PharmaSUG 2022 - Paper AD-170

Using the SAS System® to interface with the Entrez Programming Utilities (E-utilities) of the National Center for Biotechnology Information (NCBI).

Kevin R. Viel, Ph.D., Navitas Life Sciences; Histonis, Incorporated

ABSTRACT

The National Center for Biotechnology Information (NCBI) of the United States National Institute of Health has multiple databases that are generally accessed via a web interface. Popular databases include PubMed, which houses citations for biomedical literature, and PubChem, which houses chemical information, such as chemicals by name, molecular formula, or structure, among other data. NCBI provides a set of eight server-side programs, the Entrez Programming Utilities (E-utilities), that provide a stable interface into the Entrez query and database system. NCBI also made available the NCBI Datasets command line tools datasets and dataformat. These tools are helpful for work with sequences of SARS-CoV-2, for instance. The **goal** of this paper is to introduce the E-utilities and describe two SAS System® macros that use the HTTP procedure using the E-utilities URL and the command line tools.

INTRODUCTION

In observational research and clinical trials, one often uses databases to support the tasks a hand. For instance, in reading a Protocol, one may wish to obtain papers on the indication or treatment. One may need to convert a lab analyte from one unit to the another. Often, the web interfaces suffice, and search terms may not be complex. The need to alter or update the search based on the results of a process (program) or a large task necessitates a programmatic approach.

PubMed is a database of citations, abstracts, and, in some cases, full-text articles for biomedical literature that is freely available from the National Center for Biotechnology Information (NCBI) of the United States National Institute of Health (NIH). PubChem is another freely available NCBI database that contains information on chemicals such name, molecular formula, or structure, patents, among other data. The NCBI provides a set of eight server-side programs, the Entrez Programming Utilities (E-utilities), that provide a stable interface into the Entrez query and database system.

The advantages of using E-utilities included building searches based on the results of a program, which can include choosing useful search field tags. For instance, the author wrote a suite of programs (not presented) to that aligned the peptides in a protein of interest with peptides in other proteins in the reference genome, then searched various NCBI databases, for papers, SNPs (single nucleotides polymorphisms), and other information to investigate immunogenicity (ADA, anti-drug antibodies). At that time, the program ran for hours, if not days, so expanding it, for example, to the microbiome would become too cumbersome and extremely difficult to Validate to do manually. Another potential use would be to obtain a list of lab analytes that do not appear in one's internal databases and to search PubChem for a synonyms and molecular masses for conversion of lab data to controlled terminology, i.e. LBTEST, and preferred units, i.e. to convert mg/dL to mmol/L.

The E-Utility applications are:

- **EInfo** (database statistics)

  Provides the number of records indexed in each field of a given database, the date of the last update of the database, and the available links from the database to other Entrez databases.

- **ESearch** (text searches)

  Responds to a text query with the list of matching UIDs in a given database (for later use in ESummary, EFetch or ELink), along with the term translations of the query.

- **EPost** (UID uploads)

  Accepts a list of UIDs from a given database, stores the set on the History Server, and responds

with a query key and web environment for the uploaded dataset.

- **ESummary** (document summary downloads)

  Responds to a list of UIDs from a given database with the corresponding document summaries.

- **EFetch** (data record downloads)

  Responds to a list of UIDs in a given database with the corresponding data records in a specified format.

- **ELink** (Entrez links)

  Responds to a list of UIDs in a given database with either a list of related UIDs (and relevancy scores) in the same database or a list of linked UIDs in another Entrez database ; checks for the existence of a specified link from a list of one or more UIDs ; creates a hyperlink to the primary LinkOut provider for a specific UID and database, or lists LinkOut URLs and attributes for multiple UIDs.

- **EGQuery** (global query)

  Responds to a text query with the number of records matching the query in each Entrez database.

- **ESpell** (spelling suggestions)

  Retrieves spelling suggestions for a text query in a given database.

- **ECitMatch** (batch citation searching in PubMed)

  Retrieves PubMed IDs (PMIDs) corresponding to a set of input citation strings.

Table 1 presents the databases on which these applications work with their Entrez database name and UID (unique identifier) common name as listed in the documentation[1]. The Entrez name corresponds to the names found in the “All Databases” drop-down list to the left of the Search open-text bar on the main webpage (https://www.ncbi.nlm.nih.gov/), which can be used to explore and learn or confirm the results of using E-Utilities.

NCBI Datasets command line tools datasets and dataformat can be accessed within SAS using the PIPE engine to the FILENAME statement. These tools add immense versatility to SAS programs, especially those that require the large data of genomic investigations into SARS-CoV-2.

The ease of Validating a SAS System® program with its (dated) stored log and the ease of updating or replicating a search are important reasons to approach such tasks programmatically. The SAS System® provides many tools and appears to be fast enough to be practical for the tasks described. The **goal** of this paper is to introduce the E-utilities and describe a SAS macro that uses the HTTP procedure using the E-utilities URL and one that uses the NCBI command line tools datasets and dataformat.

