Data Profiling – A look back and a look forward

EDBT 2021
Felix Naumann
(Almost) Hands-Off Information Integration for the Life Sciences

- Entity Resolution
- Inclusion dependency discovery across sources
- Key and inclusion dependency discovery within sources
- Parsing

Felix Naumann
Data Profiling
EDBT 2021

[CIDR'05]
Data Profiling for Data Engineering

Data Profiling

- Schema Engineering
  - Knowledge Discovery
- Data Cleaning
  - Data Analytics
- Feature Engineering
  - AI Systems
- Data Integration
  - Query Optimization

Preparation

Metadata

Application

See here for Open Research Questions
Use Case: Query Optimization

- 58 optimization opportunities
  - Using unique column combinations (UCCs), functional dependencies (FDs), order dependencies (ODs), and inclusion dependencies (INDs)

```
SELECT A, B, AVG(C) FROM R GROUP BY A, B
```

A → B

```
SELECT A, B, AVG(C) FROM R GROUP BY A
```

Open Research Question

How to introduce dependency-based optimization to DBMS?
Use Case: Data Cleansing

1. Discover approximate/relaxed dependencies
2. Verify their genuineness
3. Detect violating records/values
4. Correct the values

<table>
<thead>
<tr>
<th>Name</th>
<th>ID</th>
<th>LVL</th>
<th>ZIP</th>
<th>ST</th>
<th>SAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>ID1</td>
<td>5</td>
<td>10001</td>
<td>NM</td>
<td>90k</td>
</tr>
<tr>
<td>Bob</td>
<td>ID2</td>
<td>6</td>
<td>87101</td>
<td>NM</td>
<td>80k</td>
</tr>
<tr>
<td>Chris</td>
<td>ID3</td>
<td>4</td>
<td>10001</td>
<td>NY</td>
<td>80k</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Contact</th>
<th>Address 1</th>
<th>Address 2</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
<th>Phone</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Felix Naumann</td>
<td>Active</td>
<td>Verified</td>
<td>ZUKOVICH ALAN</td>
<td>MICHAEL</td>
<td>3551 FORESTDALE</td>
<td>BURLINGTON</td>
<td>NC</td>
<td>32725</td>
<td>331 270 6278 V</td>
</tr>
</tbody>
</table>

106,185 rows
Open Research Question
How to support visual data profiling?
Data profiling refers to the activity of creating small but informative summaries of a database.

Ted Johnson, Encyclopedia of Database Systems

Classifying Data Profiling Tasks

Single-column tasks

- Cardinalities
- Uniqueness
  - Key discovery
- Patterns and data types
- Distributions
- Domain Classification
- ...

Multi-column tasks

- Uniqueness (UCCs)
  - Key discovery
- Inclusion dependencies (INDs)
  - Foreign key discovery
- Functional dependencies (FDs)
- Order dependencies (ODs)
- Denial constraints (DCs)
- ...

Ted Johnson, Encyclopedia of Database Systems
Scalable Profiling

- **Scalability in number of rows**

- **Scalability in number of columns**
  - “Normal” table with 100 columns:
    \[2^{100} - 1 = 1,267,650,600,228,229,401,496,703,205,375\]
    \[= 1.3\text{ nonillion column combinations (the power set)}\]

- **Large solution space**: e.g. exponential number of FDs

Open Research Question
Can we predict problem and solution size for given data?
1. Basic statistics
2. Uniques and keys
3. Functional dependencies
4. Inclusion dependencies and foreign keys
5. New directions
Cardinalities, Distributions, and Patterns

Patterns and types
- String vs. number
- String vs. number vs. date
- Categorical vs. continuous
- SQL data types
- Domains
- Regular expressions
- Semantic data types

Cardinalities
- num-rows
- value length
- null values
- distinct
- uniqueness

Value distributions
- histograms
- constancy
- quartiles
- soundex
- first digit

Open Research Question
Can we find useful regular expressions for data columns?
Benford Law Frequency ("first digit law")

- Distribution of first digits in naturally occurring numbers:
  - $P(d) = \log_{10}(d + 1) - \log_{10}(d) = \log_{10}(1 + \frac{1}{d})$
  - Holds if $\log(x)$ is uniformly distributed
  - Street addresses of the first 342 persons listed in *American Men of Science*
  - 335 NIST physical constants

