트랜잭션

Transaction

2025.07.16.

System Software Lab.

목차

데이터베이스 시간에는…

15장 회복 (1/2)

각종 시험에 잘 나옴

- Transaction: 하나의 논리적 기능을 수행하기 위한 작업의 단위
- Transaction의 ACID 성질: Atomicity, consistency, isolation, durability
  - Atomicity (원자성): 일부만 실행되지 않고 전부 실행되거나 또는 아무것도 실행되지 않아야 함
  - Consistency (일관성): 실행 후에도 데이터베이스는 일관성이 있음 (모순이 없음)
  - Isolation (격리성): 트랜잭션 실행 중에 있는 연산의 중간 결과는 다른 트랜잭션이 접근할 수 없음
  - Durability (영속성): 트랜잭션의 결과는 영속적임
- Transaction 명령
  - Begin (시작)
  - Commit (완료)
  - Rollback (복귀)

01

트랜잭션

은행 송금으로 알아보는 트랜잭션

시나리오

통장 잔고가 오만 원인 가군이,

통장 잔고가 십만 원인 나양에게

만 원을 송금한다

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다

가군

₩ 50,000

나양

₩ 100,000

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다

가군

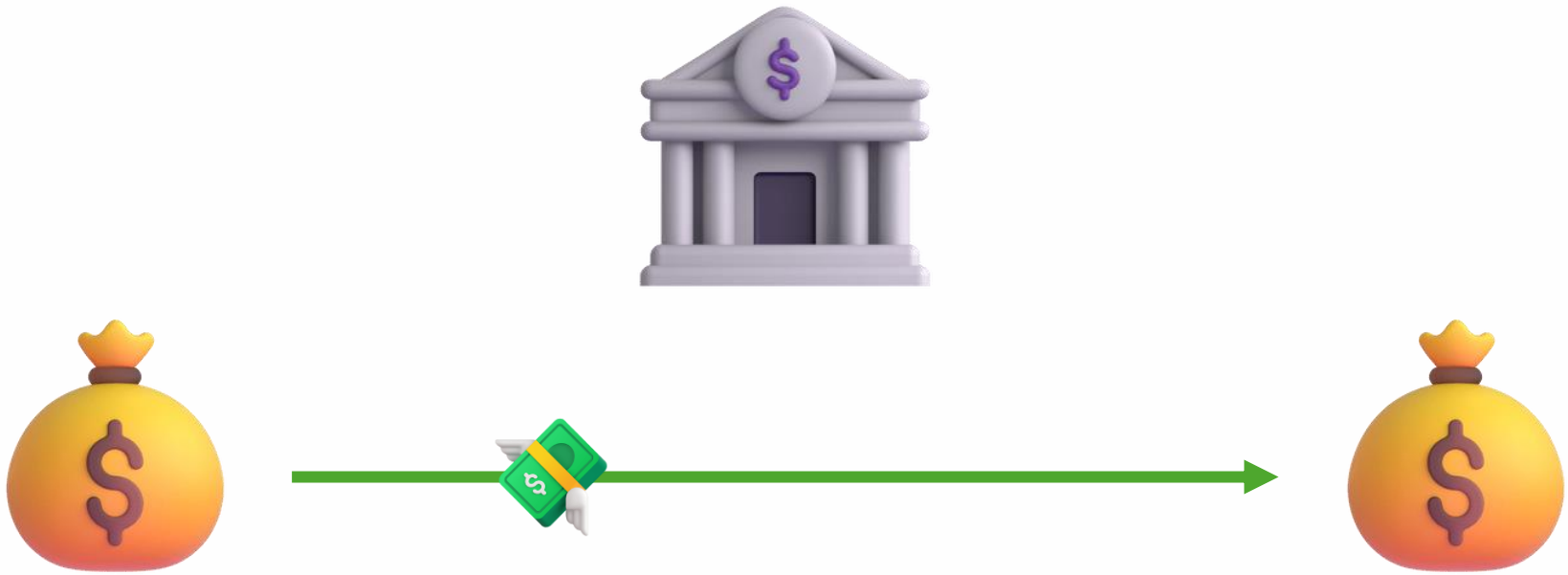
₩ 40,000

나양

₩ 110,000

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다



가군

₩ 50,000
- 10,000
₩ 40,000

나양

₩ 100,000

```
UPDATE accounts
SET money = money – 10000
WHERE username = '가군';
```

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다

가군

₩ 40,000

```
UPDATE accounts
SET money = money + 10000
WHERE username =  '나양';
```

나양

₩ 100,000
+ 10,000
₩ 110,000

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다

가군

₩ 40,000

나양

₩ 110,000

은행 송금으로 알아보는 트랜잭션

통장 잔고가 오만 원인 가군이 통장 잔고가 십만 원인 나양에게 만 원을 송금한다

가군

₩ 50,000

- 10,000

₩ 40,000

가군 잔고 차감 후 오류가 발생하면?

