

# Data Versioning to enable Identification and Citability of Dynamic Data: Recommendations of the RDA Working Group on Data Citation

Andreas Rauber

Vienna University of Technology  
Favoritenstr. 9-11/188  
1040 Vienna, Austria  
[rauber@ifs.tuwien.ac.at](mailto:rauber@ifs.tuwien.ac.at)  
<http://www.ifs.tuwien.ac.at/~andi>

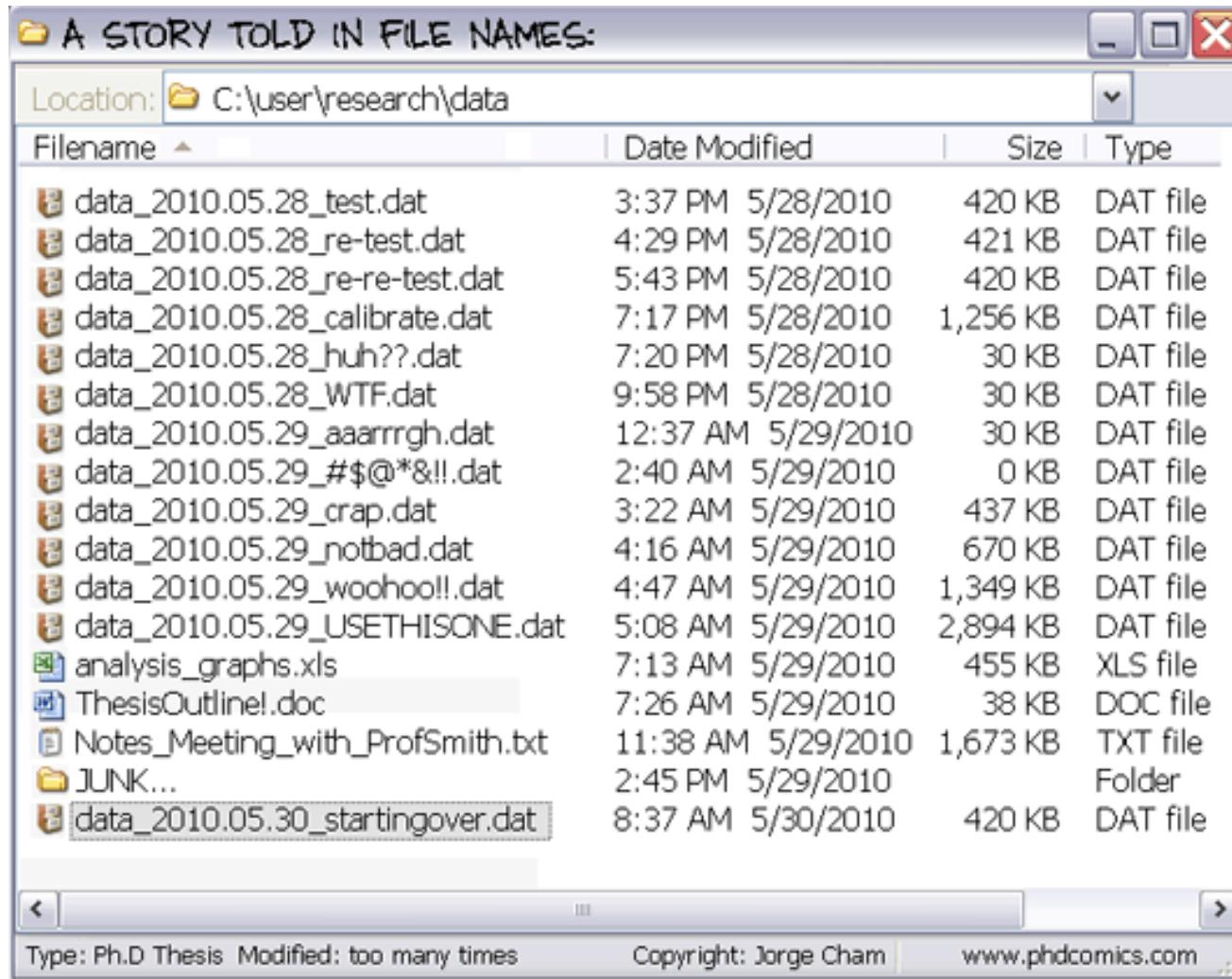
- Would like to identify precisely the **data as it existed at a specific point in time**
- Would like to be able to identify precisely the **subset of (dynamic) data used** in a process

# What we do NOT want...

- Avoid this...

