



# **3C03 Concurrency: Transactions**

**Wolfgang Emmerich**



## **Outline**

- **Motivation**
- **Transaction Concepts**
  - **Atomicity**
  - **Consistency**
  - **Isolation**
  - **Durability**
- **Nested Transactions**
- **Centralized vs. Distributed Transactions**
- **Summary**



## **Motivation**

- ***What happens if a failure occurs during modification of resources?***
- ***Which operations have been completed?***
- ***Which operations have not (and have to be done again)?***
- ***In which states will the resources be?***



## **Transaction Concepts**

### **1 ACID Properties**

- ***Atomicity***
- ***Consistency***
- ***Isolation***
- ***Durability***

### **2 Transaction Commit vs. Abort**

### **3 Flat vs. Nested Transactions**

### **4 Central vs. Distributed Transactions**



## **Atomicity**

- ***Transactions are either performed completely or no modification is done.***
- ***Start of a transaction is a continuation point to which it can roll back.***
- ***End of transaction is next continuation point.***



## **Consistency**

- ***Shared resources should always be consistent.***
- ***Inconsistent states occur during transactions:***
  - *hidden for concurrent transactions*
  - *to be resolved before end of transaction.*
- ***Application defines consistency and is responsible for ensuring it is maintained.***
- ***Transactions can be aborted if they cannot resolve inconsistencies.***



## **Isolation**

- ***Each transaction accesses resources as if there were no other concurrent transactions.***
- ***Modifications of the transaction are not visible to other resources before it finishes.***
- ***Modifications of other transactions are not visible during the transaction at all.***
- ***Implemented through:***
  - *two-phase locking or*
  - *optimistic concurrency control.*

© Wolfgang Emmerich, 1998/99

7



## **Durability**

- ***A completed transaction is always persistent (though values may be changed by later transactions).***
- ***Modified resources must be held on persistent storage before transaction can complete.***
- ***May not just be disk but can include battery-backed RAM or Flash RAM.***

© Wolfgang Emmerich, 1998/99

8



## Transaction Commands

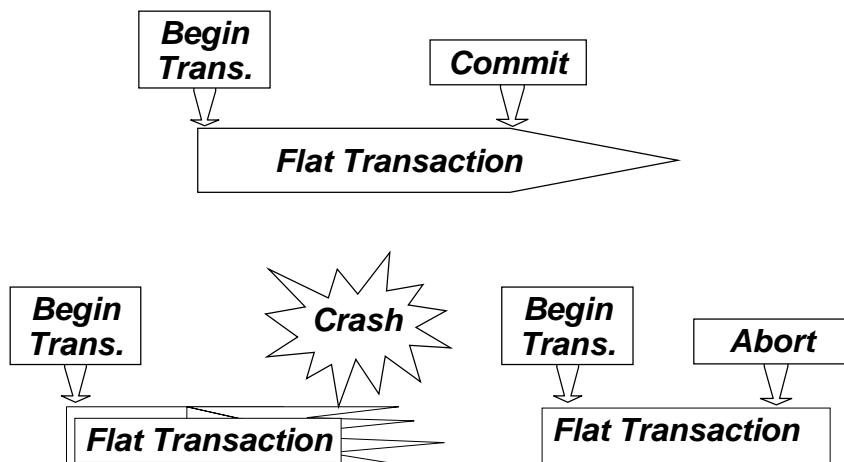
- **Begin:**
  - *Start a new transaction.*
- **Commit:**
  - *End a transaction.*
  - *Store changes made during transaction.*
  - *Make changes accessible to other transactions.*
- **Abort:**
  - *End a transaction.*
  - *Undo all changes made during the transaction.*

© Wolfgang Emmerich, 1998/99

9

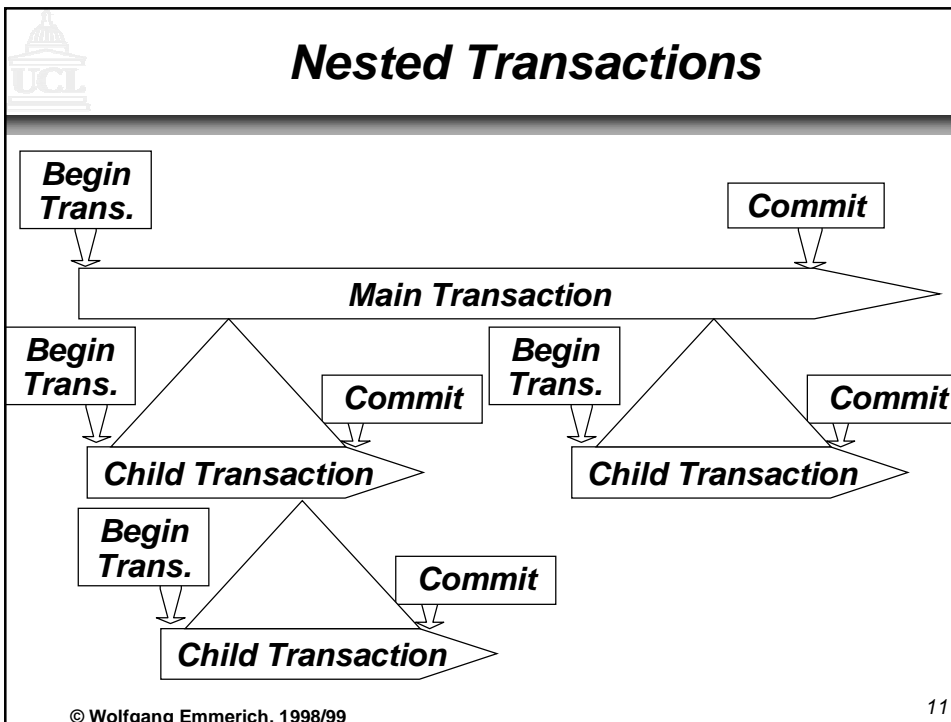


## Flat Transactions



© Wolfgang Emmerich, 1998/99

10



- 
- ## Central vs. Distributed Transactions
- **Transactions in a Database**
    - **Centralized**
    - **DBMS controls transaction execution**
    - **DBMS implements concurrency control**
    - **Transaction processing transparent to application developers**
  - **Problem occurs if:**
    - **Data kept in different databases or**
    - **Distributed objects do not use a database**
    - **Transaction processing not transparent to application developers**
- © Wolfgang Emmerich, 1998/99 12



## **Summary**

- **Motivation**
- **Transaction Concepts**
  - **Atomicity**
  - **Consistency**
  - **Isolation**
  - **Durability**
- **Nested Transactions**
- **Centralized vs. Distributed Transactions**
- **Summary**