Introduction to transactions

THANK YOU TO ANDRE HERNICH FOR BORROWING SOME SLIDES

NEW: USING CAMERA!

Overview of this video

What is a transaction?

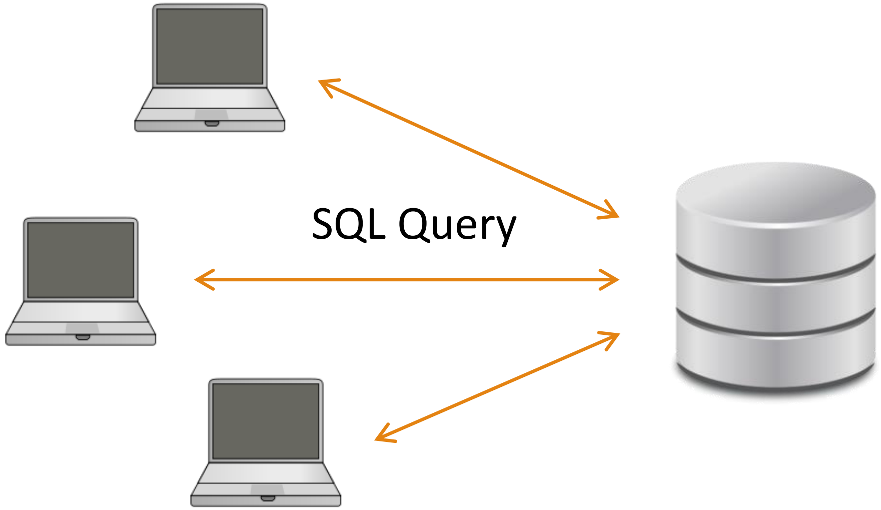
- A sequence of queries

That sounds very easy!

- The complication arises from that many might happen at the same time and that computers can fail…

Executing SQL Queries in ”Real Life”

So far: SQL queries in isolation



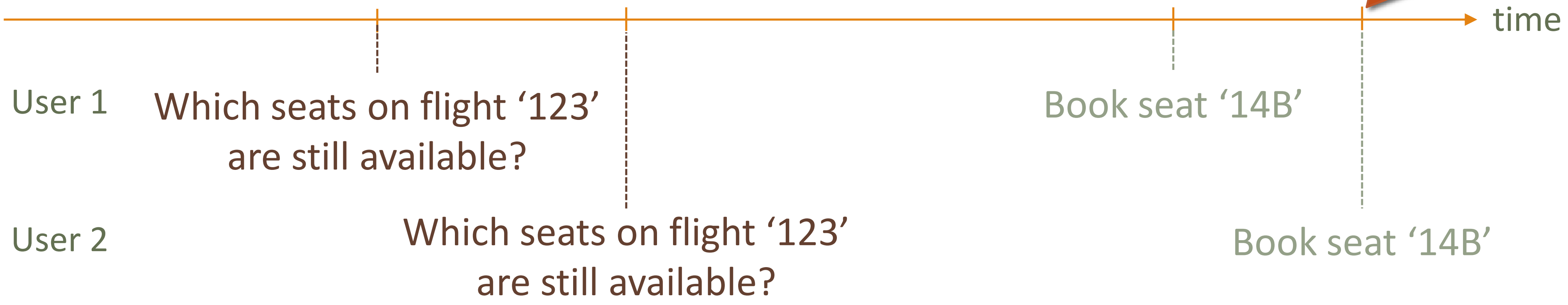
In practice, problems may arise due to

- **Concurrency**: SQL statements that overlap in time
- **Partial execution** of SQL statements (e.g., due to failures)

Problem 1: Concurrency

Flights(flightNo, date, seatNo, seatStatus)

Might lead to an inconsistent database



time

User 1

Which seats on flight '123' are still available?

Book seat '14B'

User 2

Which seats on flight '123' are still available?

