Module 15: Managing Transactions and Locks

Overview

- Introduction to Transactions and Locks
- Managing Transactions
- SQL Server Locking
- Managing Locks

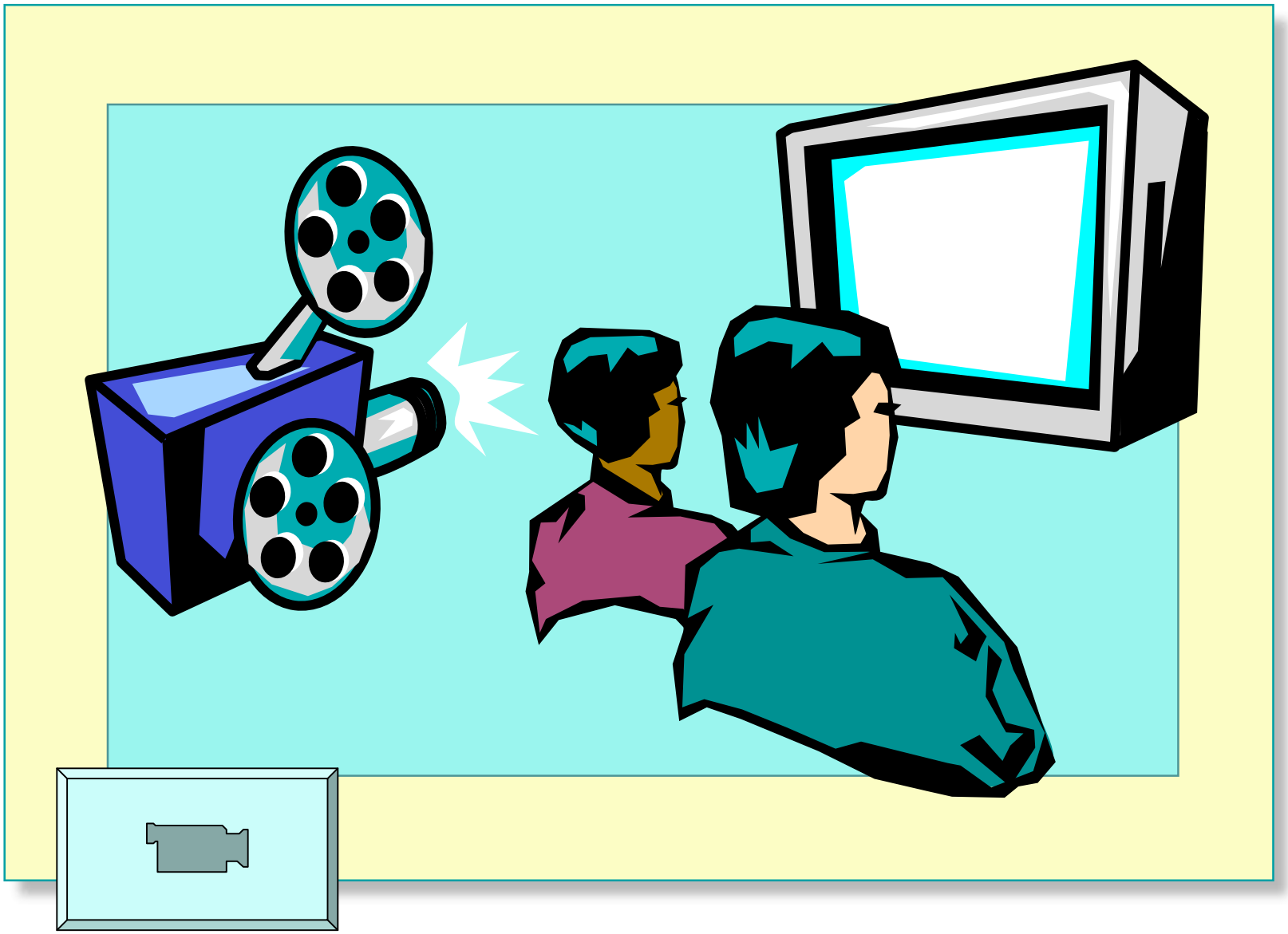
Introduction to Transactions and Locks

- **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**

