Item 280
git_comments:

1. -----ORGANIZATION_ID DESC CONTAINER_ID ASC
2. -----ORGANIZATION_ID
3. -----ORGANIZATION_ID DESC CONTAINER_ID DESC
4. -----CONTAINER_ID ASC ORGANIZATION_ID ASC
5. -----CONTAINER_ID ASC ORGANIZATION_ID DESC
6. -----ORGANIZATION_ID ASC CONTAINER_ID DESC
7. ----CONTAINER_ID
8. -----ORGANIZATION_ID ASC CONTAINER_ID ASC
9. -----CONTAINER_ID DESC ORGANIZATION_ID DESC
10. -----CONTAINER_ID DESC ORGANIZATION_ID ASC
11. -----ORGANIZATION_ID DESC CONTAINER_ID ASC
12. -----ORGANIZATION_ID
13. -----ORGANIZATION_ID DESC CONTAINER_ID DESC
14. -----CONTAINER_ID ASC ORGANIZATION_ID ASC
15. -----CONTAINER_ID ASC ORGANIZATION_ID DESC
16. -----ORGANIZATION_ID ASC CONTAINER_ID DESC
17. ----CONTAINER_ID
18. -----ORGANIZATION_ID ASC CONTAINER_ID ASC
19. -----CONTAINER_ID DESC ORGANIZATION_ID DESC
20. -----CONTAINER_ID DESC ORGANIZATION_ID ASC

git_commits:

1. **summary:** PHOENIX-3452 NULLS FIRST/NULL LAST should not impact whether GROUP BY is order preserving (chenglei)
   message: PHOENIX-3452 NULLS FIRST/NULL LAST should not impact whether GROUP BY is order preserving (chenglei)

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** NULLS FIRST/NULL LAST should not impact whether GROUP BY is order preserving
   description: This may be related to PHOENIX-3451 but the behavior is different so filing it separately. Steps to repro: CREATE TABLE IF NOT EXISTS TEST.TEST (ORGANIZATION_ID CHAR(15) NOT NULL, CONTAINER_ID CHAR(15) NOT NULL, ENTITY_ID CHAR(15) NOT NULL, SCORE DOUBLE, CONSTRAINT TEST_PK PRIMARY KEY (ORGANIZATION_ID, CONTAINER_ID, ENTITY_ID)) VERSIONS=1, MULTI_TENANT=TRUE, REPLICATION_SCOPE=1, TTL=31536000; CREATE INDEX IF NOT EXISTS TEST_SCORE ON TEST.TEST (CONTAINER_ID, SCORE DESC, ENTITY_ID DESC); UPSERT INTO test.test VALUES ('org1','container1','entityId6',1.1); UPSERT INTO test.test VALUES ('org1','container1','entityId5',1.2); UPSERT INTO test.test VALUES ('org1','container1','entityId4',1.3); UPSERT INTO test.test VALUES ('org1','container1','entityId3',1.4); UPSERT INTO test.test VALUES ('org1','container1','entityId2',1.5); UPSERT INTO test.test VALUES ('org1','container1','entityId1',1.6); SELECT DISTINCT entity_id, score FROM test.test WHERE organization_id = 'org1' AND container_id = 'container1' ORDER BY score DESC Notice that the returned results are not returned in descending score order. Instead they are returned in descending entity_id order. If I remove the DISTINCT or remove the secondary index the result is correct.

jira_issues_comments:

