ENTREZ DIRECT - QUICK TOUR

Entrez Direct (EDirect) is an advanced method for accessing the NCBI's set of interconnected data domains (publication, nucleotide, protein, structure, variation, expression, etc.) from a terminal window. Multi-step queries can be built incrementally, and functions take search terms from command-line arguments.

ENTERING MULTI-STEP QUERIES

In order to perform an Entrez search, the user need only provide the name of the database and the desired query string. For example:

```
esearch -db pubmed -query "capsaicin cancer pain management"
```

constructs the appropriate Entrez Utilities (EUtils) URL from the query terms and executes the search. EDirect handles many technical details behind the scenes.

The vertical bar ("|") UNIX pipe character is used to send the results to the next step. Piping the esearch output to an elink command:

```
esearch -db pubmed -query "capsaicin cancer pain management" | elink -related
```

will look up related articles (precomputed PubMed neighbors) of the initial results. (Using elink -target retrieves associated records between databases.)

The same set of commands can be written on multiple lines with the backslash ("\") UNIX line continuation character, for improved readability and easier editing:

```
esearch -db pubmed -query "capsaicin cancer pain management" | \
elink -related
```

The list of neighbors can be refined by further term searching in Entrez:

```
esearch -db pubmed -query "capsaicin cancer pain management" | \
elink -related | \
efilter -query "conotoxin NOT morphine"
```

QUALIFYING QUERIES BY INDEXED FIELD

Terms can be qualified by an indexed field abbreviation (e.g., [AUTH], [JOUR], [MESH], [TITL]). Boolean operators and parentheses can also be used in the query expression for more complex searches. For example:

```
"Tager H [AUTH] AND glucagon [TIAB]"
```

in pubmed or:

```
"alcohol dehydrogenase [PROT] NOT (bacteria [ORGN] OR fungi [ORGN])"
```

in protein. The scripting example at the end of this document will produce a report giving the entire set of indexed fields for every Entrez database.

EXAMINING INTERMEDIATE RESULTS

A small structured XML object is passed between each command. The Count field gives the number of records returned by the previous step. A good measure of query success is a reasonable (non-zero) count value:

```
  <ENTREZ_DIRECT>
    <Db>pubmed</Db>
    <WebEnv>NCID_1_4586177_172.16.22.25_5555_1366675055_388385336</WebEnv>
    <QueryKey>4</QueryKey>
    <Count>10</Count>
    <Step>3</Step>
  </ENTREZ_DIRECT>
```

Checking the result count after each step can help avoid unsuccessful queries.

READING PUBMED REPORTS

Record retrieval or formatting is separate from searching. Piping to efetch:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  efetch -format abstract
```

returns a report that can be read by a person:

```
  1. PLoS One. 2013;8(3):e59293. doi: 10.1371/journal.pone.0059293. Epub ...

  Expression and Pharmacology of Endogenous Cav Channels in SH-SY5Y Human
  Neuroblastoma Cells.

  Sousa SR, Vetter I, Ragnarsson L, Lewis RJ.

  Institute for Molecular Bioscience, The University of Queensland, St. Lucia,
  Australia.

  SH-SY5Y human neuroblastoma cells provide a useful in vitro model to study the
  mechanisms underlying neurotransmission and nociception. These cells are derived
  from human sympathetic neuronal tissue and thus, express a number of the Cav
  channel subtypes essential for regulation of important physiological functions,
  ...
```

XML DOCUMENT SUMMARIES

EDirect also provides document summaries and other result types that are returned in structured XML format. For example:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  esummary
```

will generate an XML document summary set:

```
  <eSummaryResult>
    <DocumentSummarySet status="OK">
      <DocumentSummary uid="23536870">
        <Id>23536870</Id>
        <PubDate>2013</PubDate>
        <EPubDate>2013 Mar 25</EPubDate>
        <Source>PLoS One</Source>
        <Authors>
```

```
          <Author>
            <Name>Sousa SR</Name>
            <AuthType>Author</AuthType>
            <ClusterID>0</ClusterID>
          </Author>
          <Author>
          ...
```

The advantage of XML is that many pieces of information are in specific locations in a well-defined data hierarchy. Assembling individual units of data that are fielded by name, such as:

```
  <Source>PLoS One</Source>
  ...
  <Volume>8</Volume>
  <Issue>3</Issue>
  <Pages>e59293</Pages>
```

is much easier than parsing the units from a long, complex string:

```
  1. PLoS One. 2013;8(3):e59293 ...
```

The disadvantage of XML is that data extraction usually requires programming. But EDirect also addresses the problem of interpreting XML data, as discussed below.

