1. ***Move Data to Azure***

To build the Knowledge Store, your data will need to be in Azure. In this scenario, we are using a blob storage container to hold our csv file, but Azure Cognitive Search can connect to [many different Azure data sources](https://docs.microsoft.com/en-us/azure/search/search-indexer-overview#supported-data-sources).

If your data is not already in Azure, create an Azure **Storage Account** resource, and upload your data to a blob storage container.

Before you leave the Storage Account page, use a link on the left navigation pane to open the Access Keys page. Get a connection string to retrieve data from Blob storage. A connection string looks similar to the following example: DefaultEndpointsProtocol=https;AccountName=<YOUR-ACCOUNT-NAME>;AccountKey=<YOUR-ACCOUNT-KEY>;EndpointSuffix=core.windows.net

1. ***Create Azure Resources***

Once in the Azure Portal, create the following resources:

* Azure Cognitive Search
* Azure Cognitive Services

**Azure Cognitive Search** will allow you to not only index your data in a format that is searchable and ready for additional analysis, but it will also be the platform for you to add enrichments to your data such as sentiment analysis, language detection and key phrase extraction. Azure Cognitive Search pricing tiers can be found [here](https://docs.microsoft.com/en-us/azure/search/search-sku-tier). A free tier can be used for the purposes of this tutorial, but it can only index a maximum of 50 MB.

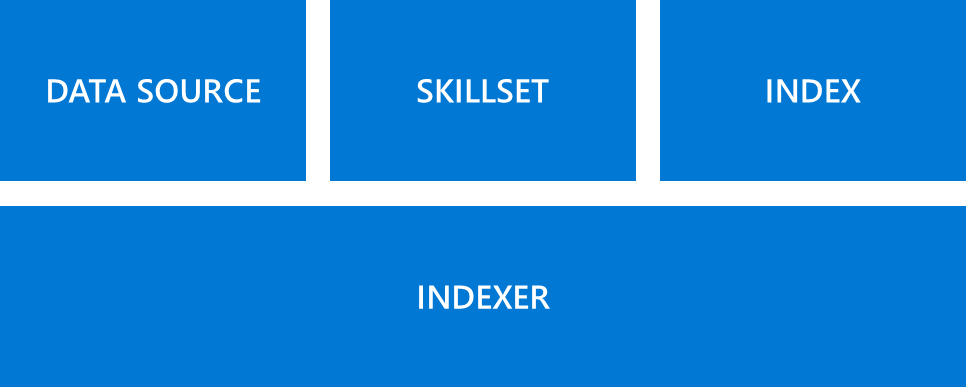
**Azure Cognitive Services** is a suite of pre-built AI models that Azure Cognitive Search will leverage for the enrichments.

Before navigating away from these resources, use a link on the left navigation pane to open the Keys page. Copy the primary admin key value for the Azure Cognitive Search resource and repeat the process for the Azure Cognitive Service resource. You will need both of these keys later.

1. ***Understand the Azure Cognitive Search Model***

Azure Cognitive Search is made up of 4 elements: data source, skillset, index and indexer. In the following steps we will create each one of these elements for your use case. The purpose of each is as follows:

* The **Data Source** contains all of the metadata needed to connect to your data. It contains information such as the connection string and the container where your data is.
* The **Index** has information about the search index (and the shape of that index, which fields are sortable, retrievable, etc.) or in this case, the structure of the tables we will use in the Knowledge Store.
* The **Skillset** contains information about each of the enrichment steps and their dependencies. For example, we will be adding enrichments such as key phrase extraction and sentiment analysis which we will specify in the skillset. The skillset is also where you specify your Azure Cognitive Services information to provide the AI to power the skills.
* The **Indexer** brings it all together. It has pointers to the other 3 and is in charge of taking data from the data source, running the skillset and pushing the data into the index.



1. ***Set up Postman***

We will be using Postman to create the Knowledge Store. If you don’t have this tool already, you can download it [here](https://www.postman.com/downloads/).

When you create a knowledge store, you must issue four HTTP requests:

* **PUT request to create the index:** This index holds the data that Azure Cognitive Search uses and returns.
* **POST request to create the datasource:** This datasource connects your Azure Cognitive Search behavior to the data and knowledge store's storage account.
* **PUT request to create the skillset:** The skillset specifies the enrichments that are applied to your data and the structure of the knowledge store.
* **PUT request to create the indexer:** Running the indexer reads the data, applies the skillset, and stores the results. You must run this request last.

