EXFOR relational database. X4Lite. Accessing data in C5, XML, JSON.

Translation data from EXFOR relational to JSON-X4DB

Viktor Zerkin
International Atomic Energy Agency, Nuclear Data Section

Excerpt from the part-II of “Developing an automatically readable, comprehensive and curated experimental reaction database”
WPEC Subgroup-50, 1-st Meeting, WebEx, 14-15 September 2020

WPEC SG50, Codes and Database SSG,
Web Meeting, 06 April 2021

Contents

Part II. EXFOR relational database. X4Lite.
Accessing data in C5, XML, JSON

1. Data formats overview: X4+, XML, StdOut, C5, JSON; concept of Dataset; Dataset vs. ENTRY/SUBENT/Pointer
2. EXFOR database: structure, content, updates
3. X4Lite: database, retrieval and converter codes; pro and con

Concluding remarks

Part III. Translation data from EXFOR relational to JSON-X4DB

1. Using database schema for creating JSON files
2. Relational database with JSON fields (hybrid implementation)
3. Universal flexible translation EXFOR Relational to JSON: automatic and configurable Java-code

Concluding remarks

Part II.

EXFOR relational database. X4Lite.
Accessing data in C5, XML, JSON.

Excerpt from
WPEC Subgroup-50, “Developing an automatically readable, comprehensive and curated experimental reaction database”
1-st Meeting, WebEx, 14-15 September 2020

Data formats overview

X4+ EXFOR-Interpreted; X4± Interactive Tree

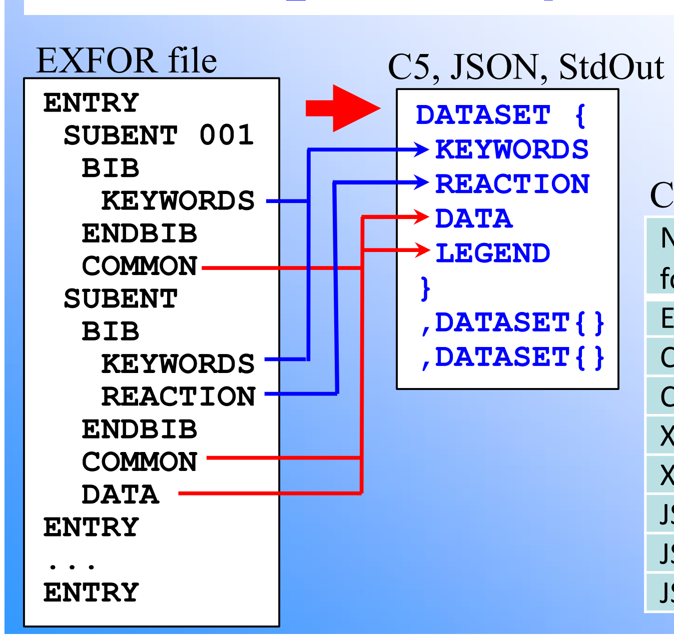
1. Presents EXFOR as it is + extra lines with information from Dictionaries, NSR, etc.
2. Numbers in traditional style
3. No limit on the number of values per line

XML

1. Repeats structure of EXFOR file using nested <elements>; includes information from EXFOR Dictionaries explaining codes
2. Numbers are presented in traditional style (no more E-less Fortran format for numbers)

C5, JSON, JSON_FY, Std_out

1. File contains Datasets; no text blocks for ENTRY, SUBENT, BIB; no Pointers
2. Dataset is identified by DatasetID (SUBENT + Pointer); includes all information related to one reaction: Reaction-code, selected/all Keywords from SUBENT-1 and current SUBENT, Data-section and Legend
3. Data are presented as function Y=Y(X1,X2,…), columns are sorted (fixed order according to Dictionary)
4. Data-section: all data from DATA and COMMON from EXFOR SUBENT-1 and current SUBENT
5. Legend and Keywords contain EXFOR codes and their interpretation (e.g. basic-units and conversion factors)
6. C5 and JSON_FY contain computational data values; StdOut, XML and JSON (as of now) – only original values



EXFOR file

```
ENTRY
  SUBENT 001
   BIB
     KEYWORDS
   ENDBIB
   COMMON
  SUBENT
   BIB
     KEYWORDS
     REACTION
   ENDBIB
   COMMON
   DATA
ENTRY
. . .
ENTRY
```

C5, JSON, StdOut

```
DATASET {
  KEYWORDS
  REACTION
  DATA
  LEGEND
}
,DATASET{}
,DATASET{}
```

Comparison of formats: summary

Nucl. data format	Numbers' format /Language	Sequence (main block)	Meta data	Interpret. from Dictionaries	Orig. data	Computa-tional data
EXFOR	Fixed-length, E-less	ENTRY	yes	no	yes	no
C4	Fixed-fmt lines	SUBENT	no	no	no	yes
C5	Fixed-fmt lines	Datasets	yes	yes	no	yes
X4+	Flex. fields /HTML	ENTRY	yes	yes	yes	no
XML	Flex. fields /XML	ENTRY	yes	yes	yes	no
JSON	Flex. fields /JSON	Datasets	yes	yes	yes	no
JSON_FY	Flex. fields / JSON	Datasets	yes	yes	no	yes
JSON_X4	Flex. fields / JSON	Datasets	yes	yes	yes	yes

