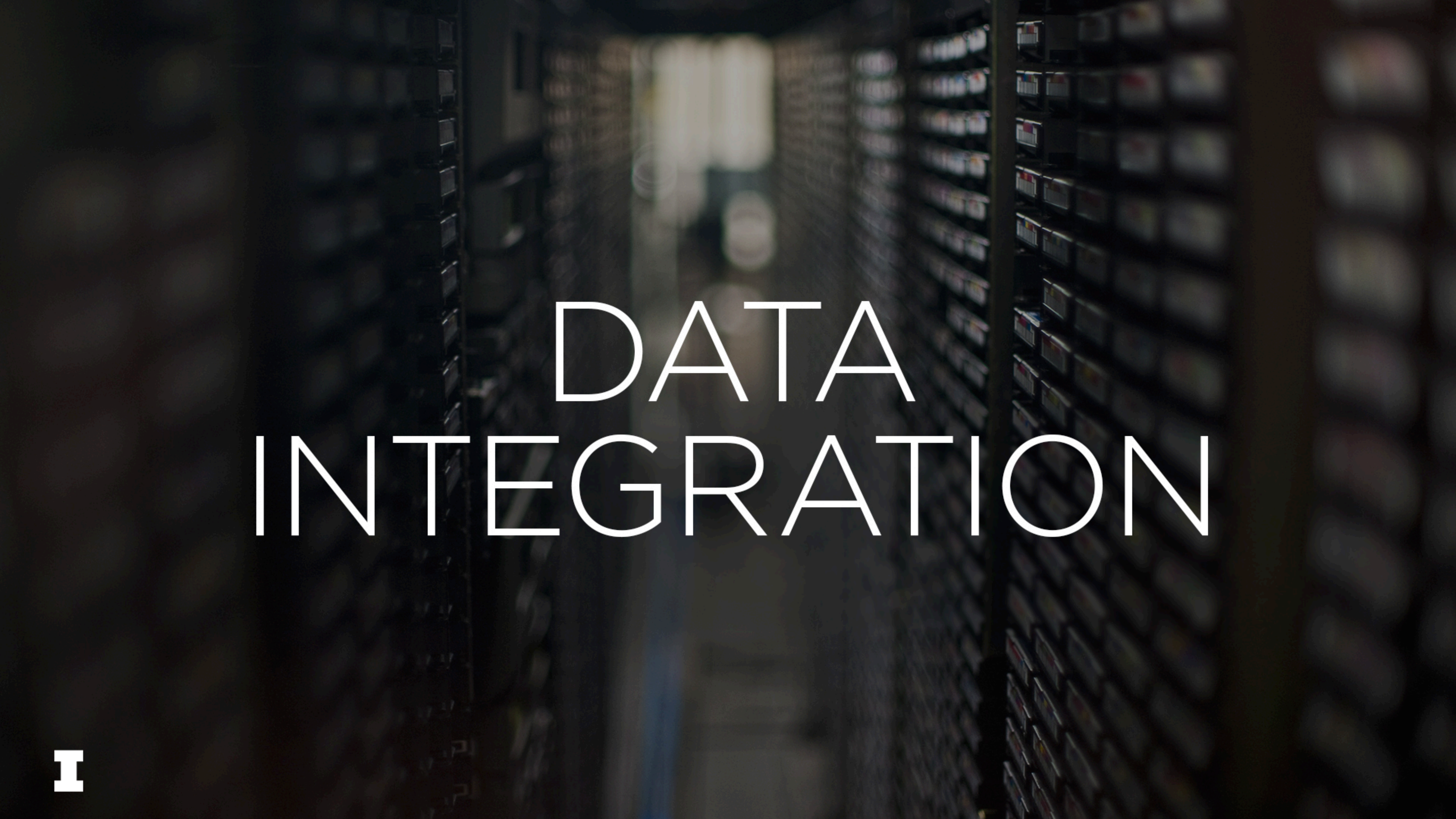
DATA INTEGRATION



1. Data cleaning, Data integration

- Data cleaning
- Data integration
- Heterogeneity
- Federation vs derived combinations

Data Cleaning

Data cleaning is a general term used colloquially to describe preparing data for analysis. When a single schema is involved the phrase typically suggests dealing with:

- duplicate records
- values that are missing, out of bounds, or inconsistent
- data typing errors or inconsistencies
- data entry errors
- attribute interpretation errors
- variation in null handling
- misapplication of standards
- inadequate normalization
- missing or inadequate schemas
- missing relationships
- failures of referential integrity or other constraints
- failure to match schema (e.g. as identified by a formal grammar or XML parser)

Data cleaning is profoundly important – without it data cannot be used reliably, or at all.
It also is a major industry expense and consume much staff time.

When multiple schemas (or disparate instances) are involved the data preparation task is *data integration*

Data Integration

Data integration:

“. . . combining data residing in difference sources and providing users with a unified view. . .”

[Lenzerini 2002]

Data model relationships

Entities, Relationships

Conceptual models, UML or ER models, ontologies

Schemas: ER, UML . . .

Schemas: RDFS, OWL . . .

Conceptual Level

Logical Level

Relations

e.g., Relational databases

Schemas: column and key descriptions

Trees

e.g., XML Documents

Schemas: grammars (e.g. DTDs),

Triples

e.g., RDF triple stores

Schemas: serialization descriptions.

Physical Level [or: Storage]

[files, records, delimiters, data structures, indexes, etc.]

Why is it important?

Real world problems are profoundly interdisciplinary, solving them requires integrating diverse data from multiple sources

> e.g., an effective response to an impending natural disaster can require understanding how many people will be affected, hospital location and capacity, and transportation routes, and so on.
>
> Many different disparate databases will need to be accessed (demographic, meteorological, geographical)
>
> And, most importantly, the data elements will need to be related: concentrations of people connected to transportation routes, the storm path, hospital capacity, etc.
>
> *If this cannot happen or cannot happen reliably, or efficiently,*
> *much valuable data will be useless, opportunities lost, and problems unaddressed.*

Why is it hard?

When datasets are developed by different communities and for specific purposes; integrating them with other datasets is often not anticipated.

(and accommodation would be hard in any case)

These datasets often use different data models, schemas, and encodings that are very hard to related to each other, even when describing the same real world feature.

But unless common concepts can be found to connect data across datasets, and to either standardize or refactor related data elements, integration is impossible.

The obstacle to data integration is therefore: *heterogeneity*.

Kinds of heterogeneity

*Relatively easy
**Often difficult
***Usually very difficult

Encoding heterogeneity *

Different mappings from bitstreams into bytes, characters, numbers, or other logical units

Syntax heterogeneity *

Different data description languages for the same model type: e.g. RDF/XML vs N3

Model heterogeneity **

Different model type; e.g., relations vs entities/relationships

Representational heterogeneity **

Different modeling choices within a model type; e.g. relationships vs entities.

Semantic heterogeneity ***

Different conceptualization of similar domain features

Processing heterogeneity **

e.g. different maintenance and update regimes

Policy heterogeneity **

e.g. different privacy and security rules, varying ownership and licensing, etc.

Adapted from Bertram Ludäscher, Kai Lin, Shawn Bowers, Efrat Jaeger-Frank, Boyan Brodaric, Chaitan Baru, "Managing scientific data: From data integration to scientific workflows", *Geoinformatics: Data to Knowledge, 2006*; and Amit Sheth. "Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics". In M. Goodchild, M. Egenhofer, R. Fegeas,and C. Kottman, editors, *Interoperating Geographic Information Systems* Kluwer, 1998.

Two general approaches to integration: federation vs derivation

Federation

For relevance: Standardized metadata attached to each dataset can be used to determine its relevance, indicating spatial and temporal location and general nature of content; this facilitates discovery.

