09/05/2025

azuremeetup VENETO

Meetup

Microsoft

BY COMMUNITY • FOR COMMUNITY •

#GLOBALAZURE

2025

Parallels™

ntnuovetecnologie

SERVINTEK
services for information technology

TD SYNNEX

ELEVATOR
INNOVATION HUB
FOR VISIONARY BUSINESS

Sergio Govoni

CTO

Microsoft Data Platform MVP

OPTIMIZED LOCKING IN AZURE SQL

Slide and demo: https://bit.ly/global-azure-2025-sgovoni

AGENDA

- Introduction
- Optimized locking
  - Key components
  - Underlying technologies
- How optimized locking works
- Demo

INTRODUCTION

Concurrency models and lock mode

CONCURRENCY MODELS

- A-C-I-D: The four properties of a transaction
- Pessimistic concurrency model
  - Uses blocking to avoid conflicts
  - Readers can block writers, and writers can block readers
- Optimistic concurrency model
  - Use row versioning
  - Readers cannot block writers, and writers cannot block readers, but the writer can block another writer

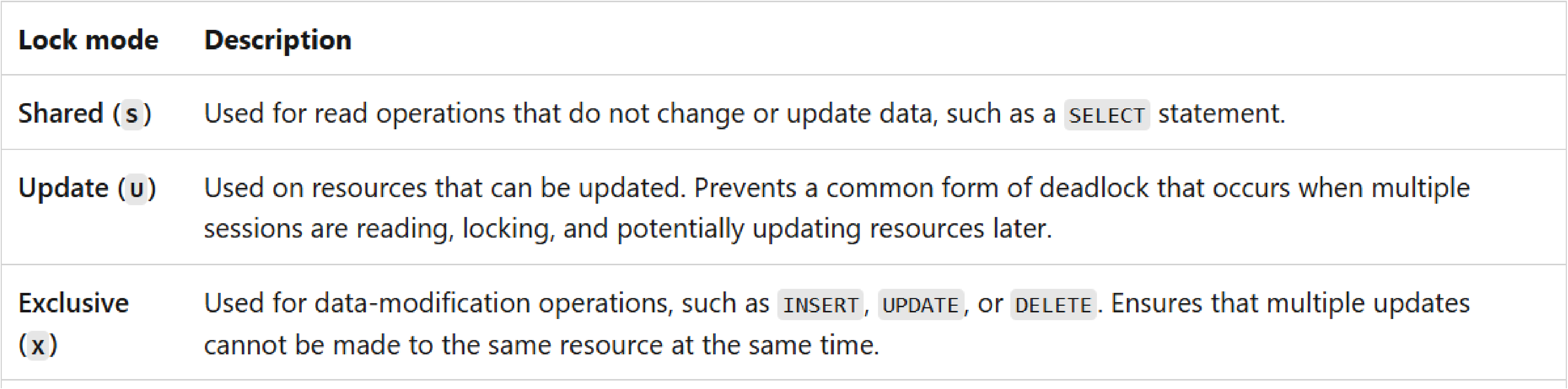
PESSIMISTIC CONCURRENCY MODELS

- Four isolation levels
  - Read Uncommitted
    - Allows the dirty read problem
  - Read Committed
    - We can only read committed data
  - Repeatable Reads
    - Acquires shared-lock until the end of the transaction
  - Serializable
    - Any transaction is waiting until the current transaction completes

OPTIMISTIC CONCURRENCY MODELS

- Two isolation levels based on row versioning
  - Snapshot
  - Read Committed Snapshot

LOCK MODES



Lock mode	Description
Shared (`S`)	Used for read operations that do not change or update data, such as a `SELECT` statement.
Update (`U`)	Used on resources that can be updated. Prevents a common form of deadlock that occurs when multiple sessions are reading, locking, and potentially updating resources later.
Exclusive (`X`)	Used for data-modification operations, such as `INSERT`, `UPDATE`, or `DELETE`. Ensures that multiple updates cannot be made to the same resource at the same time.

https://learn.microsoft.com/sql/relational-databases/sql-server-transaction-locking-and-row-versioning-guide#lock_modes

