Data Integration — 1

Faculty of Information Technology, Monash University, Australia

FIT5196 week 10

Outline

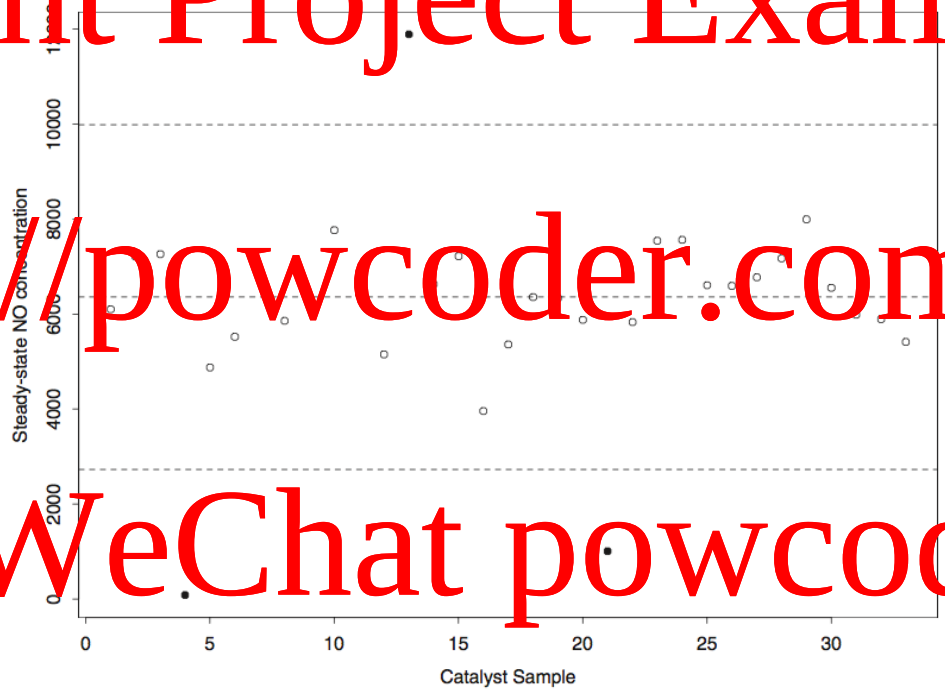
1. Recap
2. Data Integration
   - Definition & Application
   - Schema Integration
3. Summary

Outliers: the definition

- Types of outliers
  - Univariate outlier
  - Multivariate outlier

Outliers: Recap

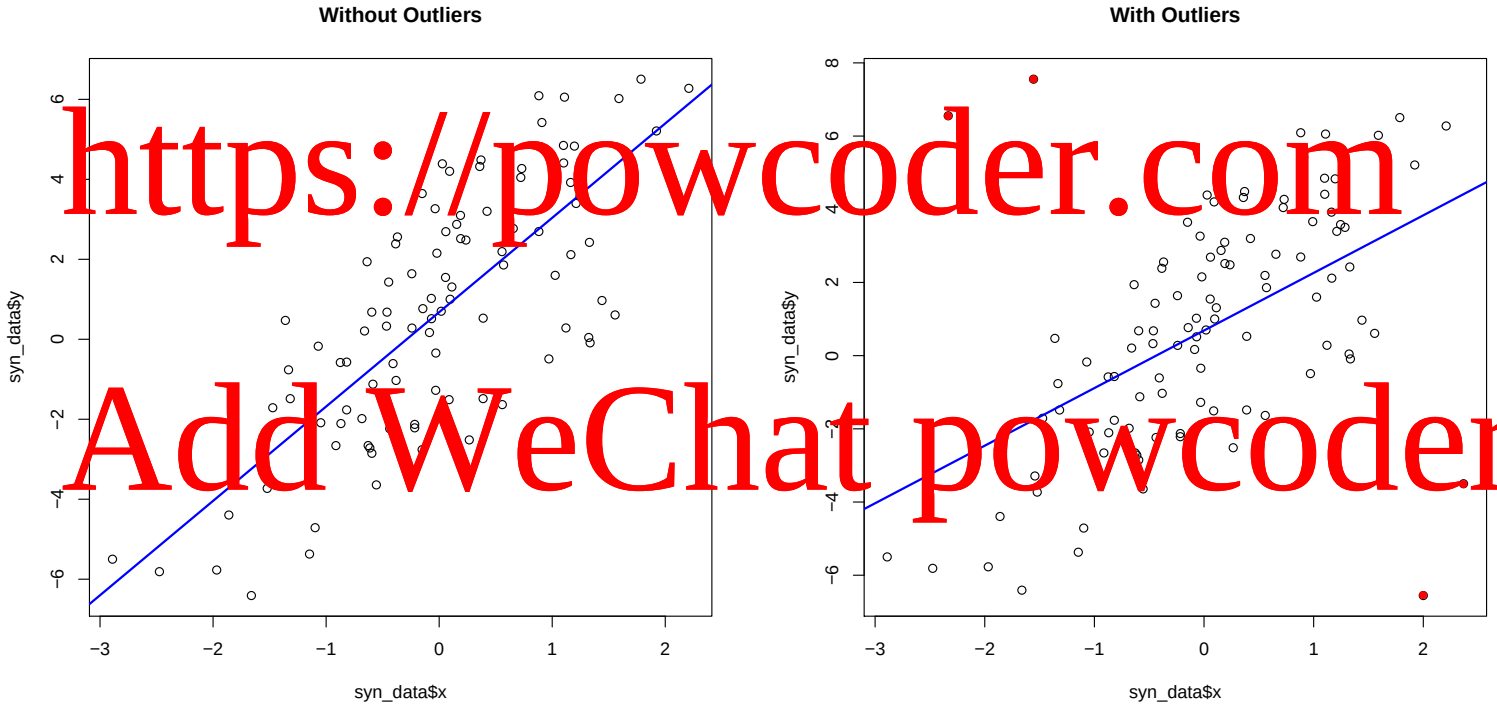
- Types of outliers
  - Univariate outlier



  - Multivariate outlier

Outliers: Recap

- Types of outliers
  - Univariate outlier
  - Multivariate outlier: for example bivariate outlier

