Item 200
git_comments:

1. Missing schema fields from the passed mapping
2. Passed wrong schema type

git_commits:

1. **summary:** ARROW-5655: [Python] Table.from_pydict/from_arrays not using types in specified schema correctly
   message: ARROW-5655: [Python] Table.from_pydict/from_arrays not using types in specified schema correctly The behaviour still seems a bit inconsistent to me: - from_arrays enforces to pass the same amount of fields as arrays - from_pydict enables to pass a schema with a subset of mapping's fields Both methods requires to pass a schema instance although. We could improve it by allowing to pass a list of fields or tuples, to allow the same inputs like `pa.schema` does. cc @jorisvandenbossche @pitrou Closes #5567 from kszucs/ARROW-5655 and squashes the following commits: 1da0547ac <Krisztián Szűcs> flake8 4b08c8e2e <Krisztián Szűcs> schema validation Authored-by: Krisztián Szűcs <szucs.krisztian@gmail.com> Signed-off-by: Wes McKinney <wesm+git@apache.org>

github_issues:

github_issues_comments:

github_pulls:

1. **title:** ARROW-5655: [Python] Table.from_pydict/from_arrays not using types in specified schema correctly
   body: The behaviour still seems a bit inconsistent to me: - from_arrays enforces to pass the same amount of fields as arrays - from_pydict enables to pass a schema with a subset of mapping's fields Both methods requires to pass a schema instance although. We could improve it by allowing to pass a list of fields or tuples, to allow the same inputs like `pa.schema` does. cc @jorisvandenbossche @pitrou
   label: code-design

github_pulls_comments:

1. https://issues.apache.org/jira/browse/ARROW-5655
2. **body:** > from_arrays enforces to pass the same amount of fields as arrays I think this is in any case to be expected, as there is otherwise no way to know which array maps to which field > from_pydict enables to pass a schema with a subset of mapping's fields This is similar as `from_pandas`. I think both the current behaviour as raising an error on fields in the dict/dataframe that are not in the schema are decent options. I personally find the current behaviour a bit more convenient.
   label: code-design
3. # [Codecov](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=h1) Report > Merging [#5567](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=desc) into [master](https://codecov.io/gh/apache/arrow/commit/af097e67ac1f06aa8c9ed3f5d60d21816e820fc0?src=pr&el=desc) will **decrease** coverage by `22.34%`. > The diff coverage is `87.34%`. [![Impacted file tree graph](https://codecov.io/gh/apache/arrow/pull/5567/graphs/tree.svg?width=650&token=LpTCFbqVT1&height=150&src=pr)](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=tree) ```diff @@ Coverage Diff @@ ## master #5567 +/- ## ========== - Coverage 88.79% 66.44% -22.35% ========== Files 983 516 -467 Lines 132170 71219 -60951 Branches 1501 0 -1501 ========== - Hits 117362 47323 -70039 - Misses 14443 23896 +9453 + Partials 365 0 -365 ``` | [Impacted Files](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=tree) | Coverage Δ | | |---|---|---| | [python/pyarrow/_orc.pyx](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvX29yYy5weXg=) | `77.77% <ø> (ø)` | :arrow_up: | | [cpp/src/arrow/filesystem/s3fs.h](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-Y3BwL3NyYy9hcnJvdy9maWxlc3lzdGVtL3MzZnMuaA==) | `100% <ø> (ø)` | :arrow_up: | | [python/pyarrow/feather.pxi](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-

cHl0aG9uL3B5YXJyb3cvZmVhdGhlci5weGk=) | `82.22% <ø> (ø)` | :arrow_up: | | [python/pyarrow/serialization.pxi](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvc2VyaWFsaXphdGlvbi5weGk=) | `83.85% <0%> (ø)` | :arrow_up: | | [python/pyarrow/_fs.pyx](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvX2ZzLnB5eA==) | `96.82% <100%> (ø)` | :arrow_up: | | [python/pyarrow/fs.py](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvZnMucHk=) | `100% <100%> (ø)` | :arrow_up: | | [python/pyarrow/_csv.pyx](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvX2Nzdi5weXg=) | `99.28% <100%> (ø)` | :arrow_up: | | [python/pyarrow/s3fs.py](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvczNmcy5weQ==) | `100% <100%> (ø)` | :arrow_up: | | [cpp/src/arrow/filesystem/s3fs.cc](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-Y3BwL3NyYy9hcnJvdy9maWxlc3lzdGVtL3MzZnMuY2M=) | `74.89% <100%> (-16.47%)` | :arrow_down: | | [python/pyarrow/tests/test_io.py](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree#diff-cHl0aG9uL3B5YXJyb3cvdGVzdHMvdGVzdF9pby5weQ==) | `96.98% <100%> (ø)` | :arrow_up: | | ... and [737 more](https://codecov.io/gh/apache/arrow/pull/5567/diff?src=pr&el=tree-more) | | ------ [Continue to review full report at Codecov](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=continue). > **Legend** - [Click here to learn more](https://docs.codecov.io/docs/codecov-delta) > `Δ = absolute <relative> (impact)`, `ø = not affected`, `? = missing data` > Powered by [Codecov](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=footer). Last update [af097e6...1da0547](https://codecov.io/gh/apache/arrow/pull/5567?src=pr&el=lastupdated). Read the [comment docs](https://docs.codecov.io/docs/pull-request-comments).

github_pulls_reviews:

1. This is already tested below (test_table_from_pydict_schema), so maybe update the `match` check there instead
2. **body:** It tests multiple missing keys, besides below is a parametrized test case and this is just a sanity check.
   label: test
3. ```suggestion raise TypeError('schema must be an instance of pyarrow.Schema') ``` (the argument is lower case)
4. **body:** You can also update the other test to have 2 missing fields, or remove the other test (and move the remaining part about less fields in the schema than in the data here), if you prefer. I personally just think there is no need tin testing the exact same thing twice (our tests are already difficult enough to navigate and find what is tested where). This is all a minor comment, though, so see what you do with it ;)
   label: code-design

jira_issues:

1. **summary:** [Python] Table.from_pydict/from_arrays not using types in specified schema correctly
   description: Example with {{from_pydict}} (from https://github.com/apache/arrow/pull/4601#issuecomment-503676534): {code:python} In [15]: table = pa.Table.from_pydict(...: {'a': [1, 2, 3], 'b': [3, 4, 5]}, ...: schema=pa.schema([('a', pa.int64()), ('c', pa.int32())])) In [16]: table Out[16]: pyarrow.Table a: int64 c: int32 In [17]: table.to_pandas() Out[17]: a c 0 1 3 1 2 0 2 3 4 {code} Note that the specified schema has 1) different column names and 2) has a non-default type (int32 vs int64) which leads to corrupted values. This is partly due to {{Table.from_pydict}} not using the type information in the schema to convert the dictionary items to pyarrow arrays. But then it is also {{Table.from_arrays}} that is not correctly casting the arrays to another dtype if the schema specifies as such. Additional question for {{Table.pydict}} is whether it actually should override the 'b' key from the dictionary as column 'c' as defined in the schema (this behaviour depends on the order of the dictionary, which is not guaranteed below python 3.6).

jira_issues_comments:

1. [~kszucs] I think this might already be fixed in the mean-time. Wes and I did some work related to schema handling the last month
2. Issue resolved by pull request 5567 [https://github.com/apache/arrow/pull/5567]