(from PhD Comics: A Story Told in File Names, 28.5.2010)

Source: <http://www.phdcomics.com/comics.php?f=1323>



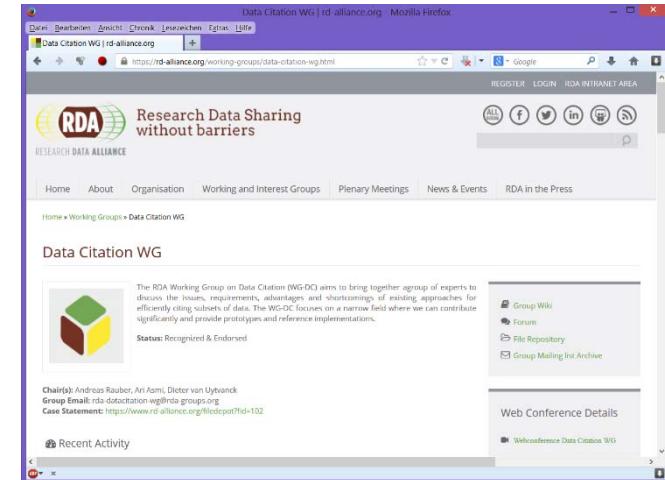
The screenshot shows a Windows-style file explorer window. The title bar reads "A STORY TOLD IN FILE NAMES". The location bar shows "C:\user\research\data". The main area is a grid view of files:

Filename	Date Modified	Size	Type
data_2010.05.28_test.dat	3:37 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_re-test.dat	4:29 PM 5/28/2010	421 KB	DAT file
data_2010.05.28_re-re-test.dat	5:43 PM 5/28/2010	420 KB	DAT file
data_2010.05.28_calibrate.dat	7:17 PM 5/28/2010	1,256 KB	DAT file
data_2010.05.28_huh???.dat	7:20 PM 5/28/2010	30 KB	DAT file
data_2010.05.28_WTF.dat	9:58 PM 5/28/2010	30 KB	DAT file
data_2010.05.29_aaarrgh.dat	12:37 AM 5/29/2010	30 KB	DAT file
data_2010.05.29_#\$@*&!!..dat	2:40 AM 5/29/2010	0 KB	DAT file
data_2010.05.29_crap.dat	3:22 AM 5/29/2010	437 KB	DAT file
data_2010.05.29_notbad.dat	4:16 AM 5/29/2010	670 KB	DAT file
data_2010.05.29_woohoo!.dat	4:47 AM 5/29/2010	1,349 KB	DAT file
data_2010.05.29_USETHISONE.dat	5:08 AM 5/29/2010	2,894 KB	DAT file
analysis_graphs.xls	7:13 AM 5/29/2010	455 KB	XLS file
ThesisOutline!.doc	7:26 AM 5/29/2010	38 KB	DOC file
Notes_Meeting_with_ProfSmith.txt	11:38 AM 5/29/2010	1,673 KB	TXT file
JUNK...	2:45 PM 5/29/2010		Folder
data_2010.05.30_startingover.dat	8:37 AM 5/30/2010	420 KB	DAT file

Type: Ph.D Thesis Modified: too many times Copyright: Jorge Cham [www.phdcomics.com](http://www.phdcomics.com)

# RDA WG Data Citation

- Research Data Alliance
- WG on **Data Citation:**  
**Making Dynamic Data Citeable**
- March 2014 – September 2015
  - Concentrating on the problems of  
**large, dynamic (changing) datasets**
- Final version presented Sep 2015  
at P7 in Paris, France
- Endorsed September 2016  
at P8 in Denver, CO



<https://www.rd-alliance.org/groups/data-citation-wg.html>

- **We have**
  - Data & some means of access („query“)
- **Make data: time-stamped and versioned**
- Prepare some way of **storing the queries with timestamp**
- **Data Citation:**
  - Store query with timestamp
  - Assign the PID to the timestamped query  
(which, dynamically, leads to the data)
- **Access:**

Re-execute query on versioned data according to timestamp
- **Dynamic Data Citation:**

Dynamic data & dynamic citation of data

# Data Citation – Output

- 14 Recommendations grouped into 4 phases:
  - Preparing data and query store
  - Persistently identifying specific data sets
  - Resolving PIDs
  - Upon modifications to the data infrastructure
- 2-page flyer  
<https://rd-alliance.org/recommendations-working-group-data-citation-revision-oct-20-2015.html>
- More detailed report: IEEE TCDL 2016  
[http://www.ieee-tcdl.org/Bulletin/v12n1/papers/IEEE-TCDL-DC-2016\\_paper\\_1.pdf](http://www.ieee-tcdl.org/Bulletin/v12n1/papers/IEEE-TCDL-DC-2016_paper_1.pdf)

**Data Citation of Evolving Data**

Recommendations of the Working Group on Data Citation (WGDC)  
 Jérôme Lacoste, Alain Lévy, University of Paris-Est Marne-la-Vallée, France  
 David Lutz, University of Vienna, Austria

Version 1.0, 2015-10-20

**1. Preparing Data Objects**

- 1.1. Data Versioning: Apply versioning to data objects to ensure that data can be tracked over time, based on metadata, prior to any update or removal.
- 1.2. Descriptions of the objects: Create a detailed description of the objects, including their provenance, dependencies, and the steps or outcomes of any recent, ongoing, or planned changes, and store it in a versioned object.

