DESIGN AND IMPLEMENTATION OF A WORKFLOW FOR QUALITY IMPROVEMENT OF THE METADATA OF SCIENTIFIC PUBLICATIONS

Stefan Wolff
0000-0003-0015-9671

November 2, 2023

ABSTRACT

In this paper, a detailed workflow for analyzing and improving the quality of metadata of scientific publications is presented and tested.

The workflow was developed based on approaches from the literature. Frequently occurring types of errors from the literature were compiled and mapped to the data-quality dimensions most relevant for publication data—completeness, correctness, and consistency—and made measurable. Based on the identified data errors, a process for improving data quality was developed. This process includes parsing hidden data, correcting incorrectly formatted attribute values, enriching with external data, carrying out deduplication, and filtering erroneous records.

The effectiveness of the workflow was confirmed in an exemplary application to publication data from Open Researcher and Contributor ID (ORCID), with 56% of the identified data errors corrected. The workflow will be applied to publication data from other source systems in the future to further increase its performance.

Keywords meta data · scientific publications · data quality managment · data quality workflow

1 Introduction

With the increasing number of IT systems in the era of Web 2.0 and the amount of data they collect, there is a growing demand for the beneficial use of the resulting data. The effort required to collect data in other systems should be reduced by integrating available data in the most value-creating way possible. Through such integration, a system's own data stock can be completed, extended, or verified, thus contributing to increased system performance. In [1, 2, 3] emphasize that the integration of external data is a central component of the construction of information systems.

To integrate external data, it is not only necessary to cope with data protection and licensing requirements [4] but also to transfer the data into the respective use context. This process includes obtaining the data, transferring it to the respective data schema, and ensuring its quality. An example of a data integration task with high data-quality demands is the implementation of a research system for scientific literature. High-quality publication data enables precise and uniform indexing and thus facilitates data retrieval. In addition to enhancing user satisfaction, this promotes the visibility of the research results contained in the literature. In addition, the completeness and accuracy of the metadata contribute to increased credibility [5]. Moreover, [6] states that Current Research Information Systems (CRIS) operators rate data quality as a critical success factor for user acceptance. Ensuring data quality is also a prerequisite for averting damage caused by poor decisions (e.g., in strategic business decisions) resulting from inadequate information [7, 8, 9].

Data Quality Management (DQM) is therefore an essential part of data management [10]. It can be realized through existing software products, e. g. from Sakdas [11], Katara [12] or DataCleaner [10], and through a system's own implementations. The use of existing software is advantageous because relevant components of the quality management process are already prepared, such as the procedure for analyzing data quality. While the use of commercial software involves a risk of high dependence on the service provider [13], open-source software permits free use and adjustment. However, both options make it difficult to align the software with the institution's technology stack because of technology decision has been preempted. Such tuning is frequently required to minimize the costs of the authority structure, both for the initial implementation of the system and for its maintenance and advancement, and to increase the potential synergies between different implementation projects, e. g. by utilizing existing architectures and components [14]. By implementing of an own solution, these requirements can be met. Such implementation requires, however, the complete conception of the DQM.

In the literature, the DQM procedure is structured around several conceptual approaches, e. g. in [15, 16, 17]. They can be divided into generic and domain-specific approaches. The former have the advantage of covering a greater number of application scenarios. However, they are too generic to guide the implementation of specific processes. In contrast, superior results can be achieved with the latter approaches due to their specific orientations. However, domain-specific approaches are often specified exclusively in terms of software. As a result, the technology and architecture decisions are largely predetermined. In addition, the workflows that led to the development of the specific software are rarely described in detail, rendering their transfer to other technologies and their conceptual evolution difficult.

1.1 Objective and procedure

In this work, the goal is to support future developments in data quality processes through the creation of a workflow. In order to achieve the highest possible level of detail, the workflow is focused on a specific domain—publication data quality enhancement. This domain was chosen because it is linked to relevant use cases, in particular literature search systems as well as CRIS, and because no workflows could be found to guide the development of corresponding data quality processes in detail. The research question of this work is therefore:

Which workflow can be used to implement a publication data quality process?

The following procedure was followed to develop the workflow:

(i) Based on a comprehensive literature review, an overview of current solutions in the research for the different tasks of DQM was provided.
(ii) Based on the literature, a workflow for improving the quality of publication data was derived and concretized.
(iii) Through practical implementation, the performance of the workflow was verified, and an example of its implementation was given.

1.2 Structure of the work

Section 2 presents thematically related work. In Section 3, the design of the workflow is presented and its derivation is explained. The practical implementation of the workflow is described in Section 4, and the results are discussed. This is followed in Section 5 by the conclusion of this work and an outlook on potentially related research.

2 Related work

The literature on data quality improvement falls into two categories. The works in the first category are focused on the individual steps of the process, specifically *analysis of data quality*, *correction of data errors*, and *deduplication*. The second category contains work that encompasses the entire *quality workflow*.

2.1 Analysis of data quality

[18] is a frequently cited publication on “data quality” that identifies the importance of data quality to consumers. To this end, an empirical study was conducted in the form of a two-phase survey. The first phase investigated the characteristics that distinguish data quality; the second phase explored the importance of these characteristics. As a result, 118 quality characteristics were identified and arranged hierarchically. The characteristics were first divided into the four main categories: “intrinsic data quality”, “contextual data quality”, “representative data quality”, and “accessibility”. While the “contextual data quality” category combines properties that arise from the specific contexts in which the data are used, the “intrinsic data quality” category groups properties that the data exhibit on their own. In contrast, “representative data quality” refers to the properties of the data presentation, and “accessibility” refers to the way the data are made available to others. These categories were further subdivided into 20 quality dimensions, e. g. “correctness” (intrinsic), “completeness” (contextual), “understandability” (representative), and “protection from unwanted modification” (accessibility). The individual quality attributes were then assigned to the quality dimensions.

In [19], a literature study examined and compared various publications on the topic of “quality of Big Data”. The frequency of use of the different quality dimensions was determined. As a result, the study presents the percentages of the 14 most frequent quality dimensions.

[20] presents an extensive literature review of quality metrics for Big Data. Among other things, it found that metrics for measuring quality currently exist for only 11 of the over 50 quality dimensions found in the literature. As part of the literature review, specific metrics for assessing various data quality dimensions were proposed. These supplement the metrics found in the literature with the use of weights. The performance of the metrics was verified by implementing them based on Apache Spark and Jupyter notebooks and comparing the results with those of other metrics. The proposed metrics were found to have increased accuracy.

2.2 Correction of data errors

[21] presents a model for generating data quality rules (processing steps for quality improvement). Based on a user-created evaluation procedure, attributes of erroneous records are deleted step by step, and the resulting data quality is calculated. The erroneous attribute values identified in this way are either replaced or the corresponding data record is deleted. Default values or calculated values, e. g. average values of the corresponding attribute, are used for replacement.

[22] describes the implementation of a data quality service based on Apache Spark and Hadoop. This process enables the assessment of data quality by first creating a data quality profile that contains information about the data structure and data types, as well as field occupancy statistics. The statistics represents the following attributes: the number of values, null values, and unique values, and for numeric values, the minimum, maximum, and standard deviation. Metrics for assessing data quality are automatically selected based on data types. To perform a contextual data quality assessment, a specification of the requirements for the data is requested from the user.

[23] outlines a process for data quality analysis and improvement. In addition, it presents metrics for the four data quality dimensions “currentness”, “accuracy”, “completeness”, and “consistency”.

A special feature of this work is that the quality of the data quality process itself, e. g. data analysis, is also examined.

[24] presents an approach for identifying data errors is presented that is based on the definition of error patterns (anti-patterns). A special XML-based syntax has been defined that is used to formulate error patterns. This approach differs from most other data quality analyses in that it focuses on potential errors rather than the target requirements for the data. Apart from the hit counts of the defined anti-patterns, there is no use of metrics for data quality assessment.

[25] presents a framework for Big Data quality assessment. A unique feature of this approach is that the timing of the assessment is taken into account. It was found that some aspects of data quality can only be meaningfully addressed at the beginning of the preparation process. This finding particularly affects the level of expertise required in data loading and initial analysis, while more automation is possible as the process progresses.

2.3 Deduplication

[26] explains the Sorted-Neighborhood approach, which aims to reduce the complexity of the deduplication process. In addition, it presents the multipass approach, which increases the efficiency of deduplication by making multiple passes of the Sorted-Neighborhood procedure, each with a different sort key. A rule system is proposed for identifying duplicates, and an edit distance, the Levenshtein distance, is used to evaluate the similarities of strings. Moreover, transitive closure of duplicates is recommended for additional efficiency enhancement. The “transitive closure of duplicates” refers to the consideration of concatenated duplicates of the following form. If it is known that Object A is a duplicate of Object B and that Object B is a duplicate of Object C, then *transitive* can be used to conclude that Object A is a duplicate of Object C.

