



# Transactions

---

## Properties

## Solution for both concurrency and failures

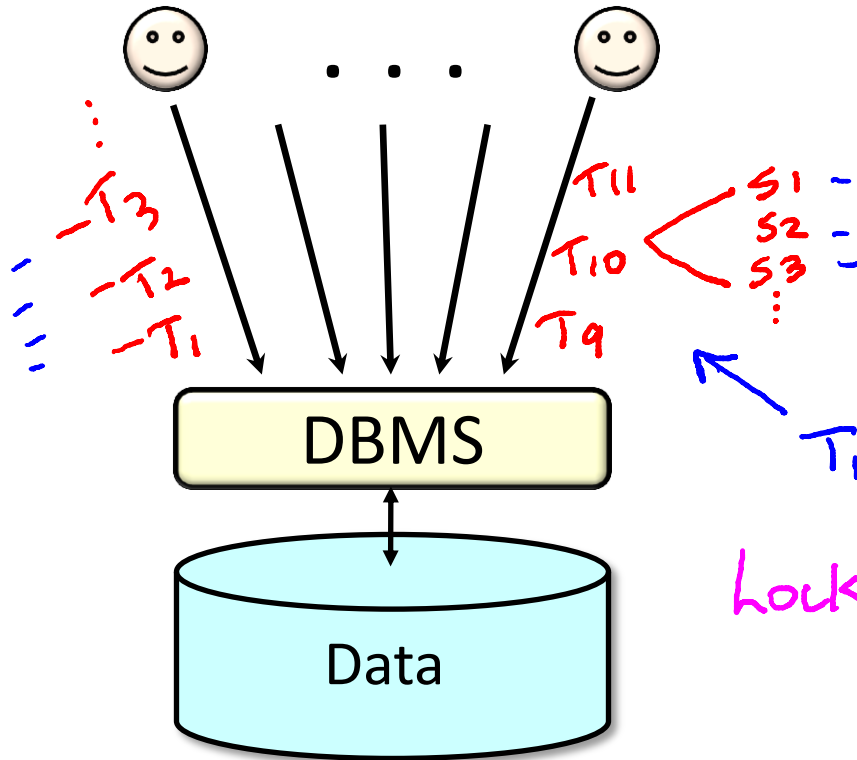
# Transactions

**A transaction is a sequence of one or more SQL operations treated as a unit**

- Transactions appear to run in isolation
- If the system fails, each transaction's changes are reflected either entirely or not at all

# ACID Properties

- ③ **A**tomicity
- ④ **C**onsistency
- ① **I**solation
- ② **D**urability

(ACID Properties) **Isolation**

→ Serializability

Operations may be interleaved, but execution must be equivalent to *some* sequential (serial) order of all transactions

# Concurrent Access: Attribute-level Inconsistency

$T_1$  Update college Set enrollment = enrollment + 1000  
where cName = 'Stanford'

concurrent with ...

$T_2$  Update college Set enrollment = enrollment + 1500  
where cName = 'Stanford'

$T_1 ; T_2$       15,000  $\rightarrow$  17,500  
 $T_2 ; T_1$

# Concurrent Access: Tuple-level Inconsistency

$T_1$  Update Apply Set major = 'CS' where sID = 123

concurrent with ...