가군의 잔고는 차감되고, 나양의 잔고는 유지됨

나양

₩ 100,000

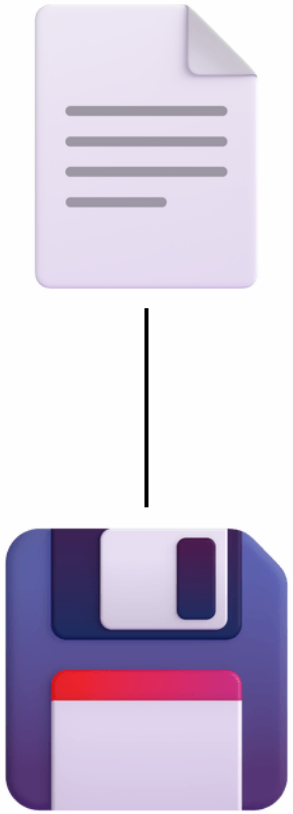
게시글 작성으로 알아보는 트랜잭션

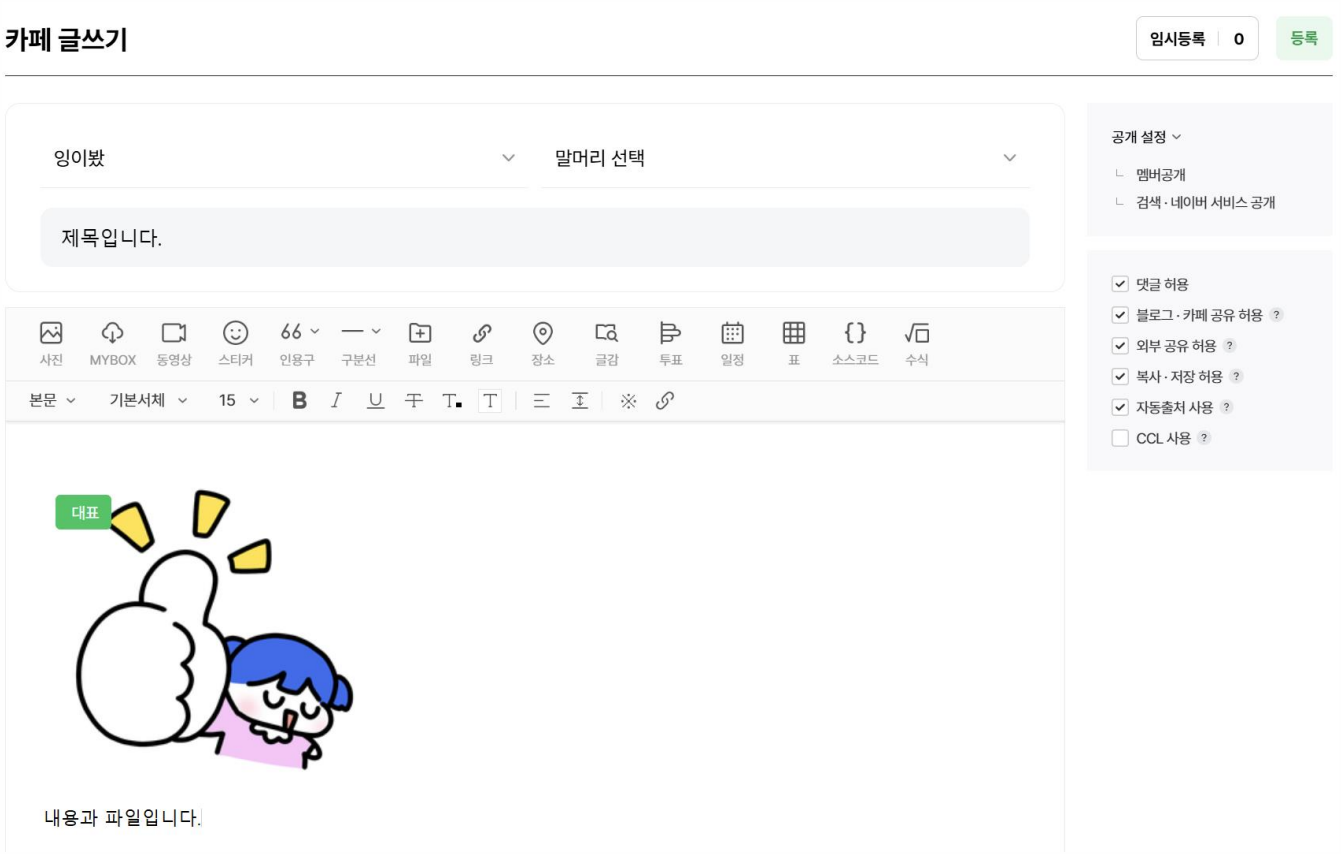
시나리오

게시글 제목과 본문을 저장한 후,

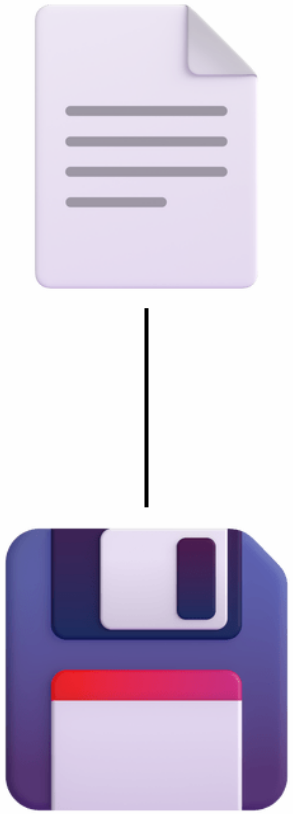
첨부 파일 경로를 저장한다.

게시글 작성으로 알아보는 트랜잭션





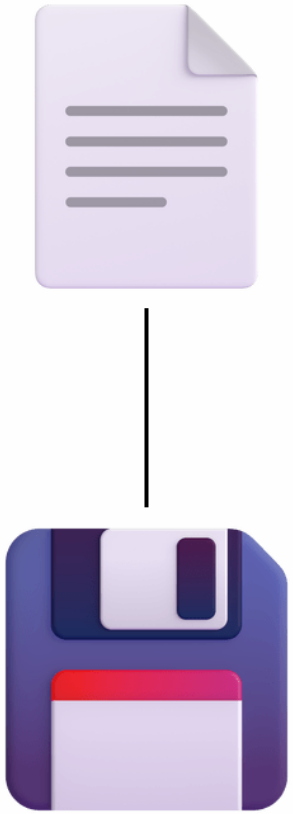
게시글 작성으로 알아보는 트랜잭션



```sql
INSERT INTO documents (title, content)
VALUES ('제목입니다.', '내용과 파일입니다.');
```

```sql
INSERT INTO files (document_id, path)
VALUES (1, '/images/ing.png');
```

게시글 작성으로 알아보는 트랜잭션



```sql
INSERT INTO documents (title, content)
VALUES ('제목입니다.', '내용과 파일입니다.');
```

게시글 등록 중 오류가 발생하면?

```sql
INSERT INTO files (document_id, path)
VALUES (1, '/images/ing.png');
```

트랜잭션

하나의 논리적 기능을 수행하기 위한 하나의 작업 단위

여러 쿼리를 논리적으로 하나의 작업 단위로 묶는 행위

더 이상 나눌 수 없는 가장 작은 하나의 단위

데이터베이스에서 수행되는 여러 작업을 하나의 논리적 단위로 수행하는 행위

**보내는 이의 통장에서 돈을 차감하고,
받는 이의 통장에 돈을 추가하는 '송금한다'라는
논리적 기능**

**게시글을 저장하고,
게시글에 포함된 첨부 파일을 저장하는 '등록한다'라는
논리적 기능**

트랜잭션

- 하나의 논리적 기능을 수행하기 위한 하나의 작업 단위
- 논리적인 작업 셋을 모두 완벽하게 처리하거나,

  처리에 실패하는 경우 작업의 일부만 적용되는 현상이 만들어지지 않게 하는 기능

  '송금한다'에서 가군의 통장에서 만 원이 차감되고 나양의 통장에는 만 원이 추가되지 않는 현상

  '등록한다'에서 게시글 작성에 실패했지만, 게시글에 포함된 파일의 경로는 등록되는 현상

- 동시에 동일한 자원에 접근할 때 한 시점에 하나의 요청만 접근할 수 있도록 보장하는 잠금과 다르게,

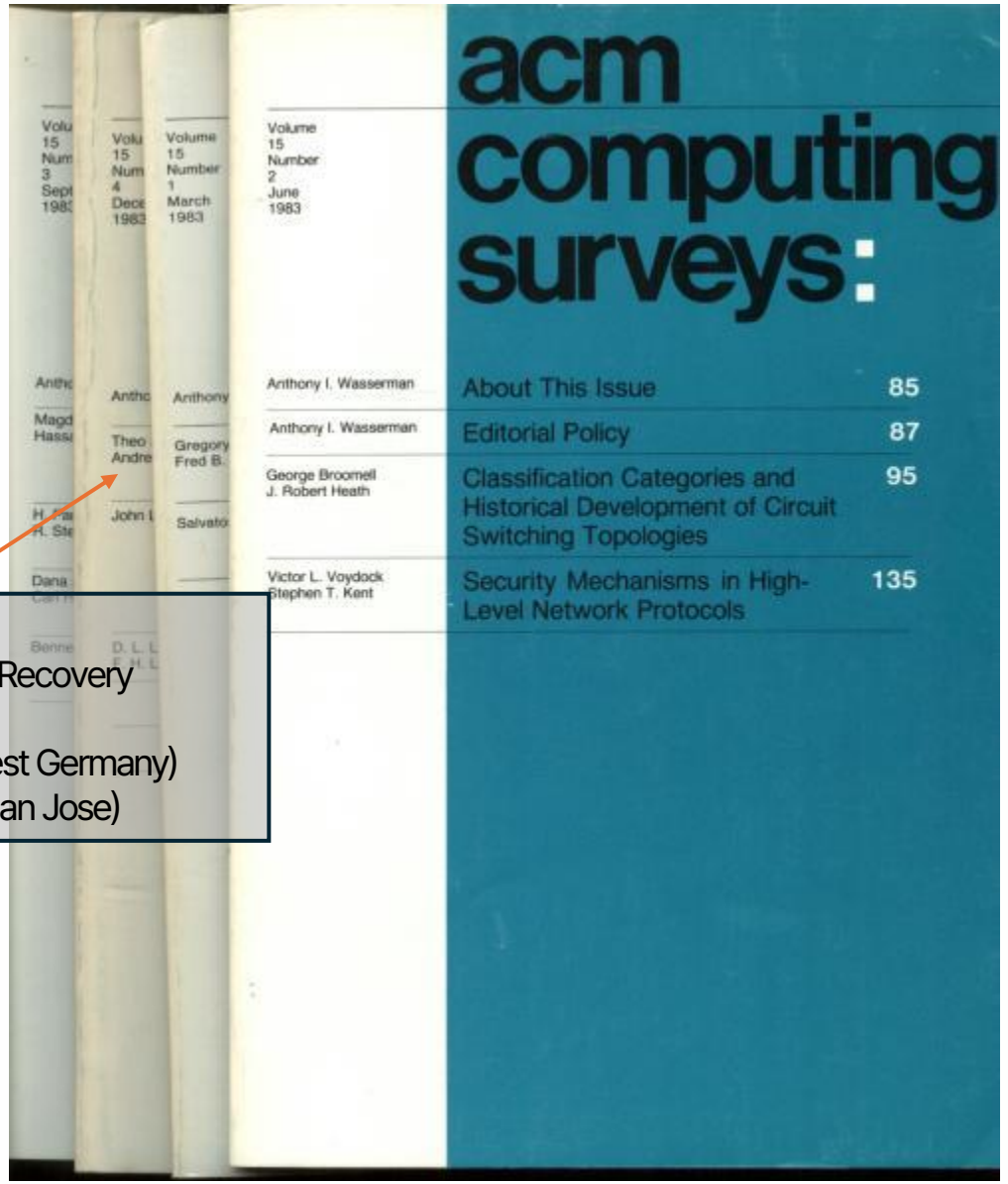
  트랜잭션은 논리적인 작업 셋 자체가 모두 적용되거나 아무 것도 적용되지 않는 것을 보장

