Transactions and Locking

Rose-Hulman Institute of Technology

Curt Clifton

Outline

- ACID Transactions
- COMMIT and ROLLBACK
- Managing Transactions
- Locks

The Setting

- Database systems are normally being accessed by many users or processes at the same time
- Operating Systems also deal with concurrent access
    - OSs allow two people to edit a document at the same time.
    - If both write, one’s **changes get lost**.
- DB can and must do better

Example

- Mom and Dad each deposit $100 from different ATMs into your account at about the same time

ACID Transactions

- Atomic
  - All or nothing
- Consistent
  - Constraints preserved
- Isolated
  - (Apparently) one user at a time
- Durable
  - Crashes can’t violate the other properties

Transactions in SQL

- SQL supports transactions
  - Generic query interface
    - Each statement issued is a transaction by itself
  - Programming interfaces
    - A transaction begins with first SQL statement
    - Ends with the procedure end (or an explicit end)

Ending Transactions

- COMMIT completes a transaction
  - Modifications are now permanent in the database
- ROLLBACK ends transaction by aborting
  - No effects on the database!
- Failures (e.g., division by 0) also cause ROLLBACK

Another Example

- Assume the usual Sells(rest,soda,price) relation
  - Suppose that Majnoo’s Rest sells only Coke for $1.50 and Salaam Cola for $1.75.
- Laila is querying Sells for
  - the highest and lowest price Majnoo charges.
- Majnoo decides
  - to stop selling Coke and Salaam Cola
  - to starting only Juice at $2.00

Laila’s Program

- Laila executes the following two SQL statements
- Call this one “max”:
  - SELECT MAX(price) FROM Sells
    WHERE rest = 'Majnoo''s Rest';
- “min”:
  - SELECT MIN(price) FROM Sells
    WHERE rest = 'Majnoo''s Rest';

Majnoo’s Program

- At about the same time, Majnoo executes the following SQL statements
- “del”
  - DELETE FROM Sells
    WHERE rest = 'Majnoo''s Rest';
- “ins”
  - INSERT INTO Sells
    VALUES('Majnoo''s Rest', 'Juice', 2.00);

Interleaving of Statements

- Constraints:
  - **max** must come before **min**
  - **del** must come before **ins**
- No other constraints on the order of the statements

Example: Strange Interleaving

- Suppose the steps execute in the order:
  - **max del ins min**
- What answers does Laila see?

Fixing the Problem: Transactions

- If we group Laila’s statements **max min** into one transaction:
  - Cannot see this inconsistency
  - Will see Majnoo’s prices at some fixed time

Problem: Undoing Changes

- Majnoo executes **del ins**
  - Changes his mind
  - Reverses the changes, say by **del'**, **ins'**
- Suppose the order is:
  del ins max min del' ins'
- What does Laila see?

Solution

- If Majnoo executes **del ins** as a transaction, its effect cannot be seen by others until the transaction executes COMMIT
  - Instead of **del' ins'** he uses ROLLBACK instead
  - Effects of transaction can never be seen.