[27] also presents an optimization of the Sorted-Neighborhood procedure. This method aims to address the weakness of the fixed window size in the Sorted-Neighborhood procedure by using priority queues. The “priority queues” represent a concept where potential duplicates are arranged in groups of variable lengths. Within the groups, objects are sorted in descending order of similarity to the first representative of each group, reducing the number of similarity calculations when inserting new objects.

2.4 Quality workflow

While the publications presented so far refer to individual phases of quality workflows, the literature cited in this chapter is focused on complete quality workflows.

[16] proposes a framework for DQM. Starting from the data quality dimensions, the requirements for the data are captured. Metrics for determining data quality are then created and validated. In formulating quality rules to improve data quality, the framework supports the user through automatic approaches. In these approaches, a set of potential improvements is determined based on the iterative deletion of attributes and subsequent quality assessment. Next,

transformation rules are formulated, which are checked using random samples. In addition, monitoring of DQM and approaches for visualizing data quality are included.

[28] also proposes a procedure for analyzing and improving data quality. In this procedure, exploratory quality profiling is performed. During this process, various transformation rules are tested to identify potentially suitable transformation steps.

[17] outlines a data cleansing process for CRIS. Examples of data quality problems are identified, e. g. inconsistencies in personal information. Response actions for poor data quality are categorized into the laissez-faire approach (do nothing), the reactive approach (fix cause of error after it occurs), and the proactive approach (prevent errors from occurring). It was found that the more frequently the data is changed, the more profitable the proactive approach becomes. The work also proposes the corrective actions “parse”, “standardize”, “enrich”, and “eliminate duplicates”, which provide data quality optimization tailored especially to publication data.

The present work builds on these approaches by deriving a workflow for quality enhancement from the current state of research. This workflow is tailored to publication data and goes beyond the approaches found in the literature due to its level of detail.

3 Data quality workflow

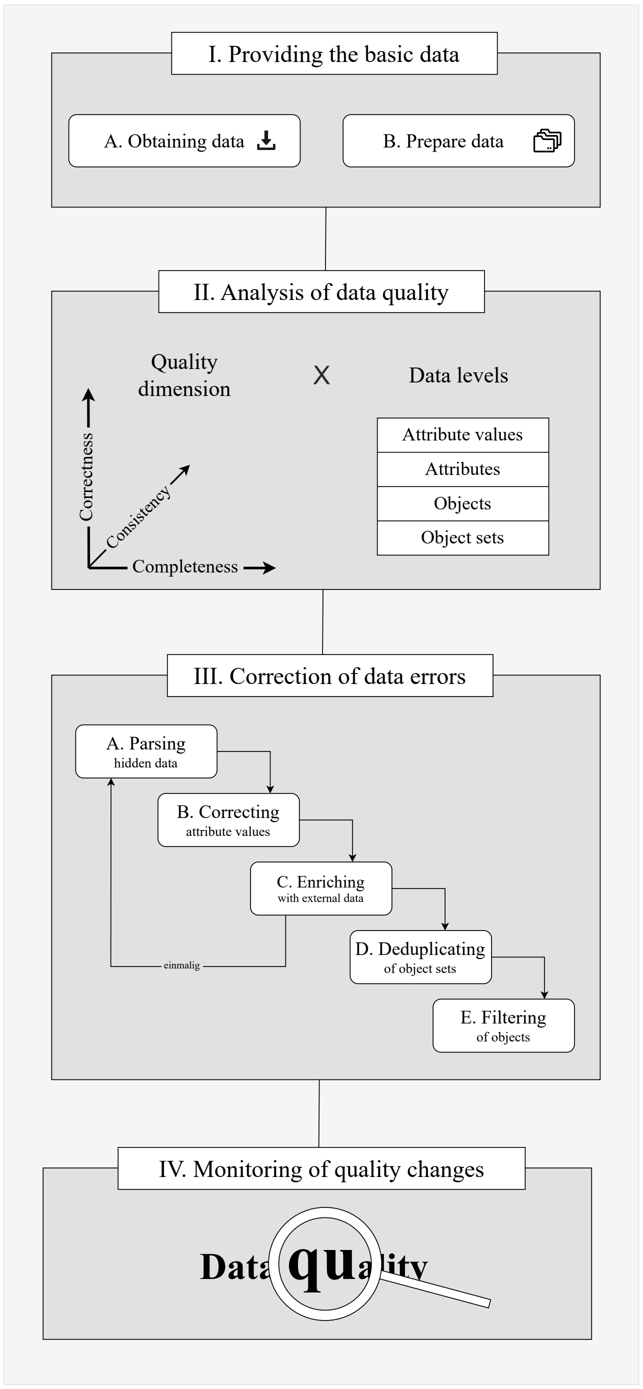
I. Providing the basic data
A. Obtaining data
B. Prepare data
II. Analysis of data quality
Quality dimension
X
Data levels
Correctness
Consistency
Completeness
Attribute values
Attributes
Objects
Object sets
III. Correction of data errors
A. Parsing hidden data
B. Correcting attribute values
C. Enriching with external data
D. Deduplicating of object sets
E. Filtering of objects
einmalig
IV. Monitoring of quality changes
Data quality

Figure 1: WQP

This chapter presents the workflow design. Due to its specific focus, it is called the workflow for improving the quality of publication data (WQP). It differs from other workflows, especially in its high level of detail. While many data quality workflows are general in nature, as noted in [29], the WQP includes detailed steps for successively implementing specific data quality processes. These steps were first derived theoretically from workflow objectives and methods in the literature. They were then tested and improved through practical implementation.

The design of the WQP is based on a proactive approach. This means that the quality of the data is comprehensively analyzed before it is used, and errors are corrected based on this analysis. In the WQP, this approach is reinforced by the “analysis of data quality” phase and the subsequent “correction of data errors” phase. This approach differs from the reactive approach, which is described in [29] as a weakness of numerous data quality methods. In this method, quality analysis during the data management process is omitted. Errors are corrected only when they become apparent during the use of the data.

Since it cannot be expected that all errors will be identified during productive use, and since the occurrence of errors may lead to a loss of user confidence, the design of WQP was based on a proactive DQM approach. To promote reproducibility [30] of the process, the phase “provide baseline data” was added to the beginning of the workflow. In addition, the “monitoring quality development” phase was connected to the data correction processes in order to detect and correct emerging errors in a timely manner through continuous quality monitoring [31]. The WQP is outlined in Figure 1 and is explained below.

3.1 Providing the basic data

In the first phase, the provision of data is automated. This data is stored on the target system and converted into a form that can be used for subsequent processing. Only the data that is required for the planned application should be transferred in this step, as a smaller volume of data can simplify processing.

3.1.1 Obtaining data

While source code is usually stored for the long term, e. g. in institutional source code repositories, the data to be processed is often stored only temporarily due to its size. Automating data retrieval can promote the reproducibility of the data processing process. In addition, it simplifies the updating of the basic data. A procedure for automating data retrieval is described in Table 1.

Table 1: Obtaining the data

I. Providing the basic data
A. Obtain data
1. Get access form: In order to be able to load the basic data, it must first be determined in which form they can be retrieved. In particular, the access target, e. g. an API endpoint, and the access protocol must be identified.
2. Ensure accessibility: The second step is to ensure that the necessary permissions or authentication credentials are in place to access the access targets identified in step 1.
3. Check authorization for data processing: Since the availability of data does not imply authorization for specific processing, authorization must be ensured relative to the intended use. The license terms of the data can often be found in the general terms and conditions.
4. Implementing the data storage in the target system: After formal requirements are met, data retrieval is automated by implementing download and storage. Except for authentication data, all configuration parameters, e. g. the address of the access target, should be included so that data retrieval can be repeated as straightforwardly as possible.

3.1.2 Preparing data

The second task of this phase is to prepare the data in terms of its scope, structure, and format for processing. However, error corrections should be avoided in this step so they can be performed exclusively at the designated points of the preparation process. It is therefore necessary to avoid making the data schema sharper than the source data allows. For example, years should not be modeled as numbers in the data schema if they are contained as strings in the base data. In this case, it is possible that values are included that cannot be converted to a correct year without corrective action, e. g. “’23” instead of “2023”. The conversion is outlined in Table 2.

Table 2: Preparing the data

I. Providing the basic data

B. Prepare data

1. Describe the required data:
   To gather the requirements for the data to be provided, the entities and the attributes are gathered and described.

