Lightning Knowledge
Global Search and Replace

@salesforcedocs

Last updated: November 30, 2020

Table of Contents

1. Global Search and Replace

1.1. Disclaimer

While significant testing (both functional and volume) has been performed with 80 thousand articles without any issue, above that number the admin could experience issues with heap size limit. Customer expectations should be set accordingly. This being said, the author can provide limited support but would look forward to any constructive feedback (and bug reports) from all sources in an effort to increase the production readiness of the utility as well as build out any functionality that may have been overlooked.

This was the most significant factor driving the batch-oriented architecture of the Global Search and Replace utility.

For non-technical readers, Sections 1.7 through 1.10 can be skipped as they detail the physical design considerations and challenges of the Global Search and Replace utility.

1.2. Background

One of the most common requirements in most enterprise implementations of Salesforce Knowledge is the ability to perform a textual search across all articles and dynamically replace one character string with another. This operation, commonly called Global Search and Replace, is so common in other enterprise knowledge base solutions that most customers presume that Salesforce supports this feature. In fact, it is most often in the implementation phases of a project that the customer is disappointed to hear that Salesforce Knowledge lacks such a feature. This disappointment quickly leads to dissatisfaction and, often, buyer's remorse.

With the introduction of the read/write API in the Summer, 2012 release of Salesforce Knowledge, it became possible to think of engineering a custom enhancement to support a global search and replace utility. The purpose of this whitepaper is to document such a solution, with the intent that the search and replace utility can immediately add value to our current (and prospective) knowledge customer base. In addition, the intent is also to allow Salesforce Knowledge to be even more competitive in the marketplace, especially for those customers who are considering migrating off of a legacy (or competitive) knowledge solution.

All comments made regarding logical or physical design and implementation of the Global Search and Replace utility are valid and accurate as of Spring 2019 release of Lightning Knowledge.

1.3. Definitions

Prior to outlining the intended functional goals as well as the logical and physical design of a search and replace utility, it is important to start with a set of definitions that are used repeatedly in this whitepaper:

Search (or Containing Search) – A search is defined as the process of performing a physical lookup of a specific character string in the value of any arbitrary text field. A search can be thought of as a character string lookup in the field value without consideration for HTML markup. The definition of search as used in this whitepaper includes searching through HTML markup (if any) that may be present in the field value. The term search as used within applies to both indexed and non-indexed content (this is key).

Character String – A text value which is either a literal (e.g., 'Hello World') or RegEx (Regular Expression) pattern (e.g., '.?(SSN=[0-9]-[0-9]-[0-9]).*').

Pattern – A valid RegEx search expression (Note 1).

Indexed Content – The value of a text field, exclusive of markup, which is indexed by the Salesforce Knowledge search engine and is therefore searchable via the Articles or Article Management tabs on the platform. HTML markup is not indexed and, therefore, by default not searchable via the Articles or Article Management tabs.

Text Field (or Text Custom Field) – A text field is defined as one of the following:

- Rich Text Fields (or RTA, Rich Text Area) – A custom field defined in a knowledge base record type that is typically used to manage HTML content. Rich text fields by default have a maximum size of 32 KB but can be defined as large as 128 KB.

- Text Fields – A custom field defined in a knowledge base record type of one of the following physical types:

  - Text Field (maximum size of 255 characters)

  - Text Area (maximum size of 128 KB characters)

Replace (or Replacement) – The process of dynamically substituting one character string value with another in any arbitrary text field value. Replacement refers to the permanent substitution of two character strings, a search string and a replacement string, where the search string is treated either as a literal or a pattern. Upon replacement, an article is either modified in place (as is the case for Draft articles) or automatically republished (as is the case for Published articles).

Text Field Value – The actual value assigned to any text custom field. For all but rich text fields, the value defined in the field is presented to the user 'as is'. For rich text fields, the markup (if any) present in the field defines the formatting of the content when presented to the user. The term text field value as used in this whitepaper refers to the content as well as markup (if any) defined in any text field.

Markup (or HTML Markup) – The group of special formatting commands in the value of a rich text field that serve the purpose of defining the presentation (formatting) of the content when presented to the user. Markup is most commonly thought of as HTML tags embedded in the content. Simple examples are <p> and <img> which define Paragraphs and Images, respectively.

1.4. Platform Functional Limitations

The following functional limitations on the platform were first considered prior to the engineering of a custom global search and replace utility.

1.4.1. Salesforce Reports (Reports Tab) – Reporting Limitations

A report containing custom field values may be created for any Salesforce Knowledge record type by first creating a custom report type in the administrative setup area of the platform. This custom report type defines both the metadata and custom field(s) that will be available for reporting when a report based on this custom report type is created by the user (Notes 2, 3). Reporting was originally thought to be a stronger solution for 'searching' knowledge custom text field values than the search engine itself since it has the ability to search markup via the 'contains' function defined in a report filter. However:

- A reporting filter defined on a custom text field value using the 'contains' function only supports the first 253 bytes of a text field value. This platform limitation, which has existed for some time (Note 4), makes it all but impossible to think about using the 'contains' filter as a tool for searching knowledge custom field values. This is especially true since the basic container for knowledge article content is the rich text

field, which is typically defined with a maximum size measured in multiple thousands of characters.

- Reporting is fully supported only on record types that contain less than 100K articles. More specifically, reports cannot be created on any Salesforce object that would return more than 100K records. While this is not going to be a limitation for most customers, some enterprise implementations currently exceed this limit and therefore a report cannot be used for 'searching' text field values unless additional constraints (report filters) are defined that would ensure the result set is below 100K articles.

1.4.2. Salesforce Knowledge – Search Engine Limitations

When article custom field values are saved, only the content itself is physically indexed in the search engine. While this may seem obvious, there is an implication for values stored in rich text fields in that HTML markup is specifically excluded.

This limitation is for good reason as it strengthens the end user experience when performing keyword searches on articles. However, it makes it all but impossible for a knowledge base administrator to search for articles that contain specific markup tags. For example, a common requirement is to determine how many articles contain a link (i.e., hyperlink or HTML anchor tag) to a specific URL, as in the following example:

- HTML markup in one or more articles:
  <a href="https://www.mydomain.com/myapp/mypage" target="_blank">

- Desired search string:
  www.mydomain.com/myapp/mypage

Such searches are not possible using the native search engine which supports Salesforce Knowledge. Markup searches can include the actual markup tags or, as in the above example, selected substrings (parameter values) within the markup tags themselves. This is not to be confused with searching the label associated with a markup tag (if applicable, such as in the anchor tag example above). Labels represent content visible to the user since they appear outside the physical markup tags and are, therefore, indexed and searchable.

1.4.3. Lightning Tab

My domain should be enabled in order to see the Search and Replace (Lightning Tab)

1.5. Design Goals

The Global Search and Replace Utility was engineered to support the following design goals:

- Provide the ability to perform a character string search on custom text field values. Provide support for searching on both content as well as markup.
- Support search filters such as Publish Status and Date Created (these complement the character string search) and channels filter “from” and “to” (by default the selected channel is internal).
- Support a string replacement operation on all articles returned in a search, substituting all values of the search string with a specific replacement string.
- Provide the ability to target searches to specific text fields (metadata or custom) in a record type.
- Maintain an activity log of all search and replace operations.
- Perform all processing in the background (i.e., batch), and allow the user to determine the current status of the background process. See Design and Constraints and Challenges section, below.

Note that the following functionality was specifically excluded from this version of Global Search and Replace:

- Provision to limit the Search and/or Replace operation to a specific number of records
- Support for field types other than text fields

1.6. Local Design (Design Theory)

The design of a Global Search and Replace utility needed to consider several Salesforce Knowledge architectural requirements. As these were thought through, other platform - based requirements became evident that also needed to be factored into the logical design.

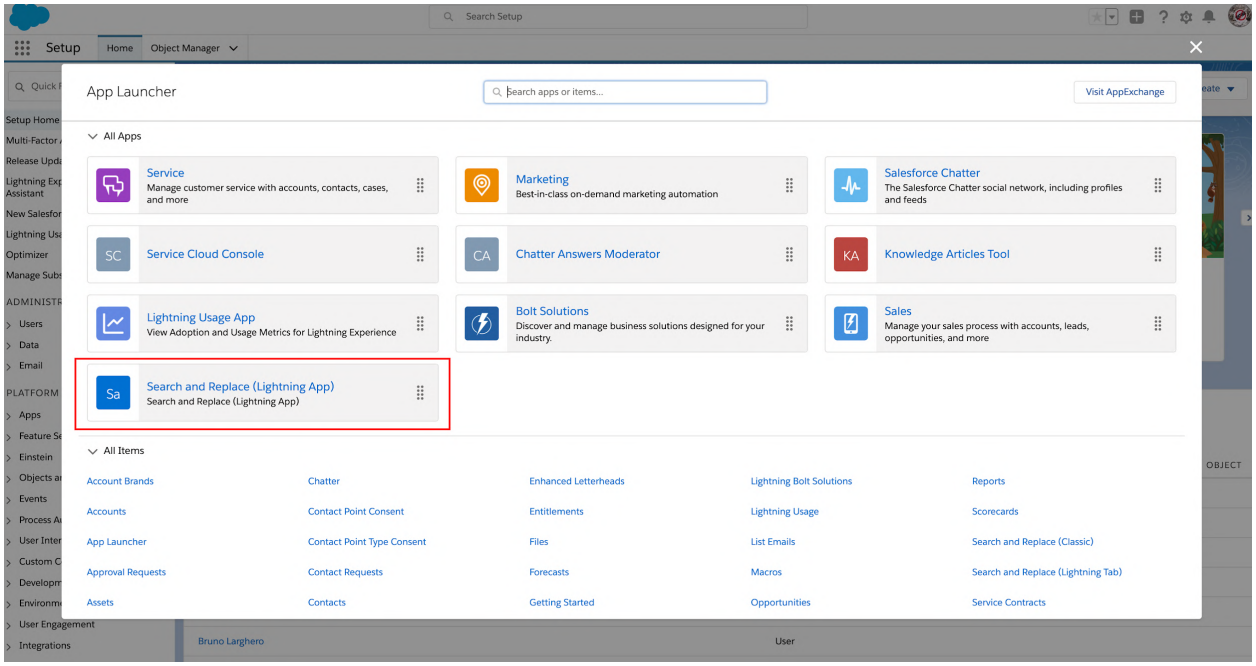
The logical design is separated into three main components:

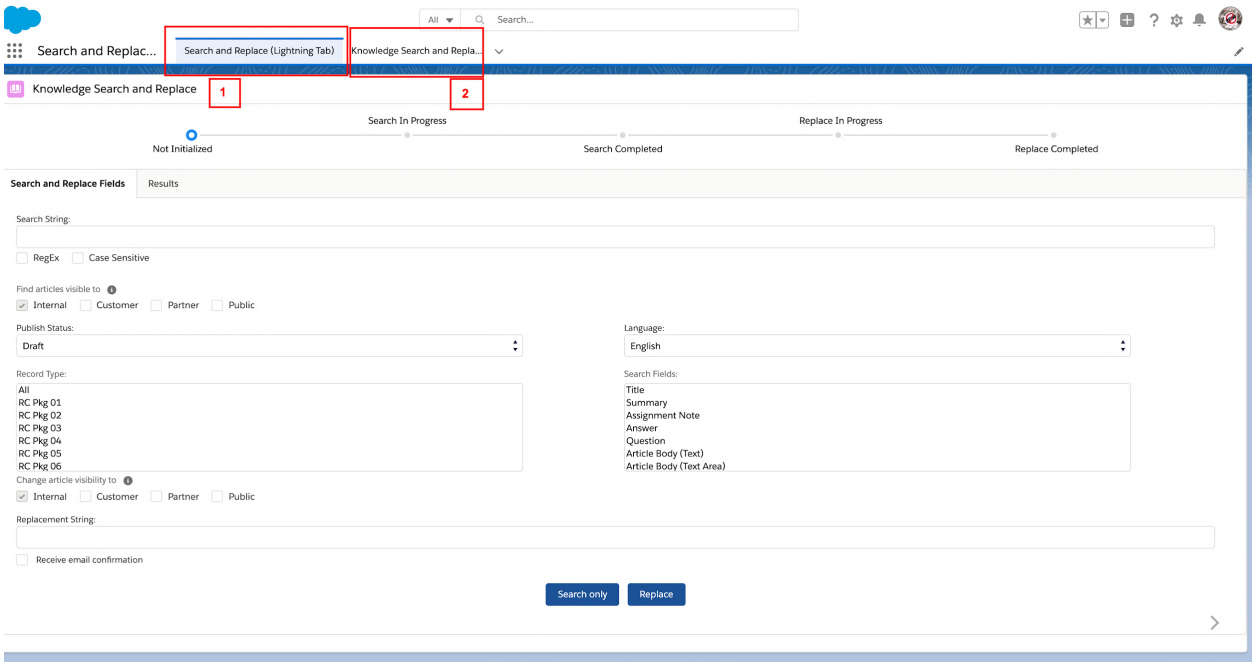
- End User Interface
- Search Process

- Replacement Process

1.6.1. End User Interface (UI)

The package contains a lightning app called ‘Search and Replace (Lightning app)’ where you can find a tab with the tool for replacement and also the records created in each execution with details of the operations.

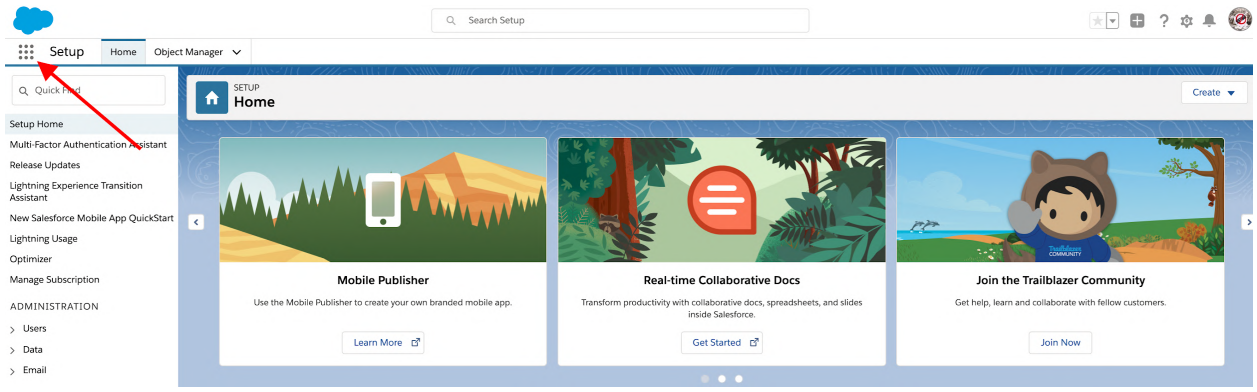


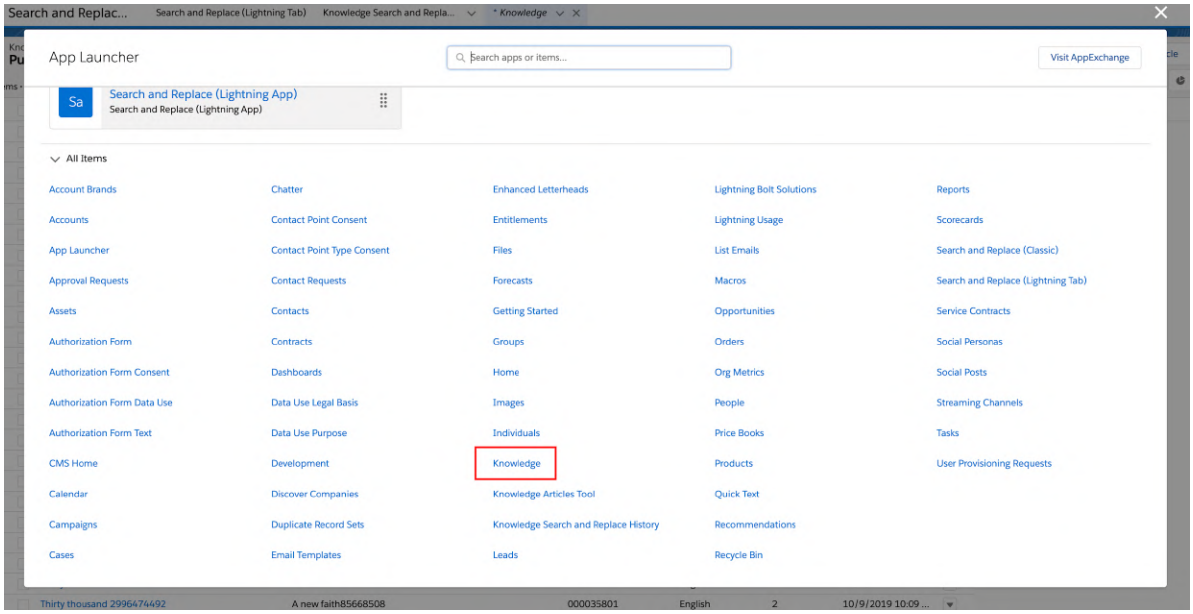


Search and replace (Lightning Tab) (1) contains the tool for searching and replacing text. The *Knowledge Search and Replace History* tab (2) displays the records created in each execution.

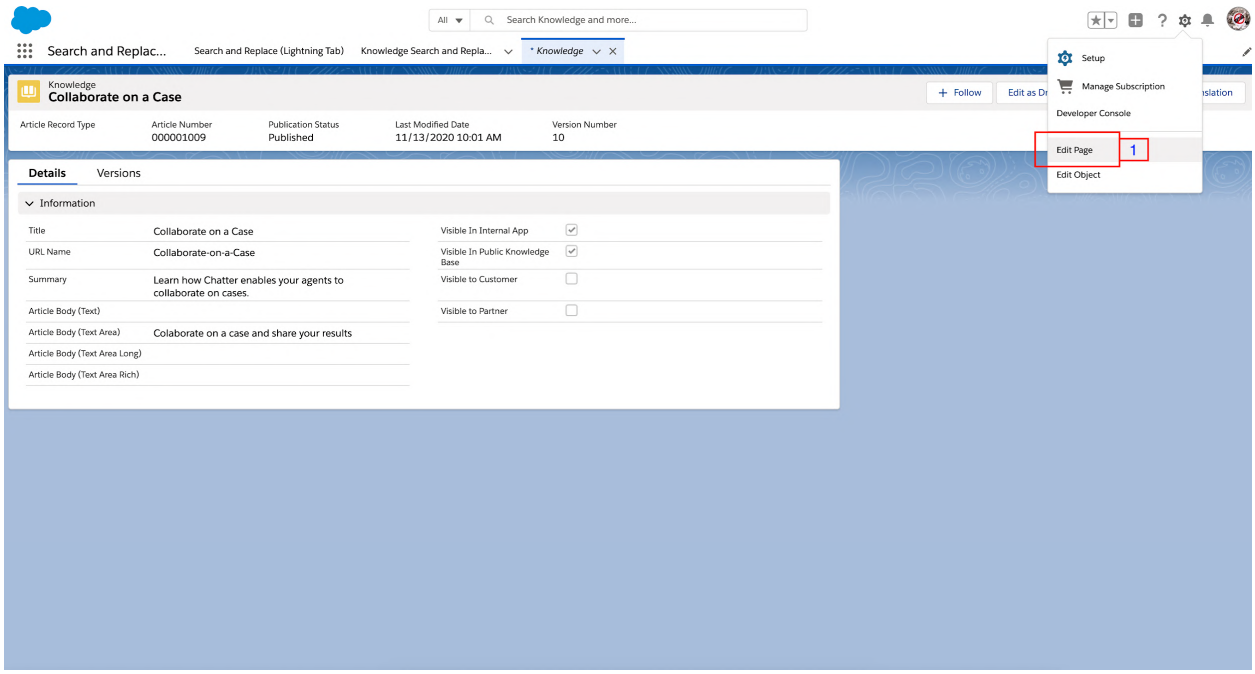
Users can use this tool or they can also use a Lightning component that can be placed in the Knowledge Record Home page via Lightning App Builder. To use the component, please follow these steps:

1. Go to Knowledge by selecting the option in the App Launcher

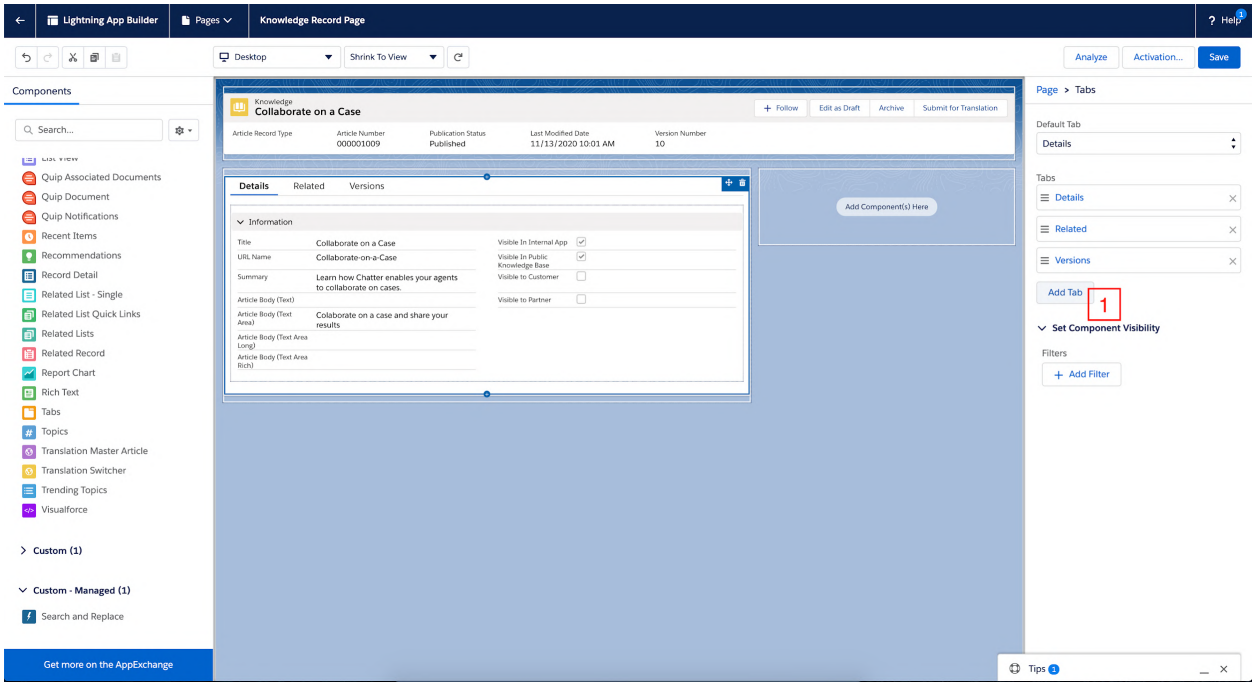




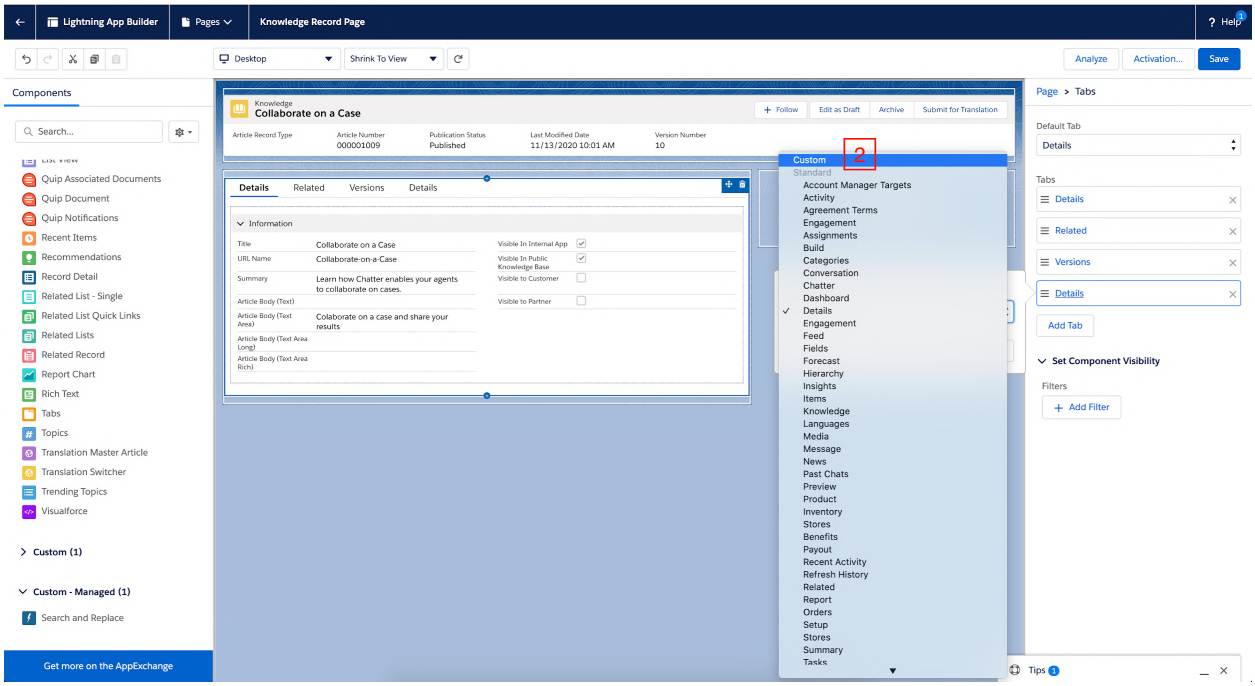
2. Open any Knowledge article and click on the *Edit Page* (1) option located in the Setup wheel.

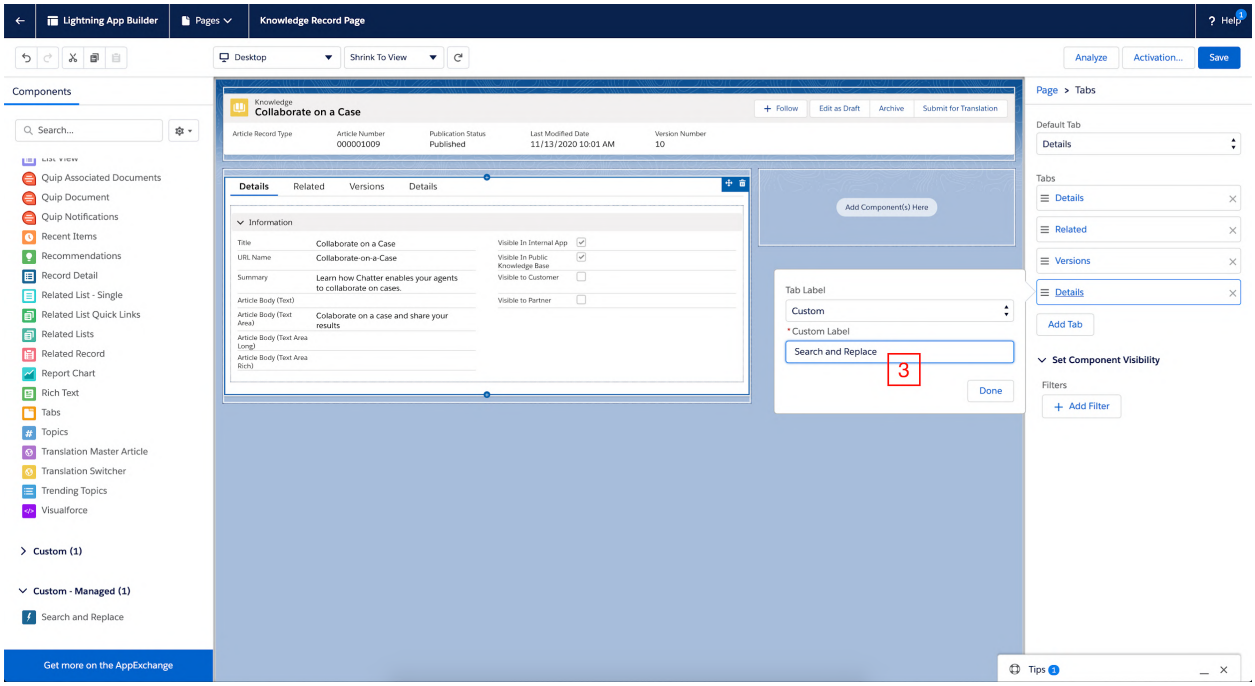


3. This will open the Lightning App Builder, where you can add the Search and Replace component and use it directly in the Knowledge Record Page. To do this, add a tab to the main section of the page by clicking the *Add Tab* button (1)



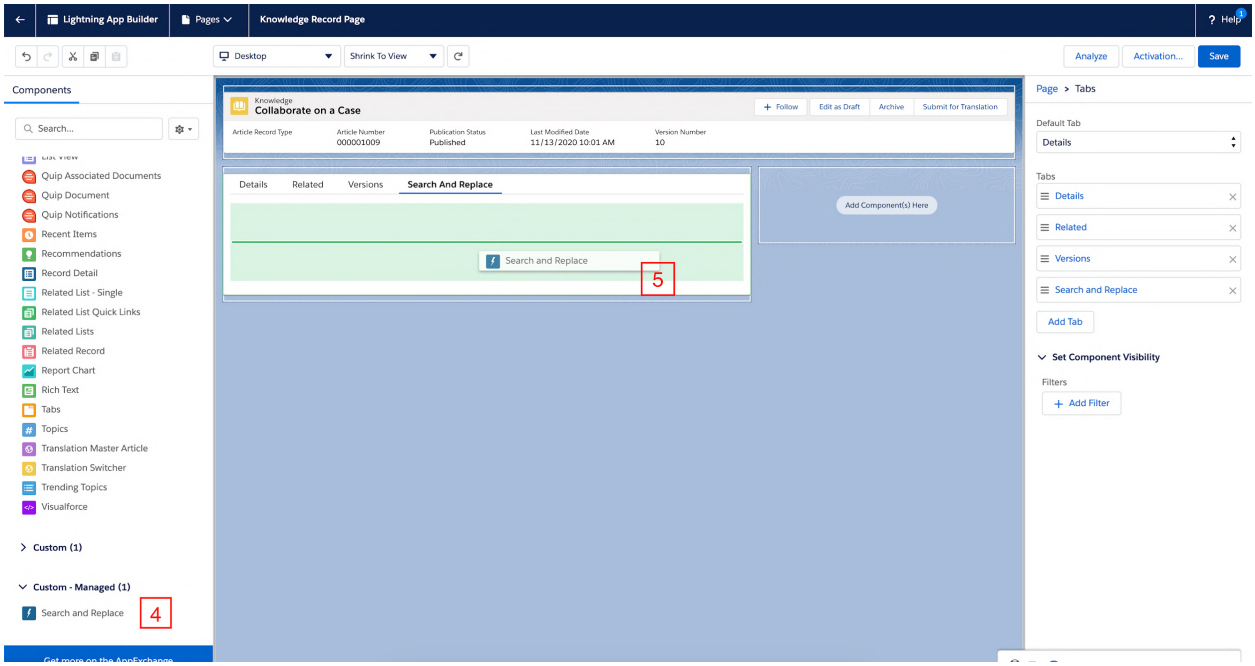
Select the *Custom* (2) option from the drop-down menu and add the custom label (3)



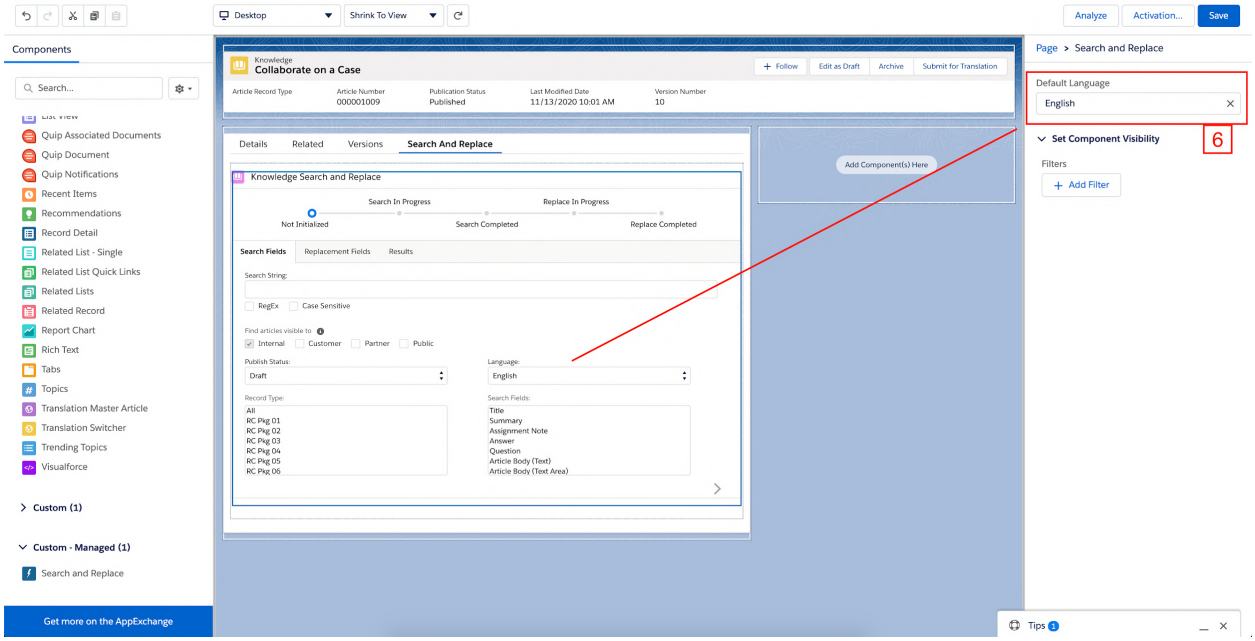


Click *Done* to save the tab's name.

