

ILLINOIS INSTITUTE OF TECHNOLOGY

CS520



Data Integration, Warehousing, and Provenance

1. Introduction

IIT DBGroup

Boris Glavic

<http://www.cs.iit.edu/~glavic/>
<http://www.cs.iit.edu/~cs520/>
<http://www.cs.iit.edu/~dbgroup/>


ILLINOIS INSTITUTE OF TECHNOLOGY

Outline

- 0) Course Info
- 1) **Introduction**
- 2) Data Preparation and Cleaning
- 3) Schema matching and mapping
- 4) Virtual Data Integration
- 5) Data Exchange
- 6) Data Warehousing
- 7) Big Data Analytics
- 8) Data Provenance

1

CS520 - 1) Introduction




ILLINOIS INSTITUTE OF TECHNOLOGY

Overview

- Topics covered in this part
 - Heterogeneity and Autonomy
 - Data Integration Tasks
 - Data Integration Architectures (Methods)
 - Some Formal Background (sorry!)

2

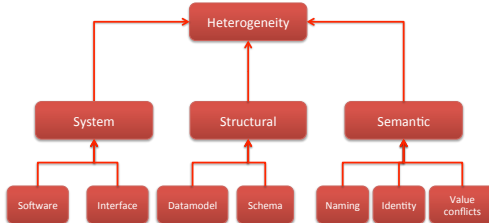
CS520 - 1) Introduction



ILLINOIS INSTITUTE OF TECHNOLOGY

1.1 Heterogeneity + Autonomy

- Taxonomy of Heterogeneity




```

    graph BT
      Heterogeneity --> System
      Heterogeneity --> Structural
      Heterogeneity --> Semantic
      System --> Software
      System --> Interface
      Structural --> Datamodel
      Structural --> Schema
      Semantic --> Naming
      Semantic --> Identity
      Semantic --> ValueConflicts[Value conflicts]
    
```

3


CS520 - 1) Introduction



ILLINOIS INSTITUTE OF TECHNOLOGY


1.1 System Heterogeneity

- Hardware/Software
 - Different hardware capabilities of sources
 - Different protocols, binary file formats, ...
 - Different access control mechanism
- Interface Heterogeneity
 - Different interfaces for accessing data from a source
 - HTML forms
 - XML-Webservices
 - Declarative language



4


CS520 - 1) Introduction



ILLINOIS INSTITUTE OF TECHNOLOGY


1.1 System Heterogeneity

- Hardware/Software
 - Different hardware capabilities of sources
 - **Mobile phone vs. server:** Cannot evaluate cross-product of two 1GB relations on a mobile phone
 - Different protocols, binary file formats, ...
 - **Order information stored in text files:** line ending differs between Mac/Window/Linux, character encoding
 - Different access control mechanism
 - **FTP-access to files:** public, ssh authentication, ..



5

CS520 - 1) Introduction



1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity
 - Different interfaces for accessing data from a source
 - HTML forms
 - Services (SOA)
 - Declarative language
 - Files
 - Proprietary network protocol
 - ...

6 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Expressiveness
 - Keyword-search vs. query language
 - Predicates:** equality (=), inequality (<, !=)
 - Logical connectives:** conjunctive (AND), disjunctive (OR), negation
 - Complex operations:** aggregation, quantification
 - Limitations:** restriction to particular tables, predicates, fixed queries with parameters, ...

7 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Examples
 - Google search (+/-, site:, intitle:, filetype:)

8 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Examples
 - SQL

9 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Examples
 - SQL

10 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Examples
 - Web-form (with DB backend?)

11 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Interface Heterogeneity – Examples
 - Email-client

12 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Problems with interface heterogeneity
 - Global query language is more powerful
 - User queries may not be executable
 - Integration system has to evaluate part of the query
 - Bound parameters are incompatible with query
 - User query may not be executable

13 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Example: more expressive global language
 - SQL with one table
 - books (title, author, year, isbn, genre)
 - Web form for books about history shown below
 - What problems do may arise translating user queries?