OPTIMIZED LOCKING

Key components and foundational technologies

INTRODUCTION TO OPTIMIZED LOCKING

- In the landscape of modern applications, scalability and concurrency are crucial
- Optimized locking is a new technology available in Azure SQL Database
  - It redefines how Azure SQL Database handles locks, improving concurrency and efficiency
  - It helps to reduce lock memory and avoids lock escalations

INTRODUCTION TO OPTIMIZED LOCKING

- Optimized locking is composed of two primary components:
  - Transaction ID (TID) locking
  - Lock After Qualification (LAQ)

- Transaction ID locking is designed to optimize memory usage in lock management
- Lock after qualification eliminates the risk of lock escalation and enhances concurrency in DML operations

INTRODUCTION TO OPTIMIZED LOCKING

- Optimized locking is built on two existing technologies in Azure SQL Database
  - Accelerated Database Recovery (ADR)
  - Read Committed Snapshot Isolation level (RCSI)

- Accelerated database recovery is mandatory, it must be enabled at the database level
- Read committed snapshot isolation level is not a strict requirement; it significantly enhances because LAQ is active only when READ_COMMITTED_SNAPSHOT option is enabled

ACCELERATED DATABASE RECOVERY (ADR)

- In Azure SQL Database, ADR is enabled by default
- It improves database availability, especially in the presence of long-running transactions, by redesigning the database engine recovery process
- When ADR is enabled, every row in the database internally contains a transaction ID (TID) that is persisted on disk

READ COMMITTED SNAPSHOT ISOLATION (RCSI)

- In Azure SQL Database, RCSI is enabled by default
- Read committed snapshot is not a separate isolation level, it is a modification of the read committed isolation level when the READ_COMMITTED_SNAPSHOT option is enabled
- When READ_COMMITTED_SNAPSHOT is set to ON, locks are not used to protect data from updates by other transactions, it allows reading the last committed version from the snapshot, reducing contention between reads and writes

HOW IT WORKS

TID locking and LAQ in action

TRANSACTION ID (TID) LOCKING IN ACTION

- With TID locking
  - Each row in the database internally contains a TID
  - TID is persisted on disk, and every transaction modifying a row assigns its own TID to that row
  - Instead of acquiring a lock on the row’s key, a lock is taken on the row’s TID

TRANSACTION ID (TID) LOCKING IN ACTION

- With TID locking
  - Page and row locks are still taken during modifications, but each page and row lock is released immediately after modification
  - The only lock held until the end of the transaction is the single X lock on the TID resource, replacing multiple page and row (key) locks

LOCK AFTER QUALIFICATION (LAQ) IN ACTION

- One major cause of DML slowdowns is acquiring locks while searching for qualifying rows
- Lock after qualification modifies the way DML statements acquire locks
- Without optimized locking, queries evaluate predicates row by row, first acquiring a U lock, which is upgraded to an X lock if the row meets the condition. The X lock remains until the transaction ends

LOCK AFTER QUALIFICATION (LAQ) IN ACTION

- With LAQ, predicates are evaluated on the latest committed row version without locks. If the condition is met, an X lock is acquired for the update and released immediately after
- This prevents blocking between concurrent queries modifying different rows

DEMO

SUMMARY

..and resources

SUMMARY

- Optimized locking in Azure SQL Database represents a significant evolution in concurrency management
- By using Transaction ID (TID) locking and Lock After Qualification (LAQ), optimized locking reduces memory consumption and eliminates the lock escalation, minimizing locks between concurrent transactions
- In Azure SQL Database, optimized locking is enabled by default

RESOURCES

- Optimized Locking in Azure SQL Database: Concurrency and performance at the next level
- Optimized Locking in Azure SQL Database: Concorrenza senza limiti
- Understanding Optimized Locking in Azure SQL Database

- UGISS - https://www.ugiss.org/

Q&A

Thank you!

https://segovoni.medium.com

https://www.linkedin.com/in/sgovoni

https://vimeo.com/ugiss/videos/

https://bit.ly/sgovoni-MVP