$T_2$  Update Apply Set decision = 'Y' where sID = 123

$T_1; T_2$

$T_2; T_1$

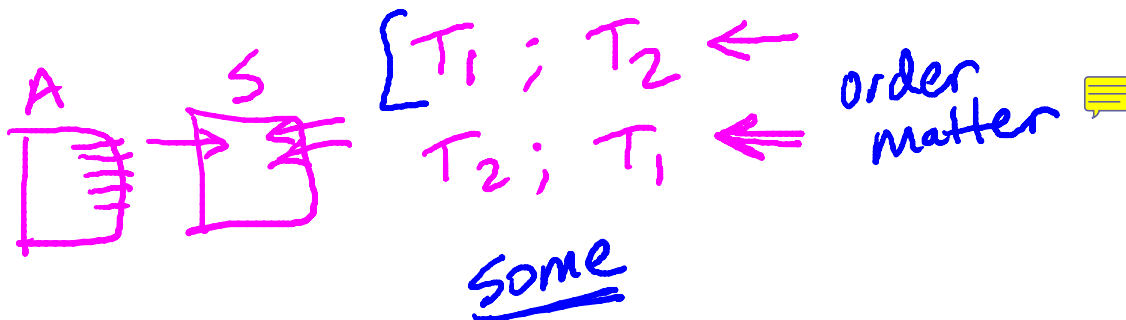
both changes

# Concurrent Access: Table-level Inconsistency

$T_1$  Update Apply Set decision = 'Y'  
where SID In (Select SID From Student where GPA > 3.9)

concurrent with ...

$T_2$  Update Student Set GPA = (1.1) \* GPA where sizeHS > 2500



# Concurrent Access: Multi-statement inconsistency

Insert Into **Archive**

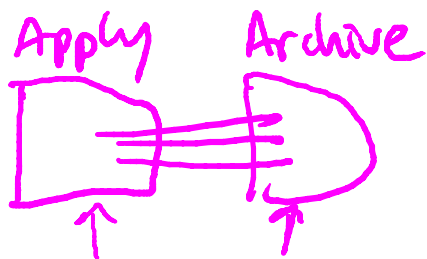
*T1* Select \* From **Apply** where **decision** = 'N';

Delete From **Apply** where **decision** = 'N';

concurrent with ...

*T2* Select Count(\*) From **Apply**;

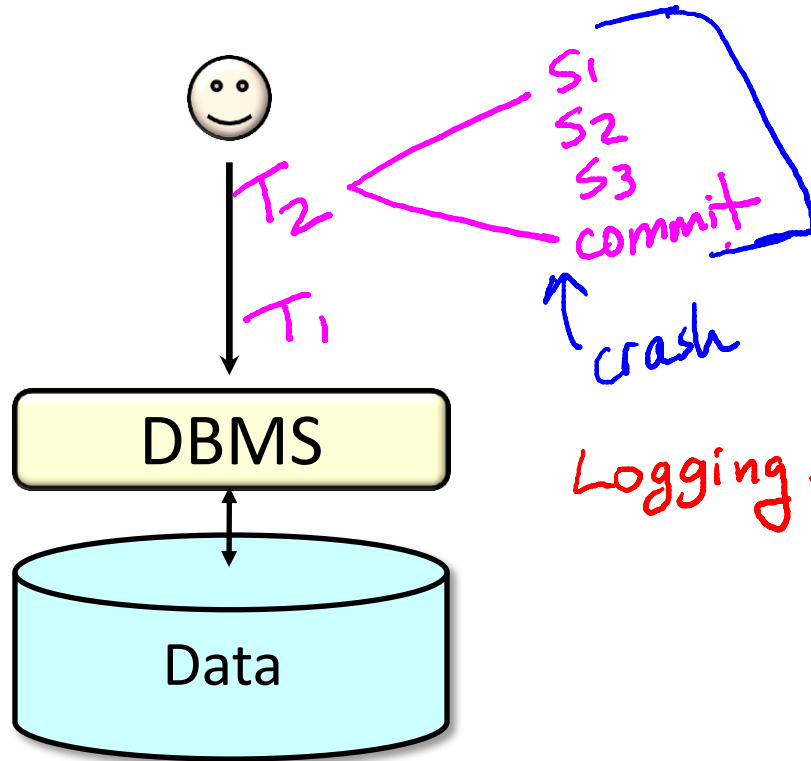
Select Count(\*) From **Archive**;



