모델이 표현하는 트랜잭션의 이해 개념서

이 문서는 데이터베이스 트랜잭션의 개념, 특성, 모델링 방법, 무결성 제약조건, 격리 수준, 그리고 실제 사례에 대해 상세히 설명합니다. ACID 특성, 트랜잭션 식별 및 패턴, 엔터티 및 관계 수준의 트랜잭션, 무결성 제약조건, 격리 수준, 그리고 주문 처리 및 계좌 이체 시스템의 트랜잭션 모델링 사례를 다룹니다. 또한 트랜잭션 성능 고려사항과 SQLD 시험 대비 팁을 제공합니다.

1. 트랜잭션의 기본 개념

1.1 정의

- 데이터베이스의 논리적 작업 단위
- 하나의 작업을 수행하기 위한 데이터 조작의 단위
- 데이터의 일관성을 보장하는 메커니즘

1.2 ACID 특성

1. **원자성 (Atomicity)**
   - 트랜잭션은 모두 실행되거나 전혀 실행되지 않아야 함
   - All or Nothing
2. **일관성 (Consistency)**
   - 트랜잭션 실행 전후의 데이터베이스 상태가 일관적이어야 함
   - 무결성 제약조건 유지
3. **격리성 (Isolation)**
   - 동시에 실행되는 트랜잭션들이 서로 영향을 미치지 않음
   - 독립적인 실행 보장
4. **지속성 (Durability)**
   - 성공적으로 완료된 트랜잭션의 결과는 영구적으로 보존
   - 시스템 장애 발생 시에도 보존

2. 데이터 모델에서의 트랜잭션

2.1 트랜잭션 식별

1. **업무 단위 분석**
   - 하나의 논리적 업무 단위 파악
   - 연관된 데이터들의 관계 분석
2. **데이터 변경 빈도**
   - INSERT, UPDATE, DELETE 연산 빈도
   - 영향을 받는 데이터량 파악

2.2 트랜잭션 패턴

```
-- 주문 처리 트랜잭션
BEGIN TRANSACTION;
  INSERT INTO 주문 VALUES (...);
  INSERT INTO 주문상세 VALUES (...);
  UPDATE 재고 SET 수량 = 수량 - 주문수량;
COMMIT;

-- 이체 처리 트랜잭션
BEGIN TRANSACTION;
  UPDATE 계좌 SET 잔액 = 잔액 - 이체금액 WHERE 계좌번호 = 출금계좌;
  UPDATE 계좌 SET 잔액 = 잔액 + 이체금액 WHERE 계좌번호 = 입금계좌;
COMMIT;
```

3. 트랜잭션 모델링

3.1 엔터티 수준의 트랜잭션

```
-- 단일 엔터티 트랜잭션
CREATE TABLE 고객 (
  고객번호 NUMBER PRIMARY KEY,
  고객명 VARCHAR2(100),
  인증등급 CHAR(1)
);

-- 트랜잭션 예시
UPDATE 고객
SET 인증등급 = 'A'
WHERE 고객번호 = '고객번호';
```

3.2 관계 수준의 트랜잭션

```
-- 관계를 포함한 트랜잭션
CREATE TABLE 주문 (
  주문번호 NUMBER PRIMARY KEY,
  고객번호 NUMBER REFERENCES 고객(고객번호),
  주문일자 DATE
);

CREATE TABLE 주문상세 (
  주문번호 NUMBER REFERENCES 주문(주문번호),
  상품코드 NUMBER,
  수량 NUMBER,
  PRIMARY KEY (주문번호, 상품코드)
);

-- 트랜잭션 예시
INSERT INTO 주문 VALUES (...);
INSERT INTO 주문상세 VALUES (...);
```

4. 트랜잭션 무결성 제약조건

4.1 참조 무결성

```
-- 외래키 제약조건
CREATE TABLE 주문 (
  주문번호 NUMBER PRIMARY KEY,
  고객번호 NUMBER,
  FOREIGN KEY (고객번호)
  REFERENCES 고객(고객번호)
  ON DELETE CASCADE
);
```

