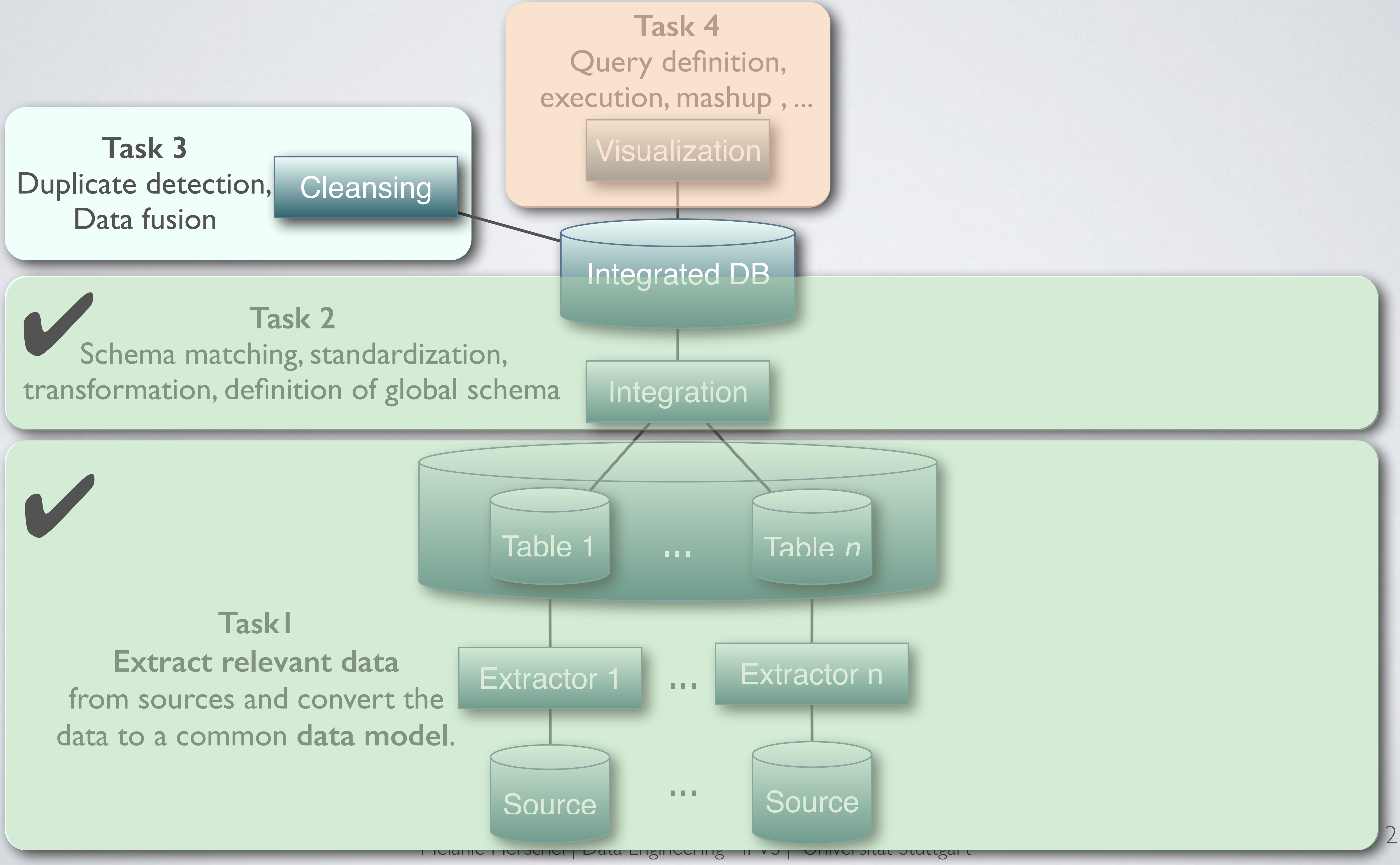
INFORMATION INTEGRATION

Practicals winter term 2016/17

Ralf Diestelkämper | Data Engineering - IPVS | Universität Stuttgart

STATUS



Task 4
Query definition, execution, mashup , ...