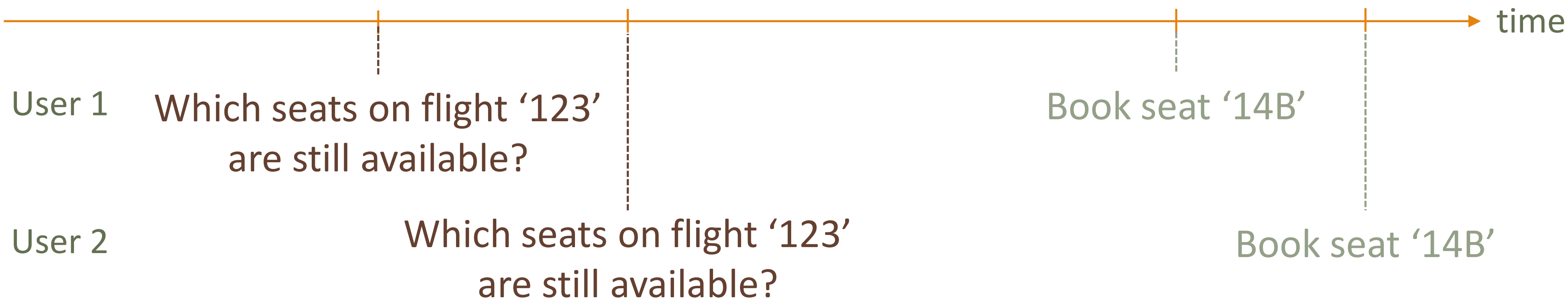
Book seat '14B'

```
SELECT seatNo
FROM   Flights
WHERE  flightNo = 123
   AND date = '2020-10-30'
   AND seatStatus = 'available';
```

```
UPDATE Flights
SET    seatStatus = 'occupied'
WHERE  flightNo = 123
   AND date = '2020-10-30'
   AND seatNo = '14B';
```

Problem 1: Concurrency

Flights(flightNo, date, seatNo, seatStatus)



time

User 1

Which seats on flight ‘123’ are still available?

Book seat ‘14B’

User 2

Which seats on flight ‘123’ are still available?

Book seat ‘14B’

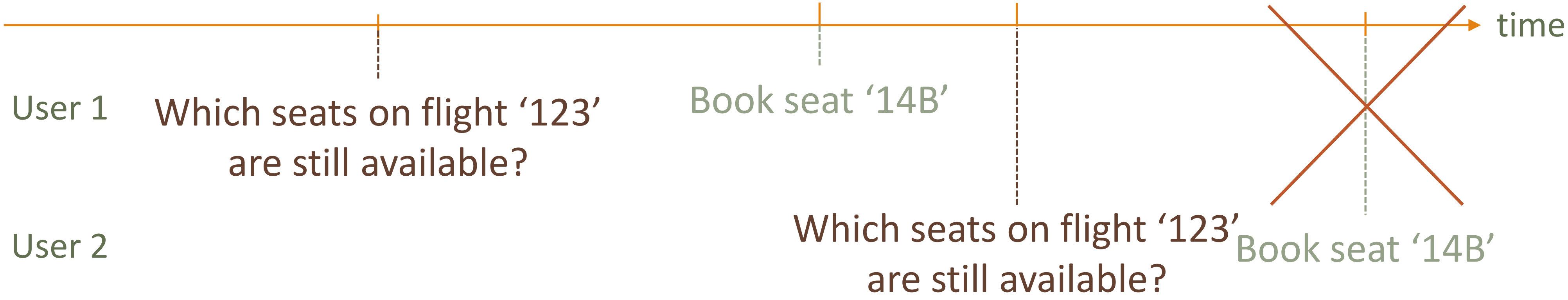
```
SELECT seatNo
FROM   Flights
WHERE  flightNo = 123
   AND date = ’2020-10-30’
   AND seatStatus = ’available’;
```

```
UPDATE Flights
SET    seatStatus = ’occupied’
WHERE  flightNo = 123
   AND date = ’2020-10-30’
   AND seatNo = ’14B’;
```

Problem 1: Concurrency

Cannot happen anymore

Flights(flightNo, date, seatNo, seatStatus)



time

User 1

Which seats on flight ‘123’ are still available?

Book seat ‘14B’

User 2

Which seats on flight ‘123’ are still available?

