Item 202
**git_comments:**

1. [[arrow::export]]
2. some known special cases
3. Reserve enough space before appending
4. Recurse.
5. append
6. convert to an R object to store as the list' ptype
7. Build an empty array to match value_type
8. nothing to do, list contain NULL by default
9. datatype.cpp
10. array.cpp
11. [[arrow::export]]
12. persistently protect `x` and return it
13. returns the namespace environment for package `name`
14. return R string vector, e.g. strings({"foo", "bar"}) returns a size 2 STRSXP

**git_commits:**

1. **summary:** ARROW-9291 [R]: Support fixed size binary/list types
   **message:** ARROW-9291 [R]: Support fixed size binary/list types some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package](https://reprex.tidyverse.org) (v0.3.0.9001)</sup> Closes #7660 from romainfrancois/ARROW-9291/FixedSize Lead-authored-by: Romain Francois <romain@rstudio.com> Co-authored-by: Neal Richardson <neal.p.richardson@gmail.com> Signed-off-by: Neal Richardson <neal.p.richardson@gmail.com>

**github_issues:**

**github_issues_comments:**

**github_pulls:**

1. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package](https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
2. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #>

<fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>

3. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
4. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
5. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
6. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1]

01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>

7. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
   **label:** code-design
8. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
9. **title:** ARROW-9291 [R]: Support fixed size binary/list types
   **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
10. **title:** ARROW-9291 [R]: Support fixed size binary/list types
    **body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
    **label:** documentation
11. **title:** ARROW-9291 [R]: Support fixed size binary/list types

**body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>

12. **title:** ARROW-9291 [R]: Support fixed size binary/list types
**body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
13. **title:** ARROW-9291 [R]: Support fixed size binary/list types
**body:** some progress here: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(vctrs::list_of(as.raw(1:4), as.raw(1:4)), type = fixed_size_binary(4L)) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 # we can skip the type= we have an R object that inherits from vctrs_list_of, # with class arrow_fixed_size_binary and a byte_width attribute raws <- vctrs::new_list_of( list(as.raw(1:4), as.raw(1:4)), ptype = raw(), byte_width = 4L, class = "arrow_fixed_size_binary" ) a <- Array$create(raws) a #> Array #> <fixed_size_binary[4]> #> [ #> 01020304, #> 01020304 #> ] v <- a$as_vector() v #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 str(v) #> fixed_size_binary<4> [1:2] #> $ : raw [1:4] 01 02 03 04 #> $ : raw [1:4] 01 02 03 04 ``` <sup>Created on 2020-07-07 by the [reprex package] (https://reprex.tidyverse.org) (v0.3.0.9001)</sup>

**github_pulls_comments:**

1. I think we need a function for the `raws <- vctrs::new_list_of(...)` part, and probably an equivalent function for the other binary types, but `binary()`, `large_binary()` and `fixed_size_binary()` which would be the obvious choice is already taken. I think that in this function: ``` template <> std::shared_ptr<arrow::DataType> InferArrowTypeFromVector<VECSXP>(SEXP x) { if (Rf_inherits(x, "data.frame") || Rf_inherits(x, "POSIXlt")) { return InferArrowTypeFromDataFrame(x); } else { if (Rf_inherits(x, "arrow_fixed_size_binary")) { SEXP byte_width = Rf_getAttrib(x, symbols::byte_width); if (Rf_isNull(byte_width) || TYPEOF(byte_width) != INTSXP || XLENGTH(byte_width) != 1) { Rcpp::stop("malformed arrow_fixed_size_binary object"); } return arrow::fixed_size_binary(INTEGER(byte_width)[0]); } SEXP ptype = Rf_getAttrib(x, symbols::ptype); if (Rf_isNull(ptype)) { if (XLENGTH(x) == 0) { Rcpp::stop( "Requires at least one element to infer the values' type of a list vector"); } ptype = VECTOR_ELT(x, 0); } // special case list(raw()) -> BinaryArray if (TYPEOF(ptype) == RAWSXP) { return arrow::binary(); } return arrow::list(InferArrowType(ptype)); } } ``` we should rather dispatch based on an r type for binary and large binary, as it's done here for fixed size binary, and forget the magic conversion of a `vctrs_list_of<raw>` to a `BinaryArray`, as we might actually want to create an Array of type `List<uint8>`: ``` r arrow::list_of(arrow::uint8()) #> ListType #> list<item: uint8> ``` <sup>Created on 2020-07-07 by the [reprex package](https://reprex.tidyverse.org) (v0.3.0.9001)</sup> but type deduction currently short circuits it.
2. https://issues.apache.org/jira/browse/ARROW-9291