Visualization

Task 3
Duplicate detection, Data fusion

Cleansing

Integrated DB

✔ Task 2
Schema matching, standardization, transformation, definition of global schema

Integration

✔ Task1
Extract relevant data from sources and convert the data to a common **data model**.

Table 1 ... Table *n*

Extractor 1 ... Extractor n

Source ... Source

DUPLICATE DETECTION

- **Duplicate Detection** means finding several representations of the same real world object (aka Entity Resolution, Reference reconciliation, record linkage, …)

- **Problem 1: Representations are not identical.**
  - Fuzzy duplicates
  - Solution: Similarity Measures
  - Value and tuple comparisons

- **Problem 2: The data set is large.**
  - Quadratic effort: Each pair needs to be compared.
  - Solution: Algorithms
  - E.g. avoiding comparisons by partitioning or Sorted Neighborhood Method

DUPLICATE DETECTION

- Measures for computing **attribute value similarity**
  - **Token-based** similarity measures
    - Token: words (separated by whitespaces)
    - n-grams(e.g. 3-grams for "token": {"tok", "oke", "ken"}
  - **Jaccard** Similarity
    - |{common tokens}| / |{all tokens}|
  - **TF-IDF**
    - Term frequency (TF), Inverse document frequency (IDF)
    - *TFIDF: log (tf+1) x log idf*
    - Intuition: Low weight for frequent tokens
    - Extension: Soft-TFIDF
  - …

DUPLICATE DETECTION

- Measures for computing **tuple similarity** (based on value similarity)
  - **Weighted sum** of value similarities
  - Example:
    *sim(a,b) = 0.5 * simTitle(a,b) + 0.3 * simYear(a,b) + 0.2 * simGenre(a,b)*
  - **Rules**
  - Example:
    If simTitle(a,b) > 0.8 and simYear(a,b) > 0.6 => Duplicate
  - **Machine Learning**:
    Learn classificator for decision duplicate/no duplicate
  - …

DUPLICATE DETECTION

- Problem: **Too many comparisons**
  - 10.000 tuples => 49.995.000 comparisons
  - $(n^2 - n) / 2$
  - Each comparison is **expensive** (complex similarity measures)

- Idea: Avoid comparisons
  - For instance, using **blocking**
  - Sorted Neighborhood Method (see lecture)
  - ...

DUPLICATE DETECTION

ID	Title	Year	Genre
17	Mask of Zorro	1998	Adventure
18	Addams Family	1991	Comedy
25	Rush Hour	1998	Comedy
31	Matrix	1999	Sci-Fi
52	Return of Dschafar	1994	Children
113	Adams Family	1991	Comedie
207	Return of Djaffar	1995	Children

- Blocking:
  - Build blocks / partitions of tuples based on equal attribute values.
  - Only compare tuples within a same block.
- Example: Tuples with same year are placed in same block (see above) --> 2 comparison in the example.

SORTED NEIGHBORHOOD METHOD

ID	Title	Year	Genre
17	Mask of Zorro	1998	Adventure
18	Addams Family	1991	Comedy
25	Rush Hour	1998	Comedy
31	Matrix	1999	Sci-Fi
52	Return of Dschafar	1994	Children
113	Adams Family	1991	Comedie
207	Return of Djaffar	1995	Children

Create key

1.

ID	Key
17	MSKAD98
18	DDMCO91
25	RSHCO98
31	MTRSC99
52	RTRCH94
113	DMSCO91
207	RTRCH95

2.