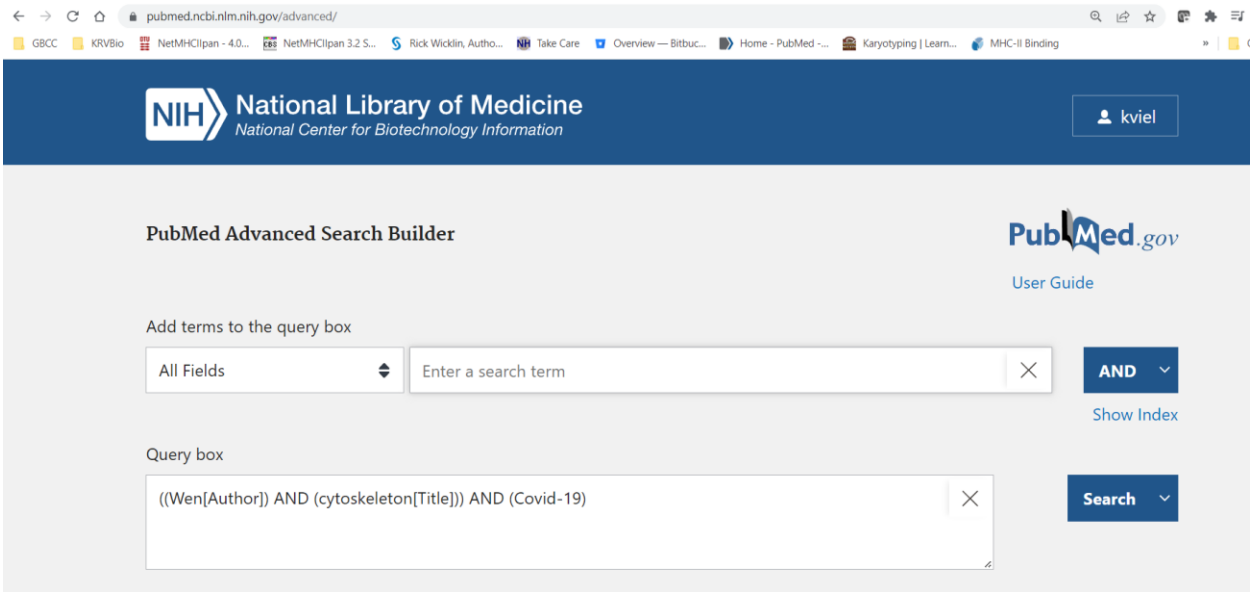
Table 1 The E-utility Database Names and their respective Entrez Databases and UID common names.

E-utility Database Name	Entrez Database	UID common name
bioproject	BioProject	BioProject ID
biosample	BioSample	BioSample ID
biosystems	Biosystems	BSID
books	Books	Book ID
cdd	Conserved Domains	PSSM-ID
gap	dbGaP	dbGaP ID
dbvar	dbVar	dbVar ID
epigenomics	Epigenomics	Epigenomics ID
nucest	EST	GI number
gene	Gene	Gene ID
genome	Genome	Genome ID
gds	GEO Datasets	GDS ID
geoprofiles	GEO Profiles	GEO ID
nucgss	GSS	GI number
homologene	HomoloGene	HomoloGene ID
mesh	MeSH	MeSH ID
toolkit	NCBI C++ Toolkit	Toolkit ID
ncbisearch	NCBI Web Site	Web Site ID
nlmcatalog	NLM Catalog	NLM Catalog ID
nuccore	Nucleotide	GI number
omia	OMIA	OMIA ID
popset	PopSet	PopSet ID
probe	Probe	Probe ID
protein	Protein	GI number
proteinclusters	Protein Clusters	Protein Cluster ID
pcassay	PubChem BioAssay	AID
pccompound	PubChem Compound	CID
pcsubstance	PubChem Substance	SID
Pubmed	PubMed	PMID
Pmc	PubMed Central	PMCID
Snp	SNP	rs number
Sra	SRA	SRA ID
structure	Structure	MMDB-ID
taxonomy	Taxonomy	TaxID
Unigene	UniGene	UniGene Cluster ID
Unists	UniSTS	STS ID

METHODS

WEB INTERFACES TO NCBI DATABASES

Even if the user is experienced using PubMed, referencing the "User Guide"[2] and "Entrez Help"[3] is highly recommended. When accessing Pubmed through its web interface, one can use the "Advanced" button to build a search with tags. In general, this is not need, but demonstrates the process, which is helpful for learning how to build the URL for E-Utilities. **Display 1** provides an example of a PubMed search built with the "Advanced" button.



Display 1 Building a search using the "Advanced" button on the web interface of PubMed.

The search built in Display 1 returns (as of April 12, 2022) the single paper as shown in **Display 2**.The author would have manually typed this search as "Wen[au] cytoskeleton[ti] Covid-19" into the web interface, noting that the quotation marks only delineate the search term in this paper and that they have the purpose of making their contents and exact match in PubMed. Further note that "AND" is implied, and parentheses as most SAS programmers and mathematicians might expect. Consider: "(cytoskeleton[ti] OR microtubule [ti]) Covid-19". The reader who is new to PubMed may wish to contrast the results of '"host cell receptor" [ti] Covid-19' and '"host cell" [ti] Covid-19' (without the single quotations marks when typed or copied to Pubmed.

Display 2 The results of the PubMed search built in Display 1.

E-UTILITES (ENTREZ PROGRAMMING UTILITIES)

The reader is encouraged to read or at least peruse "Entrez Programming Utilities Help"[3]. To obtain a list of databases, one can submit the following URL in a web browser:

```
https://eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi
```

Alternatively, calling the M_U_EUTILS SAS macro in **Appendix 1** with the following macro parameter and value invokes one of its two special uses (the other being HELP):

```
%m_u_eutils
   ( list_current_dbs = Y )
```

and generates the list of the available databases are in **Display 3** in the log (as of April 12, 2022). Note that the current list differs from this list provided in the documentation. The macro does not parse the code with the XML engine to the LIBNAME statement, but rather abstracts the names with a "quick and dirty" regular expression using the SAS PRX functions.

pubmed	clinvar	orgtrack
protein	gap	pmc
nuccore	gapplus	popset
ipg	grasp	proteinclusters
nucleotide	dbvar	pcassay
structure	gene	protfam
genome	gds	pccompound
annotinfo	geoprofiles	pcsubstance
assembly	homologene	seqannot
bioproject	medgen	snp
biosample	mesh	sra
blastdbinfo	ncbisearch	taxonomy
books	nlmcatalog	biocollections
cdd	omim	gtr

```
NOTE: 50 records were read from the infile RESP.
      The minimum record length was 0.
      The maximum record length was 126.
NOTE: DATA statement used (Total process time):
      real time           0.14 seconds
      cpu time            0.01 seconds
```

Display 3 The databases returned by the E-Utilities program EINFO as of April 22, 2022.