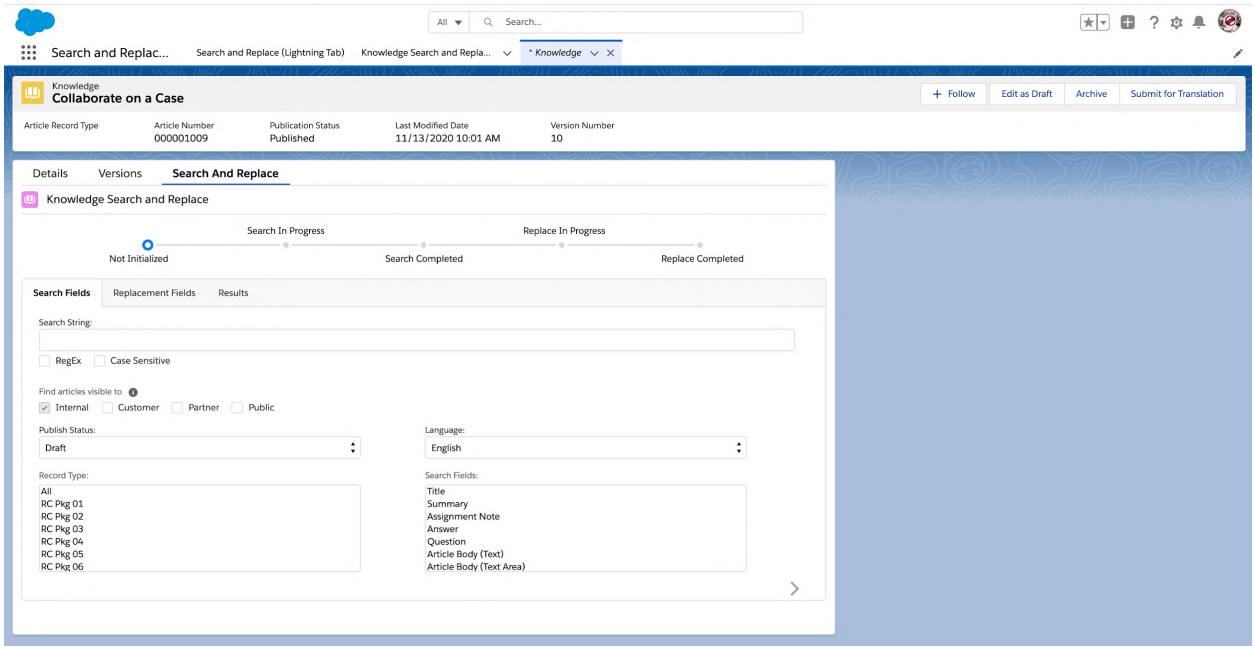
Finally, drag and drop the Search and Replace component from the Custom-Managed Components section (4) to the new tab (5).



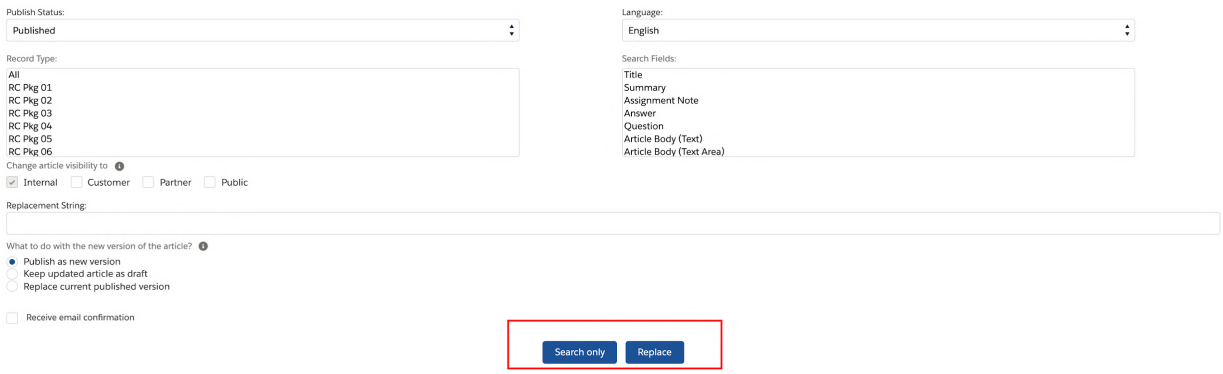
Once the component is added, you can select the default language (6) that will be displayed every time the component is opened in the Knowledge Record Page.



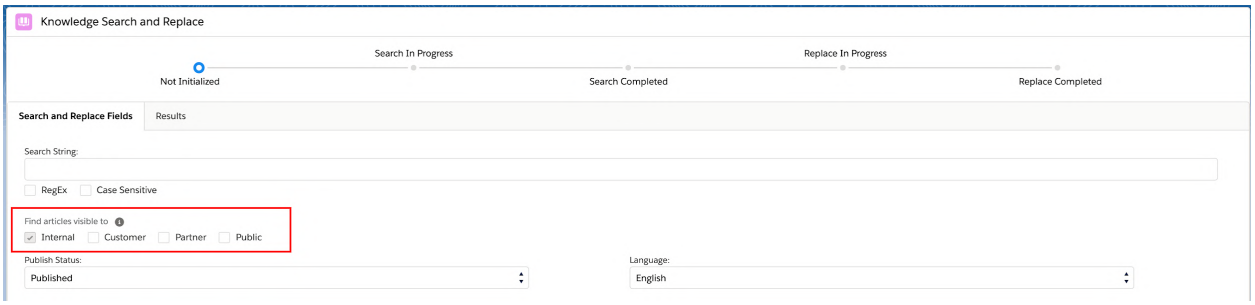
4. Click the Save button to apply the changes. After the changes are saved, the component will be available from the Knowledge Record Page.



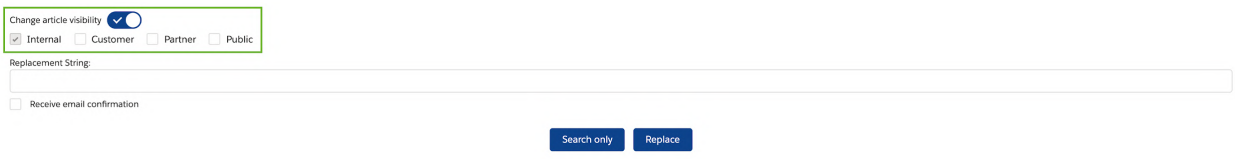
Search and replace operations are performed on a specific record type/publication status pairing. The search and replace fields in the UI support long text. The UI was designed to support both search only and full search and replace functionality. ‘Search Only’ mode returns both the number of articles as well as metadata (article number plus title) for articles which satisfy the search criteria (see next paragraph) . This was designed to allow a user to pre-determine the impact of executing a full search and replace operation across all articles of the selected type and publish status. There are two buttons on the UI in order to be able to perform a search only or a replacement.



The user can select the channel where the search will be performed. ‘Internal’ is always selected by default.



<u>Important Note:</u> This filter works as an ‘AND’ operation. The articles returned in the search results will have every channel that you selected in this search.
Additionally a switch has been added in version 1.7 to give the user the option to choose if the article visibility wants to be changed to another channel(s) after the replacement.



The UI is designed to accept some simple filtering criteria for the Search process:

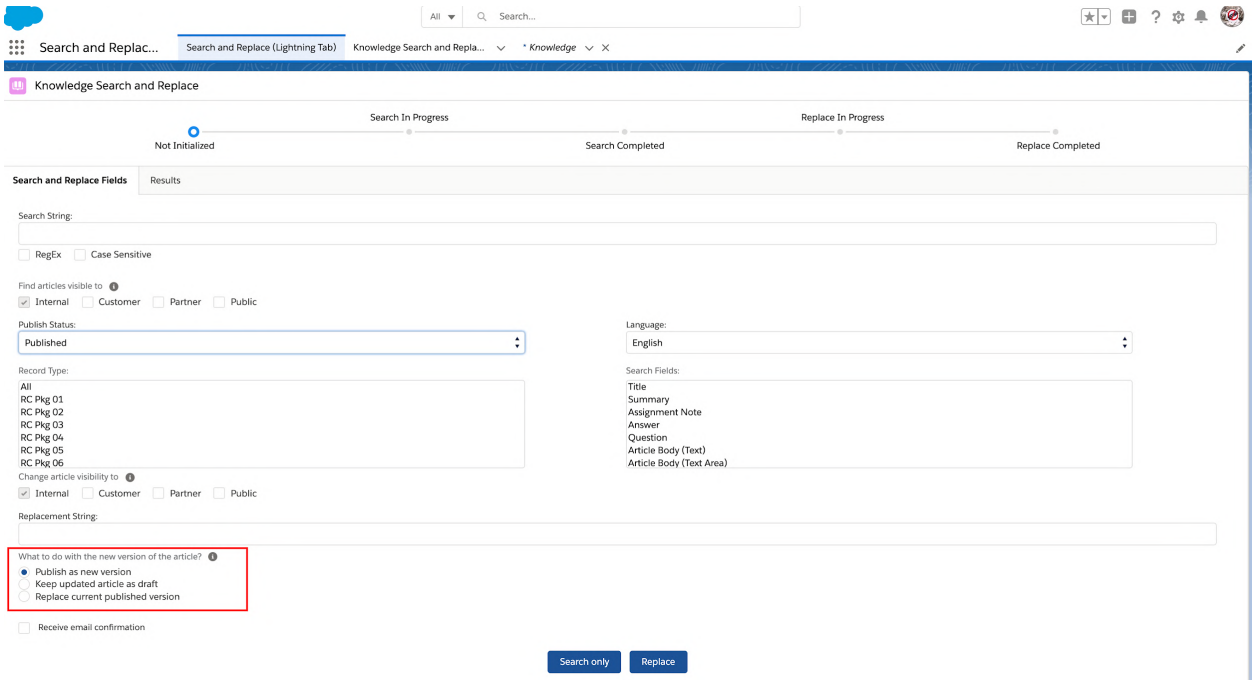
- Record Type (Note 5)
- Publish Status (draft or published)
- Create, modify, publish dates (Note 6)
- Language (Note 13)

Also, you can choose to send an email after the job is done (not selected by default).

The user is required to select one or more text fields in which the search and replace operation will be performed. The list of search fields are all of type String or Text – text, text area, long text area, rich text area. The list is not necessarily inclusive of all fields, but all fields in the list must be of type text in order to be searched (Note 7).

The Publish status for the articles search is set as to 'Draft' by default. If the status is changed to 'Published', a radio group will appear near the bottom that will let the user pick one of the following options:

- "Publish as new version" will automatically publish the article as a new version with the replacements implemented. The previous version will be saved as an older version.
- "Keep updated article as draft" will create drafts with the changes made by the tool but won't automatically publish the article. This means that for each published article that was intended to be modified there will be a draft with the updates. Users will have to manually publish them.
- "Replace current published version" will replace the current version of the article. The previous state of the article will not be visible any more because a new version was not created. You will not be able to see or go to the article as it was before replacements were made.



Two buttons are available on the bottom of the search and replace landing page that control the search and replace operation:

1. The ‘Search only’ button creates and submits a search batch job to the platform. Searches are always performed asynchronously; batch jobs are able to assign more system resources and process a relatively unlimited number of articles.

2. The ‘Replace’ button allows the user to submit a replace batch job. Articles are searched first, even if a Search was performed earlier, before starting the replacement job if any matches were found.

