Theory and Practice of Data Cleaning

Relational Data

From Syntax to Schema & Semantics

- **Regular expressions**
  - Define ***patterns*** (***syntax***) for matching and extracting data
  - Check conformance (e.g., ISO date format YYYY-MM-DD)
    - … otherwise: ***bring into canonical form***
- **OpenRefine**
  - Profile and clean data, one column at a time
  - Powerful similarity-based clustering
    - … ***bring into canonical form***

- **What about complex issues, spanning multiple columns?**
  - What about ***logical errors*** ?
  - How to deal with data quality at the ***schema*** and ***semantic*** level?

… after OpenRefine, "dirty data" can still make it into our database tables …

PERSON

Id	Name	DOB	Age	Sex	Phone	Zip	Email
43	Doe, Joe	1970-02-27	56	M	(999)-999-999	94102	
43	Jane Dunbar	1.1.1990	26	W	NULL	61820	jdunbar@foobar.com
27	Joe Doe	2/30/70	46	F	+1-530-777-1234	D-6951	joe.doe@gargle.edu

ADDRESS

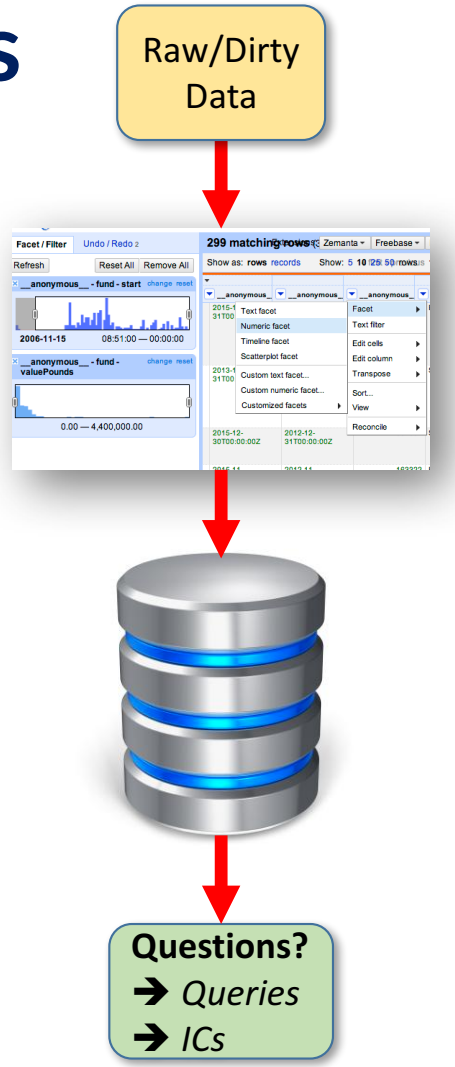
ZIP	City	State
94102	San Franzisco	CA
61821	Champagne	IL
D-6951	Obrigheim	Deutschland

- Errors and IC Violations:
  - Different representations & formats
  - Duplicates
  - Incompleteness
  - Incorrect values (typos, domain, …)
  - **Uniqueness (primary key) violation**
  - **Contradictions**
  - **Referential Integrity (FK → PK)**
    - **PERSON.ZIP → ADDRESS.ZIP**
  - …

From Syntax to Schema & Semantics

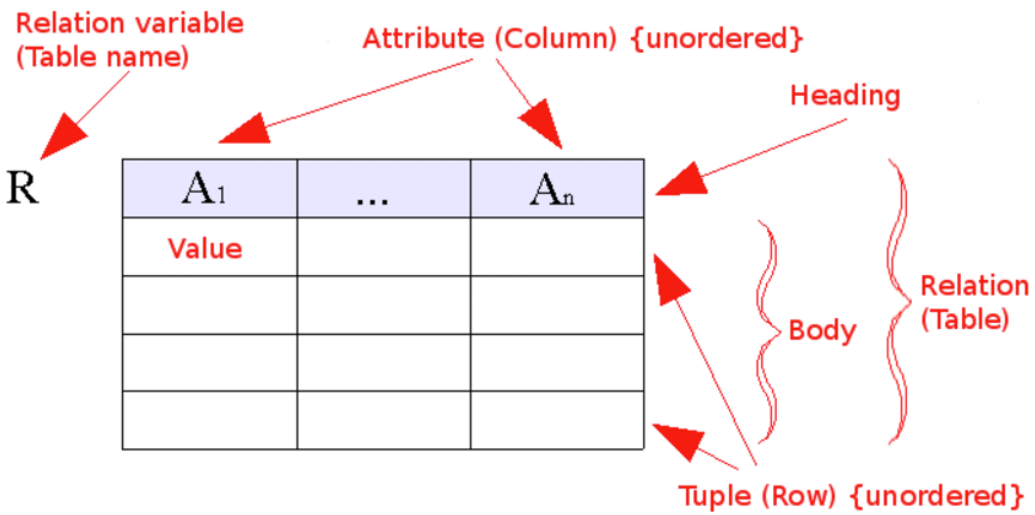
- After ***pattern-based cleaning***
  - regular expressions, OpenRefine, …
- … load data into a database system!
- … and then exploit database technology:
  - ***queries*** & ***integrity constraints***!