EXAMPLES

Using the HELP macro parameter, one can find some usage rules. To examine the URL created and exit the macro with no further action, one can use the macro parameters DEBUG_URL:

```
%m_u_eutils
   ( e_utilities               = einfo.fcgi
   , debug_url                 = Y
   )
```

which writes “https://eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi” to the log.

To examine the XML results without providing code for the CODE macro parameter, one can invoke the macro parameters DEBUG_RETURN:

```
%m_u_eutils
   ( e_utilities               = einfo.fcgi
   , debug_return              = Y
   ) ;
```

The following call obtains suggestions for spelling using the database pubmed:

```
%m_u_eutils
   ( e_utilities = espell.fcgi
   , db          = pubmed
   , term        = acetaminophin
   , api_key     =
   , code        = __rc1 = prxparse( "/<Query>(.+)<\/Query>/" ) %str(;)
                   __rc2 = prxparse( "/<CorrectedQuery>(.+)<\/CorrectedQuery>/" ) %str(;)
                   retain __rc %str(;)

                   file print %str(;)

                   length query
                          correctedquery  $ 200
                          %str(;)
                   if prxmatch( __rc1 , _infile_ )
                   then
                     do %str(;)
                        query = prxposn( __rc1
                                       , 1
                                       , _infile_
                                       ) %str(;)
                        put "Query= " @25 query %str(;)
                     end %str(;)
                   if prxmatch( __rc2 , _infile_ )
                   then
                     do %str(;)
                        correctedquery = prxposn( __rc2
                                                , 1
                                                , _infile_
                                                ) %str(;)
                        put  "CorrectedQuery=" @25 CorrectedQuery %str(;)
                     end %str(;)
   ) ;

Query=                  acetaminophin
CorrectedQuery=         acetaminophen
```

The call below replicates the search described in Display 2:

```
%m_u_eutils
   ( e_utilities               = esearch.fcgi
   , db                        = pubmed
   , term                      = Wen[Author]+AND+cytoskeleton[Title]+AND+Covid-19
   , api_key                   =
   , debug_return              = Y
   , code                      = __rc = prxparse( "/<Id>([a-z0-9]+)<\/Id>/" ) %str(;)
                                 retain __rc %str(;)

                                 file print %str(;)

                                 length pmid $ 20 %str(;)
                                 if prxmatch( __rc , _infile_ )
                                 then
                                   do %str(;)
                                      pmid = prxposn( __rc
                                                    , 1
                                                    , _infile_
                                                    ) %str(;)
                                      put pmid %str(;)
                                   end %str(;)

   ) ;
```

As of April 12, 2022, this code returns the single PubMed ID “32717049”. Note that providing code in such a way has some issues, such as the substitution of semi-colons with their masked versions “%str(;)”. The author removed his API Key, but had used his valid key.

To demonstrate the use the history server to replicate the search in the code above two calls to the macro were used. The second uses the QUERY_KEY and WEBENV values generated by the following call:

```
%symdel QueryKey_return
        webenv_return
        ;

%m_u_eutils
   ( e_utilities        = esearch.fcgi
   , db                 = pubmed
   , term               = Wen[Author]+AND+Covid-19
   , api_key            =
   , usehistory         = y
   , debug_return       = Y
   , code               = __rc1 = prxparse( "/<QueryKey>(.+)<\/QueryKey>/" ) %str(;)
                          __rc2 = prxparse( "/<WebEnv>(.+)<\/WebEnv>/" ) %str(;)
                          retain __rc1
                                 __rc2
                                 %str(;)

                          length querykey
                                 webenv   $ 200
                                 %str(;)

                          if prxmatch( __rc1 , _infile_ )
                          then
                            do %str(;)
                               querykey = prxposn( __rc1
                                                 , 1
                                                 , _infile_
                                                 ) %str(;)
                               call symputx( "querykey_return"
                                           , querykey
                                           ) %str(;)
                            end %str(;)

                            if prxmatch( __rc2 , _infile_ )
                            then
                              do %str(;)
                                 webenv = prxposn( __rc2
                                                 , 1
                                                 , _infile_
                                                 ) %str(;)
                                 call symputx( "webenv_return"
                                             , webenv
                                             ) %str(;)
                              end %str(;)
   ) ;
```

When the usehistory = y, the macro executes the %GLOBAL statement with for QUERYKEY_RETURN and WEBENV_RETURN. To demonstrate this, these global macro variables were deleted prior to call the macro. The code below runs a second search using these two macro variables and the result is the intersection of the two separate searches:

```
%m_u_eutils
   ( e_utilities               = esearch.fcgi
   , db                        = pubmed
   , term                      = cytoskeleton[Title]
   , query_key                 = &QueryKey_return.
   , webenv                    = &webenv_return.
   , api_key                   =
   , usehistory                = y
   , code                      = __rc = prxparse( "/<Id>([a-z0-9]+)<\/Id>/" ) %str(;)
                                 retain __rc %str(;)

                                 file print %str(;)

                                 length pmid $ 20 %str(;)
                                 if prxmatch( __rc , _infile_ )
                                 then
                                   do %str(;)
                                      pmid = prxposn( __rc
                                                    , 1
                                                    , _infile_
                                                    ) %str(;)
                                      put pmid %str(;)
                                   end %str(;)
   ) ;
```

As of April 12, 2022, this code returns the single PubMed ID “32717049”, matching the first example and the search using the PubMed web interface.

