Database - Bug #12027

Embedded SQL SELECT INTO: aggregate column alias defect and BETWEEN support

10/19/2026 07:50 PM - Greg Shah

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

History

#1 - 10/19/2026 07:50 PM - Greg Shah

Embedded SQL SELECT INTO: aggregate column alias defect and BETWEEN support

Project: Database

Problem

FWD converts the simpler forms of the embedded SQL SELECT ... INTO statement: a single aggregate (COUNT(*), MAX, MIN, SUM, AVG) over one table, with a WHERE clause built from comparisons combined with AND / OR. The detection is in rules/annotations/database_general.rules (around lines 590-610) and the runtime goes through RecordBuffer.aggregate(...).

Two problems were found while testing this support (see #11892-10):

1. **A runtime defect in the simplest form.** The FQL to SQL conversion emits a stray alias for the aggregate column. For example, select count(*) from Item__Impl__ becomes select count(*) as from item, which the database rejects. So even the form the conversion supports does not work today. It is not yet known whether MAX and the other aggregates are affected the same way as COUNT.
2. **BETWEEN is not converted** in the WHERE clause of an embedded SQL statement. expressions.rules marks it sql.put(kw_between, cvt_lvl_none). A typical use is:

```
SELECT MAX(modtim) INTO iTime FROM payment
   WHERE payment.moddat BETWEEN dFrom AND dTo.
```

Work

1. Fix the aggregate alias in the FQL to SQL conversion (Fql2SqlConverter or wherever the select list for an aggregate is rendered), so that the generated SQL is valid on every supported dialect: PostgreSQL, SQL Server, MariaDB and H2.
2. Check every aggregate function, not only COUNT(*): COUNT(field), MAX, MIN, SUM and AVG.
3. Add BETWEEN (and NOT BETWEEN) to the embedded SQL WHERE clause conversion, with the 4GL semantics for the unknown value at either bound.
4. Check that a qualified column reference written with whitespace before the dot (table .field), which the 4GL accepts, converts correctly.
5. Update the gap marking for count() and SELECT once the work is done. The other unsupported constructs (DISTINCT, GROUP BY, HAVING, joins, IN, LIKE, arithmetic on an aggregate, several aggregates in one statement) stay out of scope for this task unless they turn out to be trivial.

Testcases

Each testcase needs an OpenEdge baseline.

- SELECT COUNT(*) INTO over an empty table, a table with rows, and with a WHERE clause that matches none, one and many rows.

- The same for MAX, MIN, SUM and AVG on integer, decimal, date and character fields, including the result when no row matches.
- WHERE clauses using =, <>, <, >, AND, OR and parentheses, comparing with variables, literals and ''.
- BETWEEN and NOT BETWEEN with dates, integers and character values, and with an unknown value at each bound.
- All of the above on PostgreSQL, SQL Server and H2 at least.