Troubleshooting Commit Failures

by Mike Wright, Werner Co.

Editor's Note: There's nothing clients love more than those cryptic error messages that say, "something's wrong but I'm not going to tell you what". Mike Wright helps us get on the track of hunting down the root cause of the occasionally mystifying commit failure errors. He gives us not one, not two, but three places to look for answers.

Probably one of the more frustrating and difficult issues to deal with can be commit failures from within applications. I've seen top-notch application leads throw up their hands in despair when seeing these errors. These errors typically appear as a generic error on screen and they don't always show up immediately after completing a task. It could be minutes before you see the error, making it even tougher to deal with.

Commit failures are quite often data related but that does not make it any easier to handle. I'll look at three of the more common things that can cause these errors (Table problems, Row security problems, and Timeout problems) and explain each one. There are other reasons that can cause commit failures, but the three areas I will focus on will give you a head start on a good portion of these errors.

What Is A Commit Failure?

To understand a commit failure, you must first understand the concept of transaction processing. Transaction processing is used to maintain data integrity. A set of related transactions is treated as one entity or unit. What that means is that either the entire set gets written to the database tables or none of it does. For example, you would want the header and detail of a new transaction to be paired up, in a sense, upon entry. Either they both exist or neither exists. Transaction processing treats them as one unit.

If something causes a set of transactions to fail, you get a commit failure error message and the entire set is rolled back. It's as if the set of transactions was never created at that point. Primarily in JD Edwards you will see this kind of behavior with the major business functions such as sales order entry, ship confirm, purchase order entry. It is something that has to be set up however by a developer. There are check boxes on business functions that developers can use to make this functional so it's not necessarily something used everywhere in the system.

Here's an example of what I'm talking about:

You enter a sales order and when you click the Place Order button or the final OK at the end of the process, the sales order entry master business function "end doc" is kicked off. That business function runs asynchronously, which means that it runs on its own either locally or on a server, allowing you to go on to other applications. The business function uses transaction processing so the records involved for this transaction are committed, or locked, until the process either completes or fails. If it fails, you will get an error similar to the one shown in Figure 1.

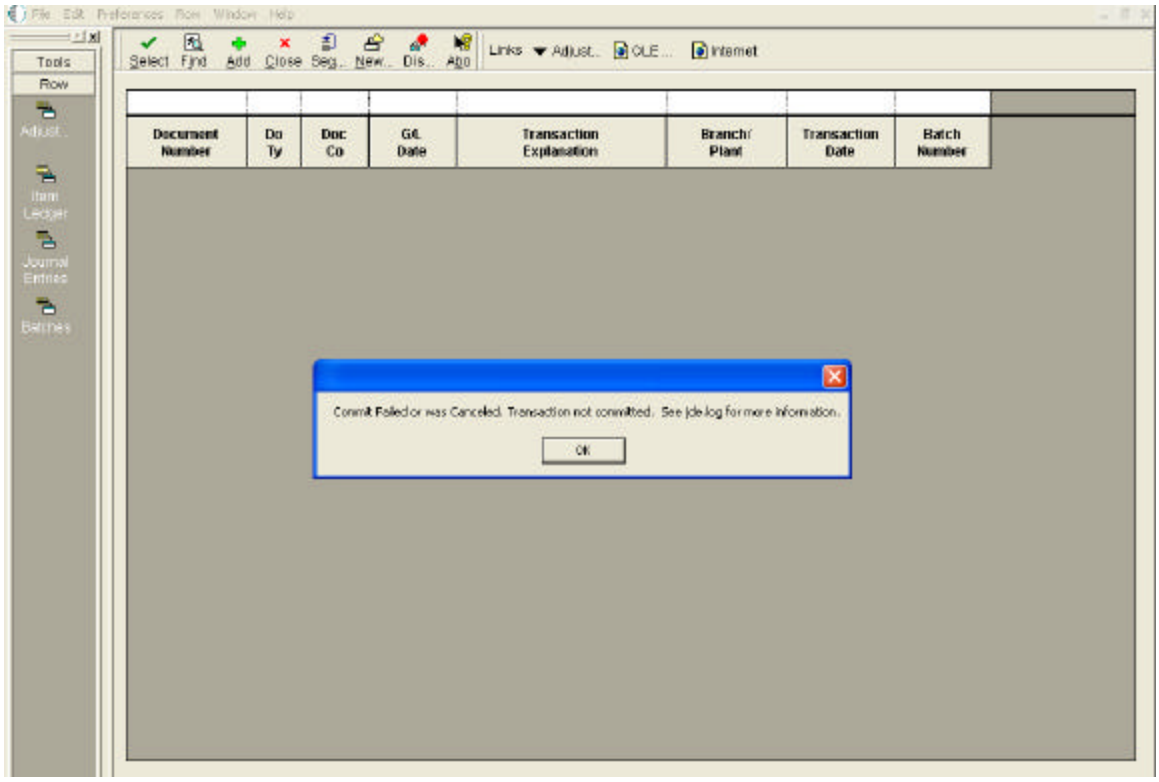

Figure 1 - Commit Failure Error

If you see that, then you won't find the sales order number in the header or the detail. It will be as if entry was cancelled in mid-stream. You will have used up a next number but that's about it. If the process completes, then you will see all of the proper information in the tables.

It's important to note that you could be in another application, even one unrelated to your previous transaction, and still get this message minutes later. It just depends on how much processing is involved before the error occurs.

As you can see from Figure 1, the system is generating a pretty generic error message. It doesn't give you much to go on, so here are three areas to start looking, and the order in which I often will investigate them.

1. Table problems
2. Row security
3. Timeout value

1. Table Problems

The first thing I do with a commit failure is actually follow the advice of the error message. It is good advice in this case. I immediately go to the jde.log looking for table error messages. You will get these messages even if the debug logging is not turned on. In Figure 2, you will see an example of a jde.log with a commit failure message.