**<3과목: 데이터베이스 구축>**

데이터베이스 정규화

데이터 구조의 안정성을 최대화

중복을 배제하여 삽입, 삭제, 갱신 이상의 발생을 방지

데이터 삽입 시 릴레이션을 재구성할 필요성을 줄임

틀린 것: 데이터베이스의 개념적 설계 단계 이전에 수행(논리적 설계 단계에서 수행)

1차 정규화: 각 행마다 열의 값이 1개씩만 있어야함, 원자값(Atomic value); 열의 값이 여러 개인 경우 행으로 쪼갠다

2차 정규화: 모든 열이 완전 함수적 종속을 만족, 기본키 중 특정 열에만 종속된 열(부분적 종속)이 없어야 한다

3차 정규화: 이행적 종속을 없애도록 테이블을 분해, A -> B, B -> C가 성립할 때 A -> C가 성립되는 것을 없앰

BCNF 정규화: 모든 결정자가 후보키가 되도록 테이블을 분해, 후보키; 최소성, 유일성 만족

4차 정규화: 함수 종속이 아닌 다치 종속을 제거(다치종속: 두 개의 독립된 속성이 1:N으로 대응하는 관계)

5차 정규화: 후보키를 통하지 않는 조인 종속을 제거(조인종속: 테이블에 포함된 필드들의 부분집합을 포함하는 테이블들을 JOIN하여 생성할 수 있는 것)

데이터베이스 트랜잭션 성질

트랜잭션: 데이터베이스 시스템에서 병행제어 밑 회복작업시 처리되는 작업의 논리적 단위

사용자가 시스템에 대한 서비스 요구 시 시스템 이 응답하기 위한 상태변환 과정의 작업 단위

Atomicity(원자성): 트랜잭션의 모든 연산들이 정상적으로 수행 완료되거나 아니면 전혀 어떠한 연산도 수행되지 않은 원래 상태가 되도록 해야 한다

Consistency(일관성): 트랜잭션이 그 실행을 성공적으로 완료하면 언제나 일관성 있는 데이터데이스 상태로 변환

시스템이 가지고 있는 고정요소는 수행 전과 수행 완료 후의 상태가 같아야 한다

Isolation(독립성, 격리성): 둘 이상의 트랜잭션이 동시에 병행 실행되는 경우 어느 하나의 트랜잭션 실행 중에 다른 트랜잭션의 연산이 끼어들 수 없다

수행중인 트랜잭션은 완전히 완료될 때까지 다른 트랜잭션에서 수행결과를 참조할 수 없다

Durability(영속성, 지속성): 성공적으로 완료된 트랜잭션의 결과는 시스템이 고장나더라도 영구적으로 반영되야 한다

분산 데이터베이스 시스템과 관련한 설명

물리적으로 분산된 데이터베이스 시스템을 논리적으로 하나의 데이터베이스 시스템처럼 사용할 수 있도록 한 것

물리적으로 분산되어 지역별로 필요한 데이터를 처리할 수 있는 지역 컴퓨터(Local Computer)를 분산처리기(Distributed Processor)라고 한다

분산 데이터베이스 시스템을 위한 통신 네트워크 구조가 데이터 통신에 영향을 주므로 효율적으로 설계해야 한다

틀린 것: 데이터베이스가 분산되어 있음을 사용자가 인식할 수 있도록 분산투명성(Distribution Transparency)를 배제해야 한다

데이터베이스의 인덱스에 관련한 설명

문헌의 색인, 사전과 같이 데이터를 쉽고 빠르게 찾을 수 있도록 만든 데이터 구조

인덱스의 기본 목적은 검색 기능을 최적화하는 것

B-트리 인덱스는 분기를 목적으로 하는 Branch Block을 가지고 있다(Ordered Index, 순서 인덱스)

