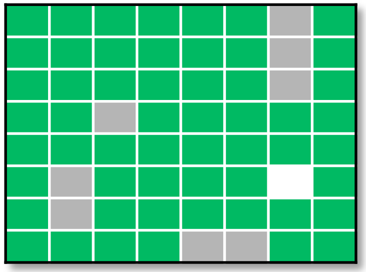


Incomplete Databases:
Missing Records and Missing Values

Werner Nutt and Simon Razniewski and Gil Vegliach

Introduction

- Data Quality research investigates how good data is

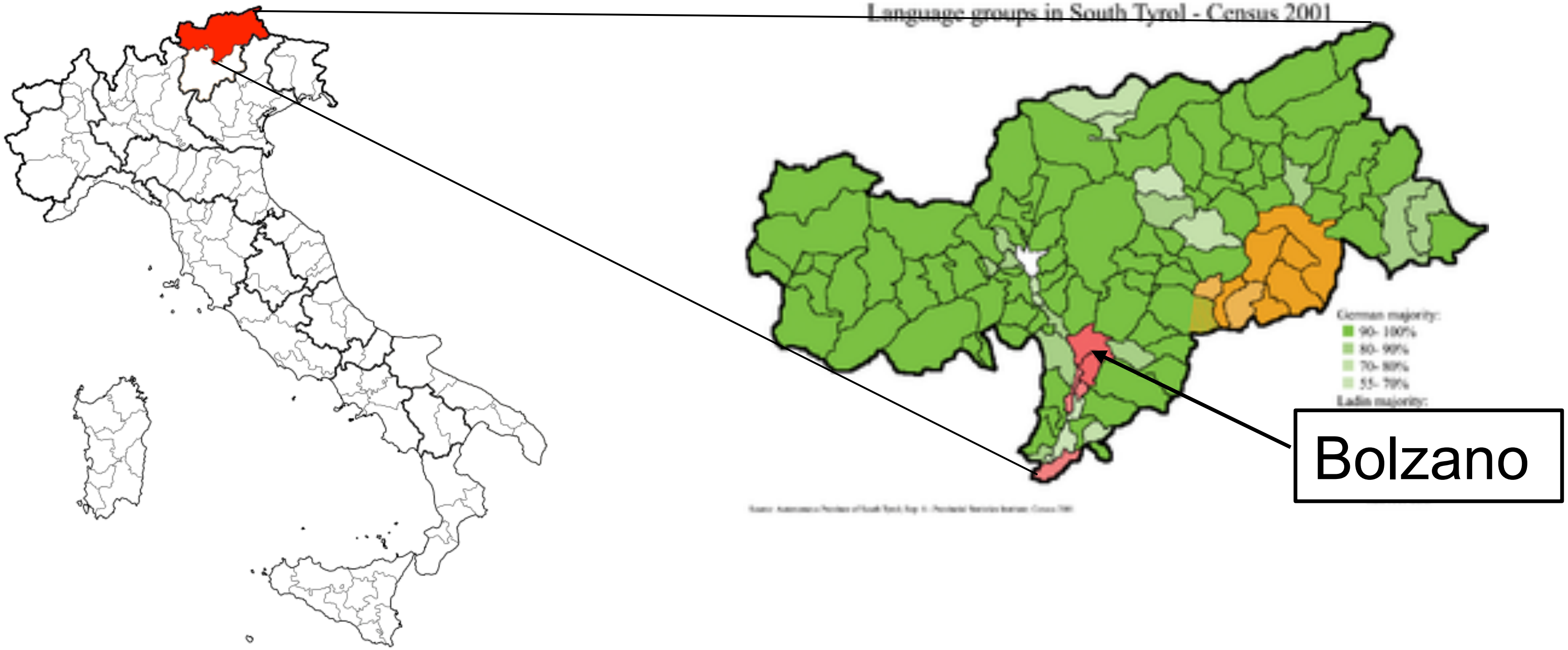
- Dimensions of Data Quality are:
  - Correctness
  - Timeliness
  - Completeness

Completeness

- Query answering over incomplete data: extensively studied
  - Codd: Null values (1975)
  - Imielinski/Lipski: Representation systems 1984

- Query completeness: Little attention
  - Previous work by us: Only on missing records