CONVERSION OF XML DATA INTO TABULAR FORM

The xtract function uses command-line arguments to direct the selective conversion of XML data into a tab-delimited table. The -pattern argument divides the results into rows, while placement of data into columns is controlled by -element.

Additional arguments can limit data extraction to specified regions of the XML, filter by data content, and customize the table presentation. These will be introduced in the examples that follow.

Piping a document summary set to:

```
  xtract -outline
```

will give a simplified overview of XML structure hierarchy:

```
  DocumentSummarySet
    DocumentSummary
      Id
      PubDate
      EPubDate
      Source
      Authors
        Author
          Name
          AuthType
          ClusterID
        Author
        ...
      LastAuthor
      Title
      SortTitle
      Volume
      ...
```

The outline can help in deciding what arguments to send to xtract, so:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  esummary | \
  xtract -pattern DocumentSummary -element Id SortFirstAuthor Title
```

returns the PubMed identifier, first author name, and article title:

```
  23536870  Sousa SR  Expression and Pharmacology of Endogenous Cav Channels ...
  22410003  Vetter I  Characterisation of Na(v) types endogenously expressed ...
  18956616  Fürst Z   [Central and peripheral mechanisms in antinociception: ...
  12566085  Lo YK     Effect of arvanil (N-arachidonoyl-vanillyl-amine), a n ...
  ...
```

INTERACTION WITH UNIX UTILITIES

A tab-delimited table can be processed by many UNIX utiltiies. For example:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  esummary | \
  xtract -pattern DocumentSummary -element Id SortFirstAuthor Title | \
  sort -t $'\t' -k 2,2f -k 3,3f
```

sorts first by author name and then (for the same author) alphabetically by title:

```
  11000661  Chiang JS  New developments in cancer pain therapy.
  18956616  Fürst Z    [Central and peripheral mechanisms in antinociception ...
  10864900  Jerman JC  Characterization using FLIPR of rat vanilloid recepto ...
  12566085  Lo YK      Effect of arvanil (N-arachidonoyl-vanillyl-amine), a  ...
  ...
```

Or the title words can be extracted and sent through a series of UNIX commands:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  esummary | \
  xtract -pattern DocumentSummary -element Title | \
  sed 's/[^a-zA-Z0-9]/ /g' | tr 'A-Z' 'a-z' | xargs -n 1 | \
  sort | uniq -c | sort -k 1,1nr -k 2,2f
```

that take the article titles, remove punctuation, convert capital letters to lower case, and place each word on a separate line. The words are sorted alphabetically, occurrence counts are calculated, and the final results are sorted by frequency:

```
  8 of
  7 in
  4 cells
  4 human
  4 neuroblastoma
  4 sh
  4 sy5y
  ...
```

PUBMED ARTICLE XML RECORDS

The PubmedArticle has a more detailed structure than the document summary:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  efetch -format xml | \
  xtract -outline
```

and more information is fielded, including author names:

```
  PubmedArticleSet
    PubmedArticle
      MedlineCitation
        PMID
        DateCreated
          Year
          Month
          Day
        Article
          Journal
            ISSN
            JournalIssue
              Volume
              Issue
              PubDate
                Year
            Title
            ISOAbbreviation
          ArticleTitle
          Pagination
            MedlinePgn
          ELocationID
          Abstract
            AbstractText
          Affiliation
          AuthorList
            Author
              LastName
              ForeName
              Initials
            ...
```

Using this information to craft a new xtract statement:

```
  esearch -db pubmed -query "capsaicin cancer pain management" | \
  elink -related | \
  efilter -query "conotoxin NOT morphine" | \
  efetch -format xml | \
  xtract -pattern PubmedArticle -element "MedlineCitation/PMID" LastName
```

results in a table of all authors for each record:

```
  23536870  Sousa   Vetter  Ragnarsson  Lewis
  22410003  Vetter  Mozar   Durek       Wingerd  Alewood  Christie  Lewis
  18956616  Fürst
  12566085  Lo      Chiang  Wu
  ...
```

EXPLORATION OF SETS WITHIN XML

Individual PubmedArticles can be retrieved directly by efetch:

```
efetch -db pubmed -id 1937004 -format xml
```

and the XML may contain a list of Medical Subject Headings (MeSH Terms):

```
...
<MeshHeadingList>
  <MeshHeading>
    <DescriptorName>Adenosine Triphosphatases</DescriptorName>
  </MeshHeading>
  <MeshHeading>
    <DescriptorName>Amino Acid Sequence</DescriptorName>
  </MeshHeading>
  <MeshHeading>
    <DescriptorName>Base Sequence</DescriptorName>
  </MeshHeading>
  ...
```