4.2 업무 규칙 무결성

```
-- CHECK 제약조건
CREATE TABLE 계좌 (
  계좌번호 VARCHAR2(20) PRIMARY KEY,
  잔액 NUMBER CHECK (잔액 >= 0),
  계좌상태 CHAR(1) CHECK (계좌상태 IN ('1','2','3'))
);
```

5. 트랜잭션 격리 수준

5.1 격리 수준의 종류

- **READ UNCOMMITTED**

```
-- 다른 트랜잭션의 미확정 데이터 읽기 가능
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
```

- **READ COMMITTED**

```
-- 확정된 데이터만 읽기 가능
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

- **REPEATABLE READ**

```
-- 트랜잭션 동안 동일한 결과 보장
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
```

- **SERIALIZABLE**

```
-- 완벽한 읽기 일관성 보장
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
```

6. 트랜잭션 모델링 사례

6.1 주문 처리 시스템

```
-- 주문 처리 트랜잭션 모델
CREATE TABLE 주문 (
  주문번호 NUMBER PRIMARY KEY,
  주문일자 DATE,
  고객번호 NUMBER,
  주문상태 CHAR(1)
);

CREATE TABLE 주문상세 (
  주문번호 NUMBER,
  상품코드 NUMBER,
  수량 NUMBER,
  단가 NUMBER,
  PRIMARY KEY (주문번호, 상품코드)
);

CREATE TABLE 재고 (
  상품코드 NUMBER PRIMARY KEY,
  재고수량 NUMBER
);

-- 트랜잭션 처리
BEGIN TRANSACTION;
  INSERT INTO 주문 VALUES (...);
  INSERT INTO 주문상세 VALUES (...);
  UPDATE 재고 SET 재고수량 = 재고수량 - 주문수량;
COMMIT;
```

6.2 계좌 이체 시스템

```
-- 계좌 이체 트랜잭션 모델
CREATE TABLE 계좌 (
  계좌번호 VARCHAR2(20) PRIMARY KEY,
  고객번호 NUMBER,
  잔액 NUMBER CHECK (잔액 >= 0)
);

CREATE TABLE 거래내역 (
  거래번호 NUMBER PRIMARY KEY,
  계좌번호 VARCHAR2(20),
  거래구분 CHAR(1),
  거래금액 NUMBER,
  거래일시 TIMESTAMP
);

-- 트랜잭션 처리
BEGIN TRANSACTION;
  UPDATE 계좌 SET 잔액 = 잔액 - 이체금액
  WHERE 계좌번호 = 출금계좌;

  INSERT INTO 거래내역 VALUES (...출금내역...);

  UPDATE 계좌 SET 잔액 = 잔액 + 이체금액
  WHERE 계좌번호 = 입금계좌;

  INSERT INTO 거래내역 VALUES (...입금내역...);
COMMIT;
```

7. 트랜잭션 성능 고려사항

7.1 트랜잭션 범위

```
-- 적절한 트랜잭션 범위
BEGIN TRANSACTION;
  -- 필수적인 작업만 포함
  INSERT/UPDATE/DELETE...
COMMIT;

-- 부적절한 트랜잭션 범위
BEGIN TRANSACTION;
  -- 불필요한 조회 포함
  SELECT...
  INSERT/UPDATE/DELETE...
  SELECT...
COMMIT;
```

7.2 동시성 제어

```
-- 락(Lock) 최소화
SELECT ... FOR UPDATE NOWAIT;
-- 데드락(Deadlock) 방지
SELECT ... FOR UPDATE WAIT 3;
```

SQLD 시험 대비 TIP

주요 출제 포인트

1. **트랜잭션의 개념**
   - ACID 특성
   - 트랜잭션 범위
2. **트랜잭션 모델링**
   - 업무 규칙 파악
   - 무결성 제약조건
3. **격리 수준**
   - 각 수준의 특징
   - 발생 가능한 문제

학습 전략

1. 트랜잭션의 기본 개념 이해
2. 다양한 모델링 사례 학습
3. 무결성 제약조건 이해
4. 실제 업무 사례 분석

실전 문제 유형

1. 트랜잭션 범위 설정
2. 무결성 제약조건 설계
3. 격리 수준 선택
4. 성능 최적화 방안