

RE: Access Append Query to update Sequence Generator in Oracle

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.queries/2007-12/msg01180.html>

- *From:* Jerry Whittle <JerryWhittle@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 21 Dec 2007 16:45:00 -0800
-

That is one very, very complex set of triggers. The logic is very complex and does a lot more than just update records with a sequence. I very much doubt that you can insert a lot of records into those tables through an append query.

You really need to talk to whoever developed the triggers and find the logic for them before inserting records this way.

—

Jerry Whittle, Microsoft Access MVP
Light. Strong. Cheap. Pick two. Keith Bontrager – Bicycle Builder.

"Jim" wrote:

Two tables are being appended to one after the other. TASSIGNMENT and TASSIGNINTERVAL.

There are two Triggers apparently for TASSIGNMENT and there doesn't appear to be a Trigger for TASSIGNINTERVAL

I looked for some code related to SEQUENCING and it's just a table asking Start Value, Min Value, Max Value, Increment, Cache and Last Number (the Last Number is what's NOT being updated after the Insert Query). There are also two check boxes: Recycle after min and max AND Guaranteed order. Neither are checked.

I have found this all using SQL Navigator.

=====

Trigger code #1 for the TASSIGNMENT table

```
DECLARE
assignmentVariable0 INTEGER := 0;
DBSTARTDATE DATE;
assignmentVariable8 INTEGER := 0;
assignmentVariable9 INTEGER := 0;
```

RE: Access Append Query to update Sequence Generator in Oracle

```
assignmentVariable11 INTEGER := 0;

BEGIN

/* Prevent insert of task linked to charges*/

BEGIN
SELECT COUNT(*) INTO assignmentVariable0
FROM TTASK
WHERE :NEW.ORGANIZATIONID = TTASK.ORGANIZATIONID
AND :NEW.TASKID = TTASK.UNIQUEID
AND TTASK.CHARGEID > 0;
EXCEPTION
WHEN NO_DATA_FOUND THEN
assignmentVariable0 := 0;

WHEN TOO_MANY_ROWS THEN
NULL;
END;
IF( assignmentVariable0) > 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20303#' ); /* ROLLBACK; */
/* Manual Intervention required.*/
END IF;

IF :NEW.ENDDATE <= :NEW.STARTDATE THEN
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20023#' ); /* ROLLBACK; */
/* Manual Intervention required.*/
END IF;
SELECT DATESTART INTO DBSTARTDATE
FROM TGLOBALPERIOD
WHERE TGLOBALPERIOD.ORGANIZATIONID = :NEW.ORGANIZATIONID;
BEGIN
SELECT COUNT(*) INTO assignmentVariable8
FROM DUAL
WHERE MOD(ROUND(:NEW.STARTDATE - DBSTARTDATE), 7) < 0;
EXCEPTION
WHEN NO_DATA_FOUND THEN
assignmentVariable8 := 0;

WHEN TOO_MANY_ROWS THEN
NULL;
END;
IF( assignmentVariable8) > 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20021#' ); /* ROLLBACK; */
/* Manual Intervention required.*/
END IF;
BEGIN
SELECT COUNT(*) INTO assignmentVariable9
FROM DUAL
WHERE ( MOD(ROUND(:NEW.ENDDATE - DBSTARTDATE), 7) < 0)
AND (:NEW.ENDDATE != TO_DATE('11/27/2737', 'MM/DD/YYYY'));
```

