Data Quality

Knowledge objectives

1. Explain what data quality is
2. Exemplify the causes of data quality problems
3. Classify the data conflicts depending on:
   a) They affect only the schema or also the instances
   b) They can happen in a single data source or need many
4. Explain the different kinds of data quality rules in relational terms
5. Name the steps of object identification
6. Name and exemplify three kinds of space reduction
7. Name and exemplify four kinds of object identification functions

Understanding Objectives

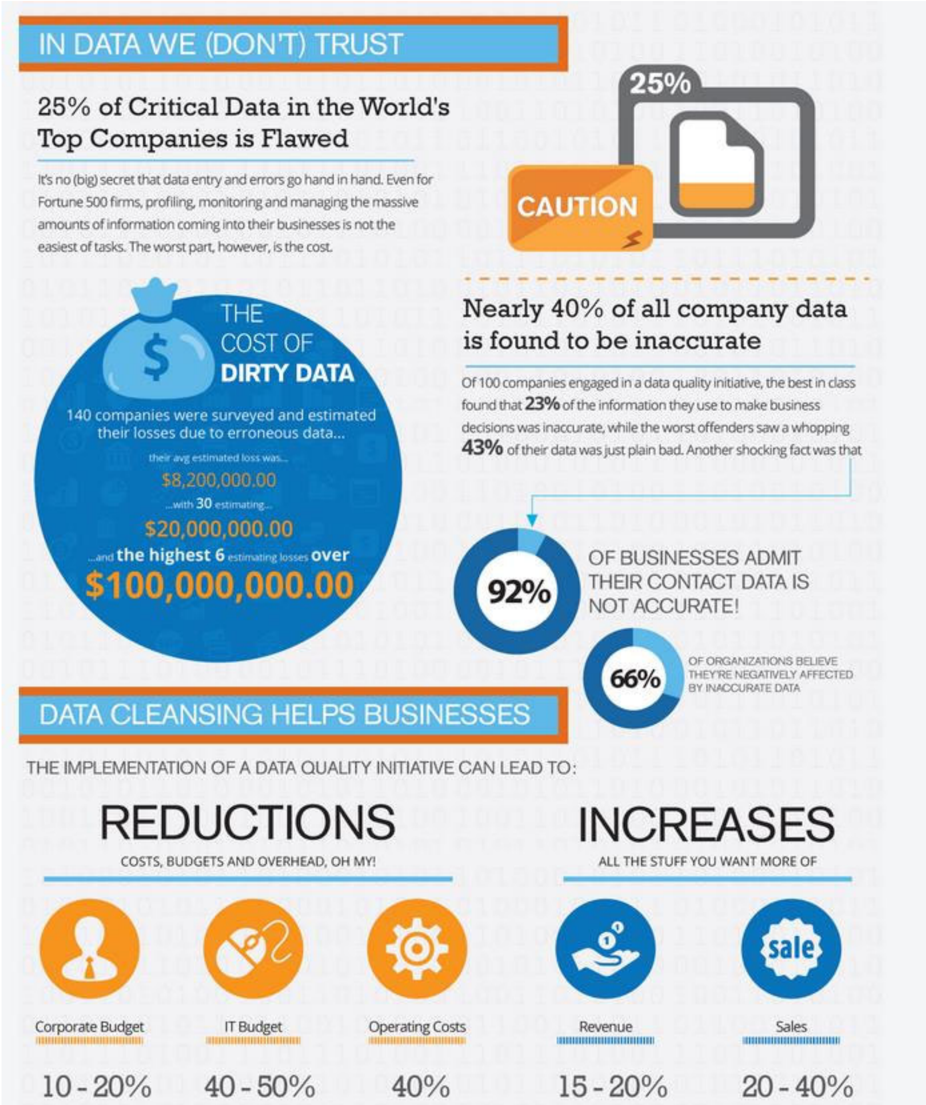
1. Calculate the value of the most prominent data quality measures (i.e., Completeness, Accuracy, Consistency, and Timeliness)