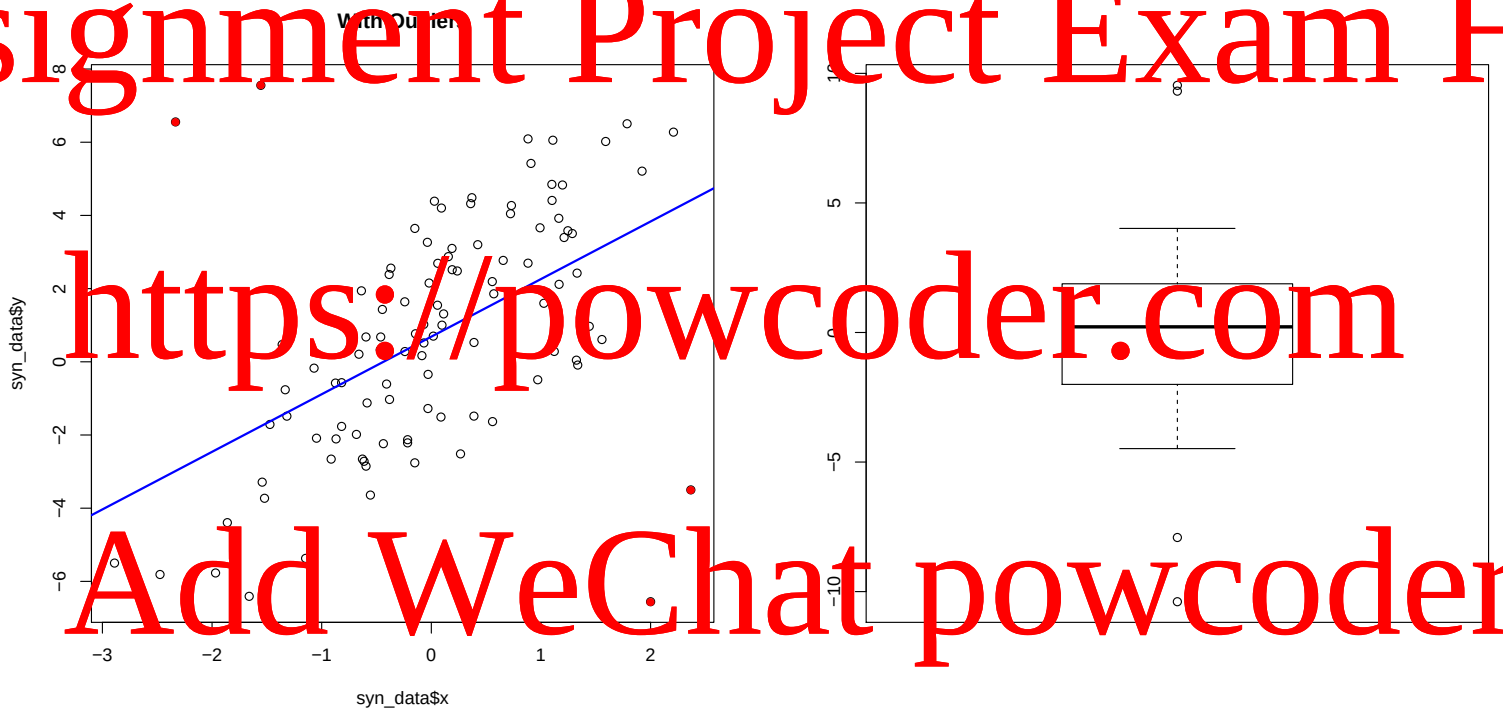


Outliers: Recap

- Univariate outlier detection
  - The 3σ edit rule
  - The Hampel identifier
  - The standard boxplot outlier rule

