rollback is an operation which returns the database to some previous state. Rollbacks are important for database integrity, because they mean that the database can be restored to a clean copy even after erroneous operations are performed. They are crucial for recovering from database server crashes; by rolling back any transaction which was active at the time of the crash, the database is restored to a consistent state.

The rollback feature is usually implemented with a transaction log, but can also be implemented via multiversion concurrency control.

Cascading rollback

A *cascading rollback* occurs in database systems when a transaction (T1) causes a failure and a rollback must be performed. Other transactions dependent on T1's actions must also be rollbacked due to T1's failure, thus causing a cascading effect. That is, one transaction's failure causes many to fail.

Practical database recovery techniques guarantee cascadeless rollback, therefore a cascading rollback is not a desirable result.

SQL

In SQL, `ROLLBACK` is a command that causes all data changes since the last `BEGIN WORK`, or `START TRANSACTION` to be discarded by the relational database management systems (RDBMS), so that the state of the data is "rolled back" to the way it was before those changes were made.

A `ROLLBACK` statement will also release any existing savepoints that may be in use.

In most SQL dialects, `ROLLBACK`s are connection specific. This means that if two connections are made to the same database, a `ROLLBACK` made in one connection will not affect any other connections. This is vital for proper concurrency.

See also

- Savepoint
- Commit
- Undo
- Schema migration