Since 1971 “American Men and Women of Science”
Agenda

1. Basic statistics
2. Uniques and keys
3. Functional dependencies
4. Inclusion dependencies and foreign keys
5. New directions
Unique column combination (UCC)
No pair of records has same value combination when projected to those columns

A minimal unique: voter_reg_num, zip_code, race_code

A maximal non-unique: voter_reg_num, status_cd, voter_status_desc, reason_cd, voter_status_reason_desc, absent_ind, name_prefix_cd, name_suffix_cd, half_code, street_dir, street_type_cd, street_suffix_cd, unit_designator, unit_num, state_cd, mail_addr2, mail_addr3, mail_addr4, mail_state, area_cd, phone_num, full_phone_number, drivers_lic, race_code, race_desc, ethnic_code, ethnic_desc, party_cd, party_desc, sex_code, sex, birth_place, precinct_abbrv, precinct_desc, municipality_abbrv, municipality_desc, ward_abbrv, ward_desc, cong_dist_abbrv, cong_dist_desc, super_court_abbrv, super_court_desc, judic_dist_abbrv, judic_dist_desc, nc_senate_abbrv, nc_senate_desc, nc_house_abbrv, nc_house_desc, county_commiss_abbrv, county_commiss_desc, township_abbrv, township_desc, school_dist_abbrv, school_dist_desc, fire_dist_abbrv, fire_dist_desc, water_dist_abbrv, water_dist_desc, sewer_dist_abbrv, sewer_dist_desc, sanit_dist_abbrv, sanit_dist_desc, rescue_dist_abbrv, rescue_dist_desc, munic_dist_abbrv, munic_dist_desc, dist_1_abbrv, dist_1_desc, dist_2_abbrv, dist_2_desc, confidential_ind, age, vtd_abbrv, vtd_desc
Candidate Set Growth for UCCs

<table>
<thead>
<tr>
<th>Number of attributes: m</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of levels: k</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

Total:
1 3 7 15 31 63 127 255 511 1,023 2,047 4,095 8,191 16,383 32,767
Pruning Supersets

minimal unique
unique

ABCDE

ABCE  ABCD  ABDE  ACDE  BCDE

ABCDE

ABD  ACD  ADE  ACE  BCD  BCE  BDE  CDE

ABE

AB  AC  AD

ABD  ACD  ADE  ACE  BCD  BCE  BDE  CDE

ABCDE
Pruning Subsets

Apriori-style enumeration and pruning
DUCC – Detecting Unique Column Combinations

Open Research Question
What are effective distribution strategies for data profiling?
Agenda

1. Basic statistics
2. Uniques and keys
3. **Functional dependencies**
4. Inclusion dependencies and foreign keys
5. New directions
Discovering Functional Dependencies

“X → A”
When two tuples have same value in attribute set X, they must have same values in attribute A.

E.g., ZIP → City

Naïve discovery approach
For each column combination X
For each A∈X
For each pair of tuples (t1,t2)
If t1[X\A] = t2[X\A] and t1[A] ≠ t2[A]: Break
Candidate Set Growth for FDs

<table>
<thead>
<tr>
<th>Total</th>
<th>0</th>
<th>2</th>
<th>9</th>
<th>28</th>
<th>75</th>
<th>186</th>
<th>1,016</th>
<th>2,295</th>
<th>5,110</th>
<th>11,253</th>
<th>24,564</th>
<th>53,235</th>
<th>114,674</th>
<th>245,745</th>
</tr>
</thead>
<tbody>
<tr>
<td>L:</td>
<td>0</td>
<td>14</td>
<td>15</td>
<td>0</td>
<td>14</td>
<td>15</td>
<td>13</td>
<td>182</td>
<td>1,365</td>
<td>30,030</td>
<td>51,480</td>
<td>90,460</td>
<td>150,070</td>
<td>245,745</td>
</tr>
</tbody>
</table>

