Item 25
git_comments:

git_commits:

1. **summary:** HIVE-24241: Enable SharedWorkOptimizer to merge downstream operators after an optimization step (#1562) (Zoltan Haindrich reviewed by Jesus Camacho Rodriguez)
   message: HIVE-24241: Enable SharedWorkOptimizer to merge downstream operators after an optimization step (#1562) (Zoltan Haindrich reviewed by Jesus Camacho Rodriguez)

github_issues:

github_issues_comments:

github_pulls:

1. **title:** HIVE-24241: Enable SharedWorkOptimizer to merge downstream operators after an optimization step
   body: <!-- Thanks for sending a pull request! Here are some tips for you: 1. If this is your first time, please read our contributor guidelines: https://cwiki.apache.org/confluence/display/Hive/HowToContribute 2. Ensure that you have created an issue on the Hive project JIRA: https://issues.apache.org/jira/projects/HIVE/summary 3. Ensure you have added or run the appropriate tests for your PR: 4. If the PR is unfinished, add '[WIP]' in your PR title, e.g., '[WIP]HIVE-XXXXX: Your PR title ...'. 5. Be sure to keep the PR description updated to reflect all changes. 6. Please write your PR title to summarize what this PR proposes. 7. If possible, provide a concise example to reproduce the issue for a faster review. --> ### What changes were proposed in this pull request? <!-- Please clarify what changes you are proposing. The purpose of this section is to outline the changes and how this PR fixes the issue. If possible, please consider writing useful notes for better and faster reviews in your PR. See the examples below. 1. If you refactor some codes with changing classes, showing the class hierarchy will help reviewers. 2. If you fix some SQL features, you can provide some references of other DBMSes. 3. If there is design documentation, please add the link. 4. If there is a discussion in the mailing list, please add the link. --> ### Why are the changes needed? <!-- Please clarify why the changes are needed. For instance, 1. If you propose a new API, clarify the use case for a new API. 2. If you fix a bug, you can clarify why it is a bug. --> ### Does this PR introduce _any_ user-facing change? <!-- Note that it means *any* user-facing change including all aspects such as the documentation fix. If yes, please clarify the previous behavior and the change this PR proposes - provide the console output, description, screenshot and/or a reproducable example to show the behavior difference if possible. If possible, please also clarify if this is a user-facing change compared to the released Hive versions or within the unreleased branches such as master. If no, write 'No'. --> ### How was this patch tested? <!-- If tests were added, say they were added here. Please make sure to add some test cases that check the changes thoroughly including negative and positive cases if possible. If it was tested in a way different from regular unit tests, please clarify how you tested step by step, ideally copy and paste-able, so that other reviewers can test and check, and descendants can verify in the future. If tests were not added, please describe why they were not added and/or why it was difficult to add. -->

github_pulls_comments:

github_pulls_reviews:

1. nit. typo 'successfull'
2. SJ is gone. Is this expected?
3. I think you could use `ExprNodeDescExprFactory.isANDFuncCallExpr` or `FunctionRegistry.isOpAnd(expr)`?
4. Iirc we order the expressions intentionally in such a way that the rest of expressions are evaluated before the SJ expression, since the probe of the bloom filter is usually more expensive than evaluating other expressions (heuristic).
5. Same as above. Filter exprs order
6. SJ went away? It is kept in the other branch (L182).
7. Are these new TS that are not reused anymore? For instance, it seems this one is the same that was reused in old L224.
8. Change of algorithm to SHUFFLE. Is this expected? It seems the same changed happened for multiple ops in the plan.
9. SJ got removed. Is this expected?
10. conditional was not reconstructed properly during filter creation - fixed
11. I've tried to retain the order - which have placed the bloom related checks at the end. I recall that there was a ticket about ordering conditionals - but I can't find the related ticket; do I remember incorrectly?
12. changes are gone in this file
13. this is hightly unfortunate: the jsonexplain api "tells" the vertex about the outgoing edge type by calling [this method] (https://github.com/apache/hive/blob/db895f374bf63b77b683574fdf678bfac91a5ac6/common/src/java/org/apache/hadoop/hive/common/jsonexplain/Vertex.java#L3
    from [here] (https://github.com/apache/hive/blob/db895f374bf63b77b683574fdf678bfac91a5ac6/common/src/java/org/apache/hadoop/hive/common/jsonexplain/Stage.java#L11
    since a single vertex can have multiple outgoing edges - setting the type of one-of-them is problematic - I think we may want to consider to simple remove this tagging of vertices instead...we should consider renaming some of the edge types...like `CUSTOM_SIMPLE_EDGE` to `PARTITION_ONLY`
14. changes are gone
15. @jcamachor this is the checker class I was talking about - right now it builds on top of the basic `digraph` class I've introduce some time ago in `PointLookupOptimizer`
16. we can definetly roll our own graph representation; however sometimes I would feel that it would make things easier to have access to basic graph algorithms (for example to do a topological order walk/etc) there is a small library called [jgrapht](https://jgrapht.org/) (EPL 2.0 license - I think it will be okay) which could be utilized for these kind of things @jcamachor what do you think about pulling in the jgrapht lib and removing the makeshift digraph classes?
17. this is not called from anywhere right now - but I've used it during debug to get a better understanding of the plan
18. we see a case when NonBlockingOpDeDupProc merges FIL-FIL, the conditionals may be reorder. https://github.com/apache/hive/pull/1308
19. Can we maybe move it to a utility class and add a comment? You could maybe move all these classes to `optimizer.graph` package
20. Yes, I think it's a good idea to use an established graph library for this. I think we could create a follow-up for that and discuss the license too, to make sure there is no issue. Any advantage to using jgrapht over Guava graph implementation, given that guava is already a Hive dependency and quite widely used?
21. In the meantime, maybe DiGraph could be made a top class.
22. Looks good. Could we add comments so it is clear what it is being represented, e.g., what is a cluster, etc.
23. Left the comment in other class but I was thinking that it may be a good idea to promote this to top class (at least until we replace it by any other library version as we were discussing).
24. Note that the filter operator is removed. We need to be careful here because not all input formats guarantee that the filter expression is being applied / does not return false positives. I would expect the Filter remains but only a single time?
25. Any idea why this is happening?
26. Filter (same as mentioned previously).
27. new SJ?
28. good!
29. date_dim is only scanned once in the new plan as well - in this query the order of things have changed a bit so far it seems like the biggest change is that `Reducer 10` is exchanged with `Reducer 15`
30. jgrapht has quite a few graph algorithms as well - guava would be a viable candidate if we don't yet have a graph class; but I think it doesn't give much more than that. I've moved these classes around - let's see how far we can get with these!
31. yes; I've moved these classes - I use these methods during development; and they are very usefull; I needed to see the join tree - and I was able to extract a simple tree after a few minutes. I plan to expose this kind of information on the hs2 web interface in some way.

jira_issues:

1. **summary:** Enable SharedWorkOptimizer to merge downstream operators after an optimization step
   description:

jira_issues_comments:

1. merged into master. Thank you Jesus for reviewing the changes!