José Barateiro, Helena Galhardas

A survey of data quality tools

Data quality tools aim at detecting and correcting data problems that affect the accuracy and efficiency of data analysis applications. We propose a classification of the most relevant commercial and research data quality tools that can be used as a framework for comparing tools and understand their functionalities.

1 Introduction

Today's organizations require a high-level of data quality in order to run efficiently their data analysis applications (e.g. decision support systems, data mining, customer relationship management) and produce accurate results. The amount of data currently handled is huge, no adequate data entry control is usually performed and data is often obtained by merging different data sources. As a consequence, data anomalies such as inconsistencies, missing values, duplicate records and non-standard representations of data are encountered more frequently than desired. The existence of these data problems, commonly called *dirty data*, degrades significantly the quality of the information with a direct impact on the efficiency of the business that it supports.

The elimination of dirty data in information systems is generally called *data cleaning* (also known as data cleansing or scrubbing) and aims at obtaining data of high quality. It is a crucial task in various application scenarios. Within a single data source (e.g., list of customers), it is important to correct integrity problems, standardize values, fill in missing data and consolidate duplicate occurrences. The construction of a data warehouse (DW) [1] requires an important step called the ETL (Extraction, Transformation and Loading) process that is responsible for extracting information from the operational data sources, transforming and loading it into the DW data schema. Data migration processes (for example, when a software package is discontinued) aim at converting legacy data stored in sources with a certain schema into target data sources whose schema is distinct and predefined [2].

Depending on the context where data cleaning is applied, it may be known under different names. For instance, when detecting and eliminating duplicate records within a single file, a record-linkage or duplicate elimination process takes place. In a DW context, the ETL process encloses data cleaning tasks, and some authors [3] designate a specific data store, named *data staging area* in the DW architecture, for gathering the intermediate results of data cleaning transformations. In this paper, we refer to a *data quality* process as the sequence of data transformations (often modeled as a graph of [4]) that must be applied to data with problems in order to get data of good quality. Data quality tools are the software artifacts that take part in this process.

Current technology tries to solve data quality problems in three different ways: *(i)* ad-hoc programs written in a programming language like C or Java, or in a Relational Database Management System (RDBMS) proprietary language like Oracle PL/SQL [5]; *(ii)* RDBMS mechanisms for guaranteeing integrity constraints; or *(iii)* data transformation scripts using a data quality tool. The use of a general purpose or an RDBMS proprietary language makes data quality programs difficult to maintain and optimize. The mechanisms supported by an RDBMS enforce integrity constraints but do not address a considerable number of problems that affect the contents of data records (e.g., data inconsistencies, data errors). There is an extensive market of tools to support the transformation of data to be loaded in a data warehouse, the so-called ETL tools, that also provide some data cleaning functionalities. Other data quality tools have been developed from scratch to address specific data quality problems, such as address standardization and name matching.

In this paper, we survey the most prominent (commercial and research) data quality tools. Our analysis use three different criteria. First, we present a taxonomy of data quality problems inspired by previous work found in the literature. Second, we classify tools according to a set of generic functionalities they usually support. These functionalities enclose the types of input and target data sources supported, the kind of interface provided, etc. Third, we group the tools in six classes depending on the aspect of data quality they address. Finally, we propose a correspondence between the types of data problems and some of the classes of tools previously identified. This mapping is particularly useful to understand how far a given data quality tool is able to correct data problems.

The paper is organized as follows. Section 2 presents the taxonomy of data quality problems. Section 3 summarizes the general functionalities for all the data quality tools analyzed. In Section 4, we group the tools in six categories. Then, Section 5 describes how data quality problems are addressed by data quality tools. Finally, we conclude in Section 6.

2 Data quality problems

The taxonomy of data problems presented here is based on four relevant works found in the literature [6, 7, 8, 9]. We divide data quality problems in schema level and instance level data problems. Schema level data problems can be avoided with an improved schema design and do not depend on the actual data contents. Instance level data problems are related with the data contents and cannot be avoided with a better schema definition as the schema definition languages are not powerful enough to specify all the required data constraints. In our taxonomy, instance level data problems include all data problems that are not schema level data problems and so the taxonomy is complete at this point.

2.1 Schema level data quality problems

Schema level data quality problems can be prevented with an improved schema design, schema translation and integration. Current RDBMS provide important mechanisms to assure schema definitions. Therefore, we distinguish between schema level data problems that can be avoided by a RDBMS and schema level data problems that cannot be avoided by a RDBMS.