FDs

Number of attributes: m

Number of levels: k
Model in Lattice – Edges Represent FDs

Apriori-style enumeration and pruning

Minimal FD

FD

Maximal non-FD

Non-FD

Minimal FDs
- A → B
- A → C
- A → D
- A → E
- BCDE → A

Felix Naumann
Data Profiling
EDBT 2021
Row-based Algorithms

<table>
<thead>
<tr>
<th>Name</th>
<th>Surname</th>
<th>Postcode</th>
<th>City</th>
<th>Mayor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thomas</td>
<td>Miller</td>
<td>14482</td>
<td>Potsdam</td>
<td>Jakobs</td>
</tr>
<tr>
<td>Sarah</td>
<td>Miller</td>
<td>14482</td>
<td>Potsdam</td>
<td>Jakobs</td>
</tr>
<tr>
<td>Peter</td>
<td>Smith</td>
<td>60329</td>
<td>Frankfurt</td>
<td>Feldmann</td>
</tr>
<tr>
<td>Jasmine</td>
<td>Cone</td>
<td>01069</td>
<td>Dresden</td>
<td>Orosz</td>
</tr>
<tr>
<td>Thomas</td>
<td>Cone</td>
<td>14482</td>
<td>Potsdam</td>
<td>Jakobs</td>
</tr>
<tr>
<td>Mike</td>
<td>Moore</td>
<td>60329</td>
<td>Frankfurt</td>
<td>Feldmann</td>
</tr>
</tbody>
</table>

- **Surname, Postcode, City, Mayor** → **Name**
- **Name, Postcode, City, Mayor** → **Surname**
- **Surname** → **Name, Postcode, City, Mayor**
HyFD: Hybrid FD Discovery

Low sampling efficiency
(new observations per comparison)

Low validation efficiency
(new FDs per validation)

column-efficient

row-efficient
## Functional Dependencies: State of the Art

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>iris</td>
<td>5</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>1.1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>balance-scale</td>
<td>5</td>
<td>625</td>
<td>7</td>
<td>1</td>
<td>1.2</td>
<td>0.2</td>
<td>0.2</td>
<td>0.3</td>
<td>0.3</td>
<td>0.3</td>
<td>0.3</td>
<td>0.1</td>
</tr>
<tr>
<td>chess</td>
<td>7</td>
<td>28,056</td>
<td>519</td>
<td>1</td>
<td>2.9</td>
<td>1.1</td>
<td>3.8</td>
<td>10.0</td>
<td>174.6</td>
<td>164.2</td>
<td>125.5</td>
<td>0.2</td>
</tr>
<tr>
<td>abalone</td>
<td>9</td>
<td>4,177</td>
<td>187</td>
<td>137</td>
<td>2.1</td>
<td>0.6</td>
<td>1.8</td>
<td>1.1</td>
<td>3.0</td>
<td>2.9</td>
<td>3.8</td>
<td>0.2</td>
</tr>
<tr>
<td>nursery</td>
<td>9</td>
<td>12,960</td>
<td>1,024</td>
<td>1</td>
<td>4.1</td>
<td>1.8</td>
<td>7.1</td>
<td>0.9</td>
<td>121.2</td>
<td>118.9</td>
<td>46.8</td>
<td>0.5</td>
</tr>
<tr>
<td>breast-cancer</td>
<td>11</td>
<td>699</td>
<td>20</td>
<td>46</td>
<td>2.3</td>
<td>0.6</td>
<td>2.2</td>
<td>0.8</td>
<td>1.1</td>
<td>1.1</td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>bridges</td>
<td>13</td>
<td>108</td>
<td>6</td>
<td>142</td>
<td>2.2</td>
<td>0.6</td>
<td>4.2</td>
<td>0.9</td>
<td>0.5</td>
<td>0.6</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>echocardiogram</td>
<td>13</td>
<td>132</td>
<td>6</td>
<td>527</td>
<td>1.6</td>
<td>0.4</td>
<td>69.9</td>
<td>1.2</td>
<td>0.5</td>
<td>0.5</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>adult</td>
<td>14</td>
<td>48,842</td>
<td>3,528</td>
<td>78</td>
<td>67.4</td>
<td>111.6</td>
<td>531.5</td>
<td>5.9</td>
<td>6039.2</td>
<td>6033.8</td>
<td>860.2</td>
<td>1.1</td>
</tr>
<tr>
<td>letter</td>
<td>17</td>
<td>20,000</td>
<td>695</td>
<td>61</td>
<td>260.0</td>
<td>529.0</td>
<td>7204.8</td>
<td>6.0</td>
<td>10900.0</td>
<td>10155.0</td>
<td>291.3</td>
<td>3.4</td>
</tr>
<tr>
<td>ncvoter</td>
<td>19</td>
<td>1,000</td>
<td>151</td>
<td>755</td>
<td>4.3</td>
<td>4.0</td>
<td>ML</td>
<td>5.1</td>
<td>11.4</td>
<td>1.9</td>
<td>1.1</td>
<td>0.4</td>
</tr>
<tr>
<td>hepatitis</td>
<td>20</td>
<td>155</td>
<td>8</td>
<td>8,250</td>
<td>12.2</td>
<td>175.9</td>
<td>ML</td>
<td>326.7</td>
<td>5576.5</td>
<td>9.5</td>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>horse</td>
<td>27</td>
<td>368</td>
<td>25</td>
<td>128,727</td>
<td>457.0</td>
<td>TL</td>
<td>ML</td>
<td>TL</td>
<td>377.2</td>
<td>382.4</td>
<td>TL</td>
<td>513.0</td>
</tr>
<tr>
<td>fd-reduced-30</td>
<td>30</td>
<td>250,000</td>
<td>69,581</td>
<td>89,571</td>
<td>41.1</td>
<td>77.7</td>
<td>ML</td>
<td>TL</td>
<td>TL</td>
<td>TL</td>
<td>26.9</td>
<td>21.8</td>
</tr>
<tr>
<td>plista</td>
<td>63</td>
<td>1,000</td>
<td>568</td>
<td>178,152</td>
<td>ML</td>
<td>ML</td>
<td>ML</td>
<td>TL</td>
<td>TL</td>
<td>TL</td>
<td>216.5</td>
<td>53.4</td>
</tr>
<tr>
<td>flight</td>
<td>109</td>
<td>1,000</td>
<td>575</td>
<td>982,631</td>
<td>ML</td>
<td>ML</td>
<td>ML</td>
<td>TL</td>
<td>TL</td>
<td>TL</td>
<td>ML</td>
<td>5254.7</td>
</tr>
<tr>
<td>uniprot</td>
<td>223</td>
<td>1,000</td>
<td>2,439</td>
<td>&gt;2,437,556</td>
<td>ML</td>
<td>ML</td>
<td>ML</td>
<td>TL</td>
<td>TL</td>
<td>ML</td>
<td>&gt;5254.7</td>
<td></td>
</tr>
</tbody>
</table>

