Database - Bug #7329

improve FieldReference resolution of getter/setter/extent accessors

05/08/2023 04:22 AM - Constantin Asofiei

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

Description

History

#1 - 05/08/2023 04:25 AM - Constantin Asofiei

Refactor FieldReference to lazily initialize the fields which are now in the constructor; this includes:

- getter
- setter
- index

Also, can getParentBuffer cache the buffer instance?

#2 - 05/09/2023 04:01 AM - Constantin Asofiei

Constantin Asofiei wrote:

> - index

Here I meant 'getExtent'

#3 - 05/09/2023 04:28 AM - Alexandru Lungu

- Status changed from New to WIP

- Assignee set to Radu Apetrii

Radu, please fix this in 7026d.

#4 - 05/09/2023 06:35 AM - Radu Apetrii

I've applied the suggested comments. The changes are on 7026d, rev. 14563.
I haven't tested yet with a large customer application, but I will get this done.

#5 - 05/10/2023 08:15 AM - Radu Apetrii

Radu Apetrii wrote:

> I haven't tested yet with a large customer application, but I will get this done.

I have identified a problem when running a customer application. I'm looking into this so that 7026d can be fully functional.

#6 - 05/10/2023 09:21 AM - Radu Apetrii

Radu Apetrii wrote:

> > I haven't tested yet with a large customer application, but I will get this done.
>
> I have identified a problem when running a customer application. I'm looking into this so that 7026d can be fully functional.

Done. There was a NPE encountered in FieldReference.getGetter. The new changes are on 7026d, rev. 14564.

#7 - 05/12/2023 09:28 AM - Alexandru Lungu

- % Done changed from 0 to 60

Review of 7026d 14563-14564:

- Extent can be null! Don't use null as a marker to check if it was computed or not. Use some separate variable (extentIsComputed).
- Note that having a null setter is also valid state if isIdAccessor is true. You only throw IllegalArgumentException if the setter is null **and** !isIdAccessor. Please use a separate setterIsComputed to ensure that null is actually generated and not just a lazy computation maker. If you want to have everything other control, you can also do a getterIsComputed, but this is optional.
- You have if (getter null) getter = getGetter(); if (getter null). Please reduce it to one single check getGetter() null. This also happens for extent. However, I think this is regarded to my last comment.
- You have if (bulk null) bulk = (bulk != null) && (extent != null) && (index == null). This makes the first condition (bulk != null) always true. Note that you have the constructor bulk you are no longer using. You shouldn't lazily initialize bulk. You can save the constructor value into a "allowBulk" or something like this and use it afterwards.
- Inline if (getter == null) getter = getGetter() return getter.getReturnType() into return getGetter().getReturnType(). The null check is already done in the getGetter. The same goes for getSetter and getExtent. The performance we try to save here is not from a simple native method call (which is already super-fast, especially with no parameters), but from some map look-ups and stuff like this. Now that you also have extentIsComputed and setterIsComputed, this whole check gets hard to manage, so leave the lazy checks **only** in getGetter, getSetter and getExtent.

#8 - 05/19/2023 05:21 AM - Radu Apetrii

I've refactored the code as you suggested. The changes are on 7026d, rev. 14570.
I found no issues while testing (with a customer app), but please let me know if you encounter any problems.

#9 - 05/22/2023 09:03 AM - Alexandru Lungu

- % Done changed from 60 to 90

Review of 7026d, rev. 14570

The changes are functionally good. I have only some minor suggestions

- In set, can you extract getSetter result into a local setter variable, so that Utils.invoke uses the local variable and not the class field? I am a bit afraid of accessing setter class field directly. Even if it isn't null due to the check, someone may decide to drop that IllegalStateException check and fail to use the right setter in the end. The same goes for getObject and unknownValue.
- In equals, this.extent = getExtent(); that.extent = that.getExtent(); doesn't look quite right. Can you use Objects.equals(getExtent(), that.getExtent()) directly?
- Please use !isBulkComputed instead of isBulkComputed false or isBulkComputed instead of isBulkComputed true.
- For the second constructor, you can set isSetterComputed on true, because the setter is null anyway. The same for bulk and extent.

#10 - 05/23/2023 06:22 AM - Radu Apetrii

Alexandru Lungu wrote:

> The changes are functionally good. I have only some minor suggestions
>
> - In set, can you extract getSetter result into a local setter variable, so that Utils.invoke uses the local variable and not the class field? I am a bit afraid of accessing setter class field directly. Even if it isn't null due to the check, someone may decide to drop that IllegalStateException check and fail to use the right setter in the end. The same goes for getObject and unknownValue.
> - In equals, this.extent = getExtent(); that.extent = that.getExtent(); doesn't look quite right. Can you use Objects.equals(getExtent(), that.getExtent()) directly?
> - Please use !isBulkComputed instead of isBulkComputed false or isBulkComputed instead of isBulkComputed true.
> - For the second constructor, you can set isSetterComputed on true, because the setter is null anyway. The same for bulk and extent.

Done. I applied the suggested changes. The commit is on 7026d, rev. 14574.

#11 - 05/24/2023 03:40 AM - Alexandru Lungu

- Status changed from WIP to Review

- % Done changed from 90 to 100

Review of 7026d/rev. 14574
I am OK with the changes. Have you tested the latest version (rev. 14574) of 7026d with a large customer application? I want to know if all the changes of 7026d are still doing fine now and we are on track with a stable 7026d.

#12 - 05/24/2023 04:08 AM - Radu Apetrii

Alexandru Lungu wrote:

> Review of 7026d/rev. 14574
> I am OK with the changes. Have you tested the latest version (rev. 14574) of 7026d with a large customer application? I want to know if all the changes of 7026d are still doing fine now and we are on track with a stable 7026d.

Yes, I have. Everything was fine, I just forgot to mention it.

#13 - 05/29/2023 09:43 AM - Alexandru Lungu

- Status changed from Review to Test

Merged 7026d to trunk as rev. 14587.
This can be closed.