- **Relational Databases**
  - *Logic-based* approach first: **Datalog**
    - Facts, rules, queries, integrity constraints
    - Rich body of research; theory & practice!
  - *Relational data everywhere*: **SQL**



Relational Model

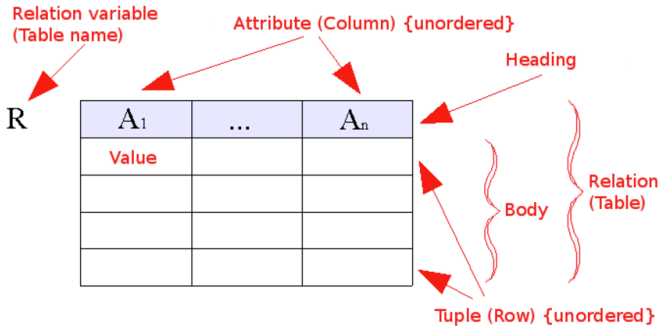
- Data in ***relations*** (*tables*) with …
  - … **rows** (*tuples*)
  - **… columns** (*attributes*)
  - … header (*schema*)
  - … body (*instances*)

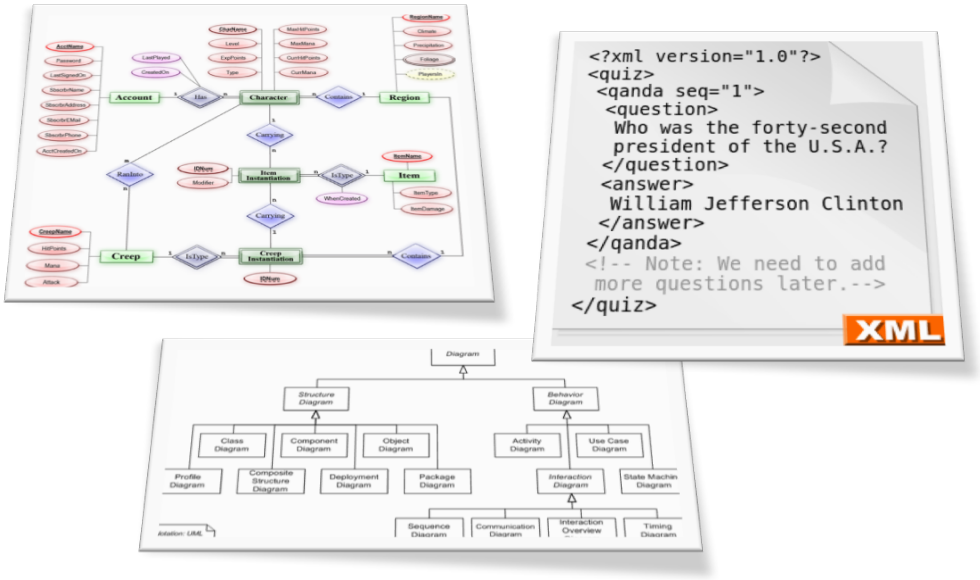


- Pioneered by Edgar F. Codd in the 1970s
- Some more **Terminology**:
  - **Relational Model**
  - Relational **Schema**
  - Relational **Database (Instance)**
  - **RDBMS**
    - Relational Database Management System

Relational Model …

- … is closely related to:
  - ***Predicate Logic***
  - ***Entity-Relationship*** (ER) model
- … can represent other data:
  - *Object-oriented / object-relational* model
  - *XML* data, *graph* data,
  - …
- … has powerful ***query languages*:**
  - ***First-order predicate logic (FOL, RC)***
  - ***Datalog***
  - ***Relational Algebra (RA)***
  - ***Structured Query Language (SQL)***
- … for checking of ***integrity constraints***





English language Wikipedia, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=2278339

So many query languages, so little time: 4-in-1

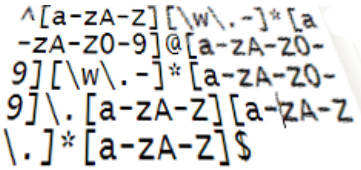
- SQL — SELECT ... FROM ... WHERE ...
- Relational Algebra (RA) — $\sigma, \pi, \bowtie, \delta, \cup, \setminus$
- Relational Calculus (RC) — $\forall x F,\ \exists x F,\ F \wedge G,\ F \vee G, \neg F$
- **Datalog** — $\approx$ RC + Recursion

EXAMPLE: Given relations `employee(Emp, Salary, DeptNo)` and `dept(DeptNo, Mgr)`, find all (employee, manager) pairs:

- SQL:
```
SELECT Emp, Mgr
FROM employee, dept
WHERE employee.DeptNo = dept.DeptNo
```
- RA: $\pi_{\texttt{Emp,Mgr}}(\texttt{employee} \bowtie \texttt{dept})$
- RC: $F(\texttt{Emp,Mgr}) =$ $\exists \texttt{Salary, DeptNo} : (\texttt{employee(Emp, Salary, DeptNo)} \wedge \texttt{dept(DeptNo,Mgr)})$
- **Datalog**: `boss(Emp,Mgr)` $\leftarrow$ `employee(Emp, Salary, DeptNo), dept(DeptNo,Mgr)`

Summary

- Syntax
  - **Regular expressions** define *patterns* that can be used to *match*, *extract*, and *transform* data, i.e., deal with syntactic variations
  - **OpenRefine**: open source tool for data wrangling



- **Schema & Semantics**
  - Using database technologies for **querying** and profiling; **integrity constraint** (IC) checking; and repair
  - **Relational Model**
  - Querying relational data**:** **Datalog** and **SQL**

- Synthesis
  - **Workflow** automation (ETL, scripts)
  - **Provenance** (data lineage and processing history)
  - **YesWorkflow**: modeling scripts as workflows, provenance