02

트랜잭션의 성질

1983년 12월 Computing Surveys에 발표
Principles of Transaction-Oriented Database Recovery

Theo Härder (University of Kaiserslautern, West Germany)
Andreas Reuter (Intel Research Laboratory, San Jose)



Oldcomputerbooks.com : ACM Computing Surveys, 4 individual issues, complete year 1983; Volume 15 nos. 1-4, March, June, September, December 1983.

트랜잭션의 성질

A 원자성

C 일관성

I 격리성

D 지속성

트랜잭션의 성질: Atomicity

A 원자성

C 일관성

I 격리성

D 지속성

Atomicity. It must be of the all-or-nothing type described above, and the user must, whatever happens, know which state he or she is in.

◈ Atomicity (원자성): 일부만 실행되지 않고 전부 실행되거나 또는 아무것도 실행되지 않아야 함

트랜잭션의 성질: Atomicity

A 원자성

C 일관성

I 격리성

D 지속성

원자성. 모든 작업이 전부 수행되거나 전혀 수행되지 않아야 하며, 어떤 상황이 발생하더라도 사용자는 현재 트랜잭션이 어느 상태에 있는지 확실히 알 수 있어야 한다.

- 트랜잭션 실행 중에 문제가 발생했을 때

  트랜잭션 내 모든 작업이 반영되어서는 안 된다.
- 트랜잭션 실행 중에 문제가 발생하지 않았다면

  트랜잭션 내 모든 작업이 반영되어야 한다.

예시로 알아보기

- 가군의 잔고 차감 후 오류가 발생했다면, 가군과 나양의 잔고는 오류 발생 전 상태여야 한다.
- 게시글 정보 등록 후 오류가 발생했다면, 게시글과 파일들은 오류 발생 전 상태여야 한다.

트랜잭션의 성질: Consistency

A **C** **I** **D**

원자성 **일관성** 격리성 지속성

Consistency. A transaction reaching its normal end (EOT, end of transaction), thereby committing its results, preserves the consistency of the database. In other words, each successful transaction by definition commits only legal results. This condition is necessary for the fourth property, durability.

◈ Consistency (일관성): 실행 후에도 데이터베이스는 일관성이 있음 (모순이 없음)

트랜잭션의 성질: Consistency

A 원자성

C 일관성

I 격리성

D 지속성

Consistency. A transaction reaching its normal end (EOT, end of transaction), dition is necessary for the fourth property, durability.

일관성. 트랜잭션이 정상적으로 종료돼 결과를 커밋하면 데이터베이스의 일관성이 유지된다. 즉, 성공적으로 끝난 트랜잭션은 정의상 합법적인 결과만을 커밋한다. 이러한 일관성 보장은 이어지는 지속성과도 밀접하게 연결된다.

- 트랜잭션이 완료된 다음 상태에서도 트랜잭션 수행 전과 마찬가지로 데이터 일관성이 보장되어야 한다.

예시로 알아보기

- 모든 통장 잔고가 0원 이상이어야 한다는 조건이 있다면 송금 후에도 이를 만족해야 한다.
- 게시글 제목과 내용, 연관된 파일의 경로는 null이 아니어야 한다는 조건이 있다면 게시글 등록 후에도 이를 만족해야 한다.

트랜잭션의 성질: Isolation

A 원자성

C 일관성

I 격리성

D 지속성

Isolation. Events within a transaction must be hidden from other transactions running concurrently. If this were not the case, a transaction could not be reset to its beginning for the reasons sketched above. The techniques that achieve isolation are known as *synchronization*, and since Gray et al. [1976] there have been numerous contributions to this topic of database research [Kohler 1981].

◈ Isolation (격리성): 트랜잭션 실행 중에 있는 연산의 중간 결과는 다른 트랜잭션이 접근할 수 없음

트랜잭션의 성질: Isolation

A 원자성

C 일관성

I 격리성

D 지속성

Isolation. Events within a transaction must be hidden from other transactions running concurrently. If this were not the case, then the measurements performed above [...] and since Gray et al. [1976] there have been numerous contributions to this topic of database research [Kohler 1981].

격리성. 트랜잭션 내 작업은 동시에 실행 중인 다른 트랜잭션에 완전히 숨겨져야 한다. 그렇지 않으면 앞서 설명한 이유로 트랜잭션을 처음 상태로 되돌릴 수 없게 된다. 이런 격리성을 구현하는 기법들을 동기화라고 부르며 Gray 등 이후 다수의 연구가 이 주제에 기여해 왔다.

- 각각의 트랜잭션은 서로 간섭 없이 독립적으로 수행되어야 한다.

예시로 알아보기

- 가군이 나양에게 송금함과 동시에 나양이 통장에 입금을 했다면 이를 순차적으로 실행한 것과 같은 결과여야 한다.
- 게시글 등록 중 다른 사람이 게시판 조회 시 등록 중인 게시글은 조회되지 않는다.

트랜잭션의 성질: Durability

A 원자성

C 일관성

I 격리성

D 지속성

Durability. Once a transaction has been completed and has committed its results to the database, the system must guarantee that these results survive any subsequent malfunctions. Since there is no sphere of control constituting a set of transactions, the database management system (DBMS) has no control beyond transaction boundaries. Therefore the user must have a guarantee that the things the system says have happened have actually happened. Since, by definition, each transaction is correct, the effects of an inevitable incorrect transaction (i.e., the transaction containing faulty data) can only be removed by countertransactions.