14 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Integration system has to process part of the query

```

SELECT title
FROM books
WHERE author = 'Steven King'
AND year = 2012;
    
```

15 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Query requires multiple requests

```

SELECT title
FROM books
WHERE author LIKE '%King%';
    
```

16 CSS20 - 1) Introduction

1.1 System Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Query cannot be answered

```

SELECT title
FROM books
WHERE genre = 'SciFi';
    
```

17 CSS20 - 1) Introduction

1.1 Heterogeneity + Autonomy

ILLINOIS INSTITUTE OF TECHNOLOGY

- Taxonomy of Heterogeneity

18 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Data model
 - Different semantic/expressiveness
 - Different structure
- Schema
 - Integrity constraints, keys
 - Schema elements:
 - use attribute or separate relations)
 - Structure:
 - e.g., normalized vs. denormalized relational schema

19 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Data model
 - Relational model
 - XML model
 - Object-oriented model
 - Ontological model
 - JSON
 - ...

20 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Example: data model
 - Relational model
 - XML model
 - JSON
 - OO
- Person and their addresses

21 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Schema
 - Modeling choices
 - Relation vs. attribute
 - Attribute vs. value
 - Relation vs. value
 - Naming
 - Normalized vs. denormalized (relational concept)
 - Nesting vs. reference

22 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

Example: Modeling choices

```

Male(Id, firstname, lastname)
Female(id, firstname, lastname)
Person(Id, firstname, lastname, male, female)
Person(Id, firstname, lastname, gender)
                    
```

Relation vs. Attribute (arrow from Male/Female to Person)

Relation vs. Value (arrow from Person to Id)

Value vs. Attribute (arrow from Person to male/female)

23 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Relation-relation conflicts**
 - Naming conflicts
 - Relations with different name representing the same data (**synonym**)
 - Relations with same name representing different information (**homonym**)
 - Structural conflicts
 - Missing attributes
 - Many-to-one
 - Missing, but derivable attributes
 - Integrity constraint conflicts

24 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

Example: Conflicts between relations

```

Person(Id, firstname, lastname, male, female)
Person(Id, name, gender, birthday)
Manager(Id, name, gender, age)
    
```

25 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

Example: Conflicts between relations

```

Person(Id, firstname, lastname, male, female)
Person(Id, name, gender, birthday)
Manager(Id, name, gender, age)
    
```

Multiple attribute vs one attribute

Missing derivable attribute: Role

Derivable attribute: Compute age from birthday

26 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Attribute-attribute conflicts**
 - Naming conflicts
 - Attributes with different name representing the same data (**synonym**)
 - Attributes with same name representing different information (**homonym**)
 - Default value conflict
 - Integrity constraint conflicts
 - Datatype
 - Constraints restricting values

27 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

Example: Conflicts between attributes and attributes

SSN	FirstName VARCHAR(40)	LastName	Age CHECK(Age > 18)
333-333-3333	Peter	Schmeter	30
333-333-9999	Hans	Glanz	NULL

SSN	FirstName VARCHAR(25)	SurName	Age
3333333333	Peter	Schmeter	30
3333339999	Hans	Glanz	-1

28 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

Example: Conflicts between attributes and attributes

SSN	FirstName VARCHAR(40)	LastName	Age CHECK(Age > 18)
333-333-3333	Peter	Schmeter	30
333-333-9999	Hans	Glanz	NULL

SSN	FirstName VARCHAR(25)	SurName	Age
3333333333	Peter	Schmeter	30
3333339999	Hans	Glanz	-1

Conflicting format

Conflicting datatype

Conflicting constraint

Conflicting default value

synonym

29 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Normalized vs. denormalized**
 - E.g., relational model: Association between entities can be represented using multiple relations and foreign keys or one relation

Example

30 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Nested vs. flat**
 - Association between entities can be represented using nesting or references (previous slides)

Example

31 CSS20 - 1) Introduction

1.1 Structural Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Problems caused by schema heterogeneity**
 - Unified access to multiple schemas or integrate schemas into new schema
 - **Schema level:** schema mapping, model management operators, schema languages
 - **Data Level:** virtual data integration, data exchange, warehousing (ETL)

32 CSS20 - 1) Introduction

1.1 Heterogeneity + Autonomy

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Taxonomy of Heterogeneity**

33 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Semantic Heterogeneity**
 - Naming Conflicts
 - Identity Conflicts (Entity resolution)
 - Value Conflicts (Data Fusion)

34 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Naming Conflicts**
 - Ontological (concepts)
 - Birds vs. Animals
 - Synonyms
 - Surname vs. last name
 - Homonyms
 - Units
 - Gallon vs. liter
 - Values
 - Manager vs. Boss

35 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Ontological concepts**
 - Relationships between concepts
 - $A = B$ - Equivalence
 - $A \subseteq B$ - Inclusion
 - $A \cap B$ - Overlap
 - $A \neq B$ - Disjunction

