

Enterprise Information Systems Lab

## **Linked Data Visualization and Exploration Technical System Documentation**

Mentor: Klaudia Thellmann, Michael Galkin

[This document describes the technical system specification for implementing LinDa Visualization Web Application Software]

Students:

Alina Arunova, Tatiana Novikova, David Ibhaluobe

## Table of Contents

1. Generalities .....	2
1.1. Overview .....	2
1.2. Reference .....	2
1.3. Definitions, Acronyms, Abbreviations.....	2
2. About LinDA Visualization Tool.....	4
3. Project Description.....	5
3.1. Use Case Diagram: .....	5
3.2. Scenario:.....	6
4. Requirements.....	12
4.1. Functional Requirements:.....	12
4.2. Non - Functional requirements:.....	13
5. Technical Software Requirement.....	14
5.1. Software Aspects .....	14
6. Operational Specification.....	15
6.1. Installing Git .....	15
6.2. Installing NodeJS .....	15
6.3. Installing EmberJS .....	15
6.4. Installing Virtuoso 7 .....	15
6.5. Installing Bower.....	16
6.6. Installing LinDA Visualization .....	17
6.7. Installing Nodemon.....	17
6.8. Docker .....	17
7. List of Datasets.....	19
8. Implementation .....	23
8.1. Horizontal Scrolling .....	23
8.2. Keyword-based search.....	23
8.3. Interface Design and Functionality – Filtering .....	24
9. Test cases .....	25
External References .....	27

# 1. Generalities

## 1.1. Overview

The goal of this project is to improve some aspect of LinDa Visualization Tool. The aspect improved are: Key-based search in a tree; filtering for numerical fields from the tree and scrolling for the Data-Table. With this improvements the software will become more powerful and flexible for end-users' needs.

## 1.2. Reference

The following are the reference document names:

- Linked Data Visualization and Exploration Tools Requirement Presentation
- Linked Data Visualization and Exploration Tools Requirement Document
- Linked Data Visualization and Exploration Tools Architectural presentation
- Linked Data Visualization and Exploration Tools Use Case Presentation
- Linked Data Visualization and Exploration Tools Documentation Presentation

These documents are available on github. Below is the link:

<https://github.com/EIS-Bonn/MA-INF3232-Lab-SS2015/tree/master/GroupD/Presentations>

## 1.3. Definitions, Acronyms, Abbreviations

- SME: Small and Medium Enterprises.
- LinDA: Linked Data Analytics.
- RDF: Resource Description framework is a family of World Wide Web

Consortium (W3C) specifications used as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntax notations and data serialization formats.

- Data-Table: Any display of information in a tabular form, with rows and/or columns.
- Filtering: refers to a wide range of solutions for refining data sets.
- Dataset: A named collection of data that contains individual data units organized (formatted) in a specific way.
- Searching: Any data structure that allows the efficient retrieval of specific items from a set of items.
- Paging: Is a method of writing data to, and reading it from, secondary storage for use in primary storage, also known as main memory.
- Sorting: Is any process of arranging items systematically, and has two common, yet distinct meaning.

## 2. About LinDA Visualization Tool

The aim of the LinDA project is to make the benefits of Linked Open Data accessible to SMEs and data providers by providing libraries for Open Data consumption. One of the main tasks in this context is to build an ecosystem of tools for visualizing Linked Data to assist SMEs in their daily tasks by hiding complexity through automation and an intuitive user interface.

To complete this task, a generic visualization work-flow is being implemented based on state-of-the-art Linked Data visualization approaches. Most existing approaches are only usable by a technical audience or limited to certain domains or data representations. LinDA proposes a generic approach for visualization selection in form of a faceted browser that imposes on the user the task of describing the visualization at an unfamiliar level of abstraction.

### 3. Project Description

The project is aim at improving some aspect of LinDA Visualization Tool.

