

Re: Help – Timing Logic

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2006-12/msg00001.html>

- *From:* "Jay" <someone@xxxxxxxxxxxxxx>
 - *Date:* Tue, 28 Nov 2006 11:09:19 -0500
-

Thanks. I do need to select an individual row at a time and all 4 threads need to do this. What transaction isolation level would you recommend? Perhaps a stored proc may be faster to execute and return the values as opposed to building the transaction in the code. What has to happen is every 5 seconds, and for each thread, a sub runs to get a single row then send the message to `dtr("numbertosendto")`. Because this app heavily relies on timing it is important that all threads run and only one distinct row can be returned at a time for each thread.

"jeff" <jhersey at allnorth dottt com> wrote in message
<news:%23qVFIYwEHHA.3576@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>

are you selecting individual rows at a time from the database table ...

if so ... use a transaction ...

begin transaction ...

select a row from database ...

update the row in table set flag = 'Processed'

end transaction

this will lock the row until the end of transaction is issued ... as long as the isolation level is set accordingly...

Again, please let us know how you are getting the information from the database... then we can help!

If you are reading a bunch of rows in one statement, storing the rows in in-memory datasets on the workstation, looping through the rows one by one ... then you may need to either ...

implement as above locking only the records you retrieve / update – need to watch out here for table locking ... may impact performance,
implement using an update flag at sent and 'check before send method'...

...

Re: Help – Timing Logic

lots of options here, just need to know how you are retrieving your data...and how you are processing the data.

Question, why do you need 4 threads running ... are they doing 'different processes', sending different 'types' of messages ... sending the same message ... just need 4 to make it faster ... what is the logic ...

Tieing a record to a thread my cause problems in the future ... what happens when a thread stops or hangs ... those messages will not be processed... When happens if somebody changes the ThreadID ... in the program ...

trying to help ..

Jeff

PS: you can lock individual rows ... look at how database transactions work and incorporate it in you program ...

"Jay" <someone@xxxxxxxxxxxxxx> wrote in message
news:%23f70LJwEHHA.2328@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Thanks for the reply.

I do need all 4 threads running (maybe even more in the future). I can not delete the row from the table... it needs to be there for later update use. I am considering marking each row at insert (1 – 4) then having each of the 4 threads only select a row based on the mark. It's becoming a little tricky... it would be great if I could lock a single row when selecting it.

"jeff" <jhersey at allnorth dottt com> wrote in message
news:OPCyvDwEHHA.4740@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Question:

What is the reason for the 4 threads?

Do you need all these threads processing the same dataset?

Just asking, maybe you could change your approach to avoid this problem...

If not, investigate Record Locking / Transactions...

Re: Help – Timing Logic

How are you retrieving your records in to memory for processing ...

ie,

– are you selecting on the entire contents of a 'message' table and looping through an in-memory dataset...

– do you retrieve 'chunks' or rows from the message table based on parameters...

You could use the internal locking mechanisms of the database to achieve this ... but procede with caution...

Start Transaction...

```
SELECT * FROM MESSAGE TABLE
```

...

stuff this into a data table on the desktop

...

```
DELETE * FROM MESSAGE TABLE
```

...

punt / clean the table.

...

End Transaction.

The transactions places locks on the table ... but selecting the entire tables contents ... you are essentially 'LOCKING' the entire table until the transaction is over...

So, each thread's 'retrieve' process will have to wait in line until the select and delete are processed. As well, anything triggering 'New Messages to be Added' will be delayed until the Transaction is completed...

You may impact the overall performance of you application ... need to investigate.

If locking does not work for you, you will need to implement some type of logging / checking approach...

ie... have a log table...once a process has sent the message, write to a log table ... message sent. Before each message is compiled and sent, check the log table to see if another process has sent the message...if not, send you message...

A snip of code would be great to figure out how you are retrieving a list of messages and how you are processing each record...

Jeff.

Re: Help – Timing Logic

"Jay" <someone@xxxxxxxxxxxxxx> wrote in message
news:O2phzwvEHHA.4280@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

I have a multi threaded VB.NET application (4 threads) that I use to send text messages to many, many employees via system.timer at a 5 second interval. Basically, I look in a SQL table (queue) to determine who needs to receive the text message then send the message to the address. Only problem is, the employee may receive up to 4 of the same messages because each thread gets the recors then sends the message. I need somehow to prevent this... just can't think of how. Somehow I need the other threads to know that another thread is already using that record and move on to the next record. I thought of getting the record then marking it (column value) as in use and writing the code to look for records only where not in use... problem is all 4 run fast enough to all use/mark the row. Any thoughts?

Thanks a lot.

Jay