36 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Ontological concepts**
 - Relationships between concepts
 - $A = B$ - Equivalence
 - $A \subseteq B$ - Inclusion
 - $A \cap B$ - Overlap
 - $A \neq B$ - Disjunction

Example

Equivalence: Human vs Homo sapiens
 Inclusion: Bird vs Animal
 Overlap: Animal vs aquatic lifeform
 Disjunction: Fish vs Mammal

37 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Naming concepts (synonyms)**
 - Different words with same meaning

Example

Person (Name, Age)
 Human (LastName, Age)

38 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Naming concepts (homonyms)**
 - Same words with different meaning

Example

Person (Title, Name)
 Movie (Title, Year)

39 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Naming concepts (units)**

Example

Person (Title, Name, Salary) \$
 Person (Title, Name, Salary) CAD

40 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- **Identity Conflicts**
 - What is an object?
 - E.g., multiple tuples in relational model
 - Central question:
 - Does object A represent the same entity as B
 - This problem has been called
 - **Entity resolution**
 - **Record linkage**
 - **Deduplication**
 - ...

41 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Identity Conflicts

Example

(IBM,300000000,USA)
(International Business Machines Corporation,50000)

42 CSS20 - 1) Introduction

1.1 Semantic Heterogeneity

ILLINOIS INSTITUTE OF TECHNOLOGY

- Value Conflicts
 - Objects representing the same entities have conflicting values for semantically equivalent attributes
 - We have to identified that these objects are represent the same entity first!
 - Resolving such conflicts require **Data Fusion**
 - Pick value from conflicting values
 - Numerical methods: e.g., average
 - Preferred value
 - ...

43 CSS20 - 1) Introduction

1.1 Autonomy

ILLINOIS INSTITUTE OF TECHNOLOGY

- How autonomous are data sources
 - One company
 - Can enforce, e.g., schema and software
 - ...
 - The web
 - Website decides
 - Interface
 - Determines access restrictions and limits
 - Availability
 - Format
 - Query restrictions
 - ...

44 CSS20 - 1) Introduction

1.2 Data integration tasks

ILLINOIS INSTITUTE OF TECHNOLOGY

- Cleaning and preparation
- Entity resolution
- Data Fusion
- Schema matching
- Schema mapping
- Query rewrite
- Data translation

45 CSS20 - 1) Introduction

1.3 Data integration architectures

ILLINOIS INSTITUTE OF TECHNOLOGY

- Virtual data integration
- Data Exchange
- Peer-to-peer data integration
- Datawarehousing
- Big Data analytics

46 CSS20 - 1) Introduction

1.4 Formal Background

ILLINOIS INSTITUTE OF TECHNOLOGY

- Query Equivalence
 - Complexity for different query classes
- Query Containment
 - Complexity for different query classes
- Datalog
 - Recursion + Negation
- Integrity Constraints
 - Logical encoding of integrity constraints
- Similarity Measures/Metrics

47 CSS20 - 1) Introduction

1.4 Integrity constraints

ILLINOIS INSTITUTE OF TECHNOLOGY

- You know some types of integrity constraints already
 - Functional dependencies
 - Keys are a special case
 - Foreign keys
 - We have not really formalized that

48 CS520 - 1) Introduction

1.4 Integrity constraints

ILLINOIS INSTITUTE OF TECHNOLOGY

- Other types are
 - Conditional functional dependencies
 - E.g., used in cleaning
 - Equality-generating dependencies
 - Multi-valued dependencies
 - Tuple-generating dependencies
 - Join dependencies
 - Denial constraints
 - ...

49 CS520 - 1) Introduction

1.4 Integrity constraints

ILLINOIS INSTITUTE OF TECHNOLOGY

- How to manage all these different types of constraints?
 - Has been shown that these constraints can be expressed in a logical formalism.
 - Formulas which consist of relational and comparison atoms. Variables represent values
 - $R(x,y,z)$
 - $x = y$

50 CS520 - 1) Introduction

1.4 Integrity Constraints

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

Primary Key $R(\underline{A}, B)$:
 $\forall x, y, z : R(x, y) \wedge R(x, z) \rightarrow y = z$

Functional Dependency $R(A, B)$ with $A \rightarrow B$:
 $\forall x, y, z, a : R(x, y) \wedge R(z, a) \wedge x = z \rightarrow y = a$