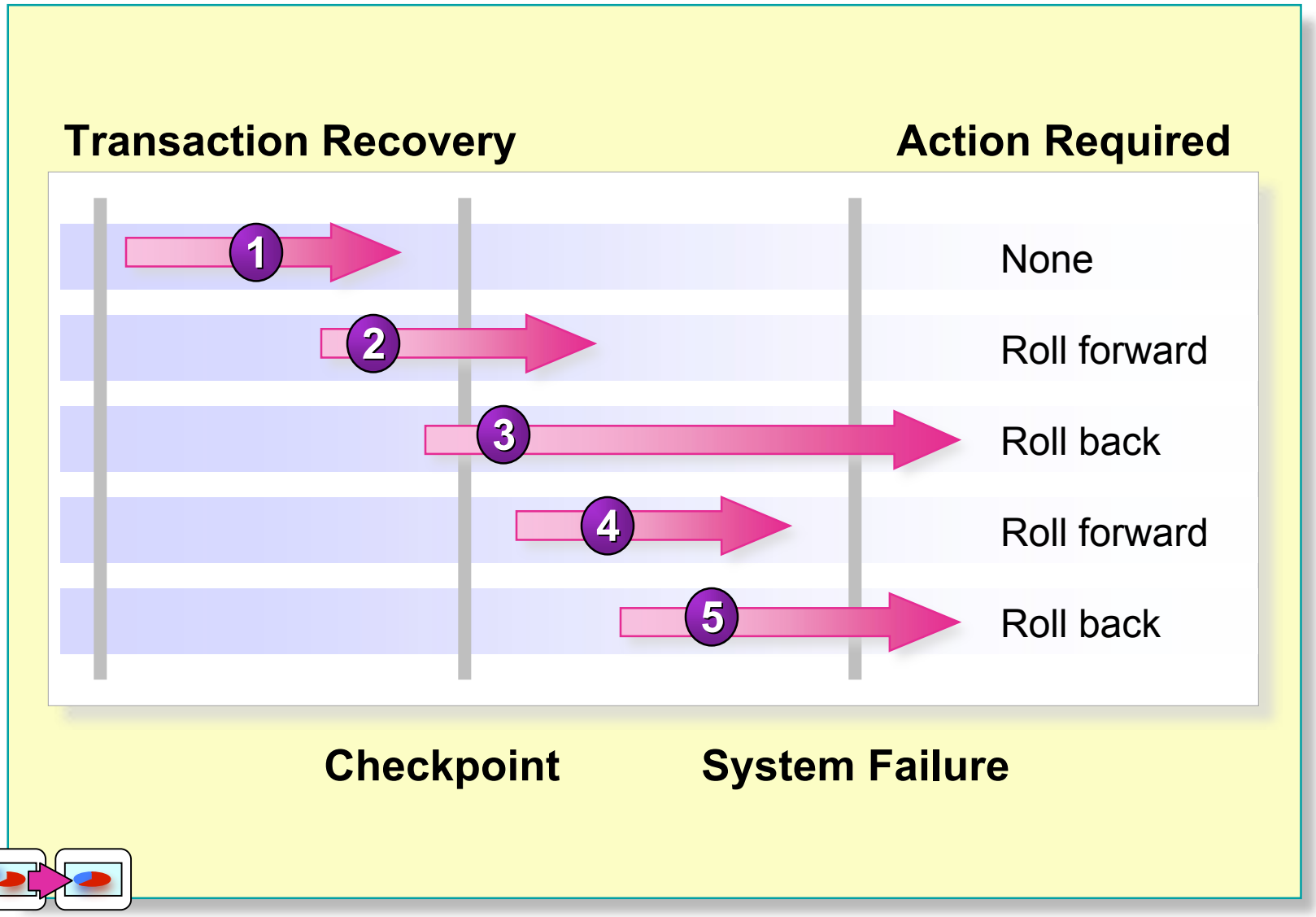
Managing Transactions

- Multimedia Presentation: SQL Server Transactions
- Transaction Recovery and Checkpoints
- Considerations for Using Transactions
- Setting the Implicit Transactions Option
- Restrictions on User-defined Transactions