1. ***Create the Search Index***

Create an Azure Cognitive Search index to represent the data that you are interested in searching, filtering, and applying enhancements to. Create the index by issuing a PUT request to https://**{your-search-service-name}**.search.windows.net/indexes/**{your-index-name}**?api-version= 2019-05-06. In the URL, replace **{your-search-service-name}** with the name of your Azure Cognitive Search resource and **{your-index-name}** with any name you would like to use for the index.

In the “Headers” section of Postman, add two rows: (1) for “api-key” as the key and use your Subscription Key from Azure Cognitive Search as the Value, and (2) for “Content-Type” and specify the value as “application/json.” In the “Body section,” add the following code to create the index\*:

{

    "name": "your-index-name",

    "fields": [

        { "name": "CREATEDDATE", "type": "Edm.DateTimeOffset", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "ItemSearchName", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "ProductType", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "Reviews", "type": "Edm.String", "filterable": false,  "sortable": false, "facetable": false },

        { "name": "Rating", "type": "Edm.Int64", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "AzureSearch\_DocumentKey", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false, "key": true },

        { "name": "metadata\_storage\_content\_type", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "metadata\_storage\_size", "type": "Edm.Int64", "searchable": false, "filterable": false, "sortable": false, "facetable": false},

        { "name": "metadata\_storage\_last\_modified", "type": "Edm.DateTimeOffset", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "metadata\_storage\_name", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "metadata\_storage\_path", "type": "Edm.String", "searchable": false, "filterable": false, "sortable": false, "facetable": false },

        { "name": "Sentiment", "type": "Collection(Edm.Double)", "searchable": false, "filterable": true, "retrievable": true, "sortable": false, "facetable": true },

        { "name": "Language", "type": "Edm.String", "filterable": true, "sortable": false, "facetable": true },

        { "name": "Keyphrases", "type": "Collection(Edm.String)", "filterable": true, "sortable": false, "facetable": true }

    ]

}

\*Note: Before issuing this request or any of the ones below, “Created Date” from the dataset, was renamed to “CreatedDate” as the search index does not accept spaces in column names.

Select **Send** to issue the PUT request. You should see the status 201 - Created. If you see a different status, in the Body pane, look for a JSON response that contains an error message.

1. ***Create the Data Source***

To create the datasource, send a POST request to https://**{your-search-service-name}**.search.windows.net/datasources?api-version=2019-05-06. You must set the api-key and Content-Type headers as discussed earlier.

In the Body section, add the request below. Replace “your-datasource-name” with the name of your choosing for the data source. For the connection string, paste the connection string you copied earlier, and ensure it’s the same structure as the one below. You can also just replace the Account Name and Key. Finally, replace “YOURCONTAINERNAME” with the name of the container where your csv file resides.

{

    "name" : "your-datasource-name",

    "type" : "azureblob",

    "credentials" : { "connectionString" : "DefaultEndpointsProtocol=https;AccountName=STORAGEACCOUNTNAME;AccountKey=STORAGEACCOUNTKEY;EndpointSuffix=core.windows.net" },

    "container" : { "name" : "YOURCONTAINERNAME" }

}

Select **Send** to issue the POST request.

1. ***Create the Skillset***

The next step is to specify the skillset, which specifies both the enhancements to be applied and the knowledge store where the results will be stored. Create a new tab with a PUT request to https:// **{your-search-service-name}**.search.windows.net/skillsets/**{your-skillset-name}**?api-version=2019-05-06. Update the URL with the name of your Search service, as well as the skillset name of your choosing. Set the api-key and Content-type headers as you did earlier.

There are two large top-level objects: skills and knowledgeStore. Each object inside the skills object is an enrichment service. Each enrichment service has inputs and outputs. The LanguageDetectionSkill has an output targetName of Language. The value of this node is used by most of the other skills as an input. The source is document/Language. The capability of using the output of one node as the input to another is even more evident in ShaperSkill, which specifies how the data flows into the tables of the knowledge store.