#### 3.1. Use Case Diagram:

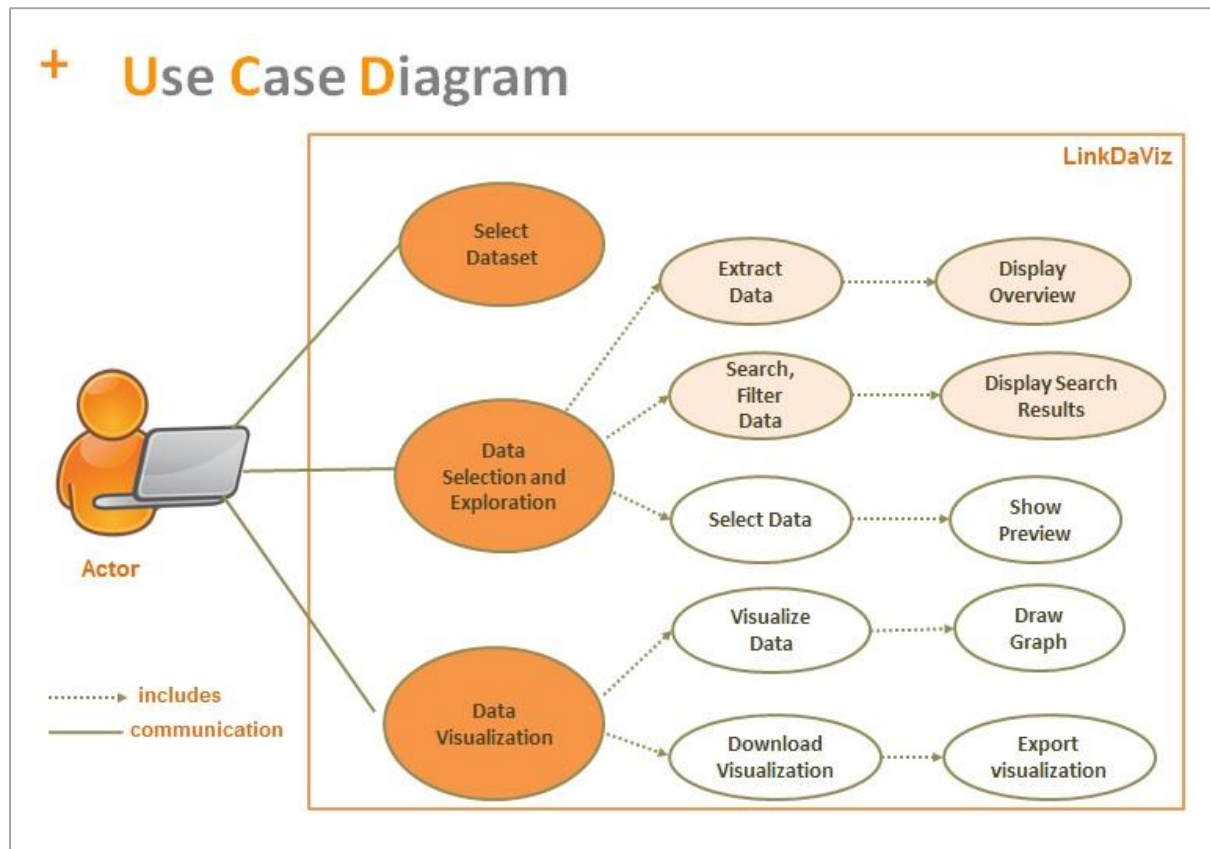


Image 1. Use case diagram

**Actor:** A UML element representing the role of a person, object or device that interacts with the system.

**Use Case Diagram:** A UML behavior diagram that visually describes the functional requirements of a proposed system and shows the relationships between Actors and Use Cases. A Use Case diagram is mainly formed by Actors, Use Cases and Associations (connectors).

## 3.2. Scenario:

### STEP 0: Selecting a dataset

User selects a RDF dataset containing statistical data for exploration and visualization (Image 2).

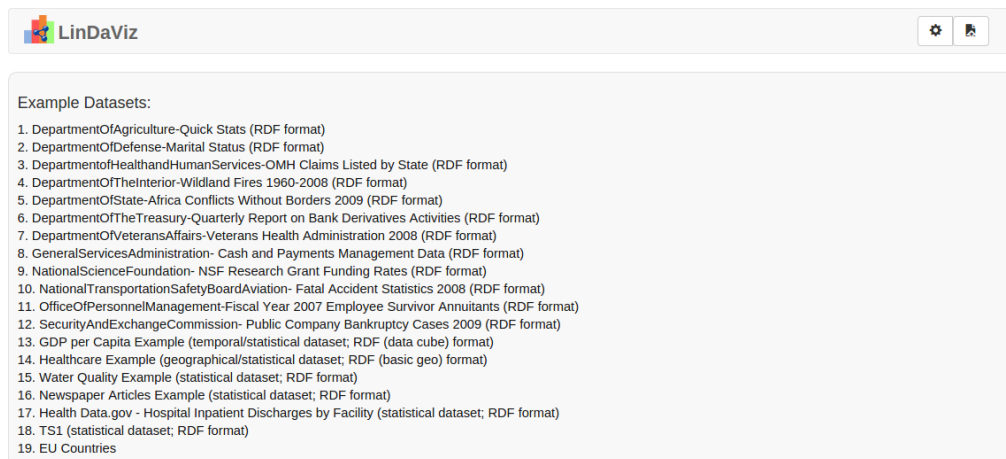


Image 2. List of datasets

### STEP 1: Data selection and exploration

A preview is generated by the Data Tree consisting of classes (Image 3), their properties (Image 4) and a filter tab for choosing the exact data (Image 4).