3. Let me see if I understand the issues you're concerned with. We have some inconsistency in how types are converted: `raw` -> `uint8`, but `uint8` -> `integer`, for example. Here, `binary` -> `list_of(raw)`, and we've special-cased `list_of(raw)` -> `binary`, but you might think that it should be `list_of(uint8)` given how the non-list type is converted. I'm fine with getting rid of the special-casing of `list_of(raw)` to `binary` and having the binary conversion set an R class (subclass of vctrs list of or whatever), which we use to determine that the list of raw should be converted back to BinaryArray. I'm not particularly worried that someone would have a list of raw vectors in R and be frustrated that that wasn't converted to a list of uint8--seems unlikely, and if you really care, you can always specify a type/schema. But it seems reasonable to be explicit and set a class attribute.
4. Some more progress today. ``` r library(arrow, warn.conflicts = FALSE) # with explicit type= # no deduction, but testing raws <- vctrs::list_of(as.raw(1:4), as.raw(1:4)) a1 <- Array$create(raws, type = binary()) a2 <- Array$create(raws, type = large_binary()) a3 <- Array$create(raws, type = fixed_size_binary(4L)) a1$as_vector() #> <binary[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 a2$as_vector() #> <large_binary[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 a3$as_vector() #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 # with type deduced from class, we should have functions for this. a1 <- Array$create( vctrs::new_list_of(list(as.raw(1:4), as.raw(1:4)), class = "arrow_binary") ) a2 <- Array$create( vctrs::new_list_of(list(as.raw(1:4), as.raw(1:4)), class = "arrow_large_binary") ) a3 <- Array$create( vctrs::new_list_of(list(as.raw(1:4), as.raw(1:4)), class = "arrow_fixed_size_binary", byte_width = 4L) ) a1$as_vector() #> <binary[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 a2$as_vector() #> <large_binary[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 a3$as_vector() #> <fixed_size_binary<4>[2]> #> [[1]] #> [1] 01 02 03 04 #> #> [[2]] #> [1] 01 02 03 04 ``` <sup>Created on 2020-07-08 by the [reprex package](https://reprex.tidyverse.org) (v0.3.0.9001)</sup> Will tackle FixedSizeList on the same model. Now that we have the R classes `arrow_binary`, `arrow_large_binary`, and `arrow_fixed_size_binary`, perhaps we don't really need to inherit from `vctrs_list_of`, I actually had to backpedal it with some methods to make it look sort of ok: ```r # vctrs support ---------- #' @importFrom vctrs vec_ptype_full vec_ptype_abbr obj_str_footer #' @method vec_ptype_full arrow_fixed_size_binary #' @export vec_ptype_full.arrow_fixed_size_binary <- function(x, ...) { paste0("fixed_size_binary<", attr(x, "byte_width"), ">") } #' @method vec_ptype_full arrow_binary #' @export vec_ptype_full.arrow_binary <- function(x, ...) { "binary" } #' @method vec_ptype_full arrow_large_binary #' @export vec_ptype_full.arrow_large_binary <- function(x, ...) { "large_binary" } #' @method vec_ptype_abbr arrow_fixed_size_binary #' @export vec_ptype_abbr.arrow_fixed_size_binary <- vec_ptype_full.arrow_fixed_size_binary #' @method vec_ptype_abbr arrow_binary #' @export vec_ptype_abbr.arrow_binary <- vec_ptype_full.arrow_binary #' @method vec_ptype_abbr arrow_large_binary #' @export vec_ptype_abbr.arrow_large_binary <- vec_ptype_full.arrow_large_binary #' @method obj_str_footer arrow_fixed_size_binary #' @export obj_str_footer.arrow_fixed_size_binary <- function(x, ..., indent.str = "", nest.lev = 0) { invisible(NULL) } #' @method obj_str_footer arrow_binary #' @export obj_str_footer.arrow_binary <- obj_str_footer.arrow_fixed_size_binary #' @method obj_str_footer arrow_large_binary #' @export obj_str_footer.arrow_large_binary <- obj_str_footer.arrow_fixed_size_binary ```
5. So now either : - we have a compatible list, i.e. a list of raw vectors and we request `type = binary/large_binary/fixed_size_binary`. - we have R objects (that still have to be list of raw vectors) with these dedicated classes In any case the conversion from arrow to R makes these classes regardless.
6. Further progress re FixedSizeList: ``` r library(arrow, warn.conflicts = FALSE) a <- Array$create(list(1:4), type = fixed_size_list_of(int32(), 4L)) a #> Array #> <fixed_size_list<item: int32>[4]> #> [ #> [ #> 1, #> 2, #> 3, #> 4 #> ] #> ] # back to R a$as_vector() #> <fixed_size_list<integer, 4>[1]> #> [[1]] #> [1] 1 2 3 4 str(a$as_vector()) #> fixed_size_list<integer, 4>[1:1] #> $ : int [1:4] 1 2 3 4 #> @ list_size: int 4 #> @ ptype : int(0) ``` <sup>Created on 2020-07-09 by the [reprex package](https://reprex.tidyverse.org) (v0.3.0.9001)</sup>
7. Looks like this conflicts with the Dictionary PR I just merged
8. **body:** Should be better now 🤞🏾
   **label:** code-design

**github_pulls_reviews:**

1. Should remove the byte_width arg since it's no longer used
2. **body:** Would you mind adding a comment/docstring to these new functions? It's not obvious what they do just from their names.
   **label:** documentation

3. Should footnote and explain what these `arrow_` types are (at least that they're subclasses of list/vctrs::list_of)
4. That's no longer the case for the `*binary` ones, they inherit from `vctrs_vctr` mostly for printing purposes. I'll add some footnotes.

**jira_issues:**

1. **summary:** [R] Support fixed size binary/list types
   **description:**
2. **summary:** [R] Support fixed size binary/list types
   **description:**
3. **summary:** [R] Support fixed size binary/list types
   **description:**

**jira_issues_comments:**

1. Issue resolved by pull request 7660 [https://github.com/apache/arrow/pull/7660]