EXFOR database: structure and content

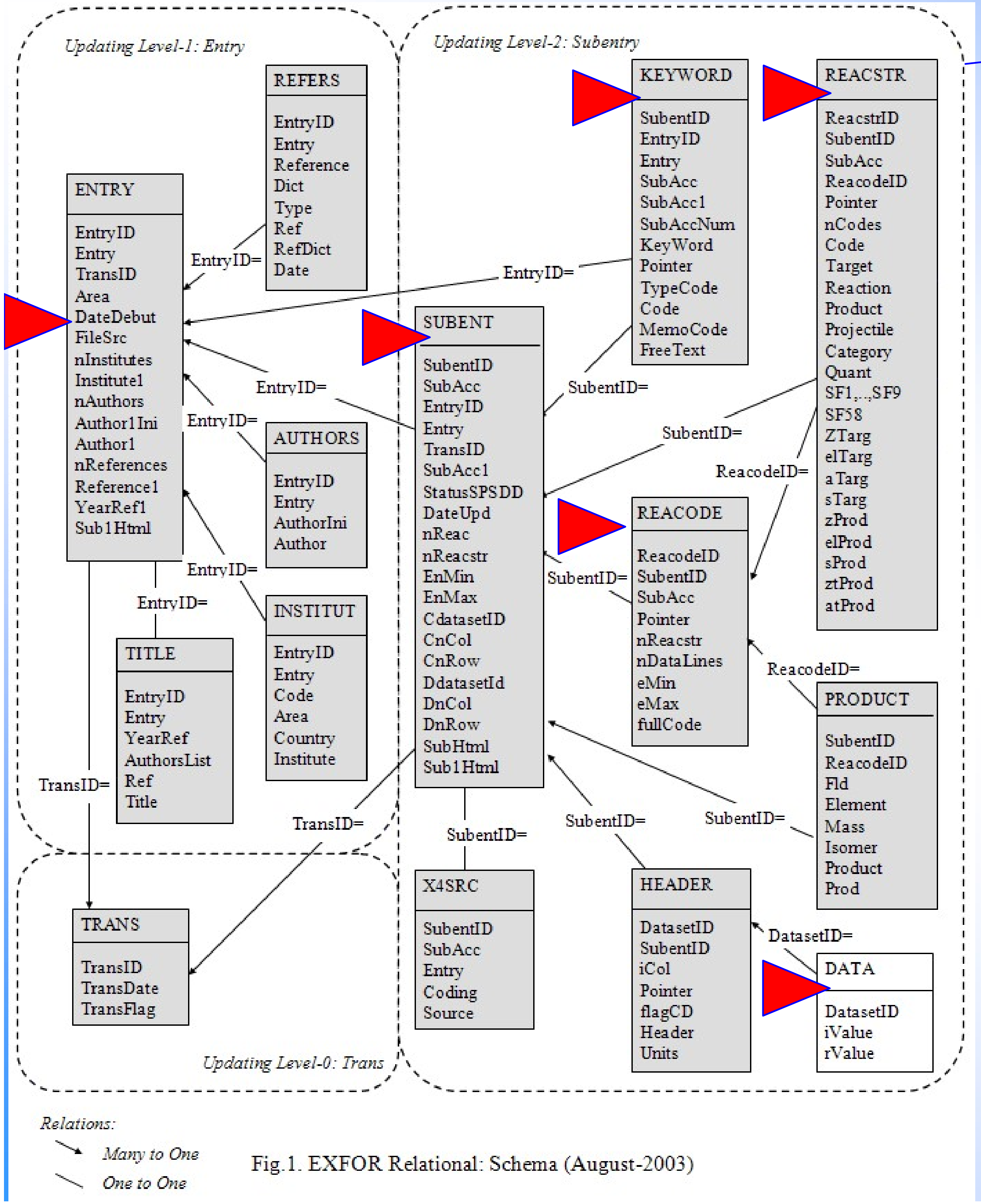


Fig.1. EXFOR Relational: Schema (August-2003)

Initial database:
EXFOR + Dictionaries

Database extensions:

Corrections

Created: 2010
Updated: 2020-09-10
Records: 15,663
Size: 9.2 Mb

Automatic and experts' corrections. Available online via C4, TAB, Plots.

X4-NSR PDF

Created: 2012
Updated: 2020-09-04
Records: 218,210
Size: 180 Gb

PDF files of published materials of EXFOR and NSR databases. Full contents available online for authorized users.

Test search

Created: 2014
Updated: 2020-09-10
Records: 1,440,084
Size: 184 Mb

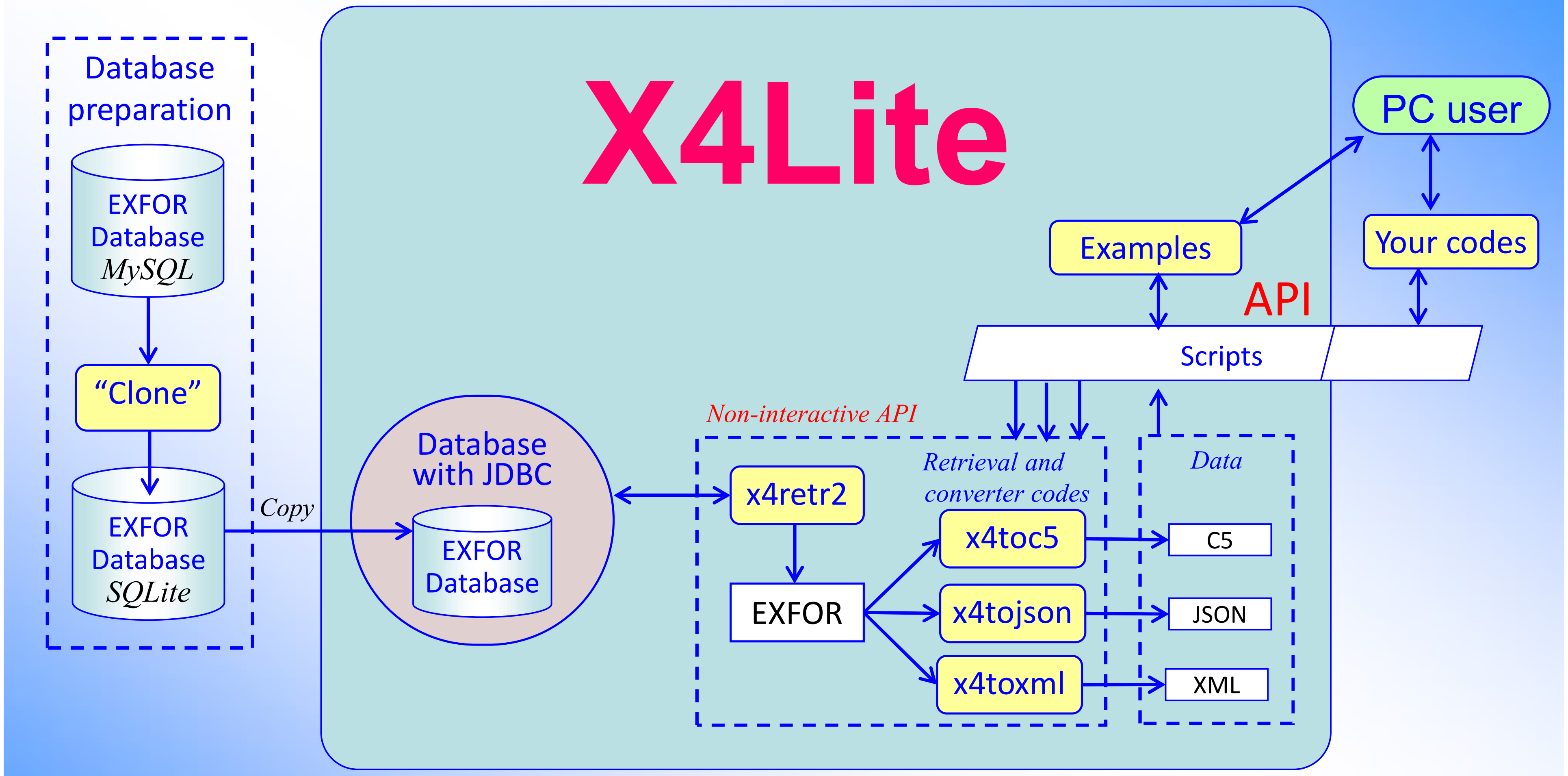
Google-like search in interpreted EXFOR, incl. free text, keywords, codes and their interpretation from dictionaries.

EXFOR Archive

Created: 2014
Updated: 2020-09-10
Entries: 99,381
Subent: 783,183
Size: 0.9 Gb

Contains current and all previous versions of every SUBENT. Available online for EXFOR compilers.

