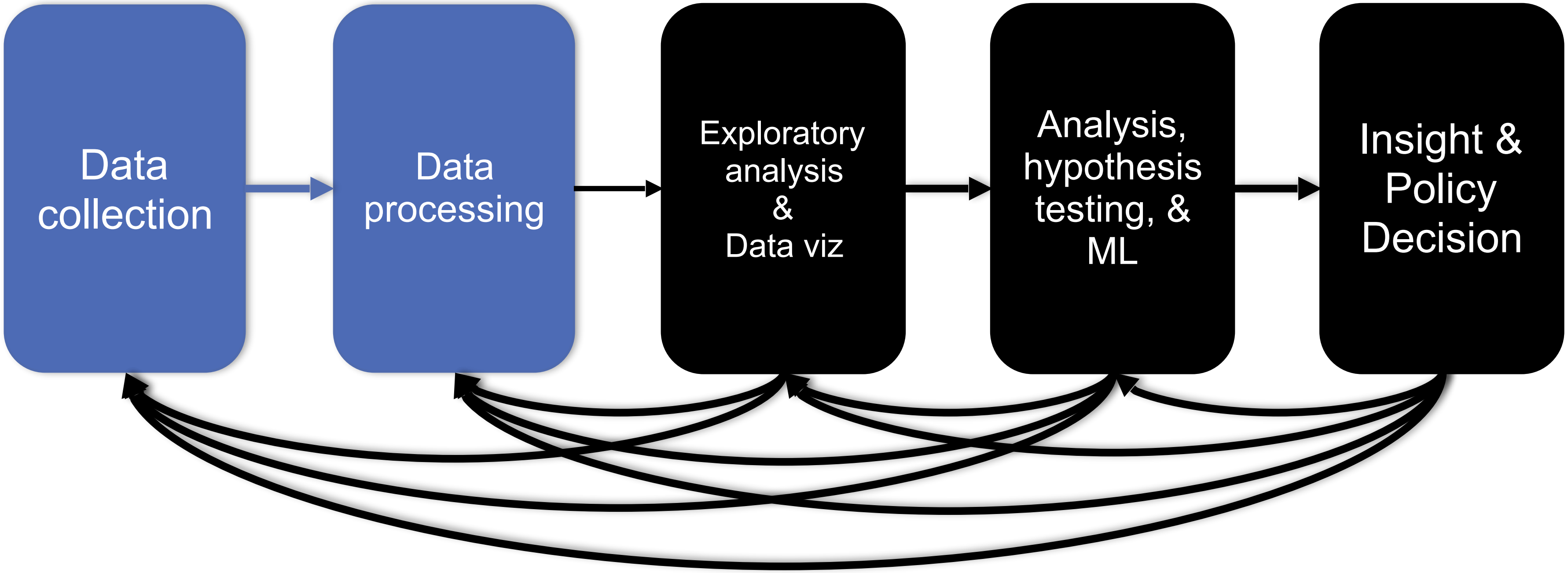
INTRODUCTION TO DATA SCIENCE

COMPUTER SCIENCE
UNIVERSITY OF MARYLAND

REST OF TODAY'S LECTURE


Data collection
Data processing
Exploratory analysis & Data viz
Analysis, hypothesis testing, & ML
Insight & Policy Decision

Continue with the general topic of data wrangling and cleaning

Many slides from Amol Deshpande (UMD)

OVERVIEW

Goal: get data into a structured form suitable for analysis

- Variously called: data preparation, data munging, data curation
- Also often called ETL (Extract-Transform-Load) process

Often the step where majority of time (80-90%) is spent

Key steps:

- Scraping: extracting information from sources, e.g., webpages, spreadsheets
- Data transformation: to get it into the right structure
- Data integration: combine information from multiple sources
- Information extraction: extracting structured information from unstructured/text sources
- Data cleaning: remove inconsistencies/errors

OVERVIEW

Goal: get data into a structured form suitable for analysis

- Variously called: data preparation, data munging, data curation
- Also often called ETL (Extract-Transform-Load) process

Often the step where majority of time (80-90%) is spent

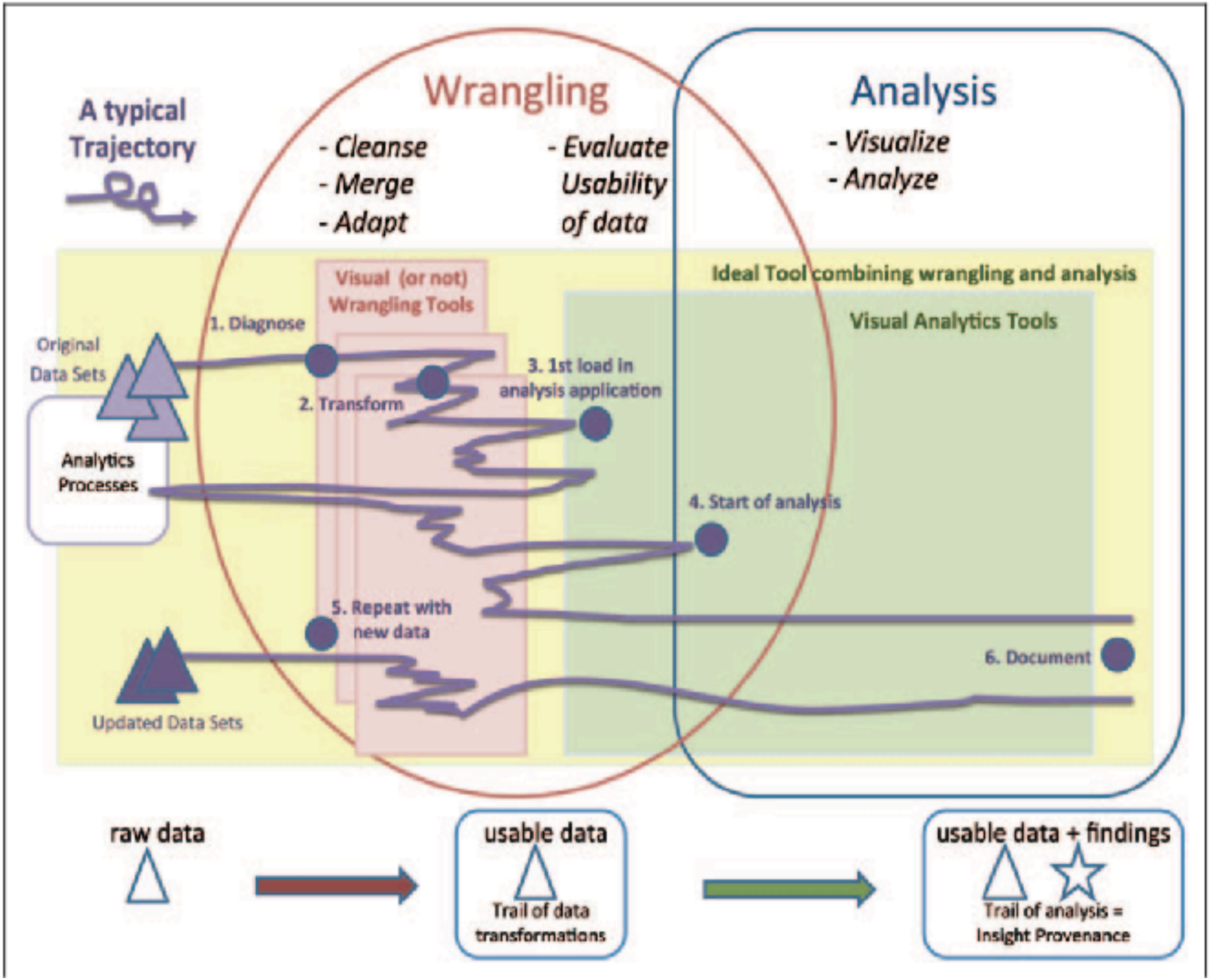
Key steps:

