Item 377
git_comments:

git_commits:

1. **summary:** [GRIFFIN-205] accuracy matched fraction
   message: [GRIFFIN-205] accuracy matched fraction https://issues.apache.org/jira/browse/GRIFFIN-205 This pull request covers only batch dq type. We need to decide is it worth to add "matched fraction" to streaming type. Accuracy transformation tests added. Author: ashutak <ashutak@griddynamics.com> Closes #434 from ashutakGG/GRIFFIN-205-accuracy-matchedFraction.

github_issues:

github_issues_comments:

github_pulls:

1. **title:** Griffin-205: accuracy matched fraction
   body: https://issues.apache.org/jira/browse/GRIFFIN-205 This pull request covers only batch dq type. We need to decide is it worth to add "matched fraction" to streaming type. Accuracy transformation tests added.

github_pulls_comments:

github_pulls_reviews:

1. does it make sense to have aliases for inner selects, and have something like that: ``` `a`.`${totalColName}` ```
2. is it possible to compute `matchedFractionColName` here, without extra level of nesting?
3. assert on emptiness would be useful, what do you think?
4. is that reordering really required?
5. is it in sync with json serializer behavior? how map/list properties are handled if missing from the document?
6. **body:** Good point. For a sec I even thought you are right (an this is a potential bug). But the following query works as expected. `spark.sql("SELECT age FROM (SELECT count(age) as age FROM person)").show()` It show 2 as a result. So, looks like high level select does not have an access to tables used for inner select. But probably I will use this approach to increase readability.
   label: code-design
7. **body:** Whole point of nesting levels here to increase readability. Now, I have an idea where it will "good enough" to read even with 1 level of nesting.
   label: code-design
8. No, it's not. But keeping fields with default values as last elements in case classes is kind of "best practice" in scala. For example, scala compilation fails for the following snippet of code ``` case class Foo(a: Int, b: Int = 22, c: Double) val foo = Foo(1, 3.3) ``` For this particular class we can avoid reordering. But I see more reasons to do it right and don't see any reasons to keep it as is (except we are scared to change any pice of code).
9. Yes, it exactly is. Actually I expected such question :) There is a bug in `jackson` support for scala ["Case classes with option types don't follow serialization configuration"] (https://github.com/FasterXML/jackson-module-scala/issues/32). Long story short, It does not support any default values and puts `null`s everywhere. So, if we want to use this classes outside of jackson at the same way as with jackson, `null`s is only chose we have.
10. yeah. Fixed it with drop tables.
11. LGTM
12. Ok, got it!
13. @bhlx3lyx7 @chemikadze has this broken Apache Griffin UI side? Not sure how UI consume from ES.
14. this alias LGTM.
15. add one more metrics, ok.
16. As I know, UI calculates the matched fraction by total and matched count, in ES the metrics are just persisted in JSON format, it's OK to add a field here.

jira_issues:

1. **summary:** Accuracy measure check should provide matchedFraction to store
   description: Currently, {{accuracy}} measure results contains "total", "miss" and "matched" counts. As a result, It's hard to analyze accuracy fraction based on results stored in ElasticSearch,
   because ElasticSearch does not provide straight forward capability to get "field divided by field" query results. {{Accuracy}} measure results should also contain {{matchedFraction}} field.

jira_issues_comments:

1. GitHub user ashutakGG opened a pull request: https://github.com/apache/incubator-griffin/pull/434 Griffin-205: accuracy matched fraction https://issues.apache.org/jira/browse/GRIFFIN-205 This pull request covers only batch dq type. We need to decide is it worth to add "matched fraction" to streaming type. Accuracy transformation tests added. You can merge this pull request into a Git repository by running: $ git pull https://github.com/ashutakGG/incubator-griffin GRIFFIN-205-accuracy-matchedFraction Alternatively you can review and apply these changes as the patch at: https://github.com/apache/incubator-griffin/pull/434.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #434 ---- commit a5c90ed258281408c196008e97b682e6337d5a5c Author: ashutak <ashutak@...> Date: 2018-10-10T13:59:55Z GRIFFIN-205: Added basic tests for accuracy dqType. commit 9df3188def1c6076bf0618d73b5f1e73e93f522f Author: ashutak <ashutak@...> Date: 2018-10-10T15:38:34Z GRIFFIN-205: Added matchedFraction for Batch dq type ----
2. Github user asfgit closed the pull request at: https://github.com/apache/incubator-griffin/pull/434
3. **body:** In my opinion, the "total", "miss" and "matched" counts in accuracy measure results are raw metrics, they could be aggregated in later calculation. But the "matchedFraction" field is a calculated metrics, which could not be used in later aggregation. Combining these metrics in the same metric value might mislead the users. I think it's OK to add "matchedFraction" field in accuracy metrics, but we need to clarify the difference between the count metric and fraction metric in document. BTW, if we add this field in batch mode, it would be better to keep the consistency in streaming mode as well.
   label: code-design
4. [~artem.shutak] as we've merged this change, can you please close the ticket?