Database - Bug #12094

NO-WAIT support for GET CURRENT and the GET methods (lock conflicts)

10/28/2026 12:36 PM - Stefanel Pezamosca

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

Related issues:

Related to Database - Bug #10026: A multi-table AdaptiveQuery running off-end...	**Test**

History

#1 - 10/28/2026 12:46 PM - Stefanel Pezamosca

Follow-up of #10026; the analysis is in #10026-24 and #10026-31. The FWD results below were tested on branch 10026a r16812, which is now in trunk (rev. 16818).

Problem

1. GET CURRENT q NO-WAIT without a lock keyword waits for the lock

Conversion keeps NO-WAIT only when it comes with a lock keyword (rules/convert/database_access.rules), so GET CURRENT q NO-WAIT is emitted as a plain current(). At runtime the query reloads the result-list entry with each component's own lock type, and waits when another session holds a conflicting lock.

Scenario: scrolling join, the buffer holds a different record than the current entry (stale), another session holds the entry's outer record with EXCLUSIVE-LOCK, then GET CURRENT q NO-WAIT.

Result	OpenEdge	FWD
Warning 4108	yes	yes
Waits for the lock	no, reads the entry record	yes, until the other session releases it
LOCKED	yes	no

FWD also waits when the buffer is not stale.

2. GET navigation with EXCLUSIVE-LOCK NO-WAIT fails on a lock conflict

Scenario: GET FIRST/LAST/NEXT/PREV with EXCLUSIVE-LOCK NO-WAIT, as a statement or as a method, while another session locks the record.

Form	OpenEdge	FWD
Method	returns yes, record read NO-LOCK, LOCKED = yes	returns no, "Could not acquire exclusive lock", LOCKED = no
Statement	record read NO-LOCK, LOCKED = yes, no error	ERROR "Could not acquire exclusive lock", block undone

3. GET methods with EXCLUSIVE-LOCK do not require a transaction

OpenEdge never takes EXCLUSIVE-LOCK outside a transaction. A static GET statement makes its block the transaction; the GET methods fail with error 7359 ("QUERY GET methods with exclusive lock requires a transaction"). FWD has no 7359.

Scenario: the session holds a SHARE lock on the record, no transaction is active, then q:GET-CURRENT(EXCLUSIVE-LOCK, NO-WAIT).

Case	OpenEdge	FWD
Another session holds SHARE	7359, the session keeps its SHARE lock, LOCKED = no	the session loses its SHARE lock, LOCKED = yes
Record free	7359, the session keeps its SHARE lock	takes EXCLUSIVE-LOCK

4. NO-WAIT in a query record phrase (optional)

OpenEdge rejects NO-WAIT in any query record phrase (FOR EACH, DO PRESELECT, OPEN QUERY, OPEN QUERY PRESELECT, static and dynamic QUERY-PREPARE) with error 247. FWD reports 247 for QUERY-PREPARE, but conversion accepts NO-WAIT in the static forms.

Proposed fix

- **GET CURRENT q NO-WAIT:** keep the NO-WAIT in conversion when no lock keyword is given, and pass it to the query. At runtime, take each component's own lock type (as PreselectQuery.reloadEntry does; CompoundQuery.reloadResultListEntry does not) and add the NO-WAIT to it. A component's own lock never carries NO-WAIT (error 247), so PreselectQuery needs the same change. On a lock conflict, read the record NO-LOCK and set LOCKED. This touches the query API and four query classes.
- **GET navigation with NO-WAIT:** on a lock conflict, read the record NO-LOCK and set LOCKED instead of raising an error, for both the statement and the method.
- **GET methods with EXCLUSIVE-LOCK:** raise 7359 when no transaction is active. The session then keeps its SHARE lock, as in OpenEdge.
- **Optional:** report 247 at conversion time for NO-WAIT in a static query record phrase.

Already fixed in #10026

- q:GET-CURRENT(EXCLUSIVE-LOCK, NO-WAIT) keeps NO-WAIT in conversion and matches OpenEdge (GL08, checked with probes in #10026-31).
- The FIND-* methods keep NO-WAIT in conversion. Static and dynamic FIND CURRENT ... NO-WAIT raise the lock conflict (445), as in OpenEdge.

Not checked

- A literal wait-mode argument of 0 (e.g. GET-CURRENT(EXCLUSIVE-LOCK, 0)). Conversion removes it, so the call waits for the lock. What OpenEdge does with this value has not been tested.

#2 - 10/28/2026 12:47 PM - Stefanel Pezamosca

- Related to Bug #10026: A multi-table AdaptiveQuery running off-end is still updating the cursor added