PHOENIX-6232

symat

2020-11-23

Contents

Chapter 1

Root issue PHOENIX-6232

1.1 Summary

Correlated subquery should not push to RegionServer as the probe side of the Hash join

1.2 Description

We were facing an interesting problem when a more complex query (with inner selects in the WHERE clause) succeeds alone, while the same query fails, if it is part of a join. I created a test table / query to reproduce the problem:

```
DROP TABLE IF EXISTS test;
CREATE TABLE test (
      id INTEGER NOT NULL,
      test_id INTEGER,
      lastchanged TIMESTAMP,
      CONSTRAINT my_pk PRIMARY KEY (id));

UPSERT INTO test VALUES(0, 100, '2000-01-01 00:00:00.0');
UPSERT INTO test VALUES(1, 101, '2000-01-01 00:00:00.0');
UPSERT INTO test VALUES(2, 100, '2011-11-11 11:11:11.0');
```

Query 1: Example query, running fine in itself:

```
SELECT id, test_id, lastchanged FROM test T
WHERE lastchanged = ( SELECT max(lastchanged) FROM test WHERE test_id = T.test_id )

Returns:
+----+---------+-------------------------+
| ID | TEST_ID |      LASTCHANGED        |
+----+---------+-------------------------+
| 1  | 101     | 2000-01-01 01:00:00.0 |
| 2  | 100     | 2011-11-11 12:11:11.0 |
+----+---------+-------------------------+
```

Query 2: Same query fails on the current master branch, when it is part of a larger (implicit) join:

```
SELECT AAA.*
```

```
FROM(
  SELECT id, test_id, lastchanged FROM test T
  WHERE lastchanged = ( SELECT max(lastchanged) FROM test WHERE test_id = T.test_id )
) as AAA,
(
  SELECT id FROM test
) as BBB
WHERE AAA.id = BBB.id;


java.lang.IllegalArgumentException
    at org.apache.phoenix.thirdparty.com.google.common.base.Preconditions.checkArgument(
        Preconditions.java:128)
    at
org.apache.phoenix.compile.TupleProjectionCompiler.createProjectedTable(
    TupleProjectionCompiler.java:66)
    at org.apache.phoenix.compile.QueryCompiler.compileSingleFlatQuery(QueryCompiler.
        java:663)
    at org.apache.phoenix.compile.QueryCompiler.compileJoinQuery(QueryCompiler.java:404)
    at org.apache.phoenix.compile.QueryCompiler.compileJoinQuery(QueryCompiler.java:302)
    at org.apache.phoenix.compile.
QueryCompiler.compileSelect(QueryCompiler.java:249)
    at org.apache.phoenix.compile.QueryCompiler.compile(QueryCompiler.java:176)
    at org.apache.phoenix.jdbc.PhoenixStatement$ExecutableSelectStatement.compilePlan(
        PhoenixStatement.java:504)
    at org.apache.phoenix.jdbc.PhoenixStatement$ExecutableSelectStatement.compilePlan(
        PhoenixStatement.java:467)
    at org.apache.phoenix.jdbc.PhoenixStatement$1.call
(PhoenixStatement.java:309)
    at org.apache.phoenix.jdbc.PhoenixStatement$1.call(PhoenixStatement.java:298)
    at org.apache.phoenix.call.CallRunner.run(CallRunner.java:53)
    at org.apache.phoenix.jdbc.PhoenixStatement.executeQuery(PhoenixStatement.java:297)
    at org.apache.phoenix.jdbc.PhoenixStatement.executeQuery(PhoenixStatement.java:290)
    at org.apache.phoenix.jdbc.PhoenixStatement.execute(Phoenix
Statement.java:1933)
    at sqlline.Commands.executeSingleQuery(Commands.java:1054)
    at sqlline.Commands.execute(Commands.java:1003)
    at sqlline.Commands.sql(Commands.java:967)
    at sqlline.SqlLine.dispatch(SqlLine.java:734)
    at sqlline.SqlLine.begin(SqlLine.java:541)
    at sqlline.SqlLine.start(SqlLine.java:267)
    at sqlline.SqlLine.main(SqlLine.java:206)
```

I am not sure what the problem is exactly. My guess is that Phoenix tries to optimize (flatten) an inner-query, which it shouldn't, if we are inside a join (according to the check in the code which throws the exception).