Already covered

- Scraping: extracting information from sources, e.g., webpages, spreadsheets
- Data transformation: to get it into the right structure
- Information extraction: extracting structured information from unstructured/text sources

In a few classes

- **Data integration: combine information from multiple sources**
- **Data cleaning: remove inconsistencies/errors**



A typical Trajectory

Wrangling

- Cleanse
- Merge
- Adapt

- Evaluate Usability of data

Analysis

- Visualize
- Analyze

Ideal Tool combining wrangling and analysis

Visual (or not) Wrangling Tools

Visual Analytics Tools

Original Data Sets

Analytics Processes

1. Diagnose

2. Transform

3. 1st load in analysis application

4. Start of analysis

5. Repeat with new data

6. Document

Updated Data Sets

raw data

usable data

Trail of data transformations

usable data + findings

Trail of analysis = Insight Provenance

OVERVIEW

Many of the problems are not easy to formalize, and have seen little work

- E.g., Cleaning
- Others aspects of integration, e.g., schema mapping, have been studied in depth

A mish-mash of tools typically used

- Visual (e.g., Trifacta), or not (UNIX grep/sed/awk, Pandas)
- Ad hoc programs for cleaning data, depending on the exact type of errors
- Different types of transformation tools
- Visualization and exploratory data analysis to understand and remove outliers/noise
- Several tools for setting up the actual pipelines, assuming the individual steps are setup (e.g., Talend, AWS Glue)

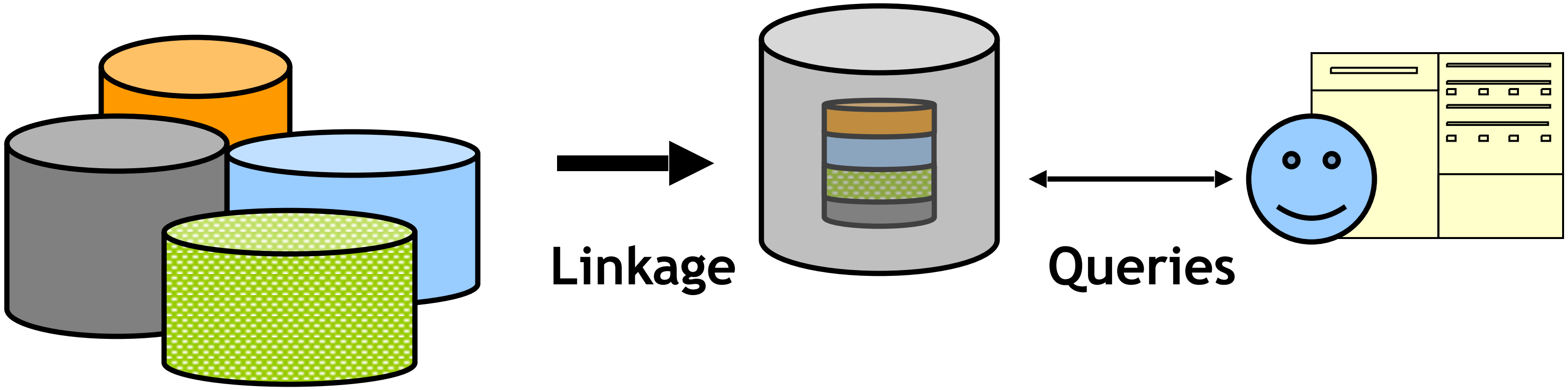
OUTLINE

- Data Integration
- Data Quality Issues
- Data Cleaning
- Entity Resolution

OUTLINE

- Data Integration
- Data Quality Issues
- Data Cleaning
- Entity Resolution

DATA INTEGRATION



- **Discovering** information sources (e.g. deep web modeling, schema learning, …)
- **Gathering** data (e.g., wrapper learning & information extraction, federated search, …)

- **Cleaning** data (e.g., de-duping and **linking records**) to form a single [virtual] database

- **Querying** integrated information sources (e.g. queries to views, execution of web-based queries, …)
- **Data mining & analyzing** integrated information (e.g., collaborative filtering/ classification learning using extracted data, …)

DATA INTEGRATION

Goal: Combine data residing in different sources and provide users with a unified view of these data for querying or analysis

- Each data source has its own schema called **local schemas** (much work assumes relational schemas, but some work on XML as well)
- The unified schema is often called **mediated schema** or **global schema**

Two different setups:

1. Bring the data together into a single repository (often called data warehousing)
2. Keep the data where it is, and send queries back and forth

