Database - Bug #8488

improve performance of WRITE-JSON

03/20/2024 12:25 PM - Constantin Asofiei

Status:	Test	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

Related issues:

Related to Database - Feature #8494: rework XmlExport with the performance im...	**New**

History

#2 - 03/20/2024 03:52 PM - Constantin Asofiei

- Status changed from New to WIP

- Assignee set to Constantin Asofiei

- % Done changed from 0 to 90

Created branch 8488a from trunk rev 15077. In rev 15088 there are performance improvements for WRITE-JSON:

- Avoid BDTs within internal FWD runtime.
- Replaced iterators with Java 'for', where it applies.
- Read the temp-table fields directly from the Record.data as Java types and not BDT.
- Directly set the record in the parent buffer instead of using a findUnique

Eric, you may want to look at the changes in persist.orm.

Ovidiu, an early review is appreciated - I still need to regression test in standalone tests (especially (de)normalized temp-table extent fields in WRITE-JSON).

#3 - 03/20/2024 06:51 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, an early review is appreciated - I still need to regression test in standalone tests (especially (de)normalized temp-table extent fields in WRITE-JSON).

Sure. I think the code is good. My observations are as follows:

- In P2JQuery.java, you added public default boolean _getNext() which is implemented as: getNext().booleanValue(). I think we should do the other way around: the getNext() method to be 'final' (evidently cannot be) and return new logical(_getNext()). The implementing classes should use _getNext() return Java native value. You did this whole implementation in QueryWrapper, I see.
- In RecordBuffer.java. We attempted to make the public API as small as possible. Does loadRecord() really need to be public?
- Related to above, Property.java, the new back-reference to PropertyMeta meta is public. I know that this is probably not the final form, maybe we can find a solution which better encapsulates and grants the new member a const/final feeling?
- JsonExport.java, line 1466/1467, I think they can be written as date p2jDate = new datetimetz((OffsetDateTime) d);, avoiding changing the value

of d argument of the lambda. IMO, it is ugly to change the d's type (and its value, of course) that way. The same for datetime and date, below;
- the XmlExport might also benefit from the same boost as JsonExport.

#4 - 03/21/2024 07:42 AM - Constantin Asofiei

Ovidiu, in rev 15080 I have the review fixes. See bellow for notes:

- In P2JQuery.java, you added public default boolean _getNext() which is implemented as: getNext().booleanValue(). I think we should do the other way around: the getNext() method to be 'final' (evidently cannot be) and return new logical(_getNext()). The implementing classes should use _getNext() return Java native value. You did this whole implementation in QueryWrapper, I see.

Done

- In RecordBuffer.java. We attempted to make the public API as small as possible. Does loadRecord() really need to be public?

Yes, it's used from JsonExport, which is in a different package.

- Related to above, Property.java, the new back-reference to PropertyMeta meta is public. I know that this is probably not the final form, maybe we can find a solution which better encapsulates and grants the new member a const/final feeling?

Fixed.

- JsonExport.java, line 1466/1467, I think they can be written as date p2jDate = new datetimetz((OffsetDateTime) d);, avoiding changing the value of d argument of the lambda. IMO, it is ugly to change the d's type (and its value, of course) that way. The same for datetime and date, below;

Fixed.

- the XmlExport might also benefit from the same boost as JsonExport.

I agree, but not in this task.

I'm placing this in ETF testing.

#5 - 03/21/2024 07:57 AM - Constantin Asofiei

- Status changed from WIP to Review

- % Done changed from 90 to 100

#6 - 03/21/2024 09:29 AM - Greg Shah

- Related to Feature #8494: rework XmlExport with the performance improvements from JsonExport added

#7 - 03/21/2024 10:59 AM - Constantin Asofiei

- Status changed from Review to Internal Test

There are regressions related to AdaptiveFind (from #8490), otherwise 8488a and #8451 passed.

I'm testing #8459 next - if this passes, too, I'll merge in this order:

1. #8459
2. #8451
3. #8488

#8 - 03/21/2024 11:08 AM - Greg Shah

I'm good with that plan.

#9 - 03/21/2024 12:42 PM - Ovidiu Maxiniuc

I have re-review the 8488a up to r15080 and I am OK with the changes.

The branch can be merged to trunk.

#10 - 03/21/2024 02:24 PM - Constantin Asofiei

- Status changed from Internal Test to Merge Pending

Merging now.

#11 - 03/21/2024 02:27 PM - Constantin Asofiei

- Status changed from Merge Pending to Test

Branch 8488a was merged into trunk as rev. 15084 and archived.