Application objectives

1. Identify redundant rules
2. Identify inconsistent rules

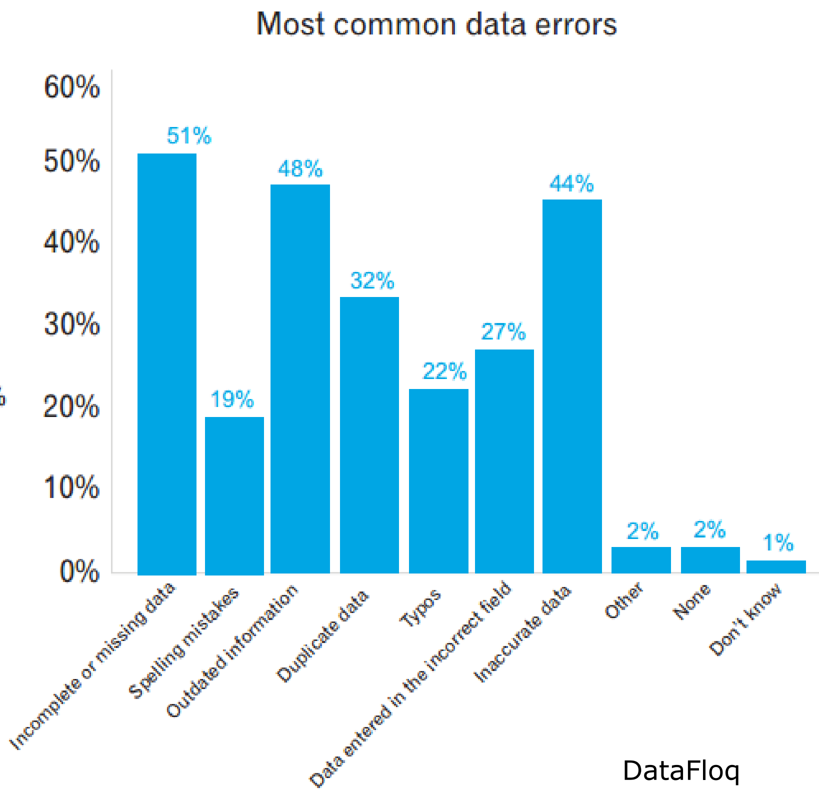
MOTIVATION AND DEFINITION

Motivation (I)



Halo Business Intelligence

Motivation (II)



DataFloq

Fitness for use

"A user can only assess the level of quality of a set of data for a particular task to be executed in a **specific context**, according to a set of criteria, thus determining whether or not these data can be used **for that purpose**."

William Edwards Deming

Sources of problems

- Data ingestion
  - Initial data conversion
  - Manual data entry
  - Batch feeds
  - Real-time interfaces
  - System consolidations
- Data processing
  - Process automation
  - Data processing
  - Data cleansing
  - Data purging
- Inaction
  - Changes not captured
  - System upgrades
  - New data uses
  - Loss of expertise

Data conflicts

- **Instances**
  - **Single source**

    (Data entry errors)

    - Misspelling
    - Redundancy
    - Contradictions
    - …
  - **Multi-source**

    (Overlapping, contradicting and inconsistent data)

    - Duplicates
    - Inconsistent aggregation
    - Inconsistent timing
    - …
- **Schema**
  - **Single source**

    (Lack of integrity constraints, poor schema design)

    - Uniqueness
    - Referential integrity
    - …
  - **Multi-source**

    (Heterogeneous data models and schema designs)

    - Naming conflicts
    - Structural conflicts
    - …

QUALITY MEASURES

Clusters of quality dimensions

- **Completeness**, …
  - …pertinence, and relevance refer to the capability of representing all and only the relevant aspects of the reality of interest
- **Accuracy**, …
  - … correctness, validity, and precisión focus on the adherence to a given reality of interest (includes syntactic, semantic as well as **temporal** accuracy)
- **Consistency**, …
  - … cohesion, and coherence refer to the capability of the information to comply without contradictions to all properties of the reality of interest, as specified in terms of integrity constraints, data edits, business rules, and other formalisms
- **Redundancy**, …
  - … minimality, compactness, and conciseness refer to the capability of representing the aspects of the reality of interest with the minimal use of information resources
- **Readability**, …
  - … comprenhensibility, clarity, and simplicity refer to easy understanding and fruition of information by users
- **Accessibility**, …
  - … and availability are related to the ability of the user to access information from his or her culture, physical status/functions, and technologies available
- **Usefulness**, …
  - … related to the advantage the user gains from the use of information
- **Trust**, …
  - … including believability, reliability, and reputation, catching how much information derives from an unauthoritative source (encompasses also issues related to securty)

C. Batini and M. Scannapieco

Completeness

"The degree to which a given collection of data describes the corresponding set of real-world objects."

