**News Headline Publishing Tool**

**Table of contents**

Design Requirements

Other Considerations and Applicant Feedback

Applicant’s Additions/Assumptions

How to invoke the application

Database Table Design

Source Code

**Design Requirements**

Stated Design Requirements are below:

**Purpose**

IBM has a corporate communications team which publishes news headlines to the internal homepage for roughly half a million employees, consultants, and business partners. Please design and implement a simple recommendation system to match relevant news headlines to internal employees.

**Assumptions**

For the purpose of this technical assessment, please assume that the employee records are stored in a file, and the news headlines are stored in another file.

An *employee* record has at least the following fields.

ID

name

location

job role

department

A *news headline* has at least the following fields

title

abstract

language

publication date

author

Please feel free to add more fields to either employee or news headline as you see fit.

**Minimum Requirements**

Generate a text file *named employee.txt* with 5,000 - 10,000 records

Generate a text file named *headlines.txt* with 500 - 3,000 records

Implement a service which recommends news headlines to employees such that

matching is performed on the publication date

this services should have a public API that takes the publication date as input and outputs every news headline published on this day to its list of matched employees

Please include documentation on how to run/invoke/install this service. GitHub is recommended.

**Other Considerations**

Ideally, this service would be implemented in a cloud native manner. IBM Cloud would be the best choice, other popular cloud provider such as AWS or Azure are also good.

Ideally, your submissions should represent production level work, so please include test coverage if you normally test your code and provide documentation if you normally provide documentation.

Optionally, you can to take this service to the next level! You may either implement more APIs or document your next steps.

Optionally, describe how you would handle a million employees and three hundred million news headlines across the globe.

**Other Considerations and Applicant Feedback**

Ideally, this service would be implemented in a cloud native manner. IBM Cloud would be the best choice, other popular cloud provider such as AWS or Azure are also good.

**Applicant:** The application is built for Azure cloud using Microsoft Visual Studio 2019 and Microsoft SQL Server v18.2. Application is built locally on applicant’s workstation and, after testing, deployed to the Azure cloud. Finally, the application is tested for quality on the Azure cloud environment.

Ideally, your submissions should represent production level work, so please include test coverage if you normally test your code and provide documentation if you normally provide documentation.

**Applicant:** The scope of application testing included the following.

* The application is tested against the data set of employees and new headlines to verify the results.
* The way to invoke the application – including the date format for the publication date – is tested.

The scope of the application is kept simple to simplify the testing process as well.

Optionally, you can to take this service to the next level! You may either implement more APIs or document your next steps.

**Applicant:** I can think of several additional ways to improve the application. It requires the news headline record to have a tags field to help associate the record to certain categories.

1. Semantic matching: The database can be added with a new semantic-mapping table to semantically map certain employee table phrases to certain key field values in the news headlines table in ways beyond the standard string comparison. Examples:
   1. If the news is about sales, and the employee is in the Sales department then semantic match! How do we know the news is about Sales? By scoring the associated news tags vs. that of each department to see if it is close enough to be assumed to be Sales related.
   2. If the news is about a hurricane and it is at/headed toward the location of the employee, it can/should be “breaking news!”
2. Emergency matching: If a natural disaster hits an area of the country, the employees at the locations close to the area should be considered impacted and related news feed to them (i.e., via the assumed news portal accessible by each employee).
3. More robust matching: The current design allows any or one author match, any or one language match and any or one keyword match (i.e. employee record keyword appearing in news headlines title or abstract text fields). Instead, the employee record can have multiple keywords so we can check if any of them show up in the news headlines title or abstract text fields.

Optionally, describe how you would handle a million employees and three hundred million news headlines across the globe.

**Applicant:** The application can be scaled to handle a million employees and three million news headlines across the globe by leveraging several key concepts such as database sharding, load balancing, redundancy, multiple end points, kubernetes/docker approaches (key concepts for microservices) and a few other techniques described below.

Since the IBM corporation and its intranet are spread across the globe, we should leverage this topographic presence. The employee records have the “location” field which we can use to figure out the global region the employee falls into. The entire dataset of employees should be divided into a number of shards (i.e. horizontal partitioning) each corresponding to a specific region. The sharding is done usually via key ranges, or key-based hash values.

What is probably more tricky is sharding the news headlines table. If we add language field to the employees table, we can div up the news headlines table based on language. Then, we can effectively shard the news headlines table allow the database queries to run much more quickly.

The mail goal in sharding is reducing the number of records to process by each process, minimize the resources needed to process the required data and minimize the execution time for timely completion.

The process of publishing the news headlines should be done per region and in parallel. If a region houses too many employees or headline news records, it should be divided further to achieve parity with other regions in terms of performance.

