흥달쌤과 함께하는

정보처리기사 실기
최종정리 특강

[2과목 - 데이터베이스 구축]

1억뷰 N잡

YouTube 흥달쌤 (https://bit.ly/3KtwdLG) E-Mail hungjik@naver.com
네이버 카페 흥달쌤의 IT 이야기 (https://cafe.naver.com/sosozl/)

01 데이터베이스 구축

Section 1. 데이터베이스 개념

1. 데이터베이스 개념

⑴ 데이터베이스의 정의

- 통합 데이터(Integrated Data) : 중복이 최소화된 데이터의 모임
- 저장 데이터(Stored Data) : 저장 매체에 저장된 데이터
- 운영 데이터(Operational Data) : 조직의 목적을 위해 필요한 데이터
- 공유 데이터(Shared Data) : 여러 응용 프로그램들이 공동으로 사용하는 데이터

⑵ 데이터베이스의 특징

- 실시간 접근성(Real Time Accessibility)
- 계속적인 변화(Continuous Evolution)
- 동시 공유(Concurrent Sharing)
- 내용에 의한 참조(Content Reference)
- 데이터의 독립성(Independence)

⑶ 데이터 언어

- DDL(Data Definition Language : 데이터 정의어)
- DML(Data Manipulation Language : 데이터 조작어)
- DCL(Data Control Language : 데이터 제어어)

⑷ 스키마(Schema)

- 데이터베이스의 구조와 제약조건에 관해 전반적인 명세를 기술한 것
- 3계층 스키마
  1) 외부 스키마(External Schema) : 사용자 뷰
  2) 개념 스키마(Conceptual Schema) : 전체적인 구조와 제약 조건
  3) 내부 스키마(Internal Schema) : 저장 스키마
- 데이터 독립성
  1) 논리적 독립성 : 개념 스키마가 변경되어도 외부 스키마에는 영향을 미치지 않도록 지원
  2) 물리적 독립성 : 내부 스키마가 변경되어도 외부/개념 스키마가 영향을 받지 않도록 지원

2. 데이터베이스 관리 시스템(Database Management System)

⑴ DBMS의 정의

- DBMS를 통해 데이터베이스를 관리하여 응용 프로그램들이 데이터베이스를 공유하고, 사용할 수 있는 환경을 제공

⑵ DBMS의 기능

- 데이터 정의
- 데이터 조작
- 데이터 제어
- 데이터 공유
- 데이터 보호
- 데이터 구축
- 유지보수

⑶ DBMS의 종류

- 계층형(Hierarchical DataBase) : 트리형태
- 네트워크형(Network DataBase) : N:N(다대다) 구성, CODASYL DBTG모델
- 관계형(Relational DataBase) : 테이블 구조로 단순화시킨 모델
- 객체 지향형(Object-Oriented DataBase) : 객체지향 프로그래밍 개념에 기반하여 만든 데이터베이스 모델
- 객체 관계형(Object-Relational DataBase)
- NoSQL : SQL뿐만 아니라 다양한 특성을 지원
- NewSQL : RDBMS의 SQL과 NoSQL의 장점을 결합한 관계형 모델

Section 2. 데이터베이스 설계

1. 데이터베이스 설계 단계

⑴ 요구조건 분석

⑵ 개념적 설계

- DBMS에 독립적으로 설계
- 데이터베이스의 개념적 스키마 구성(E-R 다이어그램)

⑶ 논리적 설계

- 목표 DBMS의 논리적 자료 구조로 변환
- 데이터베이스의 논리적 스키마 생성
- 관계형 데이터베이스인 경우 이 단계에서 테이블을 설계하는 정규화 과정 수행
- 트랜잭션 인터페이스 설계