A frequent task in clinical trials programming is to convert lab test from original to prefer units. Obtaining the list manually is cumbersome and may result in errors. Using the pccompound database might help, but the user may also need to familiarize her- or himself with this resource. The following code, consisting of two calls to the macro, demonstrates an attempt to find the molecular mass of acetaminophen:

```
%m_u_eutils
   ( e_utilities  = esearch.fcgi
   , db           = pccompound
   , term         = acetaminophen
   , api_key      =
   , usehistory   = y
   , debug_return = Y
   , code         = __rc1 = prxparse( "/<QueryKey>(.+)<\/QueryKey>/" ) %str(;)
                    __rc2 = prxparse( "/<WebEnv>(.+)<\/WebEnv>/" ) %str(;)
                    retain __rc1
                           __rc2
                           %str(;)

                    length querykey
                           webenv   $ 200
                           %str(;)

                    if prxmatch( __rc1 , _infile_ )
                    then
                      do %str(;)
                         querykey = prxposn( __rc1
                                           , 1
                                           , _infile_
                                           ) %str(;)
                         call symputx( "querykey_return"
                                     , querykey
                                     ) %str(;)
                      end %str(;)

                    if prxmatch( __rc2 , _infile_ )
                    then
                      do %str(;)
                         webenv = prxposn( __rc2
                                         , 1
```

```
                                        , _infile_
                                        ) %str(;)
                            call symputx( "webenv_return"
                                        , webenv
                                        ) %str(;)
                          end %str(;)
   ) ;

%m_u_eutils
   ( e_utilities  = esummary.fcgi
   , db           = pccompound
   , query_key    = &QueryKey_return.
   , webenv       = &webenv_return.
   , api_key      =
   , debug_return = Y
   ) ;
```

The record returns includes the following, which the user may parse or attempt to refine using options:

```
<Item Name="IUPACName" Type="String">N-(4-hydroxyphenyl)acetamide;methanol</Item>
<Item Name="string" Type="String">acetaminophen:methanol solvate</Item>
<Item Name="string" Type="String">paracetamol:methanol solvate, (1:1)</Item>
<Item Name="MolecularWeight" Type="String">549.600</Item>
```

NCBI COMMAND LINE TOOLS

Certain information for the datasets and dataformat tools is available from the tools themselves, but the interested reader should at least peruse the reference[4]. The following code demonstrates how to obtain help from the tool using the macro presented in **Appendix 2**:

```
%m_ncbi_cli_tool
   ( cli_tool              = dataformat
   , cli_tool_path         = ~\dataformat.exe
   , options               = help tsv genome
   ) ;
```

The user will have to provide the path. Note that one can build the options incrementally, for instance, the author learned of the “genome” Report Command to the tsv command by running the code above without “genome” then adding it.

To code below mimics the example:

```
%m_ncbi_cli_tool
   ( options              = %str(download gene gene-id 1,2,3,9,10,11,12,13,14,15,16,17 --filename cli_tools_test.zip)
   , api_key              =
   , pre                  = cd /D C:\Users\kviel\OneDrive\Documents\12_NIH\NCBI\NLM\nuccore\SARS_CoV_2\datasets %nrstr(&&)
   ) ;
```

Note that the PRE macro parameter changed the directory using the MS-DOS command cd before the executable on the command line. The ampersand is the line continuation in the MS-DOS command line but it needed to be masked. The zip file was 143 KB and unzipped to 474 KB and seven files.

The format of the reports is jsonl, but the dataformat tool provides a way to convert it. For example, the following code:

```
%m_ncbi_cli_tool
   ( cli_tool      = dataformat
   , cli_tool_path = ~\dataformat.exe
   , pre           = cd /D C:\Users\kviel\OneDrive\Documents\12_NIH\NCBI\NLM\nuccore\SARS_CoV_2\datasets\cli_tools_test
                     %nrstr(&&)
   , options       = tsv gene  --inputfile ncbi_dataset/data/data_report.jsonl
   ) ;
```

results in following snippet being written to the log:

```
Annotation Assemblies in Scope Accession     Annotation Assemblies in Scope Name   Annotation
Release Date   Annotation Release Name       Chromosomes    Common Name    Description    Ensembl
GeneIDs NCBI GeneID    Gene Type      Genomic Range Sequence Accession     Genomic Range
Orientatio
n       Genomic Range Start    Genomic Range Stop     Genomic Region Gene Range Sequence Accession
        Genomic Region Gene Range Orientation Genomic Region Gene Range Start       Genomic Region
Gene Range Stop        Genomic Region Genomic Region Type    Nomenclature Authority Nomenclatur
e ID    OMIM IDs       Orientation    Protein Accession      Protein Ensembl Protein       Protein
Isoform Protein Length Protein Mature Peptide Accession      Protein Mature Peptide Length Protein
Mature Peptide Name    Protein Name   Reference Standard Gene Range Sequence Accession     Refe
rence Standard Gene Range Orientation Reference Standard Gene Range Start   Reference Standard
Gene Range Stop        Reference Standard Genomic Region Type       Replaced NCBI GeneID   RNA
Type    SwissProt Accessions   Symbol  Synonyms       Taxonomic ID   Taxonomic Name Transcript Ac
cession Transcript CDS Sequence Accession     Transcript CDS Orientation    Transcript CDS Start
        Transcript CDS Stop    Transcript Ensembl Transcript Transcript Genomic Accession
        Transcript Genomic Exons Orientation  Transcript Genomic Exons Start       Transcript
Genomic
 Exons Stop    Transcript Genomic Orientation       Transcript Genomic Start      Transcript
Genomic Stop   Transcript Genomic Seq Name   Transcript Transcript Length  Transcript Transcript
Name    Transcript Protein Accession  Transcript Protein Ensembl Protein    Transcript Prote
in Isoform     Transcript Protein Length     Transcript Protein Mature Peptide Accession
        Transcript Protein Mature Peptide Length     Transcript Protein Mature Peptide Name
        Transcript Protein Name       Transcript Type
GCF_000001405.39       GRCh38.p13     2021-11-19     NCBI Homo sapiens Updated Annotation Release
109.20211119   19      human   alpha-1-B glycoprotein ENSG00000121410        1       PROTEIN_CODING
        NC_000019.10   minus   58345183       58353492
        HGNC    HGNC:5  138670  minus
                                                      P04217  A1BG
A1B,ABG,GAB,HYST2477   9606    Homo sapiens   NM_130786.4    NM_130786.4            56      1543
        ENST00000263100.8      NC_000019.10           58353404       58353492       minus
        58345183       58353492       Chromosome 19 Reference GRCh38.p13 Primary Assembly  3382
        NP_570602.2    ENSP00000263100.2              495                            alpha-1B-glyco
protein precursor      PROTEIN_CODING
GCF_000001405.39       GRCh38.p13     2021-11-19     NCBI Homo sapiens Updated Annotation Release
109.20211119   19      human   alpha-1-B glycoprotein ENSG00000121410        1       PROTEIN_CODING
        NC_000019.10   minus   58345183       58353492
        HGNC    HGNC:5  138670  minus
                                                      P04217  A1BG
A1B,ABG,GAB,HYST2477   9606    Homo sapiens   NM_130786.4    NM_130786.4            56      1543
        ENST00000263100.8      NC_000019.10           58353292       58353327       minus
        58345183       58353492       Chromosome 19 Reference GRCh38.p13 Primary Assembly  3382
        NP_570602.2    ENSP00000263100.2              495                            alpha-1B-glyco
protein precursor      PROTEIN_CODING
GCF_000001405.39       GRCh38.p13     2021-11-19     NCBI Homo sapiens Updated Annotation Release
109.20211119   19      human   alpha-1-B glycoprotein ENSG00000121410        1       PROTEIN_CODING
        NC_000019.10   minus   58345183       58353492
        HGNC    HGNC:5  138670  minus
                                                      P04217  A1BG
A1B,ABG,GAB,HYST2477   9606    Homo sapiens   NM_130786.4    NM_130786.4            56      1543
        ENST00000263100.8      NC_000019.10           58352928       58353197       minus
        58345183       58353492       Chromosome 19 Reference GRCh38.p13 Primary Assembly  3382
        NP_570602.2    ENSP00000263100.2              495                            alpha-1B-glyco
protein precursor      PROTEIN_CODING
```