Visiting each MeSH term with a -block statement, and customizing the output format (suppressing the trailing tab normally placed between columns and prefixing with a newline character):

```
efetch -db pubmed -id 1937004 -format xml | \
xtract -pattern PubmedArticle -tab "" -element "MedlineCitation/PMID" \
  -block MeshHeading -pfx "\n" -tab "" -element DescriptorName
```

produces a list of MeSH terms, one per line:

```
1937004
Adenosine Triphosphatases
Amino Acid Sequence
Base Sequence
...
Recombination, Genetic
Saccharomyces cerevisiae
Saccharomyces cerevisiae Proteins
```

MeSH terms can have one or more subheadings:

```
  ...
  <MeshHeading>
    <DescriptorName>Recombination, Genetic</DescriptorName>
    <QualifierName>genetics</QualifierName>
  </MeshHeading>
  <MeshHeading>
    <DescriptorName>Saccharomyces cerevisiae</DescriptorName>
    <QualifierName>genetics</QualifierName>
    <QualifierName>radiation effects</QualifierName>
  </MeshHeading>
  <MeshHeading>
    <DescriptorName>Saccharomyces cerevisiae Proteins</DescriptorName>
  </MeshHeading>
</MeshHeadingList>
...
```

Adding a -subset statement within the -block allows nested exploration of the subheadings for each MeSH term:

```
  efetch -db pubmed -id 1937004 -format xml | \
  xtract -pattern PubmedArticle -tab "" -element "MedlineCitation/PMID" \
    -block MeshHeading -pfx "\n" -tab "" -element DescriptorName \
      -subset QualifierName -pfx "/" -tab "" -element QualifierName
```

and results in a list of MeSH terms and associated subheadings:

```
  1937004
  Adenosine Triphosphatases
  Amino Acid Sequence
  Base Sequence
  ...
  Recombination, Genetic/genetics
  Saccharomyces cerevisiae/genetics/radiation effects
  Saccharomyces cerevisiae Proteins
```

The MeSH term and subheading fields actually have major topic attributes:

```
  <MeshHeading>
    <DescriptorName MajorTopicYN="N">Saccharomyces cerevisiae</DescriptorName>
    <QualifierName MajorTopicYN="Y">genetics</QualifierName>
    <QualifierName MajorTopicYN="N">radiation effects</QualifierName>
  </MeshHeading>
```

that can be selected in an -element argument as "DescriptorName@MajorTopicYN" or "QualifierName@MajorTopicYN". The user GUIDE shows how this can generate an asterisk ("*") character for a major ("starred") MeSH term or subheading.

EXPLORING MULTIPLE XML REGIONS

Multiple -block commands can be used in a single xtract command to explore different areas of the XML, and can disambiguate fields with the same name in different objects. Combining fields with commas allows them to be treated as sets, and the tab that normally separates these can be replaced with a -sep argument:

```
  efetch -db pubmed -id 781293,2678811,6301692,8332518 -format xml | \
  xtract -pattern PubmedArticle -element "MedlineCitation/PMID" \
    -block AuthorList -sep "|" -element LastName "#Author" \
    -block PubDate -sep " " -element Year,Month MedlineDate \
    -block DateCreated -sep "-" -element Year,Month,Day | \
  sort -t $'\t' -k 3,3n | column -s $'\t' -t
```

produces a table that allows easy parsing of author names, counts the number of authors present, and prints the date each record was published and the date it was entered into PubMed, sorting the results by the computed author count:

```
  781293   Casadaban                          1  1976 Jul      1976-10-02
  6301692  Krasnow|Cozzarelli                 2  1983 Apr      1983-06-17
  8332518  Benson|Lipman|Ostell               3  1993 Jul      1993-08-17
  2678811  Mortimer|Schild|Contopoulou|Kans   4  1989 Sep-Oct  1989-11-22
```

(The PubDate object can exist either in structured or string form, but would not contain a mixture of both types, so "-element Year,Month MedlineDate" will only contribute a single column to the output.)