1. DATA WAREHOUSING

From Data Cleaning: Problems and Current Approaches

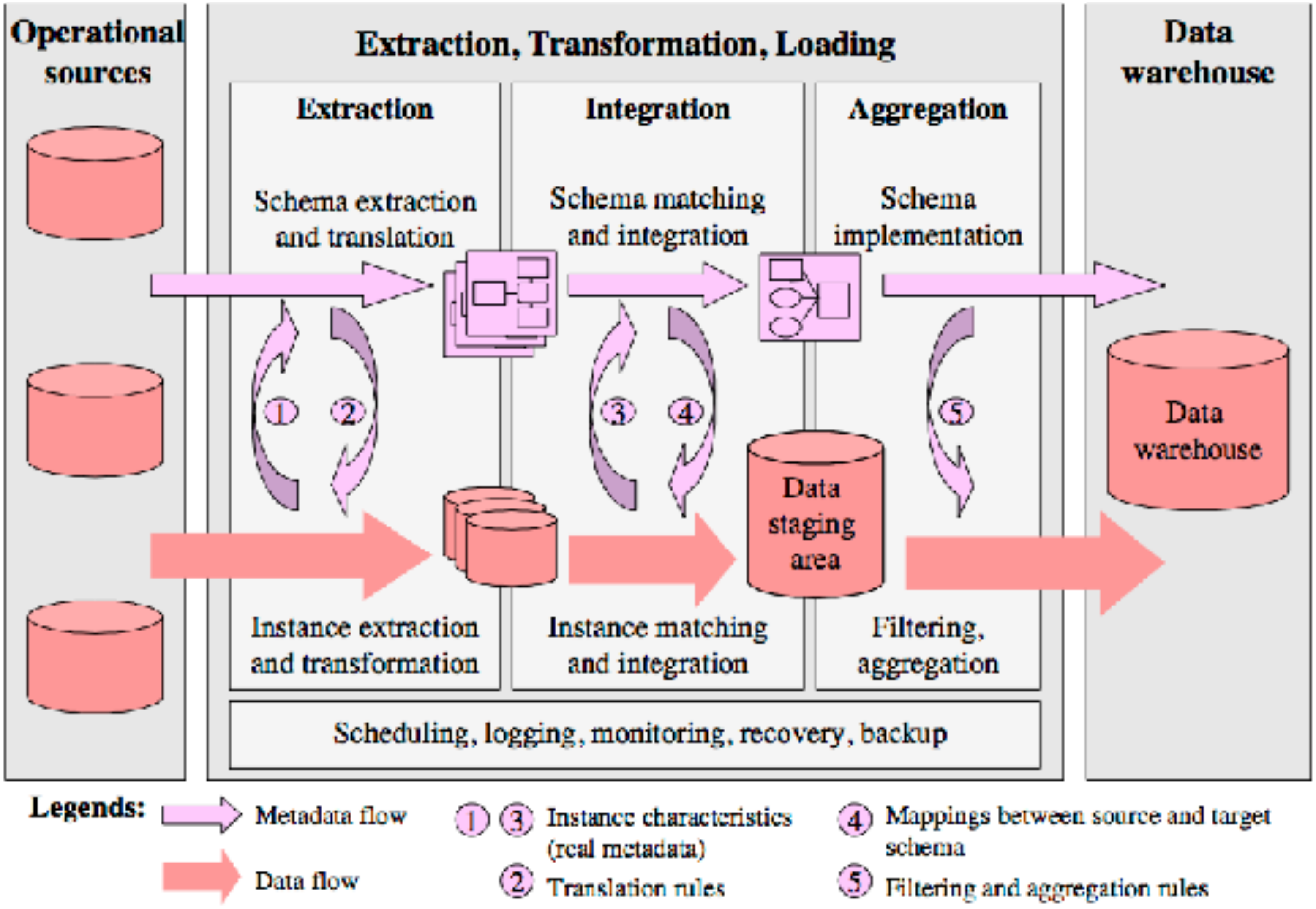
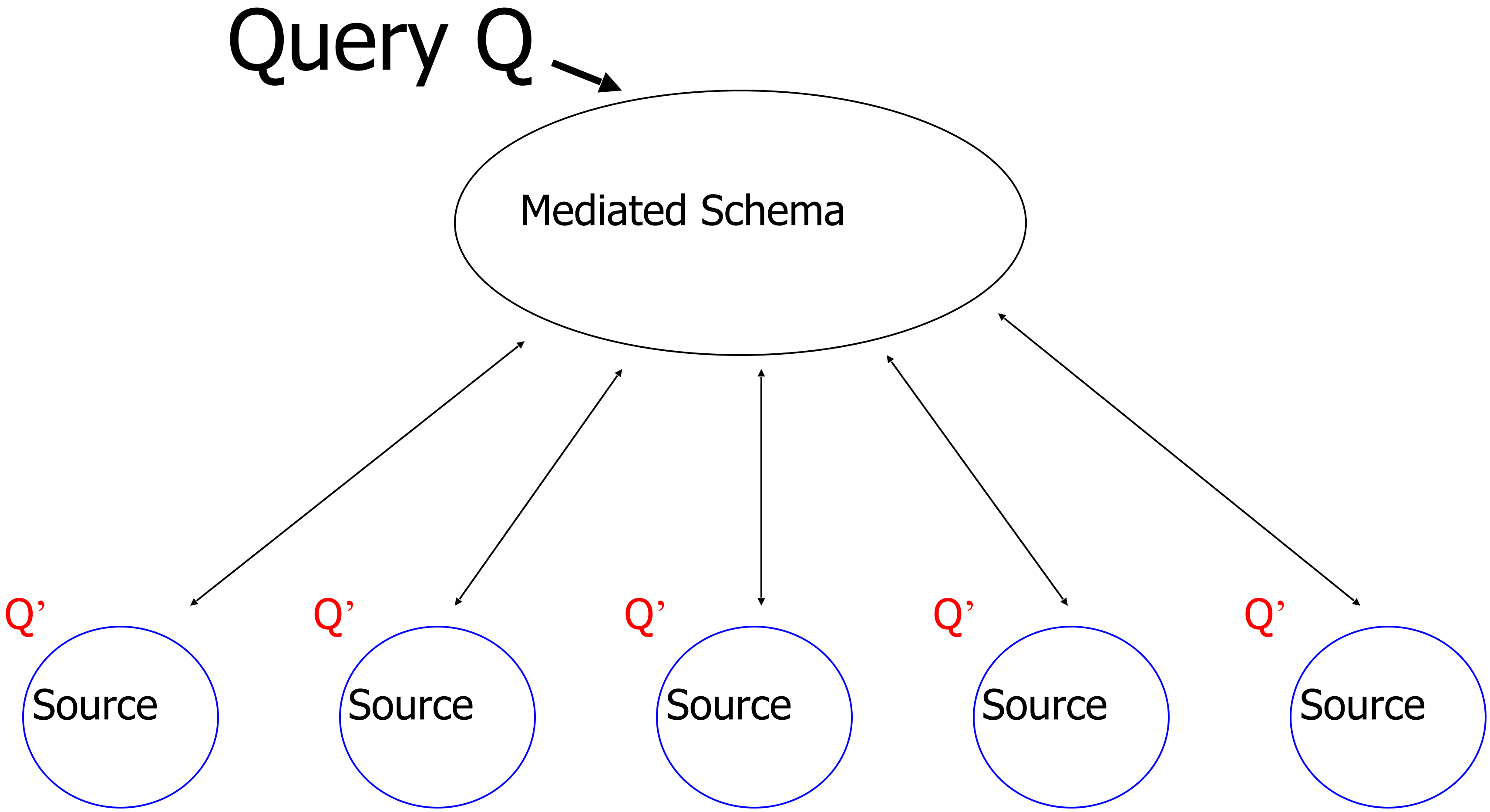

