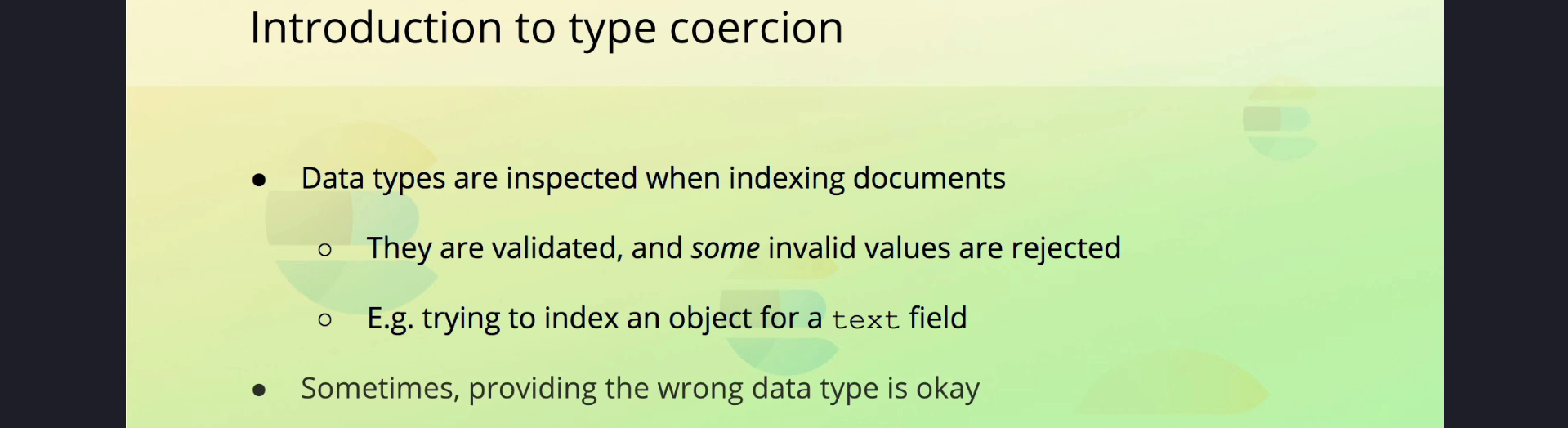
# Understanding Type Coercion in Elasticsearch



## 1. Introduction to Type Coercion

Type coercion is a process in Elasticsearch where field values are validated and, in some cases, converted to match the expected data type. This process ensures that documents with slightly incorrect data types can still be indexed, provided the values can be coerced.

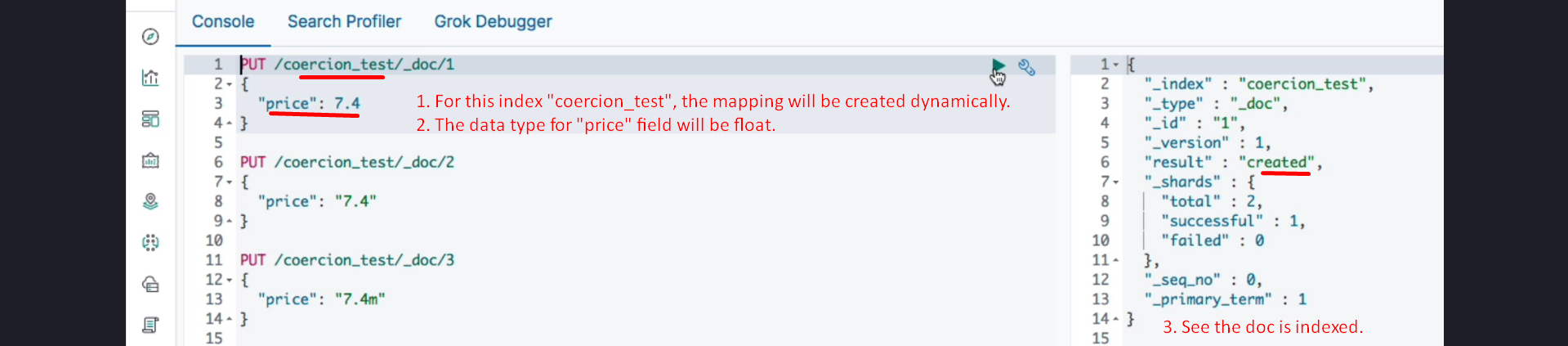
For example, attempting to index an object for a text field will result in an error. However, in some cases, Elasticsearch can convert the supplied data type to the correct one. This behavior makes Elasticsearch more forgiving.