For queries: Views on and queries against multiple databases are supported by mappings to a mediating meta-schema.

Derivation

A single dataset is derived from multiple sources and governed by a single schema.
[compare Extract, Transform, and Load (ETL) data warehouses]

In either case heterogeneity remains a huge challenge

2. Managing Heterogeneity

Managing encoding heterogeneity
Managing syntax heterogeneity
Managing model heterogeneity
Managing representational heterogeneity
Managing semantic heterogeneity
Managing processing heterogeneity
Managing policy heterogeneity

Data integration:

“. . . combining data residing in difference sources and providing users with a unified view. . . [Lenzerini 2002]

Kinds of heterogeneity (again)

*Relatively easy
**Often difficult
***Usually very difficult

Encoding heterogeneity *

Different mappings from bitstreams into bytes, characters, numbers, or other logical units

Syntax heterogeneity *

Different data description languages for the same model type: e.g. RDF/XML vs N3

Model heterogeneity **

Different model type; e.g., relations vs entities/relationships

Representational heterogeneity **

Different modeling choices within a model type; e.g. relationships vs entities.

Semantic heterogeneity ***

Different conceptualization of similar domain features

Processing heterogeneity **

e.g. different maintenance and update regimes

Policy heterogeneity **

e.g. different privacy and security rules, varying ownership and licensing, etc.

Adapted from Bertram Ludäscher, Kai Lin, Shawn Bowers, Efrat Jaeger-Frank, Boyan Brodaric, Chaitan Baru, "Managing scientific data: From data integration to scientific workflows", *Geoinformatics: Data to Knowledge, 2006*; and Amit Sheth. "Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics". In M. Goodchild, M. Egenhofer, R. Fegeas,and C. Kottman, editors, *Interoperating Geographic Information Systems* Kluwer, 1998.

Managing encoding heterogeneity

Variations in character encodings (or other logical atoms), byte interpretation, bit stream management, magnetic tape formatting etc. have created many problems.

- How many bits in the octet are coding? Is the byte to be read left to right or right to left? Are there control bytes in the bit stream? What are line end and carriage return characters?
- If sets of logical atoms are not identical there is no complete 1:1 conversion and a variety of problematic resolutions -- such as conflation, unification, omission, and named references -- must be considered:

    And if the nature of the logical atoms may be unclear is that (“ü”) a diaeresis or umlaut?

But thanks to common tools and conventions (particularly Unicode) encoding is less of an issue now.

To be sure the mapping from bit streams to characters is extremely complex.

See: *Unicode Character Encoding Model,* Ken Whistler & Mark Davis (Unicode Technical Report #17)

But global implementation of the Unicode standard is robust and complete.

Syntax Heterogeneity

Converting from one syntax to another (within the same general data model) is a very common data curation activity

- For instance, from SGML to XML or JSON or from RDF/XML to N3 or turtle.

Schemas also may need to be converted from one schema language to another

- For instance from XML/DTD schema language to (W3C) XML/Schema or XML/RelaxNG.

Syntax conversions can be challenging, but often can be accomplished without loss of information, and for common conversions there exist effective tools.

There can be problems though, particularly at the schema level. For example,

- XML/Schema has more data typing options than XML
- SGML/DTDs have grammar constraints difficult to implement in XML/DTDs

Model Heterogeneity

The same information can be expressed using completely different *types* of data models (e.g. relations vs trees, vs ontologies).

Conversion from one data model to another is often needed, for data integration, preservation, exchange, to use particular tools and applications.

Conversions such as these are particularly important in data curation

- Exporting XML files from a relational data base
- Using a relational database to store XML documents
- Converting relations to RDF graphs, and RDF graphs to relations

Methods for these conversions are exist, but they can be challenging to implement and may not provide the same functionality, data typing, or constraints

In addition they may not be “natural”: imagine storing an XHTML web page as relations!!

Most importantly: a new schema, for an equivalent but different data model, will need to be developed.

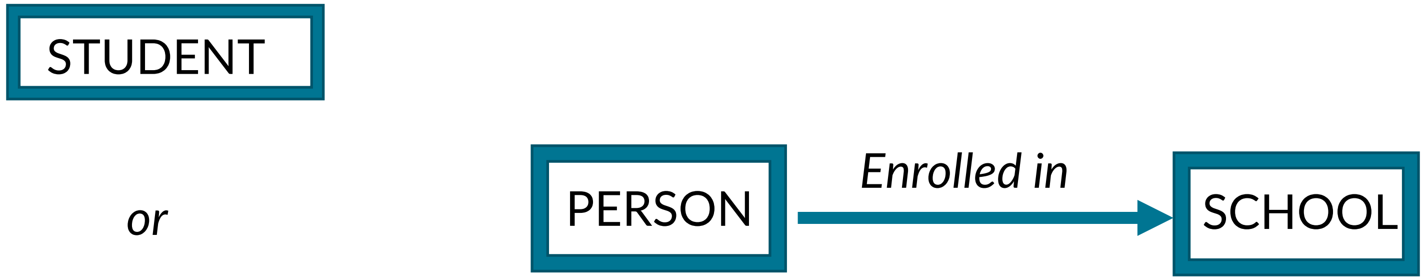
Representational Heterogeneity

Here are two classic problems for representational heterogeneity:

For XML: element or attribute?

`<chapter>` or `<div type=chapter>`?

For ontologies: entity or relationship?



The pros and cons of expressively equivalent modeling choices can be difficult to evaluate.

Semantic Heterogeneity

This is the most challenging heterogeneity. We take in up in the next video.

Processing Heterogeneity

Some examples

Modification differences (instance level)

- One dataset is updated weekly, another monthly, another when observations change
  Related: lack of atomicity
- One dataset is undergoes integrity and validation weekly, another after every update
- One dataset documents provenance of modifications, another does not

  These are particularly problems for federated datasets, but also for derived combined datasets, especially when they are routinely and perhaps automatically derived (ETL).

Modeling differences

Datasets may undergo schema changes without notification or coordination.

Metadata differences

Datasets may undergo schema changes without notification or coordination.

Policy Heterogeneity

Information may be subject to different restrictions related to ownership, privacy, libel, security, disclosure, etc.

- As a dataset may have originated in one country, be about the citizens of another, be owned by a third, be stored in a fourth, and be accessed by users in many others legal and regulatory requirements may be complex, even inconsistent.

  And another to be integrated with the first, may have…etc.

- **Data derived by analysis** from input data from multiple sources adds additional complexity.

(Imagine: a deduction from data from multiple sources)

3. Schema Integration

Definition

Representational challenges

Semantic challenges

Homonyms, synonyms, overlaps, missing relationships, generality . . .

Definition: Schema integration

Schema Integration

> *“the process of merging several conceptual schemas into a global conceptual schema that represents all the requirements of the application”.* (Batini et al)

- If the approach is federation a federating schema is created and mapped to individual schemas.
- If the approach is a derived combination dataset a single schema is created for the derived dataset.

Schema integration usually takes place at the conceptual level (as indicated above).

Or at the logical level (e.g., relational schemas or XML schemas).

Or even across levels (e.g. an XML schema and an ontology);
(particularly when the distinction between logical and conceptual level is unclear).

Two kinds of schema integration problems

Representational (aka *structural*)

Different choices about modeling constructs and integrity constraints

Semantic

Synonyms, homonyms, missing relationships, different but related concepts, etc

In what follows we focus on naming problems.

Simple naming problems: *synonyms* and *homonyms*

For example:

1) Two schemas use different names (F and G) for the same properties
2) Two schemas use the same name (F) for different properties

Once it is determined that 1) or 2) is the case the remedy, either in a meta-schema for federation, or the new comprehensive schema for derived combination is obvious.

  What is hard is determining that there is in fact a synonym/homonym problem.