Bolzano is in the province of South Tyrol

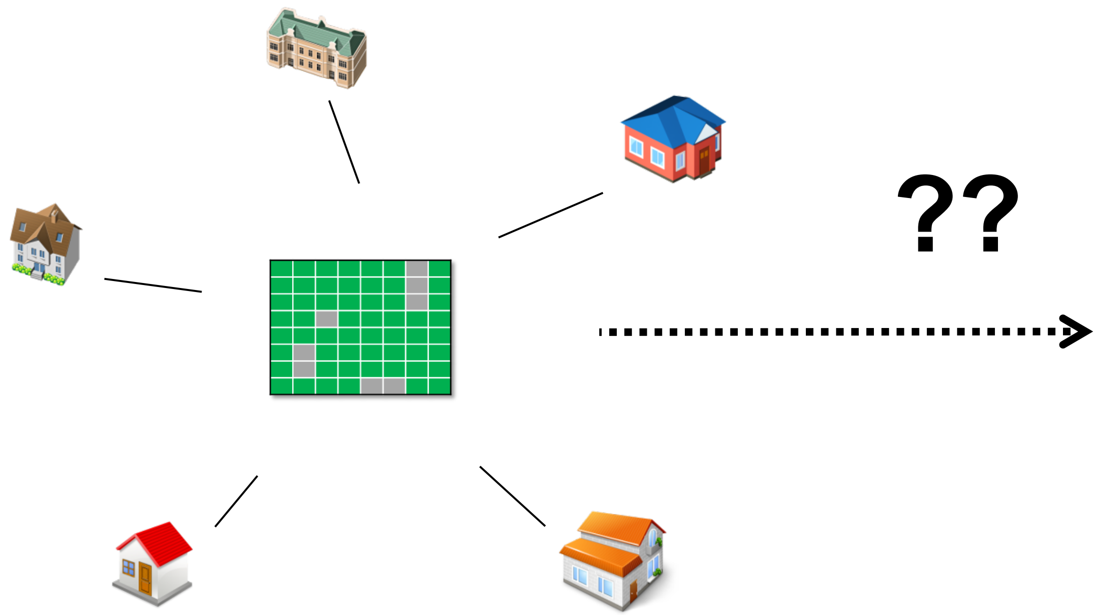


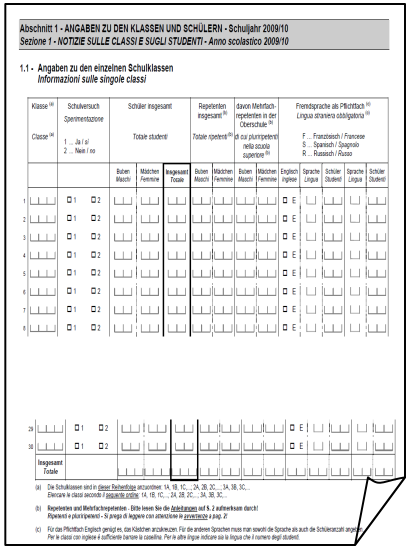
- Autonomous, trilingual province in the north of Italy

Example scenario:
School data management in South Tyrol

Central school database

Statistical reports





Abschnitt 1 - ANGABEN ZU DEN KLASSEN UND SCHÜLERN - Schuljahr 2009/10
Sezione 1 - NOTIZIE SULLE CLASSI E SUGLI STUDENTI - Anno scolastico 2009/10

1.1 - Angaben zu den einzelnen Schulklassen
Informazioni sulle singole classi

	Klasse (a) *Classe (a)*	Schulversuch *Sperimentazione* 1 ... Ja / *si* 2 ... Nein / *no*	Schüler insgesamt *Totale studenti*			Repetenten insgesamt (b) *Totale ripetenti (b)*		davon Mehrfachrepetenten in der Oberschule (b) *di cui pluriripetenti nella scuola superiore (b)*		Fremdsprache als Pflichtfach (c) *Lingua straniera obbligatoria (c)* F ... Französisch / *Francese* S ... Spanisch / *Spagnolo* R ... Russisch / *Russo*				
			Buben *Maschi*	Mädchen *Femmine*	**Insgesamt** ***Totale***	Buben *Maschi*	Mädchen *Femmine*	Buben *Maschi*	Mädchen *Femmine*	Englisch *Inglese*	Sprache *Lingua*	Schüler *Studenti*	Sprache *Lingua*	Schüler *Studenti*
1		□ 1 □ 2								□ E				
2		□ 1 □ 2								□ E				
3		□ 1 □ 2								□ E				
4		□ 1 □ 2								□ E				
5		□ 1 □ 2								□ E				
6		□ 1 □ 2								□ E				
7		□ 1 □ 2								□ E				
8		□ 1 □ 2								□ E				
29		□ 1 □ 2								□ E				
30		□ 1 □ 2								□ E				
	Insgesamt ***Totale***													

