

What happens if you don't commit a transaction to a database (say, SQL Server)?

Asked 13 years, 10 months ago Modified 1 year, 11 months ago Viewed 202k times



Suppose I have a query:

149

```
begin tran
-- some other sql code
```



And then I forget to commit or roll back.



If another client tries to execute a query, what would happen?

sql-server

transactions

commit

Share Follow

edited Feb 23, 2018 at 13:43



DavidRR

19.3k ● 27 ● 111 ● 196

asked Feb 4, 2011 at 9:36

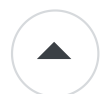


Charbel

14.7k ● 12 ● 46 ● 67

10 Answers

Sorted by: Highest score (default)



As long as you don't **COMMIT** or **ROLLBACK** a transaction, it's still "running" and potentially holding locks.

215

If your client (application or user) closes the connection to the database before committing, any still running transactions will be rolled back and terminated.



Share Follow

edited Jul 1, 2019 at 12:23



Top-Master

8,669 ● 5 ● 47 ● 85

answered Feb 4, 2011 at 9:44



marc_s

753k ● 183 ● 1.4k ● 1.5k

2 mmm, ok I figure out this was creating some sort of lock. I wasn't sure that closing the connection would actually get me out of this state. the problem was that I was getting an error when I try to commit. now I closed the connection and it all worked. – Charbel Feb 4, 2011 at 11:49

19 Side note: If using Management Studio, closing the query window will close the connection – Joe Phillips Dec 10, 2012 at 17:39

3 @BradleyDotNET: yes, definitely – [marc_s](#) Sep 27, 2014 at 6:24

4 Keep in mind that SQL Server Management Studio auto commits if you close the query window/connection, by default. – [Nuno](#) Sep 11, 2015 at 9:37

2 everyone should take care of the "potentially holding locks" part of this reply. – [fiorentinoing](#) Jul 6, 2018 at 7:28 ✎

51

You can actually try this yourself, that should help you get a feel for how this works.

Open two windows (tabs) in management studio, each of them will have it's own connection to sql.

Now you can begin a transaction in one window, do some stuff like insert/update/delete, but not yet commit. then in the other window you can see how the database looks from outside the transaction. Depending on the isolation level, the table may be locked until the first window is committed, or you might (not) see what the other transaction has done so far, etc.

Play around with the different isolation levels and no lock hint to see how they affect the results.

Also see what happens when you throw an error in the transaction.

It's very important to understand how all this stuff works or you will be stumped by what sql does, many a time.

Have fun! GJ.

Share Follow

edited Dec 29, 2022 at 16:48

answered Feb 4, 2011 at 10:15



[gjvdkamp](#)

10.5k ● 4 ● 39 ● 53

1 ok but will the transaction be written to log at very least before issuing the commit? For example, say I want to start a transaction run an insert command and "do something else" before I execute commit. will my insert command be written to log? that way if the server crashes before executing commit..it can come back to where it was and I can just issue commit later(whenever I am done doing "something else"). – [user1870400](#) Apr 21, 2019 at 21:26

1 @user1870400 It seems obvious that the answer this user would have given is... try it. – [Daniel](#) Apr 29, 2021 at 23:41

Transactions are intended to run completely or not at all. The only way to complete a transaction is to commit, any other way will result in a rollback.

24

Therefore, if you begin and then not commit, it will be rolled back on connection close (as the transaction was broken off without marking as complete).



Share Follow

answered Feb 4, 2011 at 9:41



[Piskvor left the building](#)

92.7k ● 46 ● 179 ● 225

1 That's how it should be, but it's not always the case. – [FalcoGer](#) Jul 12, 2019 at 9:19

...such as mySQL's MyISAM, which doesn't *support* transactions, sure.

– [Piskvor left the building](#) Jul 12, 2019 at 9:23



5

When you open a transaction nothing gets locked by itself. But if you execute some queries inside that transaction, depending on the isolation level, some rows, tables or pages get locked so it will affect other queries that try to access them from other transactions.



Share Follow

answered Feb 4, 2011 at 9:45



[red.clover](#)

1,828 ● 2 ● 20 ● 32



4

depends on the isolation level of the incoming transaction.