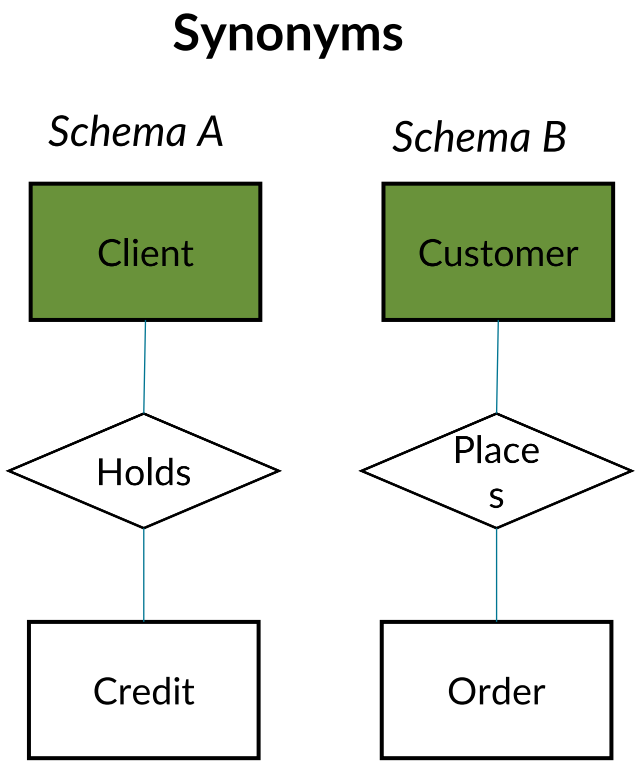
    This might involve examining documentation, interviewing schema designers and data collectors, etc.

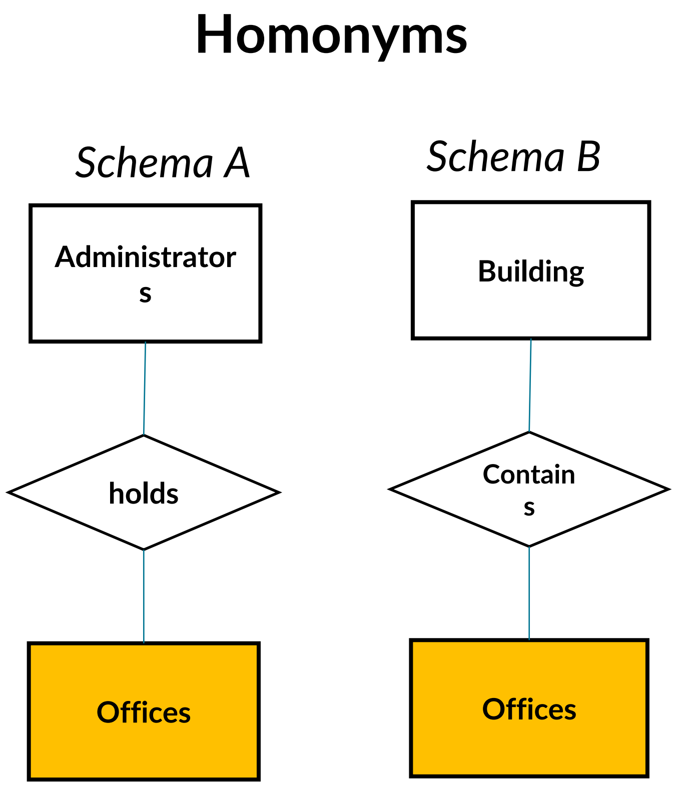
    And, of course examining the distribution of values in the dataset instances themselves

    [In all cases using background knowledge of the domain is essential.]

  But none of these are decisive.

Synonyms, Homonyms





Identifying “client” and “customer” as the same entity type in different schemas is called “schema matching”.

Not so simple naming problem: conceptual overlaps

A more challenging problem is when there is a conceptual overlap.

For instance:

- Schema A has an attribute *name* interpreted as applying to nicknames and legal names, but not to aliases
- Schema B has an attribute *name* interpreted as applying to legal names and aliases, but not nicknames

Like other earlier problems overlap may be difficult to discover.

But it is also difficult to remedy without information loss.

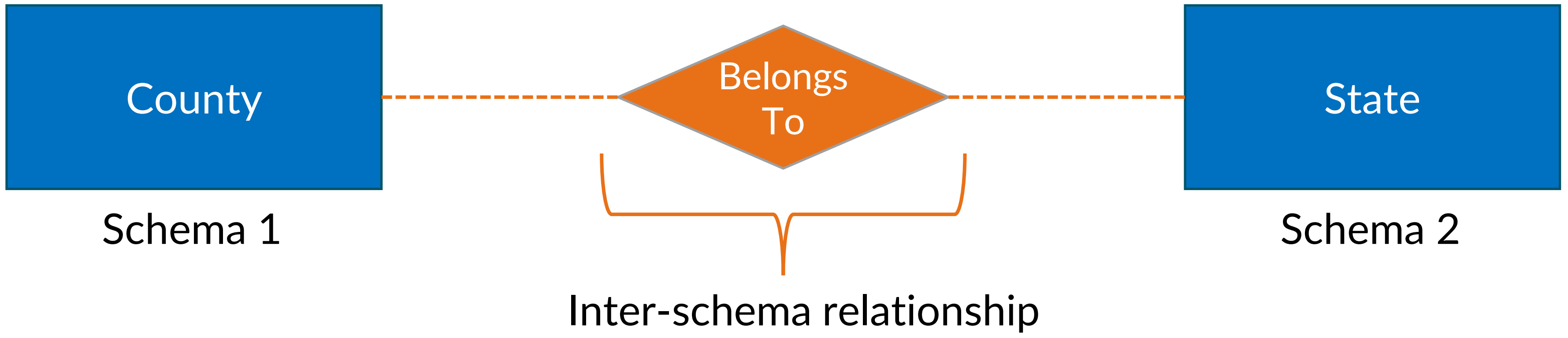
What are the options?

Inter-schema relationships

Suppose one schema has the entity type *county*, but not the entity type *state*, another has the entity type *state* but not the entity type *county*.

This limits the information we can extract from these datasets.

Adding *belongs_to* to the integrating schema and a mapping of counties to states will allow instance data in the first to be connected to states as well as counties.



NB: These relationships are not represented in either schema; they must be inferred.

Generality variation

Schema A has an entity type *vehicle* but no way to indicate kinds of vehicles.

Schema B has an entity type for *motorcycle* but no entity type for *vehicle.*

[Motorcycle is, or course, a proper subset, and so subclass, of *vehicle]*

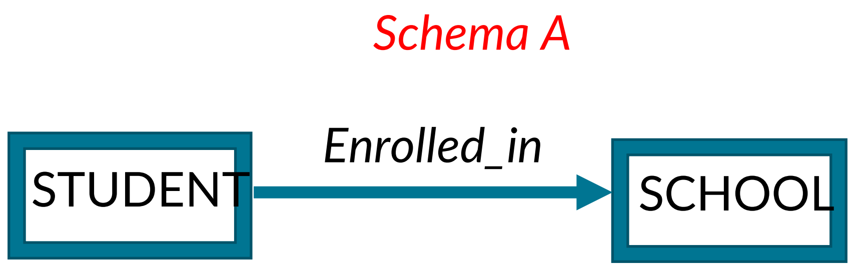
A simple solution would be to generalize *motorcycle* to *vehicle*, but this would lose information.

