Item 364
git_comments:

1. **comment:** this test case must use a real predicate, not alwaysTrue(), or binding will simplify it out
   label: code-design
2. TODO: Enable this test after #961 is fixed shouldRead = shouldRead(notNull("map_not_null")); Assert.assertTrue("Should read: map type is not skipped", shouldRead);
3. no_nulls column has all values == "", so notIn("no_nulls", "") should always be false and so should be skipped However, the metrics evaluator in Parquets always reads row group for a notIn filter
4. build struct field schema
5. ORC-623: ORC does not skip a row group for a notNull predicate on a column with all nulls boolean shouldRead = shouldRead(notNull("all_nulls"));
6. min=30, max=79, num-nulls=0 value longer than 4k will produce no stats in Parquet, but will produce stats for ORC required, always non-null never non-null includes some null values optional, but always non-null
7. struct with int
8. ORC-623: ORC seems to incorrectly skip a row group for a notIn(column, {X, ...}) predicate on a column which has only 1 non-null value X but also has nulls
9. min=30, max=79, num-nulls=0 value longer than 4k will produce no stats in Parquet required, always non-null never non-null includes some null values optional, but always non-null
10. create 50 records
11. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
12. TODO: Uncomment this after #961 is fixed ORCSchemaUtil.buildOrcProjection has a bug which does not allow addition of container types with nested required children. Enable this field after #961 is fixed along with commented tests for the same column below optional(11, "map_not_null", Types.MapType.ofRequired(12, 13, StringType.get(), IntegerType.get()))
13. TODO: Enable this test after #961 is fixed shouldRead = shouldRead(isNull("map_not_null")); Assert.assertTrue("Should read: map type is not skipped", shouldRead);
14. uuid, fixed and binary types not supported yet new Object[] { "orc", "uuid", uuid, UUID.randomUUID() }, new Object[] { "orc", "fixed", "abcd".getBytes(StandardCharsets.UTF_8), new byte[] { 0, 1, 2, 3 } }, new Object[] { "orc", "binary", "xyz".getBytes(StandardCharsets.UTF_8), new byte[] { 0, 1, 2, 3, 4, 5 } },
15. create 50 records
16. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
17. record.setField("_uuid", uuid); // Disable writing UUID value as GenericParquetWriter does not handle UUID type correctly; Also UUID tests are disabled for both ORC and Parquet anyway
18. new Object[] { "parquet", "uuid", uuid, UUID.randomUUID() }, // not supported yet
19. 2.0f, 1.99f, 1.98f, ... 2.0d, 2.01d, 2.02d, ...
20. ORC SearchArguments do not have a greaterThan predicate, so we use not(lessThanOrEquals) e.g. x > 5 => not(x <= 5)
21. ORC SearchArguments do not have a greaterThanOrEquals predicate, so we use not(lessThan) e.g. x >= 5 => not(x < 5)
22. Currently every predicate in ORC requires a PredicateLeaf.Type field which is not available for these Iceberg types
23. Cannot push down STARTS_WITH operator to ORC, so return TruthValue.YES_NO_NULL which signifies that this predicate cannot help with filtering
24. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
25. Cannot push down predicates for types which cannot be represented in PredicateLeaf.Type, so return TruthValue.YES_NO_NULL which signifies that this predicate cannot help with filtering
26. * * Generates mapping from field IDs to ORC qualified names. * <p> * This visitor also enclose column names in backticks i.e. ` so that ORC can correctly parse column names with * special characters. A comparison of ORC convention with Iceberg convention is provided below * <pre> * Iceberg ORC * field field field * struct -> field struct.field struct.field * list -> element list.element list._elem * list -> struct element -> field list.field list._elem.field * map -> key map.key map._key * map -> value map.value map._value * map -> struct key -> field map.key.field map._key.field * map -> struct value -> field map.field map._value.field * </pre>

27. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
28. if the column name contains ` then escape it with another `
29. for columns not in the file, buildOrcProjection will append field names with _r<ID> this will be passed down to ORC, but ORC will handle such cases and return a TruthValue during evaluation
30. use optional fields for performing isNull checks because Iceberg itself resolves them for required fields
31. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
32. all operations for these types should resolve to YES_NO_NULL
33. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
34. * * Generates mapping from field IDs to ORC qualified names. See {@link IdToOrcName} for details.

git_commits:

1. **summary:** ORC: Push down Iceberg filters (#973)
   message: ORC: Push down Iceberg filters (#973)

github_issues:

1. **title:** Iceberg ORC should push down predicates to the ORC reader
   body: When we create the ORC file reader options in [link](https://github.com/apache/incubator-iceberg/blob/e965d41648df3361cfb323563dec6d21c216fa51/orc/src/main/java/org/apache/iceberg/orc/OrcIterable.java#L74) we should push down predicates using [`Reader.Options.setSearchArgument`](https://orc.apache.org/api/orc-core/org/apache/orc/Reader.Options.html#searchArgument-org.apache.hadoop.hive.ql.io.sarg.SearchArgument-java.lang.String:A-)
2. **title:** Iceberg ORC should push down predicates to the ORC reader
   body: When we create the ORC file reader options in [link](https://github.com/apache/incubator-iceberg/blob/e965d41648df3361cfb323563dec6d21c216fa51/orc/src/main/java/org/apache/iceberg/orc/OrcIterable.java#L74) we should push down predicates using [`Reader.Options.setSearchArgument`](https://orc.apache.org/api/orc-core/org/apache/orc/Reader.Options.html#searchArgument-org.apache.hadoop.hive.ql.io.sarg.SearchArgument-java.lang.String:A-)
3. **title:** Iceberg ORC should push down predicates to the ORC reader
   body: When we create the ORC file reader options in [link](https://github.com/apache/incubator-iceberg/blob/e965d41648df3361cfb323563dec6d21c216fa51/orc/src/main/java/org/apache/iceberg/orc/OrcIterable.java#L74) we should push down predicates using [`Reader.Options.setSearchArgument`](https://orc.apache.org/api/orc-core/org/apache/orc/Reader.Options.html#searchArgument-org.apache.hadoop.hive.ql.io.sarg.SearchArgument-java.lang.String:A-)