SEQUENCE RECORDS IN INSDSEQ XML

Sequence records can be retrieved in an XML version of the GenBank or GenPept flatfile. Feature and qualifier names are data values, not XML tags, and require -match to select the desired object. A query for snail venom mature peptides:

```
  esearch -db protein -query "conotoxin AND mat_peptide [FKEY]" | \
  efetch -format gpc -mode xml | \
  xtract -pattern INSDSeq -ACCN INSDSeq_accession-version \
    -group INSDFeature -match ">mat_peptide<" \
        -avoid "<INSDFeature_partial" -pfx "\n" -element "&ACCN" \
      -block INSDQualifier -match ">peptide<" -element "%INSDQualifier_value" \
      -block INSDQualifier -match ">product<" -element INSDQualifier_value \
      -block INSDQualifier -match ">peptide<" -element INSDQualifier_value | \
  grep conotoxin | sort -t $'\t' -u -k 3,4 | \
  sort -t $'\t' -k 2,2n | column -s $'\t' -t
```

calculates the length of each mature peptide, and prints the product name and peptide sequence, removing redundant entries and sorting by peptide length:

```
  ADB43130.1  15  conotoxin Cal 1a       KCCKRHHGCHPCGRK
  ADB43131.1  15  conotoxin Cal 1b       LCCKRHHGCHPCGRT
  AAO33169.1  16  alpha-conotoxin GIC    GCCSHPACAGNNQHIC
  ADB43127.1  16  conotoxin Cal 5.1      DPAPCCQHPIETCCRR
  AAD31913.1  18  alpha A conotoxin Tx2  PECCSHPACNVDHPEICR
  ...
```

(The sequence accession is captured in a variable for use with each mat_peptide in the record. Prefix substitution ensures that every peptide is shown on a separate line. This also works for multi-product precursor proteins such as proinsulin.)

Multiple -avoid or -match conditions are specified with -and and -or commands:

```
  -group INSDFeature -avoid ">proprotein<" -and ">sig_peptide<" \
    -block INSDQualifier -match ">calculated_mol_wt<" -or ">peptide<" \
```

STORING COMMON PHRASES IN ALIAS FILES

Frequently used, long, or complicated search phrases can be saved in a file to avoid having to retype (or copy and paste) the full text for each query. Each line of the file has a shortcut keyword, a tab character, and the expanded search term. Shortcuts are referenced by placing them in parentheses and prefixing with a pound ("#") sign. For example, given a file named "query_aliases" containing:

```
  jour_filt    [MULT] AND ncbijournals [FILT]
  trans_imm    (transposition OR target) immunity
```

the esearch line in:

```
  esearch -alias query_aliases -db nlmcatalog -query "Science (#jour_filt)" | \
  esummary | \
  xtract -pattern DocumentSummary -element ISOAbbreviation \
    -subset ISSNInfo -sep "|" -element issn,issntype | \
  column -s $'\t' -t
```

will be expanded to:

```
  esearch -db nlmcatalog -query "Science [MULT] AND ncbijournals [FILT]"
```

with the query producing:

```
  J. Zhejiang Univ. Sci.  1009-3095|Print  1009-3095|Linking
  Science (80- )          0193-4511|Print  0193-4511|Linking
  Science                 0036-8075|Print  1095-9203|Electronic  ...
```

AUTOMATION WITH SCRIPTS

Taking an adventurous plunge into the world of programming, EDirect queries can be automated with shell scripts. Creative use of the "sh" and "xargs" commands can obtain the same behavior from the command line, without the need to write separate script files. For example:

```
  einfo -dbs | xtract -pattern DbName -element DbName | sort | \
  xargs -n 1 sh -c 'einfo -db "$0" | \
    xtract -pattern DbInfo -tab "\n\n" -element DbName \
      -block Field -pfx "[" -sep "]\t" -tab "\n" -element Name,FullName | \
    sed "s/ /*/g" | sort -k 2,2f | sed "s/*/ /g" | expand'
```

will display the indexed fields for each Entrez database:

```
  ...
  pubmed
  [AFFL]  Affiliation
  [ALL]   All Fields
  [AUTH]  Author
  [COLN]  Author - Corporate
  [FAUT]  Author - First
  [FULL]  Author - Full
  [LAUT]  Author - Last
  [AUCL]  Author Cluster ID
  [BOOK]  Book
  [CDAT]  Date - Completion
  [CRDT]  Date - Create
  [EDAT]  Date - Entrez
  [MHDA]  Date - MeSH
  [MDAT]  Date - Modification
  [PDAT]  Date - Publication
  [ECNO]  EC/RN Number
  [ED]    Editor
  [EPDT]  Electronic Publication Date
  [EID]   Extended PMID
  [FILT]  Filter
  [GRNT]  Grant Number
  [INVR]  Investigator
  [FINV]  Investigator - Full
  [ISBN]  ISBN
  [ISS]   Issue
  [JOUR]  Journal
  [LANG]  Language
  [LID]   Location ID
  [MAJR]  MeSH Major Topic
  [SUBH]  MeSH Subheading
  [MESH]  MeSH Terms
  ...
```