⑷ 물리적 설계

- 특정 DBMS의 물리적 구조와 내부적인 저장구조 설계
- 레코드 집중의 분석 및 설계
- 트랜잭션 세부 설계

⑸ 구현

- 특정 DBMS의 DDL로 기술된 명령문을 컴파일하고, 실행시켜 데이터베이스 스키마 생성

Section 3. 데이터 모델링

1. 데이터모델 개념

⑴ 데이터모델 개념

- 현실세계의 요소를 인간과 컴퓨터가 이해할 수 있는 정보로 표현한 것

⑵ 데이터모델 표시해야 할 요소

- 구조(Structure) : 개체 타입과 개체 타입들 간의 관계
- 연산(Operation) : 저장될 데이터를 처리하는 방법
- 제약조건(Constraint) : 데이터의 논리적인 제약조건

2. 개체-관계 모델(Entity Relation Model)

⑴ 개체-관계 모델 개념

- 데이터베이스에 대한 요구사항을 그래픽적으로 표현하는 방법

⑵ 개체(Entity)

- 현실 세계에서 꼭 필요한 사람이나 사물과 같이 구별되는 모든 것

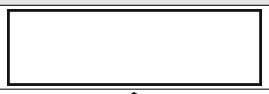
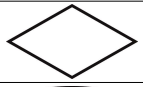
⑶ 애트리뷰트, 속성(Attribute)

- 개체나 관계가 가지고 있는 고유의 특성
- 속성의 유형
  - 단일 값 속성 : 값을 하나만 가질 수 있는 속성(이름)
  - 다중 값 속성 : 값을 여러 개 가질 수 있는 속성(취미)
  - 단순 속성 : 의미를 더는 분해할 수 없는 속성
  - 복합 속성 : 의미를 분해할 수 있는 속성
  - 유도 속성 : 다른 속성의 값에서 유도되어 결정되는 속성
  - 널 속성 : 아직 결정되지 않은 존재하지 않는 값
  - 키 속성 : 개체를 식별하는 데 사용하는 속성

⑷ 관계(Relationship)

- 서로 다른 개체가 맺고 있는 의미 있는 연관성

⑸ E-R 다이어그램 기호

기호	기호 이름	설명
	사각형	- 개체(Entity)
	마름모	- 관계(Relationship)
	타원	- 속성(Attribute)
	밑줄 타원	- 기본키 속성
	이중 타원	- 복합속성
	선 링크	- 개체와 속성 연결

3. 데이터 모델의 품질 기준

- 정확성 : 요구사항을 정확하게 반영
- 완전성 : 요구사항 및 업무영역 반영 시 누락없이 작성
- 준거성 : 준수 요건들을 누락없이 준수(표준, 규칙, 법적요건)
- 최신성 : 현행 시스템의 최신 상태 반영
- 일관성 : 모델 표현의 일관성 유지
- 활용성 : 업무 변화 시 설계 변경없이 유연하게 설계

Section 4. 논리 데이터베이스 설계

1. 논리적 데이터 모델링

⑴ 논리적 모델링

- 개념적 설계에서 추출된 실체와 속성들의 관계를 구조적으로 설계하는 단계

2. 데이터베이스 정규화(Normalization)

⑴ 정규화의 개념

- 관계형 데이터베이스의 설계에서 중복을 최소화하게 데이터를 구조화

⑵ 이상 현상(Anomaly)

- 데이터 중복으로 인해 릴레이션 조작 시 예상하지 못한 곤란한 현상이 발생
- 이상의 종류

1) 삽입 이상 : 데이터를 삽입할 때 불필요한 데이터가 함께 삽입되는 현상
2) 삭제 이상 : 한 튜플을 삭제할 때 연쇄 삭제 현상으로 인해 정보 손실
3) 갱신 이상 : 튜플의 속성값을 갱신할 때 일부 튜플의 정보만 갱신되어 정보에 모순이 생기는 현상

⑶ 함수적 종속(Functional Dependency)

1) 완전 함수적 종속(Full Functional Dependency)
   - 기본키를 구성하는 모든 속성이 포함된 기본키의 부분집합에 종속된 경우
2) 부분 함수적 종속(Partial Functional Dependency)
   - 기본키가 여러 속성으로 구성되어 있을 때, 기본키를 구성하는 속성 중 일부만 종속되는 경우
3) 이행적 함수 종속(Transitive Functional Dependecy)
   - X→Y, Y→Z 이러한 종속 관계가 있을 경우, X→Z가 성립되는 경우