*Results larger than 1,000 FDs are only counted.*

**Open Research Question**

What is a useful and fair benchmark for data profiling?
Use case: BCNF Normalization

1. Discover FDs
2. Choose key
3. Choose violating FD
4. Decompose

BCNF?
Normalization Results: TPC-H

\[(\text{linenumber}, \text{extendedprice}, \text{discount}, \text{tax}, \text{returnflag}, \text{shipdate}, \text{commitdate}, \text{receiptdate}, \text{comment}, \text{orderkey}, \text{partkey}) \text{ LINEITEM} \]

\[(\text{linenumber}, \text{extendedprice}, \text{tax}, \text{commitdate}, \text{receiptdate}, \text{shipinstruct}) \]

\[(\text{extendedprice}, \text{discount}, \text{shipmode}, \text{orderkey}) \]

\[(\text{quantity}, \text{extendedprice}, \text{partkey}) \]

\[(\text{linestatus}, \text{shipdate}) \]

\[(\text{tax}, \text{returnflag}, \text{orderkey}, \text{partkey}, \text{suppkey}) \]

\[(\text{availqty}, \text{supplicost}, \text{comment}, \text{partkey}, \text{suppkey}) \text{ PARTSUPP} \]

\[(\text{partkey}, \text{name}, \text{brand}, \text{type}, \text{size}, \text{container}, \text{retailprice}, \text{comment}) \text{ PART} \]

\[(\text{mfr}, \text{brand}) \]

\[(\text{suppkey}, \text{name}, \text{address}, \text{phone}, \text{acctbal}, \text{comment}, \text{nationkey}) \text{ SUPPLIER} \]

\[(\text{nationkey}, \text{name}, \text{comment}, \text{regionkey}) \text{ NATION} \]