2.1.1 Avoided by a RDBMS

During database design, *integrity constraints* can be defined to specify conditions that must be satisfied by database

objects [10]. SQL:1999 [11] provides a language to support the specification of the following declarative integrity constraints: *(i) not null* constraint to prevent a column from taking a null value; *(ii) default* constraint to specify the default value for a given column; *(iii) unique* constraint to define that a column or a set of columns must have unique values within a table; *(iv) primary key* constraint that corresponds to *unique* and *not null* constraints; *(v) referential integrity* constraint to specify attributes whose values must match the values of a primary or unique key of a foreign table; *(vi) check* constraint to specify a user defined condition; *(vii) domain* constraint to define restricted column domain values; *(viii) assertion* constraint defines a table-independent integrity constraint. SQL:1999 also allows a procedural definition of integrity constraints using *triggers*, that are procedures invoked by the RDBMS in response to specific database events.

The following list summarizes data quality problems that can be avoided with an adequate integrity constraint definition:

- Missing data: Data that has not been filled. A *not null* constraint can avoid this problem.
- Wrong data type: Violation of a data type constraint e.g., employee's age is »XY«. *Domain* constraints can avoid this problem.
- Wrong data value: Violation of a data range constraint. If an employee's age must belong to the range [18, 65], then an age of 15 is a wrong data value. *Check* and *domain* constraints are used to avoid this problem.
- Dangling data: Data in one table has no counterpart in another table. For example, a department identifier does not exist in the Department table and there is a reference to this value in the Employee table. This problem is addressed by *referential integrity* constraints (i.e., foreign keys).
- Exact duplicate data: Different records have the same value in a field (or a combination of fields) for which duplicate values are not allowed (for instance, the employee's social security number). *Unique* and *primary key* constraints can avoid exact duplicates.
- Generic domain constraints: Records or attribute values that violate a domain restriction (for example, attribute dependencies or a pre-defined maximum number of rows). *Domain* and *assertion* constraints intend to avoid this problem. However, these features are not provided by most of the RDBMS (e.g., Oracle 9i) that handle this data problem using triggers.

2.1.2 Not avoided by a RDBMS

The following data quality problems cannot be handled by RDBMS integrity constraints:

- Wrong categorical data: A category value that is out of the category range (e.g. countries and respective states). The use of a wrong abstraction level (e.g. »frozen food« or »frozen pizza« instead of »food«) is also considered a type of wrong categorical data.
- Outdated temporal data: Data that violates a temporal constraint that specifies the time instant or interval in which data is valid. For example, an employee salary is no longer valid when this employee salary is raised.
- Inconsistent spatial data: Inconsistencies between spatial data (e.g. coordinates, shapes) when they are stored in multiple fields. For instance, the point coordinates in a rectangle table should be combined to yield a closed rectangle.
- Name conflicts: The same field name is used for different objects (homonyms) or different names are used for the same object (synonyms).
- Structural conflicts: Different schema representations of the same object in different tables or databases. For example, an address can be represented in a free form field or decomposed in the fields street, city, state, etc.

Although, name and structural conflicts can occur within a single data schema, they frequently arise in a multi-schema scenario.

2.2 Instance level data quality problems

Instance level data problems refer to errors and inconsistencies of data that are not visible or avoided at schema level. Note that data instances also reflect schema-level problems (e.g., a record with a null value in a required field). We consider that instance level data problems are divided into single record and multiple records problems.

2.2.1 Single record

Single record data problems concern one or various attributes of a single record. In other words, this kind of problem is related to a single entity and does not depend on other information stored in the database.

- Missing data in a *not null* field: An attribute is filled with some »dummy« value. For instance, a social security number -999999 is an undefined value used to surpass the *not null* constraint.
- Erroneous data: The data is valid but does not conform to the real entity. An example of erroneous data is 31 for an Employee's age when she really is 30 years old.
- Misspellings: Misspelled words in database fields (e.g., »Jhon Stevens« instead of »John Stevens«).
- Embedded values: The existence of extraneous data in some data field. A common example of embedded values is the insertion of a title in a name field (e.g. »President John Stevens«).
- Misfielded values: Data is stored in the wrong field. For instance, consider the value »Portugal« in a city attribute.
- Ambiguous data: Data that can be interpreted in more than one way, with different meanings. Ambiguous data may occur due to the existence of abbreviation or an incomplete context, as follows:
  - Abbreviation: The abbreviated name »J. Stevens« can be expanded in different ways, as: »John Stevens«, »Jack Stevens«, »Jeff Stevens«, etc.
  - Incomplete context: The city name »Miami« can stand for the State of Florida or the State of Ohio.

2.2.2 Multiple records

Multiple record data problems cannot be detected by considering each record separately as the data problem concerns more than one record. Notice that multiple record problems can occur among records belonging to the same entity set (or table) or to different entity sets (corresponding to different tables or even to different databases).

- Duplicate records: Records that stand for the same real entity and do not contain contradicting information. The following Employee records are considered duplicates: Emp1(Name="John Stevens" Address="223, First Avenue,

New York City“, Birth=01/01/1975); Emp2(Name=“J. Stevens“ Address= “23, 1st Avenue, New York“, Birth=01/ 01/1975).