⑷ 정규화 과정

1) 제 1정규형(1NF) : 도메인이 원자값만으로 구성
2) 제 2정규형(2NF) : 부분 함수적 종속 제거
3) 제 3정규형(3NF) : 이행적 함수 종속 제거
4) 보이스/코드(BCNF) 정규형 : 결정자 중 후보키가 아닌 것들을 제거
5) 제 4정규형(4NF) : 다치 종속 제거
6) 제 5정규형(5NF) : 조인 종속 이용

Section 5. 물리 데이터베이스 설계

1. 반정규화

- 시스템의 성능향상과 개발 편의성 등을 위해 정규화에 위배되는 중복을 허용하는 기법
- 반정규화의 적용순서
  - 반정규화 대상 조사 → 다른 방법으로 유도 → 반정규화 수행

2. 데이터베이스 이중화

⑴ 데이터베이스 이중화

- 장애발생 시 데이터베이스를 보호하기 위해 동일한 데이터베이스를 중복시켜 동시에 갱신하여 관리하는 방법

⑵ 데이터베이스 이중화의 분류

- Eager 기법 : 변경 발생 즉시 반영
- Lazy 기법 : 트랜잭션 완료 후 반영

⑶ 데이터베이스 이중화의 종류

- Active-Active, Active-Standby(Hot, Warm, Cold)

3. 데이터베이스 백업

⑴ 데이터베이스 백업 개념

- 중단 사태에 대비하여 복구를 진행할 수 있도록 데이터를 주기적으로 복사하는 것

⑵ 백업 방식

- 전체 백업(Full Backup) : Data를 모두 백업
- 증분 백업(Incremental Backup) : 변경/추가된 Data만 백업
- 차등 백업(Differential Backup) : 변경/추가된 Data를 모두 포함하여 백업
- 실시간 백업(RealTime Backup) : 즉시 백업
- 트랜잭션 로그 백업(Transaction Log Backup) : 모든 SQL문을 기록한 로그
- 합성 백업 : 전체 백업본과 여러 개의 증분 백업

⑶ 복구 시간 목표/복구 시점 목표

1) 복구 시간 목표(RTO)

- 서비스 중단 시점과 서비스 복원 시점 간에 허용되는 최대 지연 시간

2) 복구 시점 목표(RPO)

- 마지막 복구 시점과 서비스 중단 시점 사이에 허용되는 데이터 손실량

4. 데이터베이스 암호화

⑴ 데이터베이스 암호화 방식

- API 방식 : 애플리케이션에서 암/복호화 수행
- Plug-in 방식 : 제품을 설치하여 암/복호화 수행
- TDE(Transparent Data Encryption) 방식 : DBMS 내장 모듈을 이용하여 암/복호화 수행
- 파일 암호화 방식 : 비정형 데이터도 암호화
- 하드웨어 방식 : 별도의 하드웨어 방식

Section 6. 데이터베이스 물리속성 설계

1. 파티셔닝

⑴ 파티셔닝 개념

- 데이터베이스를 여러 부분으로 분할하는 것

⑵ 샤딩(Sharding)

- 하나의 거대한 데이터베이스나 네트워크 시스템을 여러 개의 작은 조각으로 나누어 분산 저장하여 관리하는 것

⑶ 분할 기준

- 범위 분할(Range Partitioning)
- 목록 분할(List Partitioning)
- 해시 분할(Hash Partitioning)
- 라운드 로빈 분할(Round Robin Partitioning)
- 합성 분할(Composite Partitioning)

2. 클러스터 설계

- 디스크로부터 데이터를 읽어오는 시간을 줄이기 위해 데이터를 디스크의 같은 위치에 저장시키는 방법

3. 인덱스(Index)