Foreign Key $R(\underline{A}, B), S(C, D)$ where D is FK to R :
 $\forall x, y : S(x, y) \rightarrow \exists z : R(y, z)$

51 CS520 - 1) Introduction

1.4 Integrity constraints

ILLINOIS INSTITUTE OF TECHNOLOGY

- Types of constraints we will use a lot
 - Tuple-generating dependencies (**tgds**)
 - Implication with conjunction of relational atoms
 - Foreign keys and schema mappings (later)
 - $\forall \vec{x} : \phi(\vec{x}) \rightarrow \exists \vec{y} : \psi(\vec{x}, \vec{y})$
 - Equality-generating dependencies (**egds**)
 - Generalizes keys, FDs
 - $\forall \vec{x} : \phi(\vec{x}) \rightarrow \bigwedge_{k=1}^n x_{i_k} = x_{j_k}$

52 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

- What is datalog?
 - Prolog for databases (syntax very similar)
 - A logic-based query language
- Queries (Program) expressed as set of rules

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$
- One Q is specified as the answer relation (the relation returned by the query)

53 CS520 - 1) Introduction

1.4 Datalog - Intuition



- A **Datalog rule**

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$

- For all bindings of variables in the right-hand side (RHS) that makes the RHS true (conjunction) return bindings of \vec{x}

Example

```
Q (Name) :- Person (Name, Age) .
Return names of persons
```



54

CS520 - 1) Introduction

1.4 Datalog - Syntax



- A **Datalog program** is a set of datalog rules
 - Optionally a distinguished answer predicate

- A **Datalog rule** is

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$

- **X's** are lists of variables and constants
- **Ri's** are relation names
- **Q** is a relation name



55

CS520 - 1) Introduction

1.4 Datalog - Terminology



- Left-hand side of a rule is called it's **head**
- Right-hand side of a rule is called it's **body**
- Relation are called **predicates**
- $R(\vec{x})$ is called an **atom**
- An **instance I** of a database is the data
- The **active domain** $\text{adom}(I)$ of an instance I is the set of all constants that occur in I

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$



56

CS520 - 1) Introduction

1.4 Datalog - Terminology



Example

```
Q (N) :- Person (N, A) .
```

N, A are **variables**

Q (N) is an **atom**

Person and Q are **predicates**

Name	Age
peter	34
bob	45

Activate domain

$\text{adom}(I) = \{\text{peter}, \text{bob}, 34, 45\}$



57

CS520 - 1) Introduction

1.4 Datalog - Terminology



- **Intensional vs. extensional**
 - Extensional database (**edb**)
 - What we usually call database
 - Intensional database (**idb**)
 - Relations that occur in the head of rules (are populated by the query)
 - Usually we assume that these do not overlap

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$



58

CS520 - 1) Introduction

1.4 Datalog - Safety



- A datalog program is safe if all its rules are **safe**
- A rule is **safe** if all variables in \vec{x} occur in at least one \vec{x}_i

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$

Example

```
Q (Name) :- Person (Name, Age) . (safe)
Q (Name, Sal) :- Peron (Name, Age) . (unsafe)
```



59

CS520 - 1) Introduction

1.4 Datalog - Semantics

ILLINOIS INSTITUTE OF TECHNOLOGY

- The instance of an idb predicate Q in a datalog program for an edb instance I contains all facts that can be derived by applying rules with Q in the head
- A rule derives a fact $Q(c)$ if we can find a binding of variables of the rule to constants from $\text{adom}(I)$ such that x is bound to c and the body is true

$$Q(\vec{x}) : -R_1(\vec{x}_1), \dots, R_n(\vec{x}_n).$$

60 CS520 - 1) Introduction

1.4 Datalog - Semantics

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$Q(N) :- \text{Person}(N, A) .$

```

N=peter,A=peter: Q(peter):- Person(peter,peter) .
N=peter,A=bob: Q(peter):- Person(peter,bob) .
N=peter,A=34: Q(peter):- Person(peter,34) .
N=bob,A=peter: Q(bob):- Person(peter,peter) .
N=bob,A=bob: Q(bob):- Person(peter,bob) .
N=bob,A=34: Q(bob):- Person(peter,34) .
N=34,A=peter: Q(34):- Person(34,peter) .
N=34,A=bob: Q(34):- Person(34,bob) .
N=34,A=34: Q(34):- Person(34,34) .
    
```