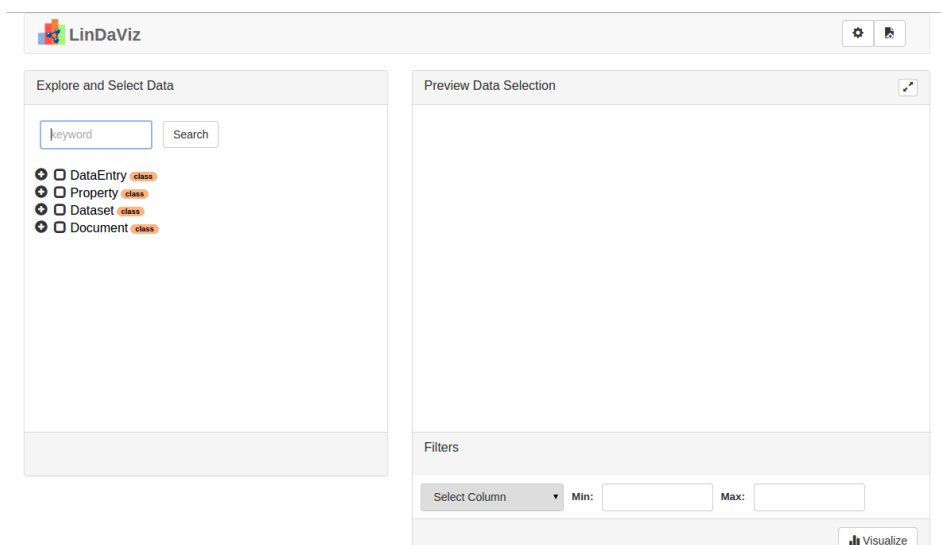


Image 3. Data Tree consisting of classes

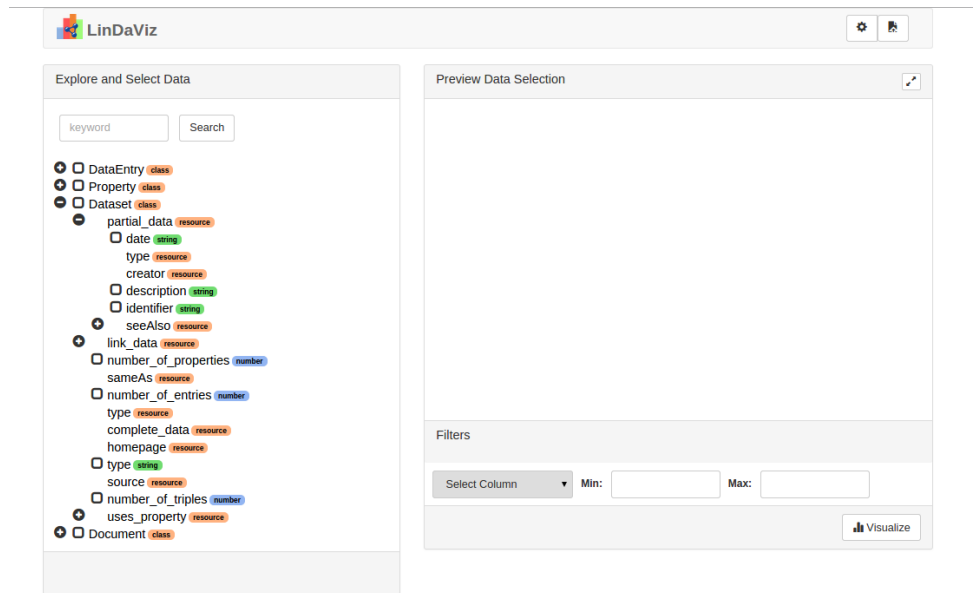


Image 3. Data Tree consisting of classes and properties

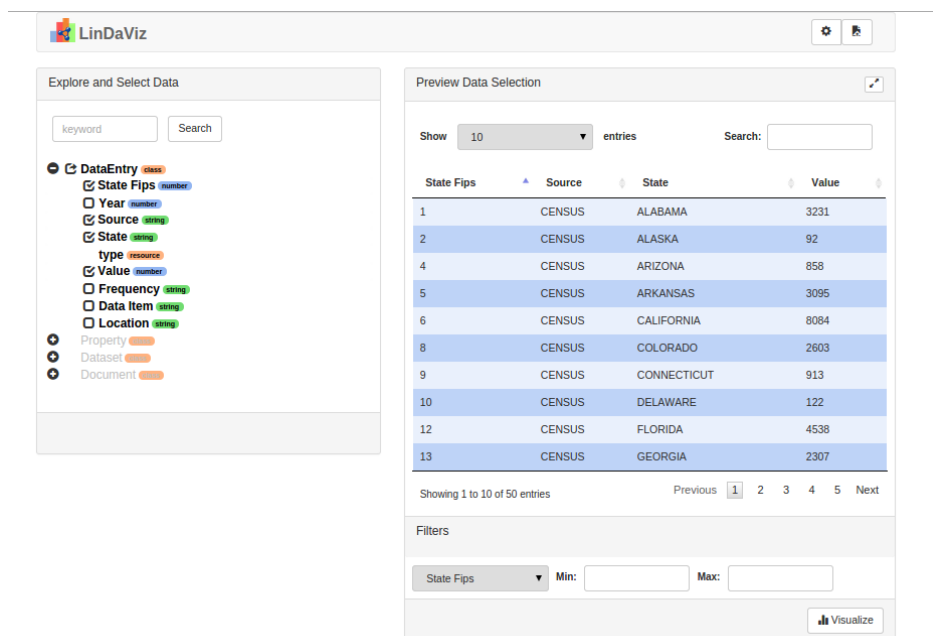


Image 4. Filter tab for choosing extract data

## STEP 2: Data filtering/searching

The User inspects the Data Tree and proceeds with filtering the data properties from classes he/she is interested in and by searching for classes, properties and subproperties containing a certain keyword.



## Filtering:

A user selects data which he wants to visualize, the data appears in a data table, and the filter dropdown list is being filled with corresponding column names.