2. To define the data schema and format:
   Based on the description of the data, its schema and format are chosen to support further processing to the extent possible. For this purpose, the data is reduced to the scope described in step 1. In addition, the data schema is chosen in such a way that the process of data correction is simplified. For example, a primary key must exist for each entity. If this consists of the combination of multiple attributes, a unique key is derived and stored in an additional attribute. Embedded entities that have a cardinality not equal to one to *n* are extracted, e. g. the authors of the publications.

3. Mapping of the source data:
   In this step, the attributes of the source data are mapped to the attributes in the data schema from step 1. This mapping should be documented in a suitable form, e. g. tabular, so that it can be effectively traced later.

4. Implement the transformation of the source data:
   In the final step of this phase, the transformation of the source data into the defined data schema and format is implemented using the elaborated mapping. As a result, the data are ready on the target system in a form that supports post-processing. They are referred to as "**base data**" going forward, as they form the basis of the ensuing process.

3.2 Analysis of data quality

In data quality analysis, the properties of the data are structured based on the different data quality dimensions in order to group and examine similar quality aspects in each case. To meet the requirement of economic efficiency of metrics, as proposed in [32], the set of data quality dimensions that can be found in the literature was limited to those most relevant for the purpose.

In [31] and [33], empirical studies were conducted regarding the relevance of data quality dimensions in CRIS. In addition, a study targeting Big Data quality in general was conducted in [19]. All three studies found that the data quality dimensions of "correctness", "completeness", and "consistency" were the most relevant. Therefore, these dimensions were used as the basis for the workflow data analysis.

3.2.1 Structure of the data analysis

To measure specific quality deficiencies, metrics are defined that express the fulfillment of quality aspects in terms of ratios. To identify the quality aspects to be measured, typical error types were first extracted from the literature [34, 35, 36]. While [34, 36] identifies generally applicable error types, [35] categorizes specific data quality issues identified in publication data from Web of Science[1].

Table 3: Error types

DQ dimension	Error type
Completeness	unmaintained attributes [34], missing data [34], missing value [36], missing attributes [36], incomplete tuples [36]
Correctness	erroneous data due to wrong input, e. g. incorrect reading [34]; typos [34], inaccurate data [34], incorrect separation of multiple values [35], misspelling [36], incorrect entry [36], interval violation [36], sentence violation (set-violation) [36], various incorrect formats [34], ambiguous formatting when combining first and last names [35], incorrect or inconsistent order in summary name of institution and organizational unit [35], syntax violation [36], incorrect recording of umlauts and special characters in proper names [35], use of different alphabets for author names [35], outliers [36], distortion [36], noise [36]
Consistency	redundant and inconsistent data [34], use of different name variants for the same person [35], multiple entries of entities each under different spellings [35], violated uniqueness of DOIs [35], violation of uniqueness of an attribute [36], inconsistency between attribute values [36], existence of synonyms [36], inconsistency within an entity [36]
Timeliness	outdated information [34], name change after marriage [35]
Uniqueness	duplicate records [34], redundancy within an entity [36]
Relevance	irrelevant record [36], misuse of attributes for additional information [34], inappropriate value with respect to attribute context [36], value outside attribute context [36]

To structure the data analysis, the error types in Table 3 are first grouped by quality dimensions. This contains, for example, the "set violation". This denotes the occurrence of values that are not from the expected set of attribute values for the attribute. For example, such an error occurs when an attribute "day of the week" contains the value "midweek" instead of "Wednesday".

[1]Web of Science is a commercial platform for publication and citation data operated by the company Clarivate

In contrast, the error “syntax violation” describes the formatting of a value that deviates from a formatting rule. For example, this error is present when an ISSN is expected in the format NNNN-NNN, e. g. “2579-0048”, but is contained in the form “2579 0048” or “25790048”.

Another error typical of publication data is “the incorrect separation of multiple values”. This occurs when multiple entries are combined into one attribute value and are separated during data processing. For example, multiple author names are often combined into one attribute value, e. g. “John Doe and Max Mustermann”. The individual names can be separated by splitting the string at each occurrence of “ and ” at that point. However, if facility names are also included, there may be an erroneous separation, for e. g. “Chair of Databases and Information Systems”.

The error “noise” can be interpreted with respect to publication data as a minimal deviation from the error-free value, where the meaning of the specification is not changed in terms of content, e. g. an additional space at the end of the attribute value.

The next step was to refine the structure by limiting the error types to those that are sufficiently concrete to be measured using a metric. For example, the error type “inaccurate data”, while typical, is too general for specific measurement. In addition, the data quality dimensions were narrowed to the prioritized dimensions of “completeness”, “correctness”, and “consistency”. This reduced set of error types was additionally grouped according to the associated data levels. The resulting structure of data quality dimensions, data levels, and error types is shown in Table 4. It is used as the basis for structuring the data analysis. The resulting analysis steps are in turn each divided into the identification of specific analysis criteria, their implementation, and the validation of the assessment.

Table 4: Error types filtered and grouped

DQ dimension	Data levels	Error type from literature
Completeness	attribute value	missing value [36]
	attribute	unmaintained attributes [34], missing attributes [36]
	object	incomplete tuples [36]
Correctness	attribute value	interval violation [36], various incorrect formats [34], ambiguous formatting when combining first and last names [35], incorrect or inconsistent order when combining facility and organizational unit names [35], syntax violation [36], incorrect recording of umlauts and special characters in proper names [35], use of different alphabets for author names [35]
Consistency	attribute	violated uniqueness of DOIs [35], violate uniqueness of an attribute [36], existence of synonyms [36]
	object	inconsistency between attribute values [36]
	object set	use of different name variants for the same person [35], multiple entry of entities under different spellings each [35], inconsistency within an entity [36]

3.2.2 Completeness

Completeness is measured according to the groupings in Table 4 for the attribute value level, attribute level, and object level, respectively. At the attribute value level, the completeness of formatted data can be measured by checking for the presence of all expected partial information based on the format definition. For example, the publication date of a source must contain at least the year, month, and day to be complete. Measuring the completeness of unstructured text attributes is more difficult due to the lack of verification criteria. However, falling below a specific minimum length can be used as an indication of incompleteness. For example, the publication title may be judged incomplete if it consists of only one character. The specific thresholds are estimated by subject matter experts. To develop the measurements, the sequence of steps listed in Table 5 is suggested.

Table 5: Measurement of the completeness of attribute values

II. Data quality analysis

A. Data quality dimension: completeness

a) Data level: attribute value

1. Specify minimum components for formatted attributes:
   Criteria based on the format definitions are formulated for formatted attributes to check for the presence of minimum sets of partial information.
2. implementation of completeness measurements:
   Instances of the metric MinValue are implemented for the criteria defined in step 1.

$$\frac{\textit{number of complete attribute values}}{\textit{number of attribute values}} \qquad \text{(MinValue)}$$

3. Setting minimum lengths for text attributes:
   Minimum lengths are set for all text attributes where falling below a certain number of characters is an indicator of incompleteness.
4. Implementation of minimum length measurements:
   Instances of the metric MinLength are implemented for each of the minimum lengths specified in step 3. The check should go beyond evaluating for null or empty strings to avoid redundancy with the attribute-level completeness check.

$$\frac{\textit{number of sufficiently long attribute values}}{\textit{number of attribute values}} \qquad \text{(MinLength)}$$

---

5. validation of the evaluation:
   To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

In Table 6 a procedure for validating the specific expressions of the metrics is proposed. This involves checking that only actual errors are included in the calculation of the metric.

Table 6: Validation of the data analysis

II. Data quality analysis

Validation of data analysis

1. Creating samples:
   Based on the implemented validation criteria, samples of each *k* tuples, attributes, or attribute values that do not meet the check criteria are created. It is possible to specify *k* relative to the number of invalid elements or absolute. At the beginning of the implementation, a potentially optimal *k* with respect to the efficiency of the implementation and the target accuracy of the evaluation is estimated, e. g. $k = 100$. To ensure that the different data quality aspects are evaluated in comparable depth, the same *k* is used for all validations.

---

2. Validate data analysis:
   For validation, all elements of the samples are checked for conformity to expectations. Thus, it is necessary to examine whether only data that contradict the test criteria are included.
3. Optimize data analysis:
   The implementation is adjusted until all sample entries have the expected errors. Accordingly, the samples should not contain false-positive error findings. Complete avoidance of false-negative error findings (i.e., uncovered errors) cannot be achieved with confidence using this procedure. However, accuracy can be increased by increasing *k*.