- Contradicting records: Records that stand for the same real entity and contain some kind of contradicting information. Consider again Emp1 and Emp2, but with the following information: Emp1(Name=“John Stevens« Address=“23, First Avenue, New York City“, Birth=01/01/1975); Emp2(Name =“John Stevens“ Address=“23, First Avenue, New York City“, Birth=01/01/ 1965).
- Non-standardized data: Different records do not use the same representations of data, thus invalidating their comparison.
  - Transposition of words: In a single field, words can appear with different orderings. For example, the names »John Stevens« and »Smith, Jack« do not use the same ordering rule.
  - Encoding format: Use of different encoding formats, e.g., ASCII, UTF-8.
  - Representation format: Different format representations for the same information. An example is the currency format which can be €10.5, 10.5€, etc.
  - Measurement unit: Different units used in distinct records, for example, distances in cm and inches.

3 Generic functionalities

In this section, we introduce the major features of data quality tools that we analyzed. First, we describe each feature. Then, we summarize, in Tables 1 and 2, the functionalities observed for the set of commercial and research tools analyzed. We would like to remark that the information has been collected from company web pages and white papers for commercial tools and scientific papers describing research tools.

Data sources: The ability to extract data from different and heterogeneous data sources. Data sources may be relational databases, flat files, XML files, spreadsheets, legacy systems, and application packages, such as SAP, Web-based sources and EAI (Enterprise Application Integration) software. Some commercial tools (e.g. Sunopsis [12]) support only relational databases. Other tools like ETLQ [13, 14] support a wide range of data source types, from conventional connectors (e.g. ODBC, JDBC) to application packages like SAP R/3.

Extraction capabilities: The process of extracting data from data sources should provide the following important capabilities: *(i)* the ability to schedule extracts by time, interval or event; *(ii)* a set of rules for selecting data from the source and *(iii)* the ability to select and merge records from multiple sources. A number of commercial tools, as Informatica [15] and Data Integrator [16], provide most of these extraction functionalities.

Loading capabilities: The process of loading data into the target system should be able to: *(i)* load data into multiple types of target systems; *(ii)* load data into heterogeneous target systems in parallel; *(iii)* both refresh and append data in the target data source and *(iv)* automatically create target tables. Sagent [17] is an example of a commercial tool that provides all loading capabilities listed above.

Incremental updates: The ability to incrementally update data targets, instead of rebuilding them from scratch every time. Ideally, the incremental update strategy should be performed at extraction time so that only new or updated

Tool	Data sources	Extraction	Loading	Incremental updates	Interface	Metadata repository	Performance	Versioning	Function library	Language binding	Debugging	Exceptions	Data lineage
Centrus Merge/Purge	DB	-	-	-	G	-	-	-	-	-	-	-	-
ChoiceMaker	DB, FF	-	-	-	G	Y	Y	-	Y	N	Y	Y	Y
Data Integrator	Several	Y	Y	Y	G	Y	Y	Y	Y	-	Y	Y	Y
DataBlade	Informix	-	Informix	-	G	-	-	-	-	-	Y	X	X
DataFusion	DB	Y	DB	Y	G	-	Y	Y	Y	N	Y	X	-
DataStage	Several	Y	Y	-	G	Y	Y	Y	Y	Y	Y	Y	Y
DeDupe	DB	-	-	-	G	-	-	-	-	-	-	-	-
dfPower	Several	Y	Y	-	G	Y	Y	-	-	-	-	-	-
DoubleTake	ODBC	-	-	-	G	-	-	-	-	Y	-	-	-
ETI*Data Cleanser	Several	-	-	-	G	Y	Y	-	Y	Y	-	Y	-
ETLQ	Several	Y	Y	-	G	Y	Y	Y	Y	N	-	-	-
Firstlogic	DB, FF	Y	Y	-	G	Y	Y	-	Y	Y	-	-	-
Hummingbird ETL	Several	Y	Y	Y	G	Y	Y	Y	Y	N	Y	M	Y
Identity Search Server	DB	-	-	Y	G	Y	-	-	-	-	-	-	-
Informatica ETL	Several	Y	Y	Y	G	Y	Y	Y	Y	Y	Y	Y	Y
MatchIT	DB	-	-	-	G	-	-	-	-	-	Y	-	-
Merge/Purge Plus	-	-	-	-	-	-	-	-	-	-	-	-	-
Migration Architect	Several	-	-	-	G	Y	-	-	-	-	-	-	-
NaDIS	-	X	-	X	G	X	-	-	X	X	-	X	X
QuickAddress Batch	ODBC	X	-	X	G	X	-	-	X	X	-	X	X
Sagent	Several	Y	Y	X	G	Y	Y	X	Y	N, SQL	-	-	-
SQL Server 2000 DTS	Several	Y	Y	X	G	X	-	-	Y	N	X	X	X
SQL Server 2005	Several	Y	Y	-	G	-	-	-	Y	N	-	-	-
Sunopsis	DB, FF	Y	Y	Y	G	Y	Y	Y	X	SQL	Y	Y	X
Trillium	Several	Y	Y	-	G	Y	Y	Y	Y	N	Y	-	Y
WizRule	DB, FF	-	-	-	G	-	Y	-	-	-	-	-	-
WizSame	DB, FF	-	-	-	G	-	Y	-	-	-	-	-	-
WizWhy	DB, FF	-	-	-	G	-	Y	-	-	-	-	-	-