◈ Durability (영속성): 트랜잭션의 결과는 영속적임

트랜잭션의 성질: Durability

A **C** **I** **D**

원자성 일관성 격리성 지속성

Durability. Once a transaction has been completed and has committed its results to the database, the system must guarantee

지속성. 트랜잭션이 완료되어 결과를 데이터베이스에 커밋한뒤에는 이후 발생할 수 있는 어떤 장애가 있어도 그 결과가 보존되어야 한다. 여러 트랜잭션을 하나의 제어 구역으로 묶는 통제 범위가 존재하지 않으므로 데이터베이스 관리 시스템(DBMS)은 트랜잭션 경계를 넘어선 부분까지는 제어할 수 없다. 따라서 시스템이 '실행되었다'고 보고한 내용은 실제로 영구히 반영되었음을 사용자에게 보장해야 한다. 정의상 각 트랜잭션은 올바르므로, 잘못된 데이터가 포함된 부정확한 트랜잭션의 영향은 오직 상쇄 트랜잭션을 통해서만 제거될 수 있다.

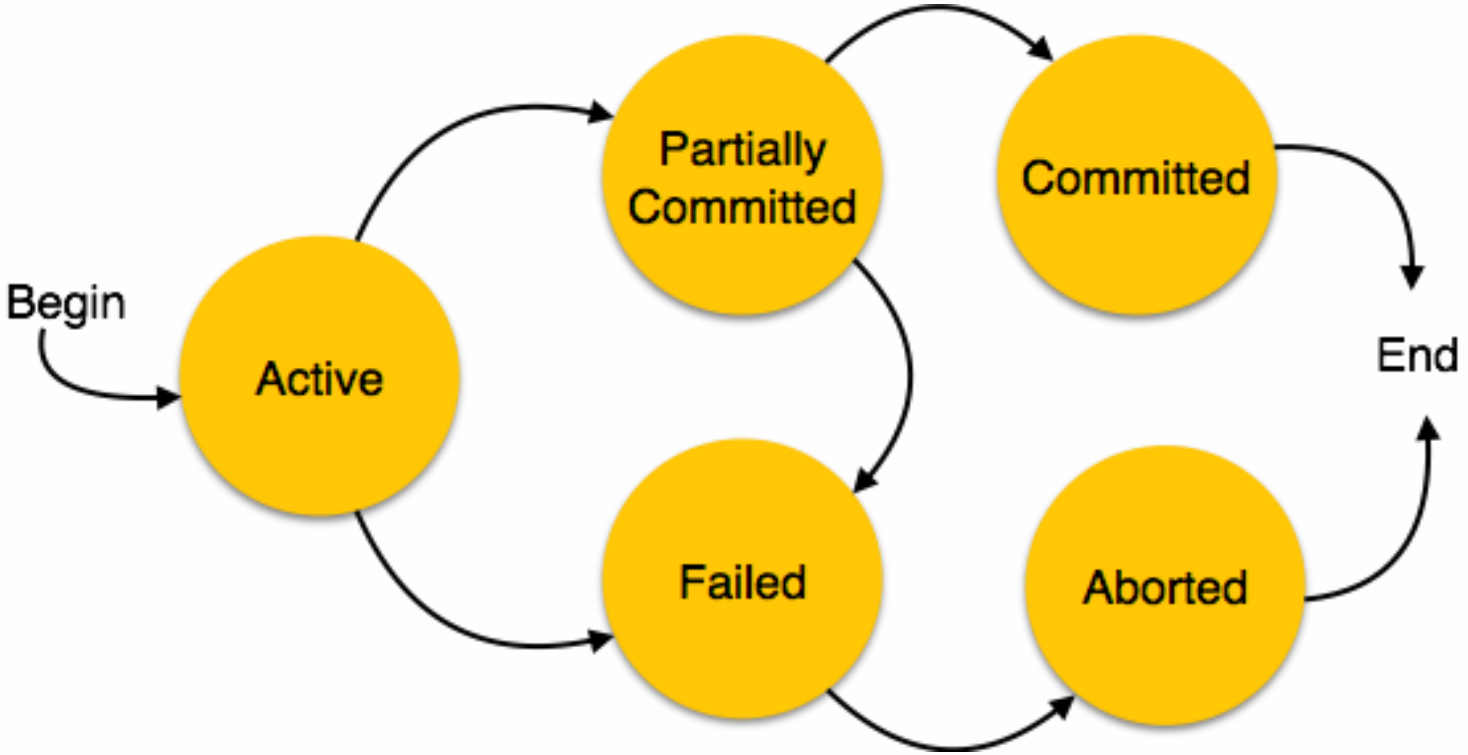
an inevitable incorrect transaction (i.e., the transaction containing faulty data) can only be removed by countertransactions.

- 트랜잭션이 반영된 후에는 영구적으로 데이터베이스에 작업 결과가 저장되어야 하며, 잘못된 데이터는 이를 수정하는 새로운 트랜잭션을 통해서만 할 수 있다.

예시로 알아보기

- 데이터베이스 서버가 정전되더라도 트랜잭션 반영 후 값으로 유지해야 한다. 로그를 통해 이를 실현할 수 있다.

트랜잭션 상태 다이어그램 (트랜잭션 주요 상태 및 전이)



데이터베이스 기말 시험에 나왔다!

(16) 트랜잭션의 4가지 성질 중에서 "트랜잭션의 모든 연산들이 정상적으로 수행 완료되거나 아니면 전혀 어떠한 연산도 수행되지 않은 원래 상태가 되도록 해야 한다." 는 것은? atomicity consistency 33