The figure below presents a view of the simple user interface provided in this version of Global Search and Replace:

Figure 1.1 – Search and Replace Utility – Search Fields Data Interface for Draft articles (Tab component)

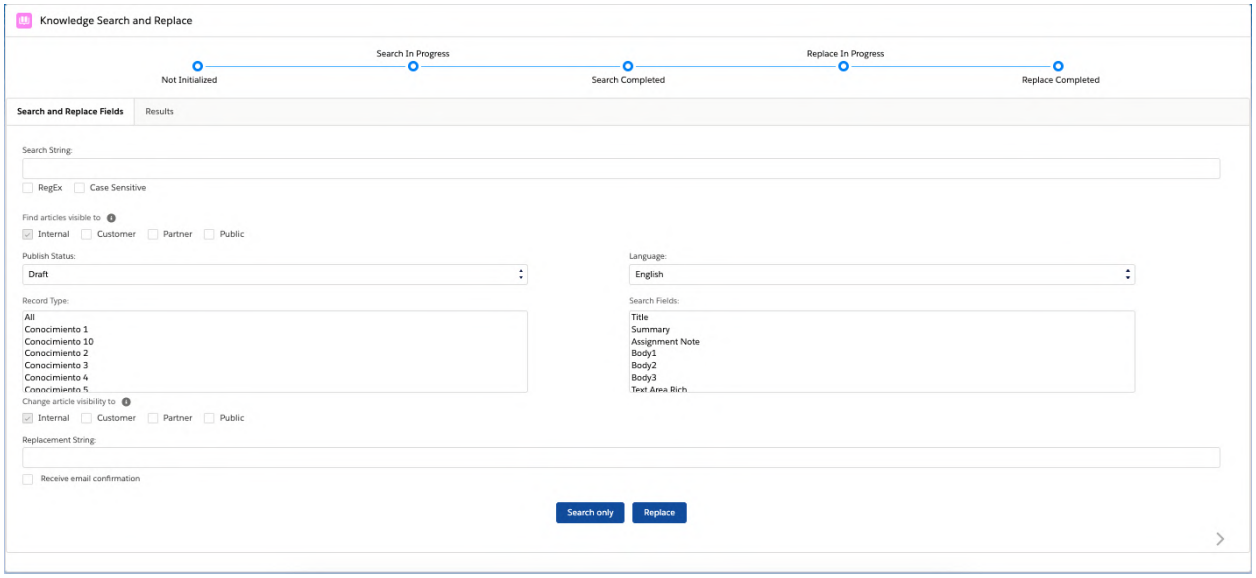


Figure 1.2 – Search and Replace Utility – Search Fields Data Interface for Published articles (Tab component)

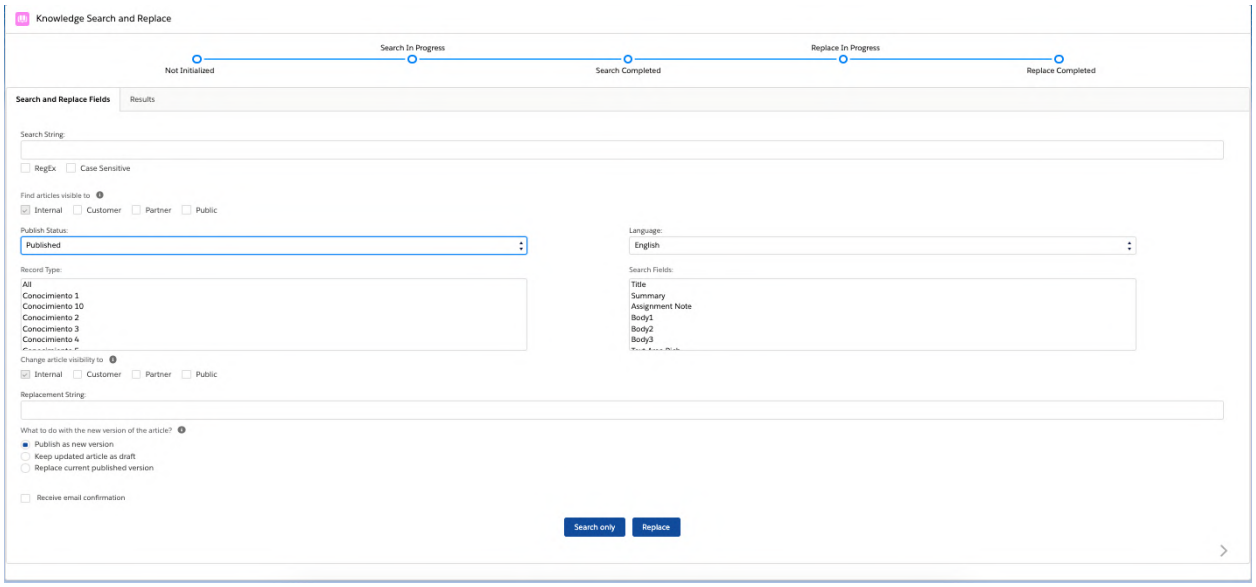
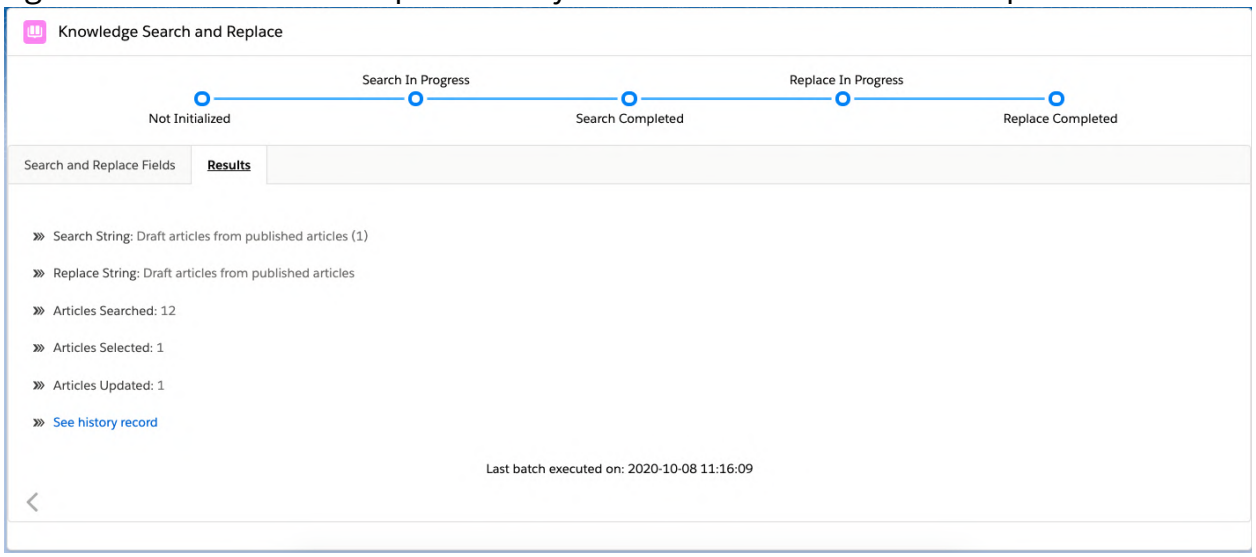


Figure 1.3 – Search and Replace Utility – Results Interface (Tab component)



1.6.2. Search Functionality

When the search and replace utility was initialized conceptualized, the process of automating the search for an arbitrary character string was presumed not to require any custom functionality since this could be easily accomplished via the features available in both SOSL (Salesforce Object Search Language) or SOQL (Salesforce Object Query Language). This presumption turns out to be completely inaccurate due to the following limitations of SOSL and SOQL:

- SOSL – While a SOSL query does support searching for a character string across all fields and all record types (somewhat obvious since SOSL supports the search function available on the native Articles tab end user interface), it does not offer full support for searching the entire value of a rich text field. This is due to the fact that HTML markup is not indexed. Since SOSL queries leverage the internal search engine for character string searches, they lack the ability to search non-indexed content and therefore do not fully support one of the main requirements of this utility (the ability to search both content and markup). They also do not support character string (containing) searches on long text fields, another main requirement.

- SOQL – SOQL queries do offer support for searching the native value of a rich text field (both content and markup), however, they provide this support through the use of the LIKE operator used in the WHERE clause (e.g., WHERE field LIKE '%string%'). This poses a number of challenges, namely,

    1. The LIKE operator is subject to the same limitation as in reporting, where only the first 253 bytes of the field value of searchable
    2. The LIKE operator can only work on one field at a time, resulting in a very complex query when multiple fields need to be searched
    3. SOQL queries have strict governor limits per transaction, among them that the object being searched can have no more than 50K rows. This limit in and of itself is enough to prevent SOQL from being used in this solution (Note 8).

Therefore, a custom solution was required to support character string search across the entire value of any type of text field, including rich text fields, for record types of any size. This solution was to leverage an Apex-based process executed in batch mode (avoids object size limitations), with the process itself designed to programmatically inspect the value of text fields on a record-by-record, field- by- field basis using String class methods (supports markup as well as text values of any length). A secondary reason to consider a batch type of architecture was to allow the process to consume more virtual processing time than is available in a single OLTP transaction (roughly two seconds; evaluation record-by-record and field-by-field is somewhat CPU intensive).

An Apex programmatic-based process does provide a number of distinct advantages over-and-above that of native SOQL and SOSL which were not immediately recognizable until the physical design was completed. Among these are the following:

- The ability to target Searches to a specific field or set of fields (SOSL does not support searches of named long text area fields and a similar construct in SOQL would be relatively complex)

- The ability to leverage Regular Expressions (RegEx) for character string searching. This is an extremely powerful benefit which provides support for very complex pattern matching, although the syntax is a bit arcane (Note 1).

- The ability to programmatically derive a quantitative search relevancy 'weight' by counting the number of times a search expression appears in a field or set of fields (this feature was not exploited in this utility but is mentioned here because of its importance)

The search string field is actually a character string that represents any form of valid Regular Expression (i.e., RegEx). Most users will be unaware of this and will be searching on simple character values. Advanced users, knowledgeable that this field supports RegEx, can perform more advanced searches on article content and/or markup. Conversely, in some limited cases, this may cause confusion as RegEx has a few 'special' characters which must be escaped in order to be treated as literals (Note 9)

1.6.3. Replace Functionality

Articles identified via the Search functionality are defined as 'candidates' for replacement. This implies that at least one occurrence of the specified search string exists in at least one selected field of the candidate article. The objective of this utility is to permanently change the value(s) of the search string field with the specified replacement string value and a field-by-field basis.

The term 'Replacement' requires some additional discussion. Published Salesforce Knowledge articles are not simply 'updated', they need to be first modified in Draft form and then republished. In addition, Drafts represent articles which are already in an active state of modification but have not yet been published. This distinction is important and is discussed in much more detail in the technical design section of this document.

The Search process, when complete, results in the creation of a 'task list' of knowledge articles which meet the criteria specified in the Search UI. The derivation of a list is intentional, as it permits the search and replacement operations to work as decoupled physical tasks. This permits the batch process(es) to be much more scalable and, in fact,

was a necessary physical design consideration in order to be able to process large numbers of article updates.

The replacement process is responsible for processing the article candidate task list in a serial order and assumes no dependency exists between articles nor any dependency in the order of operations within an article.

Drafts will remain drafts after the replacement operation if the user has performed a Search on draft status articles (i.e., the Draft will be updated in place).