- Missing entities (OWA)
- Missing values (CWA)

$$Q_{Cm}(A_i) = |R(NotNull(A_i))|/|R|$$

$$Q_{Cm}(R) = |R(\wedge_{A_i \in R} NotNull(A_i))|/|R|$$

Accuracy

"The extent to which data are correct, reliable and certified error free."

- Free of typing errors
- Appropriate precision

$$e_A = |v_A - v_{RealWorld}|$$
$$Q_A(A_i) = |R(e_{A_i} \leq \varepsilon)| / |R|$$

$$Q_A(R) = |R(\wedge_{A_i \in R} e_{A_i} \leq \varepsilon)| / |R|$$

Timeliness (Freshness)

"How old the stored value of an attribute is with regard to the current value in the real world."

$$age(v) = now\text{-}TransactionTime$$

$$f_u(v) = \text{updates per time unit}$$

$$Q_T(v) = (1+f_u(v)\cdot age(v))^{-1}$$

$$Q_T(A_i) = Avg_{v \in R[Ai]} Q_T(v)$$

$$Q_T(R) = Avg_{Ai \in R} Q_T(A_i)$$

Consistency

"The degree of violation of semantic rules defined over a set of data items."

- Integrity constraints
  - Entity
  - Domain
  - Referential
  - User-defined
- Coincidence of copies
  - Temporal
  - Permanent

$$Q_{Cn}(R,B) = |R(\wedge_{rule \in B} rule(A_1,..,A_n))| / |R|$$

Trade-offs between dimensions

- Timeliness ⇔ Accuracy, Completeness and Consistency
- Completeness ⇔ Accuracy and Consistency

QUALITY RULES

Integrity constraints in RDBMS

- Intra-Attribute
  - Domain
    - Outliers
  - Not null
- Intra-Tuple
  - Checks
- Intra-Relation
  - Primary keys
    - Unique
  - Temporal (Triggers)
    - State-dependent
- Inter-Relation
  - Foreign key
  - Assertions (Triggers)

Dependencies

- Multivalued dependencies
  - Functional dependencies

    $t1.X=t2.X \Rightarrow t1.Y=t2.Y$

    - Key dependencies
- Inclusion dependencies

    $R.X \subseteq S.Y$

    - Foreign key dependencies

Problems in the rules

- Contradictory rules generate empty results

  CHECK (a<10 AND a>20)

- Redundant rules slow down performance
  1. CHECK (a>20)
  2. CHECK (a>10)

- Imperfections

Logic properties of rules

- *Schema-satisfiability*: A schema is satisfiable if there is at least one consistent DB state containing tuples (i.e. each and every constraint is fulfilled)
- *Liveliness*: A table/view is lively if there is at least one consistent DB state, so that the table/view contains tuples
- *Redundancy*: An integrity constraint is redundant if the consistency of the DB does not depend on it (none of the tuples it tries to avoid can never exist)
- *State-reachability*: A given set of tuples is reachable if there is at least one consistent DB state containing those tuples (and maybe others)
- *Query containment* (*subsumption*): A query Q1 is contained in another Q2, if the set of tuples of Q1 is always contained in the set of tuples of Q2 in every consistent DB state

Example of Schema-satisfiability

```
CREATE TABLE employees (
    id CHAR(9) PRIMARY KEY,
    dpt VARCHAR(4) NOT NULL
    );

CREATE TABLE departments (
    id VARCHAR(4) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    basicSalary INT                 ,
    CONSTRAINT ckMinSalary CHECK (basicSalary>2000),
    CONSTRAINT ckMaxSalary CHECK (basicSalary<1000)
    );
```