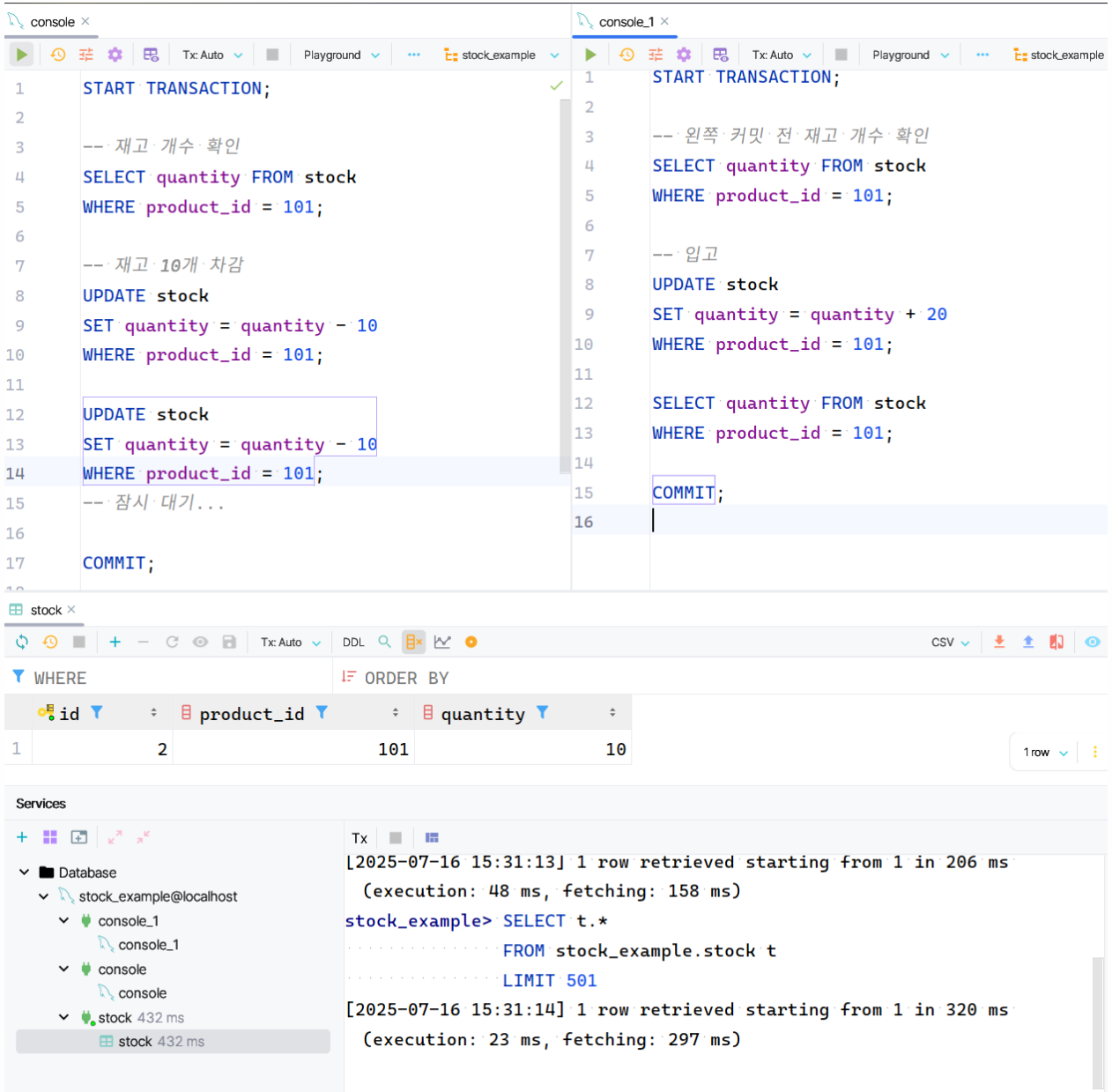
아, 안돼

틀렸다

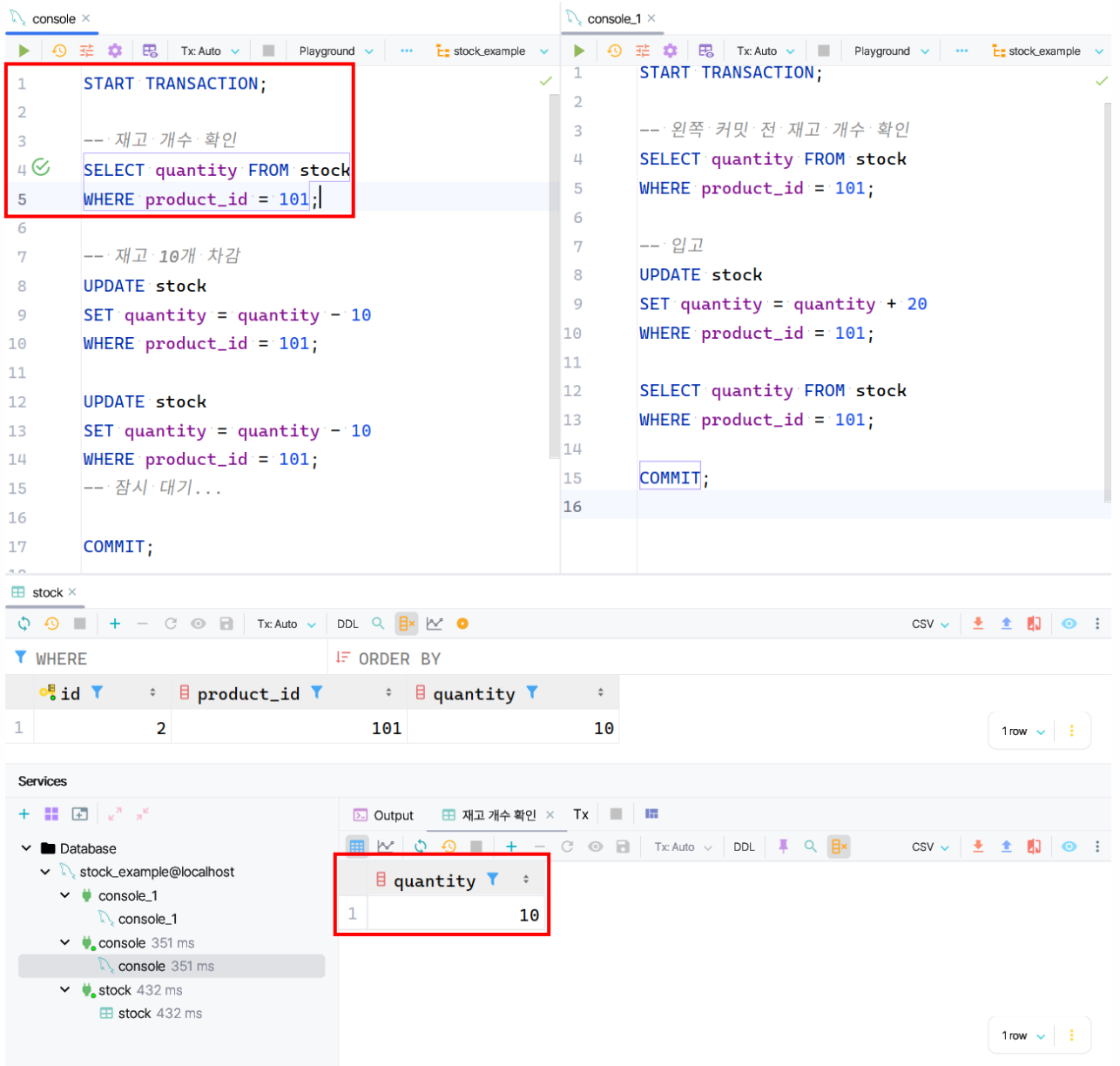
03

트랜잭션 살펴보기

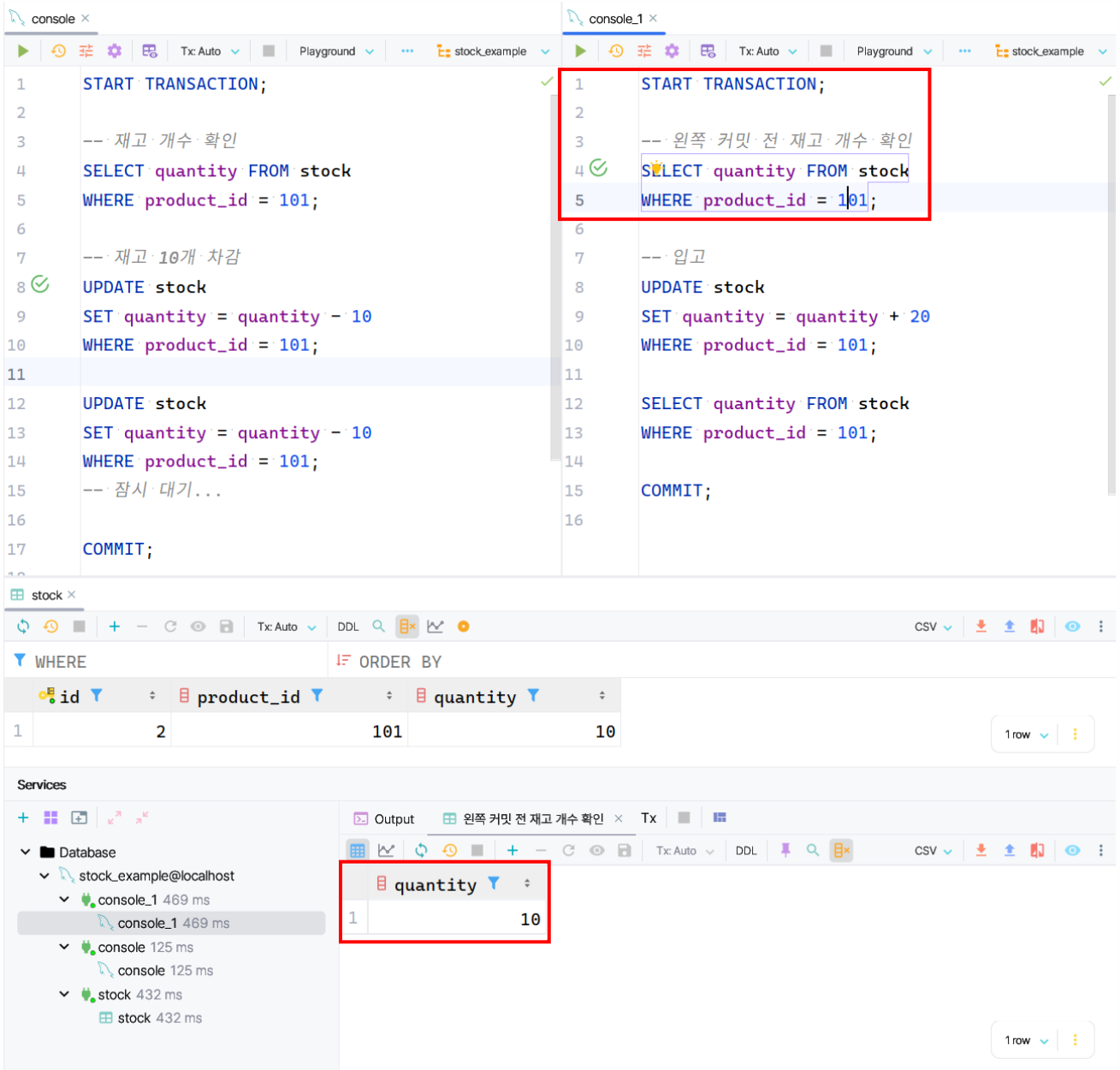
트랜잭션 살펴보기: 격리성(Isolation)