(a) Die Schulklassen sind in dieser Reihenfolge anzuordnen: 1A, 1B, 1C,...; 2A, 2B, 2C,...; 3A, 3B, 3C,...
Elencare le classi secondo il seguente ordine: 1A, 1B, 1C,...; 2A, 2B, 2C,...; 3A, 3B, 3C,...

(b) **Repetenten und Mehrfachrepetenten - Bitte lesen Sie die Anleitungen auf S. 2 aufmerksam durch!**
Ripetenti e pluriripetenti - Si prega di leggere con attenzione le avvertenze a pag. 2!

(c) Für das Pflichtfach Englisch genügt es, das Kästchen anzukreuzen. Für die anderen Sprachen muss man sowohl die Sprache als auch die Schüleranzahl angeben.
Per le classi con inglese è sufficiente barrare la casellina. Per le altre lingue indicare sia la lingua che il numero degli studenti.

Notoriously incomplete

Completeness important

Example: Final grades

- Vocational schools enter final grades, many others don‘t

- Query: How many pupils have grade ‘A‘ in Math?

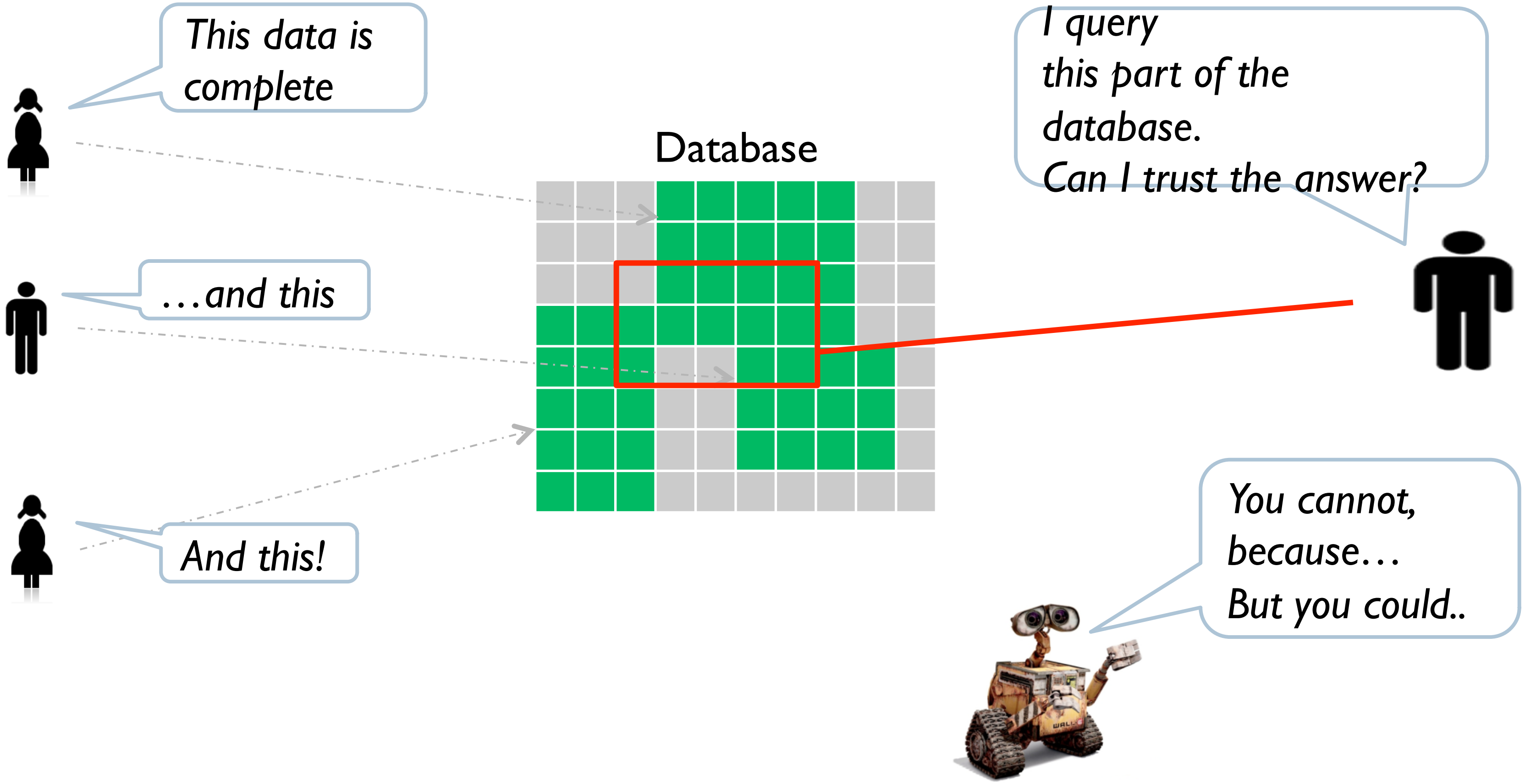
- Answer: 15.300

- Can we trust this? No!
  - Pupils from high schools could be missing in the result

Example: Final grades (2)

- Vocational schools enter final grades, many others don‘t
- Query: How many pupils at vocational schools have grade ‘A‘ in Math?
- Answer: 7.200
- Can we trust this? Yes!
  - All grades from vocational schools are in the database

General problem



This data is complete

…and this

And this!

Database

I query this part of the database. Can I trust the answer?

You cannot, because… But you could..

Existing theory for

- SQL select-project-join queries

  SELECT…
  FROM …
  WHERE…

- Bag and set semantics

  “DISTINCT“

- Aggregate queries

  “COUNT, SUM, MAX, MIN“

Schema

result(name, subject, result)

pupil(name, schoolName, schoolType)

Incomplete database (Motro 1989)

Incompleteness needs a complete reference

Incomplete databases are pairs of

an ideal database D^i and