RE: Access Append Query to update Sequence Generator in Oracle

```
EXCEPTION
WHEN NO_DATA_FOUND THEN
assignmentVariable9 := 0;

WHEN TOO_MANY_ROWS THEN
NULL;
END;
IF( assignmentVariable9) > 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20021#'); /* ROLLBACK; */
/* Manual Intervention required.*/
END IF;
BEGIN
TASSIGNMENTDATA.V_NUMENTRIES := TASSIGNMENTDATA.V_NUMENTRIES +
1;
TASSIGNMENTDATA.V_USERIDs(TASSIGNMENTDATA.V_NUMENTRIES) :=
:NEW.USERID;
TASSIGNMENTDATA.V_TASKIDs(TASSIGNMENTDATA.V_NUMENTRIES) :=
:NEW.TASKID;
TASSIGNMENTDATA.V_WORKFLOWENTRYIDs(TASSIGNMENTDATA.V_NUMENTRIES)
:=
:NEW.WORKFLOWENTRYID;
TASSIGNMENTDATA.V_ORGANIZATIONIDs(TASSIGNMENTDATA.V_NUMENTRIES)
:=
:NEW.ORGANIZATIONID;
TASSIGNMENTDATA.V_STARTDATEs(TASSIGNMENTDATA.V_NUMENTRIES) :=
:NEW.STARTDATE;
TASSIGNMENTDATA.V_ENDDATEs(TASSIGNMENTDATA.V_NUMENTRIES) :=
:NEW.ENDDATE;
TASSIGNMENTDATA.V_UNIQUEIDs(TASSIGNMENTDATA.V_NUMENTRIES) :=
:NEW.UNIQUEID;

END;

BEGIN
SELECT COUNT(*) INTO assignmentVariable11
FROM TTASK
WHERE TTASK.UNIQUEID = :NEW.TASKID
AND :NEW.ORGANIZATIONID = TTASK.ORGANIZATIONID
AND (TTASK.STARTDATE > :NEW.STARTDATE
OR TTASK.ENDDATE + 1 < :NEW.ENDDATE);
EXCEPTION
WHEN NO_DATA_FOUND THEN
assignmentVariable11 := 0;

WHEN TOO_MANY_ROWS THEN
NULL;
END;
IF( assignmentVariable11) > 0 THEN
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20233#'); /* ROLLBACK; */
/* Manual Intervention required.*/
END IF;
```

RE: Access Append Query to update Sequence Generator in Oracle

END TASSIGNMENT_ITRIG;

=====
Trigger code #2 for the TASSIGNMENT table

```
DECLARE
-- local variables here
assignmentVariable10 INTEGER := 0;
V_USERID TASSIGNMENT.USERID%TYPE;
V_TASKID TASSIGNMENT.TASKID%TYPE;
V_WORKFLOWENTRYID TASSIGNMENT.WORKFLOWENTRYID%TYPE;
V_ORGANIZATIONID TASSIGNMENT.ORGANIZATIONID%TYPE;
V_STARTDATE TASSIGNMENT.STARTDATE%TYPE;
V_ENDDATE TASSIGNMENT.ENDDATE%TYPE;
V_UNIQUEID TASSIGNMENT.UNIQUEID%TYPE;

BEGIN

FOR V_LOOPINDEX IN 1..TASSIGNMENTDATA.V_NUMENTRIES LOOP
V_USERID := TASSIGNMENTDATA.V_USERIDS(V_LOOPINDEX);
V_TASKID := TASSIGNMENTDATA.V_TASKIDS(V_LOOPINDEX);
V_WORKFLOWENTRYID :=
TASSIGNMENTDATA.V_WORKFLOWENTRYIDS(V_LOOPINDEX);
V_ORGANIZATIONID := TASSIGNMENTDATA.
V_ORGANIZATIONIDS(V_LOOPINDEX);
V_STARTDATE := TASSIGNMENTDATA.V_STARTDATES(V_LOOPINDEX);
V_ENDDATE := TASSIGNMENTDATA.V_ENDDATES(V_LOOPINDEX);
V_UNIQUEID := TASSIGNMENTDATA.V_UNIQUEIDS(V_LOOPINDEX);
BEGIN
SELECT COUNT(*) INTO assignmentVariable10
FROM TASSIGNMENT
WHERE TASSIGNMENT.USERID = V_USERID
AND TASSIGNMENT.TASKID = V_TASKID
AND TASSIGNMENT.WORKFLOWENTRYID = V_WORKFLOWENTRYID
AND TASSIGNMENT.ORGANIZATIONID = V_ORGANIZATIONID
AND ((TASSIGNMENT.STARTDATE >= V_STARTDATE
AND TASSIGNMENT.STARTDATE < V_ENDDATE)
OR (TASSIGNMENT.ENDDATE > V_STARTDATE
AND TASSIGNMENT.ENDDATE <= V_ENDDATE)
OR (TASSIGNMENT.STARTDATE <= V_STARTDATE
AND TASSIGNMENT.ENDDATE >= V_ENDDATE))
AND TASSIGNMENT.UNIQUEID != V_UNIQUEID;
EXCEPTION
WHEN NO_DATA_FOUND THEN
assignmentVariable10 := 0;

WHEN TOO_MANY_ROWS THEN
NULL;
END;
IF( assignmentVariable10) > 0 THEN
```