1. [~giacomotaylor]
2. As suspected, this is duplicate of PHOENIX-3451.
3. I think the problem is cause by the GroupByCompiler, when GroupBy.compile method called the OrderPreservingTracker.track method to track the groupBy expression's order,just as following(in GroupByCompiler.java): {code:borderStyle=solid} 144 if (isOrderPreserving) { 145 OrderPreservingTracker tracker = new OrderPreservingTracker(context, GroupBy.EMPTY_GROUP_BY, Ordering.UNORDERED, expressions.size(), tupleProjector); 146 for (int i = 0; i < expressions.size(); i++) { 147 Expression expression = expressions.get(i); 148 tracker.track(expression); 149 } {code} The track method inappropriately used the sortOrder != SortOrder.getDefault() as the thrid "isNullsLast" parameter as following(in OrderPreservingTracker.java): {code:borderStyle=solid} 101 public void track(Expression node) { 102 SortOrder sortOrder = node.getSortOrder(); 103 track(node, sortOrder, sortOrder != SortOrder.getDefault()); 104 } 105 106 public void track(Expression node, SortOrder sortOrder, boolean isNullsLast) { {code} Once the node's SortOrder is SortOrder.DESC, the "isNullsLast" is true. it affected the GroupBy 's isOrderPreserving as following(in OrderPreservingTracker.java) : {code:borderStyle=solid} 141 if (node.isNullable()) { 142 if (!Boolean.valueOf(isNullsLast).equals(isReverse)) { 143 isOrderPreserving = false; 144 isReverse = false; 145 return; 146 } 147 } {code} As far as the sql "SELECT DISTINCT entity_id,score FROM test.test WHERE organization_id = 'org1' AND container_id = 'container1' ORDER BY score DESC" is concerned, GroupBy 's "isOrderPreserving" property is false,but the OrderByComilper thinks the OrderBy is OrderBy.FWD_ROW_KEY_ORDER_BY. Because the GroupBy 's "isOrderPreserving" property is false,so the aggregated results returned by the RegionServer's GroupedAggregateRegionObserver is unordered, and after the AggregatePlan of client side gets the aggregated results, the results only be sorted by the aggregated key through MergeSortRowKeyResultIterator, so the final results is not as expected.If the GroupBy 's "isOrderPreserving" property is true,the final results is ok. Actually, the "isNullsLast" parameter is just related to orderBy ,it should only affected the display order of "Null " in the sorted results , groupBy should not be affetced by "isNullsLast". I wrote a simple unit test to reproduce this problem in my patch: {code:borderStyle=solid} @Test public void testGroupByDesc() throws Exception { Connection conn = DriverManager.getConnection(getUrl()); try { conn.createStatement().execute("DROP TABLE IF EXISTS GROUPBYDESC_TEST"); String sql="CREATE TABLE IF NOT EXISTS GROUPBYDESC_TEST ("+ "ORGANIZATION_ID VARCHAR,"+ "CONTAINER_ID VARCHAR,"+ "CONSTRAINT TEST_PK PRIMARY KEY ("+ "ORGANIZATION_ID DESC,"+ "CONTAINER_ID DESC"+ "))"; conn.createStatement().execute(sql); sql="SELECT ORGANIZATION_ID, CONTAINER_ID,count(*) FROM GROUPBYDESC_TEST group by ORGANIZATION_ID, CONTAINER_ID"; PhoenixPreparedStatement statement = conn.prepareStatement(sql).unwrap(PhoenixPreparedStatement.class); QueryPlan queryPlan = statement.optimizeQuery(sql); queryPlan.iterator(); assertTrue(queryPlan.getGroupBy().isOrderPreserving()); } finally { conn.close(); } } {code} I uploaded my patch, [~jamestaylor], please review.
4. **body:** NULLS FIRST/LAST can matter if there's an ORDER BY. Try adding a similar test with an ORDER BY ORGANIZATION_ID, CONTAINER_ID NULL LAST test. For orderPreserving to be true, then NULLS FIRST has to be true or we can do a reverse scan. It's a bit tricky - needs more tests.
   label: test
5. **body:** [~jamestaylor],ok ,I will add more tests.
   label: test
6. [~jamestaylor], I updated my patch,enhanced my test cases following your suggestion.
7. Thanks, [~comnetwork]. Try these tests as end-to-end tests and one of them will be incorrect, as either the nulls will appear first or last: {code} + sql="SELECT ORGANIZATION_ID, CONTAINER_ID,count(*) FROM GROUPBYDESC_TEST group by ORGANIZATION_ID, CONTAINER_ID order by ORGANIZATION_ID NULLS LAST, CONTAINER_ID NULLS FIRST"; + queryPlan = getQueryPlan(conn, sql); + assertTrue(queryPlan.getGroupBy().isOrderPreserving()); + assertTrue(queryPlan.getOrderBy().getOrderByExpressions().size()==2); + assertTrue(queryPlan.getOrderBy().getOrderByExpressions().get(0).toString().equals("ORGANIZATION_ID NULLS LAST")); + assertTrue(queryPlan.getOrderBy().getOrderByExpressions().get(1).toString().equals("CONTAINER_ID")); + + sql="SELECT ORGANIZATION_ID, CONTAINER_ID,count(*) FROM GROUPBYDESC_TEST group by ORGANIZATION_ID, CONTAINER_ID order by ORGANIZATION_ID NULLS LAST, CONTAINER_ID NULLS LAST"; + queryPlan = getQueryPlan(conn, sql); + assertTrue(queryPlan.getGroupBy().isOrderPreserving()); + assertTrue(queryPlan.getOrderBy()==OrderBy.REV_ROW_KEY_ORDER_BY); + {code} You need to take into account the NULLS FIRST/LAST in some way if there's an ORDER BY.