Between 등 범위(Range) 검색에 활용될 수 있다

테이블에 붙여진 색인으로 데이터 검색 시 처리 속도 향상에 도움이 된다

대부분의 데이터베이스에서 테이블을 삭제하면 인덱스도 같이 삭제

데이터베이스 성능에 많은 영향을 주는 DBMS의 구성요소

테이블과 클러스터에 연관되어 독립적인 저장 공간을 보유

[키, 값, 포인터] 쌍으로 구성되는 데이터 구조

데이터베이스의 물리적 구조와 밀접한 관계를 가짐

너무 많이 지정하면 오버헤드가 발생할 수 있으므로 적절히 지정해야 함

틀린 것:

시스템이 자동으로 생성하여 사용자가 변경할 수 없다(기본키에 대한 인덱스만 자동 생성)

인덱스의 추가, 삭제 명령어는 각각 ADD, DELETE이다

물리적 데이터베이스 구조의 기본 데이터 단위인 저장 레코드의 양식을 설계할 때 고려사항

데이터 타입

데이터 값의 분포

접근 빈도

틀린 것: 트랜잭션 모델링

SQL의 기능에 따른 분류 중에서 REVOKE문과 같이 데이터의 사용권한을 관리하는데 사용하는 언어

DCL(Data Control Language): 사용자관리, 사용자별 릴레이션 또는 데이터를 관리하고 접근하는 권한, Grant, Revoke

틀린 것:

DDL(Data Definition Language): 각 릴레이션을 정의, Create, Alter, Drop, Rename, Truncate

DML(Data Manipulaton Language): 데이터를 추가/수정/삭제, Select, Insert, Update, Delete

DUL(Data User Language): 없음

TCL(Transaction Control Language): 논리적인 작업의 단위를 묶어서 DML에 의해 조작된 결과를 작업단위 별로 제어

Commit, Rollback, Savepoint

데이터 사전에 대한 설명

시스템 카탈로그 및 시스템 데이터베이스라고도 한다

데이터베이스에 대한 데이터인 메타데이터를 저장하고 있다

데이터 사전에 있는 데이터를 실제로 접근하는 데 필요한 위치정보는 데이터 디렉토리라는 곳에서 관리한다

틀린 것: 데이터 사전 역시 데이터베이스의 일종이므로 일반 사용자가 생성, 유지 및 수정할 수 있다

데이터베이스에서 릴레이션에 대한 설명

릴레이션: 관계형 데이터베이스에서 정보를 구분하여 저장하는 기본단위, DB 테이블

모든 튜플은 서로 다른 값을 가지고 있다

각 속성은 릴레이션 내에서 유일한 이름을 가진다

모든 속성값은 원자값(Atomic value)를 가진다

한 릴레이션에 포함된 튜플 사이에는 순서가 없다

도메인(Domain)은 하나의 속성이 가질 수 있는 같은 타입의 모든 값의 집합으로 각 속성의 도메인은 원자값을 가짐

틀린 것:

하나의 릴레이션에서 튜플은 특정한 순서를 가진다

릴레이션의 각 행을 스키마(Schema)라 한다(튜플)

릴레이션의 각 열을 튜플(Tuple)이라 하며, 하나의 튜플은 각 속성에서 정의된 값을 이용하여 구성된다(속성)

속성은 릴레이션의 논리적인 구조를 정의한 것으로, 릴레이션의 이름과 릴레이션에 포함된 속성들의 집합(스키마)

데이터베이스에서 뷰(View)에 대한 설명

뷰는 다른 뷰를 기반으로 새로운 뷰를 만들 수 있다

뷰는 일종의 가상테이블이며, update에는 제약이 따른다

뷰는 기본테이블을 만드는 것처럼 create view를 사용하여 만들 수 있다

DBA는 보안성 측면에서 뷰를 활용할 수 있다

사용자가 필요한 정보를 요구에 맞게 가공하여 뷰로 만들 수 있다

틀린 것:

뷰는 논리적으로 존재하는 기본테이블과 다르게 물리적으로만 존재하며 카탈로그에 저장된다

뷰에 대한 삽입, 갱신, 삭제 연산 시 제약사항이 따르지 않는다(제약이 따름)

트랜잭션의 상태 중 트랜잭션의 마지막 연산이 실행된 직후의 상태로, 모든 연산의 처리는 끝났지만 트랜잭션이 수행한 최종 결과를 데이터베이스에 반영하지 않은 상태