RE: Access Append Query to update Sequence Generator in Oracle

```
RAISE_APPLICATION_ERROR(-20001, 'EX-ERR:20058#'); /* ROLLBACK; */  
/* Manual Intervention required.*/  
END IF;  
END LOOP;  
-- Reinitialize the counter in order the next execution will use new data  
TASSIGNMENTDATA.V_NUMENTRIES :=0;  
  
END TASSIGNMENT2_ITRIG;
```

"Jerry Whittle" wrote:

Oracle does not expect a commit after every record change. You could insert a thousand records into a table and roll them all back out by not committing. Also committing should not make any difference to the sequence or trigger.

If possible, post the script for both the trigger and sequence.

--

Jerry Whittle, Microsoft Access MVP
Light. Strong. Cheap. Pick two. Keith Bontrager – Bicycle Builder.

"Jim" wrote:

That's what I thought too, but this is the whole reason for the post ... it did sequence just fine ... the new numbers are IN the table. What happened was that the Sequence Generator in Oracle remained at the last number PRIOR to the insert query, so, even though the new larger numbers went in, Oracle still thinks the next number is the next number BEFORE the insert. See my dilemma? FYI, Access could read Oracles next number to start the sequence. It then sequenced just fine as inserted. The only issue is after the insert was complete, Oracle didnt' recognize the NEW last number.

I read something that said Access doesn't "commit" till the end of the insert query. Could this be it? Is Oracle EXPECTING to insert, then commit, record by record?? If so, how can I include this in my Access query?

"Jerry Whittle" wrote:

RE: Access Append Query to update Sequence Generator in Oracle

The trigger "should" automatically grab the next sequence number as you insert the records. It's probably fired during the BEFORE INSERT event on the table. I don't think that ODBC will mess it up; however, it's always worth testing first.

--

Jerry Whittle, Microsoft Access MVP
Light. Strong. Cheap. Pick two. Keith
Bontrager – Bicycle Builder.

"Jim" wrote:

Thanks Jerry. I wrote the query in Design View of Access, rather than VBA directly. It is a multiuser app, so of course that point is well taken. I was under the assumption that Access would be reading the available largest Unique ID on the fly (as records are appending). Sounds like that's not the case. There is a trigger in Oracle as you guessed, but I don't know how to "trigger" this from the action query. I have a datasource representing the records to insert and I append this to the actual dB.

The code you offer below, does indeed look like the Sequence information in Oracle. How would include this in my query? OR are you even saying this. I'm a novice ... your help is greatly appreciated. Jim

"Jerry Whittle" wrote:

For the
most part an
Oracle

RE: Access Append Query to update Sequence Generator in Oracle

sequence is
run by a
trigger on
the table. So
you
should
normally let
Oracle get
the next
unique ID
as the
records are
appended.

If for some
reason you
do need to
do this,
there are
problems.

If this is a
multi-user
app, you
don't know
if someone
else is
inserting a
record at the
same time.
That could
cause a
problem or
two.

As far as I
know, to
update a
sequence to
a different
number
manually,
you
need to drop
the
sequence
then
recreate it
with
something
like below.

RE: Access Append Query to update Sequence Generator in Oracle

"Jim"
wrote:

I
wrote
an
append
query
that
successfully
gets
new
Unique
ID's,
loads
them
as
the
append
query
runs,
and
appends
to
the
Oracle
tables
just
as
expected
(ODBC
connection).
However,
the
Sequence
Generator
in
Oracle
IS
NOT
updated
to
recognize
the
newly
inserted
records.

What
do

RE: Access Append Query to update Sequence Generator in Oracle

I
do
to
make
this
happen?
The
current
result
is
any
new
additions
after
the
action
query
"think"
the
sequence
begins
at
the
number
just
prior
to
the
inserted
records
(as
if
they
weren't
successfully
inserted)
...
but
they
were.
Help!

=====