 같은 형태임

인조식별자는 다음의 단점을 가진다.

1. 중복 데이터 발생 가능성 → 데이터 품질 저하
2. 불필요한 인덱스 생성 → 저장공간 낭비 및 DML 성능 저하

****인덱스는 원래 조회 성능을 향상시키기 위한 객체이며, 인덱스는 DML(INSERT/UPDATE/DELETE)시 INDEX SPLIT 현상으로 인해 성능이 저하된다.**