Tab. 1: General functionalities of commercial data quality tools. Y: supported; X: not supported; -: unknown information; N: native; DB: only relational databases; FF: only flat files; G: graphical; M: manual.

Tool	Data sources	Extraction	Loading	Incremental updates	Interface	Metadata repository	Performance	Versioning	Function library	Language binding	Debugging	Exceptions	Data lineage
Ajax	DB, FF	Y	DB	X	NG	X	Y	X	Y	JAVA	Y	Y	Y
Arktos	JDBC	-	JDBC	-	G	-	-	X	-	-	Y	Y	-
Clio	DB, XML	X	-	X	G	Y	Y	X	-	-	Y	-	-
Flamingo Project	DB	-	DB	-	NG	-	-	-	-	-	-	-	-
FraQL	-	-	-	-	NG	-	-	-	Y	N	-	-	-
IntelliClean	DB	-	DB	-	NG	-	-	-	-	-	Y	X	-
Ken State University	-	-	X	-	NG	-	-	-	-	-	-	Y	-
Potter's Wheel	Y	-	ODBC	-	G	Y	-	-	-	-	Y	-	-
TranScm	Y	-	-	-	G	-	-	-	Y	N	-	-	-

Tab. 2: General functionalities of research data quality tools. Y: supported; X: not supported; -: unknown information; DB: relational databases; FF: flat files; G: graphical; NG: non-graphical.

records are extracted, which significantly improves the efficiency and time cost. An alternative strategy is to perform incremental updates at loading time, which is a very costly and inefficient process. Hummingbird ETL [18] provides incremental updates at extraction time, using Sybase Replication Server as the engine to detect modified data.

Interface: Some products offer an integrated visual development environment that makes them easier to use. These graphical tools enable the user to define data quality processes modeled as workflows using a point-and-click interface. Most of the commercial tools, e.g., FirstLogic [19, 20], dfPower [21, 22], and Trillium [23] provide a fully integrated graphical interface. Non-graphical tools like Ajax [4, 24] and DataFusion [25], usually provide a language to define data quality programs.

Metadata repository: A repository that stores data schemas and information about the design of the data quality process. This information is consumed during the execution of data quality processes. Some tools, like Informatica, use a relational database to store the metadata repository.

Performance techniques: Set of features to speed up data cleaning processes and to ensure scalability. Important techniques to improve performance are partitioning, parallel processing, threads, clustering and load balancing. DataStage [26] and Hummingbird ETL, for instance, provide parallel processing (partitioning and pipelining) and multi-thread execution of data transformations, respectively.

Versioning: A version control mechanism with standard control options (e.g. check in, check out) that allows developers to keep track of different development versions of the source code. Some commercial tools, e.g., DataStage and Hummingbird ETL, provide version control support.

Function library: Set of pre-built functions, such as data type converters and standardization functions, that address specific data quality problems. An important feature is the possibility to extend the function library with new functions. This can be achieved through a programming language or by adding external functions from a Dynamic Link Library (DLL). A number of commercial tools, e.g. Informatica ETL, Data Integrator and Hummingbird ETL, provide function libraries with extension capabilities.

Language binding: An integrated programming language support to develop new functions in order to extend the function library. This support may range from an existing programming language (like C or Perl) to a native language. DataStage supports both a native language and existing programming languages (Perl and Basic).

Debugging and tracing: Tracing facilities document the execution of data quality programs with useful information (e.g., start and end execution times of important routines, the number of input and output records). Usually, tracing facilities are provided in the form of a detailed log file (e.g., DataStage and DataIntegrator). More sophisticated mechanisms can be found in DataFusion [25]. Debugging is the facility for tracking and analyzing the execution of data quality programs in order to detect problems and refine the specification of data transformation rules. Sophisticated debugging mechanisms are supported by research tools Ajax and Arktos [27].

Exception detection and handling: Exceptions are the set of input records for which the execution of part of the data quality process fails. Exception handling can be *manual* using a given user interface, or *automatic* by ignoring/deleting exception records, or reporting them into an exception file or table. Both commercial (e.g., Sunopsis and DataStage) and research tools (e.g., Ajax and Arktos), provide exception detection and handling.