Partially Committed

틀린 것:

활성화(Active) - Read/Write – 부분완료(Partially committed) – Permanent Change – 완료(committed)

| Failure | Failure

실패(Failed) – Rollback – 철회(Aborted)

Active: 트랜잭션이 작업을 시작하여 실행 중인 상태

Failed: 트랜잭션에 오류가 발생하여 실행이 중단된 상태

Aborted: 트랜잭션이 비정상적으로 종료되어 Rollback 연산을 수행한 상태

Committed: 트랜잭션이 성공적으로 종료되어 commit 연산을 실행한 후의 상태

키의 종류 중 유일성과 최소성을 만족하는 속성 또는 속성들의 집합

후보키(Candidate key): 릴레이션을 구성하는 속성들 중 튜플을 유일하게 식별하기 위해 사용되는 속성들의 부분집합

틀린 것:

Atomic key: 없음, Atomic 한 열에 하나의 데이터

Super key: 슈퍼키, 한 릴레이션 내에 있는 속성들의 집합으로 구성된 키, 릴레이션을 구성하는 모든 튜플 중 슈퍼키로 구성된 슈퍼키로 구성된 속성의 집합과 동일한 값은 나타내지 않는다. 릴레이션을 구성하는 모든 튜플에 대해 유일성은 만족, 최소성은 만족하지 못한다

Primary key: 기본키, 후보키 중 특별히 선정된 키, 중복된 값을 가질 수 없다. Null을 가질 수 없다

Alternate key: 대체키, 후보키 중 선정된 기본키를 제외한 나머지 후보키

Foreign key: 외래키, 다른 릴레이션의 기본키를 참조하는 속성 또는 속성들의 집합, 다른 릴레이션 간의 관계를 표현할 때 사용, 참조 릴레이션의 기본키와 동일한 키 속성을 가진다.

유일성(Unique): 하나의 키 값으로 하나의 튜플만을 유일하게 식별할 수 있어야 한다

최소성(Minimality): 키를 구성하는 속성 하나를 제거하면 유일하게 식별할 수 없도록 꼭 필요한 최소의 속성,

데이터베이스에서 개념적 설계 단계에 대한 설명

산출물로 E-R Diagram을 만들 수 있다

DBMS에 독립적인 개념 스키마를 설계한다:

DBMS: 데이터베이스 데이터의 집합, 저장, 관리 기능 제공 프로그램

스키마: 데이터베이스의 구조와 제약조건에 관한 전반적인 명세를 기술한 메타데이터의 집합,

Entity, Attribute, Relationship, Constraint

논리적 설계 단계의 앞 단계에서 수행된다

틀린 것: 트랜잭션 인터페이스를 설계 및 작성한다

테이블의 기본키로 지정된 속성에 관한 설명

Not null로 널 값을 가지지 않는다

릴레이션에서 튜플을 구별할 수 있다

외래키로 참조될 수 있다

틀린 것: 검색할 때 반드시 필요하다

데이터 모델의 구성요소 중 데이터 구조에 따라 개념 세계나 컴퓨터 세계에서 실제로 표현된 값들을 처리하는 작업

Operation: 데이터베이스에 저장된 실제 데이터를 처리하는 작업에 대한 명세, 데이터베이스를 조작하는 기본도구

틀린 것:

Relation: 개체 간의 관계 또는 속성 간의 관계

Data Structure: 논리적으로 표현된 개체 타입들 간의 관계로서 데이터 구조 밑 정적 성질을 표현

Constraint: 데이터베이스에 저장될 수 있는 실제 데이터의 논리적인 제약조건

무결성 제약조건 중 개체 무결성 제약조건에 대한 설명

무결성 제약조건: 데이터베이스의 정확성, 일관성을 보장하기 위해 저장, 삭제, 수정 등을 제약하기 위한 조건

개체 무결성: 각 릴레이션의 기본키를 구성하는 속성은 null, 중복된 값을 가질 수 없다

참조 무결성: 외래키 값은 null이거나 참조하는 릴레이션의 기본키 값과 동일해야 한다

도메인 무결성: 속성들의 값은 정의된 도메인에 속한 값이어야 한다

고유 무결성: 특정 속성에 대해 고유값을 가지도록 조건이 주어진 경우, 릴레이션의 각 튜플이 가지는 속성값들은 서로 달라야 한다

Null 무결성: 릴레이션의 특정 속성 값은 null이 될 수 없다

키 무결성: 각 릴레이션은 최소한 한 개 이상의 키가 존재해야 한다

기본키는 null값을 가져서는 안되며 릴레이션 내에 오직 하나의 값만 존재해야 한다

틀린 것:

