Database - Feature #12025

REPOSITION-FORWARD and REPOSITION-BACKWARD error handling

10/19/2026 07:46 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Alexandru Lungu	**% 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

History

#1 - 10/19/2026 07:46 PM - Greg Shah

REPOSITION-FORWARD and REPOSITION-BACKWARD error handling

Project: Database

Problem

The query methods REPOSITION-FORWARD() and REPOSITION-BACKWARD() are marked Full for conversion and Partial at runtime, with the reason *"error reporting is not complete"*.

The cause is in QueryWrapper.queryForward(NumberType) and queryBackward(NumberType) (persist/QueryWrapper.java, around lines 4837-4894):

- An unknown row count returns false under // TODO: what error should be shown?. No error or warning is raised.
- The calls to forward() and backward() are wrapped in catch (ErrorConditionException exc) { return logical.of(false); }. So the conditions raised by canReposition() are swallowed: 3163 (static query not opened), 7315 (dynamic query not opened) and 3164 (query not defined as SCROLLING). Without NO-ERROR no message appears, and under ROUTINE-LEVEL or BLOCK-LEVEL ON ERROR UNDO, THROW nothing is thrown.

#10277 removed exactly this pattern from queryReposition and queryRepositionByID (REPOSITION-TO-ROW and REPOSITION-TO-ROWID), but did not change the forward and backward methods. The equivalent methods on the null-delegate implementation (around lines 9696-9750) also just return false.

Work

1. Apply the #10277 approach to queryForward and queryBackward: call canReposition() and let its errors propagate, instead of catching ErrorConditionException.
2. Establish on OpenEdge what an unknown row count, a negative row count and zero do, and implement it.
3. Check the null-delegate versions of both methods.
4. Establish on OpenEdge whether a static query that is **not** declared SCROLLING but is opened with QUERY-PREPARE / QUERY-OPEN through its handle counts as forward-only for these methods, and make FWD match.
5. Check the REPOSITION query-name FORWARDS/BACKWARDS statement forms as well as the methods.
6. Update the gap marking for both methods.

Testcases

Each with an OpenEdge baseline, with and without NO-ERROR, and inside a block with UNDO, THROW:

- repositioning a static and a dynamic query that is not open;
- repositioning a query that is not SCROLLING;
- an unknown, negative and zero row count;
- moving past the beginning and past the end of the result list, and the result of a following GET-NEXT / GET-PREV;
- the return value of the method in each case.

#2 - 10/20/2026 04:10 AM - Alexandru Lungu

- Assignee set to Alexandru Lungu