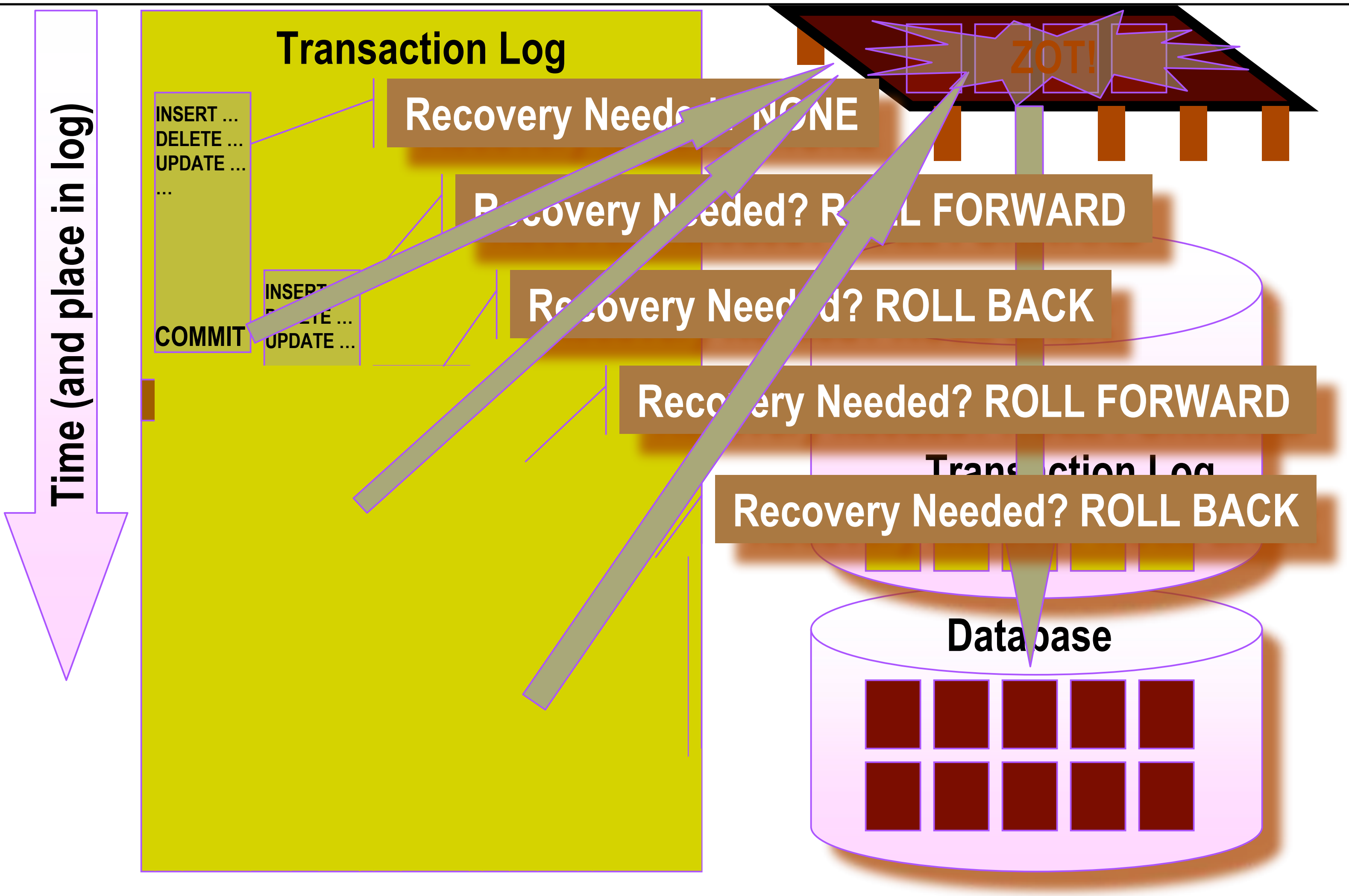
Transactions and Locks in SQL Server

- Transactions Ensure That Multiple Data Modifications Are Processed Together
- Locks Prevent Update Conflicts
  - Transactions are serializable
  - Locking is automatic
  - Locks allow concurrent use of data
- Concurrency Control

Managing Transactions (outline)

- Transaction Recovery and Checkpoints
- Considerations for Using Transactions
- Setting the Implicit Transactions Option
- Restrictions on User-defined Transactions

Transaction Recovery, Checkpoints



Time (and place in log)

Transaction Log

INSERT ...
DELETE ...
UPDATE ...
...
COMMIT

INSERT ...
DELETE ...
UPDATE ...

ZOT!

Recovery Needed? NONE

Recovery Needed? ROLL FORWARD

Recovery Needed? ROLL BACK

Recovery Needed? ROLL FORWARD

Recovery Needed? ROLL BACK

Transaction Log

Database

Considerations when Using Transactions

- Transaction Guidelines
  - Keep transactions as small as possible
  - Use caution with certain Transact-SQL statements
  - Avoid transactions that require user interaction
- Issues in Nesting Transactions
  - Allowed, but not recommended
  - Use @@trancount to determine nesting level

Implicit Transactions

- Automatically Starts a Transaction When You Execute Certain Statements
- Nested Transactions Are Not Allowed
- Transaction Must Be Explicitly Completed with COMMIT or ROLLBACK
- By Default, Setting Is Off

```
SET IMPLICIT_TRANSACTIONS ON
```

Restrictions on Transactions

- Certain Statements May Not Be Included in a Transaction:
  - ALTER DATABASE
  - BACKUP LOG
  - CREATE DATABASE
  - DROP DATABASE
  - RECONFIGURE
  - RESTORE DATABASE
  - RESTORE LOG
  - UPDATE STATISTICS

How much ACID have we done?