릴레이션 내의 튜플들이 각 속성의 도메인에 정해진 값 만을 가져야한다 -> 도메인 무결성

자식 릴레이션의 외래키는 부모 릴레이션의 기본키와 도메인이 동일해야 한다 -> 참조 무결성

자식 릴레이션의 값이 변경될 때 부모 릴레이션의 제약을 받는다 -> 참조 무결성

관계 데이터 모델에서 릴레이션에 포함되어 있는 튜플의 수

Cardinality: 릴레이션 튜플의 개수

틀린 것:

Degree: 차수, 한 릴레이션 안에 있는 attribute(속성)의 수

Attribute: 속성, 개체의 항목, 고유한 이름, 개체의 특성 기술, DB를 구성하는 가장 작은 논리적 단위, 파일구조상 데이터항목 또는 데이터 필드에 해당

Cartesian product: 곱집합; 열 개수=두 테이블 열 개수의 합, 행 개수=두 테이블 행 개수 곱

테이블 생성 권한 부여

Grant Create Table To HRD;

데이터베이스 설계 단계 중 물리적 설계 시 고려사항

응답시간

트랜잭션 처리량

저장 공간의 효율화

틀린 것: 스키마의 평가 및 정제

Delete 명령에 대한 설명

테이블의 행을 삭제할 때 사용

SQL을 사용 용도에 따라 분류할 경우 DML에 해당

기본 사용 형식은 ‘Delete From 테이블 Where 조건;’ 이다

틀린 것: Where 조건절이 없는 Delete명령을 수행하면 Drop Table명령을 수행했을 때와 동일한 효과를 얻을 수 있다

E-R 모델 표기법

마름모: 관계

사각형: 개체

선: 관계 – 속성 연결

원: 속성

다중값: 겹친 원

∀: 관계 해석에서 ‘모든 것에 대하여’의 의미를 나타내는 논리, All

∃: 존재한다, Exist

∈: t가 r에 속함(t ∈ r), t; 원소, r; 집합

⊂: a 가 b의 부분집합이다(a ⊂ b), a,b; 집합

→: Y는 X에 함수적 종속이다(X → Y), X; 결정자, Y; 종속자

함수적 종속: X의 값을 알면 Y의 값을 바로 식별할 수 있고, X의 값에 의해 Y의 값이 달라지는 것

Create Table 기능

속성의 Not null 여부 지정

기본키를 구성하는 속성 지정

Check 제약조건의 정의

틀린 것: 속성 타입 변경

SQL에 관련한 설명

데이터 정의어(Data Define Language, DDL)는 기본 테이블, 뷰 테이블, 또는 인덱스 등을 생성, 변경, 제거하는데 사용되는 명령어이다

Distinct를 활용하여 중복값을 제거할 수 있다

Join을 통해 여러 테이블의 레코드를 조합하여 표현할 수 있다

틀린 것: Revoke 키워드를 사용하여 열 이름을 다시 부여할 수 있다

SQL Union All: 각 쿼리의 모든 결과를 포함한 합집합(중복 제거 안함)

분산 데이터베이스(Distributed Database System)에 대한 설명

분산 DB는 논리적으로는 하나의 시스템에 속하지만 물리적으로는 여러 개의 컴퓨터 사이트에 분산되어 있다

위치 투명성, 중복 투명성, 병행 투명성, 장애 투명성을 목표로 한다

데이터베이스의 설계가 비교적 어렵고, 개발 비용과 처리 비용이 증가한다는 단점이 있다

틀린 것: 분산 데이터베이스 시스템의 주요 구성 요소는 분산 처리기, P2P 시스템, 단일 데이터베이스 등이 있다

(분산 데이터베이스 시스템의 주요 구성 요소는 분산처리기, 분산 데이터베이스, 통신네트워크, 분산 트랜잭션)