Data lineage: Data lineage or provenance identifies the set of source data items that produced a given data item [28]. Both commercial (e.g., Data Integrator) and research tools (e.g., Ajax) provide data lineage and inspection facilities. In data quality programs modeled as graphs of data transformations, this capability is extremely important since it enables to analyze the provenance of all data records (in particular, exception records) undergoing a transformation process.

Debugging mechanisms, exception handling and data lineage are features that support the refinement of data quality programs. Refining data quality programs that handle large amounts of data with a certain degree of dirtiness is crucial, because automatic cleaning criteria cannot cover all data items. By detecting exceptions and their source, the user is able to refine quality rules in an adequate manner or manually correct erroneous records that cannot be automatically cleaned.

4 Tool categories

It is commonly accepted that data quality tools can be grouped according to the part of a data quality process they cover [29]. Data profiling and analysis assist in detecting data problems. Data transformation, data cleaning, duplicate elimination and data enhancement propose to solve the discovered or previously known data quality problems.

Data analysis: Activities that enclose the statistical evaluation, the logical study of data values and the application of data mining algorithms in order to define data patterns and rules to ensure that data does not violate the application domain constraints. The set of commercial and research tools that provide data analysis techniques is the following:

Commercial	dfPower, ETLQ, Migration Architect [30], Trillium, WizWhy [31]
Research	Potter's Wheel [32], Ken State University Tool [33]

Data profiling: Process of analyzing data sources with respect to the data quality domain[1] [34], to identify and prioritize data quality problems. Data profiling reports on the completeness of datasets and data records, organize data problems by importance, outputs the distribution of data quality problems in a dataset, and lists missing values in existing records [29]. The identification of data quality problems before starting a data cleaning project is crucial to ensure the delivery of accurate information. The following set of commercial and research tools implement data profiling techniques:

Commercial	dfPower, ETLQ, Migration Architect, Trillium, WizWhy
Research	Ken State University Tool

Data transformation: The set of operations (schema/data translation and integration, filtering and aggregation) that source data must undergo to appropriately fit a target schema. Data transformations require metadata, such as data schemas, instance-level data characteristics, and data mappings. The set of commercial and research tools that can be classified as data transformation tools is the following.

Commercial	Data Integrator, DataFusion, DataStage, dfPower, ETLQ, Hummingbird ETL, Firstlogic, Informatica ETL, SQL Server DTS [35], Sagent, SQL Server 2005 [36, 37], Sunopsis, Trillium
Research	Ajax, Arktos, Clio [38, 39, 40], FraQL [41, 42], Potter's Wheel, TranScm [43]

1. Data quality domain is an application or use of data that imposes a set of data quality rules i.e., specification of one or more data quality problems which should not exist in a data set.

Data cleaning: The act of detecting, removing and/or correcting dirty data[2]. Data cleaning aims not only at cleaning up the data but also to bring consistency to different sets of data that have been merged from separate databases. Sophisticated software applications are available to clean data using specific functions, rules and look-up tables. In the past, this task was done manually and therefore subject to human error. The following set of commercial and research tools implement data cleaning techniques:

Commercial	DataBlade [44], dfPower, ETLQ, ETI*DataCleanser [45], Firstlogic, NaDIS [46], QuickAddress Batch [47], Sagent, Trillium, WizRule [48]
Research	Ajax, Arktos, FraQL

Duplicate elimination: The process that identifies duplicate records (referring to the same real entity) and merges them into a single record. Duplicate elimination processes are costly and very time consuming. They usually require the following steps: *(i)* to standardize format discrepancies; *(ii)* to translate abbreviations or numeric codes; *(iii)* to perform exact and approximate matching rules and *(iv)* to consolidate duplicate records. The set of commercial and research tools that provide duplicate elimination techniques is presented below.

Commercial	Centrus Merge/Purge [49], ChoiceMaker [50, 51, 52, 53], DataBlade, DeDupe [54], dfPower, DoubleTake [55], ETLQ, ETI*DataCleanser, Firstlogic, Identity Search Server [56], MatchIT [57], Merge/Purge Plus [58], NaDIS, QuickAddress Batch, Sagent, SQL Server 2005, Trillium, WizSame [59]
Research	Ajax, Flamingo Project [60, 61, 62], FraQL, IntelliClean [63]

Data enrichment (also known as data enhancement): The process of using additional information from internal or external data sources to improve the quality of the input data that was incomplete, unspecific or outdated. Postal address enrichment, geocoding and demographic data additions are typical data enrichment procedures. The set of commercial and research data enrichment tools is listed below:

Commercial	DataStage, dfPower, ETLQ, Firstlogic, NaDIS, QuickAddress Batch, Sagent, Trillium
Research	Ajax

2. Data that is incorrect, outdated, redundant, inconsistent, incomplete, or incorrectly formatted.

5 Addressing data quality problems

At this point, we identified the data quality problems (in Section 2) and classified data quality tools under six categories (in Section 4). Now, in Table 3 we propose a correspondence between each category of data quality tools and the data quality problems it addresses. We did not consider profiling and analysis tools since we focus on the classes of tools aiming at correcting data anomalies.