github_issues_comments:

1. Planning to take this up next week unless someone is working on it

github_pulls:

1. **title:** Push down Iceberg expressions to the ORC reader
   body: Fixes #787 - Adds ExpressionToSearchArgument visitor which converts Iceberg Expression to ORC SearchArgument. Most of the changes are isolated to this new class. Other changes are just to pass the expression from Iceberg to ORC and tests - Added unit tests for the same - Copied tests equivalent to `TestMetricsRowGroupFilter` and `TestMetricsRowGroupFilterTypes` from the parquet module to `TestSearchArgument` and `TestSearchArgumentTypes`. Had to put the in `iceberg-data` as `iceberg-orc` lacks a usable readerFunc/writerFunc. - Some tests are hit by to #961 and some due to [ORC-623](https://issues.apache.org/jira/browse/ORC-623) so I have disabled them for now

github_pulls_comments:

1. @omalley would be great to have your review on this as well
2. **body:** Overall, this looks great and is very thorough. Thanks @shardulm94!
   label: code-design
3. I plan to review this this week. Thanks @rdblue, @shardulm94 !
4. This is ready for another round of review
5. The test failure is python, so it shouldn't block this. I'll look into why it's failing.
6. +1 Looks like this conflicts with recent merges, so we'll have to fix those. Otherwise, this looks good to me. The test failure isn't happening in master or other PRs, so I think it's safe to merge this with the python failure.
7. Merged. Nice work, @shardulm94! Thank you for working on this, it's a great feature to have for ORC users.

github_pulls_reviews:

1. **body:** Should we move the Parquet tests into the iceberg-data module as well so that we can share the test code? It would be great to use parameterized tests (with if statements to exclude cases) instead of copying so much code.
   label: code-design
2. **body:** Does this need to be public?
   label: code-design
3. Is there a better verb than `visit` to describe this? How about `convert`?
4. **body:** Nit: missing `.` at the end of this sentence. This is also a bit long for a description. What about "Generates a map from field ID to ORC qualified name."
   label: code-design
5. Javadoc won't insert a new paragraph for a blank line so you need to add `<p>`.
6. **body:** There is no `greaterThan`? Could you add a comment here to explain that it's using `not("x" <= 5)` instead of `x > 5`?
   label: documentation
7. **body:** As a follow-up, you might consider rewriting this to use the new before/after methods instead of the custom order visitor. We just converted `IndexByName` and the implementation is much easier to read and maintain.
   label: code-design
8. Can we move this to its own test suite? I don't think it should be mixed with the expression conversion tests.
9. I'd prefer to move `IdToQuotedColumnName` to a top-level class, like `IdToOrcName` and add it as a utility method in `ORCSchemaUtil`. That will make it available for other uses so we don't implement it twice because we don't know it exists.
10. Fixed
11. Done
12. Done
13. Done
14. Done
15. Done
16. Done
17. Done. Moved TestMetricsRowGroupFilter and TestMetricsRowGroupFilterTypes into iceberg-data and merged with their ORC counterparts using parameterized tests.
18. Done

jira_issues:

1. **summary:** Potentially incorrect Sarg evaluation for not(in) and not(isNull)
   description: I seem to have stumbled upon two issues with respect to Sarg evaluation in ORC I have created two test cases at [https://github.com/shardulm94/orc/commit/b6d97cfa0325d2a14094456d338c942f61b887f2] for the same In the first case, applying {{not(isNull(column))}} on a column that has all null values seems to incorrectly mark the row group as needed. This is a rather benign issue though as some extra row groups are returned. In the second case, I create a column which has only 2 potential values, either null or 1 based on whether the row index is even or odd. So all row groups are guaranteed to have both null and 1. Applying {{not(in(column, 1))}} on this column incorrectly marks the row group as not needed. There are null values in the row group which should be matched by {{notIn(column, 1)}}. This is potentially causing some row groups to be filtered out incorrectly.

jira_issues_comments:

1. Thank you for creating a JIRA with the test case.
2. **body:** Thank you very much for the bug report with the unit test cases. That helped a lot. The first case, we weren't handling some of the types with all null values correctly. The second case is actually a typical misunderstanding of how null works in SQL. When int1 is null, "not int1 in (1)" returns null. So, the predicate is returns either false or null and thus no rows should be returned. For the record, it took me a long time to get it straight and I still get it wrong more than I'd like to admit.
   label: test
3. I committed this. Thanks for the review, Gang and Shardul!
4. Released as part of 1.6.4.