⑴ 인덱스의 개념

- 추가적인 저장 공간을 활용하여 데이터베이스 테이블의 검색 속도를 향상시키기 위한 자료구조

⑵ 인덱스의 종류

- 클러스터 인덱스 : 해당 컬럼을 기준으로 테이블이 물리적으로 정렬
- 넌클러스터 인덱스 : 레코드의 원본은 정렬되지 않고, 인덱스 페이지만 정렬
- 밀집 인덱스 : 데이터 레코드 각각에 대해 하나의 인덱스 생성
- 희소 인덱스 : 레코드 그룹 또는 데이터 블록에 대해 하나의 인덱스

⑶ 인덱스의 구조

- 트리 기반 인덱스 : B+ 트리 인덱스를 주로 사용
- 비트맵 인덱스 : 비트를 이용하여 컬럼값을 저장하고 이용
- 함수 기반 인덱스 : 함수나 수식 결과 이용
- 비트맵 조인 인덱스 : 물리적인 구조는 비트맵 인덱스와 완전히 동일
- 도메인 인덱스 : 개발자가 자신이 원하는 인덱스 타입을 생성

4. 뷰(View)

⑴ 뷰의 개념

- 하나 이상의 기본 테이블로부터 유도된, 이름을 가지는 가상 테이블

⑵ 뷰의 특징

- 논리적 데이터 독립성을 제공
- ALTER VIEW문을 사용할 수 없다.
- 뷰로 구성된 내용에 대한 삽입, 삭제, 갱신 연산에 제약

5. 시스템 카탈로그

- 데이터베이스에 저장되어 있는 모든 개체들에 대한 정의에 대한 정보가 수록되어 있는 시스템 테이블
- 시스템 카탈로그를 데이터 사전(Data Dictionary)이라고도 한다.
- 사용자가 SQL문을 이용하여 내용을 검색해 볼 수 있지만, 수정은 불가능하다.
- 시스템 카탈로그는 데이터베이스 관리 시스템에 의해 생성되고 유지된다.

Section 7. 관계 데이터베이스 모델

1. 관계 데이터 모델

⑴ 관계 데이터 모델 개념

- 데이터의 논리적 구조가 릴레이션, 즉 테이블 형태의 평면 파일로 표현되는 데이터 모델

⑵ 관계 데이터 릴레이션의 구조

〈학생 릴레이션〉

⑶ 릴레이션

1) 릴레이션의 구성

- 릴레이션 스키마 : 릴레이션 이름과 모든 속성의 이름으로 정의하는 릴레이션의 논리적인 구조
- 릴레이션 인스턴스 : 릴레이션 스키마에 실제로 저장된 데이터의 집합

2) 릴레이션의 특징

- 튜플의 유일성 : 릴레이션 안에는 똑같은 튜플이 존재할 수 없음
- 튜플의 무순서성 : 튜플 사이에는 순서가 없음
- 속성의 무순서성 : 속성 사이에는 순서가 없음
- 속성의 원자성 : 속성은 더 이상 분해할 수 없는 원자값만 가진다.
- 튜플들의 삽입, 갱신, 삭제작업이 실시간으로 일어나므로 릴레이션은 수시로 변한다.

2. 관계데이터 언어(관계대수, 관계해석)

⑴ 관계 대수의 개념

- 원하는 데이터를 얻기 위해 데이터를 어떻게 찾는지에 대한 처리 과정을 명시하는 절차적인 언어

⑵ 순수 관계 연산자

1) SELECT

- 기호 : σ(시그마)
- 표기법 : σ〈조건〉(R)

2) PROJECT
- 기호 : π(파이)
- 표기법 : π<리스트>(R)