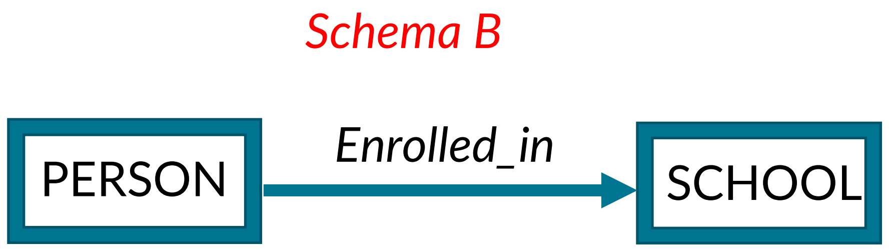
Retaining both *vehicle* and *motorcycle* with *motorcycle* as a subclass of *vehicle* retains all the information we have, similarly a *kind* attribute with <u>motorcycle</u> value could be used as well.

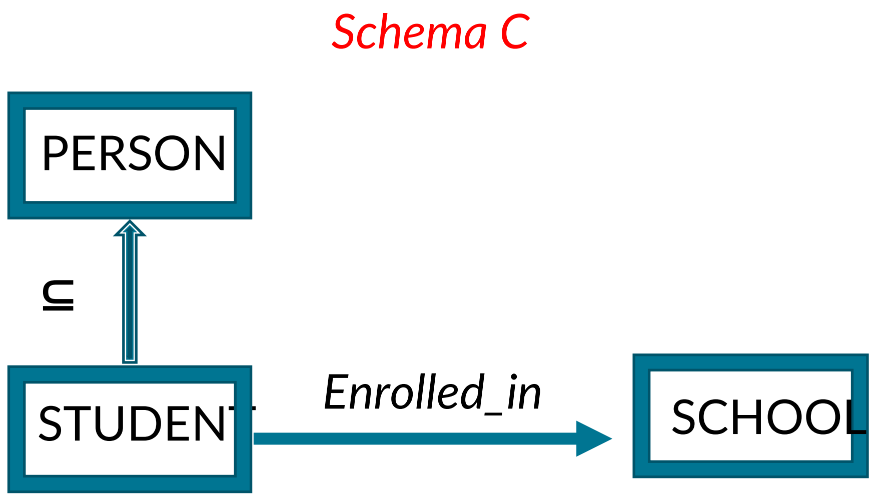
[No information is lost, but in both cases an unevenness in instance descriptions results.]

As always the basic strategy is the same whether for federations or derived combinations.

Entities, relationships, subclasses





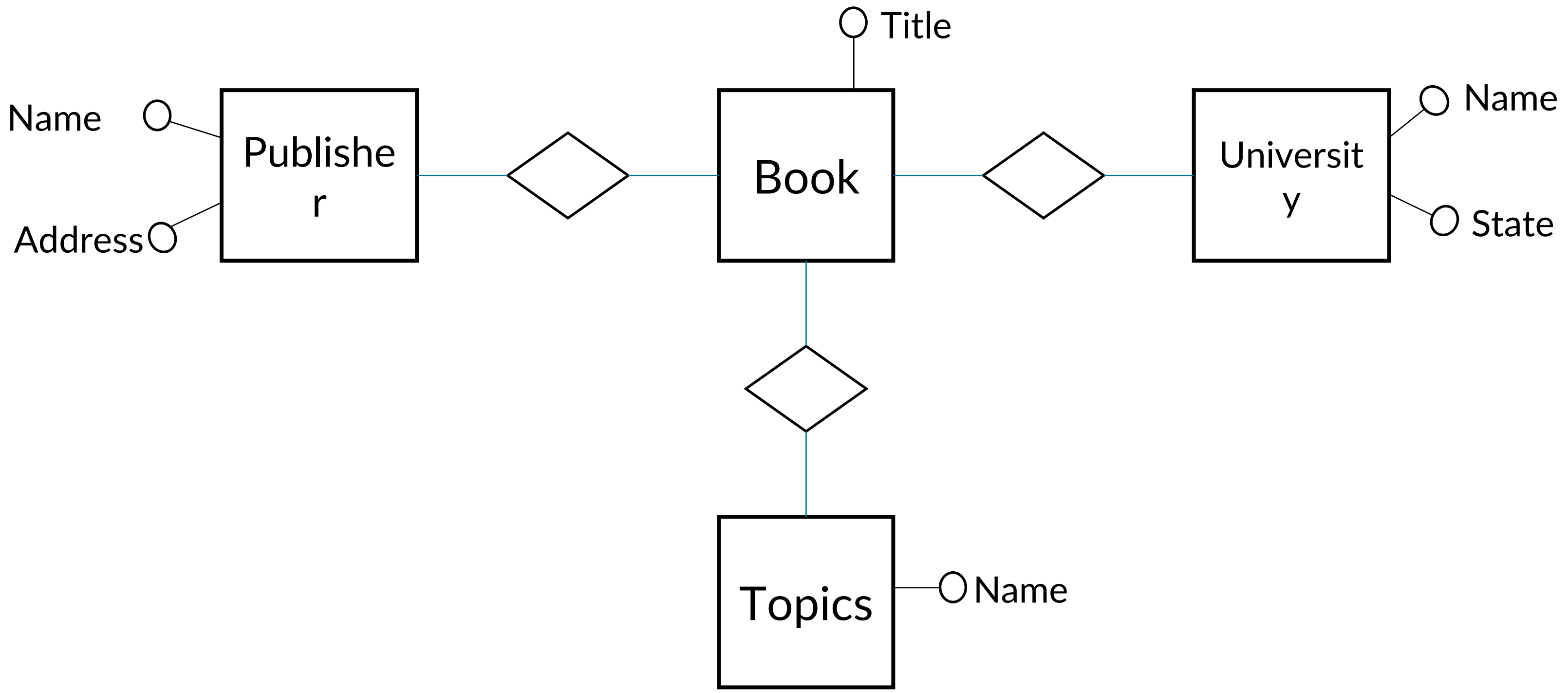


4. Schema integration: example

Integrating two conceptual schemas.

Books, again

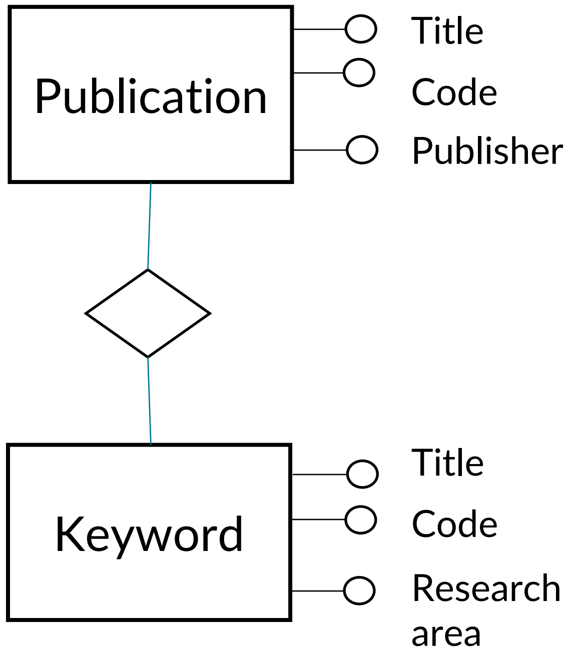
In one model…



Books are published by publishers. They are adopted by universities. They refer to topics.