## 2. Demonstrating Type Coercion

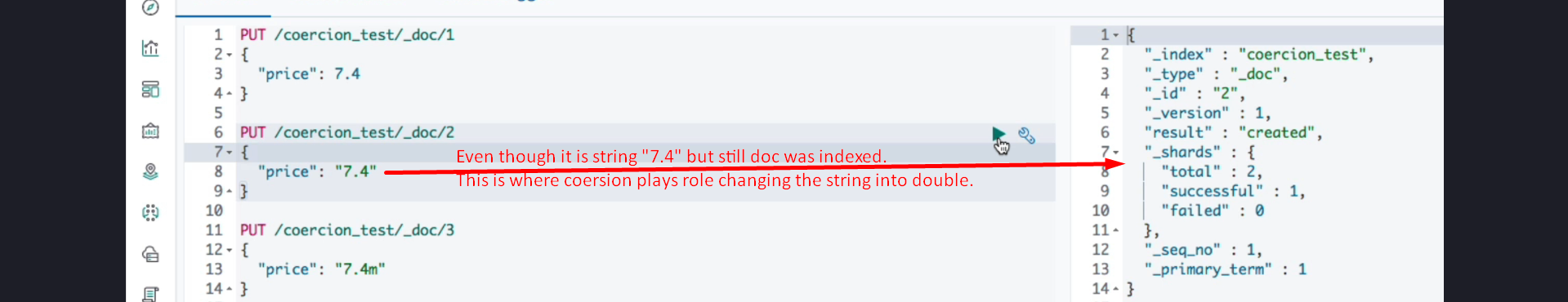
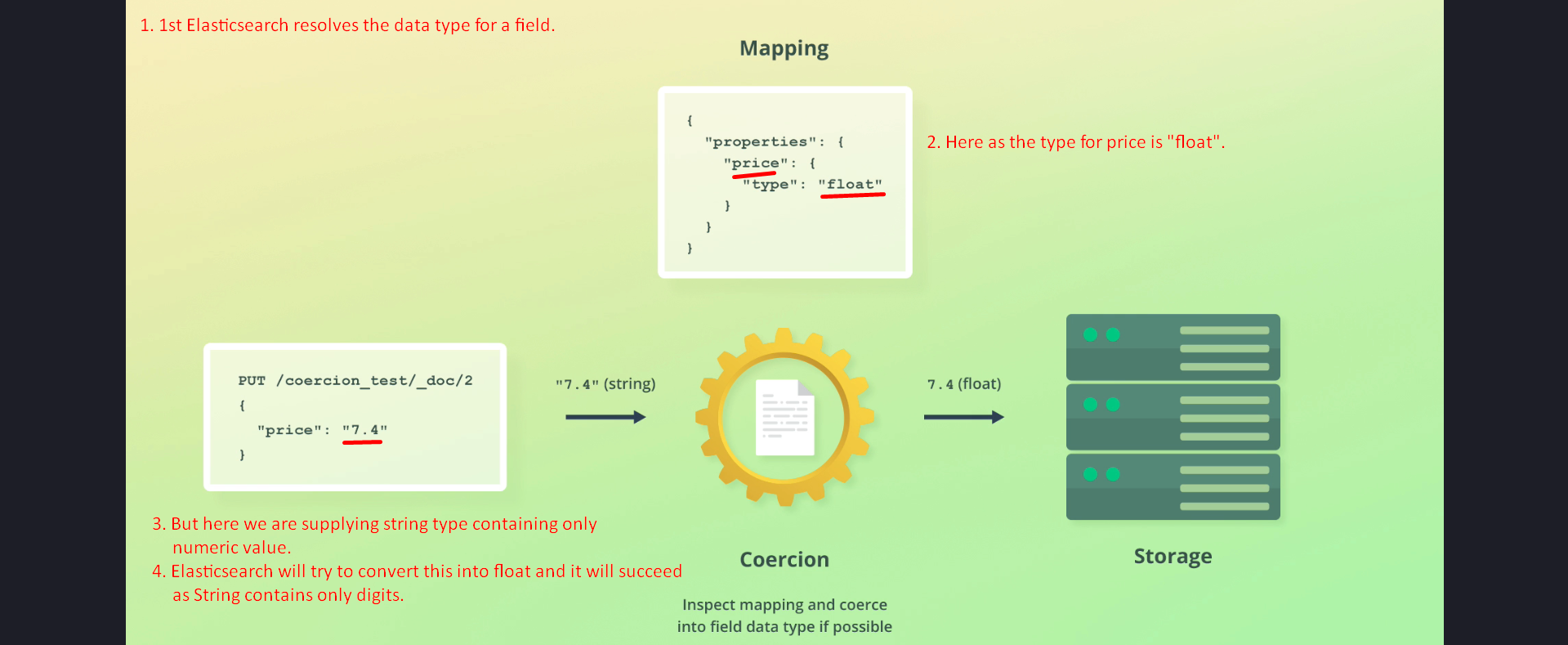
### 2.1 Example Setup

Let’s examine type coercion through a series of examples using Kibana. In these examples, we index three documents into a new throwaway index. The index will be created automatically upon indexing the first document.

