WIKIPEDIA

Data cleansing

Data cleansing or **data cleaning** is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.[1] Data cleansing may be performed interactively with data wrangling tools, or as batch processing through scripting.

After cleansing, a data set should be consistent with other similar data sets in the system. The inconsistencies detected or removed may have been originally caused by user entry errors, by corruption in transmission or storage, or by different data dictionary definitions of similar entities in different stores. Data cleaning differs from data validation in that validation almost invariably means data is rejected from the system at entry and is performed at the time of entry, rather than on batches of data.

The actual process of data cleansing may involve removing typographical errors or validating and correcting values against a known list of entities. The validation may be strict (such as rejecting any address that does not have a valid postal code) or fuzzy (such as correcting records that partially match existing, known records). Some data cleansing solutions will clean data by cross checking with a validated data set. A common data cleansing practice is data enhancement, where data is made more complete by adding related information. For example, appending addresses with any phone numbers related to that address. Data cleansing may also involve activities like, harmonization of data, and standardization of data. For example, harmonization of short codes (st, rd, etc.) to actual words (street, road, etcetera). Standardization of data is a means of changing a reference data set to a new standard, ex, use of standard codes.

Contents

Motivation

Administratively, incorrect or inconsistent data can lead to false conclusions and misdirected investments on both public and private scales. For instance, the government may want to analyze population census figures to decide which regions require further spending and investment on infrastructure and services. In this case, it will be important to have access to reliable data to avoid erroneous fiscal decisions. In the business world, incorrect data can be costly. Many companies use customer information databases that record data like contact information, addresses, and preferences. For instance, if the addresses are inconsistent, the company will suffer the cost of resending mail or even losing customers. The profession of forensic accounting and fraud investigating uses data cleansing in preparing its data and is typically done before data is sent to a data warehouse for further investigation.[2] There are packages available so you can cleanse/wash address data while you enter it into your system. This is normally done via an application programming interface (API).[3]

Data quality

High-quality data needs to pass a set of quality criteria. Those include:

- **Validity**: The degree to which the measures conform to defined business rules or constraints (see also Validity (statistics)). When modern database technology is used to design data-capture systems, validity is fairly easy to ensure: invalid data arises mainly in legacy contexts (where constraints were not implemented in software) or where inappropriate data-capture technology was used (e.g., spreadsheets, where it is very hard to limit what a user chooses to enter into a cell, if cell validation is not used). Data constraints fall into the following categories:
  - *Data-Type Constraints* – e.g., values in a particular column must be of a particular datatype, e.g., Boolean, numeric (integer or real), date, etc.
  - *Range Constraints:* typically, numbers or dates should fall within a certain range. That is, they have minimum and/or maximum permissible values.
  - *Mandatory Constraints:* Certain columns cannot be empty.
  - *Unique Constraints:* A field, or a combination of fields, must be unique across a dataset. For example, no two persons can have the same social security number.
  - *Set-Membership constraints*: The values for a column come from a set of discrete values or codes. For example, a person's gender may be Female, Male or Unknown (not recorded).
  - *Foreign-key constraints*: This is the more general case of set membership. The set of values in a column is defined in a column of another table that contains unique values. For example, in a US taxpayer database, the "state" column is required to belong to one of the US's defined states or territories: the set of permissible states/territories is recorded in a separate States table. The term foreign key is borrowed from relational database terminology.
  - Regular expression patterns: Occasionally, text fields will have to be validated this way. For example, phone numbers may be required to have the pattern (999) 999-9999.
  - Cross-field validation: Certain conditions that utilize multiple fields must hold. For example, in laboratory medicine, the sum of the components of the differential white blood cell count must be equal to 100 (since they are all percentages). In a hospital database, a patient's date of discharge from hospital cannot be earlier than the date of admission.
- **Accuracy**: The degree of conformity of a measure to a standard or a true value - see also Accuracy and precision. Accuracy is very hard to achieve through data-cleansing in the general case, because it requires accessing an external source of data that contains the true value: such "gold standard" data is often unavailable. Accuracy has been achieved in some cleansing contexts, notably customer contact data, by using external databases that match up zip codes to geographical locations (city and state), and also help verify that street addresses within these zip codes actually exist.
- **Completeness**: The degree to which all required measures are known. Incompleteness is almost impossible to fix with data cleansing methodology: one cannot infer facts that were not captured when the data in question was initially recorded. (In some contexts, e.g., interview data, it may be possible to fix incompleteness by going back to the original source of data, i,e., re-interviewing the subject, but even this does not guarantee success because of problems of recall - e.g., in an interview to gather data on food consumption, no one is likely to remember exactly what one ate six months ago. In the case of systems that insist certain columns should not be empty, one may work around the problem by designating a value that indicates "unknown" or "missing", but supplying of default values does not imply that the data has been made complete.
- **Consistency**: The degree to which a set of measures are equivalent in across systems (see also Consistency). Inconsistency occurs when two data items in the data set contradict each other: e.g., a customer is recorded in two different systems as having two different current addresses, and only one of them can be correct. Fixing