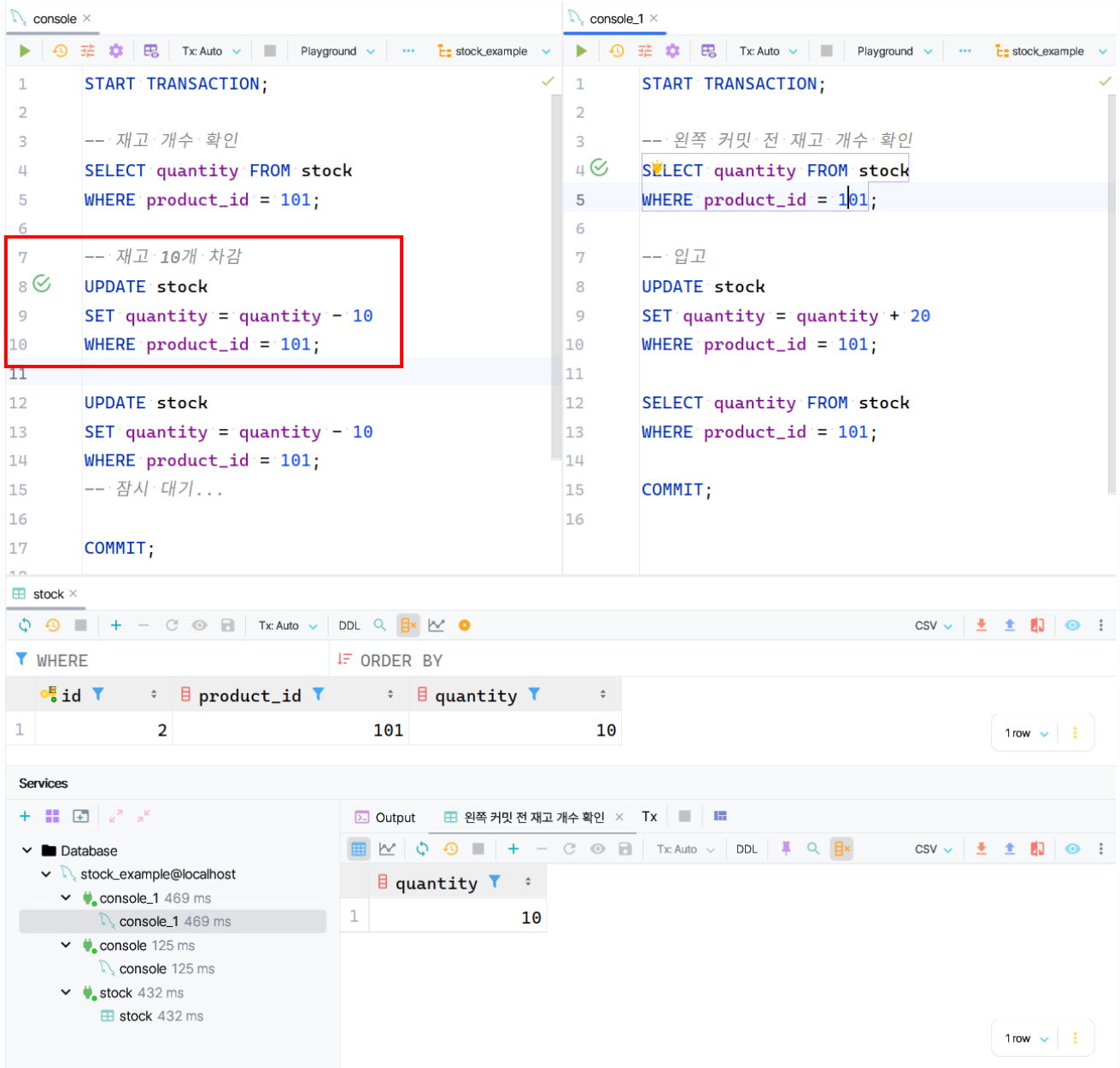
트랜잭션 살펴보기: 격리성(Isolation)



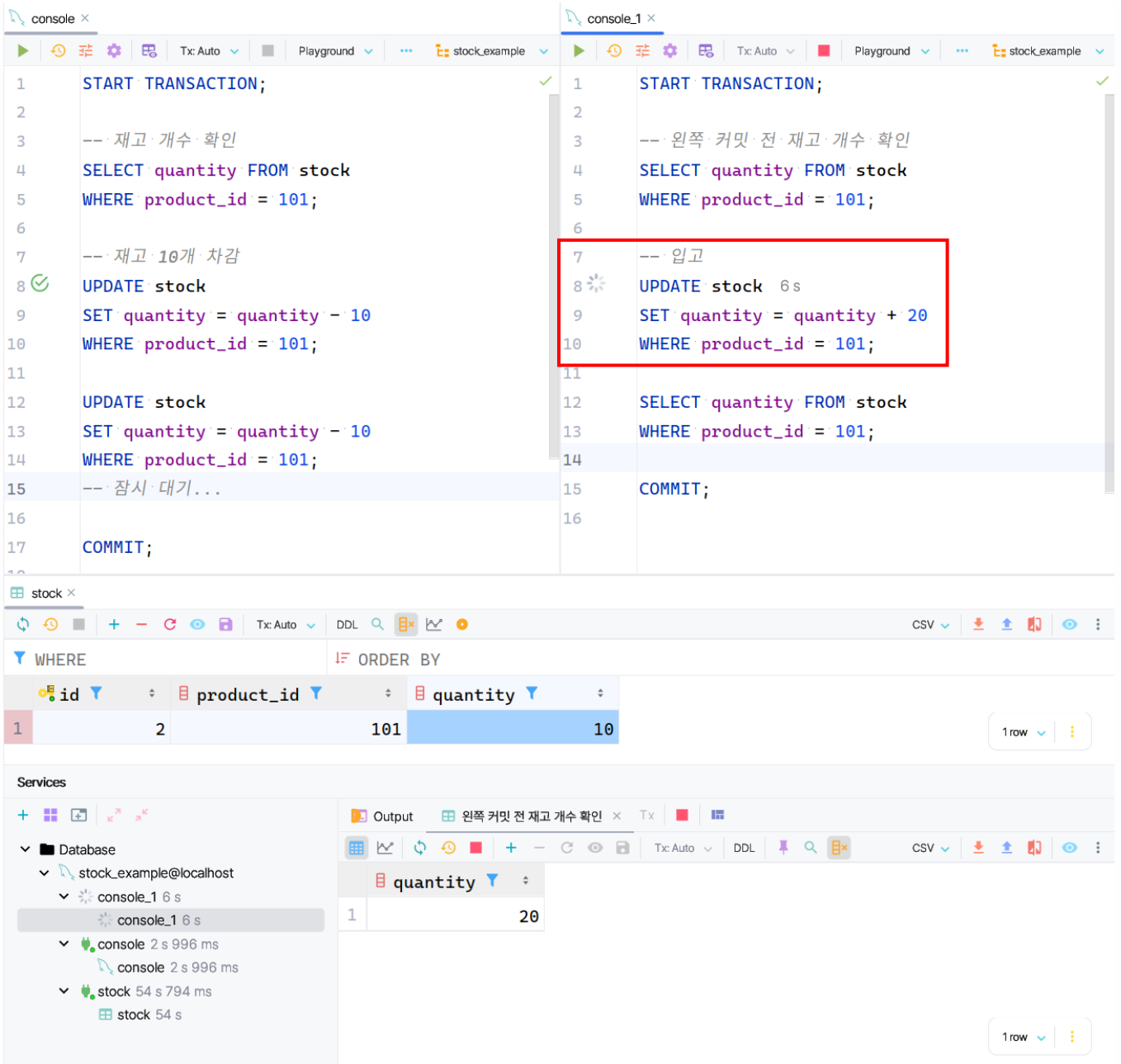
트랜잭션 살펴보기: 격리성(Isolation)