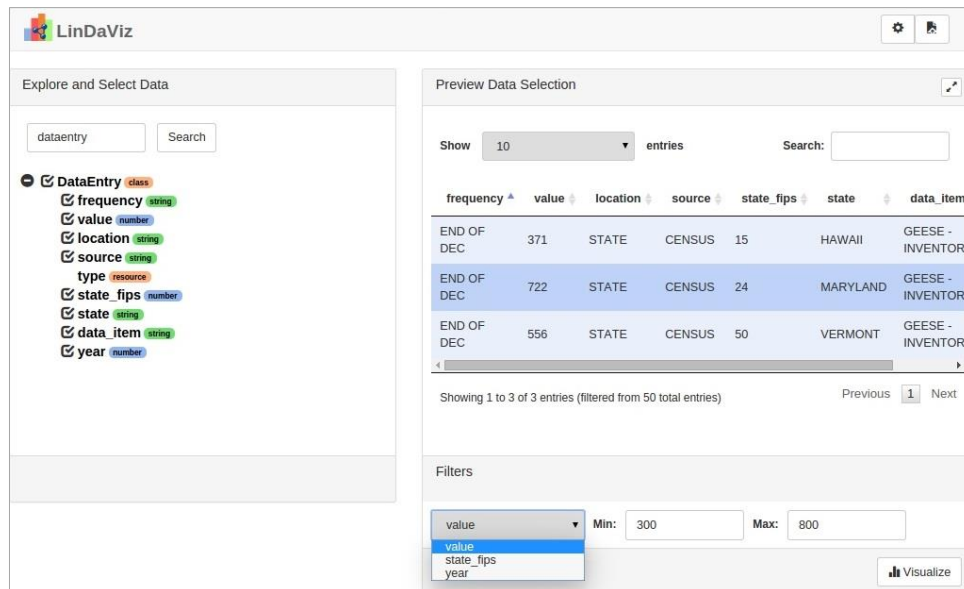


Image 5. List item

Then a user chooses a list item (Image 5), types in min and max values and sees how the table changes in real-time according to the selected criteria (Image 6).



Image 5. Filter parameters

## Searching:

A user types a keyword in a text field and presses a button “search” if there is such word in the dataset then a new tree with search results will be displayed (Image 6), otherwise “No matches” will be displayed (Image 7). If the user enters too short word, a hint will be displayed (Image 8).