Example of Liveliness

```
CREATE TABLE departments (
    id VARCHAR(4) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    basicSalary INT NOT NULL,
    CONSTRAINT ckMinSalary CHECK (basicSalary>2000));

CREATE TABLE employees (
    id CHAR(9) PRIMARY KEY,
    dpt VARCHAR(4)            REFERENCES departments (id));

CREATE VIEW unassigned AS (
    SELECT *
    FROM employees e
    WHERE NOT EXISTS (        SELECT *
                              FROM departments d
                              WHERE d.id=e.dpt));
```

Example of Redundancy

```
CREATE TABLE departments (
    id VARCHAR(4) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    basicSalary INT NOT NULL,
    CONSTRAINT ckMinSalary CHECK (basicSalary>2000),
    CONSTRAINT ckDeptName CHECK (id<>'CS')
    );

CREATE TABLE employees (
    id CHAR(9) PRIMARY KEY,
    dpt VARCHAR(4) NOT NULL REFERENCES departments (ID),
    CONSTRAINT ckEmpName CHECK (dpt<>'CS')
    );
```

Example of State-reachability

```
CREATE TABLE departments (
    id VARCHAR(4) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    basicSalary INT NOT NULL,
    CONSTRAINT ckMinSalary CHECK (basicSalary>2000));

CREATE TABLE employees (
    id CHAR(9) PRIMARY KEY,
    dpt VARCHAR(4) NOT NULL REFERENCES departments (ID));
```

Employees(id, dpt); Departments(id, name, basicSalary)

id	dpt
1	CS
2	MK

id	name	basicSalary
CS	Compu…	10000

Example of Query Containment (I)

A:

```
SELECT *
FROM departments
WHERE basicSalary>5000;
```

B:

```
SELECT *
FROM departments
WHERE basicSalary>6000;
```

Example of Query Containment (II)

A:

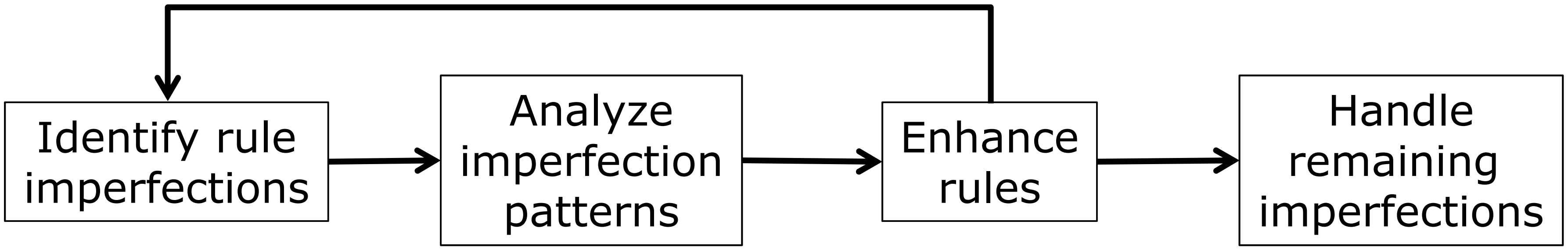
```
SELECT *
FROM departments
WHERE basicSalary>5000;
```

B:

```
SELECT *
FROM departments
WHERE basicSalary>5000 AND name>'M';
```

Fine-tuning imperfections

- Improvement management
  - False positives
  - False negatives
- Improvement monitoring



Quality improvement

- Imputation
  - Minimum change
  - Keep marginal and joint frequency distribution of values in the different attributes
- Acquisition of new data
  - Error localization
  - Source identification
    - Source trustworthiness
    - Cost optimization
  - Entity resolution
    - Standardization/Normalization
  - Record merging