Operational sources
Extraction, Transformation, Loading
Data warehouse
Extraction
Integration
Aggregation
Schema extraction and translation
Schema matching and integration
Schema implementation
1
2
3
4
5
Data warehouse
Data staging area
Instance extraction and transformation
Instance matching and integration
Filtering, aggregation
Scheduling, logging, monitoring, recovery, backup
Legends:
Metadata flow
Data flow
1 3 Instance characteristics (real metadata)
2 Translation rules
4 Mappings between source and target schema
5 Filtering and aggregation rules

Figure 1. Steps of building a data warehouse: the ETL process

2. IN-PLACE INTEGRATION



DATA INTEGRATION

Two different setups:

1. Bring the data together into a single repository (often called data warehousing)
   - Relatively easier problem - only need one-way-mappings
   - Query performance predictable and under your control

2. Keep the data where it is, and send queries back and forth
   - Need two-way mappings -- a query on the mediated schema needs to be translated into queries over data source schemas
   - Not as efficient and clean as data warehousing, but a better fit for dynamic data
   - Or when data warehousing is not feasible

DATA INTEGRATION: KEY CHALLENGES

Data extraction, reconciliation, and cleaning

- Get the data from each source in a structured form
- Often need to use wrappers to extract data from web sources
- May need to define a schema

Schema alignment and mapping

- Decide on the best mediated schema
- Figure out mappings and matchings between the local schemas and the global schema

Answer queries over the global schema

- In the second scenario, need to figure out how to map a query on global schema onto queries over local schemas
- Also need to decide which sources contain relevant data

Limitations in mechanisms for accessing sources

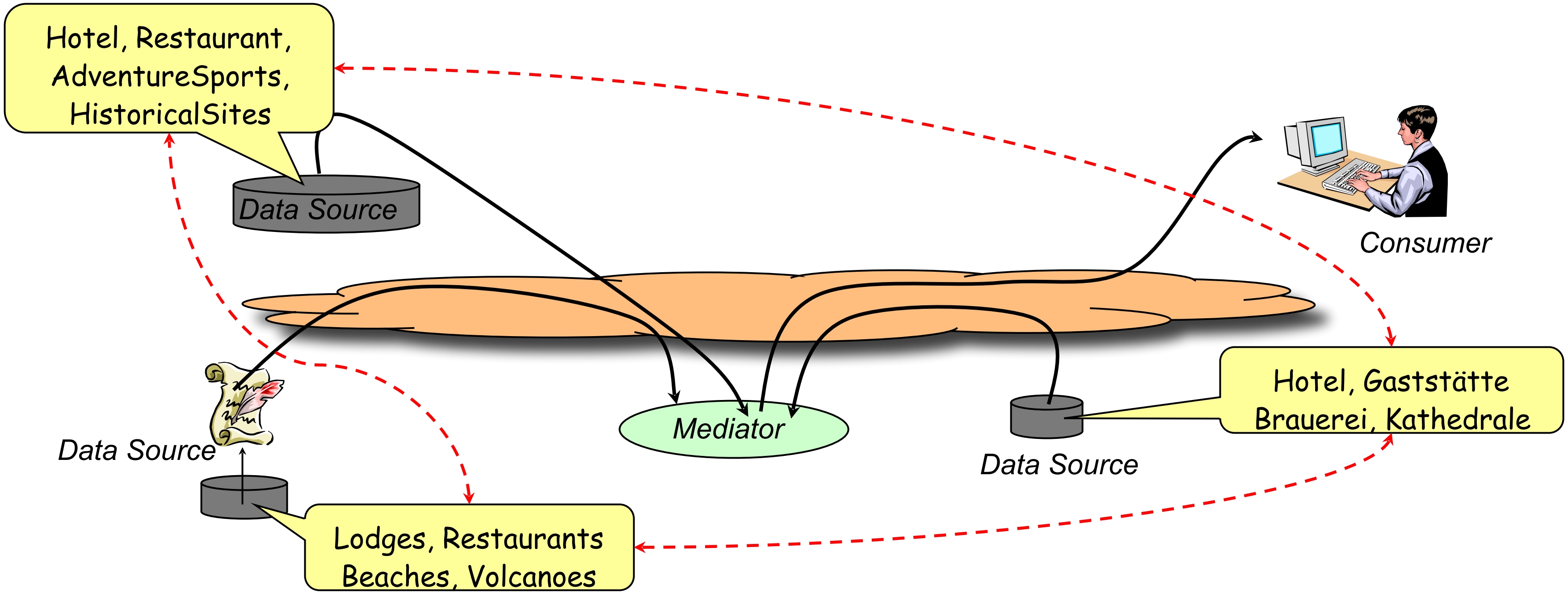
- Many sources have limits on how you can access them
- Limits on the number of queries you can issues (say 100 per min)
- Limits on the types of queries (e.g., must enter a zipcode to get information from a web source)