분산 데이터베이스 구조: 전역, 분할(단편화), 할당, 지역 스키마

Drop View V\_1 CASCADE;

CASCADE: 테이블이 다른 테이블과 참조 관계가 존재하는 경우 테이블 제거 또는 비활성화 할 수 없다, Cascade는 부모 테이블과 자식 테이블의 제약조건을 비활성화 시키기 위해 사용, 일시적으로 참조관계를 끊을 수 있다

데이터베이스에서 병행제어의 목적

시스템 활용도 최대화

사용자에 대한 응답시간 최소화

데이터베이스 일관성 유지

여러 사용자들의 데이터베이스 공동 사용을 최대화

동시에 여러 개의 트랜잭션을 병행 수행할 때, DB의 일관성을 파괴하지 않도록 제어하는 것

틀린 것: 데이터베이스 공유 최소화

병행제어 기법: 로킹, 타임 스탬프, 최적 병행 수행(검증, 확인, 낙관적), 다중 버전

틀린 것: 시분할 기법; 사용자가 여러 개의 작업을 동시에 수행하는 것처럼 느끼도록 분할하는 운영체제 운영기법

데이터베이스의 무결성 규정(Integrity Rule)과 관련한 설명

무결성 규정에는 데이터가 만족해야 될 제약조건, 규정을 참조할 때 사용하는 식별자 등의 요소가 포함

무결성 규정의 대상으로는 도메인, 키, 종속성 등이 있다

릴레이션 무결성 규정(Relation Integrity Rule)은 릴레이션을 조작하는 과정에서의 의미적 관계(Semantic Relationship)을 명세한 것

틀린 것: 정식으로 허가 받은 사용자가 아닌 불법 사용자에 의한 갱신으로부터 데이터베이스를 보호하기 위한 규정

무결성 규칙: 데이터베이스에서 무결성 규칙은 데이터 무결성을 지키기 위한 모든 제약사항을 뜻함

카티션 프로덕트(Cartesian Product): 두 테이블 곱집합, 튜플 수는 각 테이블의 곱, 열은 각 테이블의 열을 더함

물리적 데이터베이스 설계에 대한 설명

물리적 설계의 목적은 효율적인 방법으로 데이터를 저장하는 것

트랜잭션 처리량과 응답시간, 디스크 용량 등을 고려해야 한다

저장 레코드의 형식, 순서, 접근 경로와 같은 정보를 사용하여 설계한다

논리적 설계에 기반하여 설계한다

레코드 집중의 분석 및 설계

접근 경로 설계

저장 레코드의 양식 설계

틀린 것:

트랜잭션의 인터페이스를 설계하며, 데이터 타입 및 데이터 타입들 간의 관계로 표현한다(논리적 설계에서 인터페이스를 사용자의 입장에서 편리하도록 설계하는 것)

목표 DBMS에 맞는 스키마 설계

SQL에서 HAVING을 사용할 수 있는 절: GROUP BY 절, 그룹에 대한 조건을 지정

관계 데이터베이스에 있어서 관계 대수 연산

일반 집합 연산자:

합집합(Union), 교집합(Intersection), 차집합(Difference), 교차곱(Cartesian Product)

순수 관계 연산자:

Select, Project, Join, Division

Select(σ): 수평단절, 행을 다 가져옴, 조건을 만족하는 데이터를 수평적으로 도출함, 중복이 존재하지 않음

Project(π): 수직단절, 열을 다 가져옴, 원하는 데이터를 수직적으로 도출, 한 릴레이션의 속성의 부분집합

Join(⨝): 공통 속성을 통해 두 개의 릴레이션 튜플을 연결 -> 만들어진 튜플로 반환

Division(÷): 릴레이션 S의 모든 튜플과 관련있는 릴레이션 R의 튜플 반환

(): 테이블 이름

ex) ∏이름(σ학과=’교육’(학생)) -> Select 이름 From 학생 Where 학과=’교육’;