The NCBI has made an incredible resource freely available which is indispensable to researchers investigation the SARS-CoV-2 pandemic. The following code will download every DNA (RNA) sequence in the database:

```
%m_ncbi_cli_tool
   ( options    = download virus genome taxon sars-cov-2 --exclude-cds --exclude-protein --exclude-gpff --exclude-pdb --filename
                  sar2_genomes_20220316.zip
   , api_key    =
   , pre        = cd /D D: %nrstr(&&)
   ) ;
```

Note that the first line wraps due to the size of the page, but it appears on one line in the SAS Enhanced Editor. The author indented only for readability. One may wish to add a line continuation if in doubt. This process requires substantial time and the author left it as an overnight process. The data far exceeded the 1 TB hard drive, so the author bought a Dell WD 4TB USB 3.0 WD My Book desktop external hard drive, which seems to work without meaningful issues.

CONCLUSION

This paper introduced the Entrez databases that the National Center for Biotechnology Information (NCBI) of the United States National Institute of Health (NIH) has made freely available and their programming interfaces that were incorporated into two SAS System® macros. The author provided a few examples of calls of the macros and briefly discussed some issues and results. The ultimate uses will be highly individualized or proprietary, so general and in-depth discussions are not immediately possible.

Certain issues concerning design need to be addressed. XML, for instance, should not be parsed as text. The DTD of the output should be stable enough to be worth the effort to use instead. Two calls to the M_U_EUTILS macro could be avoided with a design change and the names of the global macro variables can be prefixed with an incremented number so that the value is not overwritten or the need to capture it in another global macro variable after the call is not required, for instance, by checking the global macro table.

Programmatic approaches to such work have numerous advantages and the learning curve is not steep. Searches of a great number of items can be fast, but the macro has does not control the number of submissions, which can create an issue with NCBI. Contacting the help desk, which in the experience of the author has been quite responsive and helpful, is wise, especially if one might be submitting more than three submissions per second. SAS programs and their stored (permanent) logs document work and one may update search terms based on the results of executing programs.

REFERENCES

[1] Entrez Programming Utilities Help [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2010-.

[2] National Library of Medicine, National Center for Biotechnology Information. 2022. “PubMed User Guide.” Accessed April 12, 2022. https://pubmed.ncbi.nlm.nih.gov/help/#pubmedhelp.Search_Field_Descrip.

[3] Entrez Help [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2005-. Entrez Help. 2006 Jan 20 [Updated 2016 May 31].

[4] National Library of Medicine, National Center for Biotechnology Information. 2022. “Command line tool reference.” https://www.ncbi.nlm.nih.gov/datasets/docs/v1/reference-docs/command-line/

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Kevin R. Viel, Ph.D.

Histonis, Incorporated
kviel@histonis.org

Navitas Data Sciences
kevin.viel@navitaslifesciences.com
www.navitasdatasciences.com

