SQL Server Locks

Microsoft SQL Server uses locking to ensure the integrity of transactions and consistency in the database. Locking prevents data that users are reading from being changed by other users, and prevents multiple users from simultaneously changing the same data. If locking is not used, data within a database may become incorrect, and queries executed against that data may produce unexpected or invalid results. SQL Server automatically applies locks. However, you can make your applications more efficient by customizing or minimizing database locking.

The SQL Server Locks object provides information about locks that are applied to individual resource types. Locks are held on SQL Server resources such as rows read or rows that were modified during a transaction, and prevent the concurrent use of a resource by multiple transactions.

SQL Server can lock the following resources:

Resource	Description
Database	A database
Extent	A contiguous group of eight data pages or index pages.
Key	A row lock within an index.
Page	An eight kilobyte data page or index page.
RID	The identifier for a row in the database.
Table	An entire table in the database, including all data and indexes.

SQL Server locks resources using the following resource lock modes:

- **Shared (S) Locks**
  Allow concurrent transactions to read a resource. No other transactions can modify the data while shared (S) locks exist on the resource. Shared (S) locks are released when the data has been read, unless:
  - The transaction isolation level is set to repeatable read or higher.
  - A locking hint is used to retain the shared (S) locks for the duration of the transaction.
- **Update (U) Locks**

  Prevent a common form of deadlock. A typical update pattern consists of a transaction:

  - Reading a record.
  - Acquiring a shared (S) lock on the resource.
  - Modifying the resource, which requires conversion to an exclusive (X) lock.

  If two transactions acquire shared-mode locks on a resource and then attempt to update data concurrently, one transaction attempts to convert the lock to an exclusive (X) lock.

  The shared-mode-to-exclusive lock conversion must wait because the exclusive lock for one transaction is not compatible with the shared-mode lock of the other transaction. In this case, a lock wait occurs.

  The second transaction attempts to acquire an exclusive (X) lock for its update. Because both transactions are converting to exclusive (X) locks, and they are each waiting for the other transaction to release its shared-mode lock, a deadlock occurs.

  Update (U) locks are used to avoid this problem. Only one transaction can obtain an update (U) lock to a resource at a time. If a transaction modifies a resource, the update (U) lock is converted to an exclusive (X) lock. Otherwise, the lock is converted to a shared-mode lock.
- **Exclusive (X) Locks**
  Prevent access to a resource by concurrent transactions. No other transactions can read or modify data locked with an exclusive (X) lock.
- **Intent Locks**

  Indicates that SQL Server wants to acquire a shared (S) lock or an exclusive (X) lock on some of the resources lower in the hierarchy. Intent locks include intent shared (IS), intent exclusive (IX), and shared with intent exclusive (SIX).

  For example, a shared intent lock placed at the table level means that a transaction intends to place shared (S) locks on pages or rows within that table. Setting an intent lock at the table level prevents another transaction from acquiring an exclusive (X) lock on the table containing that page.
- **Schema (Sch) Locks**

  Used when a table data definition language (DDL) operation such as adding a column or dropping a table is being performed. Schema stability (Sch-S) locks are used when compiling queries and do not block any transactional locks, including exclusive (X) locks. Other transactions can continue to run while a query is being compiled, including transactions with exclusive (X) locks on a table. However, DDL operations cannot be performed on the table.
- **Bulk Update (BU) Locks**

  Used when bulk copying data into a table and either the `TABLELOCK` hint is specified, or the Table Lock on Build Load table option is set using `sp_Tableoption`. BU locks allow processes to bulk copy data concurrently into the same table while preventing other processes that are not bulk copying data from accessing the table.

Related Documentation

Configuring SQL Server Ports

Verifying the Configuration of SQL Server Ports