X4Lite: database, retrieval and converter codes

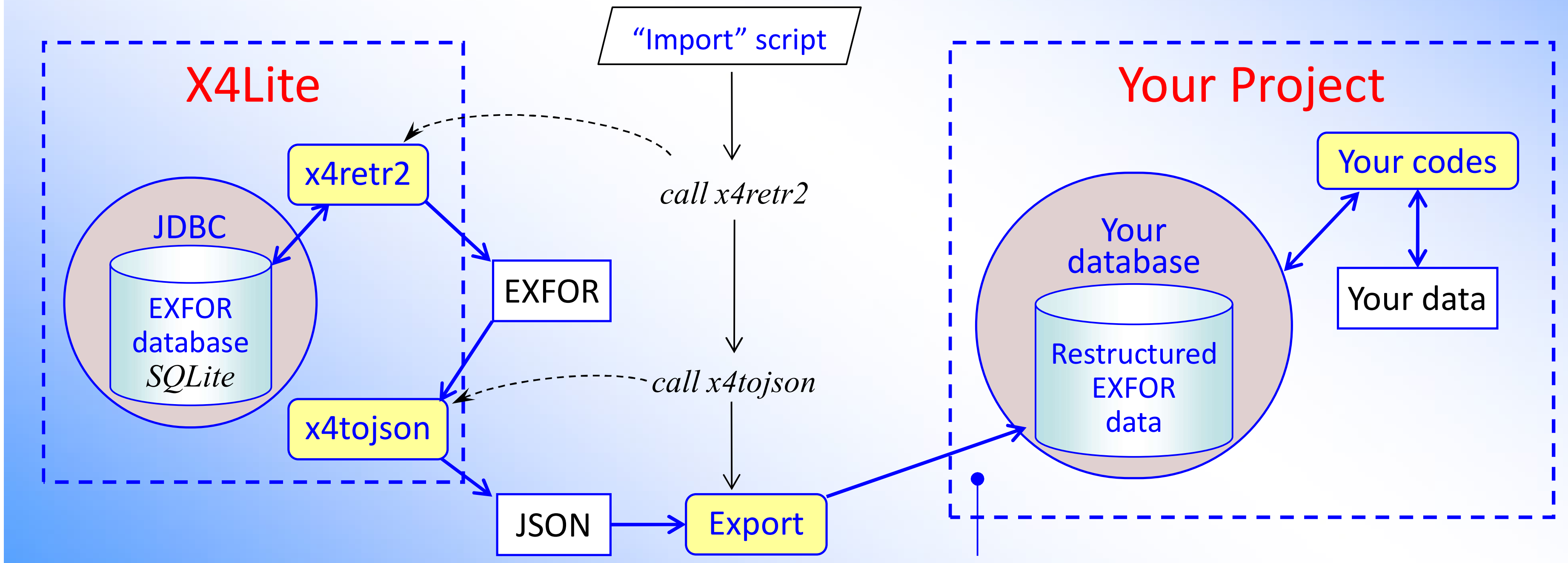


X4Lite preparation is fully automatic. Can be done on regular basis by the IAEA-NDS.

X4Lite. Specialized system for usage under other software packages and containing only
1) EXFOR relational database in SQLite: one file for Linux, Windows, MacOS
2) retrieval code producing list of datasets and/or EXFOR file
3) codes converting EXFOR file to X4+, C5, C5M, JSON, XML

Planning your database for your project?

- *Select data format (e.g. JSON or C5 or XML)*
- *Prepare your "import" script doing:*
  - *Search and retrieve EXFOR data needed in your project*
  - *[Make a loop on the list of found datasets if necessary]*
  - *Call converter from EXFOR file to selected format*
  - *Store dataset into your data structure or SQL/noSQL database*
- *Download X4Lite and run "import" script*

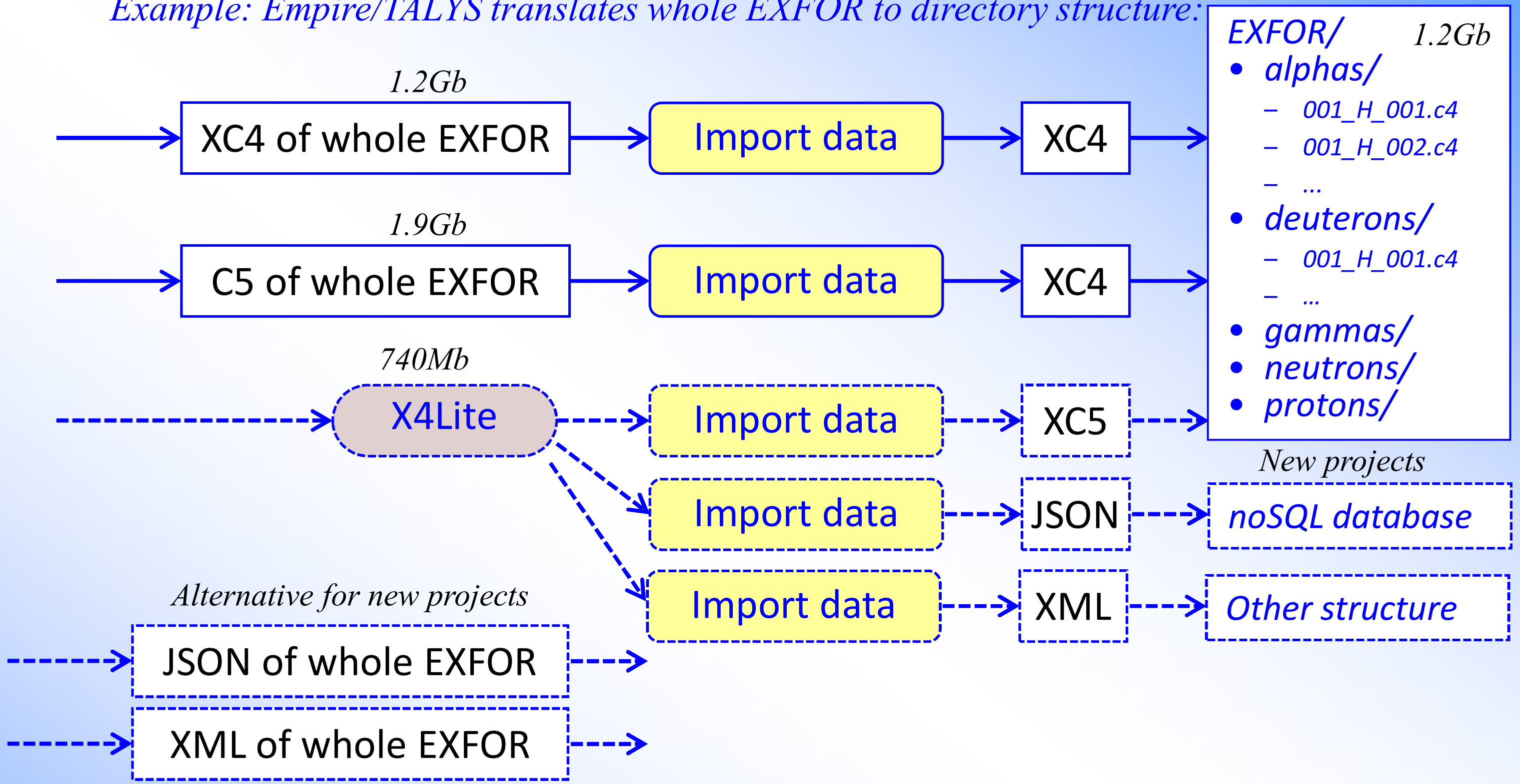


Project/team can build different databases, e.g.:

- *XC4, C5: Empire, TALYS*
- *C5M: GANDR*
- *JSON: SG50 (?)*

X4Lite: pro & con. Alternatives.

Example: Empire/TALYS translates whole EXFOR to directory structure:



What is difference?