8. Ok,[~jamestaylor],I will try some IT tests, and actually every OrderBy expression can have different NULLS LAST / NULLS FIRST., just like the ASC and DESC. For example, the clause "order by ORGANIZATION_ID NULLS LAST, CONTAINER_ID NULLS FIRST" is legal,. Before I added the IT tests, I need to make sure the meaning of "NULLS LAST" I understood is right: In my opinion, the "NULLS LAST" means the "NULL" always behind other "non-NULL" results in the sorted results, no matter the OrderBy is ASC or DESC. The "NULLS FIRST" means the "NULL" always in front of other "non-NULL" results in the sorted results,no matter the OrderBy is ASC or DESC. So if we have a following table: {code:borderStyle=solid} CREATE TABLE GROUPBYDESC_TEST (ORGANIZATION_ID VARCHAR, CONTAINER_ID VARCHAR, ENTITY_ID VARCHAR NOT NULL, CONSTRAINT TEST_PK PRIMARY KEY (ORGANIZATION_ID DESC, CONTAINER_ID DESC, ENTITY_ID)); UPSERT INTO GROUPBYDESC_TEST VALUES ('a',null,'11'); UPSERT INTO GROUPBYDESC_TEST VALUES (null,'2','22'); UPSERT INTO GROUPBYDESC_TEST VALUES ('c','3','33'); UPSERT INTO GROUPBYDESC_TEST VALUES (null,null,'44'); {code} For the following sql: {code:borderStyle=solid} select ORGANIZATION_ID, CONTAINER_ID,count(*) from GROUPBYDESC_TEST group by ORGANIZATION_ID, CONTAINER_ID order by ORGANIZATION_ID NULLS LAST, CONTAINER_ID NULLS FIRST {code} the expecting result is: {code:borderStyle=solid} a, null , 1 c, 3 , 1 null, null, 1 null, 2, 1 {code} For the following sql: {code:borderStyle=solid} select ORGANIZATION_ID, CONTAINER_ID,count(*) from GROUPBYDESC_TEST group by ORGANIZATION_ID, CONTAINER_ID order by ORGANIZATION_ID NULLS LAST, CONTAINER_ID NULLS LAST {code} the expecting result is: {code:borderStyle=solid} a, null, 1 c, 3, 1 null, 2, 1 null, null, 1 {code} [~jamestaylor],am I right ?
9. Yes, that's correct.
10. [~jamestaylor], when I wrote my IT tests,I found another new problem about NULLS LAST/NULLS FIRST, this new problem is irrelevant to PHOENIX-3452,but it indeed interfere with my IT tests, so maybe we should fix it first: PHOENIX-3469 I uploaded my patch in PHOENIX-3469,[~jamestaylor],please help me review,thanks.
11. With the patch for PHOENIX-3469 applied, can you add the tests I mentioned here[1] and see if any changes to this patch are required, [~comnetwork]? [1] https://issues.apache.org/jira/secure/EditComment!default.jspa?id=13018032&commentId=15650176
12. **body:** ok, [~jamestaylor],I will add tests soon.
    label: test
13. Looking at this a bit more, I think your patch is good, [~comnetwork]. The NULL FIRST/LAST should not impact the GroupBy.isOrderPreserving. It will impact if the ORDER BY can be compiled out, but this is done separately (which is the way we want it). It would be good to just add asserts to your tests in QueryCompilerTest that confirm that the ORDER BY is only compiled out when it should be (i.e. compiled out means that plan.getOrderBy().getOrderByExpressions().isEmpty() is true).
14. yes,my assert in QueryCompilerTest had already conside this point,it assert queryPlan.getOrderBy()==OrderBy.FWD_ROW_KEY_ORDER_BY or queryPlan.getOrderBy()==OrderBy.REV_ROW_KEY_ORDER_BY when it should be.
15. [~jamestaylor],I uploaded a new patch: PHOENIX-3452_v3.patch, enhanced unit tests and IT tests,these test cases will be ok after the PHOENIX-3469 is applied,and I will uploaed a new patch after the PHOENIX-3469 is applied. [~jamestaylor],just have a look,thanks. By the way,in my unit tests and IT tests,I only test "group by ORGANIZATION_ID, CONTAINER_ID", not "group by CONTAINER_ID,ORGANIZATION_ID", because I found PHOENIX-3451 interfering with "group by CONTAINER_ID,ORGANIZATION_ID" test cases. I will add "group by CONTAINER_ID,ORGANIZATION_ID" test cases in PHOENIX-3451 patch. I think the order of fixing these three bugs is: 1.PHOENIX-3469. 2.PHOENIX-3452 3.PHOENIX-3451,which is a very serious and important bug.
16. Ok, thanks! Do you have a patch almost ready for PHOENIX-3451?
17. [~jamestaylor], yes, I will upload my first patch for PHOENIX-3451 soon.
18. FAILURE: Integrated in Jenkins build Phoenix-master #1494 (See [https://builds.apache.org/job/Phoenix-master/1494/]) PHOENIX-3452 NULLS FIRST/NULL LAST should not impact whether GROUP BY is (jamestaylor: rev 6fd8e72045fd0a8808cb774f7f5f7a96ef0cef79) * (edit) phoenix-core/src/it/java/org/apache/phoenix/end2end/GroupByCaseIT.java * (edit) phoenix-core/src/test/java/org/apache/phoenix/compile/QueryCompilerTest.java * (edit) phoenix-core/src/main/java/org/apache/phoenix/compile/OrderPreservingTracker.java