`_source` data types

- The data type doesn't reflect how the values are indexed
- `_source` contains the field values supplied at index time
- It's common to use `_source` values from search results
    - You would probably expect a string for a `keyword` field
- We can modify the `_source` value while reindexing
- Alternatively this can be handled at the application level

Removing fields

- Field mappings cannot be deleted
- Fields can be left out when indexing documents
- Maybe we want to reclaim disk space used by a field
  - Already indexed values still take up disk space
  - For large data sets, this may be worthwhile
    - Assuming that we no longer need the values

Using `ctx.op` within scripts

- Usually, using the `query` parameter is possible
- For more advanced use cases, `ctx.op` can be used
- Using the `query` parameter is better performance wise and is preferred
- Specifying `"delete"` deletes the document within the destination index
  - The destination index might not be empty as in our example
  - The same can often be done with the Delete by Query API

Parameters for the Reindex API

- More parameters are available than the ones we covered
  - E.g. for handling version conflicts
- A snapshot is created before reindexing documents
- A version conflict causes the query to be aborted by default
- The destination index is not necessarily empty

Batching & throttling

- The Reindex API performs operations in batches
  - Just like the Update by Query and Delete by Query APIs
  - It uses the Scroll API internally
  - This is how millions of documents can be reindexed efficiently
- Throttling can be configured to limit the performance impact
  - Useful for production clusters
- Check the documentation if you need to reindex lots of documents