

Re: Help – Timing Logic

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2006-12/msg00117.html>

- *From:* "Jay" <someone@xxxxxxxxxxxxxx>
 - *Date:* Thu, 30 Nov 2006 14:21:15 -0500
-

Thanks a lot everyone... I'll re-read all messages and work this out. I appreciate all the help and assistance. I'm sure with your suggestions everything will get resolved. Thanks a lot.

"jeff" <jhersey at allnorth dottt com> wrote in message
<news:umKysi8EHHA.4016@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>

Jay,

Cant have a trigger execute a function in a remote vb.net app.

Install the application on you DATABASE SERVER– make it a local APP... If the messages are that important ... this will give you the quickest result! A function CAN call a local exe ... I do it all the time – for broadcasting status changes and sending critical alerts in an Patient Tracking system for a hospital emergency department...works like a hot dam ... even sends messages to peoples pagers! If you DBA screams and yells and jumps up and down – this is a database server, we are not install you little program on it ... find out the persons name at the bottom of the DBA's paycheck, go to them, tell them what is going on ... tell them what you are doing ... tell them how critical this application / process is to organization – MISSION CRITICAL ... tell them how much this program will impact the overall performance of the database server ... sell it to them – the persons whos name is at the bottom of the paycheck!

Again, if this is so mission critical, you must capture it at the moment the data is made available...and you need fall-over, redundancy in you messaging application.

what happens ... message in table ... 3 more seconds til the messaging thread is fired ... 2 ... 1 ... oops, messaging machine is off-line! No message sent! Now what? If on the database server ... if the data is saved, the message is sent ... and if it is not sent ... alerts are sent... there is a fail proof plan in place when messages are not SENT simply because the system knows a message was suppose to be sent! However, if you are sending messages for a machine that is off line and not polling for new messages (because some network guy re-started the box and for got to restart your process) ... there is now way of detecting unsent messages.

Re: Help – Timing Logic

What I do not understand ... is your reason to use '4' threads??? I know speed ... mission critical . But why 4 threads??? Why not have a thread for EACH MESSAGE ... 16 messages 16 threads! Using your reasoning, would this not be even quicker!

What I am saying is ... and have said before ...and is being repeated by Chris ... he has taken my suggestion one step further by giving you useable code... is you approach to multi threading the entire process is flawed ... I am not argueing the need for multithreading in the process ... I am just disagreeing with the need to multithread the entire process!

Example ... in plain text ... you figure out the code ...

1. Scenerio: Retrieve list of messages from the database... MessageBroker ... have MessageBroker create a seperate thread for each message it has to send!

So, proces is...

- Get list of message from database ...
- Loop through list ...
- for each record ... call a SendMessage method ... include all the necessary information (parameters) for selecting or sending the message
- Have the SendMessage start a new thread ... send the message information to this new thread ... so its know which message to send ... have this new thread send the message.
- while this thread is processing (sending the message) ... return to your list of messages in the messagebroker ...
- messagebroker gets next message ... messagebroker calls sendmessage ... sendmessage creates a new thread ... sendmessage send message in new thread, while program returns to the loop ...

Do you see what I am doing here ... the prgram will create a seperate thread for EACH message. The messagebroker is your control, it will ensure you do not send duplicate messages...

This will only work, if you have a single machine designated as your MessageBroker!

2. Scenerio – Use your 4 threads ...

x = 1 to 4

- Thread x retrieves a message from the database ...
- Thread x prepares message to send ... before it sends ... it checks the database ... to see if any other thread has sent the message ...
YES another thread has sent it ... Thread x , drops message, and gets another message ...

Re: Help – Timing Logic

NO, nobody has sent, thread x locks the message row in the database (transactions and isolation levels ... have you looked at these yet), updates the sending flagging ... sends the message ... updates the sent flag...

– Thread x gets another message ...

This process ... will create a lot of wasted CHECKING and UPDATING and DATABASE LOCKING to ensure messages are not duplicated between recipients. This is not faster!

By having 4 threads working independant of each other, you are slowing down the process! and adding 'more threads', you could be compound the issue!

You need some mechanism for communicating between your threads ...

- an exe ... the messagebroker approach ... telling the thread which message to send.
- the database ... implement row level locking ... look at Isolation and Transactions ... to ensure each message row in only processed by 1 thread...
- hard code it ... implement a design structure – add the ThreadToProcess field – in to your solution. This will cause maintenance and scalability issues later in life! And is poor design...and a poor work-around!

So, Jay, I have tried to shine some light on the problem for you, it is up to you to decide your next steps ... if this is so mission critical, and you are having these difficults, maybe consultant out for a solution :-)

Jeff.

"Jay" <someone@xxxxxxxxxxxxxx> wrote in message
[news:OGJfd9xEHHA.3188@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:OGJfd9xEHHA.3188@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Thanks Jeff, answers under your questions...

"jeff" <jhersey at allnorth dottt com> wrote in message
news:uekJE1xEHHA.2356@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

I know it is a time critical system ... employees need the information ASAP... but what I do not understand is ...

– where is the application running ... on a server ... on a users work station?

Re: Help – Timing Logic

Server (2003)

- who will be responsible for executing / sending messages
... one computer or many different computers?

One computer. One multithreaded vb.net app.

- will this message sender reside on a server / database
server / or a user's workstation ... ???

Remote SQL Server 2005 (EE)

- what database are you using ???

SQL Server 2005 EE

- is this functionality wrapped up in a large application?

Not very large. The logic around this loop cycle is pretty much the entire application.

- why 4 threads ?

Allows sending 4 messages at a time.

- how does you program determine which message to send
... parameter list please.