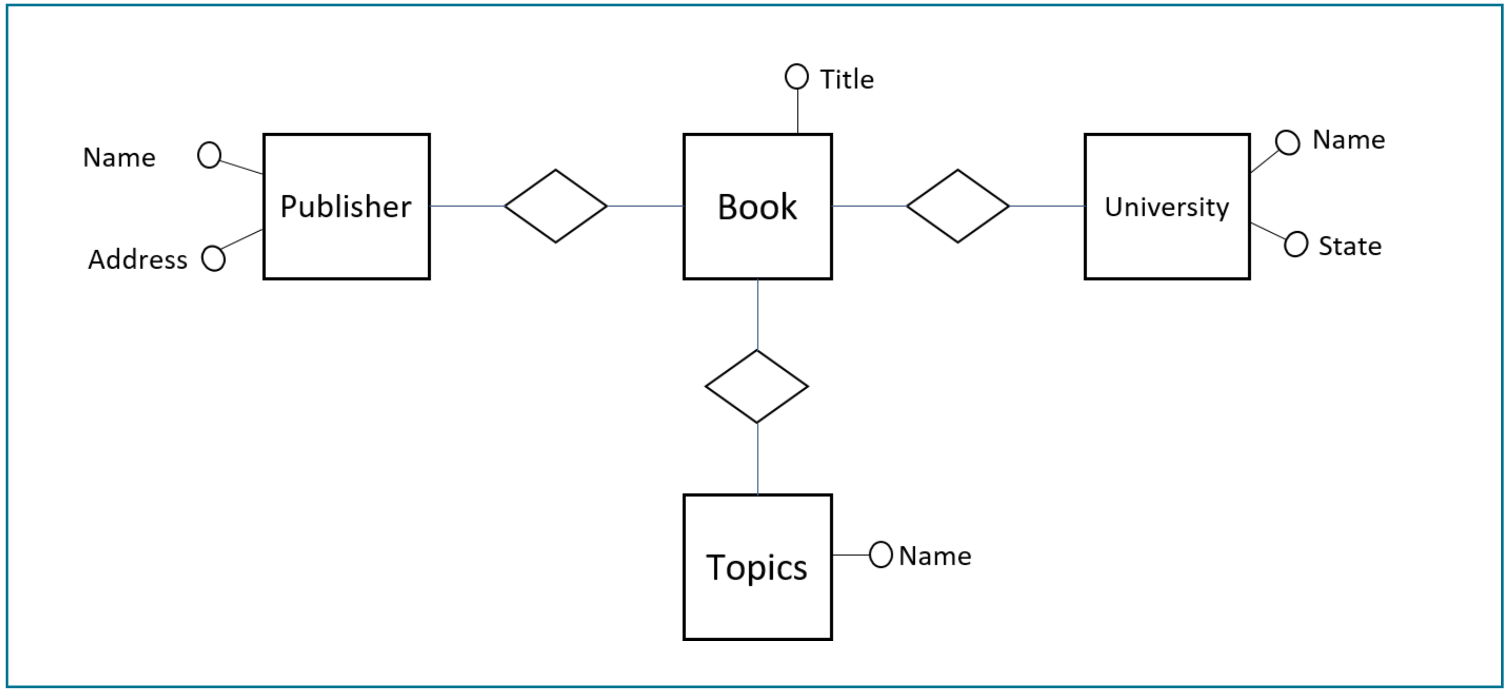
Alternative model of same (?) domain

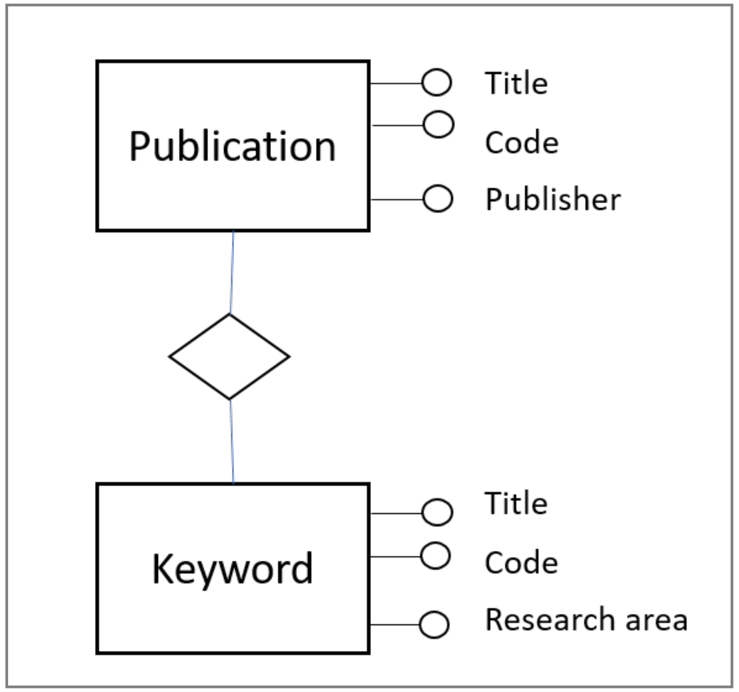
In another model…



There are publications of different types. Each publication has a list of keywords.

How to integrate these two conceptual schemas?





We observe such thing as:

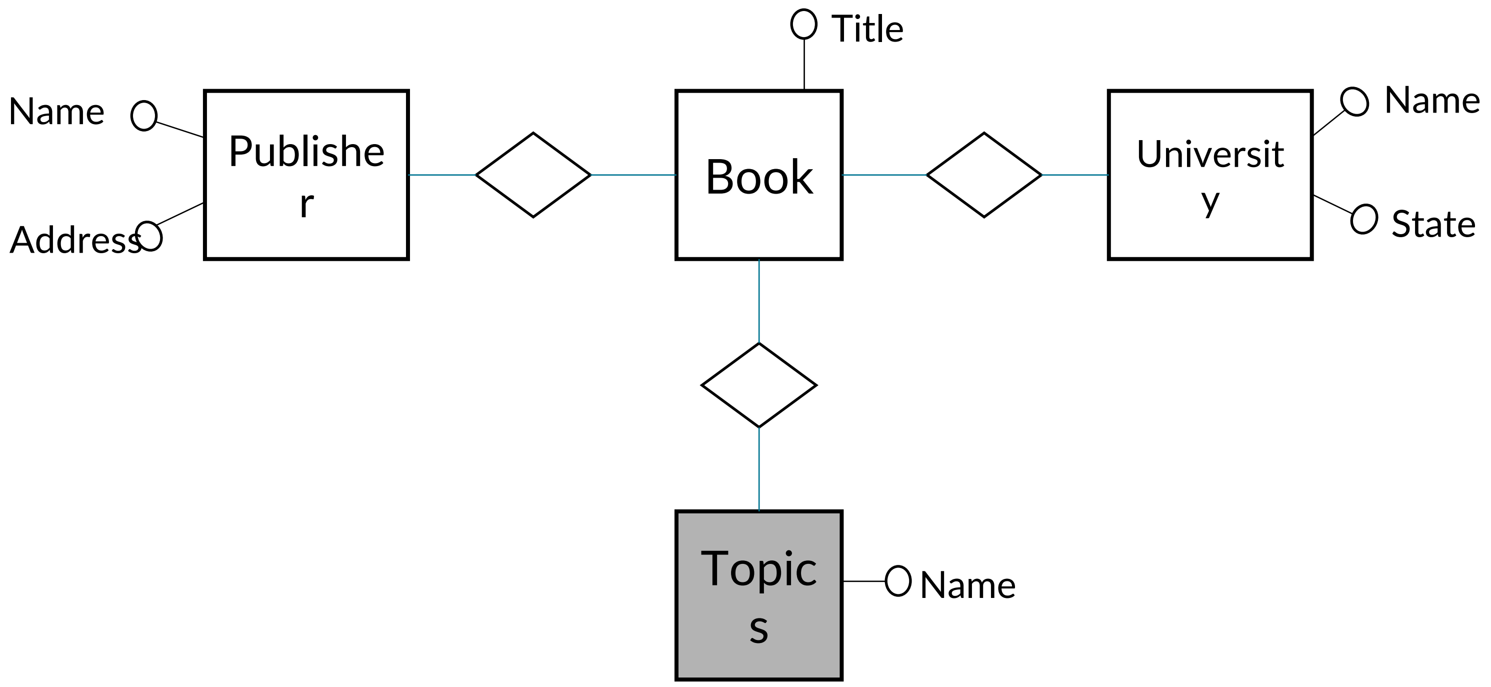
Topics and *Keyword* appear to refer to the same thing

