16강

트랜잭션(Transaction)

트랜잭션(Transaction) 이란 ?

➢ 트랜잭션(Transaction)

- 트랜잭션(Transaction)의 사전적 의미는 거래이고, 컴퓨터 과학 분야에서의 트랜잭션(Transaction)은 **"더이상 분할이 불가능한 업무처리의 단위"**를 의미한다.
- 이것은 하나의 작업을 위해 더이상 분할될 수 없는 명령들의 모음, 즉, 한꺼번에 **수행되어야 할 일련의 연산모음**을 의미한다.

➢ 트랜잭션(Transaction) 예

① A는 매달 부모님에게 생활비를 송금 받는다.
어느 날, 부모님이 A에게 생활비를 송금해 주기 위해 ATM을 이용했고 여느 날 처럼 A의 계좌로 생활비를 송금했다.

② 그러나 모종의 이유로 인하여 부모님의 계좌에선 생활비가 차감되었는데, A의 계좌에는 생활비가 입금되지 않았다.

1. 위의 예는 계좌이체 행위를 풀어서 설명한 것이다.
2. 계좌이체 라는 행위는 **인출**과 **입금** 두 과정으로 이루어진다.
3. 위와 같이 만약 인출에는 성공했는데, 입금에 실패하면 치명적인 결과가 나온다.
4. 따라서 **이 두 과정은 동시에 성공하던지 동시에 실패**해야 한다. (하나로 묶음으로써 Atomic(원자)함을 의미한다)
5. 이 과정을 동시에 묶는 방법이 바로 트랜잭션이다.

트랜잭션(Transaction) 작성 방법

➢ 트랜잭션 작성 방법

```
START TRANSACTION
    -- 이 블록안의 명령어들은 마치 하나의 명령어 처럼 처리됨
    -- 성공하던지, 다 실패하던지 둘중 하나가 됨.
    A의 계좌로부터 인출;
    B의 계좌로 입금;
COMMIT;
```

```
rollback; -- 트랜잭션을 취소하고 start transaction 이전으로 되돌림
```

Tip

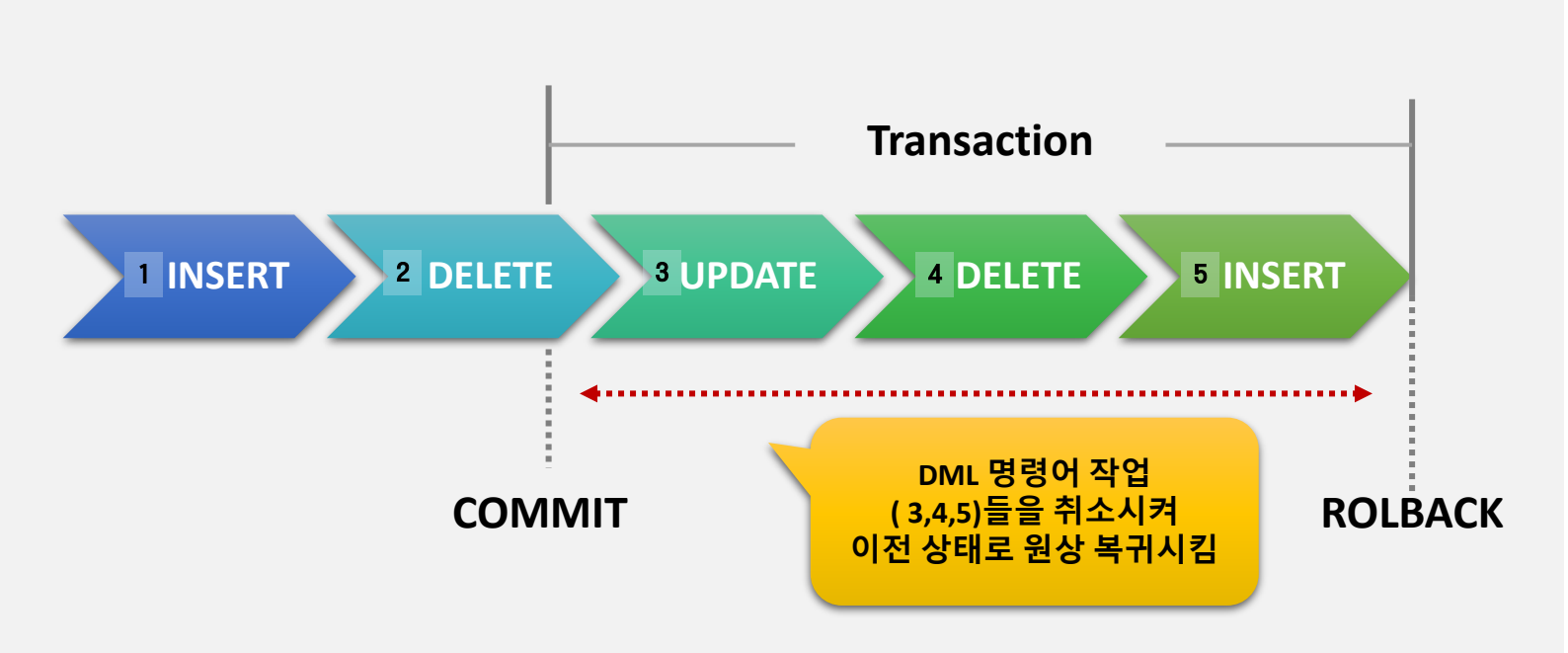
- 데이터베이스와 어플리케이션의 **데이터 거래(Transaction)에 있어서 안전성을 확보하기 위한 방법**이 **트랜잭션**인 것이다.
- 따라서 데이터베이스에서 테이블의 데이터를 읽어 온 후 다른 테이블에 데이터를 입력하거나 갱신, 삭제하는 도중에 오류가 발생하면, 결과를 재 반영 하는 것이 아니라 모든 작업을 원상태로 복구하고, 처리 과정이 모두 성공하였을때만 그 결과를 반영한다.