The best workaround I found was to define an explicit join in the original query (Query 1), basically change the inner select into a join. This modified query return the same as the original one:

Query 3:

```
SELECT T.id, T.test_id, T.lastchangedFROM  test T  LEFT JOIN (
    SELECT max(lastchanged) AS max_timestamp,          test_id AS max_timestamp_test_id
    FROM test
    GROUP BY test_id
  ) JOIN_TABLE ON JOIN_TABLE.max_timestamp_test_id = T.test_id
WHERE T.lastchanged = JOIN_TABLE.max_timestamp

Returns:
+------+-----------+----------------------+
```

```
| T.ID | T.TEST
_ID |     T.LASTCHANGED     |
+------+-----------+-----------------------+
| 1    | 101       | 2000-01-01 01:00:00.0 |
| 2    | 100       | 2011-11-11 12:11:11.0 |
+------+-----------+-----------------------+
```

Query 4: And the same modified query (query 3) now works inside a join:

```
SELECT AAA.*
FROM(
  SELECT T.id, T.test_id, T.lastchanged  FROM    test T    LEFT JOIN (
      SELECT max(lastchanged) AS max_timestamp,             test_id AS
          max_timestamp_test_id
      FROM test
      GROUP BY test_id
    ) JOIN_TABLE ON JOIN_TABLE.max_timestamp_test_id = T.test_id
  WHERE T.lastchanged = JOIN_TABLE.max_timestamp
) as AAA,
(
  SELECT id FROM te
st
) as BBB
WHERE AAA.id = BBB.id;

Returns:
+------+-----------+-----------------------+
| T.ID | T.TEST_ID |     T.LASTCHANGED     |
+------+-----------+-----------------------+
| 1    | 101       | 2000-01-01 01:00:00.0 |
| 2    | 100       | 2011-11-11 12:11:11.0 |
+------+-----------+-----------------------+
```

I think Query 4 worked, as it is forcing Phoenix to drop the idea of optimizing it's inner-query (Query 3). Although, I can be wrong about the root cause...

Anyway, I think the bug should be fixed and Query 2 should run without exception.

1.3 Attachments

1. PHOENIX-6232_v1-4.x.patch
2. PHOENIX-6232_v1-master.patch

1.4 Comments

1. **comnetwork:** [~symat]ïijŇ thank you very much for the report. It is indeed a bug, I would try to fix it.

   I reproduced your problem on the latest branch 4.x, but the exception is different from your reported exception, what is your phoenix version?

2. **symat:** [~comnetwork], thanks for looking into this!

>I reproduced your problem on the latest branch 4.x, but the exception is different from your reported exception, what is your phoenix version?

First I saw this issue on our downstream phoenix, which is based on 5.0.0 (but might contain some newer commits cherry-picked). Then I tried to reproduce the same problem on other versions. The stacktrace I copied here come froma phoenix I built from the apache master branch 2 days ago. (I haven't tested this on any 4.x versions yet)

3. **comnetwork:** [~symat]. thank you very much, I would try to fix this and to test on the 4.x and master.
4. **wangchao316:** I reproduct this issues in 5.0.0 branch. this exception is same as query 2.
5. **githubbot:** comnetwork opened a new pull request #992:
URL: https://github.com/apache/phoenix/pull/992

This is an automated message from the Apache Git Service.
To respond to the message, please log on to GitHub and use the
URL above to go to the specific comment.

For queries about this service, please contact Infrastructure at:
users@infra.apache.org

6. **comnetwork:** [~symat], I uploaded my patch to fix this bug, I think the root cause is that

```
1
2  SELECT id, test_id, lastchanged FROM test T
3
4   WHERE lastchanged = ( SELECT max(lastchanged) FROM test WHERE test_id = T.test_id
       )
```

has Correlated subquery ({{SELECT max(lastchanged) FROM test WHERE test_id = T.test_id}}), which is a join itself, so it could not as the probe side of the Hash join.

You may also use /*+ USE_SORT_MERGE_JOIN*/ hint to get around this bug.

7. **symat:** [~comnetwork], thanks for the very quick fix! :)

I'm going to build your patch and also execute the original query failed for us in production.

8. **githubbot:** symat commented on pull request #992:
URL: https://github.com/apache/phoenix/pull/992#issuecomment-736616782

I can not comment on the code (I'm not familiar with Phoenix internals) but I executed the original production query that failed for us before the patch, and after applying this patch the query succeed. Thanks for the fix!

This is an automated message from the Apache Git Service.
To respond to the message, please log on to GitHub and use the
URL above to go to the specific comment.

For queries about this service, please contact Infrastructure at:
users@infra.apache.org

9. **symat:** I executed the original production query that failed for us before the patch, and after applying this patch the query succeed. I tested it on branch 4.x, as the PR was submitted against 4.x. Thanks for the fix again!