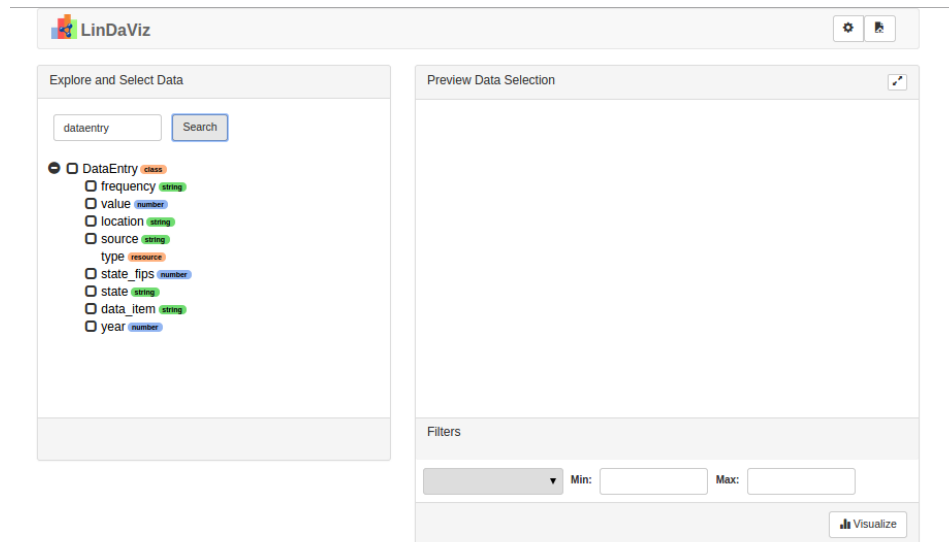


Image 6. Search results

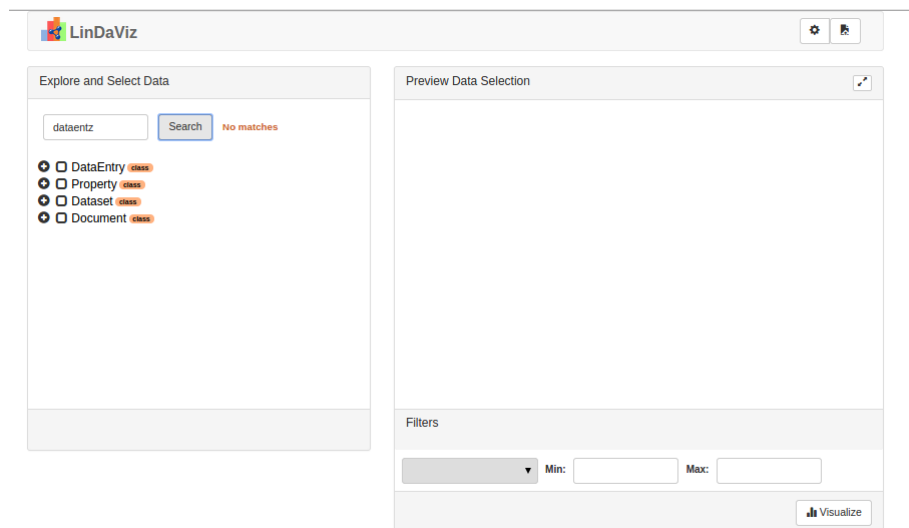


Image 7. Keyword was not found

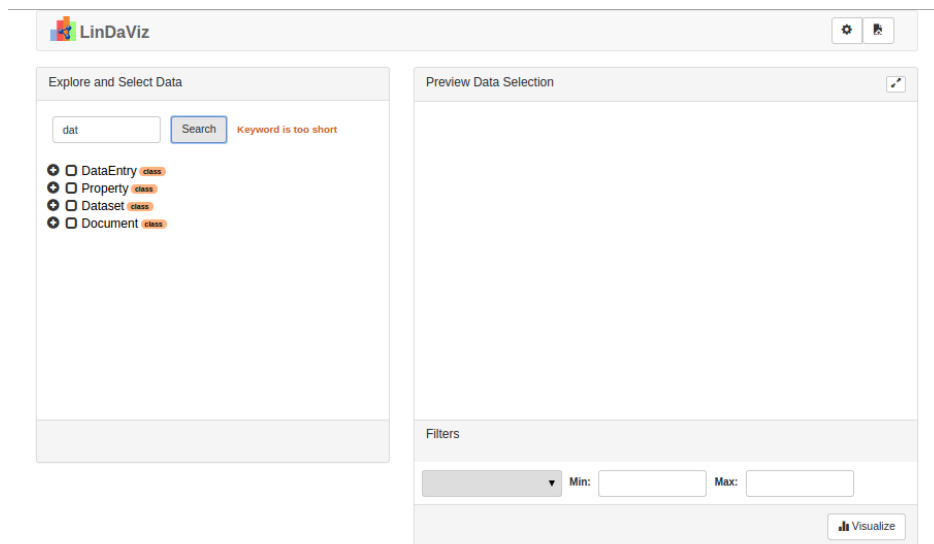


Image 8. Keyword is too short

### **STEP 3: Data table overview**

The data properties are computed and showed to a User as a table. If there is more than 7 columns, the table becomes scrollable (Image 9).

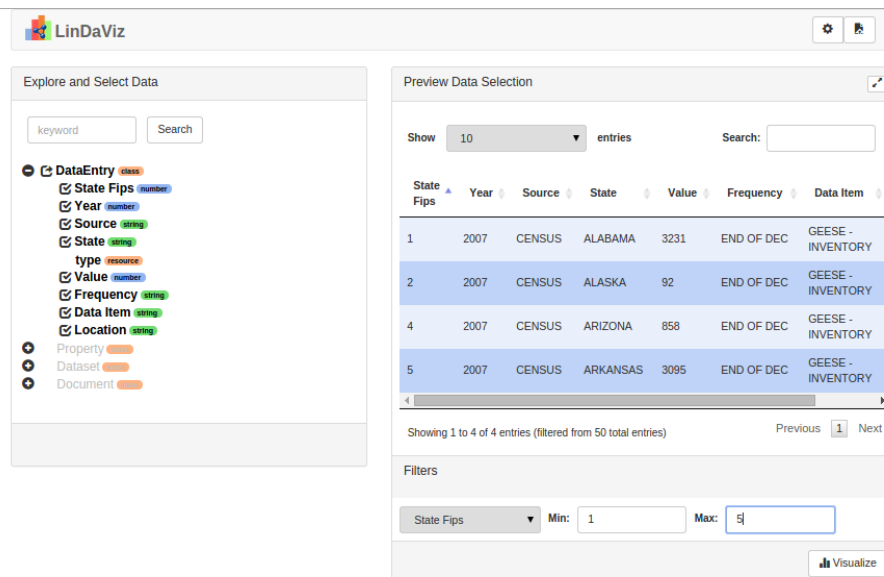


Image 9. Horizontal scrolling

## STEP 4: Visualizing the data

The User inspects the results and proceeds with configuring and visualizing the selection (Image 10).