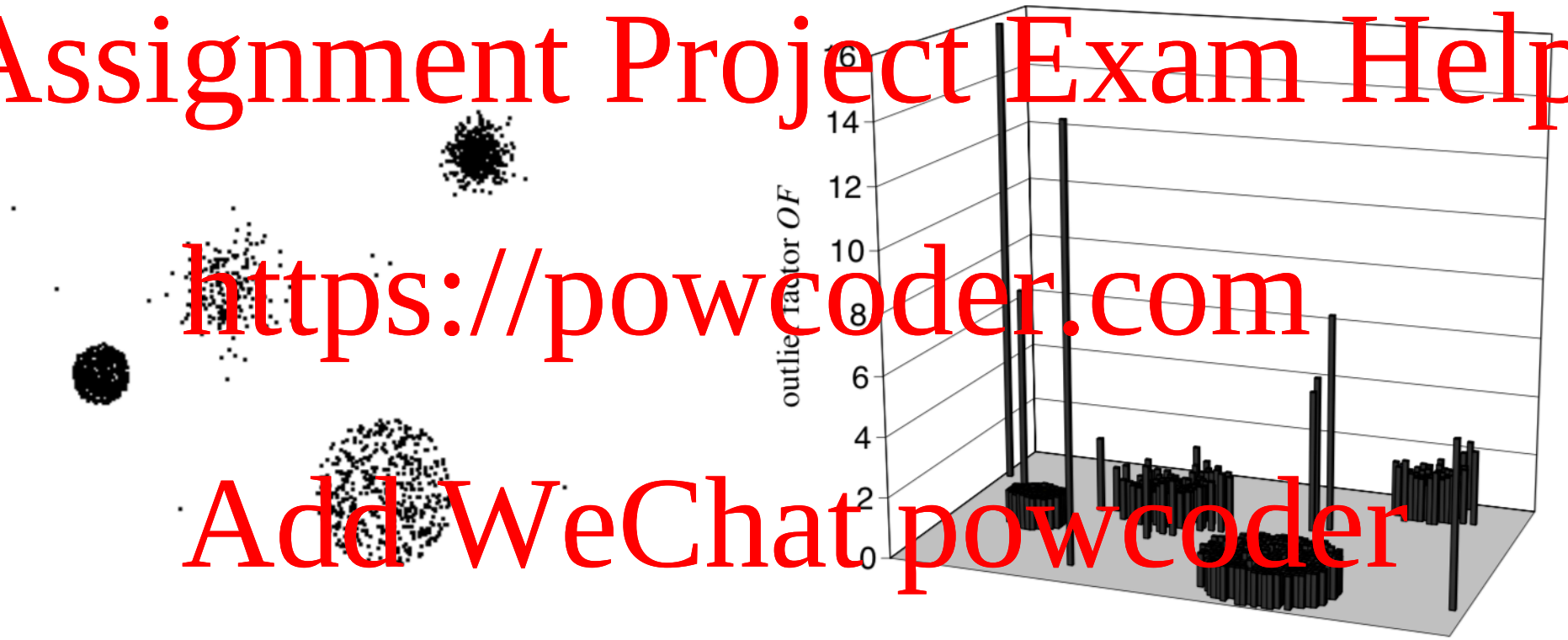
Outliers: Recap

- Multivariate outlier detection based on linear regression

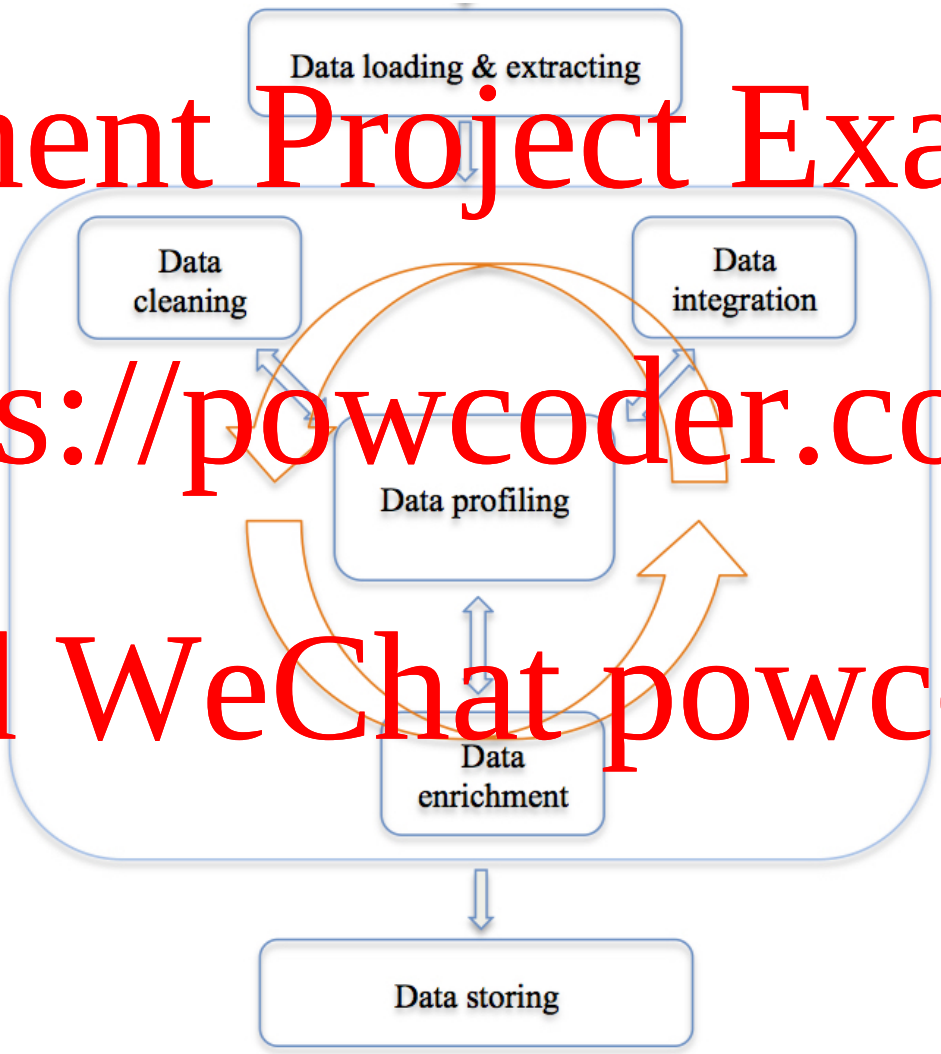


Outliers: Recap

- Multivariate outlier detection based on LOF



Data Wrangling Process



Outline

1 Recap

2 Data Integration
- Definition & Application
- Schema Integration

3 Summary

Data Integration: Definition

- A process in which heterogeneous data is retrieved and combined as an incorporated form and structure.
  - FullServe: an American-based ISP provides Internet access to homes.
  - EuroCard: an ISP acquired by FullServe

Employee Database
FullTimeEmps(ssn, empID, firstName, middleName, lastName)
Hire(empID, hireDate, recruiter)
TempEmployees(ssn, hireStart, hireEnd, name, hourlyRate)

Resume Database
Interviews(interviewDate, pID, recruiter, hireDecision, hireDate)
CVs(pID, resume)

Training Database
Courses(courseID, name, instructor)
Enrollments(courseID, empID, date)

Services Database
Services(packName, textDescription)
Customers(name, ID, zipCode, streetAdr, phone)
Contracts(custID, packName, startDate)

Sales Database
Products(prodName, prodID)
Sales(prodID, customerID, custName, address)

HelpLine Database
Calls(date, agent, custID, text, action)

FIGURE 1.1 Some of the databases a company like FullServe may have. For each database, we show some of the tables and for each table, some of its attributes. For example, the Employee database has a table FullTimeEmps with attributes ssn, empID, firstName, middleName, and lastName.

Data Integration: Definition

- A process in which heterogeneous data is retrieved and combined as an incorporated form and structure.
  - FullServe: an American-based ISP provides Internet access to homes,
  - EuroCard: an ISP acquired by FullServe

Employee Database
Emp(ID, firstName, middleInitial, lastName, salary)
Hire(ID, hireDate, recruiter)

Resume Database
Interviews(ID, date, location, recruiter)
CVs(candID, resume)

Credit Card Database
Cards(CustID, cardNum, expiration, currentBalance)
Customers(CustID, name, address)

HelpLine Database
Calls(date, agent, custID, description, followup)

FIGURE 1.2 Some of the databases of EuroCard. Note that EuroCard organizes its data quite differently from FullServe. For example, EuroCard does not distinguish between full-time and part-time employees. FullServe records the hire data of employees in the Resume database and the Employee database, while EuroCard only records the hire date in the Employee database.

Data Integration: Definition

- Data fusion: the integration of data and knowledge from several sources
  - The Human Resources Department needs to be able to query for all of its employees whether in the United States or in Europe.
  - Combining data from the HelpLine database and the Sales database will help FullServe identify issues in their products and services early on,.
  - Customer support hotline: a customer representative might need to know customers Internet service products purchased, call notes.