**2. Identifying Specific Data Sets**

- 2.1. Data Identifiers: Assign unique identifiers to the data objects, use them consistently across different versions, and make them persistent.
- 2.2. Data Descriptions: Create a detailed description of the data objects, including their provenance, dependencies, and the steps or outcomes of any recent, ongoing, or planned changes, and store it in a versioned object.
- 2.3. Data Access: Ensure that the data objects are accessible via well-known URLs or other standard access mechanisms.
- 2.4. Data Publishing: Publish the data objects in a structured format, such as JSON-LD, and include links to the descriptions and access mechanisms.

**3. Resolving PIDs**

- 3.1. Persistent Identifiers: Assign unique, persistent identifiers to the data objects, such as DOI, PURL, or ARK.
- 3.2. Data Descriptions: Create a detailed description of the data objects, including their provenance, dependencies, and the steps or outcomes of any recent, ongoing, or planned changes, and store it in a versioned object.
- 3.3. Data Access: Ensure that the data objects are accessible via well-known URLs or other standard access mechanisms.
- 3.4. Data Publishing: Publish the data objects in a structured format, such as JSON-LD, and include links to the descriptions and access mechanisms.

**4. Upon Modifications to the Data Infrastructure**

- 4.1. Data Descriptions: Create a detailed description of the data objects, including their provenance, dependencies, and the steps or outcomes of any recent, ongoing, or planned changes, and store it in a versioned object.
- 4.2. Data Access: Ensure that the data objects are accessible via well-known URLs or other standard access mechanisms.
- 4.3. Data Publishing: Publish the data objects in a structured format, such as JSON-LD, and include links to the descriptions and access mechanisms.

**Identification of Reproducible Subsets for Data Citation, Sharing and Re-Use**

Recommendations of the Working Group on Data Citation (WGDC)  
 Jérôme Lacoste, Alain Lévy, University of Paris-Est Marne-la-Vallée, France  
 David Lutz, University of Vienna, Austria

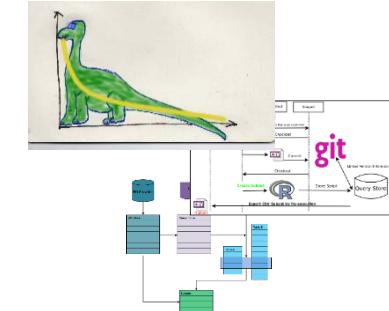
Version 1.0, 2015-10-20

**INTRODUCTION**

This document presents recommendations for the identification of reproducible subsets of data objects, which are intended to facilitate the citation, sharing, and reuse of data. It is based on the premise that data objects are often complex and may contain many components, some of which may be irrelevant or unnecessary for a particular purpose. By identifying and extracting reproducible subsets, it is possible to reduce the size and complexity of the data objects, making them easier to cite, share, and reuse. This document also provides guidance on how to handle changes to the data objects over time, ensuring that the reproducible subsets remain valid and relevant.

# Pilots / Adopters

- Several adopters
  - Different types of data, different settings, ...
  - CSV & SQL reference implementation (SBA/TUW)
- Pilots:
  - Biomedical BigData Sharing, Electronic Health Records  
(Center for Biomedical Informatics, Washington Univ. in St. Louis)
  - Marine Research Data  
Biological & Chemical Oceanography Data Management Office  
(BCO-DMO)
  - Vermont Monitoring Cooperative: Forest Ecosystem Monitoring
  - ARGO Boy Network, British Oceanographic Data Centre (BODC)
  - Virtual Atomic and Molecular Data Centre (VAMDC)
  - UK Riverflow Riverflow Archive, Centre for Ecology and Hydrology



FACULTY OF INFORMATICS

- Integrated
  - Extend original tables by temporal metadata
  - Expand primary key by record-version column
- Separated
  - Utilizes full history table
  - Also inserts reflected in history table
- Hybrid
  - Utilize history table for deleted record versions with metadata
  - Original table reflects latest version only
- SQL:2011 standard: time
  - Native support for timestamping
- Solution to be adopted depends on trade-off
  - Storage Demand
  - Query Complexity
  - Software adaption
- NOT: Audit tables (complex roll-back!) or static DB dumps

2 Reference implementations :

- **Git based Prototypes:** widely used versioning system
  - A) Using separate folders
  - B) Using branches
- **MySQL based Prototype:**
  - C) Migrates CSV data into relational database
- Data backend responsible for versioning data sets
- Subsets are created with scripts or queries via API or Web Interface
- Transparent to user: always CSV

Stefan Pröll, Christoph Meixner, Andreas Rauber

Precise Data Identification Services for Long Tail Research Data.