3) JOIN
- 기호 : ⋈(보타이)
- 표기법 : R⋈<조건>S

4) DIVISION
- 기호 : ÷(나누기)
- 표기법 : R÷S

⑶ 일반 집합 연산자

1) 합집합(Union)
- 표기법 : ∪

2) 교집합(Intersection)
- 표기법 : ∩

3) 차집합(Difference)
- 표기법 : -

4) 교차곱(Cartesian Product)
- 표기법 : X

⑷ 관계해석
- 관계 데이터 모델의 제안자인 코드(E. F. Codd)가 수학의 Predicate Calculus(술어 해석)에 기반을 두고 관계 데이터베이스를 위해 제안
- 관계해석은 원하는 정보가 무엇이라는 것만 정의하는 비절차적 특성
- 튜플 관계해석과 도메인 관계해석이 있다.
- 연산자

구분	기호	설명
연산자	∨	OR 연산
	∧	AND 연산
	¬	NOT 연산
정량자	∀	모든 가능한 튜플 “For All”
	∃	어떤 튜플 하나라도 존재

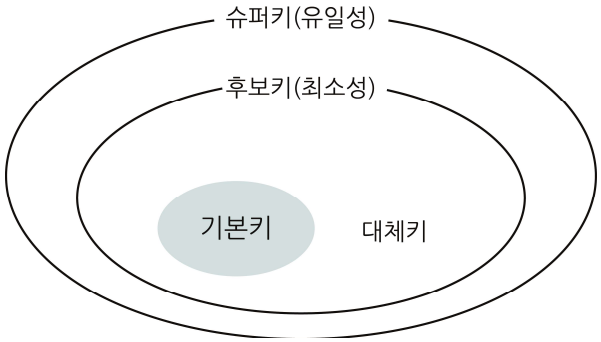
Section 8. 키와 무결성 제약조건

1. 키 종류

⑴ 키(Key)의 개념

- 릴레이션에서 다른 튜플들과 구별할 수 있는 유일한 기준이 되는 컬럼

⑵ 키(Key)의 종류



2. 데이터베이스 무결성

⑴ 데이터베이스 무결성 개념

- 데이터의 정확성, 일관성, 유효성이 유지되는 것

⑵ 데이터베이스 무결성 종류

1) 개체 무결성(Entity Integrity)
   - 모든 릴레이션은 기본 키(Primary Key)를 가져야 한다.
   - 기본키는 중복되지 않은 고유한 값을 가져야 한다.
   - 릴레이션의 기본키는 NULL 값을 허용하지 않는다.
2) 참조 무결성(Referential Integrity)
   - 외래키 값은 NULL이거나 참조하는 릴레이션의 기본키 값과 동일해야 한다.
   - 각 릴레이션은 참조할 수 없는 외래키 값을 가질 수 없다.
   - 참조 무결성 제약조건
     - 제한(Restrict) : 문제가 되는 연산을 거절
     - 연쇄(Cascade) : 부모 튜플 삭제 시 참조하는 자식의 튜플도 함께 삭제
     - 널값(Nullify) : 부모 튜플 삭제 시 참조하는 자식의 튜플은 NULL로 등록
     - 기본값(Default) : 부모 튜플 삭제 시 참조하는 자식의 튜플은 DEFAULT 값으로 등록
3) 도메인 무결성(Domain Integrity)
   - 속성들의 값은 정의된 도메인에 속한 값이어야 한다.
4) 고유 무결성(Unique Integrity)
   - 릴레이션의 특정 속성에 대해 각 튜플이 갖는 속성 값들이 서로 달라야 한다.
5) 키 무결성(Key Integrity)
   - 하나의 릴레이션에는 적어도 하나의 키가 존재해야 한다.
6) 릴레이션 무결성(Relation Integrity)
   - 삽입, 삭제, 갱신과 같은 연산을 수행하기 전과 후에 대한 상태의 제약

Section 9. 물리데이터 모델 품질 검토

1. CRUD 분석

⑴ CRUD의 개념

- 데이터 처리 기능인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 묶어서 표현한 말이다.