The other techniques we should use are server pooling (e.g. per region) and load balancing to ensure optimized and accountable response times. Having a pool of (or nodes) of application and database servers provides maximum processing power to handle large volumes of data, optimized load balancing among the servers and availability/continuity for service due to the redundancy. The publishing step can potentially utilize multi-threading for higher throughput.

From a development and management perspective, the application components should be built as separate microservices (i.e., matching vs. publishing) and allow scaling of the resource requirements where needed only. That is, matching the news headlines to the employees is relatively more compute intensive and less IO intensive. Publishing the matched news headlines to employee portals require monumental data movement across the network and more resources shall be needed. Thus, we should scale up the publishing component more compared to the former. Otherwise, we will have allocated more resources to matching than needed.

**Applicant’s Additions/Assumptions**

1. In order to match employees to the news headlines in a meaningful way, I added the following three fields to the employee record.

* Language
* Author
* Keyword [optional] : This is a keyword denoting a topic of interest of the employee. It will be looked up in the title and abstract fields in the news headlines table. In this scenario the publication date and language match are still required, but the author name match is removed to provide flexibility.

1. The Keyword in the employee table is a term name (e.g. database, micro-service, etc.) not a bare bone simple word such as ‘the’, ‘and’ etc. which does not have a translatable/relevant meaning in this exercise’s context.

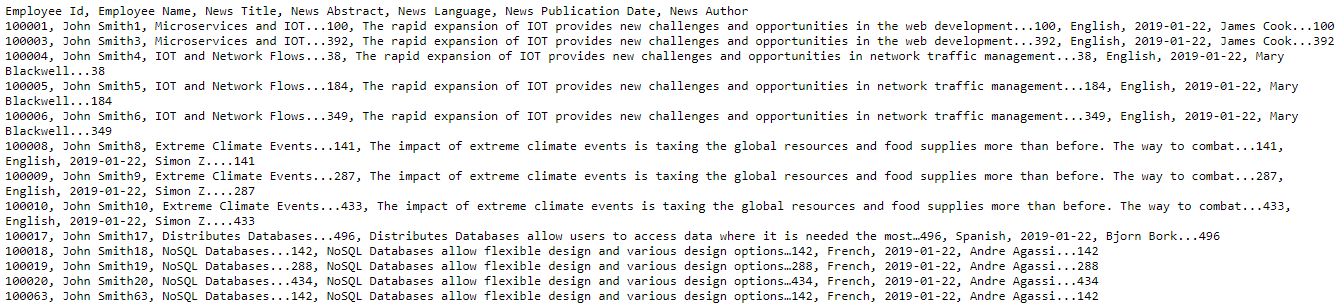
**How to invoke the application**

You can invoke the web application with the publication date included in a single call such as:

<https://corpnewsnotification.azurewebsites.net/api/GetBroadcastDate/2019-01-22>

The format of the application’s web address is <https://corpnewsnotification.azurewebsites.net/api/GetBroadcastDate/yyyy-mm-dd>

The application will return a string list of computed recommendations in string format where each line is a recommendation. The field values are separated by “, ”so that the screen output can be copied into a text file and saved as .csv format:



**Database Table Design**

The application utilizes Microsoft SQL Server v18.2 as the data storage and retrieval system.

The design of the Employees and NewsHeadlines tables are as follows.

CREATE TABLE [dbo].[Employees](

[ID] [float] NULL,

[name] [nvarchar](255) NULL,

[location] [nvarchar](255) NULL,

[jobrole] [nvarchar](255) NULL,

[department] [nvarchar](255) NULL,

[language] [nvarchar](255) NULL,

[author] [nvarchar](255) NULL,

[keyword] [nvarchar](255) NULL

) ON [PRIMARY]

CREATE TABLE [dbo].[NewsHeadlines](

[title] [nvarchar](255) NULL,

[abstract] [nvarchar](255) NULL,

[language] [nvarchar](255) NULL,

[publicationdate] [datetime] NULL,

[author] [nvarchar](255) NULL,

[F6] [nvarchar](255) NULL

) ON [PRIMARY]

**Source Code**

The source code is located at<https://github.com/MYorganci/NewsHeadlines> which is publicly accessible. The complete list of files on GitHub location is as follows.

CorpNewsNotification\_Release.zip: The complete Microsoft Visual Studio 2019 Azure we application project source code file set. I only removed the username and password for database connection from the original source code. Otherwise, it is 100% what is running the application.

employee.txt: Mock up employee data file with the added fields to allow matching the employees to relevant news headlines.

newsheadlines.txt: Mock up data for the newsheadlines file. The file structure is the same as described in business requirements.

Read Me - News Headline Publishing Tool.docx: (This file) The detailed description of the business requirements, how the application is built and what ways it can be further improved.