Data Integration: Definition

- Goal: create a single representation that provides a more accurate description than any of the individual data sources

Data Integration: Application

MONASH University

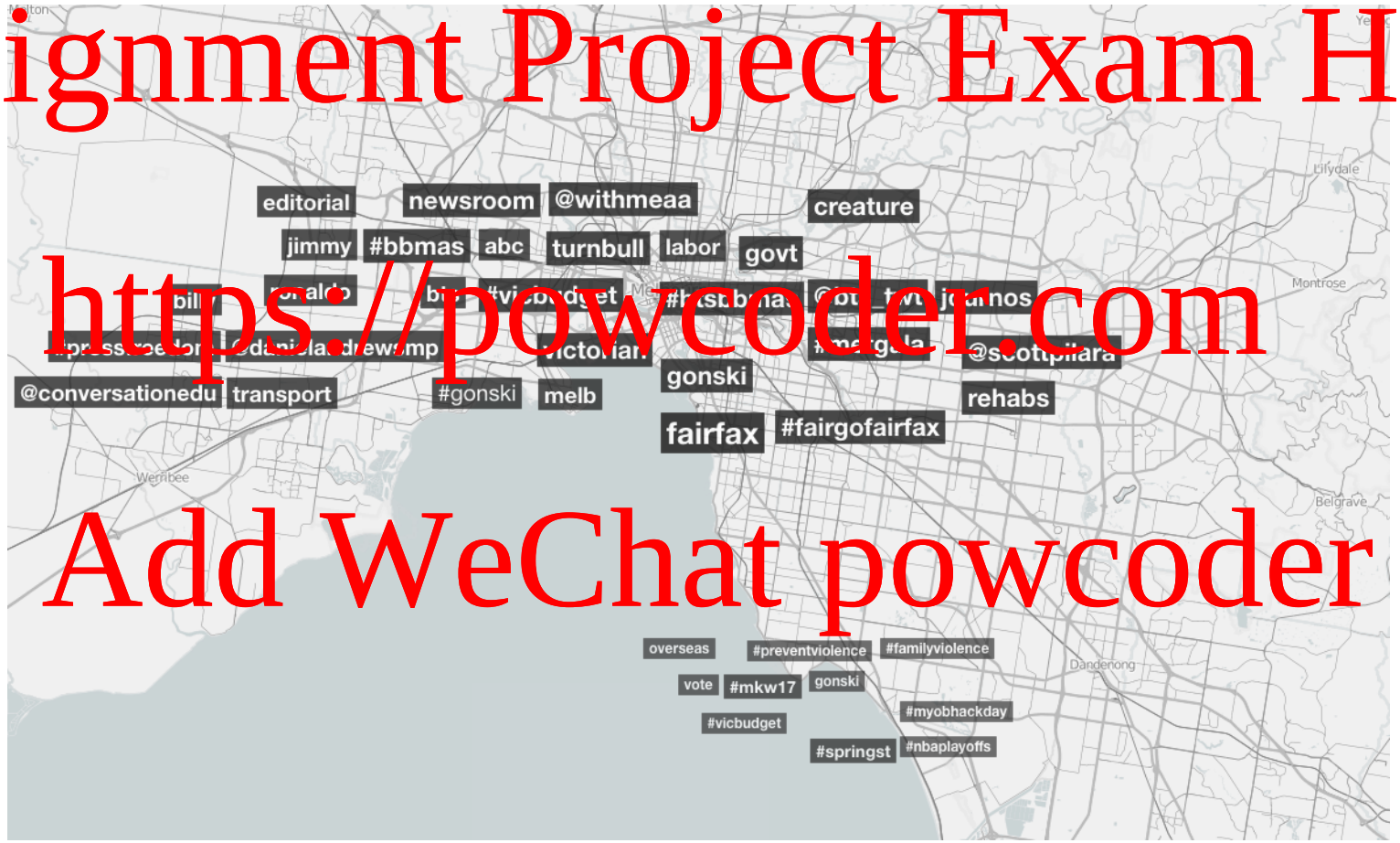
Google's knowledge graph

Data Integration: Application

Map mashup: HousingMaps

Data Integration: Application

Map mashup: TrendMaps shows the latest trend in twitter.



Data Integration: Application

Produc/service comparison portals: TheTracktor, PriceGrabber.

Why Data Integration is Challenging

Why Data Integration is Challenging

- Heterogeneous data
  - Data coming from different sources is often developed independently (e.g., different schema, different objectives)

Why Data Integration is Challenging

- Heterogeneous data
  - Data coming from different sources is often developed independently (e.g., different schema, different objectives)
- Various formats
  - Text, web logs, social networks, sensors, astronomy, genomics, medical records, surveillance, etc.

Why Data Integration is Challenging

- Heterogeneous data
  - Data coming from different sources is often developed independently (e.g., different schema, different objectives)
- Various formats
  - Text, web logs, social networks, sensors, astronomy, genomics, medical records, surveillance, etc.
- Incompatible Taxonomies
  - Different object identity and separate schema
    - Different definitions of a customer, an account, etc.

Why Data Integration is Challenging

- Heterogeneous data
  - Data coming from different sources is often developed independently (e.g., different schema, different objectives)
- Various formats
  - Text, web logs, social networks, sensors, astronomy, genomics, medical records, surveillance, etc.
- Incompatible Taxonomies
  - Different object identity and separate schema
    - Different definitions of a customer, an account, etc.
- Time synchronisation
  - Each source might have a time window that is different from each other.
  - Synchronisation of data collected in different time windows

Why Data Integration is Challenging

- Dealing with legacy data
  - Historical data stored in legacy form, such as IMS, spreadsheets, and other ad hoc structure
  - Combine historical data with modern data

Why Data Integration is Challenging

- Dealing with legacy data
  - Historical data stored in legacy form, such as IMS, spreadsheets, and other ad hoc structure
  - Combine historical data with modern data
- Abstraction levels
  - Different data sources might provide data at different level of abstraction, eg.,
    - suburb level v.s. state level
    - annual v.s. weekly

MONASH University

Why Data Integration is Challenging

- Dealing with legacy data
  - Historical data stored in legacy form, such as IMS, spreadsheets, and other ad hoc structure
  - Combine historical data with modern data
- Abstraction levels
  - Different data sources might provide data at different level of abstraction, eg.,
    - suburb level v.s. state level
    - annual v.s. weekly
- Data Quality
  - Data is often erroneous, and combining data often aggravates the problems. Erroneous data has potentially devastating impact on the overall quality of the integrated data.

Why Data Integration is Challenging

- Dealing with legacy data
  - Historical data stored in legacy form, such as IMS, spreadsheets, and other ad hoc structure
  - Combine historical data with modern data
- Abstraction levels
  - Different data sources might provide data at different level of abstraction, eg.,
    - suburb level v.s. state level
    - annual v.s. weekly
- Data Quality
  - Data is often erroneous, and combining data often aggravates the problems. Erroneous data has potentially devastating impact on the overall quality of the integrated data.
- The number of sources
  - e.g., web-scale integration.