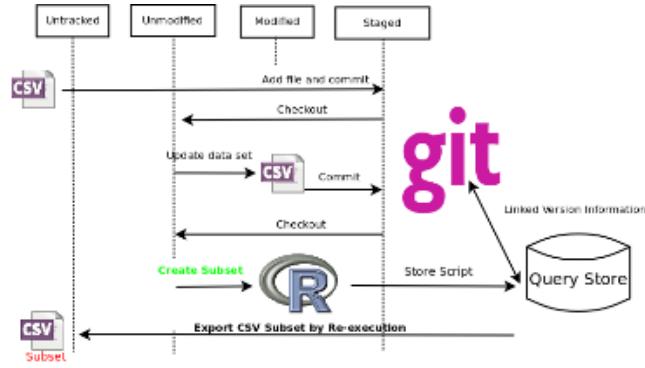
Proceedings of the intl. Conference on Preservation of  
Digital Objects (iPRES2016), Oct. 3-6 2016, Bern, CH..



FACULTY OF **INFORMATICS**

## Git Implementation

- Upload CSV files to Git repository (versioning)
- Subsets created via scripting language (e.g. R)
  - Select rows/columns, sort, returns CSV + metadata file
  - Metadata file with script parameters stored in Git
  - (Scripts stored in Git as well)
- PID assigned to metadata file
  - Use Git to retrieve proper data set version and re-execute script on retrieved file



```

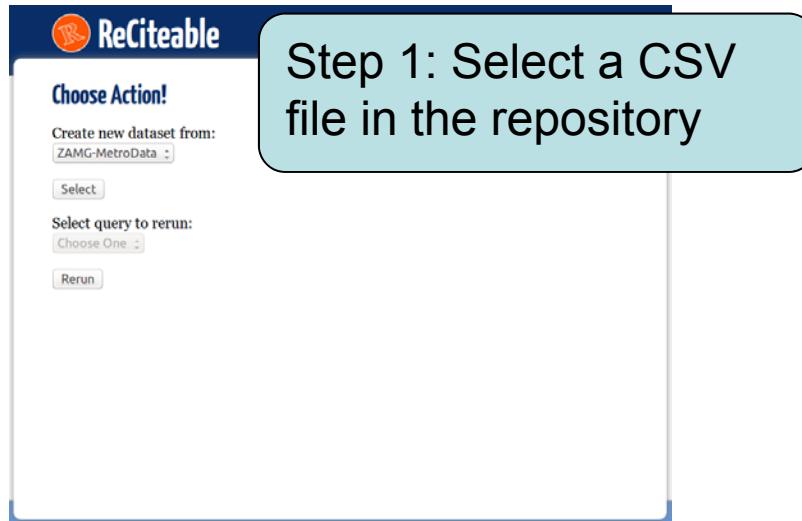
diff --git a/mallList.csv b/mallList.csv
index f9299f...fb6023 100644
--- a/mallList.csv
+++ b/mallList.csv
@@ -1,23 +1,21 @@
+supercomputer,words,datasetHeader,metaDataStringer,email
 0,C012120,0,204318299398862742,228518,gabtroy.wilkinson@gmail.com
 -1,Cat,ser,0,3886429613517521,227556,nels.witten@yahoo.com
 2,Dai,ramq,0,4725514338856503,738503,weise.cronin@gmail.com
 -3,Orl,grin,0,47274567734612135,225959,mari.vonsteeg@gmail.com
 4,Sho,sho,0,47274567734612135,225959,mari.vonsteeg@gmail.com
 5,Liu,occ,0,22146299945498462,225876,voli.pronko@hotmail.com
 -6,Martyn,0,387637031747617,485168,brennen.ebert@hotmail.com
 7,Elias,emil,0,5422903545181815,122480,marian.lapinska@gmail.com
 8,MLP,ROM,0,221846151832406,221152,claire.strangis@gmail.com
 -9,Tolka,fhi,0,3415644112215602,928725,florian.guttmann@gmail.com
 10,Liu,qu,0,35535714211560,382216,carrie.liu.souza@gmail.com
 -11,Soriente,s,0,88662641241721,484456,karenly.hwang@gmail.com
 -12,Tu,smi,0,382506947135297,884529,luqian.we@yale.edu
 13,et,et,0,2158709848518794,445823,jedidiah.vassilatos@gmail.com
 -14,Castillo,R,0,34131029348666745,156310,gestrwy.1@me.com@gmail.com
 -15,Aliq,0,224968821885128523,122849,anis.k.mitchell@gmail.com
 -16,et,et,0,224968821885128523,122849,anis.k.mitchell@gmail.com
 # PID=1234/abcdesfgh
 # Repository_Path=/media/Data/Git-Repository
 # Execution_Time=2015-09-30:11:07:09
 # Subset_Tool=R scripting front-end version 3.2.2 (2015-08-14)
 # Subset_Tool_Path=/usr/bin/Rscript
 # Input_Script_Path=supercomputing/top5-script.r
 # Input_Script_Hash=bef5d...d7861:supercomputing/top5-script.r
 # Dataset_Path=supercomputing/supercomputer.csv
 # Dataset_Commit_Hash=aead...4cf9c:supercomputer.csv
 # Output_Path=/tmp/supercomputer-top5.csv

# Original execution:
# /usr/bin/Rscript supercomputing/top5-script.r \
# /media/Data/Git-repository/supercomputing/supercomputer.csv \
# /tmp/supercomputer-top5.csv

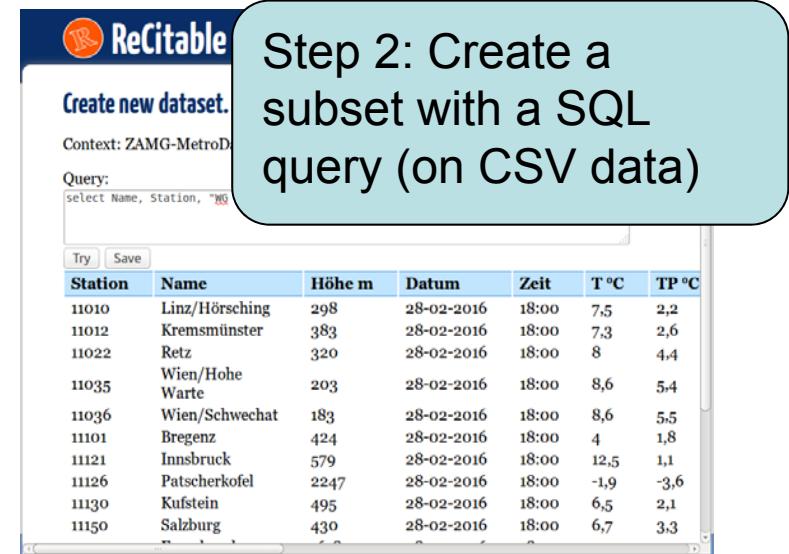
# Recommended re-execution
git --git-dir=/media/Data/Git-Repository/.git/ \
show bef5d...d7861:supercomputing/top5-script.r \
> /tmp/reproduced-datasets/top5-script.r
# Retrieve data set
git --git-dir=/media/Data/Git-Repository/.git/ \
show 47bed...b9792:supercomputing/supercomputer.csv \
> /tmp/reproduced-datasets/supercomputer.csv
# Re-execute
/usr/bin/Rscript supercomputing/top5-script.r \
/tmp/reproduced-datasets/supercomputer.csv \
/tmp/reproduced-datasets/supercomputer.csv \
/tmp/reproduced-datasets/supercomputer-top5.csv
  