inconsistency is not always possible: it requires a variety of strategies - e.g., deciding which data were recorded more recently, which data source is likely to be most reliable (the latter knowledge may be specific to a given organization), or simply trying to find the truth by testing both data items (e.g., calling up the customer).
- **Uniformity**: The degree to which a set data measures are specified using the same units of measure in all systems (see also Unit of measure). In datasets pooled from different locales, weight may be recorded either in pounds or kilos, and must be converted to a single measure using an arithmetic transformation.

The term **integrity** encompasses accuracy, consistency and some aspects of validation (see also data integrity) but is rarely used by itself in data-cleansing contexts because it is insufficiently specific. (For example, "referential integrity" is a term used to refer to the enforcement of foreign-key constraints above.)

Process

- **Data auditing**: The data is audited with the use of statistical and database methods to detect anomalies and contradictions: this eventually gives an indication of the characteristics of the anomalies and their locations. Several commercial software packages will let you specify constraints of various kinds (using a grammar that conforms to that of a standard programming language, e.g., JavaScript or Visual Basic) and then generate code that checks the data for violation of these constraints. This process is referred to below in the bullets "workflow specification" and "workflow execution." For users who lack access to high-end cleansing software, Microcomputer database packages such as Microsoft Access or File Maker Pro will also let you perform such checks, on a constraint-by-constraint basis, interactively with little or no programming required in many cases.
- **Workflow specification**: The detection and removal of anomalies is performed by a sequence of operations on the data known as the workflow. It is specified after the process of auditing the data and is crucial in achieving the end product of high-quality data. In order to achieve a proper workflow, the causes of the anomalies and errors in the data have to be closely considered.
- **Workflow execution**: In this stage, the workflow is executed after its specification is complete and its correctness is verified. The implementation of the workflow should be efficient, even on large sets of data, which inevitably poses a trade-off because the execution of a data-cleansing operation can be computationally expensive.
- **Post-processing and controlling**: After executing the cleansing workflow, the results are inspected to verify correctness. Data that could not be corrected during execution of the workflow is manually corrected, if possible. The result is a new cycle in the data-cleansing process where the data is audited again to allow the specification of an additional workflow to further cleanse the data by automatic processing.

Good quality source data has to do with “Data Quality Culture” and must be initiated at the top of the organization. It is not just a matter of implementing strong validation checks on input screens, because almost no matter how strong these checks are, they can often still be circumvented by the users. There is a nine-step guide for organizations that wish to improve data quality:[4][5]

- Declare a high level commitment to a data quality culture
- Drive process reengineering at the executive level
- Spend money to improve the data entry environment
- Spend money to improve application integration
- Spend money to change how processes work
- Promote end-to-end team awareness
- Promote interdepartmental cooperation
- Publicly celebrate data quality excellence
- Continuously measure and improve data quality

Others include:

- **Parsing**: for the detection of syntax errors. A parser decides whether a string of data is acceptable within the allowed data specification. This is similar to the way a parser works with grammars and languages.
- **Data transformation**: Data transformation allows the mapping of the data from its given format into the format expected by the appropriate application. This includes value conversions or translation functions, as well as normalizing numeric values to conform to minimum and maximum values.
- **Duplicate elimination**: Duplicate detection requires an algorithm for determining whether data contains duplicate representations of the same entity. Usually, data is sorted by a key that would bring duplicate entries closer together for faster identification.

- **Statistical methods**: By analyzing the data using the values of mean, standard deviation, range, or clustering algorithms, it is possible for an expert to find values that are unexpected and thus erroneous. Although the correction of such data is difficult since the true value is not known, it can be resolved by setting the values to an average or other statistical value. Statistical methods can also be used to handle missing values which can be replaced by one or more plausible values, which are usually obtained by extensive data augmentation algorithms.

System

The essential job of this system is to find a suitable balance between fixing dirty data and maintaining the data as close as possible to the original data from the source production system. This is a challenge for the Extract, transform, load architect. The system should offer an architecture that can cleanse data, record quality events and measure/control quality of data in the data warehouse. A good start is to perform a thorough data profiling analysis that will help define to the required complexity of the data cleansing system and also give an idea of the current data quality in the source system(s).