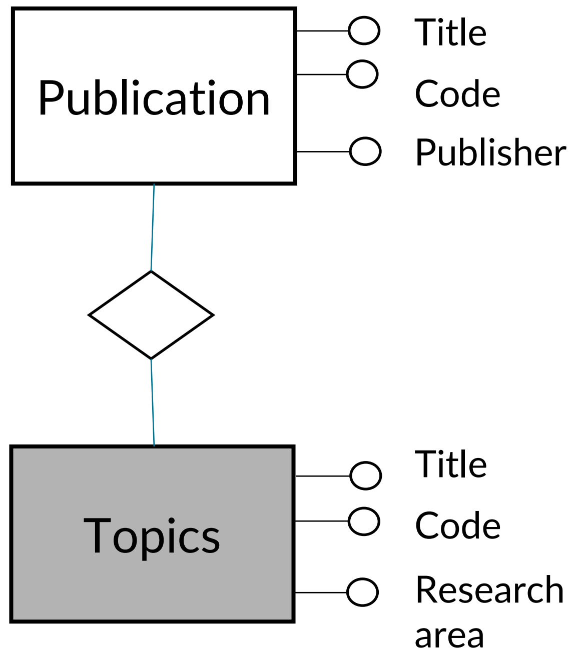
A *Book* /is kind of/ *Publication* (BOOK ⊂ PUBLICATION)

Publisher is an entity in one schema; an attribute in the other

Batini, C., Lenzerini, M., and Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. ACM Computing Surveys 18(4). doi>10.1145/27633.27634

1. Use "Topics" for "Keyword" as well

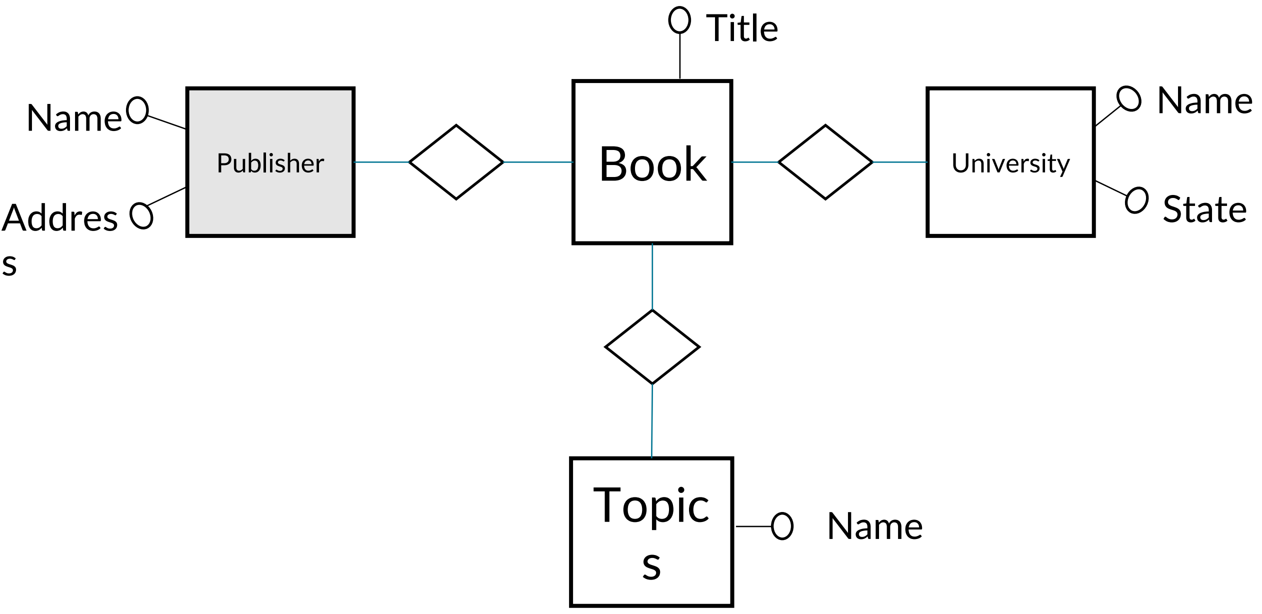


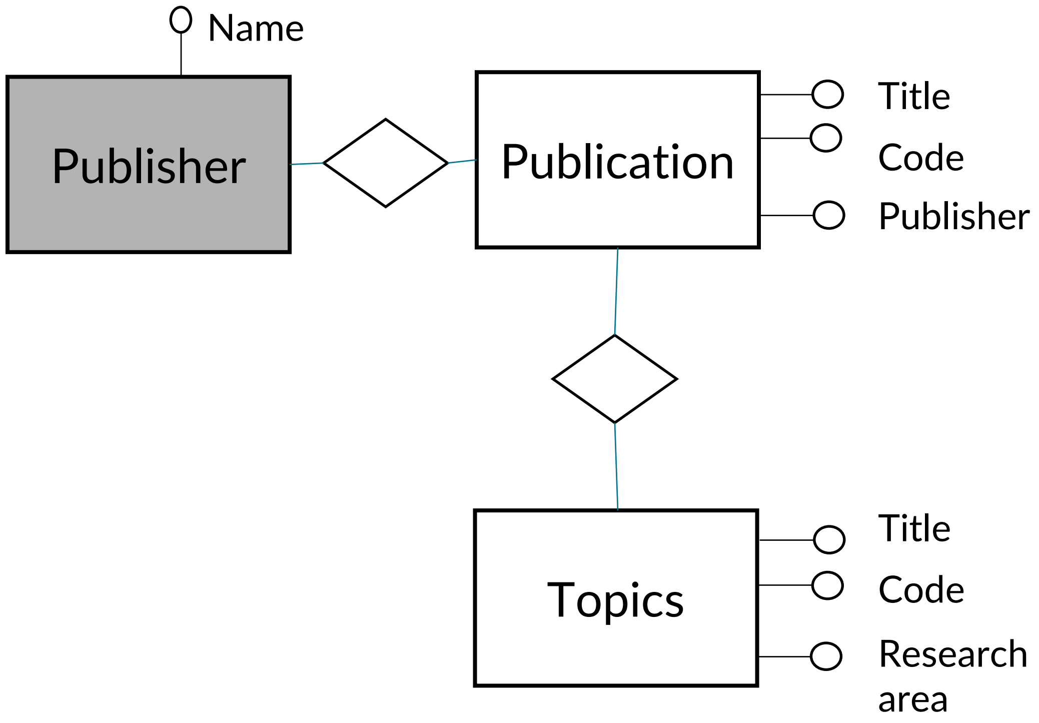


"Topics" and "Keyword" appear to be synonyms

Batini, C., Lenzerini, M., and Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. ACM Computing Surveys 18(4). doi>10.1145/27633.27634