```

# Git-based Prototype

- Prototype: <https://github.com/Mercynary/recitable>

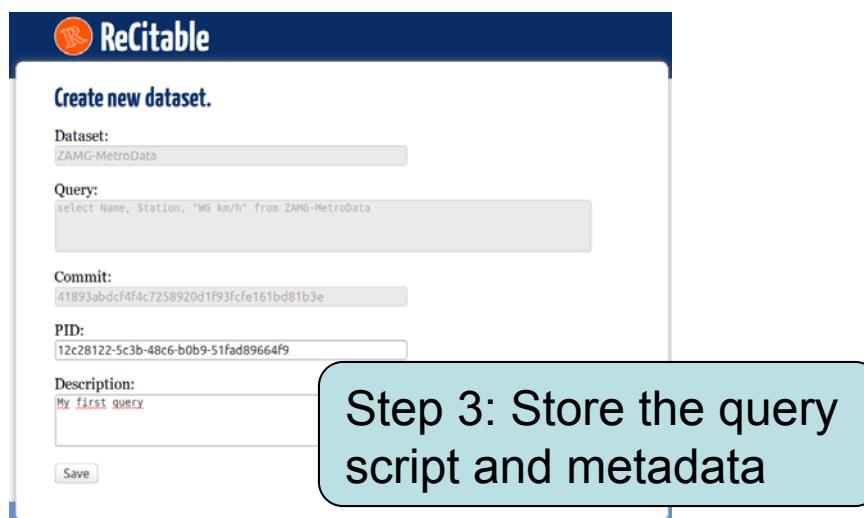


Step 1: Select a CSV file in the repository



Step 2: Create a subset with a SQL query (on CSV data)

Station	Name	Höhe m	Datum	Zeit	T °C	TP °C
11010	Linz/Hörsching	298	28-02-2016	18:00	7,5	2,2
11012	Kremsmünster	383	28-02-2016	18:00	7,3	2,6
11022	Retz	320	28-02-2016	18:00	8	4,4
11035	Wien/Hohe Warte	203	28-02-2016	18:00	8,6	5,4
11036	Wien/Schwechat	183	28-02-2016	18:00	8,6	5,5
11101	Bregenz	424	28-02-2016	18:00	4	1,8
11121	Innsbruck	579	28-02-2016	18:00	12,5	1,1
11126	Patscherkofel	2247	28-02-2016	18:00	-1,9	-3,6
11130	Kufstein	495	28-02-2016	18:00	6,5	2,1
11150	Salzburg	430	28-02-2016	18:00	6,7	3,3