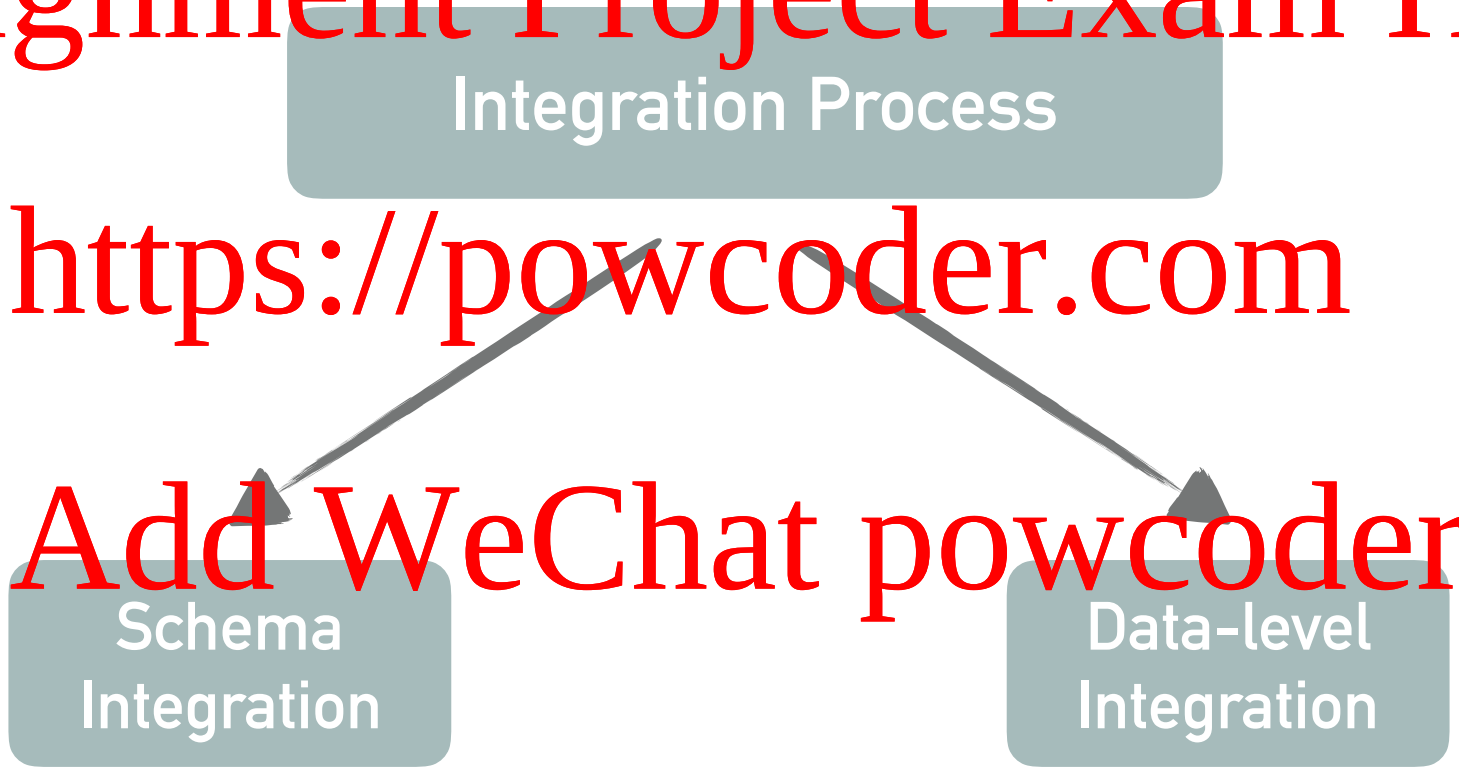
MONASH University

Where can we get the Data?

- Government and political data
- Social data
- Web resources
- Weather Data
- News data
- Preprocessed data for data analysis tasks
- Economics and Fiances

Method: Web scraping

Integration Process



Outline

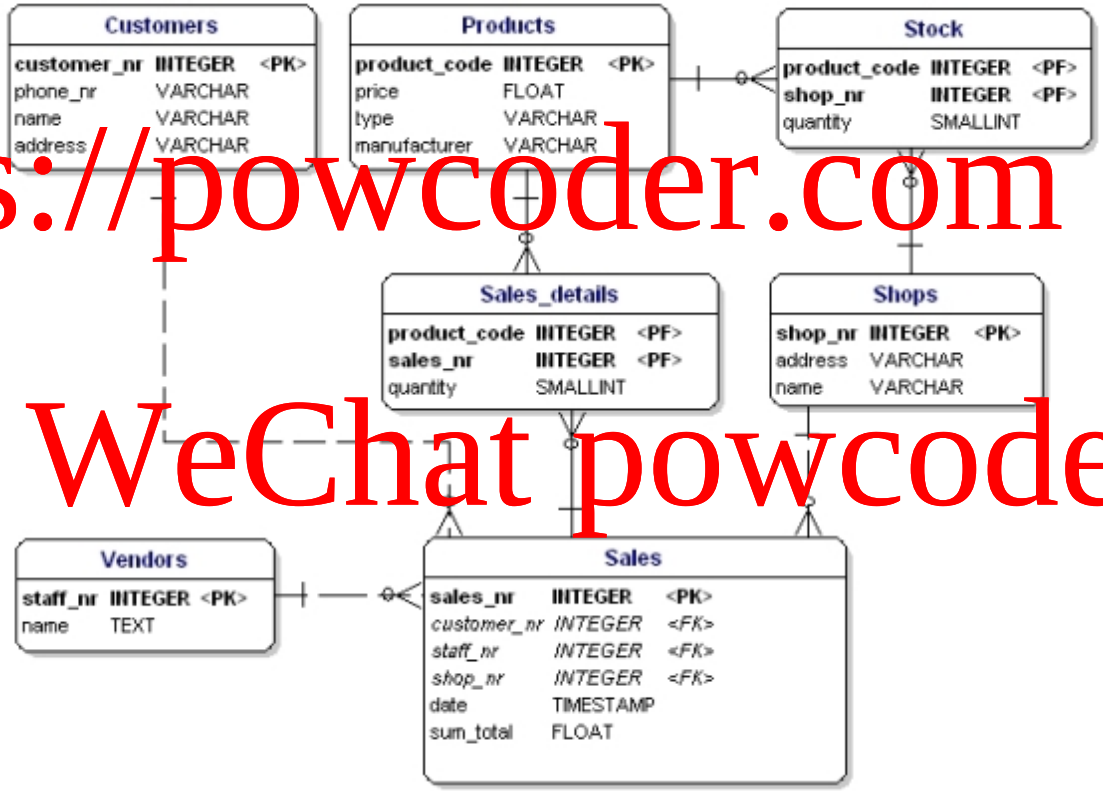
1 Recap

2 Data Integration
- Definition & Application
- Schema Integration

3 Summary

What does the schema look like?