2. 옵티마이저

⑴ SQL 처리 흐름

1) 구문분석 단계
   - SQL 문이 문법에 따라 정상적으로 작성되었는지 분석
2) 실행 단계
   - 정의된 테이블의 해당 데이터 파일로부터 테이블을 읽어서 데이터버퍼 캐시영역에 저장
3) 추출 단계
   - 데이터버퍼 캐시영역에서 관련 테이블 데이터를 읽어서 사용자가 요청한 클라이언트로 전송

⑵ 옵티마이저 개념

- 사용자가 질의한 SQL문에 대해 최적의 실행 방법을 결정하는 역할을 수행
- 옵티마이저의 구분
  1) 규칙기반 옵티마이저(Rule Based Optimizer) : 규칙(우선순위)를 가지고 실행 계획 생성
  2) 비용기반 옵티마이저(Cost Based Optimizer) : 통계정보를 활용하여 실행 계획 생성

3. SQL 성능 튜닝

⑴ 튜닝의 개념

- SQL문을 최적화하여 빠른 시간 내에 원하는 결과값을 얻기 위한 작업

⑵ 튜닝 영역

- 데이터베이스 설계튜닝 : 정규화 및 반정규화하여 재설계
- 데이터베이스 환경 : 메모리나 블록 크기 지정
- SQL 문장 튜닝 : 성능을 고려하여 SQL 문장 작성

⑶ Row Migration / Row Chaining

- Row Migration : 다른 블록에 데이터를 넣고, 링크를 남긴다.
- Row Chaining : 두 개의 블록에 작성

Section 10. 분산 데이터베이스

1. 분산 데이터베이스

⑴ 분산 데이터베이스(Distribute Database)의 정의

- 여러 곳으로 분산되어 있는 데이터베이스를 하나의 가상 시스템으로 사용할 수 있도록 한 데이터베이스

⑵ 분산 데이터베이스 구성요소

- 분산 처리기
- 분산 데이터베이스
- 통신 네트워크

⑶ 분산 데이터베이스의 적용 기법

1) 테이블 위치 분산
- 설계된 테이블의 위치를 각각 다르게 위치시키는 것

2) 테이블 분할(Fragmentation) 분산
- 각각의 테이블을 쪼개어 분산하는 방법

3) 테이블 복제(Replication) 분산
- 동일한 테이블을 다른 지역이나 서버에서 동시에 생성하여 관리하는 유형
- 부분복제, 광역복제

4) 테이블 요약(Summarization) 분산
- 지역 간 또는 서버 간에 데이터가 비슷하지만 서로 다른 유형으로 존재
- 분석요약, 통합요약

⑷ 투명성 조건
- 위치 투명성(Location) : 실제 위치를 알 필요없이 논리적인 명칭으로 액세스
- 분할 투명성(Division) : 각 단편의 사본이 여러 위치에 저장
- 지역사상 투명성(Local Mapping) : 각 지역시스템 이름과 무관한 이름 사용 가능
- 중복 투명성(Replication) : 동일 데이터가 여러 곳에 중복되어 있어도 하나처럼 사용 가능
- 병행 투명성(Concurrency) : 다수의 트랜잭션들이 동시에 실현되더라도 결과는 영향을 받지 않음
- 장애 투명성(Failure) : 장애에도 불구하고 트랜잭션을 정확하게 처리함

⑸ CAP 이론

1) 개념
- 어떤 분산 환경에서도 일관성(C), 가용성(A), 분단 허용성(P) 세 가지 속성 중, 두 가지만 가질 수 있다는 것

2) 특징의 의미
- 일관성(Consistency)
- 가용성(Availability)
- 분단 허용성(Partition Tolerance)

2. 트랜잭션

⑴ 트랜잭션의 개념
- 데이터베이스의 상태를 변환시키는 하나의 논리적인 기능을 수행하는 작업 단위

⑵ 트랜잭션의 성질
- 원자성(Atomicity) : 모두 반영되든지 아니면 전혀 반영되지 않아야 한다.(Commit과 Rollback)
- 일관성(Consistency) : 실행을 완료하면 언제나 일관성 있는 데이터베이스 상태로 변환
- 독립성, 격리성(Isolation) : 하나의 트랜잭션 실행 중에 다른 트랜잭션의 연산이 끼어들 수 없다.
- 영속성(Durability) : 트랜잭션의 결과는 시스템이 고장이 나더라도 영구적으로 반영되어야 한다.

