Item 204
git_comments:

git_commits:

1. **summary:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   message: ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns arrow is now pandas datetime timezone aware Author: ahnj <ahnj@yahoo.com> Closes #287 from ahnj/timestamp-aware and squashes the following commits: 0221ed0 [ahnj] ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns

github_issues:

github_issues_comments:

github_pulls:

1. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
2. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
3. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
4. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
5. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
6. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
   label: documentation
7. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
8. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
9. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
   body: arrow is now pandas datetime timezone aware
10. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware
11. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware
12. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware
13. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware
14. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware
15. **title:** ARROW-378: Python: Respect timezone on conversion of Pandas datetime columns
    body: arrow is now pandas datetime timezone aware

github_pulls_comments:

1. I left a comment on https://issues.apache.org/jira/browse/ARROW-378
2. @ahnj can you rebase on the the lastest master? this is based on a commit from 10 days ago https://github.com/ahnj/arrow/commits/timestamp-aware
3. @wesm - rebased with the latest master.

github_pulls_reviews:

1. To my knowledge, all pandas data having a time zone is UTC -- if data is time zone naive, then we may assume UTC in Arrow. I don't think any conversions are necessary here
2. **body:** It would be better to put this in a `Notes` section, or in the Sphinx documentation
   label: documentation
3. From my comment above, you can look at `x.values.view(np.int64)` before and after `tz_convert('US/Eastern')` to check that the internal values are not modified. If you localize with a non-UTC timestamp, that actually changes the values
4. Can you combine the null and non-null test case?
5. I still don't believe this is necessary. ```python In [11]: dr = pd.date_range('2000-01-01', periods=10) In [12]: dr_utc = dr.tz_localize('UTC') In [13]: dr_utc.values.view(np.int64) Out[13]: array([946684800000000000, 946771200000000000, 946857600000000000, 946944000000000000, 947030400000000000, 947116800000000000, 947203200000000000, 947289600000000000, 947376000000000000, 947462400000000000]) In [14]: dr_utc.tz_convert('US/Eastern').values.view(np.int64) Out[14]: array([946684800000000000, 946771200000000000, 946857600000000000, 946944000000000000, 947030400000000000, 947116800000000000, 947203200000000000, 947289600000000000, 947376000000000000, 947462400000000000]) ```
6. Oops.. now I see what you mean! let me go back and fix, and re-rebase.
7. Can you revert the code changes in this function?
8. Can you revise to say "Localized timestamps will currently be returned as UTC (pandas's native representation). Timezone-naive data will be implicitly interpreted as UTC".
9. Revised with suggested note.
10. Reverted and rebased. Thanks.

jira_issues:

1. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
2. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
   label: code-design
3. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
4. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
5. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
6. **summary:** Python: Respect timezone on conversion of Pandas datetime columns

description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.

7. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.
8. **summary:** Python: Respect timezone on conversion of Pandas datetime columns
   description: Currently we convert columns irrespectively of their timezone directly to timestamps in Arrow. While in Pandas you are able to specify timezones, in Arrow we have the assumptions that a Timestamp is always the time of the specified unit since 00:00:00.000 on 1 January 1970, UTC. Thus we have to check if {{df['column‘].tz}} is set and convert if necessary to UTC before converting to an Arrow column.

jira_issues_comments:

1. PR: https://github.com/apache/arrow/pull/287
2. **body:** [~xhochy] I think there is some confusion because pandas stores timestamps internally as UTC. I am not sure what needs to be done exactly
   label: code-design
3. If the values are internally stored as UTC there is nothing to do then. I wasn't aware of that fact.
4. [~ahnj] sorry for the confusion, could you close your pull request? Thanks
5. No worries. I'm happy to do so, but please consider taking the current pull request as this 'implicit conversion' to UTC by numpy is documented and explicitly tested. It may help others in the future avoid being fooled in a similar manner. :)
6. I agree having the extra test is useful -- I left a comment on the PR. Can you rebase and then I can merge? Thanks