Wrong categorical data can be eliminated using a lookup table that contains the set of valid categories. The erroneous data value can be detected if compared with all the entries of the lookup table. Domain-specific tools, like QuickAddress Batch for addresses, can solve this problem. Both outdated temporal data and inconsistent spatial data demand human interaction and cannot be automatically corrected. Name and structural conflicts require data schema transformations.

With respect to data instances problems, missing data can be detected and solved only if the »null« encoding (for instance, '9999') is appropriately identified. A table can be used to map the »null« encoding values and the corresponding meanings. Erroneous, misfielded data and embedded values cannot be automatically corrected. The documentation of some commercial tools refer that they include a spell checker that can be used to correct misspellings. Both data cleaning and enrichment tools (as FirstLogic or QuickAddress Batch) are able to eliminate ambiguous and non-standardized data. Approximate duplicates are detected by data cleaning and duplicate elimination tools that implement approximate matching algorithms [64]. Finally, contradicting records can be detected by data cleaning and duplicate elimination tools, but their consolidation must be assisted by the user.

	Schema Level					Instance Level								
	Not Supported by RDBMS					Single Rcord						Multiple Record		
Categories	Wrong Categorical Data	Outdated temporal Data	Inconsitent spatial Data	Name conflicts	Structural conflicts	Missing data (not null field)	Erroneous data	Misspellings	Embedded values	Misfielded values	Ambiguous data	Duplicate records	Contradicting records	Non-standard data
Data Transformation	X	X	X	Y	Y	X	X	X	X	X	X	X	X	X
Data Cleaning	Y	P	P	X	X	D	X	Y	P	X	Y	Y	P	Y
Duplicate Elimination	X	X	X	X	X	X	X	X	X	X	X	Y	P	X
Data Enrichment	Y	X	X	X	X	X	X	X	X	X	Y	X	X	Y

Tab. 3: Data quality problems addressed by each category of data quality tool. Y: addressed; X: not addressed; D: detected; P: partially addressed

6 Conclusions

Data quality tools transform data with problems into data of good quality for a certain application domain. This paper presented a classification of commercial and research data quality tools according to three different perspectives. First, we presented a taxonomy of data quality problems and highlighted those that are not addressed by current RDBMS technology. Second, we listed the generic functionalities that these tools must possess to fulfill their main objectives which are to extract data from data sources, to transform it in order to improve their quality and to load the resulting data in target sources. Then, data quality tools were divided into groups depending on the type of data quality task performed. We identified six different categories of tools: *(i)* data profiling; *(ii)* data analysis; *(iii)* data transformation; *(iv)* data cleaning; *(v)* duplicate elimination and *(vi)* data enrichment. The first two categories intend to discover data quality problems, while the last four supply techniques to correct them. These categories may overlap in the sense that some of the tasks (for instance, data cleaning) include the others (duplicate elimination and data enrichment). Finally, for each of data quality problems identified, we detailed which ones are addressed by the four categories of data quality tools that correct data problems.

Due to space restrictions, we did not describe further the methods used by data quality tools to address data quality problems. It would be interesting to make it clear how data cleaning tools solve misspellings or handle duplicate records, for instance. Furthermore, it would be helpful to establish a comparison of the various methods implemented by distinct tools to address the same problem. Another aspect that should be surveyed concerns data profiling and analysis tools. In [29], Olson provides a good insight of the functionalities these tools should provide. However, a classification of the existing tools against the list of important profiling and analysis functionalities remains to be done.

Previous surveys of data quality tools can be found in [65] and [66] or provided in some commercial reports [67], [68] and [69]. However, as far the authors are aware of, none of these intended to provide a framework of classification as we do here. Moreover, none of previous works covered commercial as well as research data quality tools.

References

[1] *Chaudhuri, S.; Dayal, U.:* An overview of data warehousing and olap technology. SIGMOD Record, 26(1):65-74, 1997.

[2] *Carreira, P.; Galhardas, H.:* Efficient development of data migration transformations. Demo paper. In: ACM SIGMOD Int'l Conf. on the Managment of Data, Paris, France, June 2004.

[3] *Kimball, R.; Reeves, L.; Ross, M.; Thornthwaite, W.:* The Datawarehouse lifecycle toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses. Wiley, 1998.