\[(\text{shippriority}, \text{regionkey}, \text{name}, \text{comment}) \text{ REGION} \]

\[(\text{orderkey}, \text{totalprice}, \text{orderdate}, \text{orderpriority}, \text{clerk}, \text{comment}, \text{custkey}) \text{ ORDERS} \]

\[(\text{orderstatus}, \text{totalprice}, \text{orderdate}) \]

\[(\text{custkey}, \text{name}, \text{address}, \text{phone}, \text{acctbal}, \text{mktsegment}, \text{comment}) \text{ CUSTOMER} \]
1. Basic statistics
2. Uniques and keys
3. Functional dependencies
4. Inclusion dependencies and foreign keys
5. New directions
Inclusion Dependencies for Foreign Key Discovery

- Unary and n-ary INDs
  \[ R[A] \subseteq S[B] \text{ and } R[ABC] \subseteq S[DEF] \]

- Use cases
  - PDB – Protein Data Bank: 175 tables
    - Not a single foreign key constraint
  - Ensembl – genome database: >200 tables
    - Not a single foreign key constraint
  - Web tables:
    - No schema, no constraints, but many connections

- Why are FKS missing?
  - Lack of database knowledge
  - Lack of FK-support in DBMS
  - Fear of performance drop
  - Independent origin
Candidate Set Growth for INDs

Number of attributes: m

Number of levels: k

<table>
<thead>
<tr>
<th>Total</th>
<th>0</th>
<th>2</th>
<th>6</th>
<th>24</th>
<th>80</th>
<th>330</th>
<th>1,302</th>
<th>5,936</th>
<th>26,784</th>
<th>133,650</th>
<th>669,350</th>
<th>3,609,672</th>
<th>19,674,096</th>
<th>113,525,594</th>
<th>664,400,310</th>
</tr>
</thead>
<tbody>
<tr>
<td>I:</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>12</td>
<td>20</td>
<td>30</td>
<td>42</td>
<td>56</td>
<td>72</td>
<td>90</td>
<td>110</td>
<td>132</td>
<td>156</td>
<td>182</td>
<td>210</td>
</tr>
</tbody>
</table>

Felix Naumann
Data Profiling
EDBT 2021
IND Detection Algorithms

Open Research Question
How to efficiently discover all n-ary INDs?

Felix Naumann
Data Profiling
EDBT 2021

[CIKM'19]
Use Case: Web Data Integration
Discovering INDs among millions of web tables
Agenda

1. Basic statistics
2. Uniques and keys
3. Functional dependencies
4. Inclusion dependencies and foreign keys
5. New directions
Are there not more types of dependencies?
Detecting Other Dependencies

- Multi-valued dependencies (MVDs) and join dependencies
- Denial constraints (DCs)
- Detecting order dependencies (ODs)
  - salary “orders” rank
  - `SELECT emp_name
    FROM employees
    ORDER BY rank, salary`

<table>
<thead>
<tr>
<th>emp_name</th>
<th>rank</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>1</td>
<td>40k</td>
</tr>
<tr>
<td>Johnson</td>
<td>1</td>
<td>40k</td>
</tr>
<tr>
<td>Williams</td>
<td>1</td>
<td>45k</td>
</tr>
<tr>
<td>Brown</td>
<td>2</td>
<td>60k</td>
</tr>
<tr>
<td>Davis</td>
<td>2</td>
<td>60k</td>
</tr>
<tr>
<td>Miller</td>
<td>3</td>
<td>70k</td>
</tr>
<tr>
<td>Wilson</td>
<td>4</td>
<td>100k</td>
</tr>
</tbody>
</table>

Open Research Question
How to combine the discovery of multiple dependency types?
But what if the data changes?
Incremental Dependency Discovery

- Insertions, deletions and updates
  - Invalidate existing dependencies
    - Check all dependencies
  - Create new (minimal/maximal) dependencies
    - Re-run entire algorithm?