로킹 단위(Locking Granularity)에 대한 설명

로킹: 병행 제어 기법의 종류 중 하나

로킹단위: 한번에 로킹할 수 있는 객체의 크기

로킹 단위가 크면 병행성 수준이 낮아진다(로킹 큼 – 로크 수 적어짐 – 병행성 수준 낮아짐 – 병행 제어 기법 간단)

직렬화 기법, 데이터베이스, 파일, 레코드 등은 로킹 단위가 될 수 있음

로킹 단위가 작아지면, 데이터베이스 공유도 증가, 로킹 오버헤드 증가, 로크의 수 증가, 병행성 수준 높아짐

관계 대수에 대한 설명

관계 대수: 관계형 DB에서 원하는 정보와 그 정보를 검색하기 위해서 어떻게 유도하는가를 기술하는 절차 언어

릴레이션 조작을 위한 연산의 집합으로 피연산자와 결과가 모두 릴레이션이다

일반 집합 연산과 순수 관계 연산으로 구분된다

질의에 대한 해를 구하기 위해 수행해야 할 연산의 순서를 명시한다

틀린 것: 원하는 릴레이션을 정의하는 방법을 제공하며 비절차적 언어이다

타임스탬프 기법: 동시성 제어를 위한 직렬화 기법으로 트랜잭션 간의 처리 순서를 미리 정하는 방법

로킹 기법: 같은 자원을 액세스하는 다중 트랜잭션 환경에서 DB의 일관성과 무결성을 유지하기 위해 트랜잭션의 순차적 진행을 보장하는 직렬화 기법

파티셔닝: 병렬 데이터베이스 환경 중 수평 분할에서 활용되는 분할 기법, 파티셔닝의 유형에는 Range, Hash, List, Composite, Round Robin 등이 있다

수평 분할: 하나의 테이블의 각 행을 다른 테이블에 분산시키는 것

라운드로빈 분할: 파티션에 행의 고른 분포를 원할 때 사용, 해시 분할과 달리 분할 열을 명시할 필요가 없음, 라운드 로빈 분할로 회전하면서 새로운 행이 파티션에 할당, 기본키가 반드시 필요하진 않음

시스템 카탈로그: DBMS가 스스로 생성하고 유지하기 때문에 Insert, Delete, Update문으로 시스템 카탈로그를 갱신하는 것은 허용되지 않음

시스템 자신이 필요로 하는 스키마 및 여러 가지 객체에 관한 정보를 포함하고 있는 시스템 데이터베이스

시스템 카탈로그에 저장되는 내용을 메타데이터 라고도 한다

데이터베이스에 포함되는 데이터 객체에 대한 정의나 명세에 대한 정보를 유지 관리

DBMS가 스스로 생성하고 유지하는 데이터베이스 내의 특별한 테이블의 집합체

틀린 것: 시스템 카탈로그의 갱신은 무결성 유지를 위하여 SQL을 이용하여 사용자가 직접 갱신해야 한다

SQL 문에서 Select에 대한 설명

From 절에는 질의에 의해 검색될 데이터들을 포함하는 테이블 명을 기술한다

Having 절은 Group by 절과 함께 사용되며, 그룹에 대한 조건을 지정한다

Order by 절은 특정 속성을 기준으로 정렬하여 검색할 때 사용한다

틀린 것: 검색결과에 중복되는 레코드를 없애기 위해서는 Where 절에 Distinct 키워드를 사용한다(Select 뒤에 사용)

Delete: 조건을 달아서 특정 행을 삭제할 때 사용, DML

Drop: View, Table, Database를 완전히 삭제할 때 사용, DDL

Intersect: 교집합

Select 학번 From R1;

Intersect

Select 학번 From R2

* R1의 학번 테이블과 R2의 학번 테이블의 교집합 출력

데이터베이스 설계: 개념적 설계, 논리적 설계, 물리적 설계

개념적 설계: 사용자의 요구사항 분석 후, 데이터베이스에 대한 추상적인 형태를 설계, 개념적 모델을 이용한 개념적 스키마 생성(데이터베이스에 대한 추상적인 설계도, 개체 관계 다이어그램)