- *More rational maintenance (at the IAEA-NDS)*
- *Freedom for user to use formats C4/C5, JSON, XML; easy to use modern tools and languages*
- *Translation from EXFOR C4: 65%, to C5: 75%, to JSON: 98%, to XML: 100%*
- *Easy programming access to all data/information from EXFOR and Dictionaries (name:value)*
- *Easier to filter out and store only data needed for a project*
- *Options to make re-calculations and include/exclude data columns: CM-Lab, RR-B/SR, inverse reactions and kinematics, dictionary information, perhaps monitor data and/or automatic renormalization*
- *Other advantages/disadvantages will be discovered during exploitation*

Concluding remarks

1. EXFOR data correction system is successfully functioning on Web at the IAEA-NDS and NNDC cites working with C4 and TABLE files. Current system can be revised, expanded or rewritten in a short term.

2. Current versions of EXFOR output to C5, X4JSON, XML have a great potential and should be propagated to users' community for practical usage in applications, for feedback and improvements.

3. X4Lite is computational EXFOR for professional nuclear data users.

Part III.

Translation data from EXFOR relational to JSON-X4DB

Prepared for
WPEC SG50, Codes and Database SSG,
Web Meeting, 06 April 2021

Features and parameters of EXFOR system

1. Planned features of the system (2000):

1. All information in EXFOR should be available for search in any order (direct access)
2. Execution time of typical request should be within 2-3 sec
3. The system should be really platform independent (tested) (simplest: no stored procedures, no foreign keys, etc.)
4. The system should guarantee integrity of original data
    - usage of BLOBs to store SUBENT
    - data are stored in their original form (not by lines as it is done in NSR database)
    - convincing other centers to switch to central database
5. Whole system (database and programs) should fit to CD-ROM=640Mb (storage of zipped BLOBs)
6. The database should be easy deployed to mirror-sites (MyISAM, MDB) without uploading system
7. Extendable set of tables and columns in the tables
8. System should allow usage of programs on several languages (legacy codes) and extensions
9. Modularity and robustness of software, re-use of modules
10. Interactive multiplatform plotting

2. Allowed to achieve:

1. Merging EXFOR libraries to common library (2002-2005)
2. Global EXFOR maintenance system in the IAEA-NDS (since 2005): TRANS files and fixed Master file for every update
3. Optimising of efforts in NRDC
4. Common (robust) EXFOR Web retrieval system: IAEA-NDS, NNDC (USA), India, China, Russia
5. Integrating with EXFOR compilation control system

3. Not foreseeing extensions (2007-2013):

1. PDF collection (authorised Web access)
2. Connection and import from NSR
3. Export to R33 (IBANDL)
4. EXFOR data re-normalization system
5. Construction covariance matrices using uncertainties
6. Uploading system for remote data checking and processing (for EXFOR compilers)
7. Web system without Internet

Current status of EXFOR-Relational

1. Relational EXFOR database: common between NDS-NNDC
   a) schema based on “EXFOR-Access CD-ROM”, discussed and initially agreed in 2000 between NDS, NNDC, CNPD (after “Nuclear database: migration to relational database and Java technology”)
   b) existing and maintained at NDS and NNDC from 2000 to 2021:
   c) OS: Windows, Linux, MacOS
   d) DBMS: MS-Access (2000), MySQL (2001), SyBase (2005), SQLite (2020)
   e) Web: NDS, NNDC, 3 Mirrors (India, China, Russia)
   f) deployed to Mirror-sites and on CD-ROM to individual users

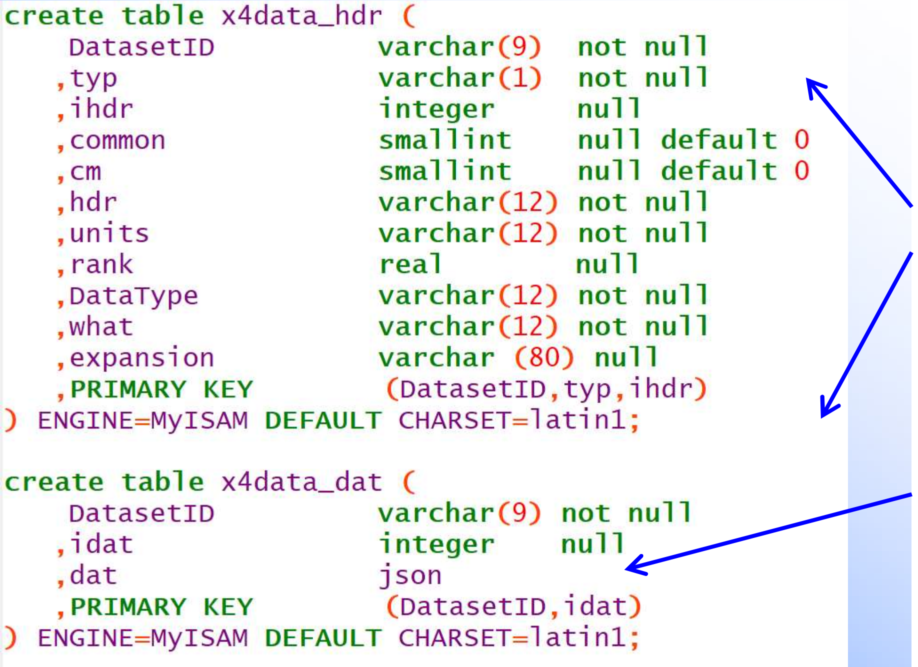
2. EXFOR-CINDA Web Retrieval system:
   official NRDC Web retrieval system since 2008

3. Current versions of EXFOR output to C5, X4JSON, XML:
   a) easier to use in users’ applications than EXFOR
   b) have fixed format, require converter

Extension of EXFOR-Relational

1. Currently EXFOR data are stored in relational EXFOR database in BLOBs as part of SUBENT and therefore need to be extracted by an external program.
   a) So, we need retrieval + converter of EXFOR to another formats.
   b) Can we avoid complicated converter?
   c) Can we store/retrieve data values in rational way? (avoid BLOBs)
2. Traditionally relational databases have problems to store/manipulate with flexible vector data
3. Now relational DBMS-s offer some functionality to deal with JSON-type fields in the tables.

