Item 206
git_comments:

git_commits:

1. **summary:** ARROW-3473: [Format] Clarify that 64-bit lengths and null counts are permitted, but not recommended
   message: ARROW-3473: [Format] Clarify that 64-bit lengths and null counts are permitted, but not recommended The Arrow metadata was changed from the initial format specification to permit 64-bit array lengths, support for which is provided by the C++ library. This clarifies that 64-bit lengths are permissible, but for best compatibility (e.g. with Java), it is recommended to use 32-bits or less in practice. see also #2733 Author: Wes McKinney <wesm+git@apache.org> Closes #2734 from wesm/ARROW-3473 and squashes the following commits: 11b4b08a9 <Wes McKinney> Add note about representing large data sets using chunks 067ec9932 <Wes McKinney> Clarify that 64-bit lengths and null counts are permitted, but not recommended for best inter-language compatibility

github_issues:

1. **title:** Spec array size int32 or int64?
   body: The spec mentions clearly that the array size and null count are signed int32_t (https://github.com/apache/arrow/blob/master/format/Layout.md#array-lengths) However, it appears to have changed to int64_t in C++ implementation (https://github.com/apache/arrow/blob/master/cpp/src/arrow/array.h#L171) So I was wondering wether the spec needs an update or if C++ just internally uses int64_t?
   label: code-design
2. **title:** Spec array size int32 or int64?
   body: The spec mentions clearly that the array size and null count are signed int32_t (https://github.com/apache/arrow/blob/master/format/Layout.md#array-lengths) However, it appears to have changed to int64_t in C++ implementation (https://github.com/apache/arrow/blob/master/cpp/src/arrow/array.h#L171) So I was wondering wether the spec needs an update or if C++ just internally uses int64_t?
3. **title:** Spec array size int32 or int64?
   body: The spec mentions clearly that the array size and null count are signed int32_t (https://github.com/apache/arrow/blob/master/format/Layout.md#array-lengths) However, it appears to have changed to int64_t in C++ implementation (https://github.com/apache/arrow/blob/master/cpp/src/arrow/array.h#L171) So I was wondering wether the spec needs an update or if C++ just internally uses int64_t?
4. **title:** Spec array size int32 or int64?
   body: The spec mentions clearly that the array size and null count are signed int32_t (https://github.com/apache/arrow/blob/master/format/Layout.md#array-lengths) However, it appears to have changed to int64_t in C++ implementation (https://github.com/apache/arrow/blob/master/cpp/src/arrow/array.h#L171) So I was wondering wether the spec needs an update or if C++ just internally uses int64_t?

github_issues_comments:

1. **body:** Hi -- best to discuss on JIRA. Yes, we permit 64-bit lengths. An implementation is valid if it does not, though. For best compatibility between languages, we recommend limiting lengths to 32 bits.
   label: code-design
2. https://issues.apache.org/jira/browse/ARROW-3473

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** [Format] Update Layout.md document to clarify use of 64-bit array lengths
   description: See https://github.com/apache/arrow/issues/2733. While 64-bit lengths are permitted, it is recommended to limit array sizes to 32-bit length or less. I will update
2. **summary:** [Format] Update Layout.md document to clarify use of 64-bit array lengths
   description: See https://github.com/apache/arrow/issues/2733. While 64-bit lengths are permitted, it is recommended to limit array sizes to 32-bit length or less. I will update
3. **summary:** [Format] Update Layout.md document to clarify use of 64-bit array lengths
   description: See https://github.com/apache/arrow/issues/2733. While 64-bit lengths are permitted, it is recommended to limit array sizes to 32-bit length or less. I will update

jira_issues_comments:

1. Issue resolved by pull request 2734 [https://github.com/apache/arrow/pull/2734]