<table>
<thead>
<tr>
<th>First</th>
<th>Last</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Doe</td>
<td>Berlin</td>
</tr>
<tr>
<td>Jane</td>
<td>Smith</td>
<td>Berlin</td>
</tr>
<tr>
<td>John</td>
<td>Miller</td>
<td>Paris</td>
</tr>
<tr>
<td>John</td>
<td>Smith</td>
<td>Berlin</td>
</tr>
<tr>
<td>John</td>
<td>Doe</td>
<td>Paris</td>
</tr>
</tbody>
</table>
Relaxed Dependencies

- Partial dependencies
- Conditional dependencies
- Matching dependencies

Approximate dependencies
- Dependencies on uncertain data
- Dependencies on incomplete data

---

But what if the data contains errors?

[Caruccio, Deufemia, Polese: Relaxed Functional Dependencies - A Survey of Approaches. TKDE '16]
But aren't these just arbitrary observations on one instance? Genuine Dependencies

- **Features for UCCs as keys**
  - "ID", "PK", etc. in name
  - Few columns, short data types
  - Early in schema
  - Serves as reference

- **Features for INDs as foreign keys**
  - "FK", "ID", etc. in name
  - Referenced columns are a UCC or key
  - Random or even distribution of values

<table>
<thead>
<tr>
<th>Sensor_status</th>
<th>Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23.343455</td>
</tr>
<tr>
<td>1</td>
<td>23.454676</td>
</tr>
<tr>
<td>0</td>
<td>24.001135</td>
</tr>
<tr>
<td>1</td>
<td>24.173099</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cust_ID</th>
<th>Cust_status</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Open Research Question
Can we effectively identify genuine dependencies?
Profiling for Other Data Models

- Traditional data profiling: Single table or multiple tables

- “Newer” data models
  - XML / nested relational / JSON
  - Graphs
  - RDF triples

- New metadata types to profile
  - XML: Nestedness; measures at each nesting level
  - RDF: Graph structure; node-degrees
Profilong Multimedia Data

- Images and videos
  - Color, video-length, volume, etc.
- Textual data
  - Statistical measures
    - Syllables per word
    - Sentence length
    - Parts of speech
  - Vocabulary measures
    - Frequencies of specific words
    - Simpson’s index
  - Content
    - Sentiment

Open Research Question
What even are interesting metadata for multimedia?

Team and Collaborators

- PhD students at HPI
  - Anja Jentzsch
  - Arvid Heise
  - Hazar Harmouch
  - Ioannis Koumarelas
  - Jan Kossmann
  - Jana Bauckmann
  - Lan Jiang
  - Leon Bornemann
  - Sebastian Kruse
  - Sebastian Schmidl
  - Thorsten Papenbrock
  - Tobias Bleifuß
  - Ziawasch Abedjan

- Data profiling collaborators
  - Edson R.L. Filho
  - Eduardo C. de Almeida
  - Eduardo Pena
  - Giuseppe Polese
  - Hannes Mühleisen
  - Heiko Müller
  - Johann Birnick
  - Jorge-Arnulfo Quiané-Ruiz
  - Laure Berti-Équille
  - Loredana Caruccio
  - Lukasz Golab
  - Martin Schirneck
  - Noël Novelli
  - Paolo Papotti
  - Thomas Bläsius
  - Tobias Friedrich
  - Ulf Leser
  - Vincenzo Deufemia
  - Zoi Kaoudi
  - Saravanan Thirumuruganathan

- Plus many great masters students
Summary and Outlook

Current focus on **efficiency**
- New algorithms
- Approximation

Future focus on **efficiency**
- Distribution
- Profiling dynamic data
- Query optimization

Current focus on **semantics**
- Keys and foreign key
- Use cases

Future focus on **semantics**
- Genuineness
- Data cleaning

Current focus on **new problems**
- Relaxed dependencies
- New dependency types

Future focus on **new problems**
- New data models
Open Research Questions

How to support **visual data profiling**?

How to introduce dependency-based **optimization** to DBMS?

Can we **predict** problem and solution size for given data?

Can we find useful **regular expressions** for data columns?

What are effective **distribution** strategies for data profiling?

Can we efficiently **maintain** metadata?

How to efficiently discover all **n-ary INDs**?

How to **combine the discovery** of multiple dependency types?

What even are interesting metadata for **multimedia**?

Can we effectively identify **genuine dependencies**?

Can we define UCCs, FDs, etc. for **trees and graphs**?

What is a useful and fair **benchmark** for data profiling?