- Relational databases:
  - A schema specifies a set of tables
  - A table contains a set of attributes associated with their data types

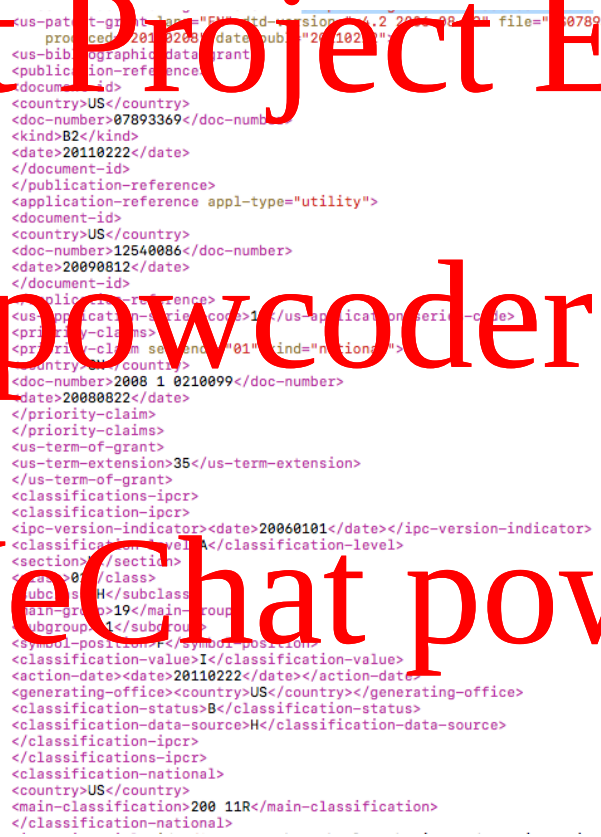


This figure is from http://www.datanamic.com/support/lt-dez005-introduction-db-modeling.html

MONASH University

What does the schema look like?

- Data models like XML and JSON
  - A schema is defined as a set of tags, classes and properties



- Data science
  - A data schema is defined as the representation of the data arrangement, relationships and contents.

Why do we need schema integration?

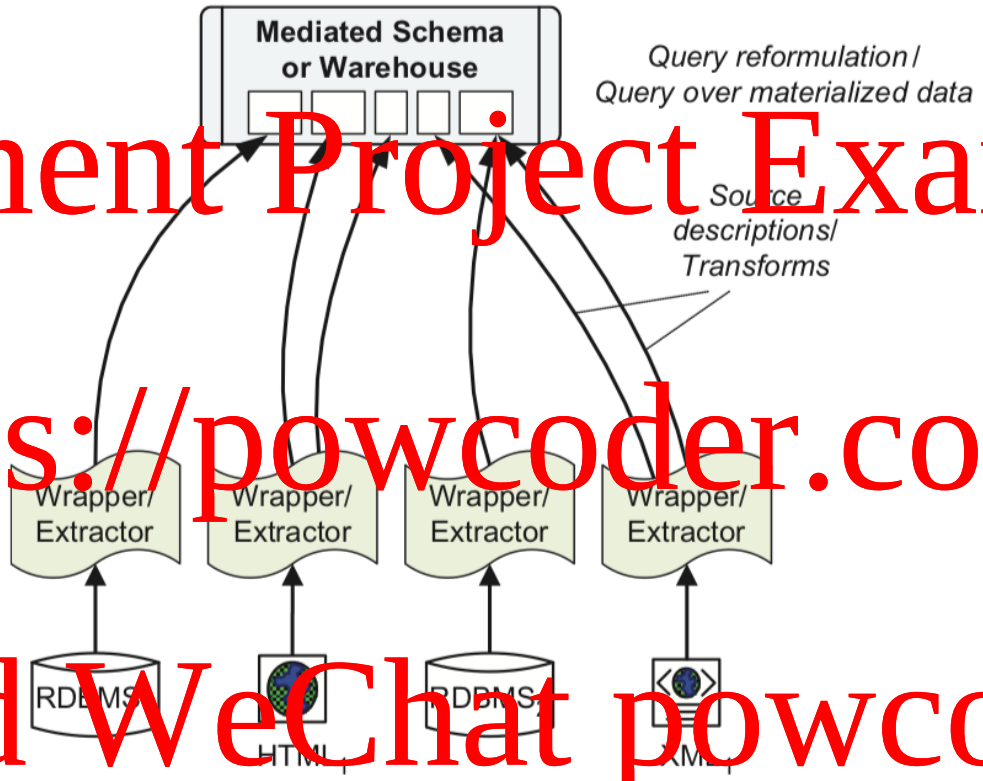


FIGURE 1.4 The basic architecture of a general-purpose data integration system. Data sources can be relational, XML, or any store that contains structured data. The *wrappers* or *loaders* request and parse data from the sources. The *mediated schema* or central *data warehouse* abstracts all source data, and the user poses queries over this. Between the sources and the mediated schema, *source descriptions* and their associated *schema mappings*, or a set of *transformations*, are used to convert the data from the source schemas and values into the global representation.