Add the code below to the “Body” section of the request. The knowledge\_store object needs to connect to the storage account so add your **storage account connection string to the request below**. Knowledge\_store contains a set of mappings between the enhanced document and tables and columns in the knowledge store.

Finally, the skillset also uses the Azure Cognitive Service resource you created for the AI enrichment. In the request below, replace the Azure Cognitive Service resource name with yours, as well as the subscription key you extracted at the beginning of the tutorial.

To generate the skillset, select the **Send** button in Postman to PUT the request:

{

    "name": "your-skillset-name",

    "description": "Skillset to detect language, extract key phrases, and detect sentiment",

    "skills": [

        {

            "@odata.type": "#Microsoft.Skills.Text.SplitSkill",

            "context": "/document/Reviews", "textSplitMode": "pages", "maximumPageLength": 5000,

            "inputs": [

                { "name": "text", "source": "/document/Reviews" },

                { "name": "languageCode", "source": "/document/Language" }

            ],

            "outputs": [

                { "name": "textItems", "targetName": "pages" }

            ]

        },

        {

            "@odata.type": "#Microsoft.Skills.Text.SentimentSkill",

            "context": "/document/Reviews/pages/\*",

            "inputs": [

                { "name": "text", "source": "/document/Reviews/pages/\*" },

                { "name": "languageCode", "source": "/document/Language" }

            ],

            "outputs": [

                { "name": "score", "targetName": "Sentiment" }

            ]

        },

        {

            "@odata.type": "#Microsoft.Skills.Text.LanguageDetectionSkill",

            "context": "/document",

            "inputs": [

                { "name": "text", "source": "/document/Reviews" }

            ],

            "outputs": [

                { "name": "languageCode", "targetName": "Language" }

            ]

        },

        {

            "@odata.type": "#Microsoft.Skills.Text.KeyPhraseExtractionSkill",

            "context": "/document/Reviews/pages/\*",

            "inputs": [

                { "name": "text",  "source": "/document/Reviews/pages/\*" },

                { "name": "languageCode",  "source": "/document/Language" }

            ],

            "outputs": [

                { "name": "keyPhrases" , "targetName": "Keyphrases" }

            ]

        },

        {

            "@odata.type": "#Microsoft.Skills.Util.ShaperSkill",

            "context": "/document",

            "inputs": [

                { "name": "name",  "source": "/document/name" },

                { "name": "CreatedDate",  "source": "/document/CreatedDate" },

                { "name": "ItemSearchName",  "source": "/document/ItemSearchName" },

                { "name": "ProductType",  "source": "/document/ProductType" },

                { "name": "Reviews",  "source": "/document/Reviews" },

                { "name": "Rating",  "source": "/document/Rating" },

                { "name": "AzureSearch\_DocumentKey",  "source": "/document/AzureSearch\_DocumentKey" },

                {

                    "name": "pages",

                    "sourceContext": "/document/Reviews/pages/\*",

                    "inputs": [

                        { "name": "SentimentScore", "source": "/document/Reviews/pages/\*/Sentiment" },

                        { "name": "LanguageCode", "source": "/document/Language" },

                        { "name": "Page", "source": "/document/Reviews/pages/\*" },

                        {

                            "name": "keyphrase", "sourceContext": "/document/Reviews/pages/\*/Keyphrases/\*",

                            "inputs": [

                                { "name": "Keyphrases", "source": "/document/Reviews/pages/\*/Keyphrases/\*" }

                            ]

                        }

                    ]

                }

            ],

            "outputs": [

                { "name": "output" , "targetName": "tableprojection" }

            ]

        }

    ],

     "cognitiveServices": {

        "@odata.type": "#Microsoft.Azure.Search.CognitiveServicesByKey",

        "description": "your-cognitive-services-resource-name",

        "key": "your-cognitive-services-key"},

    "knowledgeStore": {

        "storageConnectionString": "DefaultEndpointsProtocol=https;AccountName=STORAGEACCOUNTNAME;AccountKey=STORAGEACCOUNTKEY;EndpointSuffix=core.windows.net",

        "projections": [

            {

                "tables": [

                    { "tableName": "ReviewsDocument", "generatedKeyName": "Documentid", "source": "/document/tableprojection" },

