The Citation of Dynamic data group deals with the problem of how to give reliable citations to dynamically changing data. Giving accurate citations is essential for reproducing results.

### Goals

- Reliably cite dynamically changing data

The group defined a set of 24 principles that, when followed, ensure citable dynamic data. The solution is not usable for distributed data.

### Motivation

Digitally driven research is a rather young discipline that evolves fast. As a result the tools and the data are rarely developed with a focus of long term awareness. What matters most to researchers are fast results and prompt publications. Yet, only if results can be reproduced precisely, the validity of research experiments and business processes can be judged, evaluated and verified. This requires precise identification of the data used in any such process. However, researchers rarely use an entire dataset as provided, but select subsets, be it a specific time-range, or a set of measurements. Hence there is a strong need for data citation mechanisms that allow identifying arbitrary subsets of large data set with precision in a machine-actionable way.

An additional challenge within the area of research data is the requirement to cite evolving data reliably. Researchers need the possibility to reference data material that is subject to change. Hence, mechanisms are required that allow to cite data as they used it during a particular experiment. When the data gets updated, modified or deleted, these changes must be reflected and should be recoverable by the citation system as well. Therefore time-stamped/versioned data is an important factor. The easier and more transparently this citation process can be implemented, the higher the acceptance. The solution needs to be machine-actionable, and needs to scale from small to very large datasets, from static data to highly dynamic data, across changes in the data representation. We will provide proofs of concept, mockups and prototype implementations that can be tested and used by the community. We want to go beyond theoretical work and deliver real world prototypes and case studies for our models. In an optimal setting, a researcher, when selecting a subset of data for an experiment, will be issued with a PID that allows others to retrieve precisely the same data set again.

### Approach

There are several possible solutions that fit into the framework that the group is proposing. The principle currently proposed includes the following aspects:

- Ensuring that the data collection is versioned, i.e. changes/deletions to the data are marked as deleted and (re-) inserted with according timestamps.

- PIDs are assigned to the query/expression identifying a certain subset of the data that one wishes to cite, with the query being time-stamped as well.

- Hash keys are computed for the selection result to allow subsequent verification of identity.

- Issues such as unique sorting of results need to be considered when the operation returns data as sets and subsequent process work on the sequence the data is provided in.
These should be working across all settings where we have a combination of data sources and operations identifying subsets at specific points in time.

The exact technical implementation depends a lot on the already present local data structures and procedures. However, evaluations on a number of pilot projects across different data types (SQL, CSV, XML) so far indicate that it does work.

**Potential Impact**

The main impact of this kind of work is on the reproducibility of work. It allows for a database to be dynamically updated as needed, by adding new information, or by updating old information, while still allowing for data conditions as they were at a specific point in time to be reproduced.

The proposed approach has several advantages over current practices such as storing the actual subsets as redundant data deposits with a study that go beyond mere scalability issues. Having the query/expression as a basis for identifying the dataset provides valuable semantic information on the way the specific dataset was constructed, as opposed to merely having a dump of the data.

Furthermore, it allows to re-execute the query with the original timestamp, retrieving the original data as used in a study, as well as re-executing it with the current time-stamp, obtaining the data with all additions/corrections applied in the meantime, and analyzing the resulting differences.

As data gets migrated to new representations, the queries can be migrated accordingly, ensuring stability across technological changes. The approach works consistently for both small and large datasets, as well as for static and highly dynamic data. By promoting a consistent approach, decisions and science done on the basis of such measurements will become more transparent and reproducible.

**Applying the outputs**

The adoption of the Citation of Dynamic Data output is heavily dependent on the underlying architecture. Luckily it can be captured in a number of principles.

For more details on the theoretical background we refer to Pröll and Rauber 2013.

The principles are as follows:

R1 – Data Versioning: Apply versioning to ensure earlier states of data sets can be retrieved.

R2 – Timestamping: Ensure that operations on data are timestamped, i.e. any additions, deletions are marked with a timestamp.

R3 – Query Store Facilities: Provide means for storing queries and the associated metadata in order to re-execute them in the future.

R4 – Query Uniqueness: Re-write the query to a normalised form so that identical queries can be detected. Compute a checksum of the normalized query to efficiently detect identical queries.

R5 – Stable Sorting: Ensure that the sorting of the records in the data set is unambiguous and reproducible.

R6 – Result Set Verification: Compute fixity information (checksum) of the query result set to enable verification of the correctness of a result upon re-execution.

R7 – Query Timestamping: Assign a timestamp to the query based on the last update to the entire database (or the last update to the selection of data affected by the query or the query execution time). This allows retrieving the data as it existed at the time a user issued a query.

R8 – Query PID: Assign a new PID to the query if either the query is new or if the result set returned from an earlier identical query is different due to changes in the data. Otherwise, return the existing PID.

R9 – Store Query: Store query and metadata (e.g. PID, original and normalized query, query & result set checksum, timestamp, superset PID, data set description, and other) in the query store.

R10 – Automated Citation Texts: Generate citation texts in the format prevalent in the designated community for lowering the barrier for citing the data. Include the PID into the citation text snippet.

R11 – Landing Page: Make the PIDs resolve to a human readable landing page that provides the data (via query re-execution) and metadata, including a link to the superset (PID of the data source) and citation text snippet.

R12 – Machine Actionability: Provide an API / machine actionable landing page to access metadata and data via query re-execution.

R13 – Technology Migration: When data is migrated to a new representation (e.g. new database system, a new schema or a completely different technology), migrate also the queries and associated fixity information.

R14 – Migration Verification: Verify successful data and query migration, ensuring that queries can be re-executed correctly.

For more information and help on how to adopt this for your infrastructure please contact our team.
Implementation Examples

<Examples of imlementation that we know off>

Related Q&A

- I want to cite my data, but I have constantly changing versions. How do I do this?