Read from a table in the db server. A Queue table. (employeeid,
messageaddress)

If this information is so mission critical, and employees must
get their message ASAP ... put a trigger on the database

Re: Help – Timing Logic

table.

Cant have a trigger execute a function in a remote vb.net app.

... wrap the necessary functionality in a small stand alone exe application

Exactly what I need to do.

...
... build your application so it receives commandline parameters / string

Considering it.

...
... build a trigger on the database .. fire on inserts ... have it call your message program with a command parameter (messageID) ...
... install your application on the database server this will speed up the connection / retrievals and so on ... no network latency.

Each time a message is inserted in the table, the trigger fires, calling your EXE with the appropriate parameter string, exe starts, fires off message ... done. Small exe can run as many times on the server...

If you are using a timestamp for determining which messages to send ... incorporate another table ... LastTimeExecuted ...

table: MessengerExecution
field: LastDateTimeFired

begin transactions ... lock table.
select lastdatetimefired from this table...in a variable
update field with thread date/time ...
end transaction.
return the select value and the update value

select messages where date is between lastdatetimefired and

Re: Help – Timing Logic

the updateddatetime I just used...

Again, there are many solutions ... just do not completely understand what you are doing...

Bottom line ...

– you have determined that you need to run a mutli-threaded process for this.

– so, in order to avoid DUPLICATE MESSAGES you either have to employee...

Transaction and Database Locking – look at isolation levels / settings

or

A booker type of system...

have the broker continually pool the table ... this will only work if 1 machine is designated for messaging! If more than one machine will be used for messaging ... you will have to roll up your sleeves and look at Transactions and Isolation Levels.

CheckMessage

```
Do Until Company.Revenues < 0
  MessegeID = broker.getNextMessage()
  broker.sendMessage(MessageID)
Loop
```

SendMessage(MessageID)

Create a new thread...

SendMessage(MessageID) in this new thread...

Return to the MessageBroker.CheckMessage...while the other thread is preparing and sending the message...

This will continuous poll the database server for new messages! ...

However, if it is mission critical users receive this information ASAP ... TRIGGER on database table! If the

Re: Help – Timing Logic

user can wait 10 seconds .. build a BROKER ... and have it spawn as many threads needed to send the messages in the QUEUE ... have it control the process...and sending!

Again, many solutions, depends on your needs ... code sample, table structure ... some thing to trying and figure out how exactly your 'threads' are getting 1 message...

Jeff.

"Jay" <someone@xxxxxxxxxxxxxx> wrote in message news:OLc3ODxEHHA.996@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

It is extremely important for the employees to receive the messages almost immediately after identifying (time val in table). So, threading ensures I can execute multiple processes at the same time... only problem is these threads read from the same queue table and have the potential to send the same message to the same guy x number of times. where x is the number of threads running.

"jeff" <jhersey at allnorth dottt com> wrote in message news:uQ7%23Y4wEHHA.4404@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Still do not know why you need 4 threads ? Speed ? Does runnig the process in one thread not work fast enough for you since timing is import ? Why 4 threads why not 6,12 , 3, 36 ... ? How do you determine which row a 'thread retrieves' ??? does it simply select the next available row from the DATABASE ??? How does the thread determine 'the next row'???

If mutli-threading is duplicating 'send messages' it is not working properly... If you are multi-threading to improve performance ...

Re: Help – Timing Logic

maybe look at the design...
If you need to implement a
locking mechanism / or /
logging mechanism / or / a
checking mechanism to
avoid duplicate messages
caused by multi-threading
... these will all come at a
cost ... performance cost!

What is bottle necking your
process that you need 4
threads? Is it the READ
from the database ... is it the
SENDING the text
message? Is it connecting to
the database ??? do not
know here ...

Look at what is causing
problem ...

– you need better
performance ... so, multiple
thread it! However,
mutli-threading the entire
process is causing issues ...
duplicate records ... now
you either need to
incorporate a locking
procedure ... or a checking
procedure to avoid duplicate
messages... all this has a
cost to the overall
performance ... adding more
threads may in fact
negatively impact the
overall performance....

Maybe implement a
message broker ...

– message broker gets all
the necessary messages or
message id's from the
database to be sent ...
in-memory list..
– message broker loops
through list of messageIDs
...
– message broker starts

Re: Help – Timing Logic

another thread for
processing the
sendtextmessage
functionality for each
messageID...
– message broker will
include the messageID so
the process knows which
message to get and
process...
– message broker will
ensure each database row is
only processed once...

It is very hard to help you
without knowing exactly
what you are doing...

Again, I will ask, what
rational / reasoning did you
use for using 4 threads ?
Performance ? Speed ? ...
Where is the bottle neck in
the process that requires you
to multi-thread it? Maybe
just move the 'bottle neck
section' to another thread ...
?

Grasping at straws ...

Code would be nice how to
see what you are doing ...

Jeff

"Jay"

Re: Help – Timing Logic

<someone@xxxxxxxxxxxxxx>

wrote in message

news:uBX7QewEHHA.1224@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

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

Re: Help – Timing Logic

row can be
returned at a
time for
each thread.

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

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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

to
watch
out
here
for
table
locking
...
may
impact
performance,
implement
using
an
update
flag
at
sent
and
'check
before
send
method'...

...
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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

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
...

Re: Help – Timing Logic

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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

internal
locking
mechanisms
of
the
database
to
achieve
this
...
but
proceed
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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

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

Re: Help – Timing Logic

retrieving
a
list
of
messages
and
how
you
are
processing
each
record...

Jeff.

"Jay"

<someone@xxxxxxxxxxxxxx>

wrote

in

message

news:O2phzwvEHHA.4280@xxxxxxxxxxxxxx

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,

Re: Help – Timing Logic

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 records 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

Re: Help – Timing Logic

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