                    { "tableName": "ReviewsPages", "generatedKeyName": "Pagesid", "source": "/document/tableprojection/pages/\*" },

                    { "tableName": "ReviewsKeyPhrases", "generatedKeyName": "KeyPhrasesid", "source": "/document/tableprojection/pages/\*/keyphrase/\*" },

                    { "tableName": "ReviewsSentiment", "generatedKeyName": "Sentimentid", "source": "/document/tableprojection/pages/\*/sentiment/\*" }

                ],

                "objects": []

            },

            {

                "tables": [

                    {

                        "tableName": "ReviewsInlineDocument", "generatedKeyName": "Documentid", "sourceContext": "/document",

                        "inputs": [

                            { "name": "name", "source": "/document/name"},

                            { "name": "CreatedDate", "source": "/document/CreatedDate"},

                            { "name": "ItemSearchName", "source": "/document/ItemSearchName"},

                            { "name": "ProductType", "source": "/document/ProductType"},

                            { "name": "Reviews", "source": "/document/Reviews"},

                            { "name": "Rating", "source": "/document/Rating"},

                            { "name": "AzureSearch\_DocumentKey", "source": "/document/AzureSearch\_DocumentKey" }

                        ]

                    },

                    {

                        "tableName": "ReviewsInlinePages", "generatedKeyName": "Pagesid", "sourceContext": "/document/Reviews/pages/\*",

                        "inputs": [

                            { "name": "SentimentScore", "source": "/document/Reviews/pages/\*/Sentiment"},

                            { "name": "LanguageCode", "source": "/document/Language"},

                            { "name": "Page", "source": "/document/Reviews/pages/\*" }

                        ]

                    },

                    {

                        "tableName": "ReviewsInlineKeyPhrases", "generatedKeyName": "kpidv2", "sourceContext": "/document/Reviews/pages/\*/Keyphrases/\*",

                        "inputs": [

                            { "name": "Keyphrases", "source": "/document/Reviews/pages/\*/Keyphrases/\*" }

                        ]

                    }

                ],

                "objects": []

            }

        ]

    }

}

1. ***Create the Indexer***

The final step is to create the indexer. The indexer reads the data and activates the skillset. The definition of the indexer refers to several other resources that you already created: the datasource, the index, and the skillset.

The parameters/configuration object controls how the indexer ingests the data. In this case, the input data is in a single document that has a header line and comma-separated values. The document key is a unique identifier for the document. Before encoding, the document key is the URL of the source document. Finally, the skillset output values, like language code, sentiment, and key phrases, are mapped to their locations in the document. Although there's a single value for Language, Sentiment is applied to each element in the array of pages. Keyphrases is an array that's also applied to each element in the pages array.

After you set the api-key and Content-type headers and confirm that the body of the request is similar to the code below, select Send in Postman. Postman sends a PUT request to https:// **{your-search-service-name}**.search.windows.net/indexers/**{your-indexer-name}**?api-version2019-05-06. Azure Cognitive Search creates and runs the indexer. Before sending the request, ensure to update the URL with the name of your Azure Cognitive Search resource, as well as an indexer name of your choosing. Additionally, add the names of your indexer, datasource, skillset and index to the request body.

{

    "name": "your-indexer-name",

    "dataSourceName": "your-datasource-name",

    "skillsetName": "your-skillset-name",

    "targetIndexName": "your-index-name",

    "parameters": {

        "configuration": {

            "dataToExtract": "contentAndMetadata",

            "parsingMode": "delimitedText",

            "firstLineContainsHeaders": true,

            "delimitedTextDelimiter": ","

        }

    },

    "fieldMappings": [

        {

            "sourceFieldName": "AzureSearch\_DocumentKey",

            "targetFieldName": "AzureSearch\_DocumentKey",

            "mappingFunction": { "name": "base64Encode" }

        }

    ],

    "outputFieldMappings": [

        { "sourceFieldName": "/document/Reviews/pages/\*/Keyphrases/\*", "targetFieldName": "Keyphrases" },

        { "sourceFieldName": "/document/Language", "targetFieldName": "Language" },

        { "sourceFieldName": "/document/Reviews/pages/\*/Sentiment", "targetFieldName": "Sentiment" },

    ]

}

1. ***Run the Indexer***

In the Azure portal, go to the Azure Cognitive Search service's **Overview** page. Select the Indexers tab, and then select the name of the indexer you created. If the indexer hasn't already run (or started running), select Run. The indexing task might raise some warnings, but unless it says “Failed,” it has successfully indexed your data.