Thoughts related to Project 1 …

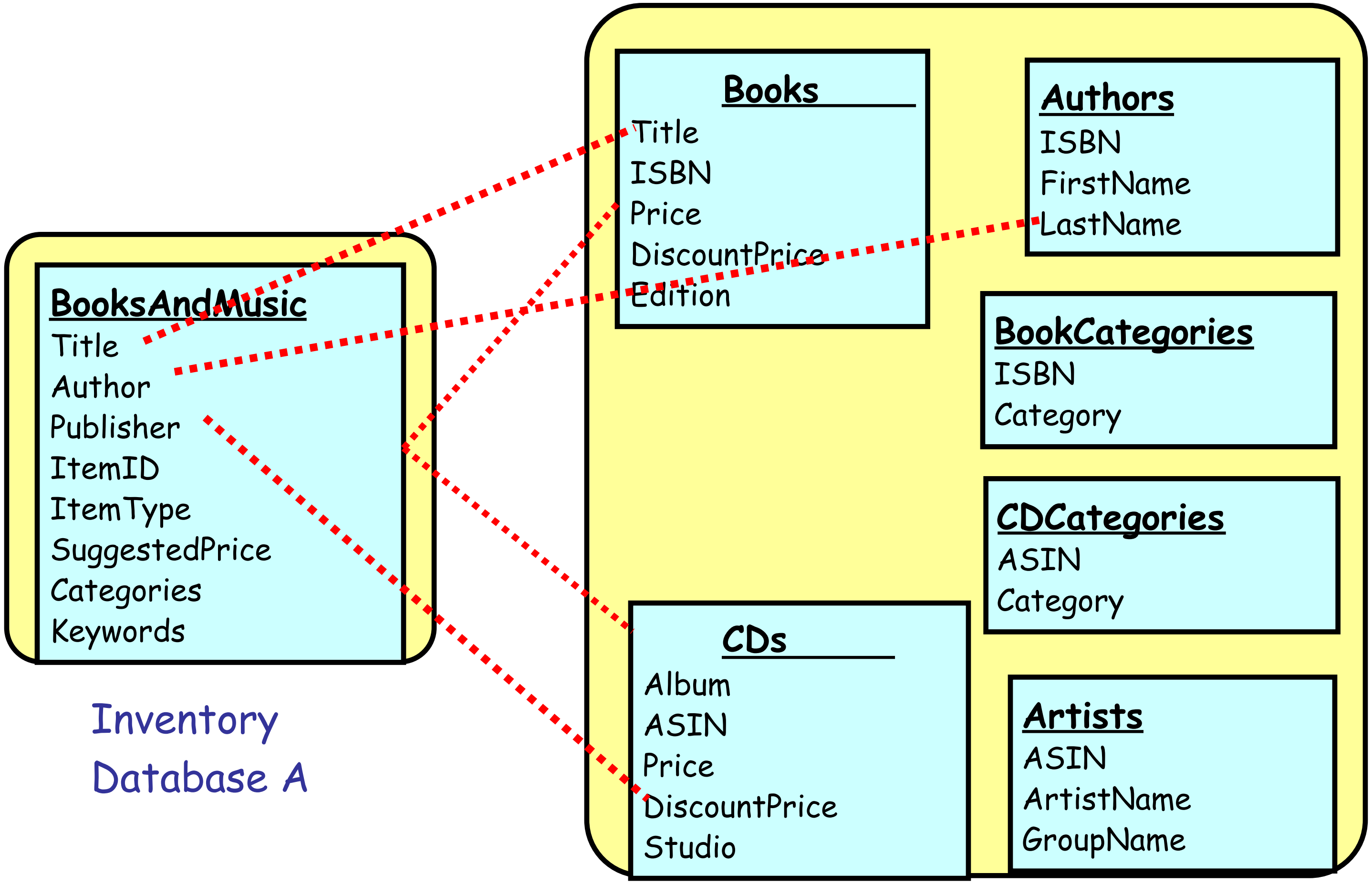
SCHEMA MATCHING OR ALIGNMENT

Goal: Identify corresponding elements in two schemas

- As a first step toward constructing a global schema
- Schema heterogeneity is a key roadblock
  - Different data sources speak their own schema


Hotel, Restaurant, AdventureSports, HistoricalSites
Data Source
Consumer
Data Source
Mediator
Data Source
Hotel, Gaststätte Brauerei, Kathedrale
Lodges, Restaurants Beaches, Volcanoes

SCHEMA MATCHING OR ALIGNMENT



BooksAndMusic
Title
Author
Publisher
ItemID
ItemType
SuggestedPrice
Categories
Keywords

Inventory Database A

Books
Title
ISBN
Price
DiscountPrice
Edition

Authors
ISBN
FirstName
LastName

BookCategories
ISBN
Category

CDCategories
ASIN
Category

CDs
Album
ASIN
Price
DiscountPrice
Studio

Artists
ASIN
ArtistName
GroupName

Inventory Database B

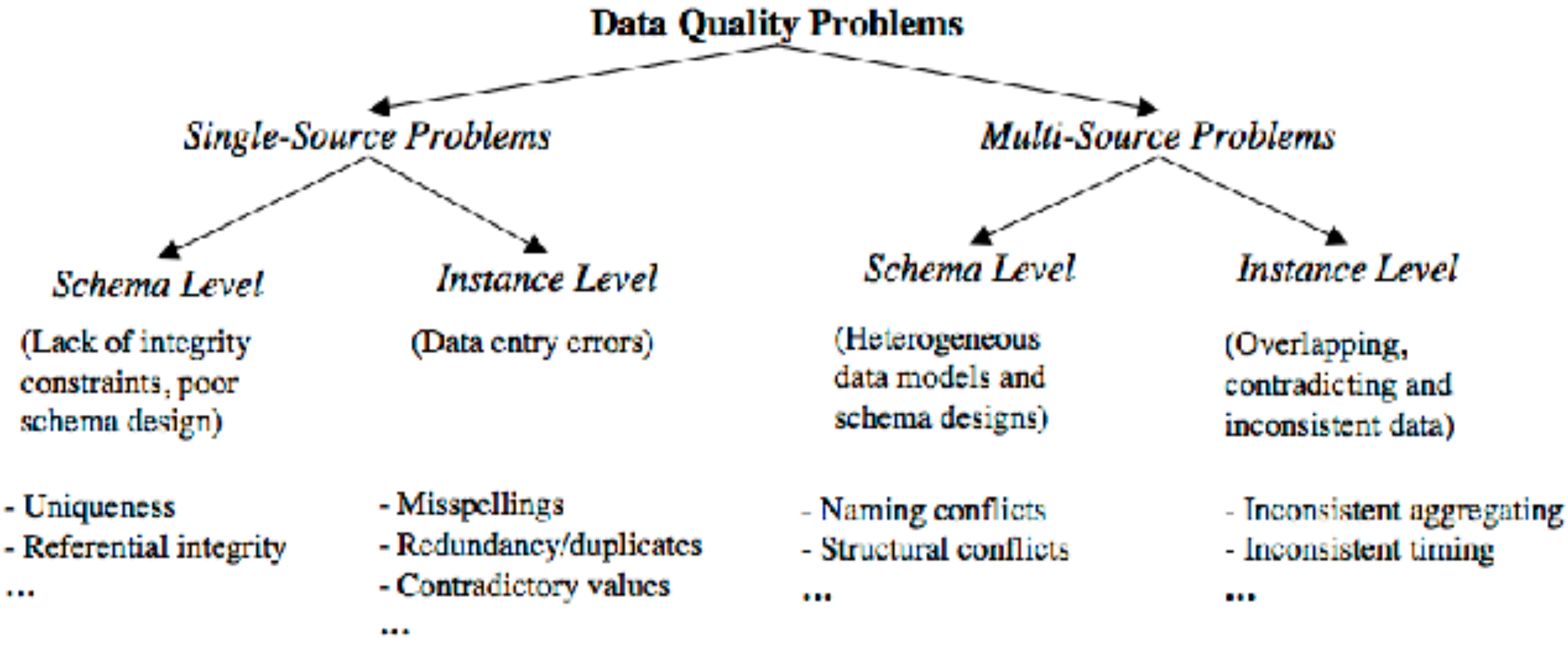
SUMMARY

- **Data integration continues to be a very active area in research and increasingly industry**
- **Solutions still somewhat ad hoc and manual, although tools beginning to emerge**
- **Need to minimize the time needed to integrate a new data source**
  - Crucial opportunities may be lost otherwise
  - Can take weeks to do it properly
- **Dealing with changes to the data sources a major headache**
  - Especially for data sources not under your control

OUTLINE

- Data Integration
- Data Quality Issues
- Data Cleaning
- Entity Resolution

DATA QUALITY PROBLEMS



Data Quality Problems

Single-Source Problems

Schema Level

(Lack of integrity constraints, poor schema design)