트랜잭션 예외

- 모든 명령어에 대하여 트랜잭션의 롤백 명령이 적용되는 것은 아니다.
- **DDL문(CREATE, DROP, ALTER, RENAME, TRUNCATE)**은 transaction의 rollback 대상이 아니다

MySQL 트랜잭션

- MySQL에서 트랜잭션은 **데이터베이스를 상태를 바꾸는 일종의 작업 단위**이다.
- 사실 우리가 MySQL의 입력하는 모든 쿼리 명령어들은 각각 하나의 트랜잭션이라고 할 수 있다.
- **INSERT, DELETE, UPDATE** 등의 SQL 명령문을 통해 데이터의 상태를 바꿀 때마다 내부적으로 자동적으로 **Commit**을 실행하여 변경된 내역을 데이터베이스의 반영하는 것이다.
- 즉 여태까지 입력한 명령어들은 MySQL에서 자동 Commit을 통해 쿼리 입력과 동시에 처리하여 데이터베이스에 반영되게 한 것이다.

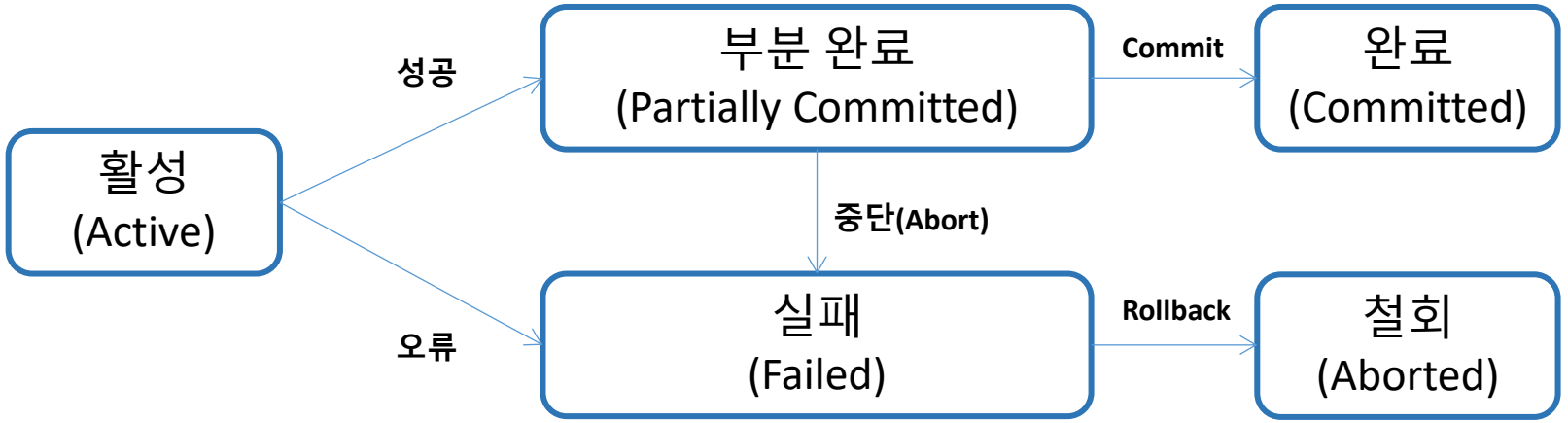


트랜잭션(Transaction) 상태

- 트랜잭션의 개념은 서술한 바와 같이 데이터베이스의 상태를 변환시키는 하나의 논리적 기능을 수행하기 위한 작업의 단위 혹은 데이터베이스 시스템에서 복구 및 병행 수행 시 처리되는 작업의 논리적 단위이다.

트랜잭션(Transaction) 도식화

- 트랜잭션의 연산 과정의 단계를 도식화 하고 각각의 단계의 상태를 아래 그림과 같이 정리할 수 있다



Commit 과 Roll-back 설명

Tip

Commit

Commit이란, 모든 작업들을 정상 처리하겠다고 확정하는 명령어로서, 해당 처리 **과정을 DB에 영구 저장** 하겠다는 의미이며, Commit을 수행하면 **하나의 트랜잭션 과정이 종료**되는 것이다.

Commit을 수행하면 이전 데이터가 완전히 반영되어 **UPDATE**된다.