   Do you plan to also cherry-pick the change to master? (I don't know how the branching works in Phoenix... but it would be great to have this fix in the next 5.x release too)

10. **githubbot:** stoty commented on pull request #992:
    URL: https://github.com/apache/phoenix/pull/992#issuecomment-736651838

    :broken_heart: **-1 overall**

    | Vote | Subsystem | Runtime | Comment |

    |:----:|----------:|--------:|:--------|

    | +0 :ok: | reexec | 5m 7s | Docker mode activated. |

    ||| _ Prechecks _ |

    | +1 :green_heart: | dupname | 0m 0s | No case conflicting files found. |

    | +1 :green_heart: | hbaseanti | 0m 0s | Patch does not have any anti-patterns. |

    | +1 :green_heart: | @author | 0m 0s | The patch does not contain any @author tags. |

    | +1 :green_heart: | test4tests | 0m 0s | The patch appears to include 2 new or modified test files. |

||| _ 4.x Compile Tests _ |

| +1 :green_heart: | mvninstall | 11m 16s | 4.x passed |

| +1 :green_heart: | compile | 0m 55s | 4.x passed |

| +1 :green_heart: | checkstyle | 1m 33s | 4.x passed |

| +1 :green_heart: | javadoc | 0m 43s | 4.x passed |

| +0 :ok: | spotbugs | 2m 56s | phoenix-core in 4.x has 950 extant spotbugs warnings. |

||| _ Patch Compile Tests _ |

| +1 :green_heart: | mvninstall | 5m 24s | the patch passed |

| +1 :green_heart: | compile | 0m 55s | the patch passed |

| +1 :green_heart: | javac | 0m 55s | the patch passed |

| -1 :x: | checkstyle | 1m 39s | phoenix-core: The patch generated 121 new + 2612 unchanged - 83 fixed = 2733 total (was 2695) |

| +1 :green_heart: | whitespace | 0m 0s | The patch has no whitespace issues. |

| -1 :x: | javadoc | 0m 41s | phoenix-core generated 1 new + 99 unchanged - 1 fixed = 100 total (was 100) |

| +1 :green_heart: | spotbugs | 3m 6s | the patch passed |

||| _ Other Tests _ |

| -1 :x: | unit | 206m 57s | phoenix-core in the patch failed. |

| +1 :green_heart: | asflicense | 0m 35s | The patch does not generate ASF License warnings. |

| | | 244m 42s | |

| Reason | Tests |

|-------:|:------|

| Failed junit tests | phoenix.end2end.PointInTimeQueryIT |

| Subsystem | Report/Notes |

|----------:|:-------------|

| Docker | ClientAPI=1.40 ServerAPI=1.40 base: https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-

PreCommit-GitHub-PR/job/PR-992/1/artifact/yetus-general-check/output/Dockerfile |

| GITHUB PR | https://github.com/apache/phoenix/pull/992 |

| JIRA Issue | PHOENIX-6232 |

| Optional Tests | dupname asflicense javac javadoc unit spotbugs hbaseanti checkstyle compile |

| uname | Linux a25f0bde3789 4.15.0-65-generic #74-Ubuntu SMP Tue Sep 17 17:06:04 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux |

| Build tool | maven |

| Personality | dev/phoenix-personality.sh |

| git revision | 4.x / 18b9f76 |

| Default Java | Private Build-1.8.0_242-8u242-b08-0ubuntu3~16.04-b08 |

| checkstyle | https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-PreCommit-GitHub-PR/job/PR-992/1/artifact/yetus-general-check/output/diff-checkstyle-phoenix-core.txt |

| javadoc | https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-PreCommit-GitHub-PR/job/PR-992/1/artifact/yetus-general-check/output/diff-javadoc-javadoc-phoenix-core.txt |

| unit | https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-PreCommit-GitHub-PR/job/PR-992/1/artifact/yetus-general-check/output/patch-unit-phoenix-core.txt |

| Test Results | https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-PreCommit-GitHub-PR/job/PR-992/1/testReport/ |

| Max. process+thread count | 6170 (vs. ulimit of 30000) |

| modules | C: phoenix-core U: phoenix-core |

| Console output | https://ci-hadoop.apache.org/job/Phoenix/job/Phoenix-PreCommit-GitHub-PR/job/PR-992/1/console |

| versions | git=2.7.4 maven=3.3.9 spotbugs=4.1.3 |

| Powered by | Apache Yetus 0.12.0 https://yetus.apache.org |

This message was automatically generated.

For queries about this service, please contact Infrastructure at:
users@infra.apache.org

11. **comnetwork:** [~symat],Thank you for quick feedback, I would make a patch for the master also.
12. **comnetwork:** Uploaded my patch for the master.
13. **symat:** Thank you [~comnetwork]!! :)