Schema Mapping

- The linkage between each data source and the mediate schema is done through semantic mapping
  - Specifies how attributes in the sources correspond to attributes in the mediated schema (when such correspondences exist)
  - Specifies how the different groupings of attributes into tables are resolved.
  - Specifies how to resolve schema conflict from different sources

Problems with Schema Integration

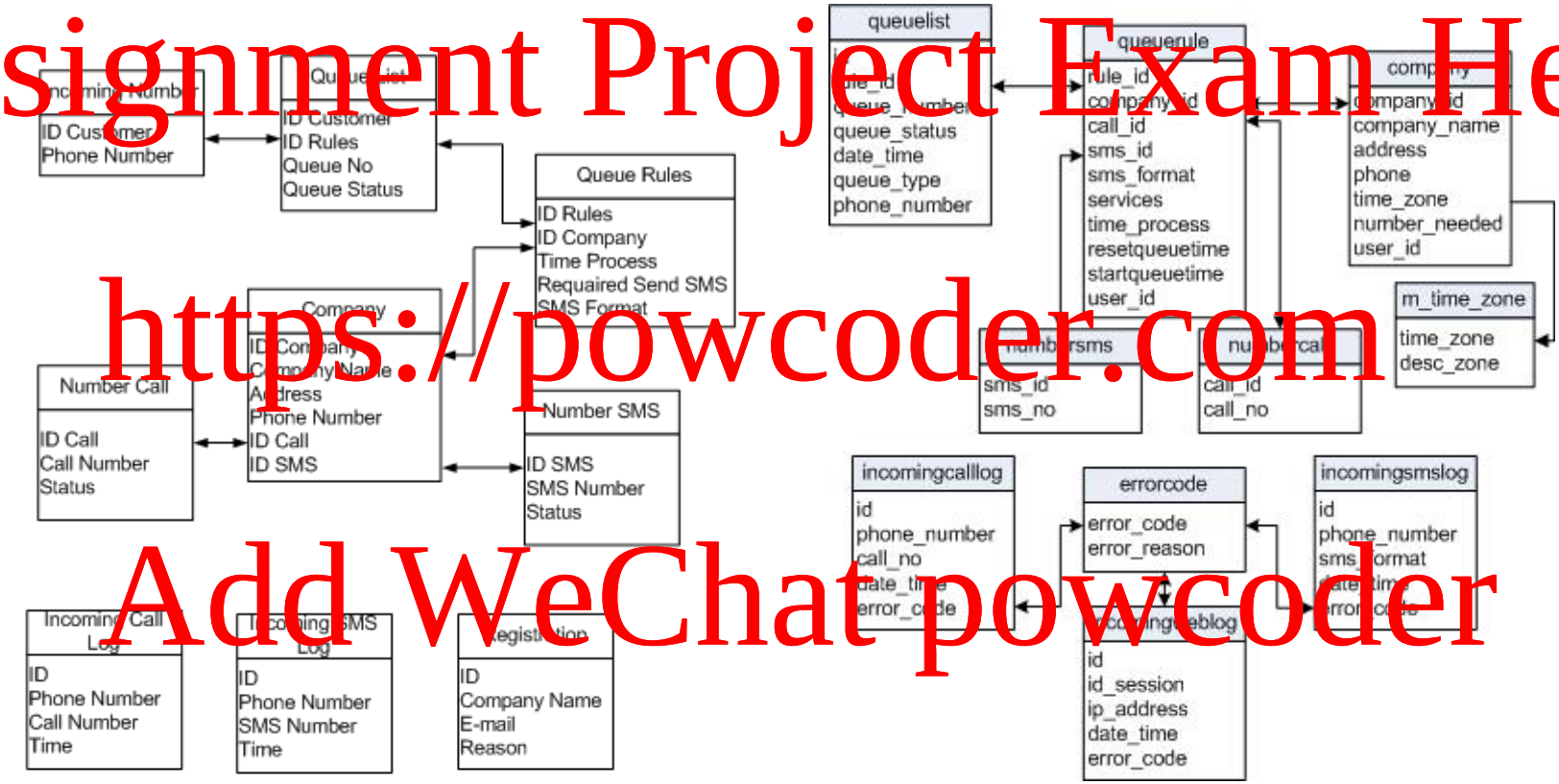
Structure conflicts

- Inconsistencies in the data structure among schemas, which include
  - Different data source origins: Data can be represented in a structure form (e.g., XML, HMTL, JSON, semistructured, or completely unstructured data.
- Inconsistencies among the set of elements inside the different schemas

Problems with Schema Integration

MONASH University

Structure conflicts



Figures are from http://www.urremote.com/untethering-the-queue-2

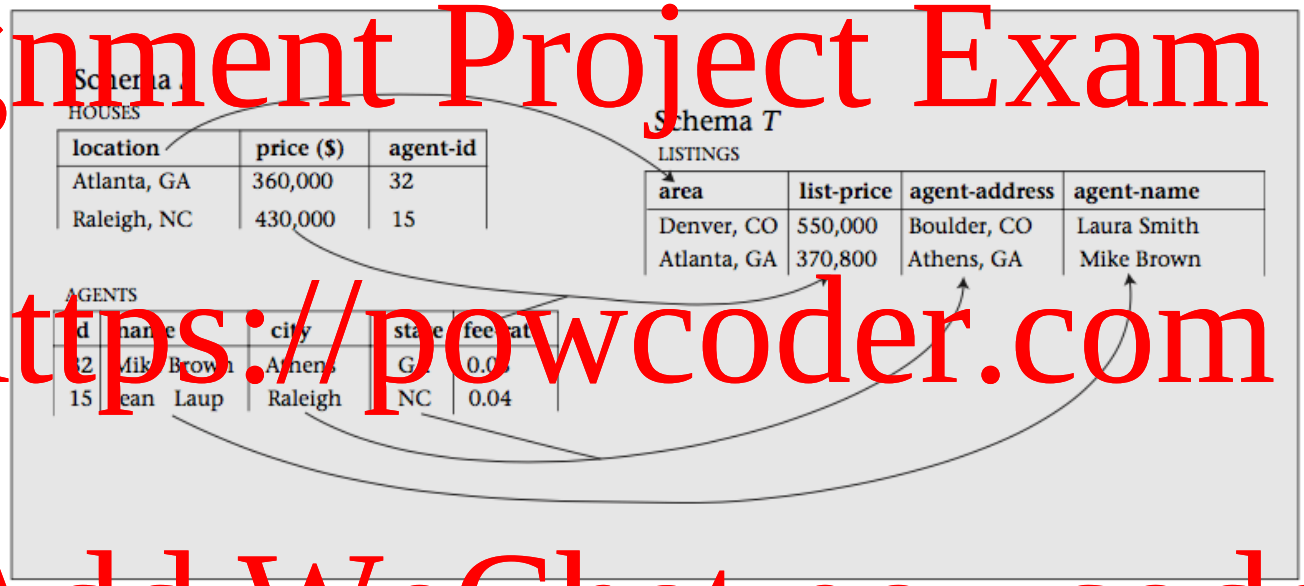
Problems with Schema Integration

Naming conflicts

- homonyms vs synonyms
  - The same name is used for different objects.
  - Different names are used for the same object.
- Examples
  - Homonyms: ID can refer to customer ID, product ID, store ID, etc.
  - Synonyms: Customer ID and Client ID can refer to the same real world object, i.e., customer/client.