At the attribute level, completeness can be measured in terms of both form and content. To determine formal completeness, the presence of attributes is checked. [37] refers to this quality property as column completeness. It expresses how often the attribute is non-null relative to the number of objects checked. Therefore, to measure it, a metric in the form of a *simple ratio* is specified. To determine content completeness, it is necessary to examine whether each value of a reference set occurs at least once in the attribute being tested. [22] specifies a fourth metric form for this purpose: population completeness.

In the context of publication data, for example, a list of journals and associated ISSNs can be generated (reference set) for which publications are expected in the base data. Population completeness can be measured with respect to the journal list by determining the number of ISSNs that are also included in the reference set and relating this to the cardinality of the journal list. The proposed procedure for measuring attribute completeness is described in Table 7.

Table 7: Measurement of the completeness of attributes

II. Data quality analysis
A. Data quality dimension: completeness
b. Data level: attribute
1. Implementation of formal completeness measurement: For each attribute, a measurement is implemented to check the presence of the corresponding attribute by counting how many times the attribute value is non-zero. $\frac{\textit{number of attributes present}}{\textit{number of objects of the entity}}$ (NotNull)
2. Identification of attributes with population completeness claims: All attributes for which there is a population completeness claim are identified. In addition, the reference quantities are defined.
3. Implementation of population completeness measurements: For each attribute identified in the previous step, a population completeness measurement is implemented. $\frac{\textit{number of contained values of reference set}}{\textit{cardinality of reference set}}$ (MinPopulation)
4. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

Object incompleteness was identified as the third typical type of error in the context of completeness. Combinations of attributes are defined for checking. If all attributes of a combination are not null, the tuple satisfies minimum completeness. For example, a publication record can be evaluated as incomplete if it has neither a title nor a reference, such as a DOI or URL, to obtain further information. The particular conditions of object completeness are determined by subject matter experts. The proposed procedure for measuring the completeness of objects can be seen in Table 8.

Table 8: Measurement of the completeness of objects

II. Data quality analysis
A. Data quality dimension: completeness
c. Data level: object
1. Set conditions for completeness: Subject matter experts select attributes whose absence marks the incompleteness of the object in question. Multiple attribute combinations can be defined, at least one of which must be complete in the object for the object to be evaluated as complete.
2. Implementation of measurements: Based on the attribute combinations selected in the first step, the measurements of completeness are implemented. In each case, a measurement includes all attribute combinations of an entity. $\frac{\textit{number of complete objects}}{\textit{total number of objects of the entity}}$ (MinObject)
3. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

3.2.3 Correctness

The error types assigned to the quality dimension “correctness” refer exclusively to the attribute value level according to the groupings in Table 4. They include syntax violations, incorrect character encoding, and the use of the wrong alphabet. In [25], the metrics for assessing correctness are divided into verification using ground truth, manual verification by domain experts, verification using rules or external sources, and automatic verification (without rules and domain knowledge). Automatic verifications achieve lower accuracy than approaches in which targeted rules are derived from domain knowledge. Since the specific focus of the guide allows the inclusion of domain knowledge, the rule-based approach is used instead of non-specific automatisms. In addition, for efficiency reasons, manual verification is limited to sample verification in this workflow.

The rules to be defined in this step are used to check for the absence of specific errors rather than for the absence of all errors. For this purpose, anti-pattern [24], i. e. patterns of typical errors, are formulated. If none of the patterns apply to the test object, correctness is inferred in the context of the tested error patterns. The statement resulting from this indirect test is limited to the included errors. Its quality is lower than that of direct testing using ground truth, since it cannot be expected that all potential errors are rule-based testable.

The benchmark for verification of syntactic correctness is established through the format definitions of the attributes. Compliance with these definitions can be checked using rules. These rules should be defined as narrowly as possible to represent the correct form of the specifications as completely as possible. For example, checking the correctness of a year specification should not be limited to checking a number because years have a range of validity depending on the context. For example, the year of birth of a publication's author must be in the past.

When measuring the correctness of the encoding, it is necessary to check whether the information is presented as expected, especially with respect to language and characters. Since language correlates with the alphabet used, the assessment of the correct character encoding and the use of the correct alphabet are combined into one measurement. This check is especially useful for text attributes, since corresponding errors in other data types are covered by syntax checks. The implementation is described in Table 9.

Table 9: Measurement of the correctness of attribute values

II. Data quality analysis
B. Data quality dimension: correctness
1. Checking the availability of ground truth: The first step is to check for each attribute whether ground truth can be found to verify correctness. These are provided based on the procedure described in Table 3.1. 2. Formulation of rules: Correctness evaluation rules are formulated for all attributes for which no ground truth was identified for verification in step 1. For this purpose, syntax rules are first formulated if format definitions exist for the attribute to be verified. For attributes for which no specific syntax rules can be found, anti-patterns are developed. In this step, an analysis of outliers is performed according to the procedure suggested in Table 10. The identified error types will be documented, including examples for developing specific metrics and correcting data errors, in the next phase.
3. Implementation of measurements: Correctness measurements will be implemented attribute-by-attribute according to the procedures developed in the previous steps. $\frac{\textit{number of correct attribute values}}{\textit{total number of existing attribute values of the entity}}$ (CorrectValue)
4. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

Table 10 describes a procedure for finding anti-patterns. It examines the edge elements of sorted lists for recognizable error patterns.

Table 10: Analysis of outliers

Analysis of outliers
1. Listing of outliers: Two listings of the attribute values to be checked are output, one sorted in descending order and the other in ascending order. The listings are limited to a workable number of rows. When using the workflow, a limit of 10 000 lines has proven useful.
2. Analyze outliers: For analysis, the listings are manually checked from top to bottom for erroneous content. For each error found, an error pattern is formulated. In addition, sample values are documented in each case. The check is continued until no new errors are found for x number of lines. The choice of the number of x is a trade-off between the analysis effort and the target accuracy of the error detection. Since the analysis effort depends on the type of values contained in the attribute, the optimal number of x must be estimated in each case with respect to a particular attribute.

3.2.4 Consistency

Consistency is measured according to the groupings in Table 4 for the attribute, object, and object set levels, respectively. At the attribute level, the consistency errors "existence of synonyms" and "violation of uniqueness" were identified. Therefore, in this step, the consistency of the attributes is measured by checking them for uniqueness (uniqueness of IDs) or uniqueness (avoidance of synonyms). The implementation is shown in Table 11.

Table 11: Measuring of attribute consistency

II. Data quality analysis
C. Data quality dimension: consistency
a) Data level: attribute
1. Checking the existence of synonyms: All attributes where standardized enumerations are expected, e. g. type specifications, are checked for the occurrence of synonyms. This can be done, for example, by analyzing a listing of the different attribute values produced by a distinct operation, i. i.e., by enumerating all occurring values once.
2. Identify the attributes requiring uniqueness: In this step, all attributes are identified whose values must occur only once. This applies especially to the primary keys of the entities.
3. Implementation of measurement: For each of the attributes identified in steps 1 and 2, measurements are implemented to check the uniqueness of the contained attribute values. $\frac{\textit{number of unique attribute values}}{\textit{total number of attribute values}}$ (UniqueValue)
4. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

Inconsistency between attribute values is another typical error that occurs at the object level, i.e. within one object at a time. For example, no ISBN is expected for the publication type "journal article" and no ISSN is expected for the publication type "book". For measurement, all consistency rules of an entity are combined in one check to evaluate an object as consistent not by the absence of a contradiction, but by all contradictions. The implementation is described in Table 12.

Table 12: Measurement of the consistency of objects

II. Data quality analysis
C. Data quality dimension: consistency
b) Data level: object
1. Formulation of consistency rules: In this step, disallowed combinations of attributes are compiled. This activity requires domain knowledge, so it is performed by domain experts.
2. Implementation of measurement: The rules formulated in step 1 are implemented entity by entity. $\frac{\textit{number of non-contradictory objects}}{\textit{total number of objects in the entity}}$ (NoContradict)
3. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

Typical consistency errors at the level of object sets in an entity can be summarized as occurrences of duplicates. For checking purposes, groups of objects are formed that refer to the same real-world object. In the case of publication data, this check concerns the publications themselves and their authors. In the latter case, the authors contained in the publication data are identified. It is then determined whether two identical author names refer to the same person or whether they are namesakes. Additionally, author names are identified that differ from each other but refer to the same person. This can occur, for example, if an author has changed his or her name between two publications due to marriage. Based on the identified authors, for example, all publications by a certain person can be listed.

Some approaches found in the literature refer to the disambiguation of publications and authors, e. g. in [10, 38, 39, 40, 41]. They rely on the computation of similarities between objects, among other things, to identify duplicates. To decide whether an object is "similar enough" to another for it to be classified as a duplicate, thresholds determined based on ground truth are used. However, the thresholds found in the literature are related to specific data sets. No universally valid thresholds could be elicited.