- Explicit transactions support **Atomicity**
- Automatic rollback on errors supports **Consistency**
- Transaction log supports **Durability**

Locks Support Isolation

Lockable Resources

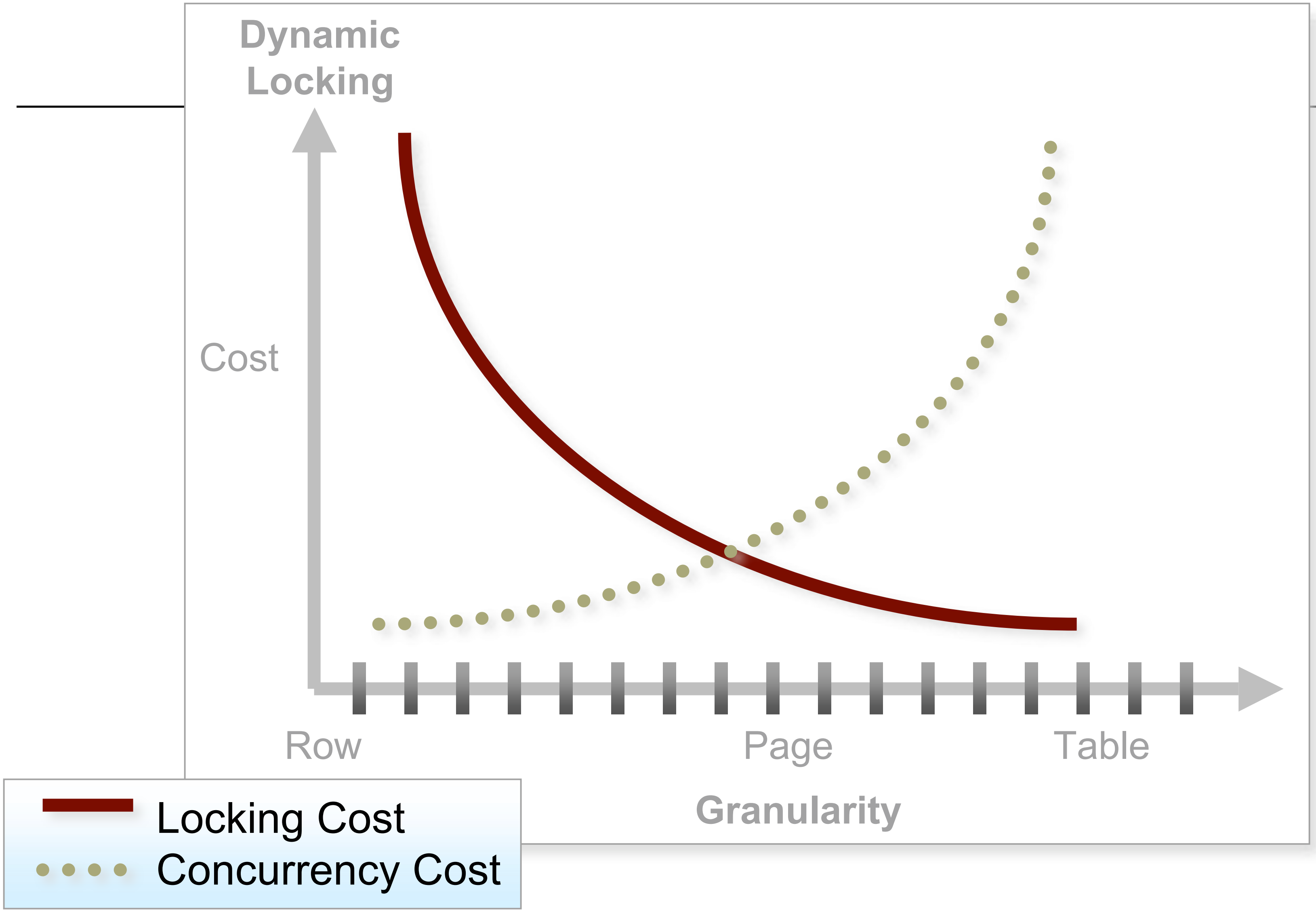
Item	Description
RID	Row identifier
Key	Row lock within an index
Page	Data page or index page
Extent	Group of pages
Table	Entire table
Database	Entire database

Types of Locks

- Basic Locks
  - Shared
  - Exclusive
- Special Situation Locks
  - Intent
  - Update
  - Schema
  - Bulk update

Lock Compatibility

- Locks May or May Not Be Compatible with Other Locks
- Examples
  - Shared locks are compatible with all locks except exclusive
  - Exclusive locks are not compatible with any other locks
  - Update locks are compatible only with shared locks


Dynamic Locking
Cost
Row
Page
Table
Granularity
Locking Cost
Concurrency Cost

Week Eight Deliverables

- Sample Reports
  - See rubric on Angel
- First draft due by Friday night (50 points)
- New versions due week nine (100 points)
- Meet with me during lab time today to agree on reports!