# Scientific Analysis by Queries in Extended SPARQL over a Scalable e-Science Data Store

Andrej Andrejev, Salman Toor, Andreas Hellander*, Sverker Holmgren, Tore Risch

*Department of Information Technology, Uppsala University*
** Department of Computer Science, University of California Santa Barbara*

## The Scientific SPARQL Database Manager (SSDM) provides:

- *scalable* data management by utilizing **standard** relational database systems
- good *documentation* by utilizing the **standard** W3C data representation RDF
- querying experimental data by the **standard** W3C query language SPARQL extended with arrays, SciSPARQL
- software *reuse* by calling **standard** and custom libraries in queries (*foreign functions* support)

**Our project homepage:**
**http://www.it.uu.se/research/group/udbl/SciSPARQL**

- **software**
- **documentation**
- **examples**

## Some interesting facts about SciSPARQL queries

Variable `?i` is bound to all and only valid subscripts that satisfy the filter expression

*"sum up certain rows of given* **?U** *array"*

```
DEFINE FUNCTION total_species(?U ?species ?mspecies)
AS SELECT (SUM(?U[?i])) AS ?res)
    WHERE { FILTER (mod(?i, ?mspecies) = ?species-1) };
```

*"Compute the sum of all species 'A' over all mesh cells as a function of time for task* `:Task1`*"*

Q1

```
SELECT (?tspan[?j] AS ?t)
       (total_species(?U,?a,?mspecies)[?j] AS ?sum_A)
 WHERE { :Task1 :U ?U ;
                :inExperiment ?experiment .
         ?experiment :A ?a ;
                     :MSpecies ?mspecies ;
                     :tspan ?tspan };
```

find data and metadata in the RDF graph

*"Find the task that has the maximal total population of species A or B for any time point"*

Q3

```
DEFINE FUNCTION max_AB_sum(?task) AS
SELECT
   (max(array_max(total_species(?U,?a,?mspecies)),
        array_max(total_species(?U,?b,?mspecies)))
   AS ?res)
 WHERE { ?task :U ?U ;
                :inExperiment ?experiment .
         ?experiment :A ?a ;
                     :B ?b ;
                     :MSpecies ?mspecies };

SELECT (ARGMAX( max_AB_sum ) AS ?maxtask);
```

Applying second-order function **ARGMAX**

*"Select the sum of all species A for time point 10s for all tasks with parameters k_a and k_d in given ranges"*

Q2

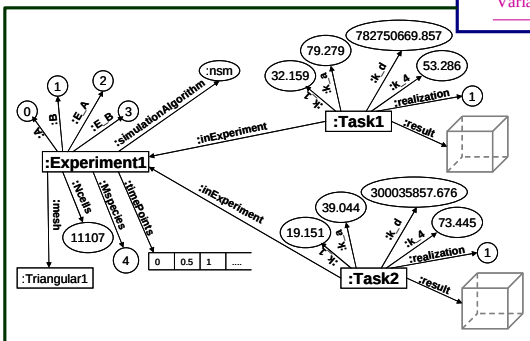
```
SELECT (array_sum(?U[?a-1::?mspecies,?j]) AS ?res)
 WHERE { ?task :U ?U ;
               :k_a ?k_a ;
               :k_d ?k_d ;
               :inExperiment ?experiment .
         ?experiment :A ?a ;
                     :MSpecies ?mspecies ;
                     :tspan ?tspan .
          FILTER (?tspan[?j] = 10 &&
                  1.0E8 <= ?k_d && ?k_d <= 1.0E9 &&
                  50 <= ?k_a && ?k_a <= 90 ) };
```

First result is returned right after first **?U** matrix with metadata that fall into the specified range is found and processed

Variable `?j` is bound to position of value **10** in the array

## Q2 in MATLAB

```
sum_of_A = [];
load('input.mat'); % parameters, tspan 'metadata
t = find(tspan==10);
a = 1; % this 'metadata' is not stored anywhere
mspecies = 8;
for ii=1:100 % amount of files should be known!
    if parameters(1,ii) >= 50
       && parameters(1,ii) <= 90
       && parameters(3,ii) >= 1.0E8
       && parameters(3,ii) <= 1.0E9
      realization = strcat( % consruct filenames
        'C:/DATA/bistab2f/realization_',
        int2str(ii),'_1.mat');
      load(realization); % load matrices 1-by-1
      sum_of_A = [sum_of_A
                  sum(UU(a:mspecies:end, t))];
    end
end
sum_of_A;
```




The experimental data is loaded into SSDM as:

- A text file (Turtle) with metadata
- Binary files with array data

```
@prefix : <http://udbl.uu.se/bistab#> .

:Experiment1 :MSpecies 8; :NCells 11107;
             :A 1; :B 2; :E_A 3; :E_B 4; :tlen 201;
             :tspan (0 0.5 1 1.5 2 2.5 ...)
:Task1 :realization 1;
       :inExperiment :Experiment1 ;
       :k_1 19.1508261629657; :k_a 39.0437365261002;
       :k_d 300035857.67601; :k_4 73.4445726669338;
       :U <file://realization_1_1.mat:matlab#U&0> .

:Task2 :realization 1; :inExperiment :Experiment1 ;
       :k_1 32.158566873436; :k_a 79.2782912066599;
       :k_d 782750669.857608; :k_4 53.2865295525529;
       :U <file://realization_2_1.mat:matlab#U&0> .

...
```

A list of numbers is recognized as an array

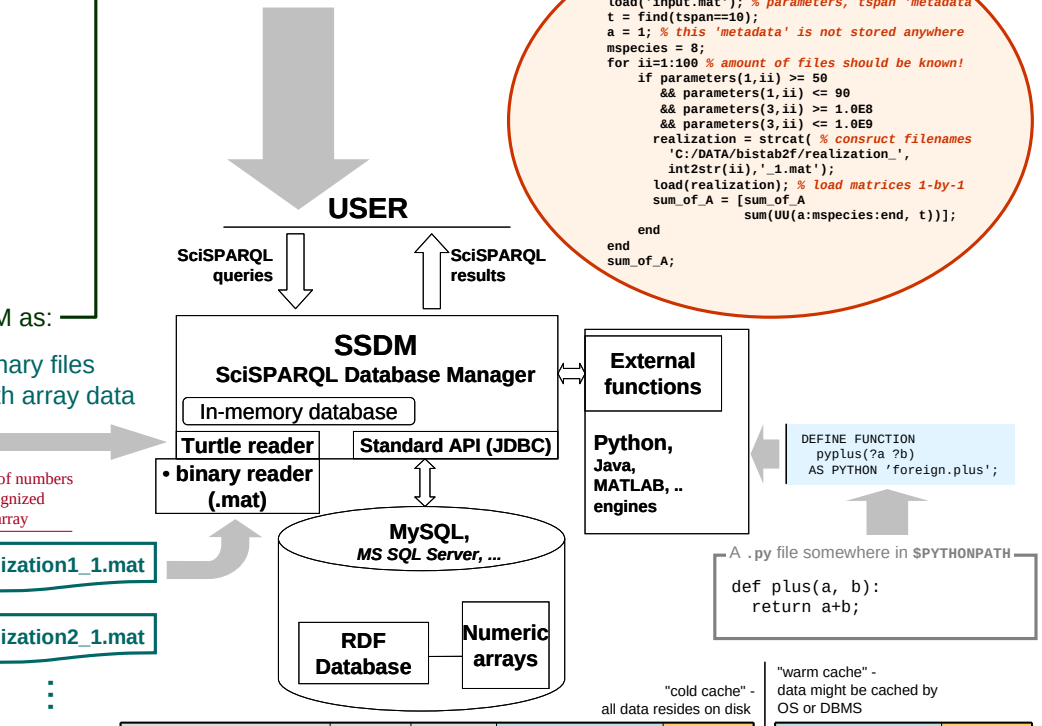
realization1_1.mat

realization2_1.mat

⋮

globally-unique namespace: every id that begins with `:` belongs to it

reader id: `matlab`
# reader parameters:
• variable: **U**
• desired type: integer




Data loading times for a sample of 100 matrices (7.15 GB)

| Task | MySQL | MS SQL Server |
|---|---|---|
| Preparing files for bulk-loader | 980 s | 82 s |
| Bulk loading | 1 543 s | 1 275 s |
| **Total** | **2 523 s** | **1 357 s** |
| Naïve one-by-one insertion | 7 577 s | 7 827 s |

"cold cache" - all data resides on disk

| Task | Data retrieved | Chunk utilization | SSDM with back-end | | MATLAB script |
|---|---|---|---|---|---|
| | | | MySQL | MS SQL Server | |
| **Q1: (selective query)** Compute an aggregate value over 1 big matrix, every 8th row | **18MB** | **50%** | **1.748** | **2.15** | **1.826** |
| **Q2: (SSDM worst case)** Select 36 matrices, access one column × every 8th row | **642MB** | **0.25%** | **80.703** | **44.512** | **30.042** |
| **Q3: (database scan)** Compute AGRMAX of Q1 across all matrices, 25% rows | **1785MB** | **100%** | **187.073** | **192.365** | **133.279** |

"warm cache" - data might be cached by OS or DBMS

| SSDM with back-end | | MATLAB script |
|---|---|---|
| MySQL | MS SQL Server | |
| **0.434 (0.138)*** | **0.526 (0.152)*** | ***0.157*** |
| **63.542** | **13.378** | **1.203** |

* all data fits into SSDM cache, back-end is not accessed