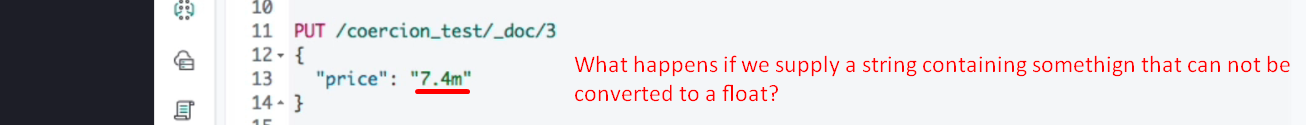
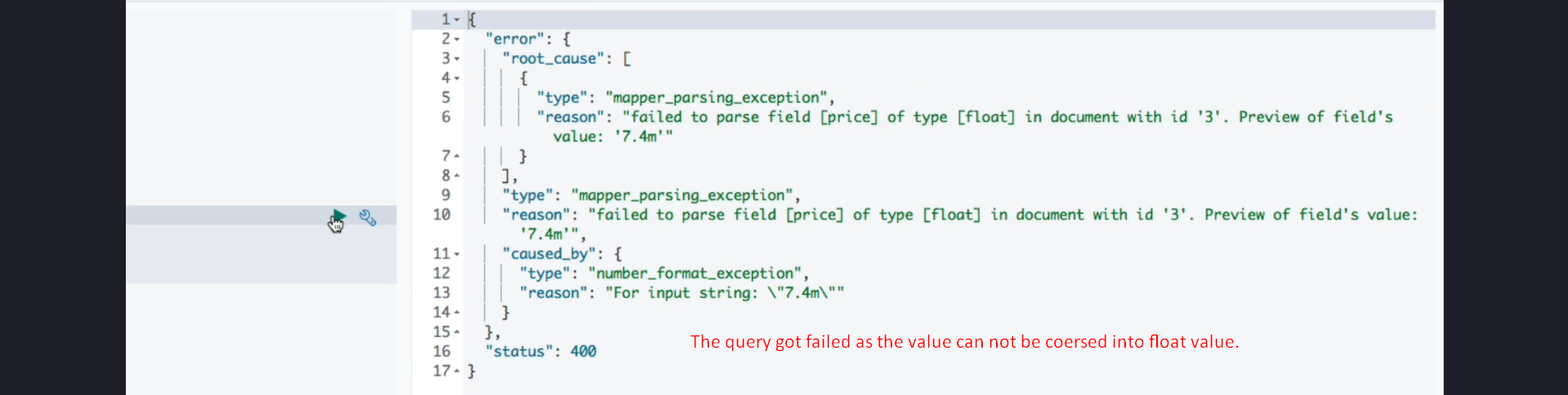
### 2.2 Case 1: Floating Point Number

In the first query, we supply a floating-point number for the 'price' field. Elasticsearch dynamically maps this field as a 'float' data type as we didn’t provide any mapping for index “coercion\_test”.  