Published articles are slightly more complex, in that a Draft may already exist for the Published article.

For example, a user could have manually edited a published article using standard platform functionality but not yet published the new draft created in that process.

These conditions can be summarized as follows:

Search Mode	Publication Status	Draft Exists	Replacement Operation(s)	Comments
Draft	Draft	Yes	Update Draft	Articles remains in Draft status
Published	Published	No	Create Draft Update Draft Publish Draft	A Draft must be created and then (re-)published in order for a published article to be updated. These tasks are completely automated by the replacement process.
Published	Published	Yes	Update Draft	Articles remains in Draft status

Consider the case of a search and replacement operation on a Published article for which a Draft already exists. This can happen, for example, when another user is currently editing the same article. The search and replacement job can perform the update and publish operation on the existing Draft, but this impacts this other user in as much as they are never informed that the Draft they are editing has just been modified and has changed status (Note 10). Any changes that the user was in the process of making, or has made but which were just of Draft quality, are effectively lost and/or moved online, which is usually not the desired action.

The default operation for replacements requested on articles for which a Draft version already exists is to perform the replacement operation on the Draft version of the article, and to leave the article in Draft status. This protects the user requesting the initial edit but is somewhat transparent to the search and replace user. For this reason it is important to examine the log generated at the end of the replacement process to determine if any Draft versions were encountered when attempting to update published articles.

1.7. Physical Design

The Salesforce Knowledge Search and Replace utility leverages three separate Apex batch classes which perform the following tasks:

1.7.1. BatchKnowledgeGlobalSearch

This process identifies articles which have at least one text field value matching the specified search criteria. This class is submitted as an Apex batch job when the user selects the Search button in the UI. An unlimited number of knowledge search processes can be submitted at any one point-in-time, by multiple users (Note 11). It should be noted that replacements on the same article attempted across multiple batch jobs are resolved in the same manner as the standard platform application; the first job to update the article 'wins' (Note 12).

Of the search and replacement processes, the search process is the more complex of the two. The physical design performs the following sets of operations (more detail on the physical design can be found in Section 1.9, Design Constraints and Challenges):

1.7.1.1 Phase One – Start Method – Application of High-Level Search Constraints

The start method of the search process issues a simple SOQL query against the selected record type. The constraints applied in this first query are the selected record type and publication status, together with the default language selected from the tool required by SOQL (Note 13).

The fields returned by this initial query are simply the article number together with the record Id and KnowledgeArticleId. This technique actually represents a Salesforce best practice; rather than retrieving the custom field values for records selected in the start method, only the Ids are returned. The actual field values will be retrieved from the record type object vis-à-vis the record Id in the execute method of the class. This overcomes several governor constraints and permits a relatively unlimited number of records to be processed in one Apex batch job.

1.7.1.2 Phase One – Execute Method – Physical Design

Records are processed off of the QueryLocator object in batches of 30 at a time. This batch size is actually defined up front by the caller of the batch job (the Apex controller of the UI, KB Global Search and Replace). For each article, a separate method is called to perform the actual search for the search string in each of these custom field values using Pattern/Matcher logic (which itself treats the search string as a RegEx expression).

If a match is found, the appropriate metadata (Article Id, Number, record type, Publish Status, Field Names, and Search/Replacement string values) is written to a record in a second custom object named KB Global Search and Replace. Each row in this object is considered a replacement candidate, is identified uniquely by the batch Id it was created in, and will be processed in a second batch job (BatchKnowledgeGlobalReplacement).

1.7.1.3 Phase One – Finish Method – Physical Design

If all source articles have now been searched, the BatchKnowledgeGlobalReplace job is called to perform the physical string replacement operation on article replacement candidate(s). This step is bypassed for Search Only requests. Next, a record is inserted into the KB Global Search History custom object which serves as the permanent log for this request. Finally, an email is generated to the user which provides a complete list of all articles selected in the search (i.e., all replacement candidates). This list is arbitrarily limited to a physical size of 2K bytes.

1.7.2. BatchKnowledgeGlobalReplace

This process performs the physical string replacement on each article identified as a replacement candidate. This physical design represents an intentional decoupling of the search and replacement processes, allowing for greater flexibility in design, the ability to process record types containing arbitrarily large numbers of records, and to support smaller size batches (scopes) required for knowledge article updates (Note 14). This process is called directly from the BatchKnowledgeGlobalSearch process as an Apex batch request.

1.7.2.1 Phase Two – Start Method – Physical Design

The start method for the replacement process is far simpler than that of the search process. A QueryLocator object is instantiated for all records that exist in the KB Global Search and Replace object that were inserted during the previous search job(s). The actual Apex batch Ids for the search job(s) are tracked by the BatchKnowledgeGlobalSearch process and passed to the replacement process by the caller. This is the main technique used to identify

a unique search and replacement request (which itself consists of at least one (and possibly more) search jobs and one replacement job).

1.7.2.2 Phase Two – Execute Method – Physical Design

The execute method is responsible for performing the physical string replacement in each candidate article. It performs the following sequence of operations:

- Determines if the article is currently in Draft or Published status. If the status is Published, then a new Draft is automatically created (if required, as the Draft may already exist).

- Retrieves the new (or existing) Draft article (thereby obtaining all selected custom field value(s) from the article) via a simple SOQL query

- For each custom field selected, update the field value(s) in the local copy of the Draft article by using Pattern/Matcher objects. Note that all occurrences of the search string are replaced in each custom field value(s)

- Publish the updated Draft as a new version (if the current status is Published)

Note that two SOQL queries are required here for each (published) articles – one to determine if a Draft for this article already exists and the second to retrieve the custom field values of the new or existing Draft. For this reason, the batch size of the BatchKnowledgeGlobalReplace execute method is much smaller than that of BatchKnowledgeGlobalSearch (50 and 500, respectively). Recall that Salesforce Knowledge API is not bulk-enabled; each update must be performed separately and therefore governor limits for max number of SOQL queries and DML statements were gating factors here.

1.7.2.1 Phase Two – Finish Method – Physical Design

The finish method of the replacement batch class is responsible for performing some important housekeeping. The record inserted into the KB Global Search History object by the search process is updated to reflect the actual number of articles that have been updated in this job.

Finally, if the user chose to, an email message is generated and sent to the original user that confirms that the replacement process has completed its execution and provides a link to the search history record with the information of the last batch and also the related tab with the csv containing the changes applied to the articles.

1.7.3. BatchDelAuxSearchAndReplaceObj

This process removes records processed from the KB Global Search and Replace object as these are no longer required.

1.7.3.1 Phase Three – Start Method – Physical Design

The start method for the removal process is far simpler than the one of the search or replace process. A QueryLocator object is instantiated for all records that exist in the KB Global Search and Replace object that were inserted during the previous search job(s).

1.7.3.2 Phase Three – Execute Method – Physical Design

The execute method is responsible for performing the records of the object removal.

1.7.3.1 Phase Three – Finish Method – Physical Design

The finish method of the removal batch class is responsible for performing some important housekeeping. The record inserted into the KB Global Search History object by the search process is updated to reflect the current final status of the batch.

1.8. Custom Object Definitions and Discussion

1.8.1. Knowledge Search and Replace History

This object manages the history records that summarize the batch processing metadata, search scope, and the results of each search and replacement run. Important metadata that are recorded are the batch IDs of the search process(es) and the replacement process, the user, time, and date of the request, and the number of articles searched, articles identified as candidates for update, and the number of articles physically updated.

A search history record is created in the BatchKnowledgeGlobalSearch batch process, and later updated in the BatchKnowledgeGlobalReplace batch process at the end of the run. A record is always created regardless of the success or failure of the replacement process, or if the replacement process is not executed (as in the case of a search only request). Records in this object are intended to be retained indefinitely as they serve as a permanent record of search and replacement activity.

1.8.2. KB Global Search and Replace

This object manages the set of tasks (search/replace requests) created in the first job of the search and replace batch process (BatchKnowledgeGlobalSearch). One record is created for each separate article identified as a replacement target. More specifically, one record is

added for each article of the current type and publishing status that meets the search criteria specified by the user (one or more selected fields contain one or more occurrences of the search string) . The object is key to the entire design as it permits the decoupling of the search and replace processes.

The sets of records written to this object are uniquely identified by the Apex job ID(s) of the current batch process(es). The BatchKnowledgeGlobalReplace process then reads this group of records as source data, performs the physical character string replacement in the article custom field(s), and then deletes the entire group of records from the search and replace object in the final finish method of that class.

1.9. Design Constraints and Challenges

There were several technical challenges that were encountered both in the design and testing phases of this project that one should be aware of prior to implementing the Global Search and Replace utility:

1.9.1. Functional Requirement: Support containing searches on both content and HTML markup

- Technical Considerations: A containing search can only be performed on a knowledge custom text field via Apex class methods (specifically by using Pattern/Matcher objects and methods or, for some applications, the indexOf String class method). SOQL queries cannot be used to perform a containing search since they are subject to the same 253-character limitation as described above for Reports. In addition, SOSL queries cannot be used since this type of search leverages the search engine which does not support searching across HTML markup. This implies that the 'search' for a specific character string or pattern must be performed serially across all articles by a programmatic inspection of each text field.

- Design Decision: Leverage batch Apex to support SOQL queries on objects with (basically) an unlimited number of records.

- Design Approach: Read all articles of a certain type and publish status serially (sorted by article number). Batch apex (see next bullet) should be able to process an arbitrarily large number of articles by automatically 'chunking' the query (and thus the batch processing) into manageable subsets of articles.

- Custom Workaround: The batch Apex execute method works properly when reading records from essentially any other object type, including custom objects. The Global

Search and Replace utility implements a custom workaround in which metadata (essentially the record Id and Knowledge Article Id) for each article in the knowledge article version object being searched is first copied to a record in a custom object (Notes 16, 17).

1.9.2. Functional Requirement: Support containing searches on an arbitrarily large set of knowledge article version records

- Technical Considerations: While it is not possible to leverage SOQL directly on a knowledge article version object to perform a containing search, SOQL is nonetheless required in the overall solution. A SOQL query supports the retrieval of article metadata that is used in the creation of records for the custom object used to support the workaround for the batch Apex execute method bug (identified above).

- Apex Transaction Governor Limits:

    - Transaction governor limits do not allow SOQL queries on objects having greater than 50K records within a single transaction.
    - Governor limits also prevent processing more than 10K DML requests
    - in a single transaction (e.g., record inserts into the custom object used for the workaround described above).
    - Transactions cannot consume more than two seconds of processing time, which is typically insufficient when performing containing searches on multiple large rich text fields.

- Design Decision: Leverage batch Apex to support SOQL queries on objects with (basically) an unlimited number of records and which support processing times in excess of two seconds.

1.10. Discussion Regarding the Need for Asynchronous (Batch Mode) of Design

The physical design in which the search process is decoupled from the replacement process, as well as the decision to use a batch mode of operation as a whole, was influenced by the following issues:

Salesforce OLTP transactions have strict governor limits.

1. Max number of articles that can be returned in a query – this number, 50,000, does not support the primary use case of searching an record type with an unlimited number of records. Batch mode supports a virtually unlimited number of records.