[4] *Galhardas, H.; Florescu, D.; Shasha, D.; Simon, E.; Saita, C. A.:* Declarative data cleaning: Language, model, and algorithms. In: Proc. of the Int'l Conf. on Very Large Data Bases (VLDB'01), Rome, Italy, September 2001.

[5] Oracle PL/SQL; *http://www.oracle.com/technology/tech/pl_sql.*

[6] *Kim, W.; Choi, B.-J.; Hong, E.-K.; Kim, S.-K.; Lee, D.:* A taxonomy of dirty data. Data Mining and Knowledge Discovery, 2559:19-34, December 2002.

[7] *Rahm, E.; Do, H.:* Data cleaning: Problems and current approaches. In: IEEE Techn. Bulletin on Data Engineering, December 2000.

[8] *Oliveira, P.; Rodrigues, F.; Henriques, P.; Galhardas, H.:* A taxonomy of data quality problems. International Workshop on Data Quality (in conjunction with CAISE'05), 2005.

[9] *Müller, H.; Freytag, J.-Chr.:* Problems, methods, and challenges in comprehensive data cleansing. Technical Report HUB-IB-164, Humboldt University Berlin, 2003.

[10] *Türker, C.; Gertz, M.:* Semantic integrity constraint support in sql:1999 and commercial object relational database management systems. In: The VLDB Journal 10, pp. 241-269, 2001.

[11] International Organization for Standardization (ISO) and American National Standards Institute (ANSI). ISO International Standard: Database Language SQL – part 2. September 1999.

[12] Sunopsis; *http://www.sunopsis.com.*

[13] ETLQ (SAS); *http://www.sas.com.*

[14] Bloor Research. ETLQ data quality report, 2004.

[15] Informatica ETL; *http://www.informatica.com.*

[16] Data Integrator (Business Objects); *http://www.businessobjects.com.*

[17] Sagent (Group 1 Software); *http://www. sagent.com.*

[18] Hummingbird ETL (Genio); *http://www.hum mingbird.com.*

[19] Firstlogic; *http://www.firstlogic.com.*

[20] Firstlogic introduces next-generation data quality platform, 2005.

[21] dfPower (DataFlux a SAS Company); *http://www.dataflux.com.*

[22] *Chandras, R.:* A smart response to bad data. 2005.

[23] Trillium; *http://www.trilliumsoft.com.*

[24] *Galhardas, H.:* Nettoyage de données: Modèle, langage déclaratif et algorithmes. PhD thesis, Université de Versailles Saint-Quentin-en-Yvelines, September 2001.