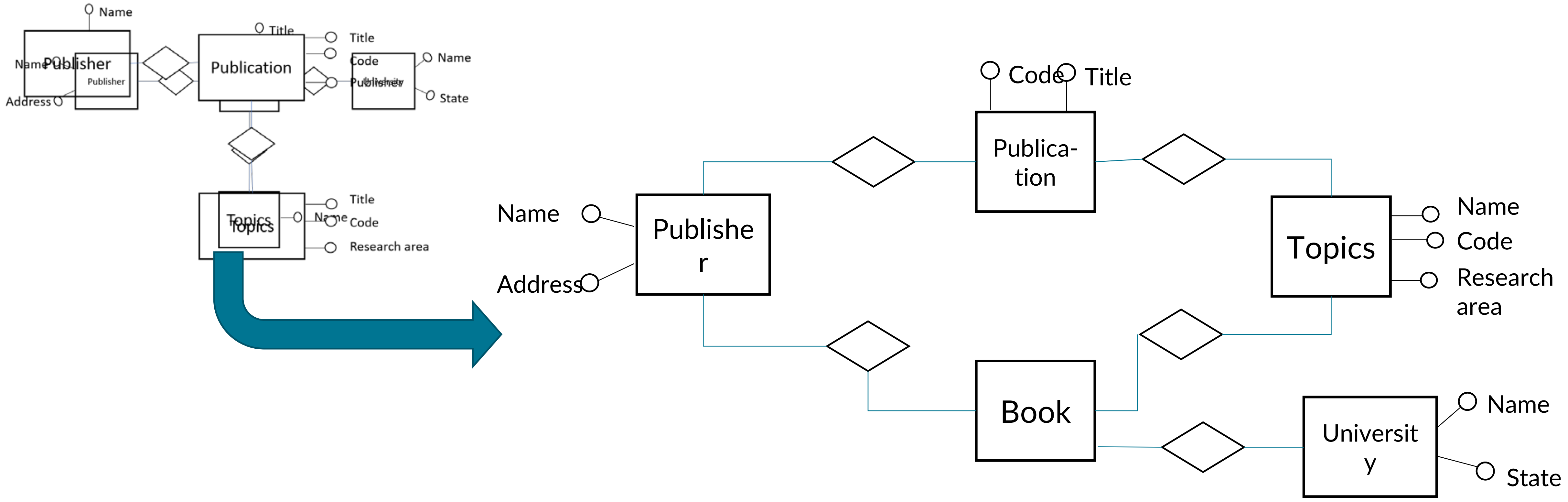
2. Make “Publisher” an entity





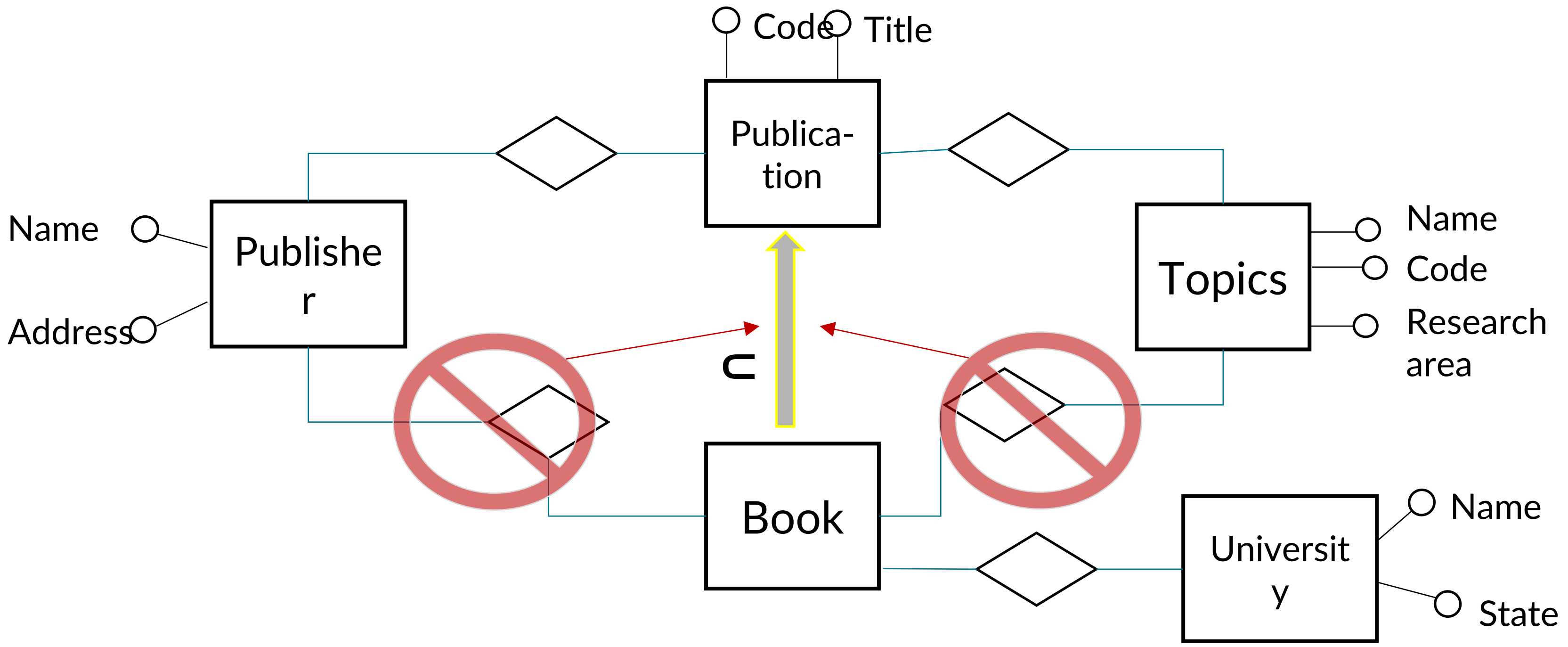
In one model, “Publisher” is an entity; in the other, an attribute

3. Combine schemas



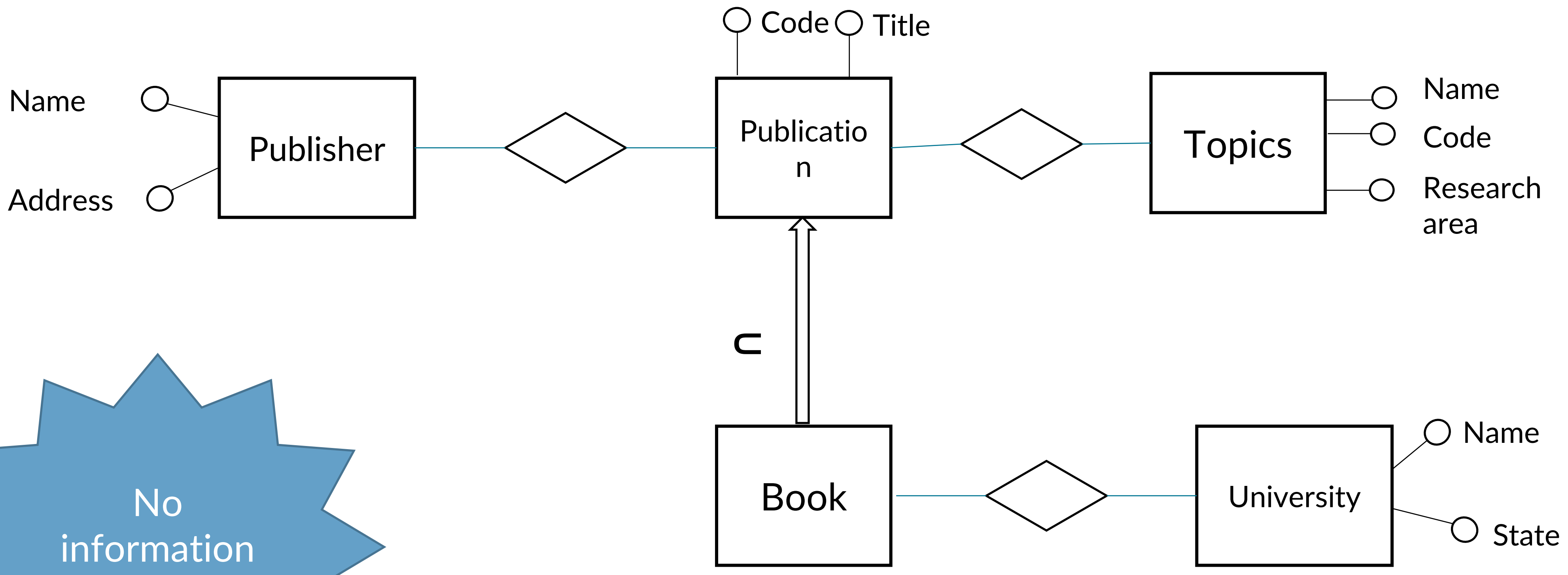
Batini, C., Lenzerini, M., and Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. ACM Computing Surveys 18(4). doi>10.1145/27633.27634