⑶ 트랜잭션의 상태
- 활동(Active) : 트랜잭션이 실행 중인 상태
- 실패(Failed) : 오류가 발생하여 중단된 상태
- 철회(Aborted) : Rollback 연산을 수행한 상태
- 부분 완료(Partially Committed) : Commit 연산이 실행되기 직전의 상태
- 완료(Committed) : Commit 연산을 실행한 후의 상태

02 SQL 활용

Section 1. 기본 SQL 작성

1. SQL(Structured Query Language)

⑴ SQL의 개념

- 데이터베이스 시스템에서 자료를 처리하는 용도로 사용되는 구조적 데이터 질의 언어

⑵ SQL 문법의 종류

1) Data Definition Language (DDL) - 데이터 정의어
   - CREATE, ALTER, DROP, RENAME, TRUNCATE
2) Data Manipulation Language (DML) - 데이터 조작어
   - SELECT, INSERT, UPDATE, DELETE
3) Data Control Language (DCL) - 데이터 제어어
   - GRANT, REVOKE
4) Transaction Control Language (TCL) - 트랜잭션 제어어
   - COMMIT, ROLLBACK, SAVEPOINT

Section 2. 절차형 SQL

1. 저장 프로시저(Stored Procedure)

⑴ 저장 프로시저의 개념

- 일련의 쿼리를 마치 하나의 함수처럼 실행하기 위한 쿼리의 집합

⑵ 저장 프로시저의 구조

```
CREATE OR REPLACE PROCEDURE 프로시저명
( 변수1 IN 변수타입, 변수2 OUT 변수타입, 변수3 IN OUT 변수타입.... )
IS
    변수 처리부
BEGIN
    처리내용
EXCEPTION
    예외처리부
END;
```

2. 트리거

⑴ 트리거의 개념

- 테이블에 대한 이벤트에 반응해 자동으로 실행되는 작업

⑵ 트리거의 유형

- 행 트리거 : FOR EACH ROW 옵션 사용
- 문장 트리거 : INSERT, UPDATE, DELETE문에 대해 단 한 번만 실행

⑶ 트리거의 실행 시기

- BEFORE : 이벤트 전
- AFTER : 이벤트 후

3. 사용자 정의 함수

⑴ 사용자 정의 함수의 개념

- 파라미터는 입력 파라미터만 가능하고, 리턴값이 하나이다.

⑵ 사용자 정의 함수의 구조

```
CREATE OR REPLACE FUNCTION 함수명
( 매개변수1, 매개변수2, 매개변수3,..... )
RETURN 데이터 타입
IS
    변수 처리부
BEGIN
    처리내용
    RETURN 반환값;
EXCEPTION
    예외처리부
END;
```

03 병행제어와 데이터전환

Section 1. 병행제어와 회복

1. 병행제어

(1) 병행제어를 하지 않았을 때의 문제점

1) 갱신 분실(Lost Update)

- 두 개 이상의 트랜잭션이 같은 자료를 공유하여 갱신할 때 갱신 결과의 일부가 없어지는 현상

데이터	트랜잭션 1	트랜잭션 2
1000	READ(1000)	
1000		READ(1000)
1000	ADD(1000)	
1000		ADD(2000)
2000	STORE(2000)	
3000		STORE(3000)

2) 비완료 의존성(Uncommitted Dependency)

- 하나의 트랜잭션 수행이 실패한 후 회복되기 전에 다른 트랜잭션이 실패한 갱신 결과를 참조하는 현상

데이터	트랜잭션 1	트랜잭션 2
1000	READ(1000)	
1000	ADD(1000)	
2000	STORE(2000)	READ(2000)
1000	장애로 ROLLBACK	

3) 모순성(Inconsistency)

- 두 개의 트랜잭션이 병행수행될 때 원치 않는 자료를 이용함으로써 발생하는 문제
- 갱신 분실과 비슷해 보이지만 여러 데이터를 가져올 때 발생하는 문제

