TCL(Transaction Control Language) 개념서

이 문서는 트랜잭션 제어 언어(TCL)에 대한 포괄적인 가이드입니다. TCL의 기본 개념, 주요 명령어, 트랜잭션의 특성, 실습 예제, 격리 수준, 동시성 제어 문제, 락과 교착상태, 그리고 성능 고려사항을 다룹니다.

1. TCL의 기본 개념

1.1 정의

- 트랜잭션 처리를 제어하는 언어
- 데이터의 일관성을 유지하고 안정적인 데이터 처리를 보장
- COMMIT, ROLLBACK, SAVEPOINT가 대표적

1.2 트랜잭션의 특성 (ACID)

1. **원자성 (Atomicity)**
   - 트랜잭션은 모두 실행되거나 전혀 실행되지 않아야 함
2. **일관성 (Consistency)**
   - 트랜잭션 전후 전체의 데이터베이스는 일관된 상태를 유지
3. **격리성 (Isolation)**
   - 동시에 실행되는 트랜잭션들은 서로 영향을 미치지 않음
4. **지속성 (Durability)**
   - 성공적으로 완료된 트랜잭션의 결과는 영구적으로 보장

2. TCL 명령어

2.1 COMMIT

```
-- 기본 사용
COMMIT;

-- 명시적 커밋이 발생하는 경우
- DDL 실행 시
- DCL 실행 시
- 정상적인 데이터베이스 접속 종료 시
```

2.2 ROLLBACK

```
-- 전체 롤백
ROLLBACK;

-- 특정 저장점까지 롤백
ROLLBACK TO SAVEPOINT 저장점이름;
```

2.3 SAVEPOINT

```
-- 저장점 생성
SAVEPOINT 저장점이름;

-- 저장점 사용 예
SAVEPOINT S1;
[DML 작업1]
SAVEPOINT S2;
[DML 작업2]
ROLLBACK TO S1; -- S1 저장점으로 롤백
```

3. 트랜잭션 제어 실습

3.1 기본 트랜잭션 처리

```
-- 트랜잭션 시작
INSERT INTO 계좌 VALUES (1001, '홍길동', 1000);
UPDATE 계좌 SET 잔액 = 잔액 - 500 WHERE 계좌번호 = 1001;
INSERT INTO 거래내역 VALUES (1001, '출금', 500);
COMMIT; -- 트랜잭션 완료

-- 오류 발생 시
ROLLBACK; -- 트랜잭션 취소
```

3.2 SAVEPOINT 활용

```
-- 여러 개의 저장점 활용
INSERT INTO 직원 VALUES (1, '김철수');
SAVEPOINT S1;

UPDATE 직원 SET 이름 = '김영희' WHERE 사번 = 1;
SAVEPOINT S2;

DELETE FROM 직원 WHERE 사번 = 1;
ROLLBACK TO S2; -- DELETE만 취소됨
```

4. 트랜잭션 격리 수준

4.1 격리 수준의 종류

1. **READ UNCOMMITTED**
   - 다른 트랜잭션의 커밋되지 않은 데이터 읽기 가능
   - Dirty Read 발생 가능
2. **READ COMMITTED**
   - 커밋된 데이터만 읽기 가능
   - Oracle의 기본 격리 수준
3. **REPEATABLE READ**
   - 트랜잭션 동안 동일한 결과 보장
   - MySQL의 기본 격리 수준
4. **SERIALIZABLE**
   - 가장 높은 격리 수준
   - 완벽한 데이터 일관성 보장

4.2 격리 수준 설정

```
-- Oracle
ALTER SESSION SET ISOLATION_LEVEL = SERIALIZABLE;

-- MySQL
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
```

5. 동시성 제어 문제

5.1 주요 문제점

1. **Dirty Read**
   - 커밋되지 않은 데이터를 읽는 현상

```
-- 트랜잭션 1
UPDATE 계좌 SET 잔액 = 1000;

-- 트랜잭션 2가 변경된 데이터를 읽음
ROLLBACK;
```

2. **Non-Repeatable Read**
   - 동일 트랜잭션 내에서 같은 데이터를 두 번 읽을 때 값이 다른 현상

```
-- 트랜잭션 1
SELECT 잔액 FROM 계좌; -- 1000원

-- 트랜잭션 2가 UPDATE 수행
SELECT 잔액 FROM 계좌; -- 2000원
```

3. **Phantom Read**
   - 동일 트랜잭션 내에서 조회 결과가 달라지는 현상

```
-- 트랜잭션 1
SELECT * FROM 계좌; -- 10개 행

-- 트랜잭션 2가 INSERT 수행
SELECT * FROM 계좌; -- 11개 행
```

6. 락(Lock)과 교착상태(Deadlock)

6.1 락의 종류

1. **공유 락 (Shared Lock)**
   - 읽기 작업 시 사용
   - 다른 공유 락과 호환됨
2. **배타적 락 (Exclusive Lock)**
   - 쓰기 작업 시 사용
   - 다른 모든 락과 호환되지 않음

6.2 교착상태 처리

```
-- 교착상태 예방
SET LOCK_TIMEOUT 10000; -- 10초 후 자동 롤백

-- 교착상태 감지
SELECT blocking_session, sid, serial#
FROM v$session
WHERE blocking_session IS NOT NULL;
```

7. 성능 고려사항

7.1 트랜잭션 설계

1. **트랜잭션 범위**
   - 최소한의 작업만 포함
   - 불필요한 락 경합 방지
2. **커밋 주기**
   - 대량 작업 시 적절한 커밋 주기 설정
   - 메모리 사용량 고려

7.2 성능 최적화

```
-- 일괄 처리 예제
DECLARE
  v_count NUMBER := 0;
BEGIN
  FOR r IN (SELECT * FROM 테이블)
  LOOP
    -- 처리 로직
    v_count := v_count + 1;
    IF v_count MOD 1000 = 0 THEN
      COMMIT; -- 1000건마다 커밋
    END IF;
  END LOOP;
  COMMIT;
END;
```

SQLD 시험 대비 TIP

주요 출제 포인트

1. **트랜잭션의 특성**
   - ACID 특성
   - 격리 수준
2. **TCL 명령어**
   - COMMIT, ROLLBACK, SAVEPOINT 사용
   - 암시적 커밋 상황
3. **동시성 제어**
   - 락의 종류와 특징
   - 교착상태 해결

학습 전략

1. 트랜잭션 기본 개념 이해
2. TCL 명령어 사용법 숙지
3. 격리 수준별 특징 파악
4. 동시성 문제 이해

실전 문제 유형

1. 트랜잭션 결과 예측
2. 격리 수준에 따른 동작 차이
3. 교착상태 해결 방안
4. 성능 최적화 방안