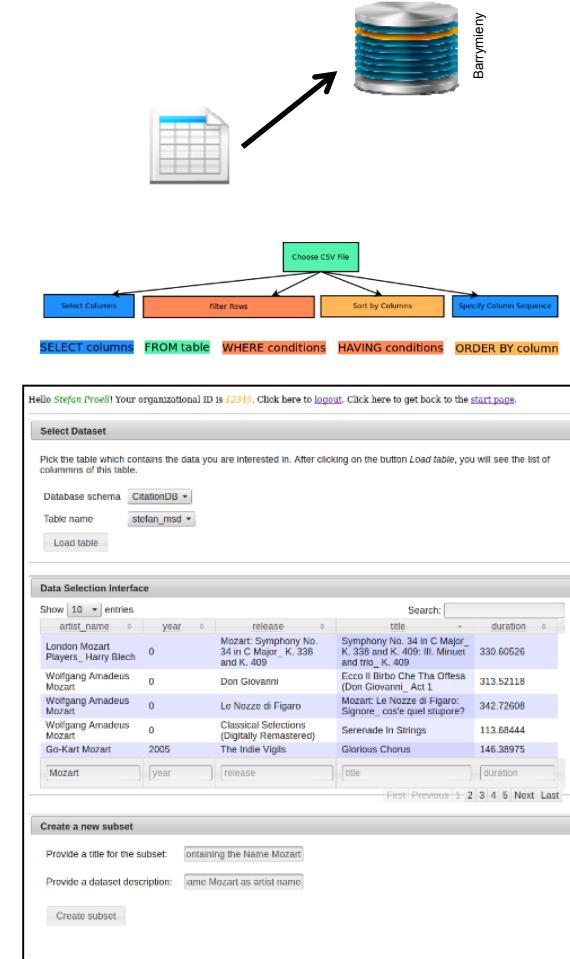
Step 3: Store the query script and metadata



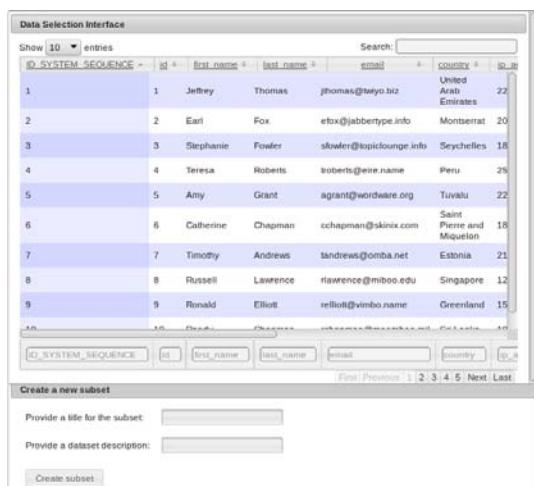
Step 4: Re-Execute!

## MySQL Prototype

- Data upload
  - User uploads a CSV file into the system
- Data migration from CSV file into RDBMS
  - Generate table structure
  - Add metadata columns (versioning)
  - Add indices (performance)
- Dynamic data
  - Insert, update and delete records
  - Events are recorded with a timestamp
- Subset creation
  - User selects columns, filters and sorts records in web interface
  - System traces the selection process
  - Exports CSV



- Source at Github:
  - <https://github.com/datascience/RDA-WGDC-CSV-Data-Citation-Prototype>
- Videos:
  - Login: <https://youtu.be/EnraIwbQfM0>
  - Upload: <https://youtu.be/xJruifX9E2U>
  - Subset: <https://www.youtube.com/watch?v=it4sC5vYiZQ>
  - Resolver: <https://youtu.be/FHsvjsUMiiY>
  - Update: <https://youtu.be/cMZ0xoZHUYl>



The screenshot shows a table with 9 rows of data:

ID	SYSTEM_SEQUENCE	first_name	last_name	email	COUNTRY
1	1	Jeffrey	Thomas	jthomas@myo.biz	United Arab Emirates 22
2	2	Earl	Fox	etbx@jabbertype.info	Montserrat 20
3	3	Stephanie	Fowler	sfowler@topiclounge.info	Seychelles 18
4	4	Teresa	Roberts	troberts@eire.name	Peru 25
5	5	Amy	Grant	agrant@wordware.org	Tuvalu 22
6	6	Catherine	Chapman	cchapman@skinix.com	Saint Pierre and Miquelon 18
7	7	Timothy	Andrews	tandrews@ombi.net	Estonia 21
8	8	Russell	Lawrence	rlawrence@nimbo.edu	Singapore 12
9	9	Ronald	Elliott	relliott@vimbo.name	Greenland 15

Below the table is a "Create a new subset" form with fields for subset title and description, and a "Create subset" button.



The screenshot shows the main menu with the following options:

- CSV Data Creator Main Menu
- Upload new data
- Update existing data
- Create a subset
- Resolve PIDs



The screenshot shows the main menu with the following options:

- CSV Data Creator Main Menu
- Upload a new CSV file and migrate the data into a database.
- Update existing data
- Create a subset from an existing dataset.
- View instances of the subsets and datasets.