[25] DataFusion (Oblog); *http://www.oblog.pt.*

[26] DataStage (Ascential); *http://www.ascential.com.*

[27] *Simitsis, A.:* Modeling and managing {ETL} processes. In: VLDB PhD Workshop, Berlin, 2003.
[28] *Cui, Y:* Lineage Tracing in Data Warehouses. PhD thesis, Stanford University, 2001.
[29] *Olson, J.:* Data Quality – The Accuracy Dimension. Morgan Kaufmann Publishers, 2003.
[30] Migration Architect (Evoke Software); *http://www.evokesoft.com.*
[31] WhizWhy (WizSoft); *http://www.wizsoft.com.*
[32] *Raman, V.; Hellerstein, J.:* Potter's wheel: An interactive data cleaning system. VLDB, 27:381-390, 2001.
[33] *Maletic, J.; Marcus, A.:* Automated identification of errors in data sets. IQ2000, 2000.
[34] *Marshall, B.:* Data Quality and Data Profiling A Glossary of Terms; *http://www.agt.net/public/bmarshal/dataqual.htm.*
[35] SQL Server 2000 DTS (Microsoft); *http://www.microsoft.com.*
[36] SQL Server 2005 (Microsoft); *http://www. microsoft.com.*
[37] *Chaudhuri, S.; Ganjam, K.; Ganti, V.; Kapoor, R.; Narasayya, V.; Vassilakis, T.:* Data Cleaning in Microsoft SQL Server 2005. In: ACM SIGMOD/PODS Conference, 2005.
[38] *Miller, R. J.; Haas, L. M.; Hernández, M.; Ho, C. T. H.; Fagin, R.; Popa, L.:* The Clio Project: Managing Heterogeneity. SIGMOD Record, 1(30), March 2001.
[39] *Miller, R. J.:* Using Schematically Heterogeneous Structures. Proc. of ACM SIGMOD Int'l Conf. on the Managment of Data, 2(22):189-200, June 1998.
[40] *Miller, R. J.; Haas, L. M.; Hernández, M.:* Schema Mapping as Query Discovery. In: Proc. of the Int'l Conf. on Very Large Data Bases (VLDB'00), pp. 77-78, Cairo, Egypt, September 2000.
[41] *Sattler, K.-U.; Conrad, S.; Saake, G.:* Adding conflict resolution features to a query language for database federations. Int. Workshop on Engineering Federated Information Systems, EFIS'00, pp. 42-52, June 2000.
[42] *Sattler, K.-U.; Schallehn, E.:* A data preparation framework based on a multidatabase language. Proceedings of the International Database Engineering and Applications Symposium, pp. 219-228, 2000.
[43] *Milo, T.; Zhoar, S.:* Using schema matching to simplify heterogeneous data translation. In: Proc. of the Int'l Conf. on Very Large Data Bases (VLDB'98), New York, USA, August 1998.
[44] DataBlade (IBM); *http://www.informix.com.*
[45] ETI*Data Cleanser (ETI); *http://www.eti.com.*
[46] NaDIS (MSI); *http://www.msi.com.au.*
[47] QuickAddress Batch (QAS); *http://www. qas.com.*
[48] WhizRule (WizSoft); *http://www.wizsoft.com.*
[49] Centrus Merge/Purge Library (Group 1 Software); *http://www.centrus.com.*
[50] Choicemaker; *http://www.choicemaker.com.*
[51] *Buechi, M.; Borthwick, A.; Winkel, A.; Goldberg, A.:* Cluemaker: A language for approximate record matching. In: 8th International Conference on Data Quality, Boston, November 2003.
[52] *Borthwick, A.; Soffer, M.:* Business requirements of a record matching system. In: Ninth International Conference on Information Quality (MIT ICIQ), Cambridge, September 2004.
[53] *Borthwick, A.:* The ChoiceMaker 2 record matching system. White Paper, November 2004.
[54] DeDupe (HelpIT Systems); *http://www.ded upeit.com.*
[55] DoubleTake (Peoplesmith); *http://www. peoplesmith.com.*
[56] Identity Search Server (Identity Systems); *http://www.identitysystems.com.*
[57] MatchIT (HelpIT Systems); *http://www. helpit.com.*
[58] Merge/Purge Plus (Group 1 Software); *http://www.g1.com.*
[59] WhizSame (WizSoft); *http://www.wizsoft.com.*
[60] *Jin, L.; Li, C.; Mehrotra, S.:* Efficient record linkage in large data sets. In: Eighth International Conference on Database Systems for Advanced Applications, Kyoto, Japan, 2003.
[61] *Jin, L.; Koudas, N.; Li, C.:* NNH: Improving performance of nearest-neighbor searches using histograms. In: Extending Database Technology (EDBT), Crete, Greece, 2004.
[62] *Jin, L.; Koudas, N.; Li, C.; Tung, A.:* Indexing mixed types for approximate retrieval. In: VLDB, 2005.
[63] *Lee, M.; Ling, T.; Low, W.:* Intelliclean: A knowledge-based intelligent data cleaner. In: ACM SIGKDD, Boston, 2000.
[64] *Cohen, W. W.; Ravikumar, P.; Fienberg, S.:* A comparison of string metrics for matching names and records. KDD Workshop on Data Cleaning and Object Consolidation, 2003.
[65] Data Quality products review DMReview; *http://www.dmreview.com/resources/reviews3.cfm?Topic=230005.*
[66] Data Cleaning and Integration tools H. Galhardas; *http://web.tagus.ist.utl.pt/ helena.galhardas/cleaning.html.*
[67] Data Quality Tools METAspectrum Evaluation; *http://www.firstlogic.com/pdfs/METAspectrumDQT.pdf.*
[68] *Eckerson, W.; White, C.:* Evaluating ETL and data integration platforms. TDWI Report Series, 2003.
[69] Data Quality Tools for Data Warehousing A Small Sample Survey; *http://www.ctg.albany.edu/publications/reports/data_quality_tools].*
[70] *Beg, J.; Hussain, S.:* Data quality – a problem and an approach. White Paper, March 2003.

José Barateiro is finishing his MSc. thesis in Computer Engineering at Instituto Superior Técnico (IST), Technical University of Lisbon. His research interests include data quality (cleaning and transformation) and relational and XML databases (architecture, query processing and optimization).

Helena Galhardas is an assistant professor in the Department of Information Systems and Computer Engineering of the Instituto Superior Técnico (IST), Technical University of Lisbon and a researcher at INESC-ID. Her current research topics focus around data cleaning and transformation, data quality, relational and XML data processing and optimization. Helena received her PhD on Informatics from the University of Versailles in 2001, on »Data Cleaning: Model, Declarative Language and Algorithms«.

José Barateiro
Prof. Dr. Helena Galhardas
Instituto Superior Técnico (IST) Tagus Park
Av. Prof. Cavaco Silva
2780-990 Porto Salvo
Portugal
{jose.barateiro, helena.galhardas}@tagus.ist.utl.pt
http://www.tagus.ist.utl.pt