N
peter
bob

Activate domain

$\text{adom}(I) = \{\text{peter}, \text{bob}, 34\}$

Name	Age
peter	34
bob	34

61 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

- Different flavors of datalog
 - Conjunctive query**
 - Only one rule
 - Expressible as Select-project-join (SPJ) query in relational algebra
 - Union of conjunctive queries**
 - Also allow union
 - SPJ + set union in relational algebra
 - Rules with the same head in Datalog
 - Conjunctive queries with inequalities**
 - Also allow inequalities, e.g., <

62 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

- Different flavors of datalog
 - Recursion**
 - Rules may have recursion:
 - E.g., head predicate in the body
 - Fix point semantics based on immediate consequence operator
 - Negation (first-order queries)**
 - Negated relational atoms allowed
 - Require that every variable used in a negated atom also occurs in at least on positive atom (**safety**)
 - Combined Negation + recursion**
 - Stronger requirements (stratification)

63 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

```

Q1(x, y) : R(x, y), R(x, z) .
Q2(x, y) : R(x, y) .
Q3(x, x) : R(x, x) .
Q4(x, y) : R(x, y) .
Q5(x, x) : R(x, y), R(x, x) .
Q6(x, z) : R(x, y), R(y, z) .
    
```

64 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

Relation **hops(A, B)** storing edges of a graph.

```

Q2hop(x, z) : hop(x, y), hop(x, z) .

Qreach(x, y) : hop(x, y) .
Qreach(x, z) : Qreach(x, y), Qreach(y, z) .

Qnode(x) : hop(x, y) .
Qnode(x) : hop(y, x) .
    
```

65 CS520 - 1) Introduction

1.4 Datalog

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

Relation **hops(A,B)** storing edges of a graph.

$Q_{node}(x) : \text{hop}(x,y).$
 $Q_{node}(x) : \text{hop}(y,x).$

$Q_{notReach}(x,y) : Q_{node}(x), Q_{node}(y),$
 $\text{not } Q_{reach}(x,y).$

66 CSS20 - 1) Introduction

1.4 Containment and Equivalence

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition: Query Equivalence

Query Q is equivalent to Q' iff for every database instance I both queries return the same result

$$Q \equiv Q' \Leftrightarrow \forall I : Q(I) = Q'(I)$$

Definition: Query Containment

Query Q is contained in query Q' iff for every database instance I the result of Q is contained in the result of Q'

$$Q \sqsubseteq Q' \Leftrightarrow \forall I : Q(I) \subseteq Q'(I)$$

67 CSS20 - 1) Introduction

1.4 Equivalence

ILLINOIS INSTITUTE OF TECHNOLOGY

- The problem of checking query equivalence is of different complexity depending on the **query language** and whether we consider **set or bag semantics**

68 CSS20 - 1) Introduction

1.4 Containment and Equiv.

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$Q_1(x,y) : R(x,y), R(x,z).$
 $Q_2(x,y) : R(x,y).$
 $Q_3(x,x) : R(x,x).$
 $Q_4(x,y) : R(x,y).$
 $Q_5(x,x) : R(x,y), R(x,x).$
 $Q_6(x,z) : R(x,y), R(y,z).$

69 CSS20 - 1) Introduction

1.4 Containment and Equiv.

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

Relation **hops(A,B)** storing edges of a graph.

$Q_{2hop}(x,z) : \text{hop}(x,y), \text{hop}(x,z).$

$Q_{up2Hop}(x,z) : \text{hop}(x,y), \text{hop}(x,z).$
 $Q_{up2Hop}(x,z) : \text{hop}(x,z).$

$Q_{sym}(x,y) : \text{hop}(x,y).$
 $Q_{sym}(x,y) : \text{hop}(y,x).$
 $Q_{sym2Hop}(x,y) : Q_{sym}(x,y), Q_{sym}(y,z).$

70 CSS20 - 1) Introduction

1.4 Complexity of Eq. and Cont.

ILLINOIS INSTITUTE OF TECHNOLOGY

Set semantics	Relational Algebra	Conjunctive Queries (CQ)	Union of Conjunctive Queries (UCQ)	Monotone Queries/CQ*
Query Evaluation (Combined Complexity)	PSPACE-complete	NP-complete	NP-complete	NP-complete
Query Evaluation (Data Complexity)	LOGSPACE (that means in P)	LOGSPACE (that means in P)	LOGSPACE (that means in P)	LOGSPACE (that means in P)
Query Equivalence	Undecidable	NP-complete	NP-complete	Π_2^P -complete
Query Containment	Undecidable	NP-complete	NP-complete	Π_2^P -complete