APPENDIX

Appendix 1 The M_U_EUTILS macro.

```
%macro m_u_eutils
   ( e_utilities                  = %str()
   , db                           = %str()
   , term                         = %str()
   , extra_option                 = %str()
   , usehistory                   = %str()
   , api_key                      = %str()
   , query_key                    = %str()
   , webenv                       = %str()

   , results_out                  = %sysfunc( pathname( work ))\results_out.xml
   , out_ds                       = %str()
   , code                         = %str()

   , list_current_dbs             = N

   , debug_url                    = N
   , debug_return                 = N

   /*************/
   , help                         = N
   ) ;

 %if &help. = Y
 %then
    %do ;
        %let mprint_orig = %sysfunc(getoption(mprint)) ;
        options nomprint ;

        skip ;
        skip ;
        %put ____________________________________________________________________________________________________________________________________________________;
        %put Purpose of program:                This utility macro allows the user to submit NCBI Entrez Programming Utilities (E-utilities)                 ;
        %put %str(                                   )See Entrez Programming Utilities Help                                                                ;
        %put %str(                                   )Last Updated: January 3, 2022                                                                        ;
        skip ;
        %put %str(                                   )The Entrez Programming Utilities (E-utilities) are a set of eight server-side programs that provide a ;
        %put %str(                                   )stable interface into the Entrez query and database system at the National Center for Biotechnology  ;
        %put %str(                                   )Information (NCBI). The E-utilities use a fixed URL syntax that translates a standard set of input   ;
        %put %str(                                   )parameters into the values necessary for various NCBI softwaresoftwaresoftwarecomponents to search for ;
        %put %str(                                   )and retrieve the requested data. The E-utilities are therefore the structured interface to the Entrez ;
        %put %str(                                   )system, which currently includes 38 databases covering a variety of biomedical data, including      ;
        %put %str(                                   )nucleotide and protein sequences, gene records, three-dimensional molecular structures, and the     ;
        %put %str(                                   )biomedical literature.                                                                               ;
        skip ;
        %put Macro Parameter                     Description                                                                                               ;
        %put _______________                     ___________________________________________________________________________________________________________;
        %put e_utilities                       = Name of E-utility to use.                                                                                 ;
        skip ;
        %put %str(                                   )EInfo (database statistics)                                                                          ;
        %put %str(                                      )Provides the number of records indexed in each field of a given database, the date of the last update ;
        %put %str(                                      )of the database, and the available links from the database to other Entrez databases.             ;
        skip ;
        %put %str(                                   )ESearch (text searches)                                                                              ;
        %put %str(                                      )Responds to a text query with the list of matching UIDs in a given database (for later use in     ;
        %put %str(                                      )ESummary, EFetch or ELink), along with the term translations of the query.                        ;
        skip ;
        %put %str(                                   )EPost (UID uploads)                                                                                  ;
        %put %str(                                      )Accepts a list of UIDs from a given database, stores the set on the History Server, and responds with ;
        %put %str(                                      )a query key and web environment for the uploaded dataset.                                         ;
        skip ;
        %put %str(                                   )ESummary (document summary downloads)                                                                ;
        %put %str(                                      )Responds to a list of UIDs from a given database with the corresponding document summaries.       ;
        skip ;
        %put %str(                                   )EFetch (data record downloads)                                                                       ;
        %put %str(                                      )Responds to a list of UIDs in a given database with the corresponding data records in a specified ;
        %put %str(                                      )format.                                                                                           ;
        skip ;
        %put %str(                                   )ELink (Entrez links)                                                                                 ;
        %put %str(                                      )Responds to a list of UIDs in a given database with either a list of related UIDs (and relevancy ;
        %put %str(                                      )scores) in the same database or a list of linked UIDs in another Entrez database %str(;) checks for ;
        %put %str(                                      )the existence of a specifiedspecified link from a list of one or more UIDs %str(;) creates a      ;
        %put %str(                                      )hyperlink to the primary LinkOut provider for a specificspecific UID and database, or lists LinkOut ;
        %put %str(                                      )URLs and attributes for multiple UIDs.                                                            ;
        skip ;
        %put %str(                                   )EGQuery (global query)                                                                               ;
        %put %str(                                      )Responds to a text query with the number of records matching the query in each Entrez database.   ;
        skip ;
        %put %str(                                   )ESpell (spelling suggestions)                                                                        ;
        %put %str(                                      )Retrieves spelling suggestions for a text query in a given database.                              ;
        skip ;
        %put %str(                                   )ECitMatch (batch citation searching in PubMed)                                                       ;
        %put %str(                                      )Retrieves PubMed IDs (PMIDs) corresponding to a set of input citation strings.                    ;
        skip ;
        skip ;

        %put %str(                                   )Handling Special Characters Within URLs                                                              ;
        %put %str(                                   )When constructing URLs for the E-utilities, please use lowercase characters for all parameters except ;
        %put %str(                                   )%nrstr(&)WebEnv. There is no required order for the URL parameters in an E-utility URL, and null values ;
        %put %str(                                   )or inappropriate parameters are generally ignored. Avoid placing spaces in the URLs, particularly in ;
        %put %str(                                   )queries. If a space is required, use a plus sign (+) instead of a space:                             ;
        skip ;
        %put %str(                                   )Incorrect: %nrstr(&)id=352, 25125, 234                                                               ;
        %put %str(                                   )Correct:   %nrstr(&)id=352,25125,234                                                                 ;
        skip ;
        %put %str(                                   )Incorrect: %nrstr(&)term=biomol mrna[properties] AND mouse[organism]                                 ;
        %put %str(                                   )Correct: %nrstr(&)term=biomol+mrna[properties]+AND+mouse[organism]                                   ;
        skip ;
        %put %str(                                   )Other special characters, such as quotation marks (%str(")) or the # symbol used in referring to a  ;
        %put %str(                                   )query key on the History server, should be represented by their URL encodings (%str(%%)22 for %str(%");) ;
        %put %str(                                   )%str(%%)23 for #).                                                                                   ;
        skip ;
        %put %str(                                   )Incorrect: %nrstr(&)term=#2+AND+"gene in genomic"[properties]                                        ;
```

```
104         %put %str(                                  )Correct: %nrstr(&)term=%str(%%)232+AND+%str(%%)22gene+in+genomic%str(%%)22[properties]                    ;
105
106         skip ;
107         %put NOTE that not every tool can use the following and each may have a database-specific use.                                                    ;
108         %put db                                 = The name of the database to search.                                                                    ;
109         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
110         %put term                               = The term to search (see above for usage rules).                                                        ;
111         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
112         %put extra_option                       = Any free text to include in the URL, such as a database-specific option.                               ;
113         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
114         %put usehistory                         = y/n indicator to interact with the History server.                                                     ;
115         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
116         %put query_key                          = The value return in the QueryKey field. Requires usehistory = y.                                       ;