### 2.3 Case 2: String Containing a Number

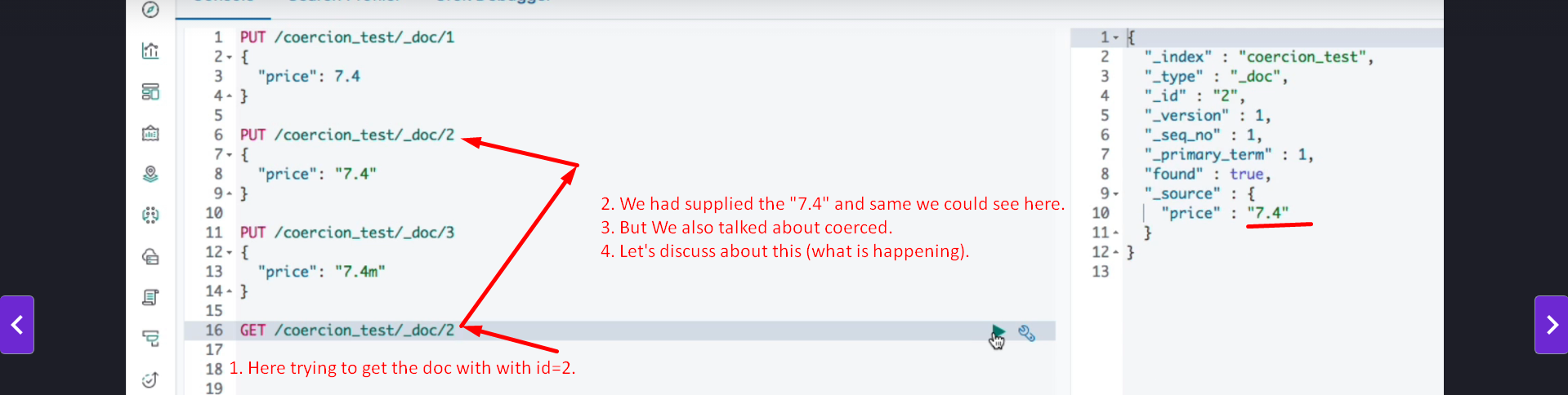
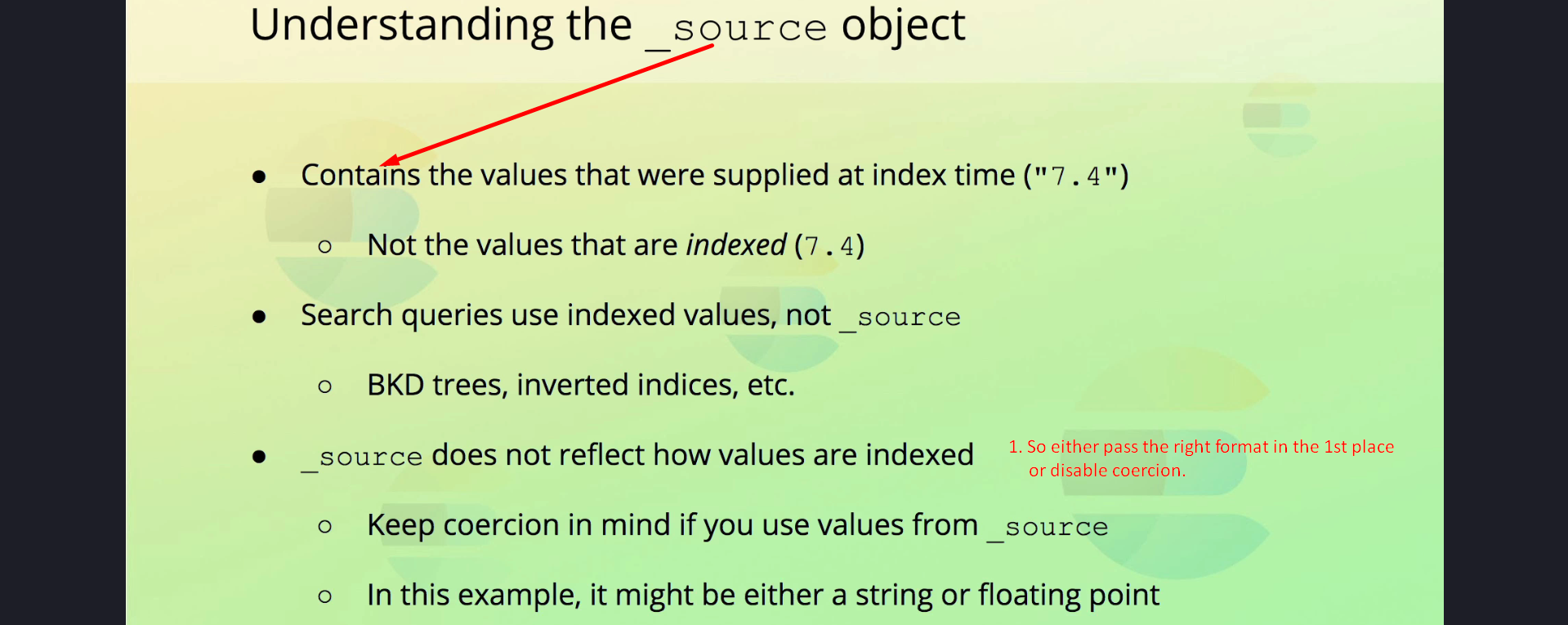
In the second query, we supply a string containing a numeric value (e.g., **"7.4"**). Elasticsearch **coerces** the string into a float and indexes the document successfully. This demonstrates how type coercion can handle mismatched but convertible data types.  
  
  


### 2.4 Case 3: Invalid String

In the third query, we supply a string that cannot be converted to a numeric value (e.g., "not\_a\_number"). Elasticsearch throws an error because the value cannot be coerced into the expected data type.  
 