Book seat ‘14B’

```
SELECT seatNo
FROM   Flights
WHERE  flightNo = 123
   AND date = ’2020-10-30’
   AND seatStatus = ’available’;
```

```
UPDATE Flights
SET    seatStatus = ’occupied’
WHERE  flightNo = 123
   AND date = ’2020-10-30’
   AND seatNo = ’14B’;
```

Transactions to the Rescue

SQL allows us to state that a group of SQL statements must be executed so that no conflicts arise
(we’ll see later how this is enforced)

```
START TRANSACTION;
```

Can often be omitted

```
SELECT seatNo
FROM   Flights
WHERE  flightNo = 123 AND date = ’ 2020-10-30’
   AND seatStatus = ’available’;
```

```
UPDATE Flights
SET    seatStatus = ’occupied’
WHERE  flightNo = 123 AND date = ’ 2020-10-30’
   AND seatNo = ’14B’;
```

Before this, all changes to the database are tentative.

```
COMMIT;
```

Transactions in SQL

Transaction in SQL: a sequence of SQL statements

- Special case: each individual SQL statement is a transaction

By telling a DBMS that a sequence of SQL statements forms a transaction, it ensures *serialisable behaviour*

- The transaction is executed as if it was executed in isolation from all other transactions
- *Equivalently:* … as if all transactions were executed one after the other

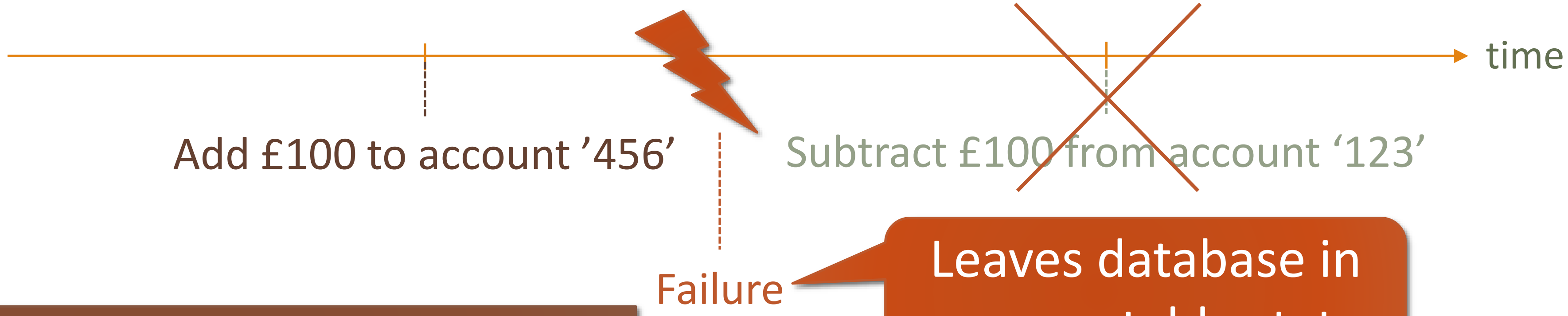
How strict can be set in SQL

It also ensures other properties…

Problem 2: Partial Execution

Accounts(accountNo, accountHolder, balance)

Goal: Transfer £100 from account ‘123’ to account ‘456’


time
Add £100 to account ’456’
~~Subtract £100 from account ‘123’~~
Failure

Leaves database in unacceptable state

```
UPDATE Accounts
SET    balance = balance + 100
WHERE  accountNo = 456;
```

```
UPDATE Accounts
SET    balance = balance - 100
WHERE  acountNo = 123;
```

Transactions to the Rescue

SQL allows us to state that a transaction must be executed atomically (as a whole or not at all)

```
START TRANSACTION;
```

```
UPDATE Accounts
SET    balance = balance + 100
WHERE  accountNo = 456;
```

```
UPDATE Accounts
SET    balance = balance - 100
WHERE  acountNo = 123;
```

```
COMMIT;
```

Transactions in SQL - Summary

Transaction in SQL: a sequence of SQL statements

- Special case: each individual SQL statement is a transaction
- General form:

```
START TRANSACTION;
```

Starts a transaction (can often be omitted)

SQL statements

```
COMMIT;
```

or

```
ROLLBACK;
```

Writes all changes to the database

Aborts the transaction