117         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
118         %put webenv                             = The value return in the WebEnv field. Requires usehistory = y.                                         ;
119         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
120         %put results_out                        = The path and filename of the physical file produced by the HTTP procedure.                             ;
121         %put %str(                                    )In general the results will be xml or json.                                                   ;
122         %put %str(                                      )Default: %nrstr(%%)sysfunc( pathname( work ))\output.xml                                    ;
123         %put out_ds                             = The name of the output data set resulting from INFILEing the FILEREF for results_out.                  ;
124         %put %str(                                    )If null, the default, then _NULL_ is used.                                                    ;
125         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
126         %put code                               = Custom code placed at the end of the data step that reads the results in results_out.                  ;
127         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
128
129         skip ;
130         %put list_current_dbs                   = y/n indicator to list the currently available databases and exit the macro                              ;
131         %put %str(                                      )Default: N                                                                                  ;
132
133         %put debug_url                          = y/n indicator to write the URL to the log.                                                             ;
134         %put %str(                                      )Default: N                                                                                  ;
135         %put debug_return                       = y/n indicator to write the INFILEd results to the log.                                                 ;
136         %put %str(                                      )Default: N                                                                                  ;
137
138         skip ;
139         %put api_key                            = On December 1%str(,) 2018%str(,) NCBI will begin enforcing the use of API keys that will offer enhanced levels of ;
140         %put %str(                                    )supported access to the E-utilities. After that date%str(,) any site (IP address) posting more than 3      ;
141         %put %str(                                    )requests per second to the E-utilities without an API key will receive an error message. By including      ;
142         %put %str(                                    )an API key%str(,) a site can post up to 10 requests per second by default. Higher rates are available      ;
143         %put %str(                                    )by request (eutilities@ncbi.nlm.nih.gov). Users can obtain an API key now from the Settings page of        ;
144         %put %str(                                    )their NCBI account (to create an account%str(,) visit http://www.ncbi.nlm.nih.gov/account/). After        ;
145         %put %str(                                    )creating the key%str(,) users should include it in each E-utility request by assigning it to the new      ;
146         %put %str(                                    )api_key parameter.                                                                            ;
147         %put %str(                                    )See:  https://support.nlm.nih.gov/knowledgebase/article/KA-05317/en-us                        ;
148         %put %str(                                      )Default: %nrstr(%%)str%str(())                                                              ;
149
150         options &mprint_orig. ;
151
152         %goto __END ;
153      %end ;
154
155   /************/
156   filename resp
157            "%sysfunc( pathname( work ))/work_tmp.xml"
158            ;
159
160   proc http
161      url    = "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi"
162      out    = resp
163      method = "GET"
164      ;
165   run ;
166
167   data %if &list_current_dbs. ne Y
168        %then __current_dbs
169                  ( keep = eutility_database_name )
170        ;
171         %else _null_ ;
172        ;
173     infile resp ;
174     input ;
175     length eutility_database_name $ 50 ;
176     __rc = prxparse( "/<DbName>([a-z0-9]+)<\/DbName>/" ) ;
177     retain __rc ;
178
179     if prxmatch( __rc , _infile_ )
180     then
181       do ;
182          eutility_database_name = prxposn( __rc
183                                          , 1
184                                          , _infile_
185                                          ) ;
186          output ;
187          %if &list_current_dbs. = Y
188          %then put eutility_database_name %str(;) ;
189       end ;
190   run ;
191
192   filename resp ;
193
194   /* Delete the work xml file */
195   %let __rc = %sysfunc( filename ( temp
196                                  , %sysfunc( pathname( work ))/work_tmp.xml
197                                  )
198                       ) ;
199   %let __rc = %sysfunc( fdelete( &temp. )) ;
```

```
200
201    %if &list_current_dbs. = Y
202    %then
203       %do ;
204           %goto __END ;
205       %end ;
206
207    /************/
208    %if %sysfunc( prxmatch(
209  /^(?:einfo.fcgi|esearch.fcgi|epost.fcgi|esummary.fcgi|efetch.fcgi|elink.fcgi|egquery.fcgi|es
210  pell.fcgi|ecitmatch.cgi)$/i , &e_utilities. )) = 0
211    %then
212       %do  ;
213           %put E%str(RROR: ) E_UTILITIES = &e_utilities. must be one of: ;
214           %put E%str(RROR: ) einfo.fcgi    ;
215           %put E%str(RROR: ) esearch.fcgi  ;
216           %put E%str(RROR: ) epost.fcgi    ;
217           %put E%str(RROR: ) esummary.fcgi ;
218           %put E%str(RROR: ) efetch.fcgi   ;
219           %put E%str(RROR: ) elink.fcgi    ;
220           %put E%str(RROR: ) egquery.fcgi  ;
221           %put E%str(RROR: ) espell.fcgi   ;
222           %put E%str(RROR: ) ecitmatch.cgi ;
223
224           %goto __END ;
225       %end ;
226
227    /***********/
228    %if &db. ne %str()
229    %then
230       %do  ;
231            proc sql
232               noprint ;
233              select eutility_database_name
234              from __current_dbs
235              where eutility_database_name = "&db."
236              ;
237            quit ;
238
239            %if &sqlobs. = 0
240            %then
241               %do ;
242                   %put E%str(RROR: ) DB = &db. must be one of (case specific): ;
243
244                   data _null_ ;
245                     set __current_dbs ;
246                     put "E" "RROR: " eutility_database_name ;
247                   run ;
248
249                   %goto __END ;
250               %end ;
251       %end ;
252
253    /*********************************************************************/
254    %let url = https://eutils.ncbi.nlm.nih.gov/entrez/eutils/&e_utilities. ;
255
256    %if %nrbquote(&db.) ne %str()
257    %then %let url = &url.?db=&db. ;
258
259    %if %nrbquote(&term.) ne %str()
260    %then %let url = &url.%nrstr(&)term=&term. ;
261
262    %if %nrbquote(&query_key.) ne %str()
263    %then %let url = &url.%nrstr(&)query_key=&query_key. ;
264
265    %if %nrbquote(&webenv.) ne %str()
266    %then %let url = &url.%nrstr(&)WebEnv=&webenv. ;
267
268    %if %nrbquote(&usehistory.)  ne %str()
269    %then
270       %do ;
```

```
271          %global querykey_return
272                  webenv_return
273                  ;
274          %let url = &url.%nrstr(&)usehistory=&usehistory. ;
275       %end ;
276
277    %if %nrbquote(&api_key.) ne %str()
278    %then %let url = &url.%nrstr(&)api_key=&api_key. ;
279
280    %if &debug_url. = Y
281    %then
282       %do ;
283           %put &url. ;
284           %goto __END ;
285       %end ;
286
287    /***************/
288    filename resp
289             "&results_out."
290             ;
291
292    proc http
293       url    = %unquote(%str(%')&url.%str(%'))
294       out    = resp
295       method = "GET"
296       ;
297    run ;
298
299    data %if %nrbquote(&out_ds.) = %str() %then _null_ ;
300          %else &out_ds. ;
301         ;
302
303      infile resp ;
304      input ;
305
306      %if &debug_return. = Y %then put _infile_ %str(;) ;
307
308      &code.
309
310    run ;
311
312    filename resp ;
313
314    %__END:
315
316    skip ;
317    %put ********************************************************************************************************************************************** ;
318    %put Please see: https://www.ncbi.nlm.nih.gov/home/about/policies/                                                                                    ;
319    skip ;
320    %put Disclaimer and Copyright Issues                                                                                                                  ;
321    skip ;
322
323    %put If you use the E-utilities within software, NCBI%str(%')s Disclaimer and Copyright notice                                                       ;
324    %put (https://www.ncbi.nlm.nih.gov/About/disclaimer.html) must be evident to users of your                                                           ;
325    %put product. Please note that abstracts in PubMed may incorporate material that may be                                                              ;
326    %put protected by U.S. and foreign copyright laws. All persons reproducing, redistributing,                                                          ;
327    %put or making commercial use of this information are expected to adhere to the terms and                                                            ;
328    %put conditions asserted by the copyright holder. Transmission or reproduction of protected                                                          ;
329    %put items beyond that allowed by fair use (PDF) as defined in the copyright laws requires                                                           ;
330    %put the written permission of the copyright owners.                                                                                                 ;
331    skip ;
332    %put ********************************************************************************************************************************************** ;
333    skip ;
334    %put _____________________________________________________________________________________________________________________________________ ;
335    %put End of help                                                                                                                                      ;
336
337 %mend  m_u_eutils  ;
```