4. Create subset relationship



"Publisher" is an entity; an attribute in the other

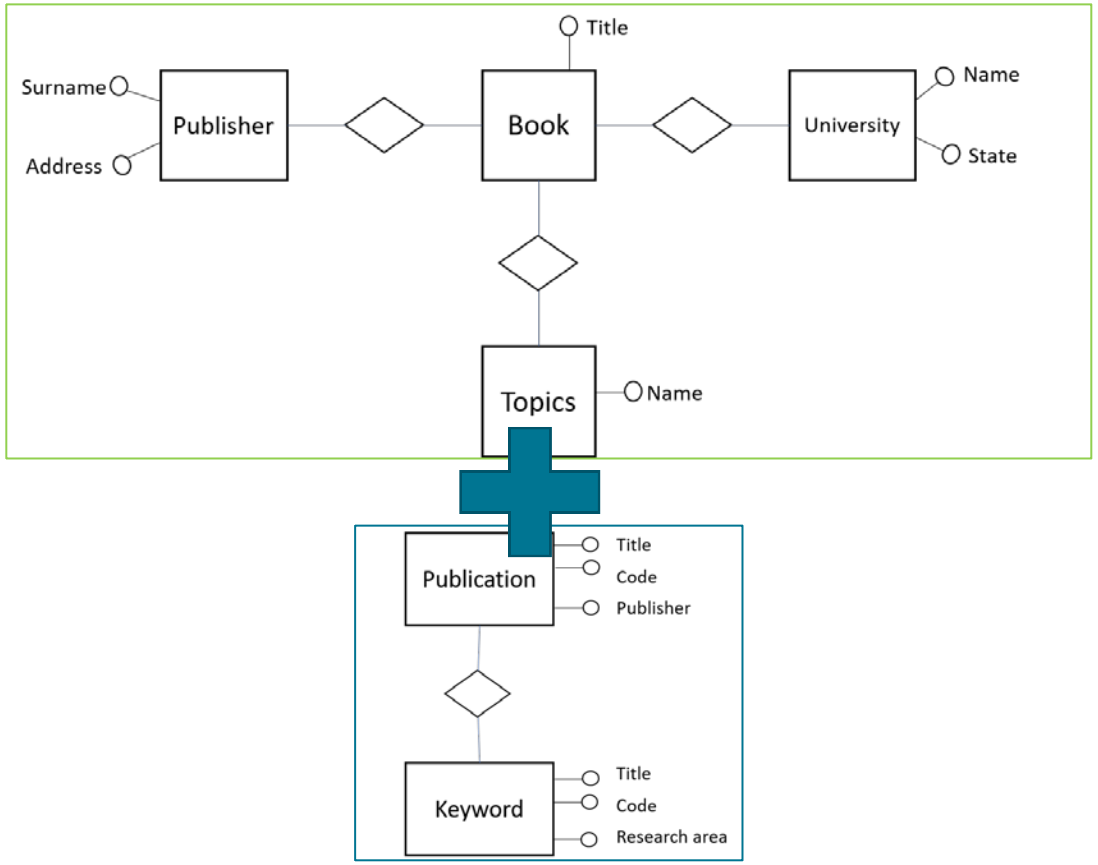
5. Integrated schema



No information lost!

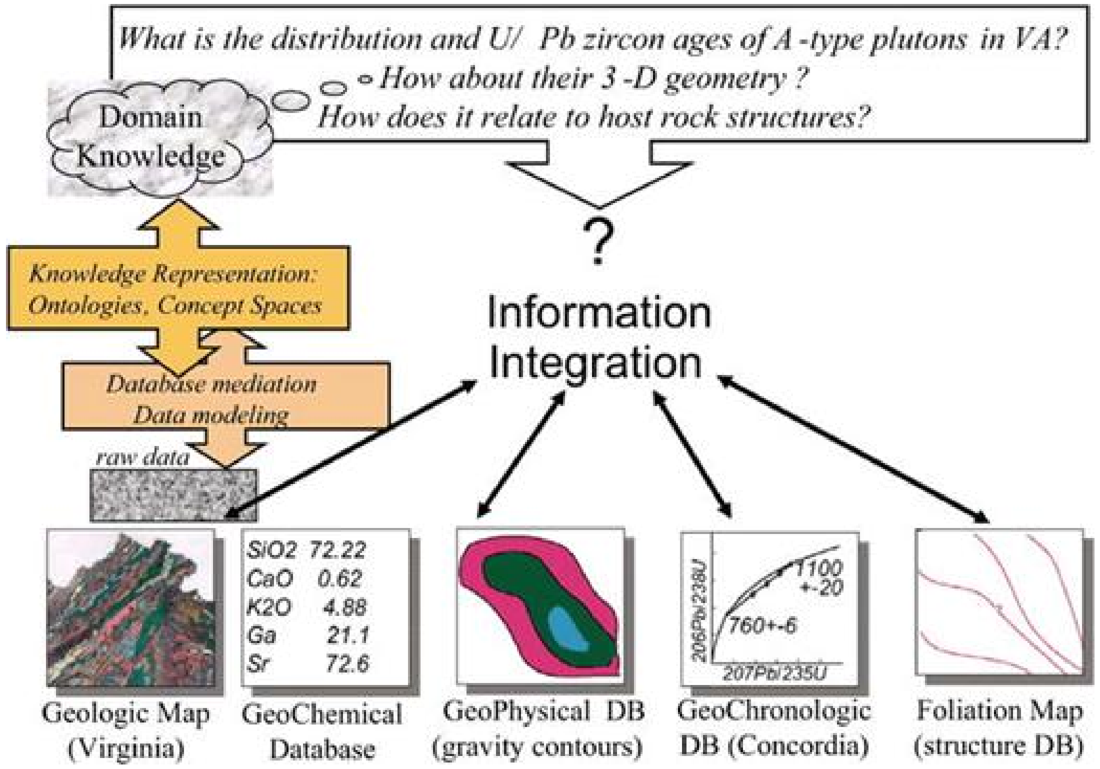
Ok, it isn't really so easy

One-world scenario



Batini, C., Lenzerini, M., and Navathe, S. B. (1986). A comparative analysis of methodologies for database schema integration. ACM Computing Surveys 18(4). doi>10.1145/27633.27634

Complex, multiple-worlds scenario


What is the distribution and U/ Pb zircon ages of A -type plutons in VA?
How about their 3 -D geometry ?
How does it relate to host rock structures?
Domain Knowledge
Knowledge Representation: Ontologies, Concept Spaces
Database mediation Data modeling
raw data
?
Information Integration
SiO2 72.22
CaO 0.62
K2O 4.88
Ga 21.1
Sr 72.6
206Pb/238U
1100 +-20
760+-6
207Pb/235U
Geologic Map (Virginia)
GeoChemical Database
GeoPhysical DB (gravity contours)
GeoChronologic DB (Concordia)
Foliation Map (structure DB)

Bertram Ludäscher, Kai Lin, Shawn Bowers, Efrat Jaeger-Frank, Boyan Brodaric, Chaitan Baru, 2006. "Managing scientific data: From data integration to scientific workflows* ", Geoinformatics: Data to Knowledge, A. Krishna Sinha