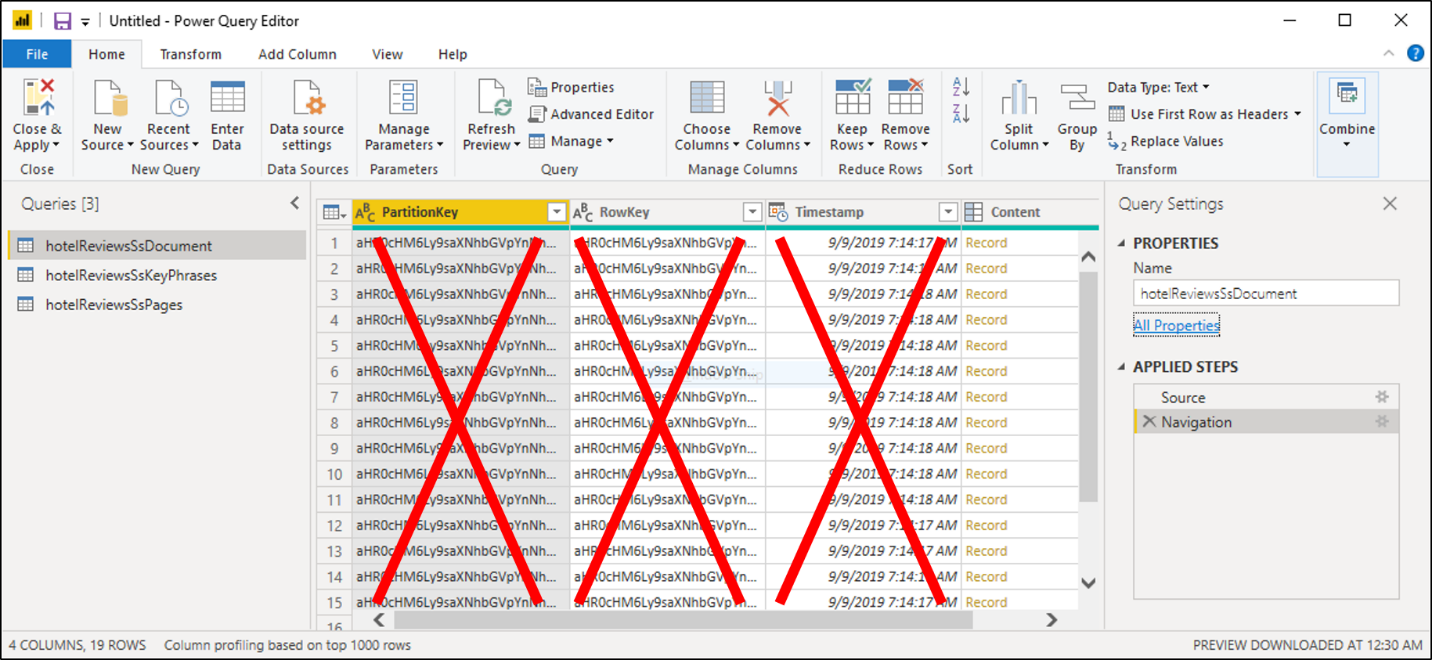
1. ***Connect to Power BI***

After creating the Knowledge Store, you can use Power BI to create powerful visuals for your data.

* Start Power BI Desktop and click **Get data**.
* In the Get Data window, select Azure, and then select Azure Table Storage.
* Click Connect.
* For Account Name or URL, enter in your Azure Storage account name (the full URL will be created for you).
* If prompted, enter the storage account key.
* Select the tables containing the data created by the previous walkthrough:
  + ReviewsDocument
  + ReviewsPages
  + ReviewsKeyPhrases
* Click Load.

On the top ribbon, click **Transform Data** to open the **Power Query Editor**.