2. Max number of seconds of CPU time that can be assigned to a transaction – this number, approx. two seconds, was proven to be insufficient due to the CPU-intensive operations being performed
3. Max number of custom object records that can be modified (DML operations) – this number, 10,000, was far too restrictive, especially in light of the requirement for a workaround to the knowledge record type QueryLocator object issue.
4. Max number of knowledge article records that can be modified (KBManagement.PublishingService) – the knowledge publishing API is not bulk-operation enabled, effectively limiting the number of articles that can be updated during one transaction to approx. 50.

1.11. Installing and Configuring the Global Search and Replace Utility Installation prerequisites:

- Salesforce Knowledge must be activated and configured
- At least one record type must be defined that has a minimum of one custom field of type text, text area, or rich text area (this is required in order for the test classes to run successfully)
- At least one draft and/or published article should be created for testing

Install the managed package in your development or production org via Appexchange.

The managed package will develop the following objects to your org:

-Application:

Search And Replace App (Classic app)
Search And Replace Lightning App (Lightning app)

-Apex Classes:

BatchDelAuxSearchAndReplaceObj (batch class)
BatchDelAuxSearchAndReplaceObj_Test (test class)
BatchKnowledgeGlobalSearch (batch class)
BatchKnowledgeGlobalSearch_Test (test class)
BatchKnowledgeGlobalReplace (batch class)
BatchKnowledgeGlobalReplace_Test (test class)
LanguagePicklist (controller class)
LanguagePicklist_Test (test class)
Response (JSON wrapper class)
Response (test class)
SearchReplaceController (controller class)
SearchReplaceController_Test (test class)

SearchReplaceHandler (handler class)
SearchReplaceHandler_Test (test class)
SearchReplaceUtils (utils class)
SearchReplaceUtils_Test (test class)

-Visualforce Pages:

KnowledgeSearchAndReplacePage
WarningPage (displays an alert when trying to create auxiliar records from record tab)

-Custom Objects:

KB Global Search History (Appendix 2.1.1)
Mapping Articles (Appendix 2.1.2)
KB Global Search and Replace (Appendix 2.1.3)

-Custom Tabs:

KB Global Search History
Search and Replace Lightning Tab
Search and replace Vfp

-Permission Set:

Search Replace User Permissions

Once the custom objects and page/classes have been created, the necessary security (profile) permissions must be defined for each of these objects to provide end users with access to the utility. Note that it is presumed that the user has the Manage Articles permission, as well as full CRUD permissions on each record type. Since the utility is designed using batch Apex, a full set of permissions may not be required since batch jobs run in the context of the system, not the user. A permission set (Search Replace User Permissions) is available with permissions for the objects provided in this package but no Knowledge permissions were added.

In addition, the KB Global Search and Replace and KB Global Search History objects have been defined with two rich text area fields named Search String and Replacement String. The length of these two RTA fields has been set to the default of 32KB (precisely 32, 768 bytes) . In order for the utility to properly support search and replace operations these field sizes may need to be increased. Specifically, the maximum length for both of these fields in these two objects should reflect the maximum size allowed for a rich text field in the org

into which this package has been installed (maximum size is 128,000, this is an org-wide setting and differs across orgs).

The custom tab entitled Search and Replace Lightning Tab represents the main end user interface to the utility and was designed to be added to the Salesforce application(s) that are used by knowledge base administrators.

Ensure that the test classes execute correctly and provide a minimum of 75% coverage for each of the Apex classes documented above.

1.12. Using the Global Search and Replace Utility

1.12.1. General Instructions

These general instructions apply to both of the main modes of operation for Global Search and Replace – Search Only and Search and Replace. See the next two sections for comments specific to each one of these operations.

Navigate to the custom tab that has been defined for the Knowledge Global Search and Replace utility. This can be accessed from the existing application or may be accessed from the Salesforce application tab list.

First, specify a value for the search string. The search string provides full support for Regular Expressions (RegEx) and, in fact, this must be considered when defining a search expression. Regular Expressions are case sensitive and provide support for complex pattern matching within the selected search field(s). The following is a list of the RegEx reserved characters that require an escape symbol '\' if they are to be treated as a literal (these are referred to as metacharacters as they have a special meaning when included as part of a Regular Expression pattern):

```
[ \ ^   $ . | ?  *      + ( )
```

For more information on RegEx syntax, consult one of the many online references to the subject (Note 1).

Two options are provided for simplifying search expressions. To disable RegEx pattern matching completely and treat the entire search string as a literal (including all occurrences of metacharacters), do not select the RegEx checkbox. To make the search string case sensitive, select the Case Sensitive checkbox. These two options are complementary and can be used together or independently of each other.

Next, you can select the channel where you want to base your search on. This is optional and its purpose is to help refine the search query.

Select a publication status (either Draft or Published) and a Language. This is important because knowledge articles can be found in only one publishing status and language at a time.

Next, select one record type. If a search and replace operation is required across the entire knowledge base, then it needs to be performed separately for each record type defined in the org (see general caveats, below, and Note 18).

Select at least one text field in which to perform the search. Multiple text fields are fully supported and, in fact, all fields may be selected if a universal search and replacement is desired. This feature allows the search to be targeted to one or more specific fields in each record type.

The Search Only button submits just a search and does not replace any articles. Statistics will be shown after the batch is completed.

The Replace button submits the search and replacement process for background execution (specifically, it creates and queues an Apex batch job). Since batch jobs are assigned resources based on availability, there may be a significant delay between the time the Search button is selected and the actual execution of the request. The status of the batch job can be monitored from this page by watching the progress bar above. The status message will be one of the following:

- Not initialized – Default message, displayed prior to selecting Search
- Searching in progress - Search job has been submitted and is either queued or is processing
- Search Complete (Search Only mode) – Search job has completed
- Replace in Progress – Search job has completed processing and Replacement job is either queued or is processing
- Replacement Completed – Replacement job has completed processing

If the user navigates away from this page and returns at a later point in time, the context of the original request will be shown.

1.12.2. Search-only Mode

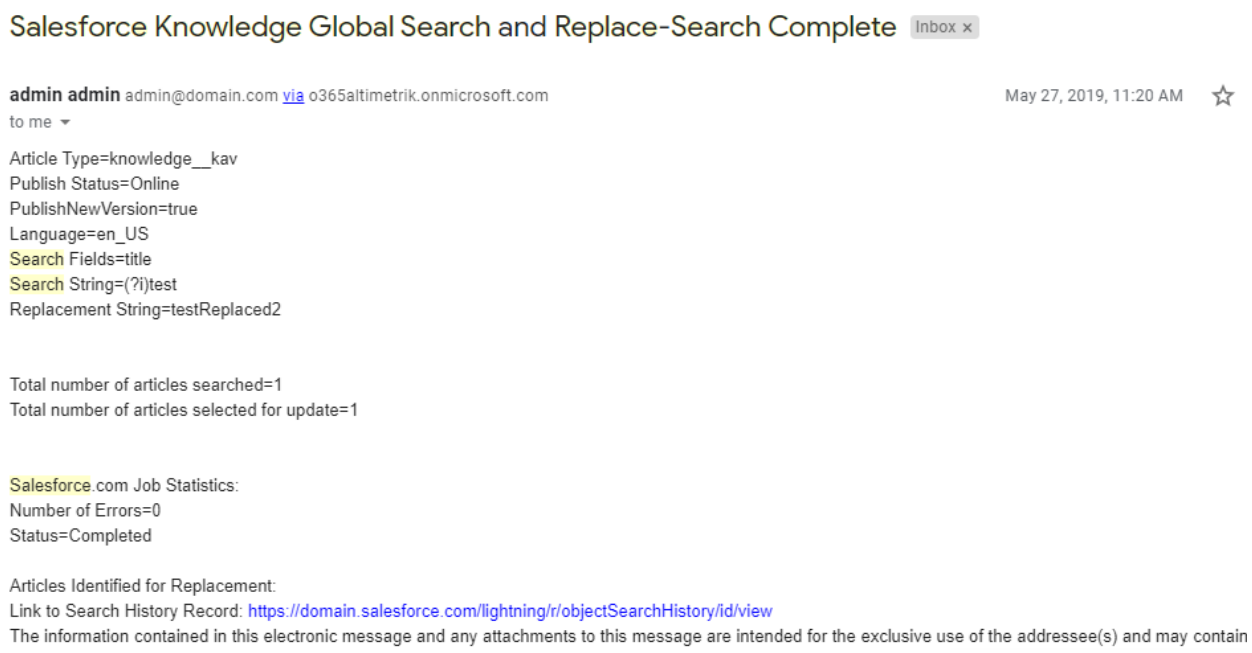
When the user selects the ‘Search Only’ checkbox, the replacement operation is not performed, regardless of any value that may have been entered in the replacement value field. This mode of operation was designed to allow users to determine the impact of a

replacement in terms of the number of articles that would have been updated if the replacement operation had been processed.

The system will confirm the number of articles searched as well as the number of articles that meet the search criteria (articles selected). The status message will change to Search Complete indicating that the search job has completed successfully.

In addition to the simple counts provided on the Search and Replace home page, an email notification can also be created and is sent to the user to confirm the successful completion of the search job. The email notification contains a complete statistics and a link to the csv generated in the process with all articles that were searched (and potentially replaced).

Figure 3 – Search-Only Mode Email Notification


Salesforce Knowledge Global Search and Replace-Search Complete Inbox ×

admin admin admin@domain.com via o365altimetrik.onmicrosoft.com May 27, 2019, 11:20 AM

to me

Article Type=knowledge__kav
Publish Status=Online
PublishNewVersion=true
Language=en_US
Search Fields=title
Search String=(?i)test
Replacement String=testReplaced2

Total number of articles searched=1
Total number of articles selected for update=1

Salesforce.com Job Statistics:
Number of Errors=0
Status=Completed

Articles Identified for Replacement:
Link to Search History Record: https://domain.salesforce.com/lightning/r/objectSearchHistory/id/view
The information contained in this electronic message and any attachments to this message are intended for the exclusive use of the addressee(s) and may contain

1.12.3. Search-and-Replace Mode

If the Replace button is selected, a complete search and replace operation is performed on all articles that meet the search criteria. It is important to note that all replacement operations are permanent and cannot automatically be reversed (Note 20)

When the search job completes, a second batch job is submitted to perform the string replacement operation. This separate batch job processes each article identified in the search process according to the following rules:

- If the publishing status selection is set to Published, the replacement job first checks to see if a Draft version of this article currently exists:
    - If a draft version exists, then the replacement is made on the Draft (the Draft is updated) but the article is not republished
    - If a draft version does not currently exist, then a new Draft is created, the replacement is performed on this new Draft, and the Draft is automatically republished as a new version.
- If the publishing status selection is set to Draft, the replacement(s) are made to the existing Draft versions (the Draft(s) are updated). No publishing operations are performed.

This design was selected to preserve the integrity of any existing Drafts that represent work-in-progress by other users (Note 21)

The Knowledge Search and Replace functionality has been designed to support multiple concurrent search and replace requests, submitted by the same user or by multiple users. Each request is independent; in the event that an update operation is requested on the same knowledge article, the replacement operations are processed in a sequential (FIFO) manner.

The system will confirm the number of articles searched, selected, and updated. The status message will change to Search Complete – Replacement Complete indicating that the search and replacement job has completed successfully.

In addition to the simple counts provided on the Search and Replace home page, an email notification can also be created and is sent to the user to confirm the successful completion of the search and replacement job.

The replacement confirmation email will also include a link to a log with any exceptions, such as existing Drafts for published articles or failure to update a Draft article. A close examination of this log is recommended after each search and replace operation.