In order to perform consolidation without ground truth, the proposal for identifying duplicates is based solely on the matching of identity-defining properties instead of a similarity consideration. The standard blocking procedure [42] is used for this purpose. It involves the generation of a key that is used to encode the selected features. This key is used to uniquely identify duplicates. When selecting the attributes to be used, care must be taken that their combination has the uniqueness of a primary key and that they are as frequent as possible. Objects must be excluded from the duplicate check if at least one of the key attributes is not assigned (zero), otherwise, it would be incorrectly concluded that the missing attributes are identical.

[43] emphasizes that this procedure is highly efficient with $O(|E|)$. However, it is susceptible to noise since even small variations in the data lead to the generation of different keys. Therefore, the method is expected to produce both a few true-positive and a few false-positive findings. To reduce the inaccuracy caused by the noise, the included text attributes are unified based on the procedure proposed in Table 13. This serves to produce a normal form to convert the text attributes that differ only due to noise into the same value. The procedure was specified by grouping duplicates of publications based on DOI and examining variations in attribute values within a duplicate group.

Table 13: Unification of text attributes

Unification of text attributes
1. Multiple consecutive separators are reduced to one separator. 2. Separators are replaced by dots. 3. Letters are converted to lowercase. 4. Quotation marks and leading and trailing spaces are deleted. 5. Characters which occur in different spellings are standardized, e. g. *é* by *e* and *ß* by *ss*.[2]

For the implementation, the procedure described in [10] is incorporated and limited to the preparation of the data, the definition of the search space, and the verification. The similarity consideration steps have been removed as they are omitted in WQP. Table 14 shows the proposed procedure for measuring the absence of duplicates.

Table 14: Measurement of consistency of object sets

II. Data quality analysis
C. Data quality dimension: consistency
c. Data level: object set
1. Identification of entities to be checked: The first step identifies all entities whose consistency cannot be verified by ground truth. 2. Selection of identity-creating data: For each entity, attributes are selected whose combination uniquely describes the identity of the associated objects. 3. Generation of keys: A key is then generated for each object. (i) The attributes selected in step 2 are unified according to the procedure proposed in Table 13. (ii) The unified values are unified using a delimiter, e. g. “#”, concatenated into a key. (iii) The key is verified by randomly checking whether objects of the same key are duplicates. Subsequently, the generation of the key is improved if necessary. For this purpose, the selection of attributes or the syntax of the concatenation can be adjusted, for example. 4. Implementation of measurement: For each of the entities identified in step 1, measurements are implemented to count the number of unique objects based on the generated keys. Here, an object is unique if the associated key is unique. In a group of identified duplicates, the first object is counted as a unique object, while all others are counted as duplicates. This counting method is used because only the second representative of the real world object is considered a duplicate. $$\frac{\textit{number of unique objects}}{\textit{total number of objects in the entity}} \qquad \text{(UniqueObject)}$$ 5. Validation of the evaluation: To ensure correct operation, the implementation is validated based on the procedure described in Table 6.

In order to reduce the resulting metrics to a manageable number, it is suggested to calculate a weighted average from the metrics of the same metric, resulting in one metric for each metric. These metrics are in turn used to calculate a weighted average for each quality dimension. This results in one key figure for each of the three prioritized quality dimensions. These key figures are not aggregated further, as the quality dimensions cannot be clearly weighted against each other due to their interdependencies.

For the determination of weightings, the form of a utility value analysis described in the “Manual for Organizational Studies and Determination of Personnel Requirements”[3] is suggested, since it is an uncomplicated procedure that can be used in any context. In this method, the elements to be weighted against each other are to be recorded in a matrix as

[2]A mapping is proposed in WorksKey.NORM_CHARS of `https://github.com/Stefan-Wolff/dqm-pipeline/blob/master/lib/duplicates.py`.

[3]`https://www.orghandbuch.de/OHB/DE/Organisationshandbuch/6_MethodenTechniken/65_Wirtschaftlichkeitsuntersuchung/652_Qualitative/qualitative-node.html`, visited on June 16, 2023

columns and rows. Each element of this matrix is compared with all the other elements, cell by cell. If the first element is more relevant than the second element, a 2 is entered in the cell. For the same relevance, a 1 is noted, and if the first element is more insignificant than the second, a 0 is noted. Then, row-by-row sums are calculated. These represent the respective weights of the elements in the associated row.

In this procedure, however, a weight of 0 is calculated for the element that is more insignificant than all others. Thus, in a calculation of the weighted average of different metrics, the most insignificant metric would not be considered, even if it is not completely irrelevant. To remedy this weakness, it is suggested that a 1 be entered in the cell where the element would be compared to itself, i.e., in the diagonal of the table. An example of this approach is given in Table 15.

Table 15: Setting weights

	A	B	C	D	Σ
A	1	1	1	0	2
B	2	1	2	1	4
C	1	0	1	0	2
D	2	1	2	1	5

3.3 Correction of data errors

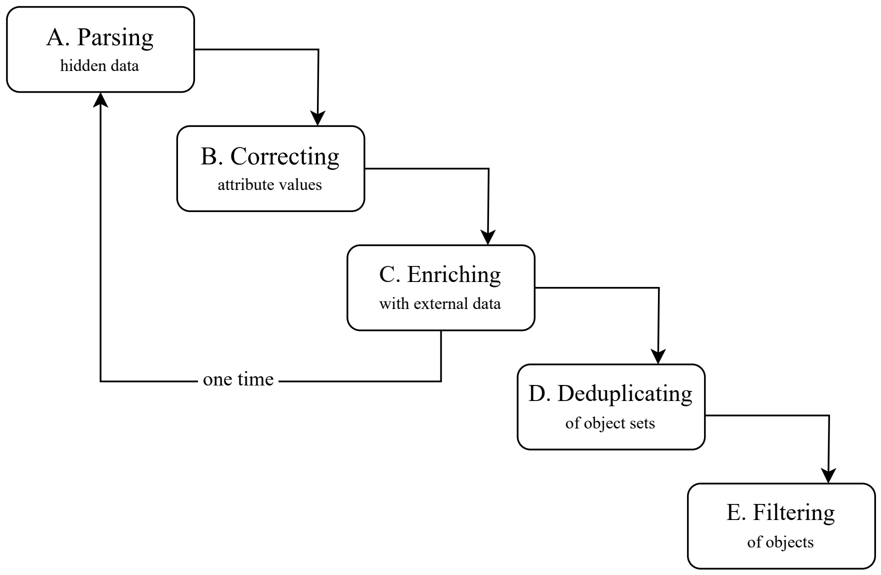

Figure 2: Flow of data correction

The procedure for correcting data errors is based on the one described in [17].

1. Parse: the first step is to establish a basic order by parsing data from unstructured or incorrectly formatted strings and, if necessary, transferring it to the intended attribute.
2. Correct and Standardize: subsequently, attribute values are corrected or standardized according to the underlying format definitions.
3. Enrich: after the data has been largely corrected, it is supplemented with missing or additional data from external data sources based on the included identifiers.
4. Matching: in this step, duplicates are identified.
5. Consolidate: in the final step, the identified duplicates are consolidated through merging.

A loop is added to this process between the "parse" and "enrich" steps to ensure that the enriched data also passes through the previous corrective actions. The "matching" and "consolidate" steps are combined into "deduplicate" because they are directly linked. In addition, a step for filtering faulty objects is added at the end of the processing chain to delete those objects that fall below the minimum quality level even after all correction steps have been performed. The process flow is outlined in Figure 2.

3.3.1 Parsing hidden data

The first step is to fully exploit the potential contained in the data by parsing data that was previously hidden. The term “hidden data” refers to data that do not appear in the expected attribute. Therefore, they cannot be used without reprocessing. For example, it is possible that the ISSN is attached to the journal title, although a separate attribute is provided for it in the data schema. To make hidden data usable, it must be identified, located, and isolated [17]. The implementation is described in Table 16.

Table 16: Parsing hidden data

III. Correction of data errors

A. Parsing of hidden data

1. Identifying hidden data:
   Each attribute is checked for hidden data.
   (i) The data schema is checked for potentially redundantly captured data.
   (ii) Each attribute is searched for hidden data based on the regular expressions defined in the analysis phase.
   (iii) Each attribute is searched for keywords. For example, a search for “ISBN” may identify a hidden ISBN of the form “ISBN: 977-NNNNN”.