Tip

Roll-back

Roll-back 은 작업 중 문제가 발생되어 트랜잭션의 처리 과정에서 **발생한 변경사항을 취소**하는 명령어이다.

해당 명령을 트랜잭션에게 하달하면, 트랜잭션 **시작되기 이전의 상태로 되돌아 간다.**

이것은 마지막 Commit을 완료한 시점으로 돌아간다는 말과 상통한다.

즉, Rollback 은 **Commit 하여 저장한 예전 상태를 복구**하는 것이다.

➢ 트랜잭션(Transaction) 문법

- 명령어들은 MySQL에서 자동 Commit을 통해 쿼리 입력과 동시에 처리하여 데이터베이스에 반영된다.
- 그렇다면 **Commit을 수동으로 바꿔서 한 트랜잭션 안에 여러 개의 명령문을 넣고 실행** 해보자

```
START TRANSACTION; -- 트랜잭션 시작

SELECT * FROM sections; -- 초기상태 보여줌

DELETE FROM sections
WHERE section_id > 0; -- 수정상태 보여줌

SELECT * FROM sections; -- 적용된 결과 조회
```

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1

section_id	section_name	floor
NULL	NULL	NULL

```
-- 롤백
ROLLBACK;

SELECT * FROM sections;
```

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1

트랜잭션(Transaction) 문법 - COMMIT

```
START TRANSACTION; -- 트랜잭션 시작

SELECT * FROM sections; -- 초기상태 보여줌

DELETE FROM sections
WHERE section_id > 0; -- 수정상태 보여줌

COMMIT; -- 트랜잭션을 DB에 적용

SELECT * FROM sections; -- 적용된 결과 조회
```

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1

section_id	section_name	floor
NULL	NULL	NULL

Commit 명령어로 DB에 적용이 되었기때문에 Rollback 을 실행해도 Transaction 실행 전 상태로 되돌아 가지 않음을 주의하자.

```
-- 롤백
ROLLBACK; -- 트랜잭션을 취소하고 START TRANSACTION 실행 전 상태로 롤백함

SELECT * FROM sections;
```

section_id	section_name	floor
NULL	NULL	NULL

☞ 여기서 잠깐 !!

greendb 스키마안에 <businesses>테이블의 status 컬럼이 'OPN'인 데이터를 모두 삭제하고, 삭제 하고 난 후 Rollback하면 status 컬럼의 'OPN' 데이터를 삭제하기 이전으로 돌아가도록 START TRANSACTION 명령어를 이용하여 <출력결과>와 같이 출력되도록 SQL문을 작성하시오.

(단, Start Transaction 명령문 이용)

OPN 삭제 후 결과

business_id	fk_section_id	business_name	status	can_takeout
3	5	얄코렐라	RMD	1
7	5	에그사라다	VCT	1
12	4	사사서셔소쇼스시	VCT	1
13	3	린민짬뽕	CLS	1
15	1	할매장국	CLS	0
18	2	신림동백순대	VCT	1

Rollback

Rollback 후 결과

business_id	fk_section_id	business_name	status	can_takeout
1	3	화룡각	OPN	1
2	2	철구분식	OPN	1
3	5	얄코렐라	RMD	1
4	2	바른떡볶이	OPN	1
5	1	북극냉면	OPN	0
6	1	보쌈마니아	OPN	1
7	5	에그사라다	VCT	1
8	6	달다방	OPN	1
9	7	마카오마카롱	OPN	1
10	2	김밥마라	OPN	1
11	7	소소스윗	OPN	1
12	4	사사서셔소쇼스시	VCT	1
13	3	린민짬뽕	CLS	1
14	7	파시조아	OPN	1
15	1	할매장국	CLS	0
16	5	노션이탈리아	OPN	1
17	6	커피앤코드	OPN	1
18	2	신림동백순대	VCT	1

코드로 확인하는 Transaction

코드 예제

```
try{
    conn = DriverManager.getConnection(...);
    // 트랜잭션 시작
    conn.setAutoCommit(false);

    ... // 쿼리 실행
    ... // 쿼리 실행

    // 트랜잭션 커밋
    conn.commit();
}catch(SQLException ex){
    if(conn != null){
        // 트랜잭션 롤백
        conn.rollback();
    }
}finally{
    if(conn != null){
        try{
            conn.close();
        }catch(SQLException ex){}
    }
}
```

트랜잭션(Transaction) 문법 - SAVEPOINT

➢ SAVEPOINT

- **Rollback** 할 중간지점 설정

```
START TRANSACTION;

INSERT INTO sections
(section_name, floor)
VALUES ('인도식', 2);

SAVEPOINT indian;

INSERT INTO sections
(section_name, floor)
VALUES ('남미식', 3);

SELECT * FROM sections;

ROLLBACK TO indian;

SELECT * FROM sections;

COMMIT;
```

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1
8	인도식	2

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1
8	인도식	2
9	남미식	3

Rollback To indian을 실행하면 Rollback 할 중간 지점인 savepoint indian으로 되돌아 감을 주의하자.

section_id	section_name	floor
1	한식	2
2	분식	2
3	중식	3
4	일식	3
5	양식	3
6	카페	1
7	디저트	1
8	인도식	2