- Uniqueness
- Referential integrity

…

Instance Level

(Data entry errors)

- Misspellings
- Redundancy/duplicates
- Contradictory values

…

Multi-Source Problems

Schema Level

(Heterogeneous data models and schema designs)

- Naming conflicts
- Structural conflicts

…

Instance Level

(Overlapping, contradicting and inconsistent data)

- Inconsistent aggregating
- Inconsistent timing

…

Figure 2. Classification of data quality problems in data sources

From Data Cleaning: Problems and Current Approaches

SINGLE-SOURCE PROBLEMS

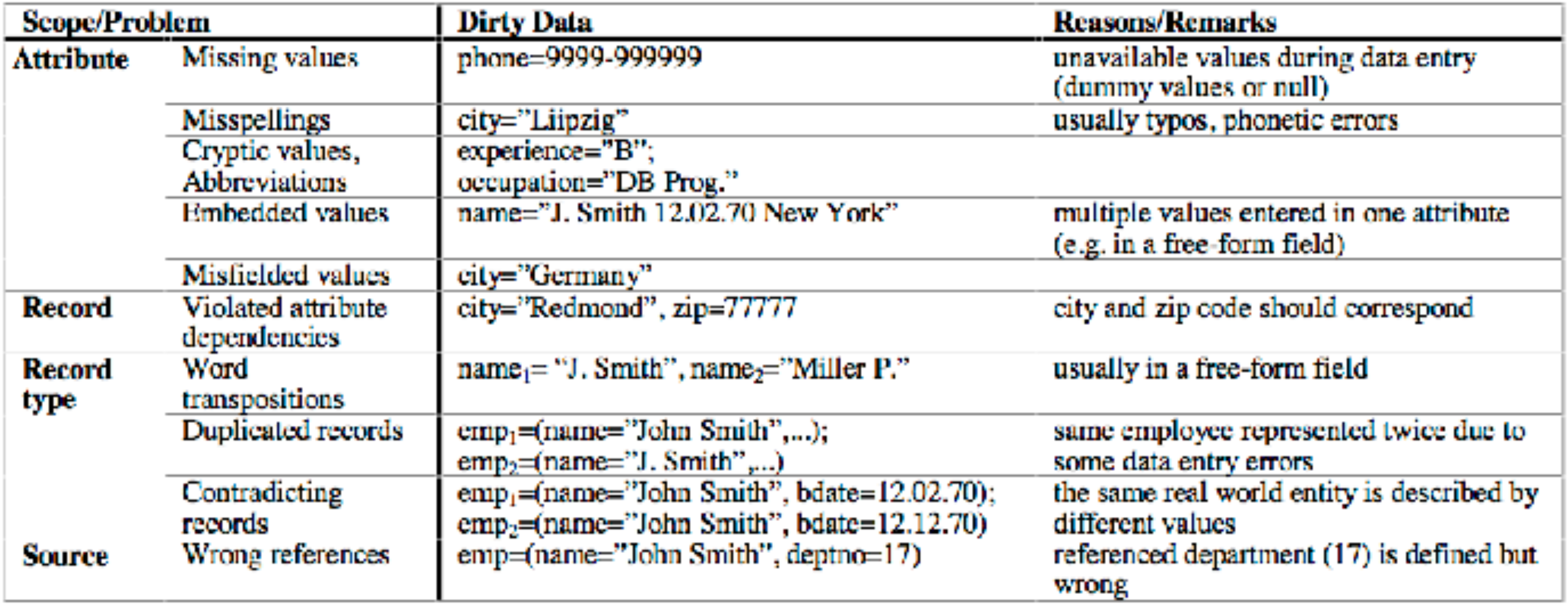
Depends largely on the source

Databases can enforce constraints, whereas data extracted from files or spreadsheets, or scraped from webpages is much more messy

Types of problems:

- Ill-formatted data, especially from webpages or files or spreadsheets
- Missing or illegal values, Misspellings, Use of wrong fields, Extraction issues (not easy to separate out different fields)
- Duplicated records, Contradicting Information, Referential Integrity Violations
- Unclear default values (e.g., data entry software needs something)
- Evolving schemas or classification schemes (for categorical attributes)
- Outliers

DATA QUALITY PROBLEMS



Scope/Problem		Dirty Data	Reasons/Remarks
Attribute	Missing values	phone=9999-999999	unavailable values during data entry (dummy values or null)
	Misspellings	city="Liipzig"	usually typos, phonetic errors

	Cryptic values, Abbreviations	experience="B";
		occupation="DB Prog."	

	Embedded values	name="J. Smith 12.02.70 New York"	multiple values entered in one attribute (e.g. in a free-form field)
	Misfielded values	city="Germany"	
Record	Violated attribute dependencies	city="Redmond", zip=77777	city and zip code should correspond
Record type	Word transpositions	$name_1$= "J. Smith", $name_2$="Miller P."	usually in a free-form field
	Duplicated records	emp_1=(name="John Smith",...); emp_2=(name="J. Smith",...)	same employee represented twice due to some data entry errors
	Contradicting records	emp_1=(name="John Smith", bdate=12.02.70); emp_2=(name="John Smith", bdate=12.12.70)	the same real world entity is described by different values
Source	Wrong references	emp=(name="John Smith", deptno=17)	referenced department (17) is defined but wrong

Table 2. Examples for single-source problems at instance level

MULTI-SOURCE PROBLEMS

Different sources are developed separately, and maintained by different people

Issue 1: Mapping information across sources (schema mapping/transformation)

- Naming conflicts: same name used for different objects
- Structural conflicts: different representations across sources
- We will cover this later

Issue 2: Entity Resolution: Matching entities across sources

Issue 3: Data quality issues

- Contradicting information, Mismatched information, etc.

OUTLINE

- **Data Integration**
- **Data Quality Issues**
- **Data Cleaning**
    - Outlier Detection
    - Constraint-based Cleaning
    - Entity Resolution

UNIVARIATE OUTLIERS

A set of values can be characterized by metrics such as center (e.g., mean), dispersion (e.g., standard deviation), and skew

Can be used to identify outliers

- Must watch out for "masking": one extreme outlier may alter the metrics sufficiently to mask other outliers
- Should use **robust statistics**: considers effect of corrupted data values on distributions – we will talk about this in depth later
- Robust center metrics: median, k% trimmed mean (discard lowest and highest k% values)
- Robust dispersion:
    - Median Absolute Deviation (MAD): median distance of all the values from the median value

A reasonable approach to find outliers: any data points 1.4826x MAD away from median

- The above assumes that data follows a **normal** distribution
- May need to eyeball the data (e.g., plot a histogram) to decide if this is true

UNIVARIATE OUTLIERS

Wikipedia Article on Outliers lists several other normality-based tests for outliers

If data appears to be not normally distributed:

- Distance-based methods: look for data points that do not have many neighbors
- Density-based methods:
    - Define *density* to be average distance to *k* nearest neighbors
    - *Relative density* = density of node/average density of its neighbors
    - Use relative density to decide if a node is an outlier

Most of these techniques start breaking down as the dimensionality of the data increases

- *Curse of dimensionality*
- Can project data into lower-dimensional space and look for outliers there
    - Not as straightforward

OTHER OUTLIERS

Timeseries outliers

- Often the data is in the form of a timeseries
- Can use the historical values/patterns in the data to flag outliers
- Rich literature on *forecasting* in timeseries data

Frequency-based outliers

- An item is considered a "heavy hitter" if it is much more frequent than other items
- In relational tables, can be found using a simple *groupby-count*
- Often the volume of data may be too much (e.g., internet routers)
  - Approximation techniques often used
  - To be discussed sometime later in the class

Things generally not as straightforward with other types of data

- Outlier detection continues to be a major research area

WRAP-UP

Data wrangling/cleaning are a key component of data science pipeline

Still largely ad hoc although much tooling in recent years

Specifically, we covered:

- Schema mapping and matching
- Outliers

Next up:

- Constraint-based Cleaning
- Entity Resolution/Record Linkage/Data Matching

DATA CLEANING: OUTLIER RESOLUTION

From: **Entity Resolution Tutorial**

Identify different manifestations of the same real world object

- Also called: identity reconciliation, record linkage, deduplication, fuzzy matching, Object consolidation, Coreference resolution, and several others

Motivating examples: ?????????????

- Postal addresses
- Entity recognition in NLP/Information Extraction
- Identifying companies in financial records
- Comparison shopping
- Author disambiguation in citation data
- Connecting up accounts on online networks
- Crime/Fraud Detection
- Census
- ...

DATA CLEANING: OUTLIER RESOLUTION

Important to correctly identify references

- Often actions taken based on extracted data
- Cleaning up data by entity resolution can show structure that may not be apparent before

Challenges

- Such data is naturally ambiguous (e.g., names, postal addresses)
- Abbreviations/data truncation
- Data entry errors, Missing values, Data formatting issues complicate the problem
- Heterogeneous data from many diverse sources

No magic bullet here !!

- Approaches fairly domain-specific
- Be prepared to do a fair amount of manual work

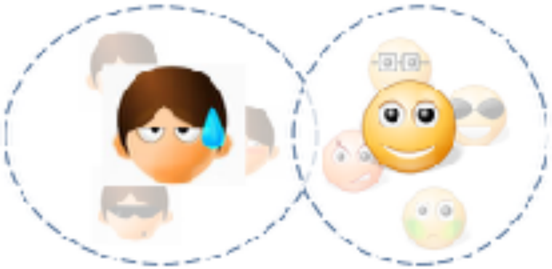
ENTITY RESOLUTION: THREE SLIGHTLY DIFFERENT PROBLEMS

Setup:

- Real world: there are entities (people, addresses, businesses)
- We have a large collection of noisy, ambiguous "references" to those entities (also called "mentions")
- Somewhat different techniques, but a lot of similarities

Deduplication

- Cluster records/mentions that correspond to the same entity
- Choose/construct a cluster representative
  - This is in itself a non-trivial task (e.g., averaging may work for numerical attributes, but what about string attributes?)



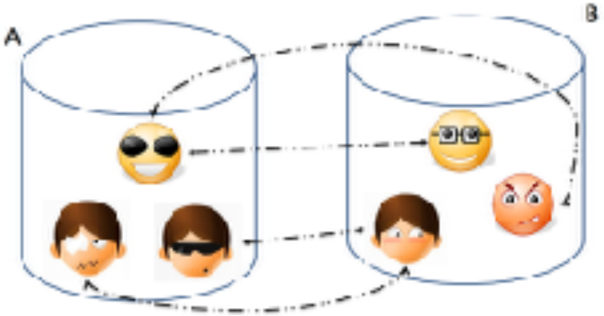
ENTITY RESOLUTION: THREE SLIGHTLY DIFFERENT PROBLEMS

Setup:

- Real world: there are entities (people, addresses, businesses)
- We have a large collection of noisy, ambiguous "references" to those entities (also called "mentions")
- Somewhat different techniques, but a lot of similarities

Record Linkage

- Match records across two different databases (e.g., two social networks, or financial records w/ campaign donations)
- Typically assume that the two databases are fairly clean



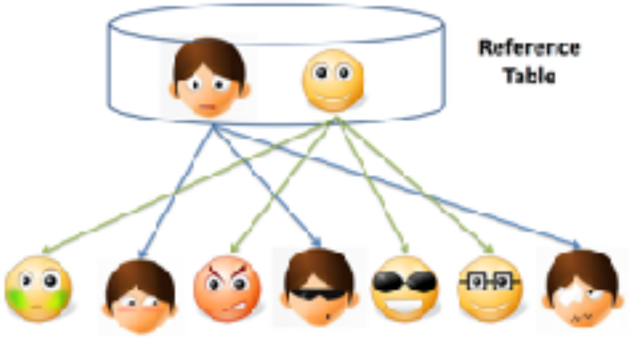
ENTITY RESOLUTION: THREE SLIGHTLY DIFFERENT PROBLEMS

Setup:

- Real world: there are entities (people, addresses, businesses)
- We have a large collection of noisy, ambiguous "references" to those entities (also called "mentions")
- Somewhat different techniques, but a lot of similarities

Reference Matching

- Match "references" to clean records in a reference table
- Commonly comes up in "entity recognition" (e.g., matching newspaper article mentions to names of people)



ENTITY RESOLUTION: DATA MATCHING

Comprehensive treatment: Data Matching; P. Christen; 2012 (Springer Books -- not available for free)

One of the key issues is finding similarities between two references

- What similarity function to use?

Edit Distance Functions

- Levenstein: min number of changes to go from one reference to another
  - A change is defined to be: a single character insertion or deletion or substitution
  - May add transposition
- Many adjustments to the basic idea proposed (e.g., higher weights to changes at the start)
- Not cheap to compute, especially for millions of pairs

Set Similarity

- Some function of intersection size and union size
- E.g., Jaccard distance = size of intersection/size of union
- Much faster to compute

Vector Similarity

- Cosine similarity – we'll talk about this much more in NLP lectures

ENTITY RESOLUTION: DATA MATCHING

Q-Grams

- Find all length-q substrings in each string
- Use set/vector similarity on the resulting set

Several approaches that combine the above (especially q-grams and edit distance, e.g., Jaro-Winkler)

Soundex: Phonetic Similarity Metric

- Homophones should be encoded to the same representation so spelling errors can be handled
- Robert and Rupert get assigned the same code (R163), but Rubin yields R150

May need to use Translation Tables

- To handle abbreviations, nicknames, other synonyms

Different types of data requires more domain-specific functions

- E.g., geographical locations, postal addresses
- Also much work on computing distances between XML documents etc.