OBJECT IDENTIFICATION

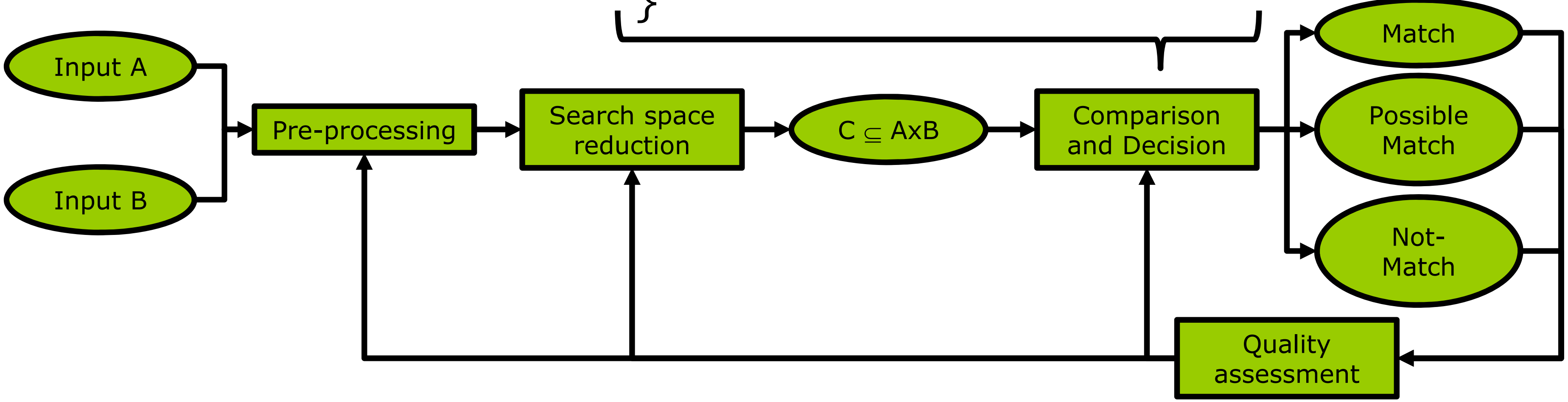
Object identification example

Agency	Identifier	Name	Type of activity	Address	City
Agency 1	CNCBTB765SDV	Meat production of John Ngombo	Retail of bovine and ovine meats	35 Niagara Street	New York
Agency 2	0111232223	John Ngombo canned meat production	Grocer's shop, beverages	9 Rome Street	Albany
Agency 3	CNDBTB76SSDV	Meat production in New York state of John Ngombo	Butcher	4, Garibaldi Square	Long Island

C. Batini and M. Scannapieco

Process for Object identification

```
O:= ∅;
while (I<> ∅ ) do {
    pick up r∈I; I:=I-r;
    if (∃s∈O so that s ≈ r) {
        O:=O-s; I:=I∪{r ∧ s};
    } else {
        O:=O∪{r};
    }
}
```



C. Batini and M. Scannapieco

Search space reduction

- Blocking
  - Choose a key to partition the sets
- Sorted neighbourhood
  - Define a window to probe only records in it
- Pruning
  - Find some rules (heuristics)

Comparison/Similarity function

- Distance-based
  - Decide that two records represent the same entity if their distance is below a threshold (after removing affixes)
    - Hamming distance
    - Edit distance (insert, delete and replace)
      - Smith-Waterman algorithm (assign weights)
    - n-Grams comparison
    - Jaro algorithm (insert, delete and transpose)
  - Compare lists of items
    - Jaccard distance
    - Token Frequency-Inverse Document Frequency (TF-IDF)
  - May use phonetics
    - Soundex code
- Rule-based
  - Depending on the concrete attributes
- Probabilistic techniques
  - Based on conditional probabilities
- Classification-based
  - Provide positive and negative training pairs of instances

CLOSING

Summary

- Quality measures:
  - Accuracy
  - Completeness
  - Consistency
  - Timeliness
- Quality rules
  - Dependencies
  - Relational constraints
  - Consistency checks
  - Redundancy check
  - Imputation
- Object identification

Bibliography

- A. Maydanchik. *Data Quality Assessment*. Technics Publications, 2007
- J. Bleiholder and F. Naumann. Data Fusion. ACM Computing Surveys, 41(1), 2008
- C. Batini, et al. *Methodologies for data quality assessment and improvement*. ACM Compututing Surveys, 41(3), 2009
- H. Garcia-Molina et al. *Database Systems*. Prentice Hall, 2009
- C. Batini and M. Scannapieco. Data and Information Quality. Springer, 2016