

## CSC DBA Use Cases for Understanding Locks

---

**Business Scenario** → **What happens internally** → **Lock types** → **What DBA should observe** → **Fix / Best practice**

### Use Case 1: OLTP Update Blocking a Report (Classic S vs X)

#### Business scenario

- Finance team runs a **morning overdue compliance report**
- At the same time, a batch job updates filing statuses

---

#### What happens internally

- Report query takes **Shared (S) locks**
- Update query needs **Exclusive (X) locks**
- X and S are incompatible → **blocking**

---

#### Locks involved

Level	Lock
Table	IX
Page	IX
Row	X
Reader	S (blocked)

---

#### What DBA sees

- LCK\_M\_S wait on report query
- blocking\_session\_id populated
- Long-running UPDATE holding X locks

---

#### DBA Fix

- Enable **RCSI (Read Committed Snapshot Isolation)**
- Or shorten transactions
- Or index the UPDATE to avoid large scans

---

## Key learning

Shared vs Exclusive locks cause most OLTP blocking.

---

## Use Case 2: Missing Index → Lock Escalation

### Business scenario

- UPDATE without proper index
  - SQL scans millions of rows
- 

### What happens internally

- SQL takes thousands of **row X locks**
  - Memory pressure → **lock escalation**
  - Row locks escalate to **table-level X lock**
- 

### Locks involved

Before	After
KEY X	OBJECT X
PAGE IX	OBJECT X

---

### What DBA sees

- Everything blocked
  - Many sessions waiting on LCK\_M\_X
  - sys.dm\_tran\_locks shows OBJECT-level X lock
- 

### DBA Fix

- Create proper index
  - Reduce rows affected
  - Use batching (TOP (1000) loop)
- 

## Key learning

Missing indexes don't just hurt performance — they escalate locks.

---

### ✅ Use Case 3: Long SELECT Blocking UPDATE (S Locks Problem)

#### 🧠 Business scenario

- Analyst runs long SELECT (no WHERE clause)
  - User tries to update same table
- 

#### 🔄 What happens internally

- SELECT holds **S locks for long time**
  - UPDATE waits for **X lock**
- 

#### 🔒 Locks involved

Query	Lock
SELECT	S
UPDATE	waiting for X

---

#### 🔍 What DBA sees

- LCK\_M\_X wait on UPDATE
  - SELECT session running long
- 

#### 🔧 DBA Fix

- Enable **RCSI**
  - Use READ UNCOMMITTED (only if safe)
  - Optimize SELECT (index, filter)
- 

#### 🎯 Key learning

Reads can block writes unless row versioning is enabled.

---

## ✅ Use Case 4: Schema Change Blocking Everything (Sch-M)

### 🧠 Business scenario

- DBA runs ALTER TABLE during business hours
- App users suddenly freeze

---

### 🔄 What happens internally

- ALTER TABLE requests **Schema Modification (Sch-M)**
- Blocks all queries
- Waits until all Sch-S locks are released

---

### 🔒 Locks involved

#### Lock

Sch-M (blocking)

Sch-S (blocked)

---

### 🔧 What DBA sees

- LCK\_M\_SCH\_M
- DDL waiting forever
- Application timeouts

---

### 🔧 DBA Fix

- Run DDL during maintenance window
- Use ONLINE = ON (if available)
- Kill long-running SELECTs

---

### 🎯 Key learning

Schema locks are the most dangerous locks.

---

## ✅ Use Case 5: Deadlock Between Two Transactions

### 🧠 Business scenario

- Transaction A updates Table A then Table B
  - Transaction B updates Table B then Table A
- 

### 🔄 What happens internally

- Circular lock dependency
  - SQL Server chooses a **deadlock victim**
- 

### 🔒 Locks involved

Session	Locks
A	X on A, waiting X on B
B	X on B, waiting X on A

---

### 🔧 What DBA sees

- Error 1205 (deadlock victim)
  - Deadlock graph in Extended Events
  - LCK\_M\_X waits
- 

### 🔧 DBA Fix

- Enforce consistent access order
  - Shorten transactions
  - Proper indexing
  - Use retry logic in app
- 

### 🎯 Key learning

Deadlocks are logic problems, not SQL Server bugs.

---

## ✅ Use Case 6: High Concurrency OLTP (Intent Locks Everywhere)

## Business scenario

- Many concurrent inserts/updates
  - DBAs panic seeing IX locks
- 

## What happens internally

- IX locks are **normal**
  - They signal intent to lock lower levels
- 

## Locks involved

Level	Lock
OBJECT	IX
PAGE	IX
KEY	X

---

## What DBA sees

- Many IX locks in sys.dm\_tran\_locks
  - No blocking actually happening
- 

## DBA Action

- Do nothing 😊
  - Focus only if blocking occurs
- 

## Key learning

Intent locks are healthy — don't panic.

---

## Use Case 7: Serializable Isolation → Range Locks

## Business scenario

- Compliance team uses **SERIALIZABLE** isolation
  - Inserts start blocking
- 

## What happens internally

- SQL takes **range locks**
  - Prevents phantom reads
  - Blocks inserts into range
- 

## Locks involved

- RangeS-S
  - RangeX-X
- 

## What DBA sees

- Inserts blocked even when rows don't exist yet
  - Confusing blocking patterns
- 

## DBA Fix

- Avoid **SERIALIZABLE** unless required
  - Use **SNAPSHOT** isolation if possible
- 

## Key learning

Higher isolation = stronger locks = lower concurrency.

---

## CSC DBA Locking Cheat Sheet

Symptom	Likely Cause
LCK_M_S	Writer blocking reader
LCK_M_X	Reader blocking writer
LCK_M_IX	High OLTP contention

Symptom	Likely Cause
LCK_M_SCH_M	DDL blocking
Many IX locks	Normal concurrency
Deadlock 1205	Bad transaction order

---

### Final DBA Takeaway

“Locks are not the enemy. **Long transactions, missing indexes, wrong isolation levels, and poor access patterns are the real problems.** A good DBA identifies the lock type, understands why it exists, and fixes the root cause — not the symptom.”

---