Mulgara - Bug #183

project() isn't considered with index selection.

02/17/2009 08:30 AM - Andrae Muys -

Status:	In Progress	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Andrae Muys -	**% Done:**	0%
Category:	Mulgara	**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

select $p from <graph:data> where $s $p $o; uses the GSPO index which requires a full external sort of the store to extract $p. If we use GPOS we get $p without the sort.

The problem is that project() call doesn't participate in index ordering within the query optimisation.

The solution is to introduce an Annotation that allows the project to specify a preferred order. Initially supporting this Annotation with StatementStoreResolution will be sufficient to satisfy the query above. A general solution will also require supporting it with Join and Append tuples.

Associated revisions

Revision 1500 - 02/17/2009 09:32 AM - Andrae Muys -

refs #183

Add a PartialOrderAnnotation to StatementStoreResolution and exploit in TuplesOperations::project() to allow optimisation of index selection with respect to projection.

Revision 1510 - 02/17/2009 11:45 AM - Andrae Muys -

refs #183

Migrate the ReresolvableResolution interface to the ReresolveAnnotation.

This clears up the dependency on class introspection for optimisation, allowing the wrapping of reresolvable tuples without losing their ability to participate in optimisation.

Revision 1542 - 02/25/2009 01:31 AM - Andrae Muys -

refs #183

This double array allows us to identify which index will satisfy a desired ordering of the unbound variables, or if we don't have the appropriate index available. Should we find an appropriate index we can avoid sorting the resulting tuples. If the table returns -1, the sort is unavoidable, so we should default to the default index selection to improve cache efficiency.

Revision 1558 - 02/26/2009 12:24 PM - Andrae Muys -

refs #183

Refactored XAStatementStoreImpl to seperate the selection of the index from the selection of the find method on the index. This allows us to select the best valid index for a set of bound variables matching an ordering preference for the unbound variables. This is necessary if higher level code is to avoid a sort.

Revision 1564 - 02/28/2009 02:17 AM - Andrae Muys -

refs #183

Passes all regression tests, untested is the actual preference based index selection, one difficulty will be matching up the nodes with the index columns based on index order.

Next step is to eliminate the switch statement by lifting it to a lookup table, and writing some unit-tests for the new functionality.

Revision 1569 - 03/02/2009 06:48 AM - Andrae Muys -

refs #183

Added a phaseId to TripleAVLFile.Phase, to allow us to track which index a phase is being used for.

Added bind[1-4] methods to ensure findTuples parameters are matched up with the correct columns for the relevant index.

Revision 1570 - 03/02/2009 08:11 AM - Andrae Muys -

refs #183

Missed updating unit-test to reflect new constructor signatures.
Updated XA11 to reflect new constructor signatures.

Revision 1593 - 03/05/2009 08:44 AM - Andrae Muys -

refs #183

Code cleanups.

History

#1 - 02/17/2009 08:57 AM - Andrae Muys -

- Status changed from New to In Progress

#2 - 02/17/2009 09:32 AM - Andrae Muys -

(In r1500) refs #183

Add a PartialOrderAnnotation to StatementStoreResolution and exploit in TuplesOperations::project() to allow optimisation of index selection with respect to projection.

#3 - 02/17/2009 11:45 AM - Andrae Muys -

(In r1510) refs #183

Migrate the ReresolvableResolution interface to the ReresolveAnnotation.

This clears up the dependency on class introspection for optimisation, allowing the wrapping of reresolvable tuples without losing their ability to participate in optimisation.

#4 - 02/25/2009 01:31 AM - Andrae Muys -

(In r1542) refs #183

This double array allows us to identify which index will satisfy a desired ordering of the unbound variables, or if we don't have the appropriate index available. Should we find an appropriate index we can avoid sorting the resulting tuples. If the table returns -1, the sort is unavoidable, so we should default to the default index selection to improve cache efficiency.

#5 - 02/26/2009 12:24 PM - Andrae Muys -

(In r1558) refs #183

Refactored XAStatementStoreImpl to seperate the selection of the index from the selection of the find method on the index. This allows us to select the best valid index for a set of bound variables matching an ordering preference for the unbound variables. This is necessary if higher level code is to avoid a sort.

#6 - 02/28/2009 02:17 AM - Andrae Muys -

(In r1564) refs #183

Passes all regression tests, untested is the actual preference based index selection, one difficulty will be matching up the nodes with the index columns based on index order.

Next step is to eliminate the switch statement by lifting it to a lookup table, and writing some unit-tests for the new functionality.

#7 - 03/02/2009 06:48 AM - Andrae Muys -

(In r1569) refs #183

Added a phaseId to TripleAVLFile.Phase, to allow us to track which index a phase is being used for.

Added bind[1-4] methods to ensure findTuples parameters are matched up with the correct columns for the relevant index.

#8 - 03/02/2009 08:11 AM - Andrae Muys -

(In r1570) refs #183

Missed updating unit-test to reflect new constructor signatures.
Updated XA11 to reflect new constructor signatures.

#9 - 03/05/2009 08:44 AM - Andrae Muys -

(In r1593) refs #183

Code cleanups.