Tools

There are lots of data cleansing tools like Trifacta, OpenRefine, Paxata, Alteryx, and others. It's also common to use libraries like Pandas (software) for Python (programming language), or Dplyr (http://dplyr.tidyverse.org) for R (programming language).

One example of a data cleansing for distributed systems under Apache Spark is called Optimus (https://hioptimus.com), an OpenSource framework for laptop or cluster allowing pre-processing, cleansing, and exploratory data analysis. It includes several data wrangling tools.

Quality screens

Part of the data cleansing system is a set of diagnostic filters known as quality screens. They each implement a test in the data flow that, if it fails records an error in the Error Event Schema. Quality screens are divided into three categories:

- Column screens. Testing the individual column, e.g. for unexpected values like NULL values; non-numeric values that should be numeric; out of range values; etc.
- Structure screens. These are used to test for the integrity of different relationships between columns (typically foreign/primary keys) in the same or different tables. They are also used for testing that a group of columns is valid according to some structural definition to which it should adhere.
- Business rule screens. The most complex of the three tests. They test to see if data, maybe across multiple tables, follow specific business rules. An example could be, that if a customer is marked as a certain type of customer, the business rules that define this kind of customer should be adhered to.

When a quality screen records an error, it can either stop the dataflow process, send the faulty data somewhere else than the target system or tag the data. The latter option is considered the best solution because the first option requires, that someone has to manually deal with the issue each time it occurs and the second implies that data are missing from the target system (integrity) and it is often unclear what should happen to these data.

Criticism of existing tools and processes

Most data cleansing tools have limitations in usability:

- **Project costs**: costs typically in the hundreds of thousands of dollars
- **Time**: mastering large-scale data-cleansing software is time-consuming
- **Security**: cross-validation requires sharing information, giving an application access across systems, including sensitive legacy systems

Error event schema

The Error Event schema holds records of all error events thrown by the quality screens. It consists of an Error Event Fact table with foreign keys to three dimension tables that represent date (when), batch job (where) and screen (who produced error). It also holds information about exactly when the error occurred and the severity of the error. In addition there is an Error Event Detail Fact table with a foreign key to the main table that contains detailed information about in which table, record and field the error occurred and the error condition.

Challenges and problems

- **Error correction and loss of information**: The most challenging problem within data cleansing remains the correction of values to remove duplicates and invalid entries. In many cases, the available information on such anomalies is limited and insufficient to determine the necessary transformations or corrections, leaving the deletion of such entries as a primary solution. The deletion of data, though, leads to loss of information; this loss can be particularly costly if there is a large amount of deleted data.
- **Maintenance of cleansed data**: Data cleansing is an expensive and time-consuming process. So after having performed data cleansing and achieving a data collection free of errors, one would want to avoid the re-cleansing of data in its entirety after some values in data collection change. The process should only be repeated on values that have changed; this means that a cleansing lineage would need to be kept, which would require efficient data collection and management techniques.
- **Data cleansing in virtually integrated environments**: In virtually integrated sources like IBM's DiscoveryLink, the cleansing of data has to be performed every time the data is accessed, which considerably increases the response time and lowers efficiency.
- **Data-cleansing framework**: In many cases, it will not be possible to derive a complete data-cleansing graph to guide the process in advance. This makes data cleansing an iterative process involving significant exploration and interaction, which may require a framework in the form of a collection of methods for error detection and elimination in addition to data auditing. This can be integrated with other data-processing stages like integration and maintenance.

See also

- Data editing
- Data mining
- Record linkage
- Single customer view

References

1. Wu, S. (2013), "A review on coarse warranty data and analysis" (http://www.sciencedirect.com/science/article/pii/S0951832013000100), *Reliability Engineering and System*, **114**: 1–11, doi:10.1016/j.ress.2012.12.021 (https://doi.org/10.1016%2Fj.ress.2012.12.021)
2. Nigrini, M. Forensic Analytics: Methods and Techniques for Forensic Accounting Investigations, Wiley. 2011
3. The importance of data cleansing user-generated-content (https://spotlessdata.com/blog/importance-data-cleansing-user-generated-content)
4. Kimball, R., Ross, M., Thornthwaite, W., Mundy, J., Becker, B. *The Data Warehouse Lifecycle Toolkit*, Wiley Publishing, Inc., 2008. ISBN 978-0-470-14977-5
5. Olson, J. E. *Data Quality: The Accuracy Dimension", Morgan Kaufmann, 2002. ISBN 1-55860-891-5*

Sources

- Han, J., Kamber, M. *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2001. ISBN 1-55860-489-8.
- Kimball, R., Caserta, J. *The Data Warehouse ETL Toolkit*, Wiley and Sons, 2004. ISBN 0-7645-6757-8.