71 CSS20 - 1) Introduction

1.4 Complexity of Eq. and Cont.

Bag semantics	Relational Algebra	Conjunctive Queries (CQ)	Union of Conjunctive Queries (UCQ)	Monotone Queries/ CQ \neq
Query Equivalence	Undecidable	Equivalent to graph isomorphism		It is in PSPACE, lower-bound unknown
Query Containment	Undecidable	Open Problem	Undecidable	Π_2^P -complete

72 CSS20 - 1) Introduction

1.4 Containment Mappings

- NP-completeness for set semantics CQ and UCQ for the containment, evaluation, and equivalence problems is based on reducing these problems to the same problem
 - [Chandra & Merlin, 1977]
- Notational Conventions:
 - head(Q)** = variables in head of query Q
 - body(Q)** = atoms in body of Q
 - vars(Q)** = all variable in Q

73 CSS20 - 1) Introduction

1.4 Boolean Conjunctive Queries

- A conjunctive query is boolean if the head does not have any variables
 - Q() :- hop(x,y), hop(y,z)**
 - We will use $Q :- \dots$ as a convention for $Q() :- \dots$
 - What is the result of a boolean query
 - Empty result {}, e.g., no **hop(x,y), hop(y,z)**
 - If there are tuples matching the body, then a tuple with zero attributes is returned {}
 - \rightarrow We interpret {} as **false** and {} as **true**
 - Boolean query is essentially an existential check

74 CSS20 - 1) Introduction

1.4 Boolean Conjunctive Queries

- BCQ in SQL

```

Example
Hop relation: Hop(A, B)

Q :- hop(x, y)

SELECT EXISTS (SELECT * FROM hop)

Note: in Oracle and DB2 we need a
from clause
  
```

75 CSS20 - 1) Introduction

1.4 Boolean Conjunctive Queries

```

Example
SELECT
  CASE WHEN EXISTS (SELECT *
                    FROM hop)
  THEN 1 ELSE 0
  END AS x
FROM dual;

Notes:
- Oracle and DB2 FROM not optional
- Oracle has no boolean datatype
  
```

76 CSS20 - 1) Introduction

1.4 Boolean Conjunctive Queries

- BCQ in SQL

```

Example
Q :- hop(x, y), hop(y, z)

SELECT EXISTS
  (SELECT *
   FROM hop l, hop r
   WHERE l.B = r.A)
  
```

77 CSS20 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

- How to check for containment of CQs (set)

Definition: Variable Mapping
 A variable mapping ψ from query Q to query Q' maps the variables of Q to constants or variables from Q'

Definition: Containment Mapping
 A containment mapping from query Q to Q' is a variable mapping ψ such that:

$$\Psi(\text{head}(Q)) = \text{head}(Q')$$

$$\forall R(\vec{x}_i) \in \text{body}(Q) : \Psi(\vec{x}_i) \in \text{body}(Q')$$

78 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Theorem: Containment Mapping and Query Containment
 Query Q is contained in query Q' iff there exists a containment mapping ψ from Q' to Q

Example

$$Q_1(u, z) : R(u, z) .$$

$$Q_2(x, y) : R(x, y) .$$

Can we find a containment mapping?

79 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Theorem: Containment Mapping and Query Containment
 Query Q is contained in query Q' iff there exists a containment mapping ψ from Q' to Q

Example

$$Q_1(u, z) : R(u, z) .$$

$$Q_2(x, y) : R(x, y) .$$

$$Q_1 \rightarrow Q_2 : \Psi(u) = x, \Psi(z) = y$$

$$Q_2 \rightarrow Q_1 : \Psi(x) = u, \Psi(y) = z$$

80 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$$Q_1(a, b) : R(a, b), R(c, b) .$$

$$Q_2(x, y) : R(x, y) .$$

81 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$$Q_1(a, b) : R(a, b), R(b, c) .$$

$$Q_2(x, y) : R(x, y) .$$

Do containment mappings exist?