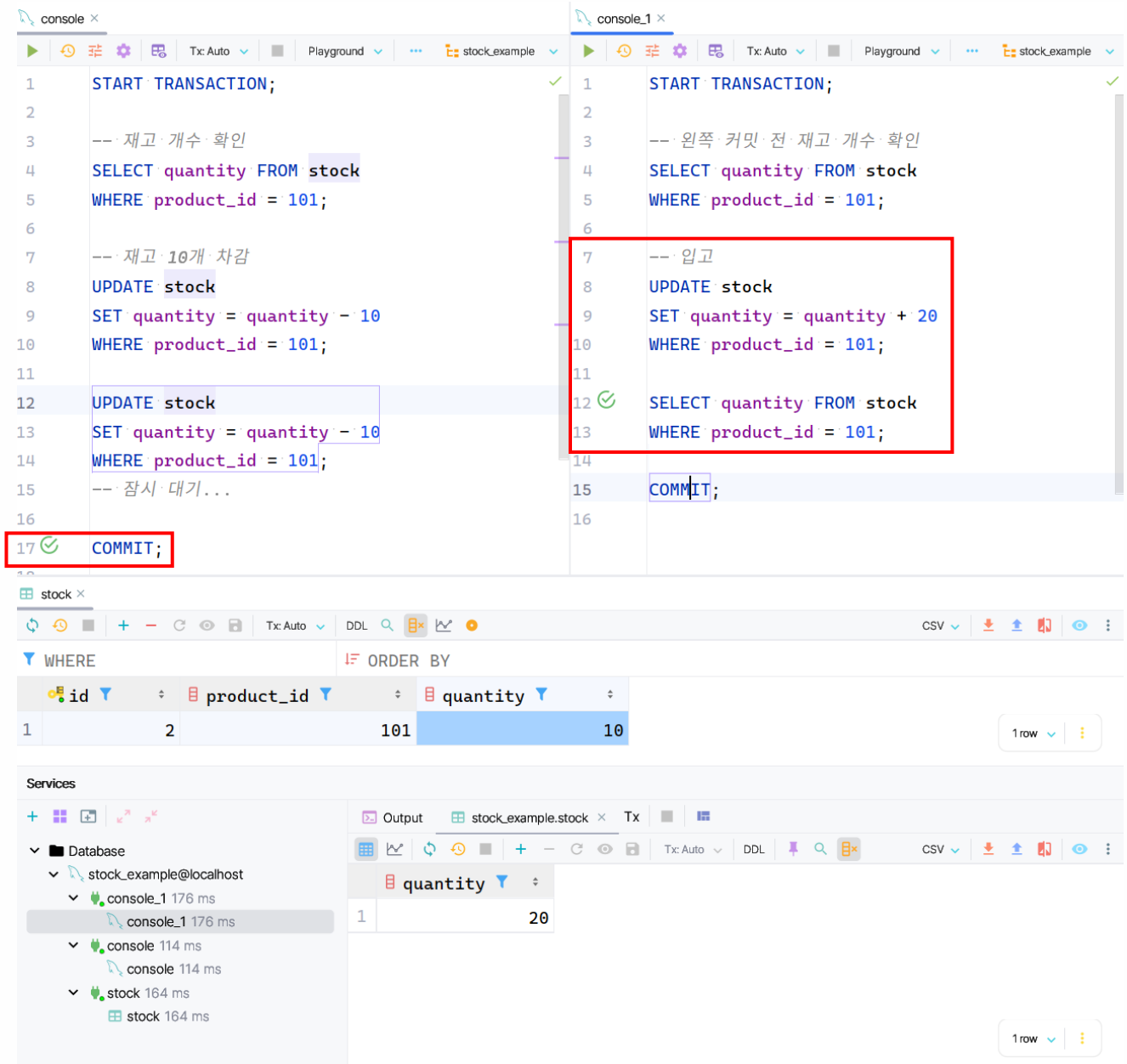
트랜잭션 살펴보기: 격리성(Isolation)