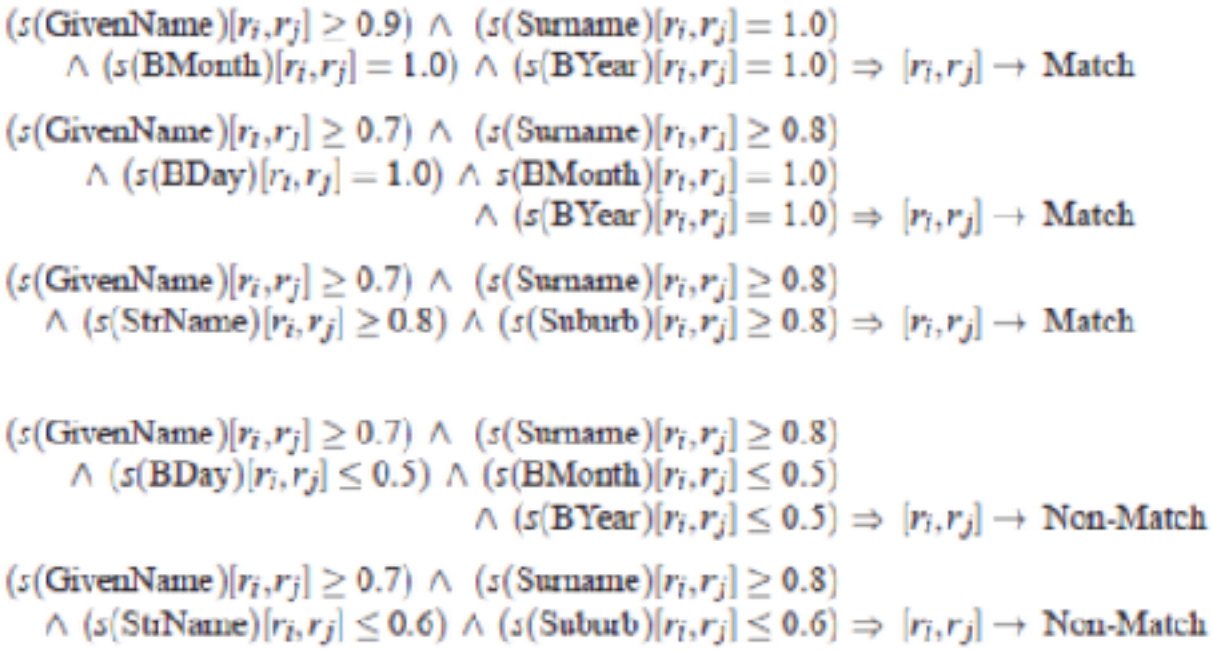
ENTITY RESOLUTION: ALGORITHMS

Simple threshold method

- If the distance below some number, the two references are assumed to be equal
- May review borderline matches manually

Can be generalized to rule-based:

- Example from Christen, 2012



$$(s(\text{GivenName})[r_i,r_j] \geq 0.9) \wedge (s(\text{Surname})[r_i,r_j] = 1.0) \wedge (s(\text{BMonth})[r_i,r_j] = 1.0) \wedge (s(\text{BYear})[r_i,r_j] = 1.0) \Rightarrow [r_i,r_j] \rightarrow \text{Match}$$

$$(s(\text{GivenName})[r_i,r_j] \geq 0.7) \wedge (s(\text{Surname})[r_i,r_j] \geq 0.8) \wedge (s(\text{BDay})[r_i,r_j] = 1.0) \wedge s(\text{BMonth})[r_i,r_j] = 1.0) \wedge (s(\text{BYear})[r_i,r_j] = 1.0) \Rightarrow [r_i,r_j] \rightarrow \text{Match}$$

$$(s(\text{GivenName})[r_i,r_j] \geq 0.7) \wedge (s(\text{Surname})[r_i,r_j] \geq 0.8) \wedge (s(\text{StrName})[r_i,r_j] \geq 0.8) \wedge (s(\text{Suburb})[r_i,r_j] \geq 0.8) \Rightarrow [r_i,r_j] \rightarrow \text{Match}$$

$$(s(\text{GivenName})[r_i,r_j] \geq 0.7) \wedge (s(\text{Surname})[r_i,r_j] \geq 0.8) \wedge (s(\text{BDay})[r_i,r_j] \leq 0.5) \wedge (s(\text{BMonth})[r_i,r_j] \leq 0.5) \wedge (s(\text{BYear})[r_i,r_j] \leq 0.5) \Rightarrow [r_i,r_j] \rightarrow \text{Non-Match}$$

$$(s(\text{GivenName})[r_i,r_j] \geq 0.7) \wedge (s(\text{Surname})[r_i,r_j] \geq 0.8) \wedge (s(\text{StrName})[r_i,r_j] \leq 0.6) \wedge (s(\text{Suburb})[r_i,r_j] \leq 0.6) \Rightarrow [r_i,r_j] \rightarrow \text{Non-Match}$$

ENTITY RESOLUTION: ALGORITHMS

May want to give more weight to matches involving rarer words

- More naturally applicable to record linkage problem
- If two records match on a rare name like "Machanavajjhala", they are likely to be a match
- Can formalize this as "probabilistic record linkage”

Constraints: May need to be satisfied, but can also be used to find matches

- Often have constraints on the matching possibilities
- Transitivity: M1 and M2 match, and M2 and M3 match, and M1 and M3 must match
- Exclusivity: M1 and M2 match --> M3 cannot match with M2
- Other types of constraints:
    - E.g., if two papers match, their venues must match

ENTITY RESOLUTION: ALGORITHMS

Clustering-based ER Techniques:

- Deduplication is basically a clustering problem
- Can use clustering algorithms for this purpose
- But most clusters are very small (in fact of size = 1)
- Some clustering algorithms are better suited for this, especially Agglomerative Clustering
    - Unlikely K-Means would work here

ENTITY RESOLUTION: ALGORITHMS

Crowdsourcing

- Humans are often better at this task
- Can use one of the crowdsourcing mechanisms (e.g., Mechanical Turk) for getting human input on the difficult pairs
- Quite heavily used commercially (e.g., to disambiguate products, restaurants, etc.)

ENTITY RESOLUTION: SCALING TO BIG DATA

One immediate problem

- There are O(N^2) possible matches
- Must reduce the search space

Use some easy-to-evaluate criterion to restrict the pairs considered further

- May lead to false negative (i.e., missed matches) depending on how noisy the data is

Much work on this problem as well, but domain-specific knowledge likely to be more useful in practice

One useful technique to know: min-hash signatures

- Can quickly find potentially overlapping sets
- Turns up to be very useful in many domains (beyond ER)