Sort

ID	Key
18	DDMCO91
113	DMSCO91
17	MSKAD98
31	MTRSC99
25	RSHCO98
52	RTRCH94
207	RTRCH95

Merge

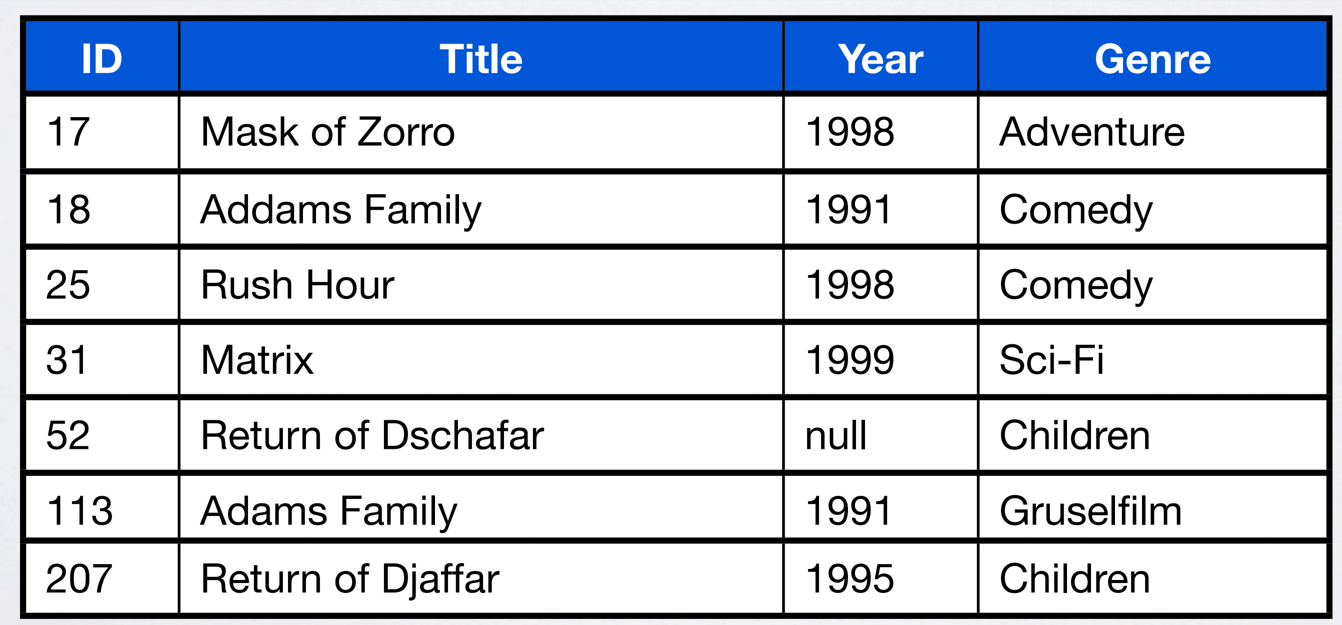
3.

ID	Key
18	DDMCO91
113	DMSCO91
17	MSKAD98
31	MTRSC99
25	RSHCO98
52	RTRCH94
207	RTRCH95

classify(18,113) → duplicates

classify(52,207) → duplicates

DATA FUSION



ID	Title	Year	Genre
17	Mask of Zorro	1998	Adventure
18	Addams Family	1991	Comedy
25	Rush Hour	1998	Comedy
31	Matrix	1999	Sci-Fi
52	Return of Dschafar	null	Children
113	Adams Family	1991	Gruselfilm
207	Return of Djaffar	1995	Children

- Given duplicates, we need to create a unique representation for each entity.
- In a sense, some kind of user defined aggregation function could be used for fusion of (different) attribute values.
- Problem: what aggregation function? I.e., what values to keep? Combination / concatenation of values to not loose any data?
- Unfortunately, no general solution, highly domain and application dependent.

TASK 3: PRESENTATIONS

- Your presentation should at least include:
  - An estimate of how many tuples need to be processed for duplicate detection in your application.
  - Examples of duplicates (highlighting the challenges)
  - A definition of a similarity measure or other duplicate classifier that applies in your application (explain why it is a "good" choice).
  - A description on how you implement duplicate detection efficiently.
  - A description of your data fusion strategy.

TASK 3: PRESENTATIONS

- Create slides to present your solution
- Put your slides into ILIAS
  - The day before your presentation at 3 p.m the latest
  - As PDF, PPT(X), Keynote or Open Office file)
- Language: English
- Date: 11.01.2017
- Duration: 8-10 min
- Presence is mandatory