an available database D^a

$$\boldsymbol{D} = (D^i, D^a)$$

such that

D^a is a subset of D^i

Incomplete database example

D^i = { result(Giulia, Math, A) pupil(Giulia, Da Vinci, primary)
result(Paul, Math, A) pupil(Paul, Hofer, vocational)
result(Paolo, Sports, B) }

D^a ={ result(Giulia, Math, A)
result(Paul, Math, A) }

Query completeness

Query Q

“The set (bag) of answers to Q is complete“

Notation: $\mathrm{Compl}^s(Q)$ $(\mathrm{Compl}^b(Q))$

Semantics (for set):

$$(D^i, D^a) \models \mathrm{Compl}^s(Q) \quad \text{iff} \quad Q^s(D^i) = Q^s(D^a)$$

Query completeness: Example

D^i = { result(Giulia, Math, A) pupil(Giulia, Da Vinci, primary)
result(Paul, Math, A) pupil(Paul, Hofer, vocational)
result(Paolo, Sports, B) }

D^a ={ result(Giulia, Math, A)
result(Paul, Math, A) }

Query: *All grades in Math* $Q_{math}(x)$:-result(n,Math,x)

$$Q_{math}(D^i) = \{(A),(A)\}$$
$$Q_{math}(D^a) = \{(A)\}$$

→ Q_{math} is set-complete, but not bag-complete

Table completeness

The available database contains all grades from vocational schools

$\text{result}^{i}(n,s,g), \text{pupil}^{i}(n,sn,\text{‘vocat‘}) \rightarrow \text{result}^{a}(n,s,g)$

Every result of a pupil from a vocational school according to the ideal db is also in the available db

This is a full tuple-generating dependency (TGD)

The example again..

Our database contains

- All pupils
- All grades from vocational schools

$\left.\right\}$ TC Statements $\mathcal{C}$

Query

"How many pupils at vocational schools have grade A in Math?

$\left.\right\}$ QC Statement $\mathrm{Compl(Q)}$

TC-QC entailment

$$\mathcal{C} \models \mathrm{Compl(Q)} \ ?$$

Reasoning

Query: *Pupils at vocational schools with A in Math*

Q_{pupils}(n):-result(n, Math, ‘A‘), pupil(n, sn, ‘voc‘)

1. Construct a generic query answer for Q_{pupils} over D^i

n‘ in Q(D^i)