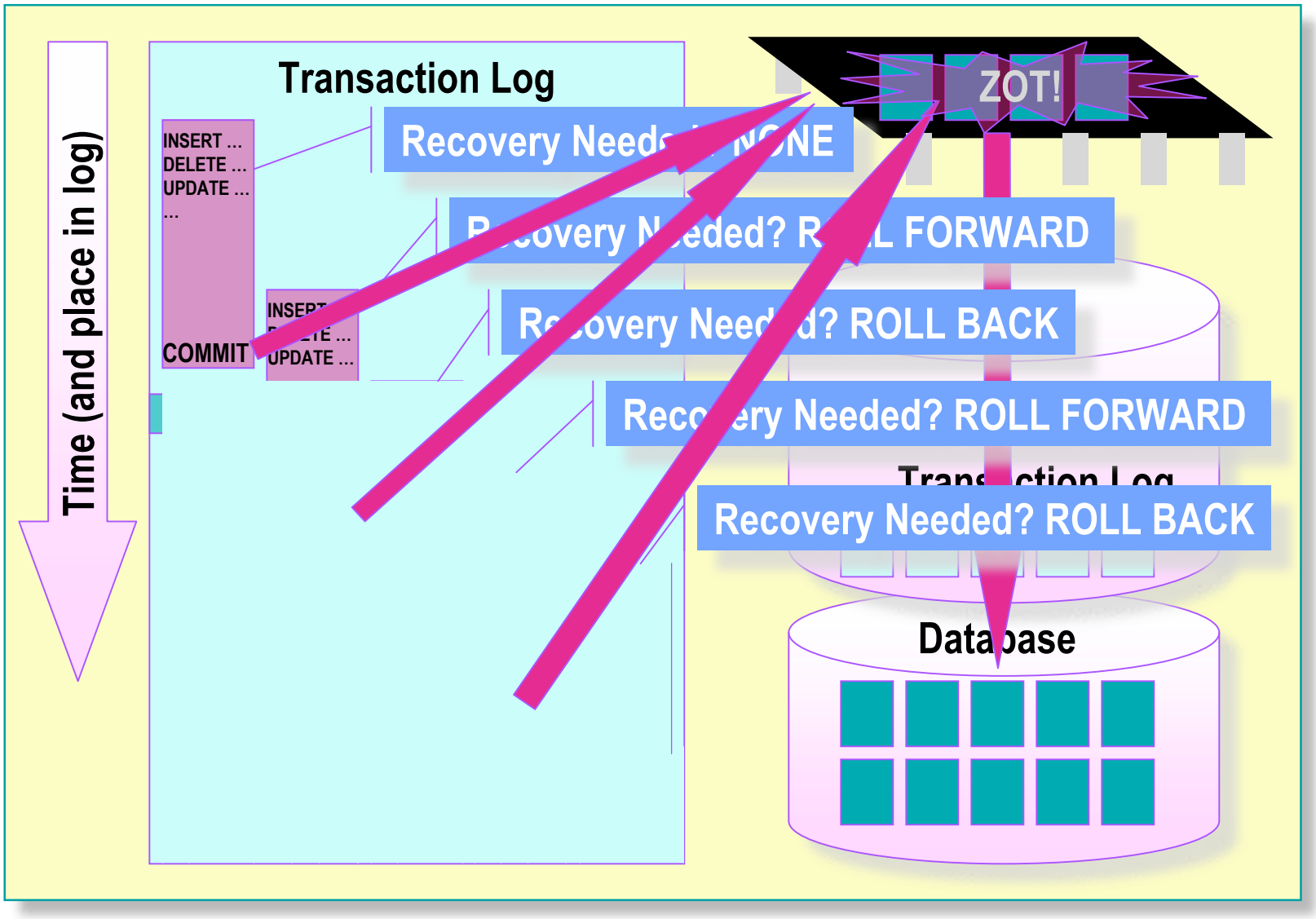
Multimedia Presentation: SQL Server Transactions



Transaction Recovery and Checkpoints



Transaction Recovery and Checkpoints



Considerations for 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

Setting the Implicit Transactions Option

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

```
SET IMPLICIT_TRANSACTIONS ON
```

Restrictions on User-defined 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

SQL Server Locking

- Concurrency Problems Prevented by Locks
- Lockable Resources
- Types of Locks
- Lock Compatibility

Concurrency Problems Prevented by Locks

- Lost Update
- Uncommitted Dependency (Dirty Read)
- Inconsistent Analysis (Nonrepeatable Read)
- Phantoms Reads

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

Managing Locks

- Session-Level Locking Options
- Dynamic Locking Architecture
- Table-Level Locking Options
- Deadlocks
- Displaying Locking Information

Session-Level Locking Options

- **Transaction Isolation Level**
  - READ COMMITTED (DEFAULT)
  - READ UNCOMMITTED
  - REPEATABLE READ
  - SERIALIZABLE
- **Locking Timeout**
  - Limits time waiting for a locked resource
  - Use SET LOCK_TIMEOUT