2. Pattern development:
   Patterns are developed in each case to locate and isolate the data identified in step 1.
   (i) If available and specific enough, the patterns defined in the analysis phase are used to transform the hidden data into the intended attribute.
   (ii) Then, the result is validated by comparing the extracted values to the respective base values from which they were extracted.
   (iii) Based on this, the patterns are adjusted to eliminate false positives and false negatives. Steps (ii) and (iii) are repeated until there is no more potential for improvement.

3. Implementation of correction rules:
   Finally, the correction rules specified in the previous step are implemented.

3.3.2 Correcting attribute values

In this phase, incorrect attribute values are corrected and converted to a consistent format. Erroneous data can be corrected by replacing it with data from ground truth. This requires a reliable data source (ground truth). Alternatively, data can be corrected by subject matter experts. However, this method is not only costly but also carries the risk of individual errors. An examination of the organizational quality assurance processes of the CRIS domain can be found in [44]. For data for which ground truth is not available, rule-based correction is performed. The rules are developed by subject matter experts and are based on attribute format definitions and patterns of typical errors. Table 17 shows the proposed procedure for developing the corrective actions.

Table 17: Correcting attribute values

III. Correction of data errors

B. Correcting attribute values

1. Correct based on ground truth:
   The first step is to check if ground truth is available for data correction. If ground truth is identified in the form of a reliable source, the data is provided according to the procedure proposed in Table 3.1. Organizational quality assurance processes are planned for manual correction of data by subject matter experts. Subsequently, the replacement of erroneous data with data from ground truth is implemented.

2. Correction based on rules:
   Based on the specific analysis rules, systematic errors in the data are identified. For this purpose, the procedure proposed in Table 10 is used. Subsequently, patterns for locating and correcting the errors are formulated and implemented.
   (a) Erroneous components of attribute values are deleted, e. g. superfluous spaces.
   (b) Attributes whose format is defined are unified, e. g. dates.
   (c) Incorrect attribute values that cannot be corrected are deleted or accepted. This trade-off between quality and quantity is to be evaluated by subject matter experts.

3. Verification of corrections:
   Finally, samples of the corrected data are compared to the associated baseline data for verification. The correction rules are adjusted until there is no more room for improvement.

3.3.3 Enriching with external data

In the third step, the base data is completed or enhanced with data from an additional source. The implementation is described in Table 18.

Table 18: Enrichment with external data

III. Correction of data errors
C. Enrichment with external data
1. Identification of additional data sources: Data sources are sought that contain a sufficiently large subset of the baseline data to be enriched. Note that for data enrichment, a key must be available to link internal and external data sets on both sides.
2. Provision of external data sources: Based on the procedure proposed in Table 3.1, the external source data is provided for internal post-processing.
3. Implementation of data enrichment: During the implementation of the enrichment process, the internal database is matched with the data from the enrichment source. Then, all values that are null in the internal database are applied.

3.3.4 Deduplicating sets of objects

In this step, duplicates are reduced. To identify the duplicates, the procedure presented in Table 14 is used, which was already a basis for counting duplicates. To clean up the duplicates, the objects of a duplicate group are merged into a single object. It should be noted that linked data may also need to be adjusted in this step. The following strategies can be chosen for merging duplicates:

(A) All attribute values are retained and assigned as multiple values to the merged object.
(B) Only the attribute value with the highest quality is taken.
(C) The first attribute value of a descending sort order of attribute values is merged.
(D) Any attribute value is taken.

Strategy A has the advantage that no data is lost and no decision must be made to select certain data. However, this approach does not remove the inconsistency but rather moves it from the object level to the attribute level.

Strategies B and C are qualitative procedures. While the selection of the highest quality value in each case (Strategy B) is costly to implement, it produces the best result. In contrast, the selection of the first value of a descending sorted list (Strategy C) is faster to implement and still delivers accurate results. On the one hand, sorting the values leads to the selection of the longest values. On the other hand, for values of the same length, those values that do not begin or end with spaces are preferred. Strategy C thus represents a compromise between effort and quality.

Strategy D is a pragmatic procedure whose advantage lies in its small expenditure. It is suitable if no quality difference between the values can be determined. A procedure for implementing deduplication is described in Table 19.

Table 19: Deduplicating sets of objects

III. Correction of data errors
D. Deduplicating sets of objects
1. Identification of duplicates: In the first step, the identification of duplicates is implemented according to the procedure proposed in Table 14.
2. Choice of merge strategy: Next, a strategy for merging the attributes is chosen. This can be chosen individually for each attribute.
3. Implementation of the merge: Finally, the merging of the duplicates is implemented based on the groups created in step 1 and the strategies chosen in step 2.

3.3.5 Filtering objects

The minimum occupancy criteria defined in the analysis phase and the object-level consistency checks are used to identify objects to be deleted. The goal of this last step is to remove all objects that do not meet the quality requirements at the end of the correction process. Objects linked by key relationships must be taken into account.

For example, in the case of publication data, an object of the entity “person” is linked by an authorship to at least one object of the entity “publication”. If it is now determined that an author name does not meet the minimum requirements due to incompleteness, the linked publication must be deleted in addition to the author. This example also shows that the linked objects must be included in the consideration of removing an object due to insufficient data. The implementation is described in Table 20.

Table 20: Filtering objects

III. Correction of data errors
E. Filtering objects
1. Evaluate the links of incomplete objects: First, the objects that do not meet the minimum occupancy according to the procedure proposed in Table 8 are identified. Then, it is evaluated whether the joint deletion of these and their linked objects leads to an increase in quality. Based on this, it is decided in which cases the objects will be deleted.
2. Evaluate the deletion of inconsistent objects: Adequately to the first step, the objects that do not meet the consistency requirements according to the procedure proposed in Table 12 are identified. Then, it is evaluated whether the joint deletion of these and their linked objects leads to an increase in quality. Based on this determination, it is decided in which cases the objects will be deleted.
3. Implementing the deletion of objects: In the last step, the deletion of objects is implemented according to the conditions specified in steps 1 and 2.

3.4 Monitoring quality changes

Data quality monitoring aims to identify potential quality deficiencies in new or updated data before they are transferred to a target system. Quality indicators are recorded for each execution of the correction process and compared to historical indicators. While quality monitoring in a narrow sense is limited to checking quality changes, in a broader sense it often includes triggering adjustments to corrective actions [16, 17].

Due to the number of specific metrics and the resulting complexity of potential change dynamics, it is suggested that the WQP perform the monitoring of quality changes manually after each further run of the correction process and to derive appropriate improvements individually. In this process, the differences among the key figures are examined for conformity to expectations and the causes are determined in cases of deviations. The goal is to adjust the data correction process on this basis to maintain the targeted level of data quality. It should be noted that adjustments to data analysis processes can reduce comparability with historical indicators. To support manual analysis, comparisons are automated according to Table 21.

Table 21: Comparing the quality indicators

IV. Monitoring of quality changes
1. Integrating quality measurements: In the processing process, quality measurements are integrated according to the quality analyses defined in Chapter 3.2 for all prioritized quality dimensions. (i) All indicators related to the uncorrected baseline data are determined and stored. (ii) These indicators are determined and stored for the corrected data.
2. Implementation of the quality comparisons: (i) The differences of the indicators of the initial data between the last and the second to last run are determined and stored. (ii) Then, the differences of the indicators between the uncorrected baseline data and the corrected data are identified. The resulting indicators represent the improvement in data quality. They are determined for both the last run and the penultimate run. (iii) For the metrics of quality improvement identified in the previous step, the differences are formed and stored. These metrics indicate the difference in quality improvement between the last run and the previous run.

4 Application of the workflow

This chapter demonstrates the feasibility of the WQP by showing how it was applied to implement a specific quality assurance process. The source code is published as open source.[4]

4.1 Use case

A literature search system is intended to enable the retrieval of scientific publications. External data should be integrated as a database. Because the quality of this data is essential for the usefulness of the search system, it should be subjected to a quality assurance process as part of the data integration. This quality assurance was carried out using the WQP exemplary for a data set from Open Researcher and Contributor ID (ORCID). Data sets published in October 2022[5]

[4]`https://github.com/Stefan-Wolff/dqm-pipeline`

[5]`https://orcid.figshare.com/articles/dataset/ORCID_Public_Data_File_2022/21220892`, accessed 06/18/2023

were used. ORCID is a nonprofit system whose primary goal is to make researchers identifiable with respect to their publications by assigning them a persistent identifier, the ORCID iD [45]. In addition, metadata related to each individual's scientific work is collected, e. g. name variants and affiliations under which publications, projects, and patents are published. The complete listing of all data available through the public API version 3.0 can be found in the schema definitions[6].