## 1. Git-based

## 2. Branch Copy:

- Every Element holds two attributes ("xmlDB\_inserted" and "xmlDB\_deleted") implying their validity
- Insertion/Deletion of elements:
  - "xmlDB\_inserted" & "xmlDB\_deleted" are set accordingly ("null" = valid till present)
- Editing of attribute or text nodes:
  - Element is copied
  - Old version is disabled from now on („xmlDB\_deleted“)
  - New version is valid from now on („xmlDB\_inserted“)

## 3. Parent – child based implementation

- Database containing measurements

## Up to Date Database:

```
<measurementList>
  <measurement id="1">
    <temperature measure="kelvin">25</temperature>
  </measurement>
</measurementList>
```

## History Database:

```
<measurementList xmlIDB_inserted="01.01.2015" xmlIDB_deleted="null">
  <measurement id="1" xmlIDB_inserted="01.01.2015"
  xmlIDB_deleted="null">
    <temperature measure="kelvin" xmlIDB_inserted="01.01.2015"
    xmlIDB_deleted="null">25</temperature>
  </measurement>
</measurementList>
```

# Versioning XML Data

```
<measurementList xmlDB_inserted="01.01.2015" xmlDB_deleted="null">
  <measurement id="1" xmlDB_inserted="01.01.2015" xmlDB_deleted="null">
    <temperature measure="kelvin" xmlDB_inserted="01.01.2015"
      xmlDB_deleted="null">25</temperature>
  </measurement>
</measurementList>

<measurementList xmlDB_inserted="01.01.2015" xmlDB_deleted="null">
  <measurement id="1" xmlDB_inserted="01.01.2015" xmlDB_deleted="null">
    <temperature measure="kelvin" xmlDB_inserted="01.01.2015"
      xmlDB_deleted="null">25</temperature>
    <airPressure measure="bar" xmlDB_inserted="11.11.2015"
      xmlDB_deleted="null">1</airPressure>
  </measurement>
</measurementList>
```

- **Parent-Child based solution**
  - Elements inherit both timestamps from parents (in case they would be the same)
  - “xmlDB\_deleted” only inserted when element is actually deleted.
  - Versioning of text/attribute nodes with “versioning blocks” (elements holding different versions of values)

# Versioning XML Data

```
<measurementList xmlDB_inserted="01.01.2015">
  <measurement id="1">
    <temperature measure="kelvin">25</temperature>
  </measurement>
</measurementList>

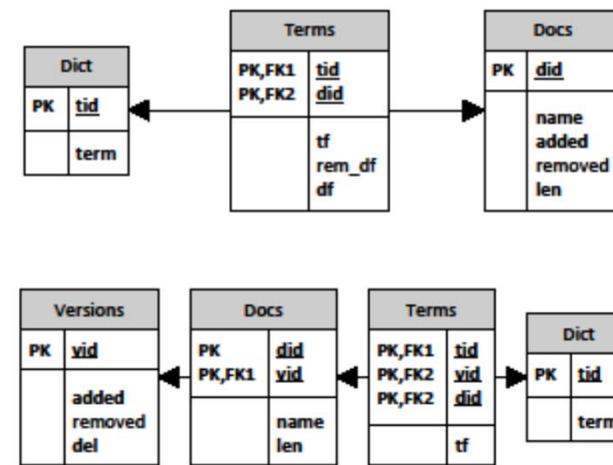
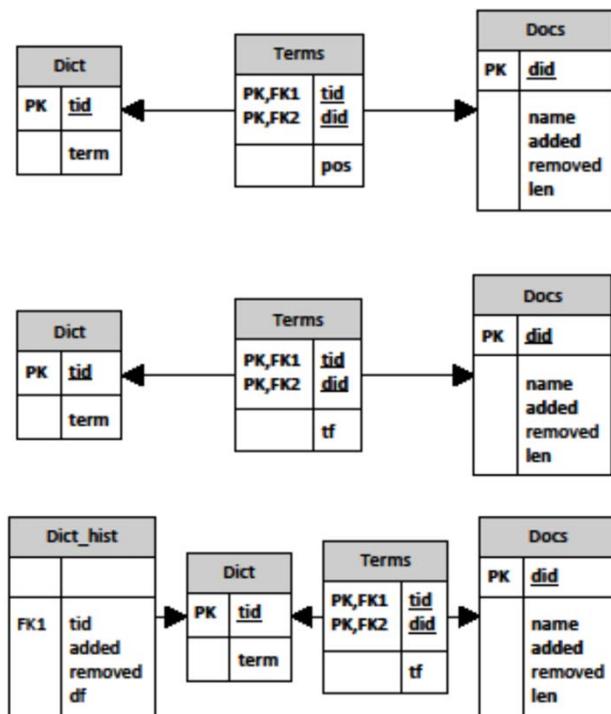
<measurementList xmlDB_inserted="01.01.2015">
  <measurement id="1">
    <temperature measure="kelvin">25</temperature>
    <airPressure measure="bar"
      xmlDB_inserted="11.11.2015">1</airPressure>
  </measurement>
</measurementList>
```