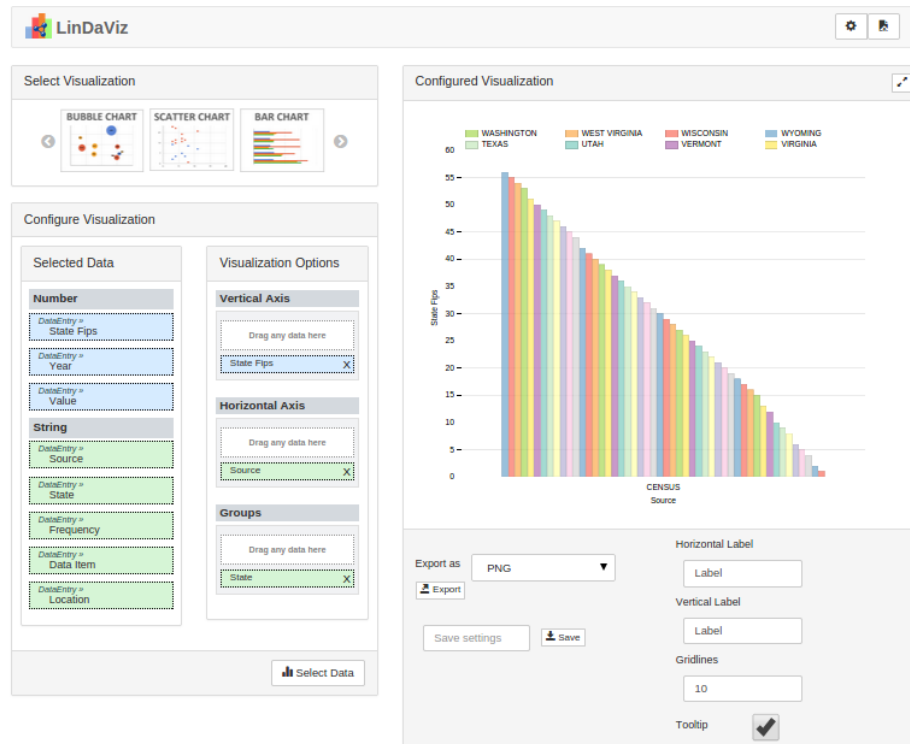


Image 10. Visualization

## 4. Requirements

### 4.1. Functional Requirements:

REQ_ID	REQ_NAME	DESCRIPTION	PRIORITY
1	<b>Display overview</b>	Improvement of display overview by decreasing the number of columns in the data table	3
2	<b>Sorting</b>	Provide sorting for the Data Tree (e.g. by A-Z)	3
3	<b>Filtering</b>	Provide filtering for the data properties from the Data Tree	1
4	<b>Searching</b>	Provide means for searching by keywords	1
5	<b>Hiding/Loading</b>	Provide the possibility to hide/load classes in the Data Tree for improving the Tree overview	2
6	<b>Export data</b>	Data table export selection to .csv format	3

## 4.2. Non - Functional requirements:

REQ_ID	REQ_NAME	DESCRIPTION	PRIORITY
2	<b>Data table scalability</b>	The ability to scale large data in order to get the full overview. (e.g. by scrolling)	1
3	<b>Usability (Search, Filter data)</b>	Easy to use search for filtering classes and properties	1
4	<b>Technical System Documentation</b>	The document must contain the whole procedure of testing (quick-reference guides)	1
7	<b>Backup</b>	The ability to restore the original data after the loss (copying, archiving)	4
8	<b>Robustness</b>	The ability of the system to deal with errors and normally operate despite abnormalities in input, calculations	5

## 5. Technical Software Requirement

### 5.1. Software Aspects

The following are the software requirement for LinDA visualization tool:

- Git
- Nodejs
- Emberjs
- Virtuoso 7
- Bower
- LinDA
- Ubuntu
- Nodemon

## 6. Operational Specification

### 6.1. Installing Git

The git can be installed by the following command in Ubuntu:

```
sudo apt-get install git-core
```

Git is mainly used to download the LinDA software from the GitHub repository.

### 6.2. Installing NodeJS

The following commands are used to install Nodejs in Ubuntu:

```
sudo apt-get update
```

```
sudo apt-get install nodejs
```

```
sudo apt-get install npm
```

```
npm install -g nodemon
```

### 6.3. Installing EmberJS

The link below is where you can steps for installing emberjs:

<http://guides.emberjs.com/v1.12.0/getting-started/>

Ember.js installs through the npm. Install the Ember.js build tools with npm.