논리적 설계: 논리적 모델을 이용하여 논리적 스키마 생성, ERD를 이용하여 데이터베이스 스키마를 설계, 테이블 구조도, 개념적 설계 단계에서 생성된 ERD를 바탕으로 생성되는 테이블들의 집합

물리적 설계: 특정 DBMS가 제공하는 물리적 구조에 따라 테이블 저장 구조 설계, 필드의 데이터타입, 인덱스, 테이블 저장 방법 등을 정의

보안의 3요소

기밀성(Confidentiality): 인가된 사용자만 정보 자산에 접근할 수 있는 것, 방화벽, 암호, 패스워드 등이 대표적

무결성(Integrity): 적절한 권한을 가진 사용자가 인가한 방법으로만 정보를 변경할 수 있도록 하여 접근 통제

가용성(Availability): 필요한 시점에 정보 자산에 대한 접근이 가능하도록 하는 것, 백업

데이터 기밀성(Data Confidentiality): 데이터 기밀성은 공격자가 데이터를 볼 수 없게 보호

데이터 무결성(Data Integrity): 데이터의 무결성은 변경, 삽입, 삭제, 재연 등으로부터 정보를 보호

SQL Update

Update (테이블) Set (열 = 변경할 값) Where (조건)

Anomaly(이상): 데이터 속성 간의 종속성에 대한 엄밀한 고려 없이 잘못 설계된 데이터베이스에서는 데이터 처리 연산 수행 시 발생할 수 있는 현상, 데이터의 중복으로 인하여 관계연산을 처리할 때 예기치 못한 현상이 발생하는 것

갱신 이상(Modification Anomaly: 반복된 데이터 중에 일부를 갱신할 시 데이터의 불일치가 발생

삽입 이상(Insertion Anomaly): 불필요한 정보를 함께 저장하지 않는 한 어떤 정보를 저장하는 것이 불가능

삭제 이상(Deletion Anomaly): 필요한 정보를 함께 삭제하지 않는 한 어떤 정보를 삭제하는 것이 불가능

개념 스키마: 사용자와 데이터베이스 관리자 관점의 스키마, DB에 실제로 어떤 데이터가 저장되었으며 데이터 간의 관계는 어떻게 되는지를 정의, 전체 관점으로 한 개만 존재하며 접근권한, 보안 및 무결성 등에 관한 정의

내부 스키마: 저장 장치와 데이터베이스 설계자 및 개발자 관점의 스키마, 개념적 스키마를 물리적 저장장치에 구현하는 방법을 정의, 물리적 구조 및 내부 레코드의 물리적 순서 등을 표현

외부 스키마: 사용자 관점의 스키마, 사용자 또는 프로그램의 입장에서의 논리적 구조로 여러 개가 존재

데이터베이스 설계 단계

요구조건 분석/명세: DB의 사용자, 사용목적, 사용범위, 제약조건 등에 대한 내용을 정리하고 명세서를 작성

개념적 설계: 정보를 구조화하기 위해 추상적 개념으로 표현하는 과정으로 개념 스키마 모델링과 트랜잭션 모델링을 병행하고, 요구 조건 분석을 통해 DBMS 독립적인 E-R 다이어그램을 작성

(개념 스키마 모델링 / 트랜잭션 모델링 / 독립적인 개념 스키마 설계 / E-R 다이어그램)

논리적 설계: 자료를 컴퓨터가 이해할 수 있도록 특정 DBMS의 논리적 자료 구조로 변환하는 과정

(트랜잭션 인터페이스 설계 / 스키마 평가 및 정제 / 목표 DBMS에 맞는 논리스키마 설계 / 논리적 구조의 데이터로 모델화)

물리적 설계: 논리적 구조로 표현된 데이터를 물리적 구조의 데이터로 변환하는 과정

(저장 구조 및 접근 경로 설정 / 레코드 집중의 분석,설계 / 저장 레코드 양식 설계)

Recovery: 트랜잭션을 수행하는 도중 장애로 인해 손상된 데이터베이스를 이전의 정상적인 상태로 복구시키는 작업