# Scripted Updates in Elasticsearch

## 1. Introduction

Scripted updates in Elasticsearch allow you to write custom logic for updating document fields. This eliminates the need to retrieve, modify, and reindex documents manually. Scripts are powerful and can include conditional logic, parameters, and custom operations.

## 2. Basic Scripted Updates

1. Scripted updates are performed using the Update API.

2. The script is specified in the 'script' object within the request body.

3. Example: Decreasing the 'in\_stock' field value by 1.

POST /products/\_update/100  
{  
 "script": {  
 "source": "ctx.\_source.in\_stock -= 1"  
 }  
}

4. Explanation:

- 'ctx': Refers to the document context.

- '\_source': Provides access to the document's fields.

- The script subtracts 1 from the 'in\_stock' field.

5. Response Details:

- 'result': Indicates the outcome of the operation. Possible values include:  
 - 'updated': Document was successfully updated.  
 - 'noop': No changes were applied.

## 3. Using Parameters in Scripts

1. Parameters can be defined within the 'params' object in the request body.

2. Example: Reducing the 'in\_stock' field by a dynamic value (e.g., 4).

POST /products/\_update/100  
{  
 "script": {  
 "source": "ctx.\_source.in\_stock -= params.quantity",  
 "params": {  
 "quantity": 4  
 }  
 }  
}

3. Explanation:

- 'params': Defines key-value pairs for script parameters.

- 'quantity': The value by which 'in\_stock' is reduced.

## 4. Conditional Logic in Scripts

1. Scripts can include conditional logic using 'if' statements.

2. Example: Reducing 'in\_stock' only if its value is greater than zero.

POST /products/\_update/100  
{  
 "script": {  
 "source": "if (ctx.\_source.in\_stock > 0) { ctx.\_source.in\_stock -= 1 }"  
 }  
}

3. Explanation:

- The script checks the value of 'in\_stock' before applying the update.

4. Advanced: Setting the operation to 'noop' if a condition is not met.

POST /products/\_update/100  
{  
 "script": {  
 "source": "if (ctx.\_source.in\_stock == 0) { ctx.op = 'noop' } else { ctx.\_source.in\_stock -= 1 }"  
 }  
}

5. Explanation:

- 'ctx.op': Allows explicit control of the operation, such as 'noop' or 'delete'.

## 5. Deleting Documents with Scripts

1. Scripts can also delete documents by setting 'ctx.op' to 'delete'.

2. Example: Deleting a document if 'in\_stock' is less than or equal to 1.

POST /products/\_update/100  
{  
 "script": {  
 "source": "if (ctx.\_source.in\_stock <= 1) { ctx.op = 'delete' } else { ctx.\_source.in\_stock -= 1 }"  
 }  
}

3. Explanation:

- The script deletes the document if the condition is met.

## 6. Multiline Scripts

1. Complex scripts can span multiple lines by using triple double quotes.

2. Example: Multiline script to handle advanced logic.

POST /products/\_update/100  
{  
 "script": {  
 "source": """  
 if (ctx.\_source.in\_stock > 0) {  
 ctx.\_source.in\_stock -= 1  
 } else {  
 ctx.op = 'noop'  
 }  
 """  
 }  
}

## 7. Summary

1. Scripted updates in Elasticsearch enable advanced logic for updating documents, including conditional logic and parameterized operations.

2. The 'ctx' variable provides access to the document context, allowing you to modify fields or set operations.

3. Parameters make scripts dynamic and flexible, especially in application-driven queries.

4. Multiline scripts allow complex logic to be written more clearly and are ideal for advanced use cases.

5. Scripted updates abstract multiple operations into a single request, reducing network latency and improving efficiency.