2. See which facts must be in D^i

$result^i$(n‘, ‘Math‘, ‘A‘), $pupil^i$ (n‘, sn‘, ‘vocat‘) in D^i

Reasoning (2)

result^i(n‘, ‘Math‘, ‘A‘), pupil^i (n‘, sn‘, ‘vocat‘) in D^i

3. Use table completeness to derive facts in D^a

All results from vocational schools there:

result^i(n, s, g), pupil^i (n, sn, ‘vocat‘) → result^a (n, s, g)

All pupils there:

pupil^i (n, sn, st) → pupil^a (n, sn, st)

→ result^a(n‘, ‘Math‘, ‘A‘) in D^a pupil^a (n‘, sn‘, ‘vocat‘) in D^a

Reasoning (3)

result^a(n‘, ‘Math‘, ‘A‘), pupil^a (n‘, sn‘, ‘vocat‘) in D^a

4. Query the available database

$Q(D^a) = \{n'\}$ → n' in $Q(D^a)$

Conclusion: Query is complete given the table completeness

Reasoning: Summary

1. Construct a generic query answer for Q over D^i
2. See which facts must be in D^i
3. Use table completeness to derive facts in D^a
4. Query D^a
5. If the generic query answer is returned,
   the query is complete

Reasoning: Complexity

- From PTIME to Π^P_2 for queries and statements corresponding to SQL SELECT-PROJECT-JOIN (conjunctive queries with arithmetic comparisons)

Adding nulls

Problem: Ambiguity

result(John, Math, null)

- no result?
- result unknown?
- unknown which of the two?

Theory needs extensions

Incomplete databases:

- D^a need not be a subset of D^i, but contain less information (tuplewise)
- Nulls in both databases

D^i	D^a
result(John, Math, A)	result(John, Math, null)
result(Mary, Sports, null)	-

Theory needs extensions (2)

TC statements need projections

For each student, the subjects are known
where he/she is enrolled – but not necessarily the grades

$$\mathrm{result}^{i}(n,s,g_1) \rightarrow \exists g_2\colon \mathrm{result}^{a}(n,s,g_2)$$

TGDs with existentially quantified variables

Extensions of incomplete databases create hassle

$D^i = \{ R(a,b) \}$

$D^a = \{ R(a,b), R(a,null) \}$

$Q(y) \text{ :- } R(x,y)$

$Q(D^i) = \{ b \}, \; Q(D^a) = \{ b, null \}$

→ db tables are complete, but query is not complete!

Way out 1: Disallow duplicates

D^i = { R(a,b) }

D^a = { R(a,b), R(a,null) }

→ Require that each fact in D^a stands for a different fact in D^i

Motivation: Scenarios where keys are never unknown

Problem: Not always feasible (e.g. in data integration)

Way out 2: Forget redundant query results

$Q(D^a) = \{ (a,b), (a,\mathrm{null}) \}$

(a,null) is less informative than (a,b)

→ Forget such less-informative results

Problem: Nulls may carry information (that no value exists)

Nulls create hassle even when values are complete

Every grade in D^i appears (at least once) in D^a

Set-query: *All grades that students in class 4b received*

Available query answer: {A, B, C, D, E, null}
Ideal query answer: {A, B, C, D, E} or {A, B, C, D, E, null}?

In both cases, D^a contains all information from D^i

→Having all values is not sufficient

Preliminary results/conjectures

- Reasoning for bag-queries reduces to query containment under combined bag/set-semantics
  - Bag-containment: decidability unknown!

- Reasoning for set-queries reduces to query containment under set semantics over dbs with nulls
  - Decidable, but exact complexities unknown

Conclusion

- Existing theory for reasoning about query completeness
  - Considers only missing records
- Missing values (nulls) practically important
  - Challenge: Ambiguity of standard SQL-nulls
- What we also work on
  - Implementation of reasoning using logic-programming
  - Extraction and verification of completeness over business processes

Questions?

Other possible approach: Make different nulls explicit

Introduce three null values

- $null_{not\_applicable}$
- $null_{unknown}$
- $null_{unknown\_whether\_applicable}$

Only $null_{not\_applicable}$ may occur in D^i

If we are complete for all values, $null_{unknown}$ and $null_{unknown\_whether\_applicable}$ may be forgotten in D^a

Pure theory?

Ambiguity can be resolved by boolean guards

result‘			
name	…	graded	grade
John	…	yes	B
Mary	…	yes	null
Alice	…	no	null
Bob	…	null	null

Unknown

Not applicable

Unknown whether applicable

Allows to count how many pupils received a grade (2-3)

Boolean guards possibly already used where needed