Database - Bug #12176

BREAK BY walk: row created ahead of the cursor is hidden after UNDO, RETRY on a two-field index

11/16/2026 06:21 AM - Artur Școlnic

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 #10806: Presort queries can sometimes dynamically f...	**Review**

History

#1 - 11/16/2026 06:22 AM - Artur Școlnic

Description

A FOR EACH ... BREAK BY iteration creates a row in the same table that sorts ahead of the cursor, then does UNDO, RETRY. The retry runs as NEXT (infinite loop protection). Outside a transaction the row stays in the table. OE 12.8 then hides that row from the rest of the walk, but only for some indexes:

- index (nm): delivered;
- index (nm, seq): hidden, for rows created up to the second row ahead of the current one (on K01: K01A and K02A hidden, K03A delivered);
- without the RETRY: delivered on both indexes.

10806b delivers the row on both indexes. Trunk hides it, but only because trunk never delivers any row created after the query opens (24 of 38 cases in the same battery differ from OE on trunk, #10806). The fix attempted in 10806b (r16843: record the keys created by the iteration and skip them) was reverted: it applied the rule to every index, leaked into nested walks, and added key registries to BufferManager / RecordBuffer (#10806-106).

The goal is to find out what OE does here. It is most likely a side effect of the UNDO, the RETRY-as-NEXT or the index cursor, not deliberate tracking. Then reproduce it in AdaptivePresortQuery without a key registry.

Test case

```
DEFINE TEMP-TABLE ta FIELD nm AS CHARACTER FIELD seq AS INTEGER INDEX ixa nm.
DEFINE TEMP-TABLE tb FIELD nm AS CHARACTER FIELD seq AS INTEGER INDEX ixb nm seq.
DEFINE BUFFER bax FOR ta.
DEFINE BUFFER bbx FOR tb.
DEFINE VARIABLE i     AS INTEGER   NO-UNDO.
DEFINE VARIABLE trace AS CHARACTER NO-UNDO.
DEFINE VARIABLE lDone AS LOGICAL   NO-UNDO.

DO i = 1 TO 10:
   CREATE ta. ta.nm = "K" + STRING(i, "99").
   CREATE tb. tb.nm = "K" + STRING(i, "99").
END.
RELEASE ta. RELEASE tb.

/* index (nm) */
FOR EACH ta BREAK BY ta.nm:
   trace = trace + ta.nm + ",".
   IF ta.nm = "K01" AND NOT lDone THEN DO:
      lDone = TRUE.
      CREATE bax. bax.nm = "K02A".
      UNDO, RETRY.
```

```
   END.
END.
MESSAGE "(nm)       " trace.

/* index (nm, seq) */
ASSIGN trace = "" lDone = FALSE.
FOR EACH tb BREAK BY tb.nm:
   trace = trace + tb.nm + ",".
   IF tb.nm = "K01" AND NOT lDone THEN DO:
      lDone = TRUE.
      CREATE bbx. bbx.nm = "K02A".
      UNDO, RETRY.
   END.
END.
MESSAGE "(nm, seq) " trace.
```

Index	OE 12.8	trunk	10806b
(nm)	K01,K02,K02A,K03,...,K10	K02A missing	K01,K02,K02A,K03,...,K10
(nm, seq)	K02A missing	K02A missing	**K01,K02,K02A,K03,...,K10**

In both cases the table has 11 rows afterwards.

To do

1. Find the OE rule: three-field, unique and descending indexes, a BREAK BY field that is not the index's lead, a database table, and rows created further ahead.
2. Explain why the second index component changes the result, e.g. by tracing where OE's index cursor is after the retry.
3. Reproduce it in AdaptivePresortQuery, scoped to the query: no key collections in BufferManager / RecordBuffer, no effect on nested walks.

#2 - 11/16/2026 06:22 AM - Artur Școlnic

- Related to Bug #10806: Presort queries can sometimes dynamically find records created after opening the query added