Appendix 2 The M_NCBI_CLI_TOOL macro.

```
1  %macro m_ncbi_cli_tool
2     ( cli_tool          = datasets
3     , cli_tool_path     = ~\datasets.exe
4     , api_key           = %str()
5     , cli_tool_help     = N
6     , options           = %str()
7     , pre               = %str()
8     , records_to_read   = %str()
9     /*************/
10    , help              = N
```

```
    ) ;

  %if &help. = Y
  %then
     %do ;
         %let mprint_orig = %sysfunc(getoption(mprint)) ;
         options nomprint ;

         skip ;
         skip ;
         %put ____________________________________________________________________________________________________________________________________________;
         %put Purpose of program:  This macro runs the NCBI Datasets Command line interface (CLI) tool reference                                          ;
         %put %str(                )The NCBI Datasets datasets command line tools include datasets and dataformat. They can be used to          ;
         %put %str(                )download and convert metadata into tabular format.                                                            ;
         %put %str(                )See: https://www.ncbi.nlm.nih.gov/datasets/docs/v1/reference-docs/command-line/                               ;
         skip ;
         %put Macro Parameter      Description                                                                                                      ;
         %put ____________________ _________________________________________________________________________________________________________ ;
         %put cli_tool            = The CLI tool to use: datasets or dataformat.                                                                   ;
         %put %str(                )Default: datasets                                                                                             ;
         %put cli_tool_path       = The path to executable (.exe) files.                                                                           ;
         %put %str(                )Default: %nrstr(%%)str%str(())                                                                                ;
         %put api_key             = NCBI API key for the user.                                                                                     ;
         %put %str(                )See:  https://support.nlm.nih.gov/knowledgebase/article/KA-05317/en-us                                        ;
         %put %str(                )Default: %nrstr(%%)str%str(())                                                                                ;
         %put cli_tool_help       = Whether to display the tool help, this is not the macro help.                                                  ;
         %put options             = Free-text options to add to the options of the command line.                                                   ;
         %put %str(                )Default: %nrstr(%%)str%str(())                                                                                ;
         %put pre                 = Free-text command line to add before the tool. For instance, this can be used to change directory.             ;
         %put %str(                )Default: %nrstr(%%)str%str(())                                                                                ;
         %put records_to_read     = A number value of the number of records to read in the pipe results.                                           ;
         %put %str(                )Default: %nrstr(%%)str%str(())                                                                                ;

         skip ;
         skip ;
         %put ********************************************************************************************************************************************** ;
         skip ;
         %put ____________________________________________________________________________________________________________________________________________ ;
         %put End of help                                                                                                                                   ;

         options &mprint_orig. ;

         %goto __END ;
      %end ;

  %if &cli_tool_help. = Y
  %then
     %do ;
         filename cli_tool
                  pipe
                  %unquote(%str(%')"&cli_tool_path." -h%str(%'))
                  ;

         data _null_ ;
           infile cli_tool ;
           input ;
           put _infile_ ;
         run ;

         filename cli_tool ;

         %goto __END ;
     %end ;

  /****************/
  %if %nrbquote(&api_key.) ne %str() %then %let options = &options --api-key &api_key. ;

  /****************/
  filename cli_tool
           pipe
           %unquote(%str(%') &pre.  "&cli_tool_path." &options. %str(%'))
           ;

  data _null_ ;
     infile cli_tool ;
     input ;
     put _infile_ ;

     %if     %sysfunc( prxmatch( /^\d+$/ , %nrbquote(&records_to_read.)))
         and &records_to_read. > 0
     %then if _n_ > &records_to_read. then stop %str(;) ;

  run ;

  filename cli_tool ;


  %__END:

%mend  m_ncbi_cli_tool ;
```