Flexible solution: use single JSON cell to store one experimental data point

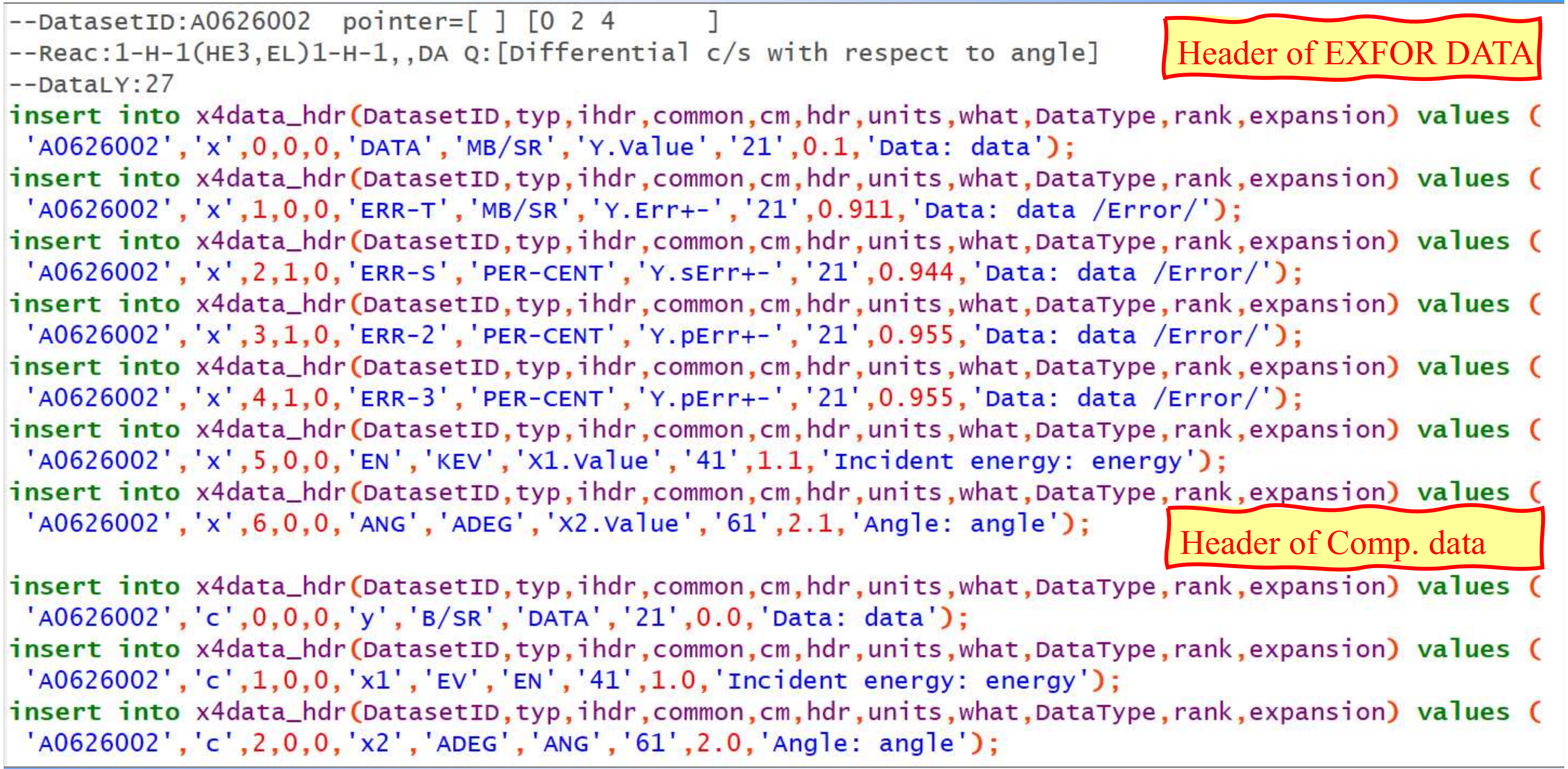


```
create table x4data_hdr (
    DatasetID            varchar(9)   not null
   ,typ                  varchar(1)   not null
   ,ihdr                 integer      null
   ,common               smallint     null default 0
   ,cm                   smallint     null default 0
   ,hdr                  varchar(12)  not null
   ,units                varchar(12)  not null
   ,rank                 real         null
   ,DataType             varchar(12)  not null
   ,what                 varchar(12)  not null
   ,expansion            varchar (80) null
   ,PRIMARY KEY           (DatasetID,typ,ihdr)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

create table x4data_dat (
    DatasetID            varchar(9) not null
   ,idat                 integer    null
   ,dat                  json
   ,PRIMARY KEY           (DatasetID,idat)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

1. Idea is to store for data points based on the concept of Dataset (sorted EXFOR): original EXFOR data and computational data
2. Two new tables for Headers and Data: x4data_hdr and x4data_dat
3. Headers have type “x” and “c” and description of the Data from EXFOR Dictionaries
4. Table x4data_dat has a column with type JSON

Extension of EXFOR-Relational

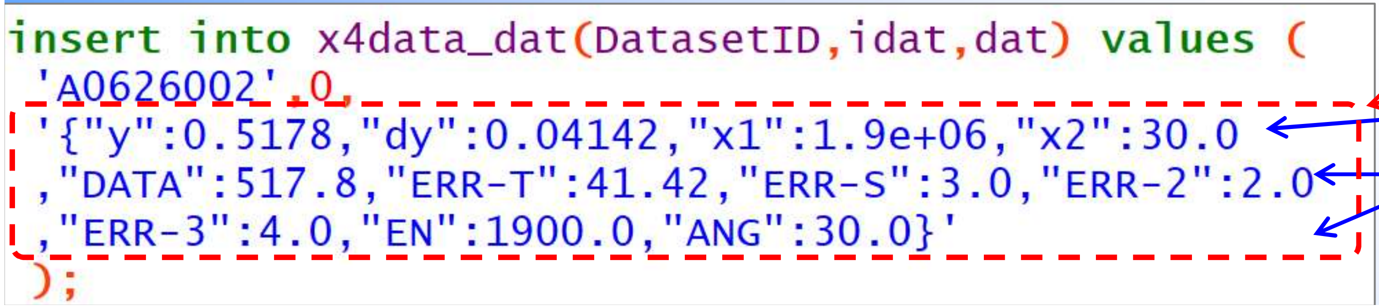


```
--DatasetID:A0626002  pointer=[ ] [0 2 4       ]
--Reac:1-H-1(HE3,EL)1-H-1,,DA Q:[Differential c/s with respect to angle]
--DataLY:27
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',0,0,0,'DATA','MB/SR','Y.Value','21',0.1,'Data: data');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',1,0,0,'ERR-T','MB/SR','Y.Err+-','21',0.911,'Data: data /Error/');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',2,1,0,'ERR-S','PER-CENT','Y.sErr+-','21',0.944,'Data: data /Error/');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',3,1,0,'ERR-2','PER-CENT','Y.pErr+-','21',0.955,'Data: data /Error/');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',4,1,0,'ERR-3','PER-CENT','Y.pErr+-','21',0.955,'Data: data /Error/');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',5,0,0,'EN','KEV','X1.Value','41',1.1,'Incident energy: energy');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','x',6,0,0,'ANG','ADEG','X2.Value','61',2.1,'Angle: angle');

insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','c',0,0,0,'y','B/SR','DATA','21',0.0,'Data: data');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','c',1,0,0,'x1','EV','EN','41',1.0,'Incident energy: energy');
insert into x4data_hdr(DatasetID,typ,ihdr,common,cm,hdr,units,what,DataType,rank,expansion) values (
 'A0626002','c',2,0,0,'x2','ADEG','ANG','61',2.0,'Angle: angle');
```

Header of EXFOR DATA

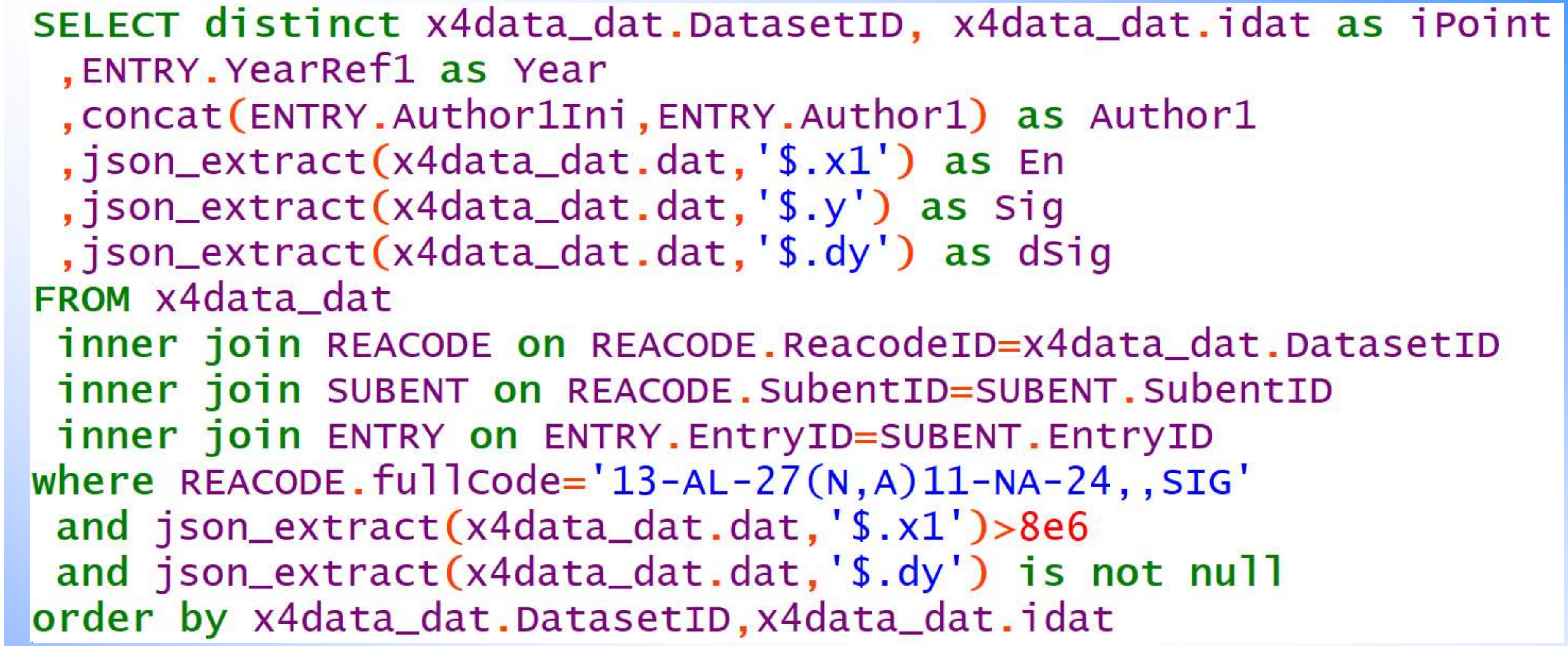
Header of Comp. data



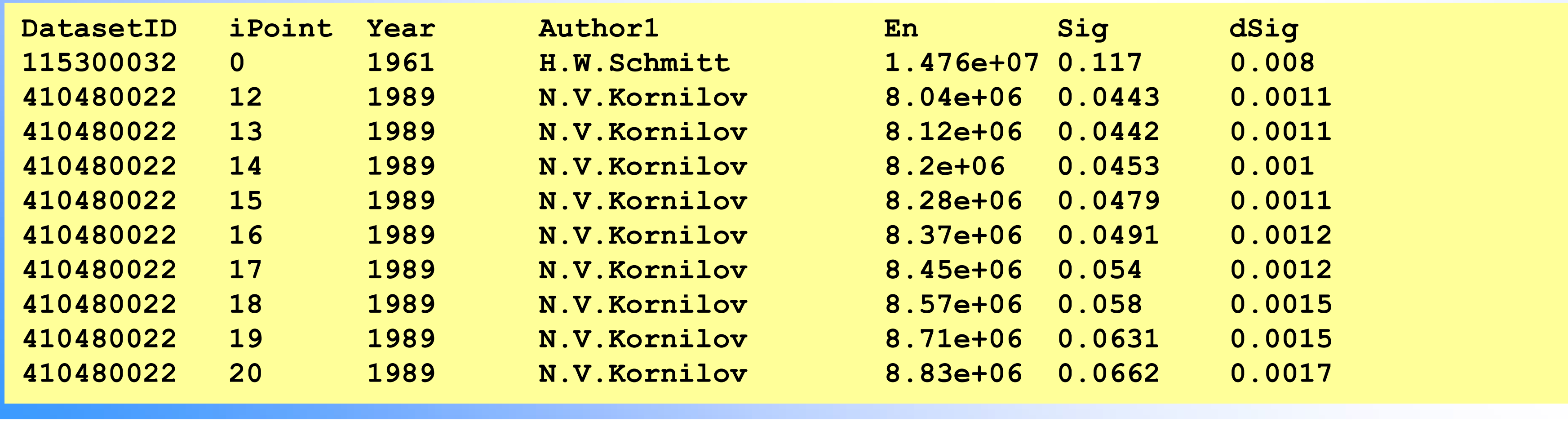
```
insert into x4data_dat(DatasetID,idat,dat) values (
 'A0626002',0,
 '{"y":0.5178,"dy":0.04142,"x1":1.9e+06,"x2":30.0
 ,"DATA":517.8,"ERR-T":41.42,"ERR-S":3.0,"ERR-2":2.0
 ,"ERR-3":4.0,"EN":1900.0,"ANG":30.0}'
 );
```

JSON object {
  Comp. data: y(x1,x2…)
  ,EXFOR DATA
}

Example of SQL query extracting data from JSON fields



```
SELECT distinct x4data_dat.DatasetID, x4data_dat.idat as iPoint
 ,ENTRY.YearRef1 as Year
 ,concat(ENTRY.Author1Ini,ENTRY.Author1) as Author1
 ,json_extract(x4data_dat.dat,'$.x1') as En
 ,json_extract(x4data_dat.dat,'$.y') as Sig
 ,json_extract(x4data_dat.dat,'$.dy') as dSig
FROM x4data_dat
 inner join REACODE on REACODE.ReacodeID=x4data_dat.DatasetID
 inner join SUBENT on REACODE.SubentID=SUBENT.SubentID
 inner join ENTRY on ENTRY.EntryID=SUBENT.EntryID
where REACODE.fullCode='13-AL-27(N,A)11-NA-24,,SIG'
 and json_extract(x4data_dat.dat,'$.x1')>8e6
 and json_extract(x4data_dat.dat,'$.dy') is not null
order by x4data_dat.DatasetID,x4data_dat.idat
```