The WQP was implemented based on Python 3 and Apache Spark. Apache Spark was used because this software is optimized for efficient processing of large amounts of data. Python was used because it is currently the most widely used programming language and provides access to Apache Spark's API through the PySpark library.

4.2 Application results

The improved data quality resulting from the realization of the WQP is summarized in Table 22. This lists the quality metrics before and after the execution of the WQP. These underlie the measurement results stored in "repo/quality_ORCID_2023.json"[7]. Since these metrics in themselves only indicate the quantity of error-free quality checks, the improvement in data quality cannot be read directly. Therefore, the proportion of *eliminated quality errors* was additionally calculated for illustration purposes.

Table 22: Improving data quality

Metrics	Data quality before performing the WQP	Data quality after performing the WQP	Eliminated quality errors
MinLength	99%	100%	**100%**
MinObject	91%	100%	**100%**
MinPopulation	99%	99%	**0**
MinValue	19%	21%	3%
NotNull	60%	67%	17%
CorrectValue	93%	99%	**79%**
NoContradict	74%	85%	40%
UniqueObject	48%	100%	**100%**
UniqueValue	84%	94%	**64%**
Average quality errors removed			**56%**

Legend:

The numbers are rounded.
Green marks an above-average number of quality errors removed.
Red marks quality aspects that have not been improved.

The table shows that the fulfillment of the quality aspect measured by the metric "NotNull" was increased from 60 to 67%, with 17% of the quality errors corrected. It also shows that a total of 56% of the quality errors were eliminated.

4.3 Discussion of results

The applicability of the WQP was tested based on publication data from ORCID for the specific use case of a literature search system. The implementation resulted in a processing chain, the execution of which gradually improved the quality of the data. As a result, a total of 56% of the quality deficiencies identified in the process were resolved.

The quality metrics presented in Chapter 4.2 show that the quality before the implementation of the WQP was the lowest with respect to the quality aspects addressed by the metrics "MinValue", "NotNull", and "UniqueObject". The low ratio of the first metric means that often only the year occurres in the baseline data instead of the publication date.

The figure for the second metric indicates that numerous attributes were assigned zero, i.e., they did not contain any information. It can be concluded that the input masks used to enter these data did not validate the publication date and thus accepted incomplete data. Moreover, it appears that only a few attributes were mandatorily requested, which explains the large number of missing entries.

However, the low ratio calculated by the metric "UniqueObject" indicates that approximately every second object in the base data is a duplicate. A closer look at the data structure in ORCID shows that this high number of duplicates is expected because it is enforced by the data structure. This structure ensures that each scientist generates his or her

[6]`https://github.com/ORCID/orcid-model/blob/master/src/main/resources/record_3.0`, accessed 06/18/2023
[7]https://github.com/Stefan-Wolff/dqm-pipeline/blob/master/repo/quality_ORCID_2023.json

publication lists independently from the entries of the others. Therefore, those publications in which several authors were involved are recorded several times in the system, once by each author.

By applying the WQP, the errors identified by the metrics “MinLength”, “MinObject”, and “UniqueObject” were fully resolved. The largest increase in quality refers to the latter metric, whose quality was increased from 48 to 100%, i.e., the largest increase in quality was achieved by reducing duplication. This was followed by the correction of incorrectly formatted data (metric “CorrectValue”), which was improved to 79%.

In contrast, the quality aspects of the data quality dimension “Completeness” were improved to a lesser extent. The errors in the metrics “MinLength” and “MinObject” were completely eliminated. However, they occurred much less frequently than other quality aspects of completeness, such as the presence of attribute values (metric “NotNull”).

In summary, data quality deficiencies that resulted from an incorrect type of specification, e. g. incorrect formatting or redundant data entry (duplicates), were more successfully corrected. In contrast, quality errors that resulted from a lack of available information were resolved to a lesser extent. It can therefore be hypothesized that by adding additional sources of publication data, the resulting data quality can be further improved. This hypothesis is based on the assumption that publication data from different sources are significantly complementary in their scope. However, this assumption does not include the extent to which potential data providers copy publication data from each other.

5 Conclusion and outlook

Ensuring the quality of publication data is crucial for various contexts of use, such as literature search systems and CRIS, as the available data sets are often of insufficient quality. However, the implementation of a specific process to improve data quality requires a concept of how to proceed, especially regarding how to analyze the data quality and how to correct the data errors. The goal of this work was to support this implementation by providing a workflow that is as detailed as possible and describes the procedure in a structured way. For this purpose, the WQP was developed, and its performance was demonstrated on the dataset from ORCID.

To establish the workflow, those quality aspects that are particularly relevant for publication data were first identified. Frequently occurring error types were compiled from the literature and mapped to the data quality dimensions most relevant for publication data – completeness, correctness, and consistency. Nine metrics were specified to assess the quality of the various aspects. The resulting metrics were used to determine the potential for improvement of the baseline data and to verify the success of the data corrections.

The procedure for correcting the data was extracted from the literature and concretized. It includes parsing of hidden data, correction of erroneous attribute values, enrichment with external data, deduplication, and filtering of objects. The result is a detailed description of the data correction steps specific to the quality attributes of a given set of publication data. An exemplary execution of the workflow was used to demonstrate its performance; the workflow automatically corrected 56 % of the identified errors. The implementation also provides input for further applications, e. g. a set of regular expressions for verification of attributes typical of publication data. In addition to the core of the quality enhancement process – analyzing data quality and correcting data errors – the workflow includes the process for providing the data to be processed, as well as tools for continuously monitoring quality development over multiple executions.

The results of the practical implementation of the WQP serve to recommend its use, as it can lead to a significant increase in data quality. Therefore, the research question of this work is answered as follows: by using the WQP, the quality of publication data can be significantly increased. In particular, the quality aspects that can be improved are those related to the way the data are specified, e. g. the formatting of the data or their one-time collection. In contrast, quality aspects resulting from the insufficient completeness of the data can be improved to a lesser extent.

Through the WQP, this work contributes to the conceptual development of processes to improve the quality of publication data. The detailed description enables a targeted application of the workflow and promotes its extensibility because it minimizes the scope for interpretation.

The findings of this work indicate the following conclusion: the quality of publication data can be improved by implementing a targeted process. However, the scope of the WQP (4 phases and 31 individual steps) indicates that some effort is required to implement this process. This finding raises the question of whether the efficiency of quality assurance can be increased by preventing certain quality errors from occurring during the acquisition process rather than correcting them after the fact. For example, standards, e. g. mandatory specification of ORCID iDs of authors, could greatly simplify DQM.

This work also leaves unanswered the question of whether adding more data sets to supplement the data can further increase the resulting data quality. It is unclear to what extent potential data providers exchange publication data with each other. If such sharing is extensive, a significant impact would not be expected from matching with additional data sources.

In addition, developing a procedure to optimize the execution order of the individual data correction steps exceeded the scope of this work. However, it is assumed that a corresponding extension of the workflow would lead to increased performance.

The task of deduplication offers further potential for connection. With the procedure used in WQP, a minimum of false-positive duplicate finds was achieved. However, this method generates only a small number of correct-positive finds. It was selected because no ground truth is required. The workflow could be extended to include methods that use ground truth and, thus provide a comprehensive increase in the number of correct-positive duplicate finds.

In the future, the author of this work plan to apply the WQP to further data sources and develop it further if necessary. In addition, the work will be published to make its results available to researchers and, in particular, to the specialist community.

References

[1] Lisa Madlberger, Andreas Thöni, Peter Wetz, Alexander Schatten, and A. Min Tjoa. Ontology-based data integration for corporate sustainability information systems. In *Proceedings of International Conference on Information Integration and Web-Based Applications & Services*, IIWAS '13, page 353–357. Association for Computing Machinery, December 2013.

[2] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, page 233–246. Association for Computing Machinery, June 2002.

[3] Gianluca Cima, Marco Console, Maurizio Lenzerini, and Antonella Poggi. Abstraction in data integration. In *Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '21. Association for Computing Machinery, 2021.

[4] Ashwin Machanavajjhala and Jerome P. Reiter. Big privacy: Protecting confidentiality in big data. *XRDS*, 19(1):20–23, September 2012.

[5] Otmane Azeroual, Gunter Saake, and Eike Schallehn. Analyzing data quality issues in research information systems via data profiling. *International Journal of Information Management*, 41:50–56, August 2018.

[6] Otmane Azeroual, Gunter Saake, Mohammad Abuosba, and Joachim Schöpfel. Data Quality as a Critical Success Factor for User Acceptance of Research Information Systems. *Data*, 5(2):35, June 2020.

[7] Anders Haug, Frederik Zachariassen, and Dennis van Liempd. The costs of poor data quality. *Journal of Industrial Engineering and Management*, 4:168–193, July 2011.