4) 연쇄 복귀(Cascading Rollback)

- 병행수행된 트랜잭션들 중 어느 하나에 문제가 생겨 Rollback하는 경우 다른 트랜잭션도 함께 Rollback되는 현상

데이터	트랜잭션 1	트랜잭션 2
1000	READ(1000)	
1000	ADD(1000)	
2000	STORE(2000)	
2000		READ(2000)
2000		ADD(2000)
4000		STORE(4000)
1000	장애로 ROLLBACK	

⑵ 병행제어 기법

1) 로킹(Locking)

- 트랜잭션이 어떤 데이터에 접근하고자 할 때 로킹 수행
- 로킹 단위에 따른 구분

구분	로크 수	병행성	오버헤드
로킹 단위가 크면	적어짐	낮아짐	감소
로킹 단위가 작으면	많아짐	높아짐	증가

2) 2단계 로킹 규약(Two-Phase Locking Protocol)

- 확장단계 : 새로운 Lock은 가능하고 Unlock은 불가능하다.
- 축소단계 : Unlock은 가능하고 새로운 Lock은 불가능하다.

3) 타임스탬프(Time Stamp)

- 데이터에 접근하는 시간을 미리 정해서 정해진 시간(Time Stamp)의 순서대로 데이터에 접근하여 수행

4) 낙관적 병행제어(Optimistic Concurrency Control)

- 트랜잭션 수행 중에는 어떠한 검사도 하지 않고, 트랜잭션 종료 시에 일괄적으로 검사

5) 다중 버전 병행제어(Multi-version, Concurrency Control)

- 여러 버전의 타임스탬프를 비교하여 스케줄상 직렬가능성이 보장되는 타임스탬프를 선택

2. 회복(Database Recovery)

⑴ 로그 기반 회복 기법

1) 지연갱신 회복 기법(Deferred Update)

- 커밋이 발생하기 전까진 데이터베이스에 기록하지 않음
- 중간에 장애가 생기더라도 데이터베이스에 기록되지 않았으므로 UNDO가 필요 없음(미실행된 로그 폐기)

2) 즉시갱신 회복 기법(Immediate Update)

- 트랜잭션 수행 도중에도 변경 내용을 즉시 데이터베이스에 기록
- 커밋 발생 이전의 갱신은 원자성이 보장되지 않는 미완료 갱신이므로 장애 발생 시 UNDO 필요

⑵ 검사점 회복 기법(Checkpoint Recovery)

- 장애 발생 시 검사점(Checkpoint) 이전에 처리된 트랜잭션은 회복에서 제외하고 검사점 이후에 처리된 트랜잭션은 회복 작업 수행

⑶ 그림자 페이징 회복 기법(Shadow Paging Recovery)

- 트랜잭션이 실행되는 메모리상의 Current Page Table과 하드디스크의 Shadow Page Table 이용

⑷ 미디어 회복 기법(Media Recovery)

- 디스크와 같은 비휘발성 저장 장치가 손상되는 장애 발생을 대비한 회복 기법

⑸ ARIES 회복 기법(Algorithms for Recovery and Isolation Exploiting Semantics)

- 분석단계 → REDO 단계 → UNDO 단계

Section 2. 데이터 전환

1. ETL(Extraction, Transformation, Loading)

⑴ ETL 개념

- 기존의 원천 시스템에서 데이터를 추출(Extraction)하여 목적 시스템의 데이터베이스에 적합한 형식과 내용으로 변환(Transformation)한 후, 목적 시스템에 적재(Loading)하는 일련의 과정

⑵ ETL 기능

- 추출(Extraction) : 하나 또는 그 이상의 데이터 소스로부터 데이터 획득
- 변환(Transformation) : 데이터 클렌징, 형식 변환 및 표준화, 데이터 통합
- 적재(Load) : 변형 단계의 처리가 완료된 데이터를 목표 시스템에 적재