Problems with Schema Integration

Naming conflicts



Schema *S*

HOUSES

location	price ($)	agent-id
Atlanta, GA	360,000	32
Raleigh, NC	430,000	15

AGENTS

id	name	city	state	fee (%)
32	Mike Brown	Athens	GA	0.03
15	Jean Laup	Raleigh	NC	0.04

Schema *T*

LISTINGS

area	list-price	agent-address	agent-name
Denver, CO	550,000	Boulder, CO	Laura Smith
Atlanta, GA	370,800	Athens, GA	Mike Brown

Figure 2: The Schemas of Two Relational Databases S and T on Home Listing, and the Semantic Correspondences between Them

Figure is from "Semantic-Integration Research in the Database community" by AnHai Doan and Alon Y. Halevy

- "area" can refer to different real-world entities, e.g., location or square-feet area
- "area" and "location" can refer to the same real-world entity, e.g., the location of the house

Problems with Schema Integration

Entity resolution/conflict resolution

- Different units:
  - Temperature units: Celsius and Fahrenheit
  - Currencies
- Data type heterogeneity
  - Same kind of attributes with different data types
    - phone number can be stored as string in one database and integer in another database
- Value heterogeneity
  - The use of Abbreviations: Professor v.s. Prof, Street v.s. St, Road v.s. Rd

Problems with Schema Integration

Entity resolution/conflict resolution

- Semantic heterogeneity: differences in meaning and interpretation of data values[1]
  - Naming
    - Case sensitivity
    - Synonyms/Homonyms
    - Acronyms
  - Generalisation/Specialisation: one schema may refer to "phone" but the other schema has multiple elements such as "home phone", "work phone" and "cell phone"
- Level of abstraction: different aggregation levels for an attributes
  - Address can be split into multiple fields, street number, street name, suburb, city, post-code, etc.
- Different points of time
  - Fortnight and monthly payment

[1] https://en.wikipedia.org/wiki/Semantic_heterogeneity

Schema Integration: semantic matching

Semantic matching: relates a set of elements in schema S to a set of elements in schema T.

DVD-VENDOR
Movies(id, title, year)
Products(mid, releaseDate, releaseCompany, basePrice, rating, saleLocID)
Locations(lid, name, taxRate)

AGGREGATOR
Items(name, releaseInfo, classification, price)

FIGURE 5.1 Example of two database schemas. Schema DVD-VENDOR belongs to a DVD vendor, while AGGREGATOR belongs to a shopping site that aggregates products from multiple vendors.

Figure is from chapter 5 of "Principles of data integration"

- One-to-One match
  - Movies.title ≈ Items.name
  - Movies.year ≈ Items.year
  - Product.rating ≈ Items.classification
- One-to-Many match
  - Items.price ≈ Products.basePrices × (1 + Locations.taxRate)

Schema Integration: Name-Based Matcher

Name-Based Matcher: compares the names of attributes (or column headers) in the hope that the names convey the true semantics of the elements.

- Split names according to certain delimiters, such as capitalization, numbers, or special symbols.
  - ClientName ⇒ Client Name
  - saleLocID ⇒ Sale Loc ID
- Expand known abbreviations or acronyms
  - loc ⇒ location
  - cust ⇒ customer
  - St ⇒ Street
  - DOB ⇒ Date of Birth
- Expand a string with its synonyms
  - Location ⇒ Address
  - Cost ⇒ Price
- Expand a string with its hypernyms
  - product ⇒ book, DVD, etc.
- Remove articles, propositions, and conjunctions
  - Exclude words like “in”, “at”

Schema Integration: Name-Based Matcher

Name-Based Matcher: compares the names of attributes (or column headers) in the hope that the names convey the true semantics of the elements.

DVD-VENDOR
Movies(id, title, year)
Products(mid, releaseDate, releaseCompany, basePrice, rating, saleLocID)
Locations(lid, name, taxRate)

AGGREGATOR
Items(name, releaseInfo, classification, price)

(a)

name-based matcher: name ≈ ⟨name: 1, title: 0.2⟩
releaseInfo ≈ ⟨releaseDate: 0.5, releaseCompany: 0.5⟩
price ≈ ⟨basePrice: 0.8⟩
(b)

data-based matcher: name ≈ ⟨name: 0.2, title: 0.8⟩
releaseInfo ≈ ⟨releaseDate: 0.7⟩
classification ≈ ⟨rating: 0.6⟩
price ≈ ⟨basePrice: 0.2⟩
(c)

average combiner: name ≈ ⟨name: 0.6, title: 0.5⟩
releaseInfo ≈ ⟨releaseDate: 0.6, releaseCompany: 0.25⟩
classification ≈ ⟨rating: 0.3⟩
price ≈ ⟨basePrice: 0.5⟩
(d)

FIGURE 5.3 (a) Two schemas (reproduced from Figure 5.1); (b)-(c) the similarity matrices produced by two matchers for the above two schemas; and (d) the combined similarity matrix.

Schema Integration: Instance-Based Matcher

Instance-Based Matcher makes use of the data values.

- Rule-based matching method
  - Handcrafted rules exploit schema information such as element names, data types, structures, number of subelements, and integrity constraints.
  - For DVD-vendor database:
    - All possible classification: G, PG, PG-13, R, etc
    - Given a new attribute, if most of its values appear in the list above.
  - Advantages:
    - Relatively inexpensive, do not require training
  - Disadvantages:
    - Cannot exploit data instances effectively (e.g., value format, frequently occurring values, etc.)

Schema Integration: Instance-Based Matcher

Instance-Based Matcher makes use of the data values.

- Learning-based matching method: learning techniques that can exploit both schema and data information.
  - Classification-based methods
  - (semi-)automated but Needs training

Example 5.6

If s_i is address, then positive examples may include "Madison WI" and "Mountain View CA," and negative examples may include "(608) 695 9813" and "Lord of the Rings." Now suppose that element t_j is location and that we have access to three data instances of this element: "Milwaukee WI," "Palo Alto CA," and "Philadelphia PA." Then the classifier C_i may predict confidence scores 0.9, 0.7, and 0.5, respectively. In this case we may return the average confidence score of 0.7 as the similarity score between s_i = address and t_j = location.

Summary

- Recap of outlier detection
- Data integration
  - Schema integration
    - Naming conflict
    - Structure conflict
    - Entity resolution
    - Semantic mapping