[8] Nuno Laranjeiro, Seyma Nur Soydemir, and Jorge Bernardino. A survey on data quality: Classifying poor data. In *2015 IEEE 21st Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 179–188, November 2015.

[9] Helen-Tadesse Moges, Véronique Van Vlasselaer, Wilfried Lemahieu, and Bart Baesens. Determining the use of data quality metadata (dqm) for decision making purposes and its impact on decision outcomes — an exploratory study. *Decision Support Systems*, 83:32–46, March 2016.

[10] Otmane Azeroual, Meena Jha, Anastasija Nikiforova, Kewei Sha, Mohammad Alsmirat, and Sanjay Jha. A Record Linkage-Based Data Deduplication Framework with DataCleaner Extension. *Multimodal Technologies and Interaction*, 6(4):27, April 2022.

[11] Sakda Loetpipatwanich and Preecha Vichitthamaros. Sakdas: A Python Package for Data Profiling and Data Quality Auditing. In *2020 1st International Conference on Big Data Analytics and Practices (IBDAP)*, pages 1–4, September 2020.

[12] Xu Chu, John Morcos, Ihab F. Ilyas, Mourad Ouzzani, Paolo Papotti, Nan Tang, and Yin Ye. KATARA: A Data Cleaning System Powered by Knowledge Bases and Crowdsourcing. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 1247–1261. Association for Computing Machinery, May 2015.

[13] Mazen Sh and Nicholas Berente. Software Development Outsourcing, Asset Specificity, and Vendor Lock-in Software Development Outsourcing, Asset Specificity, and Vendor Lock-in Emergent Research Forum (ERF). In *AMCIS 2019 Proceedings*, 17, January 2019.

[14] Frederik Ahlemann, Sven Dittes, Tim Fillbrunn, Kevin Rehring, Stefan Reining, and Nils Urbach. Managing In-Company IT Standardization: A Design Theory. *Information Systems Frontiers*, 25(3):1161–1178, June 2023.

[15] Maria Teresa Baldassarre, Ismael Caballero, Danilo Caivano, Bibiano Rivas Garcia, and Mario Piattini. From big data to smart data: a data quality perspective. In *Proceedings of the 1st ACM SIGSOFT International Workshop on Ensemble-Based Software Engineering*, EnSEmble 2018, pages 19–24. Association for Computing Machinery, November 2018.

[16] Ikbal Taleb, Mohamed Adel Serhani, Chafik Bouhaddioui, and Rachida Dssouli. Big data quality framework: a holistic approach to continuous quality management. *Journal of Big Data*, (1):76, May 2021.

[17] Otmane Azeroual, Gunter Saake, and Mohammad Abuosba. Data Quality Measures and Data Cleansing for Research Information Systems, January 2018.

[18] Richard Y. Wang and Diane M. Strong. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, 12(4):5–33, 1996.

[19] Oumaima Reda, Imad Sassi, Ahmed Zellou, and Samir Anter. Towards a Data Quality Assessment in Big Data. In *Proceedings of the 13th International Conference on Intelligent Systems: Theories and Applications*, SITA'20, pages 1–6. Association for Computing Machinery, November 2020.

[20] Widad Elouataoui, Imane El Alaoui, Saida El Mendili, and Youssef Gahi. An Advanced Big Data Quality Framework Based on Weighted Metrics. *Big Data and Cognitive Computing*, 6(4):153, December 2022.

[21] Ikbal Taleb and Mohamed Adel Serhani. Big Data Pre-Processing: Closing the Data Quality Enforcement Loop. In *2017 IEEE International Congress on Big Data (BigData Congress)*, pages 498–501, June 2017.

[22] Cinzia Cappiello, Walter Samá, and Monica Vitali. Quality awareness for a Successful Big Data Exploitation. In *Proceedings of the 22nd International Database Engineering & Applications Symposium*, IDEAS '18, pages 37–44. Association for Computing Machinery, June 2018.

[23] Mohamed Adel Serhani, Hadeel T. El Kassabi, Ikbal Taleb, and Alramzana Nujum. An Hybrid Approach to Quality Evaluation across Big Data Value Chain. In *2016 IEEE International Congress on Big Data (BigData Congress)*, pages 418–425, June 2016.

[24] Arno Kesper, Viola Wenz, and Gabriele Taentzer. Detecting quality problems in research data: a model-driven approach. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, MODELS '20, pages 354–364. Association for Computing Machinery, October 2020.

[25] Valerie Restat, Meike Klettke, and Uta Störl. *FAIR is not enough – A Metrics Framework to ensure Data Quality through Data Preparation*. Gesellschaft für Informatik e.V., 2023.

[26] Mauricio A. Hernández and Salvatore J. Stolfo. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Data Mining and Knowledge Discovery*, 2(1):9–37, January 1998.

[27] Alvaro Monge. Matching Algorithms within a Duplicate Detection System. *IEEE Data(base) Engineering Bulletin*, January 2000.

[28] Ikbal Taleb, Mohamed Adel Serhani, and Rachida Dssouli. Big Data Quality: A Data Quality Profiling Model. In Yunni Xia and Liang-Jie Zhang, editors, *Services – SERVICES 2019*, Lecture Notes in Computer Science, pages 61–77, Cham, June 2019. Springer International Publishing.

[29] Reza Vaziri, Mehran Mohsenzadeh, and Jafar Habibi. TBDQ: A Pragmatic Task-Based Method to Data Quality Assessment and Improvement. *PLOS ONE*, 11(5):e0154508, May 2016.

[30] Juliana Freire, Philippe Bonnet, and Dennis Shasha. Computational reproducibility: State-of-the-art, challenges, and database research opportunities. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 593–596. Association for Computing Machinery, May 2012.

[31] Otmane Azeroual and Joachim Schöpfel. Quality Issues of CRIS Data: An Exploratory Investigation with Universities from Twelve Countries. *Publications*, 7(1):14, March 2019.

[32] Bernd Heinrich, Diana Hristova, Mathias Klier, Alexander Schiller, and Michael Szubartowicz. Requirements for Data Quality Metrics. *Journal of Data and Information Quality*, 9(2):12:1–12:32, January 2018.

[33] Otmane Azeroual, Gunter Saake, Mohammad Abuosba, and Joachim Schöpfel. Quality of Research Information in RIS Databases: A Multidimensional Approach. In Witold Abramowicz and Rafael Corchuelo, editors, *Business Information Systems*, Lecture Notes in Business Information Processing, pages 337–349. Springer International Publishing, May 2019.

[34] Otmane Azeroual. Data Wrangling in Database Systems: Purging of Dirty Data. *Data*, 5(2):50, June 2020.

[35] Otmane Azeroual and Włodzimierz Lewoniewski. How to Inspect and Measure Data Quality about Scientific Publications: Use Case of Wikipedia and CRIS Databases. *Algorithms*, 13(5):107, May 2020.

[36] Valerie Restat, Gerrit Boerner, André Conrad, and Uta Störl. GouDa - generation of universal data sets: improving analysis and evaluation of data preparation pipelines. In *Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning*, DEEM '22, pages 1–6. Association for Computing Machinery, June 2022.

[37] Leo L. Pipino, Yang W. Lee, and Richard Y. Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, April 2002.

[38] Cristian Santini, Genet Asefa Gesese, Silvio Peroni, Aldo Gangemi, Harald Sack, and Mehwish Alam. A Knowledge Graph Embeddings based Approach for Author Name Disambiguation using Literals, June 2022.

[39] Tobias Backes and Stefan Dietze. Lattice-based progressive author disambiguation. *Information Systems*, 109:102056, November 2022.

[40] Zeyd Boukhers and Nagaraj Bahubali Asundi. Deep Author Name Disambiguation using DBLP Data, March 2023.

[41] IJAZ HUSSAIN and SOHAIL ASGHAR. Incremental author name disambiguation using author profile models and self-citations. *Turkish Journal of Electrical Engineering and Computer Sciences*, 27(5):3665–3681, January 2019.

[42] Ivan P. Fellegi and Alan B. Sunter. A Theory for Record Linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, December 1969.

[43] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. Blocking and Filtering Techniques for Entity Resolution: A Survey. *ACM Computing Surveys*, (2):31:1–31:42, March 2020.

[44] Otmane Azeroual, Mohammad Javad Ershadi, Amir Azizi, Melikasadat Banihashemi, and Reza Edris Abadi. Data Quality Strategy Selection in CRIS: Using a Hybrid Method of SWOT and BWM. *Informatica*, 45(1), March 2021.

[45] About ORCID. `https://info.orcid.org/what-is-orcid/`.