Dynamic Locking Architecture

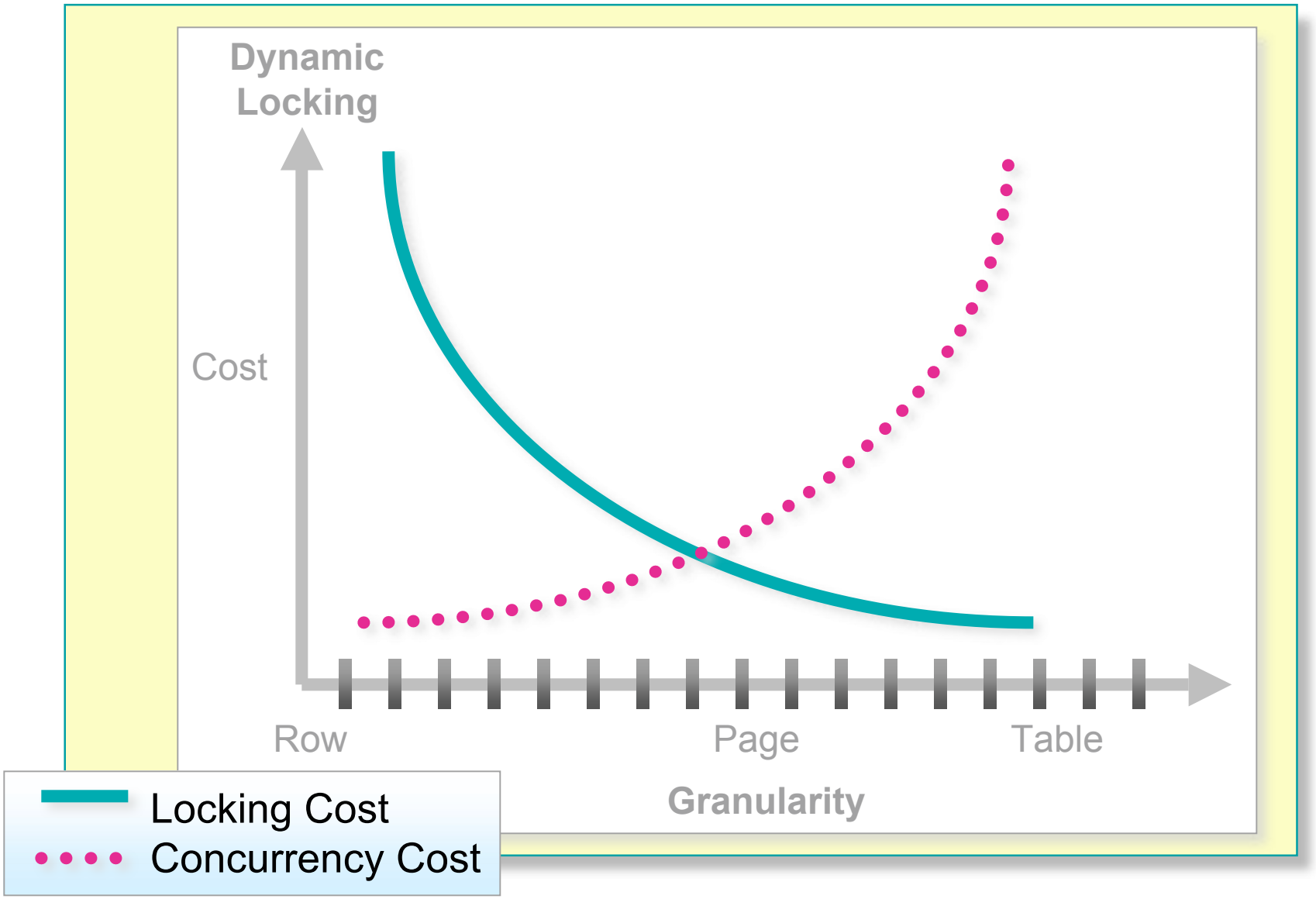


Table-Level Locking Options

- Use with Caution
- Can Specify One or More Locking Options for a Table
- Use *optimizer_hints* Portion of FROM Clause in SELECT or UPDATE Statement
- Overrides Session-Level Locking Options

Deadlocks

- How SQL Server Ends A Deadlock
- How to Minimize Deadlocks
- How to Customize the Lock Time-Out Setting

Displaying Locking Information

- Current Activity Window
- sp_lock System Stored Procedure
- SQL Profiler
- Windows 2000 System Monitor
- Additional Information

Recommended Practices

- Keep Transactions Short
- Design Transactions to Minimize Deadlocks
- Use SQL Server Defaults for Locking
- Be Careful When You Use Locking Options

Review

- Introduction to Transactions and Locks
- Managing Transactions
- SQL Server Locking
- Managing Locks