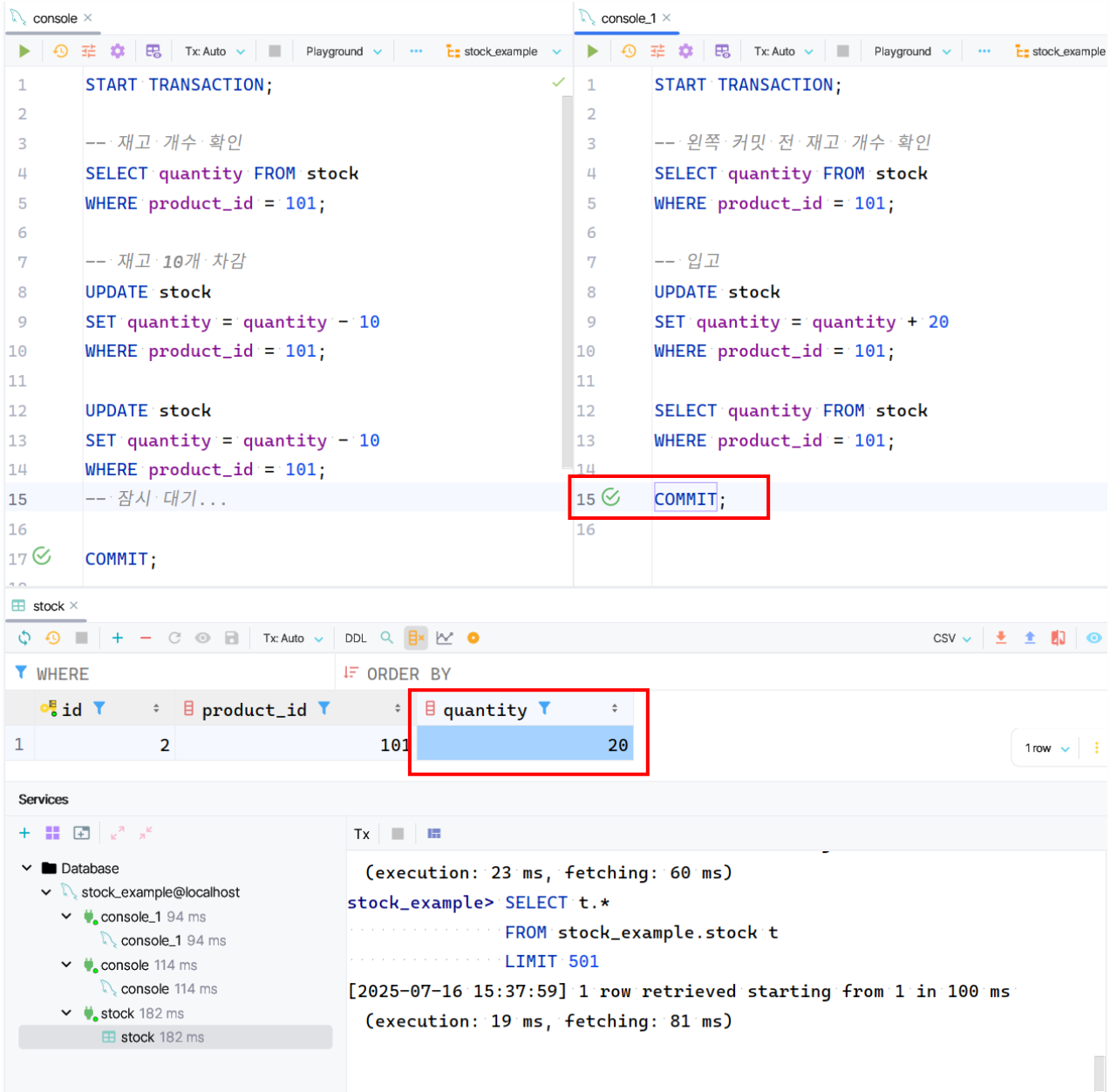
트랜잭션 살펴보기: 격리성(Isolation)



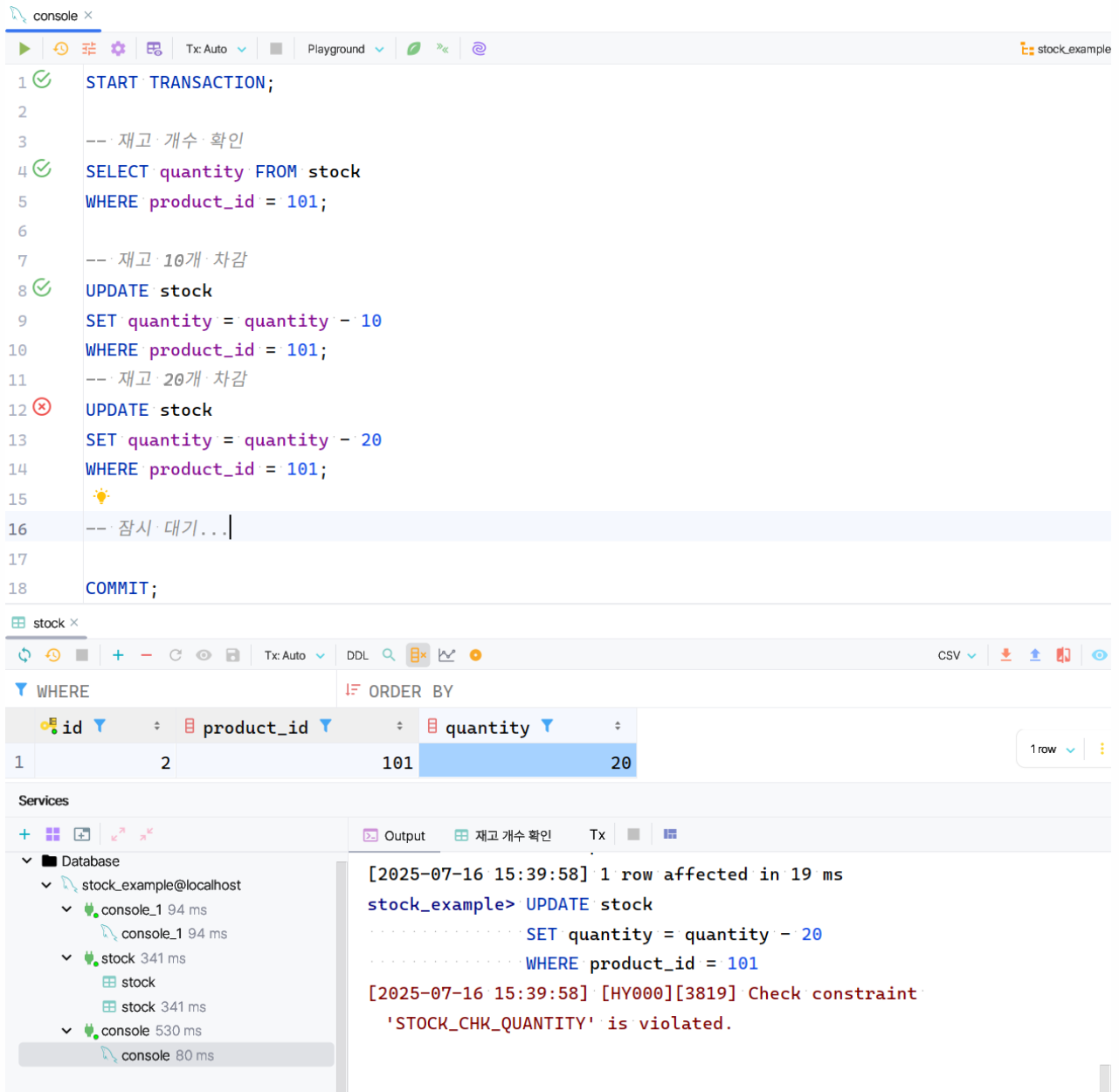
트랜잭션 살펴보기: 격리성(Isolation)



트랜잭션 살펴보기: 격리성(Isolation)



트랜잭션 살펴보기: 원자성(Atomicity) , 일관성(Consistency)



NEXT

트랜잭션 격리 수준

트랜잭션의 성질: Isolation

A 원자성

C 일관성

I 격리성

D 지속성

Isolation. Events within a transaction must be hidden from other transactions running concurrently. If this were not the case, then the measurements made above [illegible] and since Gray et al. [1976] there have been numerous contributions to this topic of database research [Kohler 1981].

격리성. 트랜잭션 내 작업은 동시에 실행 중인 다른 트랜잭션에 완전히 숨겨져야 한다. 그렇지 않으면 앞서 설명한 이유로 트랜잭션을 처음 상태로 되돌릴 수 없게 된다. 이런 격리성을 구현하는 기법들을 동기화라고 부르며 Gray 등 이후 다수의 연구가 이 주제에 기여해 왔다.

- 각각의 트랜잭션은 서로 간섭 없이 독립적으로 수행되어야 한다.

예시로 알아보기

- 가군이 나양에게 송금함과 동시에 나양이 통장에 입금을 했다면 이를 순차적으로 실행한 것과 같은 결과여야 한다.
- 게시글 등록 중 다른 사람이 게시판 조회 시 등록 중인 게시글은 조회되지 않는다.

NEXT

트랜잭션 격리 수준

들어 주셔서 감사합니다

E-mail	park@duck.com
LinkedIn	yeonjong-park
Instagram	yeonjong.park
GitHub	patulus

2025.07.16.

System Software Lab.