The following commands are used to install Emberjs in Ubuntu:

```
npm install -g ember-cli
```

### 6.4. Installing Virtuoso 7

- Virtuoso 7 can be downloaded from the below link:

<http://sourceforge.net/projects/virtuoso/files/virtuoso/7.1.0/>

- Navigate to the downloaded folder.
- Extract the package.
- Run the following comand in Ubuntu:



```
sudo apt-get install libssl-dev
```

- Go the extracted package location. It is better to run autogen.sh by typing ./autogen.sh, which checks for the presence and right version of some of the required components, and if it reports any missing package then, install that package.
- Set following environmental variable by typing

```
CFLAGS="-O2"
```

```
export CFLAGS
```

- Execute './configure' command to configure the package for your system.
- Execute 'sudo make' command to compile the package
- Type 'sudo make install' to install the programs and any data files and documentation.
- Start the Virtuoso Server by the following step:

```
sudo /usr/local/virtuoso-7.1.0/bin/virtuoso-t +config /usr/local/virtuoso-7.1.0/var/lib/virtuoso/db/virtuoso.ini
```

## 6.5. Installing Bower

The link below is where you can steps for installing bower:

Bower installs through the npm. Install the Ember.js build tools with npm.

The following commands are used to install Bower in Ubuntu:

```
npm install -g bower
```

Bower depends on Nodejs and npm. Also make sure that git is installed as some bower packages require it to be fetched and installed.

## 6.6. Installing LinDA Visualization

The LinDA Visualization tool can be installed in Ubuntu by the following commands:

```
git clone https://github.com/LinDA-tools/visualisation.git
cd visualisation/frontend
npm install
bower install
cd ../backend
npm install
```

The LinDA can be started by the following commands:

```
nodemon & (from linda-viz-be)
ember server (from linda-viz-fe)
sudo /usr/local/virtuoso-7.1.0/bin/virtuoso-t +config /usr/local/virtuoso-7.1.0/var/lib/virtuoso/db/virtuoso.ini
```

## 6.7. Installing Nodemon

Nodemon is installed through the npm. Install the Nodemon build tools with npm.

The following commands are used to install Nodemon in Ubuntu:

```
npm install Nodemon
```

## 6.8. Docker

As showed on Image 1, Docker allows you to package an application with all of its dependencies into a standardized unit for software development. Docker containers wrap up a piece of software in a complete file system that contains everything it needs to run: code, runtime, system tools, system libraries – anything you can install on a server. This guarantees that it will always run the

same, regardless of the environment it is running in.

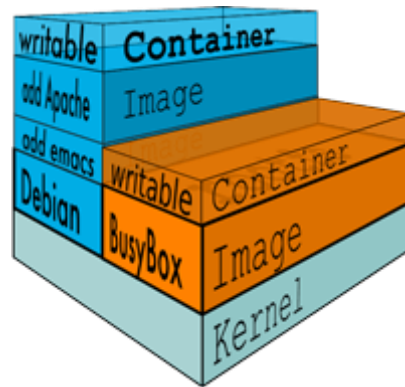


Image 1

## 7. List of Datasets

- Department of agriculture – Quick Stats

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentOfAgriculture-QuickStats.nt>

- Department of defense – Marital status

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentOfDefense-MaritalStatus.nt>

- Department of health and human services – OMH claims listed by state

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentofHealthandHumanServices-OMHClaimsListedbyState.nt>

- Department of the interior – Wildland fires 1960-2008

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentOfTheInterior-WildlandFiresand1960-2008.nt>

- Department of state Africa conflicts without borders 2009

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentofState-AfricaConflictsWithoutBorders2009.nt>

- Department of the treasury quarterly report on bank derivatives activities

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentOfTheTreasury-QuarterlyReportonBankDerivativesActivities.nt>

- Department of veterans affairs veterans health administration 2008

Type: rdf

Source: <http://www.linda-project.org/examples/DepartmentOfVeteransAffairs-VeteransHealthAdministration2008.nt>

- General services administration cash and payments management data

Type: rdf

Source: <http://www.linda-project.org/examples/GeneralServicesAdministration-CashandPaymentsManagementData.nt>

- National science foundation NSF research grant funding rates

Type: rdf

Source: <http://www.linda-project.org/examples/NationalScienceFoundation-NSFResearchGrantFundingRates.nt>

- National transportation safety board aviation accidents, fatalities, and rates, 2008

Type: rdf

Source: <http://www.linda-project.org/examples/NationalTransportationSafetyBoardAviation-AccidentStatistics2008.nt>

- Office of personnel management fiscal year 2007 employee and survivor annuitants by geographic distribution

Type: rdf

Source: <http://www.linda-project.org/examples/OfficeofPersonnelManagement-FiscalYear2007EmployeeSurvivorAnnuitants.nt>