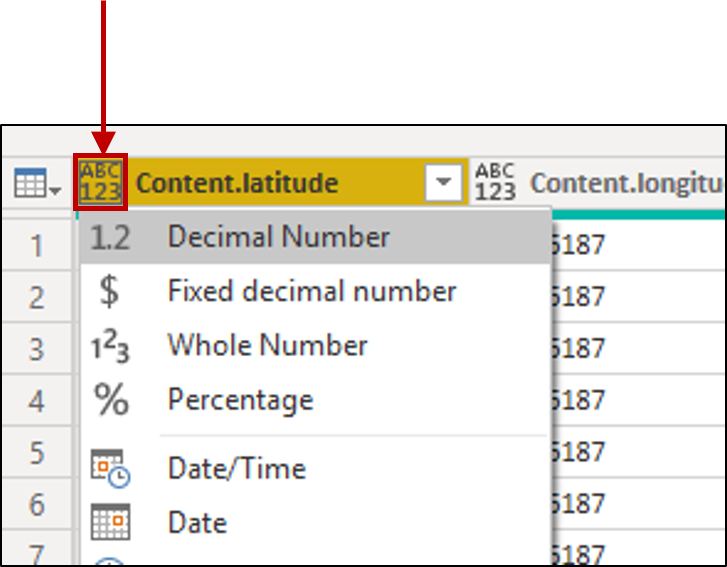
Select the ReviewsDocument table and then remove the PartionKey, RowKey, and Timestamp columns.



Click the icon with opposing arrows at the upper right side of the table to expand the Content. When the list of columns appears, select all columns, and then deselect columns that start with 'metadata'. Click **OK** to show the selected columns.

Repeat these steps for the remaining tables.

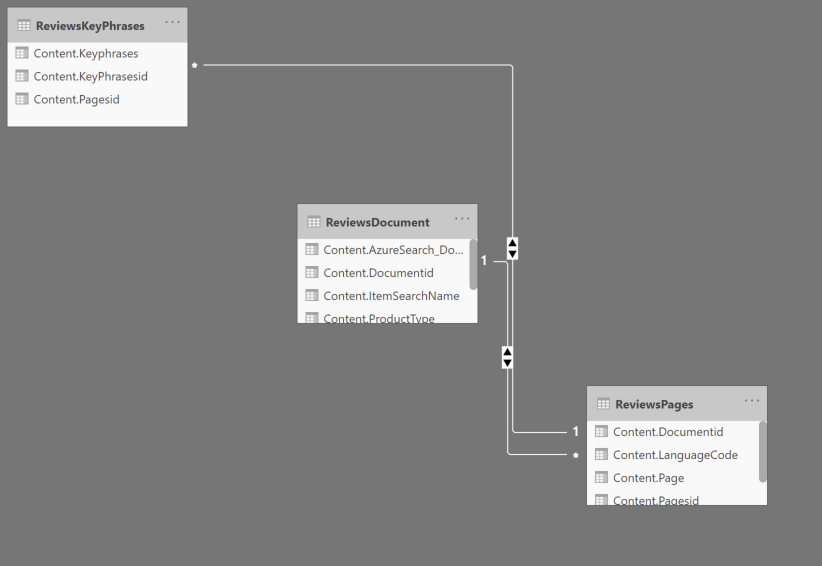
\*Note that in the ReviewsDocument table you will need to change the data type to Whole Number, and in the ReviewsPages table, change the data type for Sentiment Score to Decimal Number (similar to as shown in the image below).



On the command bar, click **Close and Apply**.

Click on the Model tile on the left navigation pane and validate that Power BI shows relationships between all three tables.

Double-click each relationship and make sure that the Cross-filter direction is set to Both. This enables your visuals to refresh when a filter is applied.



Click on the Report tile on the left navigation pane to explore data through visualizations. For text fields, tables and cards are useful visualizations. You can choose fields from each of the three tables to fill in the table or card.

In addition to the built-in visualizations, there are many interesting visualizations in the Visualizations Marketplace. We used one called *Force-Directed-Graph Visualization* to build a network map of products and key phrases.

Additional Resources and references tutorials:

* Create a knowledge store using REST and Postman: <https://docs.microsoft.com/en-us/azure/search/knowledge-store-create-rest>
* Connect a knowledge store with Power BI: <https://docs.microsoft.com/en-us/azure/search/knowledge-store-connect-power-bi>