```
DatasetID    iPoint   Year      Author1             En         Sig       dSig
115300032    0        1961      H.W.Schmitt         1.476e+07  0.117     0.008
410480022    12       1989      N.V.Kornilov        8.04e+06   0.0443    0.0011
410480022    13       1989      N.V.Kornilov        8.12e+06   0.0442    0.0011
410480022    14       1989      N.V.Kornilov        8.2e+06    0.0453    0.001
410480022    15       1989      N.V.Kornilov        8.28e+06   0.0479    0.0011
410480022    16       1989      N.V.Kornilov        8.37e+06   0.0491    0.0012
410480022    17       1989      N.V.Kornilov        8.45e+06   0.054     0.0012
410480022    18       1989      N.V.Kornilov        8.57e+06   0.058     0.0015
410480022    19       1989      N.V.Kornilov        8.71e+06   0.0631    0.0015
410480022    20       1989      N.V.Kornilov        8.83e+06   0.0662    0.0017
```

Translation EXFOR database to JSON

1. We can use names of columns from database schema to generate JSON
2. We can use SQL SELECT query to rename, filter and combine columns from EXFOR database
3. We can build a program generating any JSON hierarchy automatically, or semi-automatically using EXFOR hierarchy and configuration file
4. Such a program could generate JSON files for selected part of EXFOR database

Program: x4db2json1.java, 2021-04-05

1. Generates one JSON file for single ENTRY
2. Hierarchy:

```
{Entry
        [Subentry
             [Keyword]
             [Dataset
                   [Header]
                   [Data]
             ]
        ]
}
```

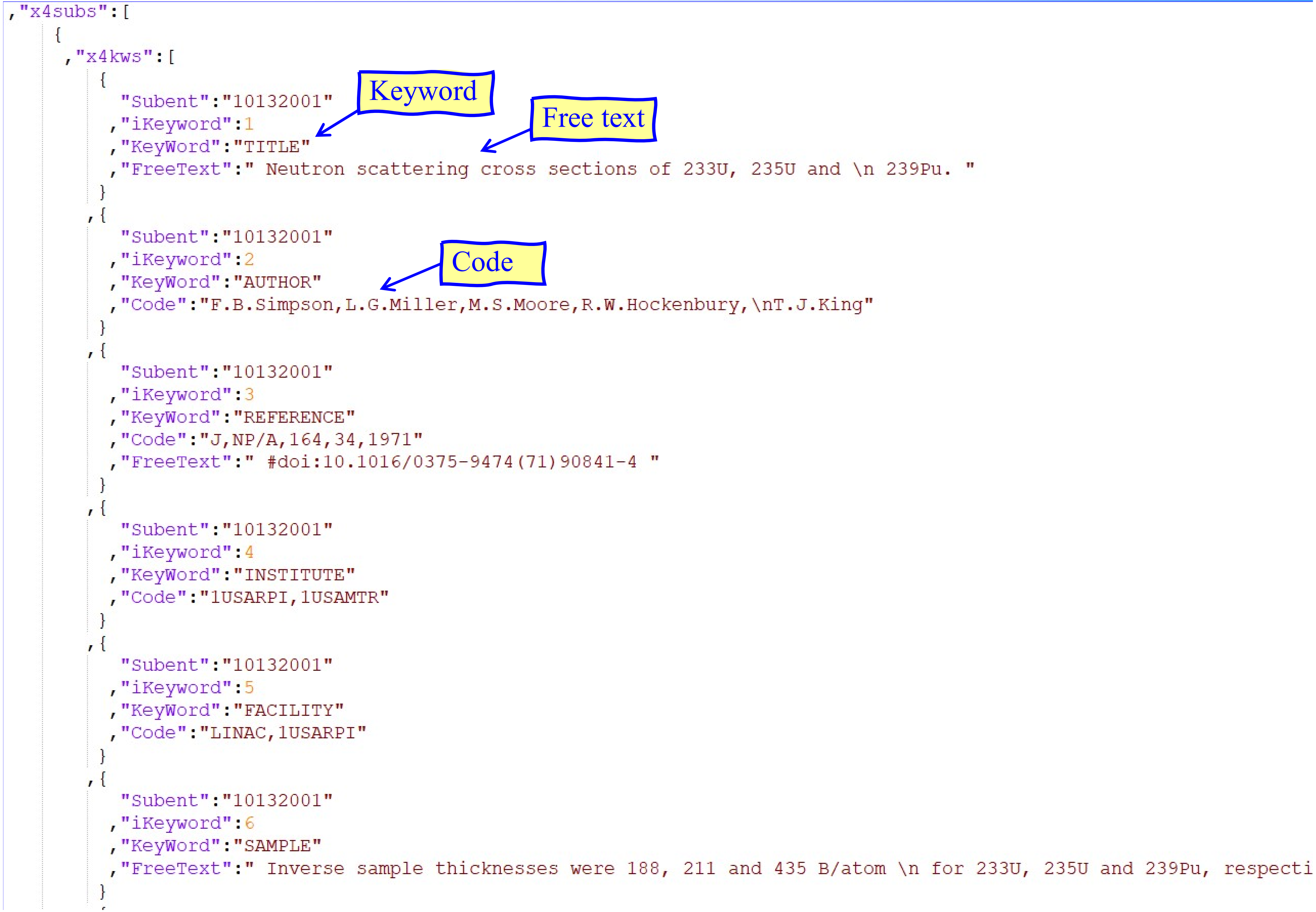
3. “x4db2json1.java” ~500 lines (main recursive method: exeSQL2json ~100 lines)

Automatically generated JSON file

```
{
  "format":"JSON.X4DB-0.0.1"
 ,"now":"2021/04/05T14:26:26.632"
 ,"program":"x4db2json1, by V.Zerkin, IAEA-NDS, 2021 (ver.2021-04-05)",
  "EntryID":10001
 ,"Entry":"10001"
 ,"Area":"1"
 ,"expArea":"1"
 ,"CenterID":1
 ,"DateDebut":"1973-06-26"
 ,"UpdateNo":267
 ,"TransID":"0000"
 ,"TransDate":"20050926"
 ,"TransFile":"EXFOR-2015-05-05.bck"
 ,"nInstitutes":1
 ,"Institute1":"1USARPI"
 ,"nAuthors":5
 ,"Author1Ini":"R.W."
 ,"Author1":"Hockenbury"
 ,"nReferences":1
 ,"Reference1":"J,PR,178,1746,196902"
 ,"Ref1":"J,PR"
 ,"YearRef1":1969
 ,"Publication1":"J,PR:,178,1746:196902"
 ,"stdFileName":"J,PR,178,1746,1969"
 ,"TypeRef1":"J"
 ,"NsrKeyNo":"1969HO12"
 ,"DOI":"10.1103/PhysRev.178.1746"
 ,"origEntry":"10001"
 ,"x4subs":[
    {
```

```
,"x4subs":[
    {
      "SubentID":10001001
     ,"SubAcc":"10001001"
     ,"EntryID":10001
     ,"Entry":"10001"
     ,"SPSDD":"0"
     ,"DateUpd":"2005-09-26 00:00:00.0"
     ,"DateCompil":"1998-09-14"
     ,"UpdateNo":267
     ,"TransID":"0000"
     ,"TransDate":"20050926"
     ,"TransFile":"EXFOR-2015-05-05.bck"
     ,"nReac":0
     ,"nReacstr":0
     ,"CDatasetID":10001001
     ,"CnCol":0
     ,"CnRow":0
     ,"DDatasetID":1010001001
     ,"DnCol":0
     ,"DnRow":0
     ,"origEntry":"10001"
     ,"origSubent":"10001001"
     ,"x4kws":[
        {
          "Subent":"10001001"
         ,"iKeyword":1
         ,"KeyWord":"INSTITUTE"
         ,"Code":"1USARPI"
        }
       ,{
          "Subent":"10001001"
```

Automatically generated JSON file (cont.)