- Security and exchange commission public company bankruptcy cases opened and monitored for fiscal year 2009

Type: rdf

Source: <http://www.linda-project.org/examples/SecurityAndExchangeCommission-PublicCompanyBankruptcyCases2009.nt>

- GDP per capita example

Type: rdf (data cube)

Source: <http://www.linda-project.org/examples/worldbank-slice-5000>

- Healthcare example

Type: rdf

Source: [http://www.hospitals\\_reviewer.com/2014](http://www.hospitals_reviewer.com/2014)

- Water quality example

Type: rdf

Source: [http://water\\_quality\\_check.it/info](http://water_quality_check.it/info)

- Newspaper articles example

Type: rdf

Source: [http://newspaper.org/articles\\_2007](http://newspaper.org/articles_2007)

- Health data.gov – Hospital inpatient discharges by facility

Type: rdf

Source: <http://HealthData.govHospitalInpatientDischargesbyFacility>

- TS1

Type: rdf

Source: [http://www.linda-project.org/TS1\\_LinearRegression\\_Result\\_Original](http://www.linda-project.org/TS1_LinearRegression_Result_Original)

- EU countries

Type: rdf

Source: <http://eucountries.org>

## 8. Implementation

### 8.1. Horizontal Scrolling

As we simulate a work-flow to see the behavior of the data-table; we discover that as a dataset is selected while the instance data property is queried from the back-end and display as a column on the right side, the more data you select the larger the table gets and grows out of proportion.

Our task is to look at how to improve on this. We introduced horizontal scrolling in the data-table since there was already a data-source controller and handler template. We implemented this in the data-table component of the LinDA project that is basically an integrated JavaScript plug-in which is used to generating preview. We achieved this through data-table [examples](#) extension. This will enable users view hidden columns.

### 8.2. Keyword-based search

There may be occasions when a user wants to find a keyword in a data tree. As soon as the user typed the keyword and pressed a button “Search” ‘*search action*’ in *datasource.js* controller is activated. It invokes a function *search* in *tree-selection-data-module.js*. Then there is checking if a length of the keyword is more than 3 or not. If the length is less than 4, null is returned and an initial tree is displayed. Otherwise, a promise is returned and *fullTextSearch* function from *sparql-data-module.js* is invoked. A query with the keyword is executing and after execution the results are got. The next step is parsing the results in a tree structure as the initial tree has. There are two steps: 1) *find children* function for properties and subproperties; 2) creating roots (classes) and adding children there; Roots are returned as the result of *fullTextSearch* function and *createSearchResultTreeContent* function is invoked and after executing results for visualization are returned.



### 8.3. Interface Design and Functionality – Filtering

There may be occasions when the end user wishes to narrow down the data shown in the Data-Table based on specific criteria. A common example is the number range search (in between two numbers). DataTables<sup>1</sup> provides an API method `$.fn.dataTable.ext.search` that allows to implement custom search function in our project.

New filtering function with lower and upper limits for numeric columns in Linda Data Table was implemented.

---

<sup>1</sup> <http://datatables.net> - is a plug-in for the jQuery Javascript library.

## 9. Test cases

No	Functionality	Test case description	Expect results	Passed/not passed
1	Scrolling			Passed
				Passed
2	Filtering	Agriculture dataset is chosen. Selecting the following fields from the tree: value, year, state-fips, location and state. The filter drop-down list is being filled with only numerical items: value, year and state-fips.	The filter drop-down list contains only numerical fields.  When the min and max values for one particular field are chosen, the data table changes automatically with corresponding limits.	Passed
		Agriculture dataset is chosen. Selecting the following fields from the tree: value, year, state-fips, location and state. The filter drop-down list is being filled with only numerical items: value, year and state-fips. Deselecting value and year fields, the filter drop-down list will show only year field.	Deselecting some numerical fields, the filter drop-down list will show only fields which are selected in the tree.	Passed
		Agriculture dataset is chosen. Selecting the following fields	Changing selection of numerical fields	Passed

		from the tree: value, year, state-fips, location and state. The filter drop-down list is being filled with only numerical items: value, year and state-fips. Firstly year field is chosen, min and max values are typed in, the data table changed automatically, then value field is chosen, min and max values are typed in, the data table changed automatically.	will allow to type in min and max values again and the data table will change continuously, depending on what numerical field is now selected.	
<b>3</b>	<b>Searching</b>			<b>passed</b>
				<b>Passed</b>
				<b>Passed</b>
				<b>Passed</b>

## External References

- <https://datatables.net/reference/option/scrollX>
- <http://guides.emberjs.com/v1.12.0/getting-started/>
- <https://www.npmjs.com/package/bower>
- <http://sourceforge.net/projects/virtuoso/files/latest/download?source=files>
- <http://www.rdfdata.org/>
- <https://github.com/LinDA-tools/Visualization/tree/master/backend/testsets>
- <http://www.w3.org/wiki/DataSetRDFDumps>