# Versioning XML Data

```
<measurementList xmlDB_inserted="01.01.2015">
  <measurement id="1">
    <temperature measure="kelvin">25</temperature>
  </measurement>
</measurementList>

<measurementList xmlDB_inserted="01.01.2015">
  <measurement id="1">
    <temperature measure="celsius">25
      <xmlDB_version type="attribute" name="measure"
        lastEdited="11.11.2015" nrPositions="0">
        <xmlDB_v xmlDB_inserted="01.01.2015"
          xmlDB_deleted="11.11.2015">kelvin</xmlDB_v>
      </xmlDB_version>
    </temperature>
  </measurement>
</measurementList>
```

- Column store solution supporting IR models
- Range of different ways evaluated to implement versioning of raw and derived data
- Differ in storage requirements, query/update performance



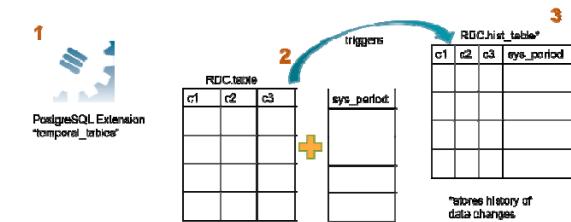
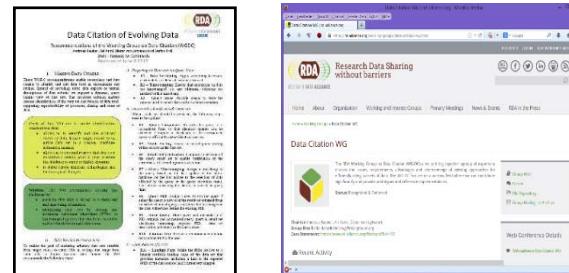
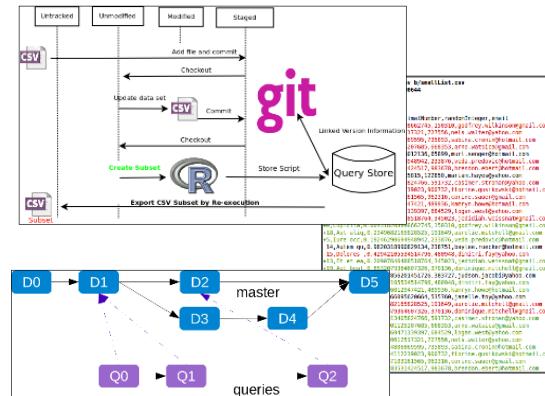
- Linked data: no implementation yet, concepts:
  - File-based: Git, versioned file systems
  - RDBMS: expand triple store to add timestamp columns
  - Within LOD-mindset: add time validity triples
- Generic principle, holds for all kinds of data

- Concrete implementation will differ, depending on
  - Size
  - Dynamics
  - Impact of changes / APIs / ...
- Timestamp as universal concept, granularity defineable
- Optimizations, e.g.  
no versioning when there has been no read-access  
(unless provenance requirement...)
- Now collecting practical evidence / deployment stories /  
good (or best) practices  
→ Webinars from adopters

# Adoption

- **Series of Webinars: Adoption reports**  
<https://www.rd-alliance.org/group/data-citation-wg/webconference/webconference-data-citation-wg.html>
- **Adoption of the RDA Data Citation of Evolving Data Recommendation to Electronic Health Records**  
Leslie McIntosh, PhD, MPH, Director Center for Biomedical Informatics, Washington University in St.Luis  
Tue, Jan 17 2017
- **Implementation of Dynamic Data Citation at the Vermont Monitoring Cooperative** Presenter: James Duncan, VMC, University of Vermont, Burlington, VT  
Mon, Feb 13 2017
- **Implementing the RDA Data Citation Recommendations in the Distributed Infrastructure of the Virtual and Atomic Molecular Data Center (VAMDC)**  
Presenter: Carlo Maria Zwölf, VAMDC, Observatoire de Paris, FR  
Fri, Mar 31 2017

# Thank you!



Thanks!

<https://rd-alliance.org/working-groups/data-citation-wg.html>



Editing data for Data 3

Show changes Save to a new version

```

1 UPDATE Z0001_test SET SiteID = 'Stevensville Brook' WHERE db_table_pk=30
2 DELETE FROM Z0001_test where db_table_pk=30
3 DELETE FROM Z0001_test where db_table_pk>35

```

Actions	SiteID	LabID	Date	MeanDensity	Mear
edit	Stevensville Brook	2000.187	0000-00-00	4644322354	39.0
edit	Winhall River	2011.081	2011-10-07	201	47.5
edit	Winhall River	2012.089	2012-09-27	1981	52.0
edit	Winhall	2013.150	2013-10-15	1002	50.0