$Q_1 \rightarrow Q_2$: none exists
 $Q_2 \rightarrow Q_1$: $\Psi(x) = a, \Psi(y) = b$

82 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$$Q_1(a, b) : R(a, b), R(c, b) .$$

$$Q_2(x, y) : R(x, y) .$$

$Q_1 \rightarrow Q_2$: $\Psi(a) = x, \Psi(b) = y, \Psi(c) = y$
 $Q_2 \rightarrow Q_1$: $\Psi(x) = a, \Psi(y) = b$

83 CS520 - 1) Introduction

1.4 Containment Background

ILLINOIS INSTITUTE OF TECHNOLOGY

- It was shown that query evaluation, containment, equivalence as all reducible to homomorphism checking for CQ
 - Canonical conjunctive query Q^I for instance I
 - Interpret attribute values as variables
 - The query is a conjunction of all atoms for the tuples
 - $I = \{\text{hop}(a,b), \text{hop}(b,c)\} \rightarrow Q^I :- \text{hop}(a,b), \text{hop}(b,c)$
 - Canonical instance I^Q for query Q
 - Interpret each conjunct as a tuple
 - Interpret variables as constants
 - $Q :- \text{hop}(a,a) \rightarrow I^Q = \{\text{hop}(a,a)\}$

84 CS520 - 1) Introduction

1.4 Containment Background

ILLINOIS INSTITUTE OF TECHNOLOGY

- Containment Mapping \leftrightarrow Containment
- Proof idea (boolean queries)
 - (if direction)
 - Assume we have a containment mapping Q_1 to Q_2
 - Consider database D
 - $Q_2(D)$ is true then we can find a mapping from $\text{vars}(Q_2)$ to D
 - Compose this with the containment mapping and prove that this is a result for Q_1

85 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$$Q_1() : R(a,b), R(c,b).$$

$$Q_2() : R(x,y).$$

$$Q_2 \rightarrow Q_1 : \Psi(x)=a, \Psi(y)=b$$

$$D = \{R(1,1), R(1,2)\}$$

$$Q_1(D) = \{(1,1), (1,2)\}$$

$$\varphi(a)=1, \varphi(b)=2, \varphi(c)=1$$

$$\Psi \varphi(x)=1, \Psi \varphi(y)=2$$

86 CS520 - 1) Introduction

1.4 Containment Background

ILLINOIS INSTITUTE OF TECHNOLOGY

- Containment Mapping \leftrightarrow Containment
- Proof idea (boolean queries)
 - (only-if direction)
 - Assume Q_2 contained in Q_1
 - Consider canonical (frozen) database I^{Q_2}
 - Evaluating Q_1 over I^{Q_2} and taking a variable mapping that is produced as a side-effect gives us a containment mapping

87 CS520 - 1) Introduction

1.4 Containment Mappings

ILLINOIS INSTITUTE OF TECHNOLOGY

Example

$$Q_1() : R(a,b), R(c,b).$$

$$Q_2() : R(x,y).$$

$$Q_2 \rightarrow Q_1 : \Psi(x)=a, \Psi(y)=b$$

$$I^{Q_1} = \{(a,b), (c,b)\}$$

$$Q_2(I^{Q_1}) = \{()\}$$

$$\varphi(x)=a, \varphi(y)=b$$

φ is our containment mapping Ψ

88 CS520 - 1) Introduction

1.4 Containment Background

ILLINOIS INSTITUTE OF TECHNOLOGY

- If you are not scared and want to know more:
 - Look up Chandra and Merlins paper(s)
 - The text book provides a more detailed overview of the proof approach
 - Look at the slides from Phokion Kolaitis excellent lecture on database theory
 - <https://classes.soe.ucsc.edu/cmcs277/Winter10/>

89 CS520 - 1) Introduction

1.4 Containment Background



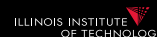
- A more intuitive explanation why containment mappings work
 - Variable naming is irrelevant for query results
 - If there is a containment mapping Q to Q'
 - Then every condition enforced in Q is also enforced by Q'
 - Q' may enforce additional conditions

90

CS520 - 1) Introduction



1.4 Containment Mappings



Example

```

 $Q_1() : R(a, b), R(c, b).$ 
 $Q_2() : R(x, y).$ 
 $Q_2 \rightarrow Q_1 : \Psi(x) = a, \Psi(y) = b$ 

```

If there exists tuples
 $R(a, b)$ and $R(c, b)$
 in R that make Q_1 true, then we take
 $R(a, b)$
 to fulfill Q_2

91

CS520 - 1) Introduction



1.4 Containment Background



- From boolean to general conjunctive queries
 - Instead of returning true or false, return bindings of variables
 - Recall that containment mappings enforce that the head is mapped to the head
 - \rightarrow same tuples returned, but again Q' 's condition is more restrictive

92

CS520 - 1) Introduction



1.4 Containment Mappings



Example

```

 $Q_1(a) : R(a, b), R(c, b).$ 
 $Q_2(x) : R(x, y).$ 
 $Q_2 \rightarrow Q_1 : \Psi(x) = a, \Psi(y) = b$ 

```

For every
 $R(a, b)$ and $R(c, b)$
 Q_1 returns (a) and for every
 $R(a, b)$
 Q_2 returns (a)

93

CS520 - 1) Introduction



1.4 Similarity Measures



- Problem faced by multiple integration tasks
 - Given two objects, how similar are they
 - E.g., given two attribute names in schema matching, given two values in data fusion/entity resolution, ...