[Sql transaction isolation explained](#)



Share Follow

answered Feb 4, 2011 at 9:40



[Xhalent](#)

3,944 ● 23 ● 21

7 The behavior of the transactions doesn't depend on the isolation level. The amount of locks they might cause does. – [marc_s](#) Feb 4, 2011 at 9:45

I'm pretty sure what data is able to be read by a connection is definitely dependant on the isolation level. If you have the isolation set to READ UNCOMMITTED you can read data not yet committed and may in fact be rolled back at some point the track, but this ensures there is no locking. If you have READ COMMITTED as your isolation level, then you can't read uncommitted rows - the second client will hang unless you use SNAPSHOT. – [Xhalent](#) Feb 4, 2011 at 10:39



Example for Transaction

1 begin tran tt


▼ Your sql statements

🔖 if error occurred rollback tran tt else commit tran tt

🕒 As long as you have not executed commit tran tt , data will not be changed

Share Follow

answered Mar 6, 2014 at 4:10

 [user3386471](#)
11 ● 1

1 Note that naming transactions is not only unnecessary in MS SQL, it can give a false sense of control. `BEGIN TRAN X ... BEGIN TRAN Y ... ROLLBACK Y` does not work, for example. See [stackoverflow.com/questions/1273376/...](https://stackoverflow.com/questions/1273376/) – user565869 Mar 12, 2015 at 20:39

▲ Any uncommitted transaction will leave the server locked and other queries won't execute on the server. You either need to rollback the transaction or commit it. Closing out of SSMS will also terminate the transaction which will allow other queries to execute.

1

▼

🔖 Share Follow

answered Mar 17, 2016 at 20:34

 [Josh Moorish](#)
113 ● 10

In my case I only closed tab(file), where was query with executed transaction, and SSMS asked me if I want to commit transaction. – [Lazar Đorđević](#) Jul 29, 2022 at 18:50

▲ In addition to the potential locking problems you might cause you will also find that your transaction logs begin to grow as they can not be truncated past the minimum LSN for an active transaction and if you are using snapshot isolation your version store in tempdb will grow for similar reasons.

0


▼

🔖 You can use `dbcc opentran` to see details of the oldest open transaction.

🔖

🕒 Share Follow

answered Feb 4, 2011 at 10:20

 [Martin Smith](#)
452k ● 94 ● 767 ● 870

▲

I really forget to commit a transaction. I have a query like codes below.

0

This stored procedure is called by .Net. When I test the function in .Net application, the exception will be captured in .Net application.



Exception message like below:

Transaction count after EXECUTE indicates a mismatching number of BEGIN and COMMIT statements. Previous count = 0, current count = 1.

When I realize the mistake, I have tried many times, both in .Net application and SQL Server Management Studio (2018). (In SSMS, the output statement will successfully output the result in **Results** tab, but shows the error message in **Messages** tab.)

Then I find the tables used in this transaction are locked. When I only select top 1000 without **order desc**, it can select the result. But when I select top 1000 with **order desc**, it will be running for a long time.

When I close the .Net application, the transaction was not committed (based on the data not changed in the transaction).

When I close the **EXEC ...** tab (which execute the forged commit query), SSMS will pop a warning window:

There are uncommitted transactions. Do you wish to commit these transactions?

I have tested the both the **Yes** and **No** choices.

If I click **yes**, the transactions are committed.

If I click **No**, the transactions aren't committed.

After I close the tab, my locked table will be released, then I can query successfully.

```
begin try
  -- some process
  begin transaction
  update ...
  output ...

  insert ...

  -- I missing this commit statement below
  commit transaction
end try
begin catch
  if (xact_state()) = -1
  begin
```

```
        rollback transaction;
        ;throw
    end;

    -- this statement I want to compare to 1, but mistake write to -1, but
    since the throw statement let the mistake can't be triggerd
    if (xact_state()) = 1
    begin
        commit transaction;
    end;
end catch;
```

Share Follow

edited Aug 23, 2022 at 8:33

answered Aug 23, 2022 at 8:24



Jun Yu

424 ● 1 ● 8 ● 22



-4



The behaviour is not defined, so you must explicit set a commit or a rollback:

http://docs.oracle.com/cd/B10500_01/java.920/a96654/basic.htm#1003303

"If auto-commit mode is disabled and you close the connection without explicitly committing or rolling back your last changes, then an implicit COMMIT operation is executed."

Hsqldb makes a rollback

```
con.setAutoCommit(false);
stmt.executeUpdate("insert into USER values ('" + insertedUserId +
"', 'Anton', 'Alaf')");
con.close();
```

result is

```
2011-11-14 14:20:22,519 main INFO [SqlAutoCommitExample:55]
[AutoCommit enabled = false] 2011-11-14 14:20:22,546 main INFO
[SqlAutoCommitExample:65] [Found 0# users in database]
```

Share Follow

edited Nov 14, 2011 at 14:39

answered Nov 14, 2011 at 13:20



LarsTech

81.5k ● 14 ● 158 ● 231



Bernd Schatz

1 ● 1

2 This may be true for Oracle (I have no idea), but questioner is asking about MS-SQL – [PaulG](#) Nov 2, 2012 at 13:24

The first quote applies to the JDBC driver, not to the server. – [djechlin](#) Sep 17, 2013 at 20:23