T<sub>1</sub>; T<sub>2</sub> order  
T<sub>2</sub>; T<sub>1</sub>

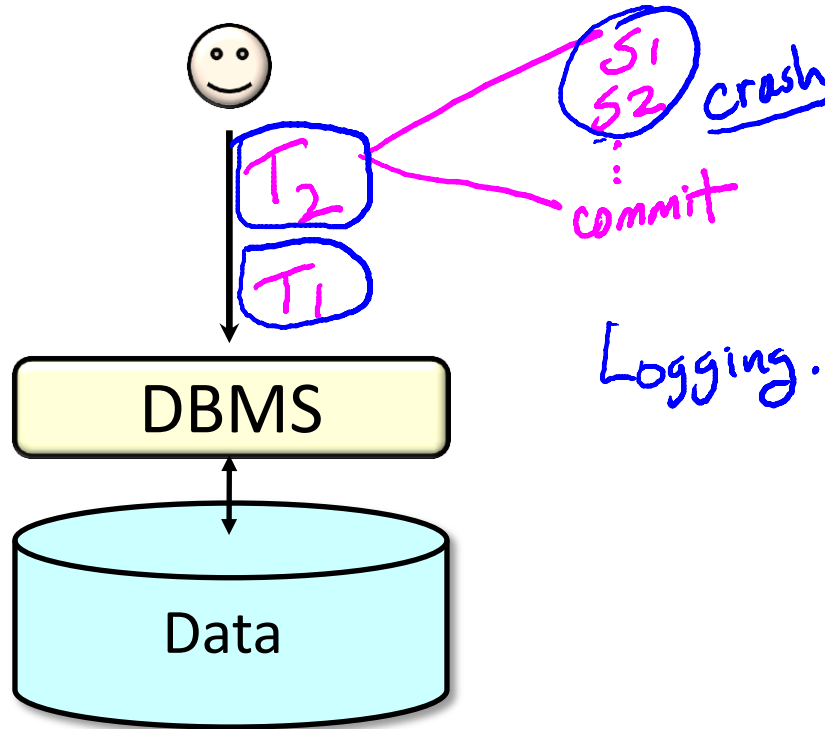


# (ACID Properties) **Durability**



If system crashes  
after transaction commits,  
all effects of transaction  
remain in database

# (ACID Properties) **Atomicity**



Each transaction is  
“all-or-nothing,”  
never left half done

## Transaction Rollback (= Abort) ☆

- Undoes partial effects of transaction
- Can be system- or client-initiated

Each transaction is  
“all-or-nothing,”  
never left half done

Begin Transaction; ←

→ <get input from user> ←

SQL commands based on input

→ <confirm results with user> ←

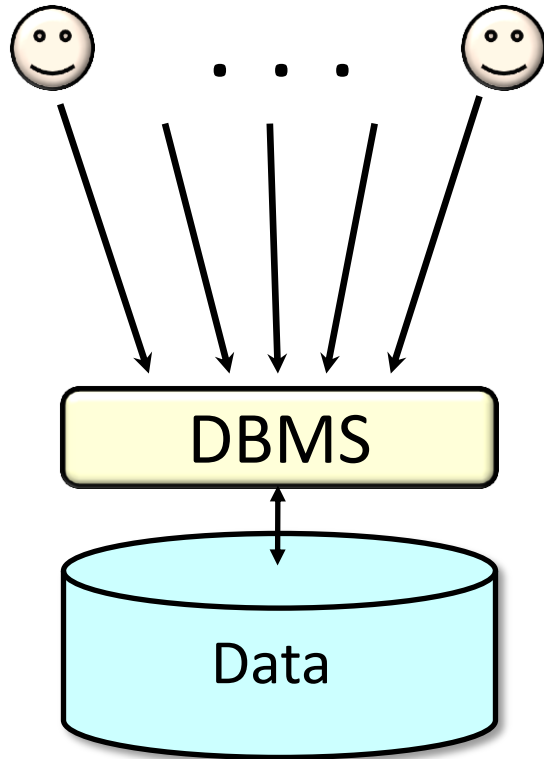
If ans=‘ok’ Then Commit; Else Rollback;

vars  
delivering  
cash



Locking

# (ACID Properties) Consistency



Each client, each transaction:

- Can assume all constraints hold when transaction begins
- Must guarantee all constraints hold when transaction ends

Serializability  $\Rightarrow$  constraints always hold

$T_1$   $T_2$   $T_3$

Three handwritten pink arrows point upwards towards the labels  $T_1$ ,  $T_2$ , and  $T_3$ , which are also written in pink.

# Solution for both concurrency and failures

## Transactions

✓ **A**tomicity

✓ **C**onsistency

✓ **I**solation 

✓ **D**urability