94

CS520 - 1) Introduction



1.4 Similarity Measures



- Object models
 - **Multidimensional (feature vector model)**
 - Object is described as a vector of values - one for each dimension out of a given set of dimensions
 - E.g., Dimensions are gender (male/female), age (0-120), and salary (0-1,000,000). An example object is [male, 80, 70,000]
 - **Strings**
 - E.g., how similar is "Poeter" to "Peter"
 - **Graphs and Trees**
 - E.g., how similar are two XML models

95

CS520 - 1) Introduction



1.4 Similarity Measures

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition: Similarity Measure


Function $d(p,q)$ where p and q are objects, that returns a real score with

- $d(p,p) = 0$
- $d(p,q) \geq 0$

- Interpretation: the lower the score the “more similar” the objects are
- We require $d(p,p)=0$, because nothing can be more similar to an object than itself
- Note: often scores are normalized to the range $[0,1]$

96

CSS20 - 1) Introduction



1.4 Similarity Measures

ILLINOIS INSTITUTE OF TECHNOLOGY

Example


String equality: $d(p,q) = 0$ if $p=q$
strings $d(p,q) = 1$ else

Euclidian distance: $d(p,q) = \sqrt{\sum_{i=1}^n (p[i] - q[i])^2}$
N-dimensional space

Edit distance: $d(p,q) =$ minimum number of single character insertions, deletions, replacements to transform p into q

97

CSS20 - 1) Introduction



1.4 Similarity Measures

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition: Metric


Function $d(p,q)$ where p and q are objects, that returns a real score with

- Non-negative $d(p,q) \geq 0$
- Symmetry $d(p,q) = d(q,p)$
- Identity of indiscernibles $d(p,q) = 0$ iff $p=q$
- Triangle inequality $d(p,q) + d(q,r) \geq d(p,r)$

- Metric is a stricter definition
- Which of the previous similarity measure is a metric?

98

CSS20 - 1) Introduction



1.4 Similarity Measures

ILLINOIS INSTITUTE OF TECHNOLOGY

Definition: Metric


Function $d(p,q)$ where p and q are objects, that returns a real score with

- Non-negative $d(p,q) \geq 0$
- Symmetry $d(p,q) = d(q,p)$
- Identity of indiscernibles $d(p,q) = 0$ iff $p=q$
- Triangle inequality $d(p,q) + d(q,r) \geq d(p,r)$

- Metric is a stricter definition
- Which of the previous similarity measure is a metric?
 - All of them!

99

CSS20 - 1) Introduction




1.4 Similarity Measures

ILLINOIS INSTITUTE OF TECHNOLOGY

- Why do we care whether d is a metric?
 - Some data mining algorithms only work for metrics
 - E.g., some clustering algorithms such as k-means
 - E.g., clustering has been used in entity resolution
 - Metric spaces allow optimizations of some methods
 - E.g., Nearest Neighborhood-search: find the most similar object to an object p . This problem can be efficiently solved using index structures that only apply to metric spaces

100

CSS20 - 1) Introduction




Summary

ILLINOIS INSTITUTE OF TECHNOLOGY

- Heterogeneity
 - Types of heterogeneity
 - Why do they arise?
 - Hint at how to address them
- Autonomy
- Data Integration Tasks
- Data Integration Architectures
- Background
 - Datalog + Query equivalence/containment + Similarity + Integrity constraints

101

CSS20 - 1) Introduction



Outline

ILLINOIS INSTITUTE
OF TECHNOLOGY

- 0) Course Info
- 1) Introduction
- 2) Data Preparation and Cleaning**
- 3) Schema matching and mapping
- 4) Virtual Data Integration
- 5) Data Exchange
- 6) Data Warehousing
- 7) Big Data Analytics
- 8) Data Provenance



102

CSSM - 11 Introduction