Figure 4 – Search and Replacement Mode Email Notification

Salesforce Knowledge Global Search and Replace-Replacement Complete

Inbox ×

2 Stnd via o365altimetrik.onmicrosoft.com Wed, 0

to me

Record Type = RC Pkg 04
Publish Status = Online
Search Fields = title
Search String = \QTHOUSAND\E
Replacement String = thousand

Total number of articles searched = 96460
Total number of articles updated = 96460

Salesforce.com Job Statistics:
Number of Errors=0
Status=Completed

Link to Search History Record: https://snr-pkg.my.salesforce.com/lightning/r/sr_KB_Global_Search_History__c/a043i0000071ZhAAAU/view

1.13. Footnotes

1. An example of an online RegEx reference guide and tutorial is http://www.regular-expressions.info/tutorial.html
2. If a custom report type is not created, knowledge reporting is limited to article metadata fields only.
3. When reporting on article custom field values (i.e., using a custom report type defined on a record type), reporting is limited to one record type at a time
4. It has been determined that this limitation is not going to be relaxed any time soon.
5. Salesforce Knowledge limits searches on custom field objects to one record type at a time
6. Date filtering has not been implemented in this version of the utility.
7. The specific exclusion is Pick List fields, which are not indexed by Salesforce Knowledge and therefore are not searchable.
8. While many customers will not be impacted by the 50K row limitation, many others will and one of the main design goals was to support knowledge objects of an arbitrary size
9. The RegEx 'special' characters (referred to as metacharacters) include [\ ^ $. | ? * + ( ). These must be escaped with a backslash '\' in order to be treated as literals. A checkbox labeled 'Regex' should be disabled to treat the entire search string, including metacharacters, as a literal without the need to escape each special character. The replacement string field is a character string which is always treated as a literal (i.e., special characters are not applicable).
10. Notification could be automated by adding the proper logic to the Replacement job.
11. Salesforce limits the number of Apex batch jobs that can be active at any one point-in-time to five. There may be a noticeable delay from the time the search job was submitted up until the point it is actively processing. The Apex job status during this delay will reflect as 'queued'.
12. This may or may not result in any contention across requests. The search and replace process does not lock any records, and the replacement operation is always separate and distinct from the search. If a previous update has in some way modified the search string in one or more fields (the string can no longer be found), the replacement is simply skipped.
13. For multilingual knowledge bases, a language selector was implemented in the search UI. Note that only one language may be searched at a time, implicitly making a search and replacement operation effective against a specific record type-publication status-language combination.
14. See also https://success.salesforce.com/issues_view?id=a1p30000000SwH3
15. The workaround documented for this issue, namely to disable the feature known as 'Apex Chunking', could not be used, primarily due to the requirement that chunking is absolutely necessary to process record types with a large number of records.

16. Records in this custom object are then queried and referenced by the batch Apex execute method. For a complete technical discussion of this workaround please see the Appendix of this whitepaper.
17. To assist with performance and to limit the number of records written to the custom object, some high-level search constraints (such as Publish Status and (future) Create, Last Modified, and/or Published Dates are applied to the initial knowledge article version object via a SOQL query.
18. Future enhancements may include metadata date fields such as date created.
19. Metadata and internal text fields such as UrlName are not supported and are specifically excluded from the list.
20. It is possible to revert back to a previous version of an article since articles are updated using Publish as a New Version. To back out a replacement, the user would need to manually edit each article and revert it to the previous version, one at a time.
21. There is a chance that an existing Draft is currently being edited by another user when the Draft is updated by the replacement operation. When the other user saves their work, the replacement operation would be effectively lost. This is the same behavior that would occur if the replacement operation were being performed from the native platform application.

1.14. Suggested Enhancements

Recommended Next Steps for Preparing for Production Deployment:

- Additional search filters, such as metadata dates including date published and date last modified, should be added to the search home page to support a greater variety of use cases.

2. Appendices

2.1. Custom Object Definitions

2.1.1. Knowledge Search and Replace History

Details

Description

API Name
sr_KB_Global_Search_History__c

Custom
✓

Singular Label
Lightning KB Global Search History

Plural Label
Lightning KB Global Search History

Enable Reports

Track Activities

Track Field History

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

Fields & Relationships

26 Items, Sorted by Field Label

FIELD LABEL ▲	FIELD NAME	DATA TYPE
Apex Batch Id	Apex_BatchId__c	Text(18)
Apex Replacement Batch Id	Apex_Replacement_BatchId__c	Text(18)
Articles Searched	Articles_Searched__c	Number(7, 0)
Articles Selected	Articles_Selected__c	Number(7, 0)
Articles Updated	Articles_Updated__c	Number(7, 0)
Batch Status	Batch_Status__c	Text(20)
Channels	Channels__c	Text(64)
Created By	CreatedById	Lookup(User)
Error	Error__c	Text(255)
Field Names	Field_Names__c	Long Text Area(32768)
Knowledge Search and Replace History	Name	Text(80)
Language	Language__c	Text(5)
Last Modified By	LastModifiedById	Lookup(User)
NewKnowledgeArticleId	NewKnowledgeArticleId__c	Text(18)

OldKnowledgeArticleId	OldKnowledgeArticleId__c	Text(18)
Owner	OwnerId	Lookup(User,Group)
PublishNewVersion	PublishNewVersion__c	Text(5)
PublishStatus	PublishStatus__c	Text(12)
Record Type	Record_Type__c	Text(80)
RecordTypeId	RecordTypeId__c	Text(18)
Replacement Channels	Replacement_Channels__c	Text(64)
Replacement Log	Replacement_Log__c	Long Text Area(32768)
Replacement String	Replacement_String__c	Long Text Area(32768)
Search Only	Search_Only__c	Checkbox
Search String	Search_String__c	Long Text Area(32768)
User_Id	User_Id__c	Text(18)

2.1.2. SR Mapping Articles

Details

Description
Map of old to new article id if the id change

API Name
sr_Mapping_Articles__c

Custom
✓

Singular Label
Mapping Articles

Plural Label
Mapping Articles

Enable Reports

Track Activities

Track Field History

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

Fields & Relationships

6 Items, Sorted by Field Label

Quick Find

FIELD LABEL	FIELD NAME	DATA TYPE
Created By	CreatedById	Lookup(User)
Last Modified By	LastModifiedById	Lookup(User)
Mapping Articles Name	Name	Text(80)
NewKnowledgeArticleId	NewKnowledgeArticleId__c	Text(18)
OldKnowledgeArticleId	OldKnowledgeArticleId__c	Text(18)
Owner	OwnerId	Lookup(User,Group)

2.1.3. Lightning KB Global Search and Replace

Details

Description

API Name
sr_KB_Global_Search_And_Replace__c

Custom
✓

Singular Label
Lightning KB Global Search And Replace

Plural Label
Lightning KB Global Search And Replace

Enable Reports

Track Activities

Track Field History

Deployment Status
Deployed

Help Settings
Standard salesforce.com Help Window

Fields & Relationships

21 Items, Sorted by Field Label

Quick Find

FIELD LABEL	FIELD NAME	DATA TYPE
Apex Batch Id	Apex_BatchId__c	Text(18)
Article Id	ArticleId__c	Text(18)
Article Number	ArticleNumber__c	Text(9)
Channels	Channels__c	Text(64)
Created By	CreatedById	Lookup(User)
Exception	Exception__c	Checkbox
Exception Comments	Exception_Comments__c	Text Area(255)
Field Names	Field_Names__c	Long Text Area(32768)
IsMasterLanguage	IsMasterLanguage__c	Text(5)
KB Global Search And Replace Name	Name	Text(80)
Knowledge Article Id	KnowledgeArticleId__c	Text(18)
Language	Language__c	Text(5)
Last Modified By	LastModifiedById	Lookup(User)
Owner	OwnerId	Lookup(User,Group)
PublishNewVersion	PublishNewVersion__c	Text(5)

PublishStatus	PublishStatus__c	Text(12)
Record Type	Record_Type__c	Text(80)
RecordTypeId	RecordTypeId__c	Text(18)
Replacement Channels	Replacement_Channels__c	Text(64)
Replacement String	Replacement_String__c	Long Text Area(32768)
Search String	Search_String__c	Long Text Area(32768)

2.2. Custom Tab Definitions

2.2.1. Article Search/Replace Visual Force Page

Visualforce Tab

Search and Replace (Classic)

Below is the information for the custom tab. Click Edit to change the custom tab.

Custom Tab Definition Detail Edit Delete

Tab Label	Search and Replace (Classic)	Tab Style	Globe
Tab Name	sr_Search_and_Replace_vfp	Splash Page Custom Link	
Content	sr_KnowledgeSearchAndReplacePage		
Description			
Created By	Fabricio Aldecosea, 7/19/2019 9:43 AM	Modified By	Fabricio Aldecosea, 7/23/2019 11:40 AM

2.2.2. Article Search/Replace Lightning Component

Lightning Component Tab

Search and Replace (Lightning Tab)

Below is the information for the custom tab. Click Edit to change the custom tab.

Custom Tab Definition Detail Edit Delete

Tab Label	Search and Replace (Lightning Tab)	Tab Style	Lightning
Tab Name	sr_Search_and_Replace_Lightning_Tab	Splash Page Custom Link	
Content	sr_LightningSearchAndReplace_Tab_Component		
Description	Search and Replace lightning tab		
Created By	Fabricio Aldecosea, 8/8/2019 10:07 AM	Modified By	Fabricio Aldecosea, 8/8/2019 10:12 AM

2.2.3. Custom Objects

Custom Object Tab

Lightning KB Global Search History

Below is the information for the custom tab. Click Edit to change the custom tab.

Custom Tab Definition Detail Edit Delete

Tab Label	Lightning KB Global Search History	Tab Style	Telescope
Object	Lightning KB Global Search History	Splash Page Custom Link	
Description			
Created By	Fabricio Aldecosea, 6/13/2019 7:09 AM	Modified By	Fabricio Aldecosea, 6/13/2019 7:09 AM

Application Changelog

[1.9] - 2023-02-16

Fixed	• Email confirmation for Search-Only jobs not being delivered

[1.8] - 2021-04-15

Fixed	• The tool was not displayed on the Classic UI

[1.7] - 2021-02-03

Added	• Toggle to change article visibility. Now the user can decide if after the replacement the article visibility stays as it is, changes to be visible only on internal or changes to be visible in other channels too.
Fixed	• Logic on the toggle to select channel: When the switch for "Change article visibility" was left disabled and then a replacement was performed, the article was not visible in the channels where it was visible before the replacement. Instead, the article was visible in Internal.
Changed	• Maximum number of articles that can be replaced without hitting a Heap size limit decreased from 100.000 to 80.000. This was updated in the official guide.

[1.6] - 2020-10-23

Added	• Users can now choose to not publish articles that were modified and keep the new versions as drafts. Users can manually publish them later. • CSV file with replacement job contains some more data.
Changed	• Minor UI/UX refactor

[1.5] - 2019-03-10

Fixed	• This is the same package as v1.4 - It was uploaded again because some users had issues installing the package uploaded in Appexchange. The new version helped solve this problem.

[1.4] - 2019-02-03

Added	• In the Results tab, a direct link to the history record that just finished running was added
Fixed	• The character limit for the Record Type label was extended to avoid errors when too many record types are selected at the same time.

[1.3] - 2019-11-13

Fixed	• Security issues found on source scanner report

[1.0-1.2] - 2019-10-10

Added	• Initial Drafts of the application