```
,"x4subs":[
    {
     ,"x4kws":[
        {
          "Subent":"10132001"
         ,"iKeyword":1
         ,"KeyWord":"TITLE"
         ,"FreeText":" Neutron scattering cross sections of 233U, 235U and \n 239Pu. "
        }
       ,{
          "Subent":"10132001"
         ,"iKeyword":2
         ,"KeyWord":"AUTHOR"
         ,"Code":"F.B.Simpson,L.G.Miller,M.S.Moore,R.W.Hockenbury,\nT.J.King"
        }
       ,{
          "Subent":"10132001"
         ,"iKeyword":3
         ,"KeyWord":"REFERENCE"
         ,"Code":"J,NP/A,164,34,1971"
         ,"FreeText":" #doi:10.1016/0375-9474(71)90841-4 "
        }
       ,{
          "Subent":"10132001"
         ,"iKeyword":4
         ,"KeyWord":"INSTITUTE"
         ,"Code":"1USARPI,1USAMTR"
        }
       ,{
          "Subent":"10132001"
         ,"iKeyword":5
         ,"KeyWord":"FACILITY"
         ,"Code":"LINAC,1USARPI"
        }
       ,{
          "Subent":"10132001"
         ,"iKeyword":6
         ,"KeyWord":"SAMPLE"
         ,"FreeText":" Inverse sample thicknesses were 188, 211 and 435 B/atom \n for 233U, 235U and 239Pu, respecti
        }
```

Keyword

Free text

Code

Automatically generated JSON file (cont.)

```
,"x4reac":[
   {
     "ReacodeID":"100010061"
    ,"SubentID":10001006
    ,"SubAcc":"10001006"
    ,"Pointer":"1"
    ,"nReacstr":1
    ,"nDataLines":24
    ,"eMin":2350.0
    ,"eMax":129000.0
    ,"zaTarget1":26056
    ,"zaIncident1":1
    ,"MF":402
    ,"MT":6001
    ,"reacCombi":"R1#"
    ,"fullCode":"26-FE-56(N,0),,EN"
    ,"x4reacstr":[
       {
         "ReacstrID":"1000100611"
        ,"SubentID":10001006
        ,"ReacodeID":"100010061"
        ,"Pointer":"1"
        ,"iReacstr":1
        ,"Code":"26-FE-56(N,0),,EN"
        ,"Target":"Fe-56"
        ,"Reaction":"N,0"
        ,"Projectile":"N"
        ,"ReactionType":"RE"
        ,"CindaQuantity":"RP"
        ,"Quant":"RP"
        ,"SF1":"26-FE-56"
        ,"SF2":"N"
        ,"SF3":"0"
        ,"SF6":"EN"
        ,"SF58":",EN"
        ,"zIncident":0
        ,"zTarg":26
        ,"elTarg":"Fe"
        ,"aTarg":56
        ,"ztTarg":"26"
        ,"atTarg":"56"
        ,"zProd":-1
        ,"aProd":-1
       }
     ]
```

```
,"x4data_hdr":[
   {
     "DatasetID":"100010061"
    ,"typ":"c"
    ,"ihdr":0
    ,"common":0
    ,"cm":0
    ,"hdr":"y"
    ,"units":"EV"
    ,"rank":0.0
    ,"DataType":"21"
    ,"what":"DATA"
    ,"expansion":"Data: data"
   }
  ,{
     "DatasetID":"100010061"
    ,"typ":"x"
    ,"ihdr":0
    ,"common":0
    ,"cm":0
    ,"hdr":"DATA"
    ,"units":"KEV"
    ,"rank":0.1
    ,"DataType":"21"
    ,"what":"Y.Value"
    ,"expansion":"Data: data"
   }
 ]
,"x4data_dat":[
   {
     "DatasetID":"100010061"
    ,"idat":0
    ,"dat":{"y":2350.0,"DATA":2.35}
   }
  ,{
     "DatasetID":"100010061"
    ,"idat":1
    ,"dat":{"y":11200.0,"DATA":11.2}
   }
```

Configuration

Fully automatic.

SQL query:

```
select * from ENTRY
```

```
 {
   "format":"JSON.X4DB-0.0.1"
  ,"now":"2021/04/05T14:26:26.632"
  ,"program":"x4db2json1, by V.Zerkin, IAEA-NDS",
   "EntryID":10001
  ,"Entry":"10001"
  ,"Area":"1"
  ,"expArea":"1"
  ,"CenterID":1
  ,"DateDebut":"1973-06-26"
  ,"UpdateNo":267
  ,"TransID":"0000"
  ,"TransDate":"20050926"
  ,"TransFile":"EXFOR-2015-05-05.bck"
  ,"nInstitutes":1
  ,"Institute1":"1USARPI"
  ,"nAuthors":5
  ,"Author1Ini":"R.W."
  ,"Author1":"Hockenbury"
  ,"nReferences":1
  ,"Reference1":"J,PR,178,1746,196902"
  ,"Ref1":"J,PR"
  ,"YearRef1":1969
  ,"Publication1":"J,PR:,178,1746:196902"
  ,"stdFileName":"J,PR,178,1746,1969"
  ,"TypeRef1":"J"
  ,"NsrKeyNo":"1969HO12"
  ,"DOI":"10.1103/PhysRev.178.1746"
  ,"origEntry":"10001"
  ,"x4subs":[
     {
```

```
create table ENTRY (
    EntryID          integer NOT NULL,
    Entry            char(5),
    Area             char(1),
    expArea          char(1),
    CenterID         smallint    null,
    DateDebut        date,
    UpdateNo         smallint,
    TransID          char(5)     null,
    TransDate        char(8)     null,
    TransFile        varchar(20) null,
    nInstitutes      smallint    null,
    Institute1       char(7)     null,
    nAuthors         smallint    null,
    Author1Ini       varchar(12) null,
    Author1          varchar(55) null,
    nReferences      smallint    null,
    Reference1       varchar(32) null,
    Ref1             varchar(32) null,
    YearRef1         smallint    null,
    Publication1     varchar(40) null,
    stdFileName      varchar (40) null,
    TypeRef1         char   (1)  null,
    NsrKeyNo         varchar(8)  null,
    DOI              varchar(40) null,
    CompilerID varchar (40) null,
    PRIMARY KEY      (EntryID)
)
```

Automatic, but with user's config.

SQL query:

```
select Entry,concat(Author1Ini,Author1) as
Author1,Reference1,NsrKeyNo,DOI from ENTRY
```

```
 {
   "format":"JSON.X4DB-0.0.1"
  ,"now":"2021/04/06T10:32:39.465"
  ,"program":"x4db2json1, by V.Zerkin, IAEA-NDS",
   "Entry":"10001"
  ,"Author1":"R.W.Hockenbury"
  ,"Reference1":"J,PR,178,1746,196902"
  ,"NsrKeyNo":"1969HO12"
  ,"DOI":"10.1103/PhysRev.178.1746"
  ,"x4subs":[
     {
```

Concluding remarks

1. Extension of EXFOR relational database to store computational and EXFOR data points as JSON objects can be useful for users’ applications

2. Extended EXFOR database can be used for translation EXFOR data to JSON to be initial input for users NoSQL database

3. Translation program can be configurable depending on user needs

4. Automatically created NoSQL clone of EXFOR database can simplify of JSON database maintenance.

Thank you.