## 3. Internal Handling of Coerced Values

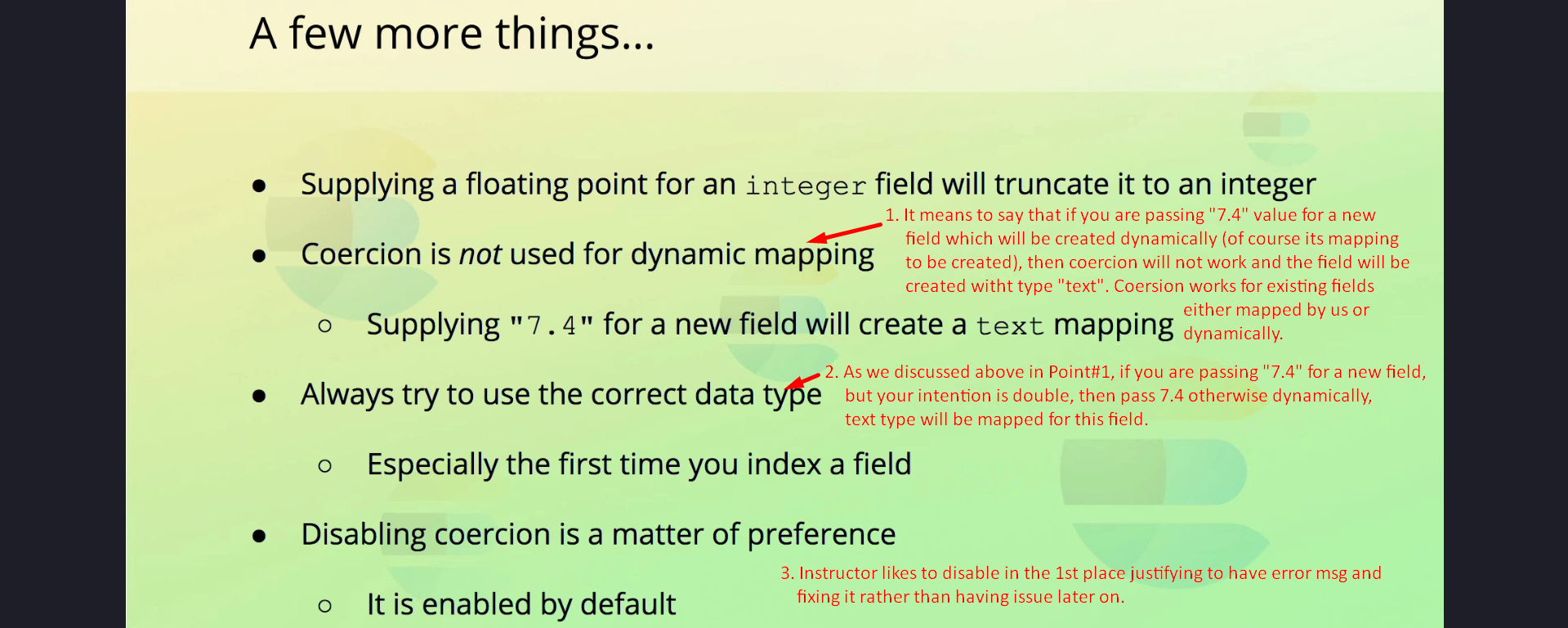
When a value is coerced, the original value is stored under the '\_source' key, while the converted value is stored in Elasticsearch's internal data structures, such as the inverted index or Lucene storage.

For instance, when a string '7.4' is coerced into a float, the '\_source' retains the string format, but Lucene stores it as a float for efficient searching and retrieval.  
  


## 4. Limitations and Best Practices

1. Coercion is not used to determine the data type for dynamic mappings. If a string is provided, the data type will be set to 'text', even if the string contains only numeric values.

2. It is essential to use the correct data types, particularly when indexing new fields with dynamic mapping.

3. While coercion is convenient, relying on it can lead to unexpected results. Consider disabling coercion for stricter validation, especially in production environments.  


## 5. Enabling or Disabling Coercion

Coercion is enabled by default to make Elasticsearch forgiving. However, you can disable it to enforce stricter data type requirements. Disabling coercion ensures that Elasticsearch rejects incorrect data types, allowing issues to be fixed proactively.

## 6. Cleanup and Conclusion

To conclude the demonstration, we delete the throwaway index that was created for these examples.

Key Takeaways:  
- Type coercion allows Elasticsearch to handle minor mismatches in data types gracefully.  
- Correct data types should always be used to avoid unexpected behavior.  
- Coercion can be disabled for stricter validation, which is often preferred in production environments.