Database Theory Basics

Jose Luis Ambite
University of Southern California

Motivation

How does a data integration system decide which sources are relevant to a query?

Which are redundant?

How to combine multiple sources to answer a query?

- By reasoning about the contents of data sources
- Data sources are often described by queries/views
- This lecture describes fundamental tools for reasoning about queries

[based on slide by Doan, Halevy, Ives]

Basic Database Theory: Some Concepts

- Relational data model
- Queries and answers
- Recursive Queries: Datalog
- Query Containment

Relational Data Model

Relational schemas

- Relations/Tables, Attributes/Columns/Fields

Relation instances

- Sets (or bags) of tuples (rows)

Integrity constraints

- Keys, foreign keys, inclusion dependencies

[based on slide by Doan, Halevy, Ives]

Relational schema, instance

Relation Schema

Relation (table) name

Product(Pname, Price, Category, Manufacturer)

Product

Attribute names

Relation Instance

Tuples (rows)

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

[based on slide by Doan, Halevy, Ives]

Basic Database Theory Concepts

- Relational data model
- Queries and answers
- Recursive Queries: Datalog
- Query Containment

Query Answers

- Q(D): the set (or bag) of rows resulting from applying the query Q on the database D

- Unless otherwise stated, we will consider sets rather than bags.

[based on slide by Doan, Halevy, Ives]

Conjunctive Queries (CQs): Core SQL (select-project-join)

Conjunctive queries form the core or SQL

- Select-project-join queries
  - select A, B from R, S where R.C=S.D
- Most common type of queries

Example:

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

```
select recruiter, candidate
from Interview i, EmployeePerf e
where i.recruiter=e.name and e.grade < 2.5
```

[bad recruiters, re-interview candidates]

[based on slide by Doan, Halevy, Ives]

Conjunctive Queries: Rule notation

$\underbrace{Q(X,C)}_{\text{head}}$:- $\underbrace{Interview(C,D,X,H,G), EmployeePerf(E,X,Y,G2,R), G2<2.5}_{\text{body}}$

- Joins expressed with multiple occurrences of same variable
- Projections are indicated by variables in the head
- Selections with explicit interpreted predicates (ex: =, <, …) or implicitly with constants in regular predicates

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

```
select recruiter, candidate
from Interview i, EmployeePerf e
where i.recruiter=e.name and e.grade < 2.5
```

[based on slide by Doan, Halevy, Ives]

Safe Conjunctive Queries: Interpreted predicates

Q(X,C) :- Interview(C,D,X,H,G), EmployeePerf(E,X,Y,G2,R), G2<2.5

- Safety Condition: Variables in interpreted (e.g., comparison) predicates must also appear in regular predicates

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

select recruiter, candidate
from **Interview i**, **EmployeePerf e**
where i.recruiter=e.name **and** e.grade < 2.5

[based on slide by Doan, Halevy, Ives]

Safe Conjunctive Queries: Negated subgoals

Q(C,R) :- Interview(C,D,R,H,G), ¬OfferMade(C, D2), G > 3.5

- Safety Condition: every head variable must appear in a positive subgoal.

Interview(candidate, date, recruiter, hireDecision, grade)

OfferMade(candidate, date)

select candidate, recruiter
from **Interview**
where grade > 3.5 **and**
    candidate **not in** (select candidate from OfferMade)

[candidates with good interviews, but not hired, and their recruiters]

[based on slide by Doan, Halevy, Ives]

Unions of Conjunctive Queries (UCQs) 1

- Multiple rules with the same head predicate express union

Q(R,G) :- Interview(C,D,R,H,G1), EmployeePerf(E,R,Y,G,W), G>4

Q(A,B) :- Interview(C,D,A,H,G1), EmployeePerf(E,A,Y,B,W), B<2

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

select i.recruiter, e.grade
from **Interview i**, **EmployeePerf e**
where i.candidate=e.name **and** (e.grade > 4 **or** e.grade < 2)

[very good or very bad recruiters]

[based on slide by Doan, Halevy, Ives]

Unions of Conjunctive Queries (UCQs) 2

- Multiple rules with the same head predicate express union

Q(C) :- Interview(C,D,R,H,G)

Q(R) :- Interview(C,D,R,H,G)

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

select candidate **from** **Interview**

union

select recruiter **from** **Interview**

[all candidates and recruiters]

[based on slide by Doan, Halevy, Ives]

Unions of Conjunctive Queries (UCQs) 3

- Multiple rules with the same head predicate express union

Q(C) :- Interview(C,D,R,H,G)

Q(N) :- EmployeePerf(E,N,Y,G,W)

Interview(candidate, date, recruiter, hireDecision, grade)

EmployeePerf(empID, name, year, grade, reviewer)

select candidate **from** **Interview**

union

select name **from** **EmployeePerf**

[all candidates and employees]

[based on slide by Doan, Halevy, Ives]

Basic Database Theory Concepts

- Relational data model
- Queries and answers
- Recursive Queries: Datalog
- Query Containment

Datalog

- Datalog Program = set of datalog rules
- Datalog rule ~ conjunctive query

big-LA-sales(Buyer,Seller, Price) ← head Datalog

person(Buyer, "Los Angeles", Phone),
purchase(Buyer, Seller, Product, Price),
Price > 10000.

body

First-Order Logic

∀ Buyer, Seller, Price

[∃ *Phone, Product* [person(Buyer, "Los Angeles", *Phone*) ^
purchase(Buyer, Seller, *Product*, Price) ^
Price > 10000)]

antecedent

→ big-LA-sales(Buyer,Seller, Price)]

consequent

Datalog rule (strictly): function-free logical implication with single predicate consequent and conjunctive antedecent (function-free horn rule)

[based on slide by Jose Luis Ambite]

Conjunctive Queries and Views

Datalog rule ~ conjunctive view definition
Rule body ~ CQ, select-from-where construct of SQL

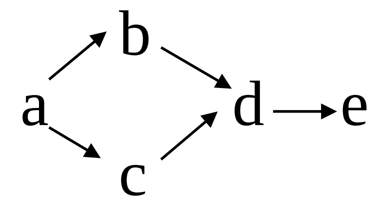
```
big-LA-sales(Buyer,Seller, Price) :-
        person(Buyer, “Los Angeles”,Phone),
        purchase(Buyer, Seller, Product, Price),
        Price > 10000.
```

```
CREATE VIEW  big-LA-sales  AS
   SELECT  buyer, seller, price
   FROM    Person, Purchase
   WHERE   Person.city = “Los Angeles”  AND
           Person.buyer = Purchase.buyer AND
           Purchase.price > 10000
```

[based on slide by Jose Luis Ambite]

Recursion in Datalog

Compute all paths:



path(X, Y) :- arc(X, Y)
path(X, Y) :- path(X, Z), path(Z, Y)

Semantics: evaluate the rules bottom-up until a fixpoint:
Iteration #0: arc: {(a,b), (a,c), (b,d), (c,d), (d,e)}
path: {}
Iteration #1: path: {(a,b), (a,c), (b,d), (c,d), (d,e)}
Iteration #2: path gets the new tuples: (a,d), (b,e), (c,e)
Iteration #3: path gets the new tuple: (a,e)
Iteration #4: Nothing changes => stop.

Basic Database Theory Concepts

- Relational data model
- Queries and answers
- Recursive Queries: Datalog
- Query Containment

Query Containment

- *Query Containment*: $q1 \subseteq q2$
  - $\forall D \;\; q1(D) \subseteq q2(D)$
  - $q1 \models q2$ (q1 logically implies q2)
- *Query Equivalence*: $q1 \equiv q2 \leftrightarrow q1 \subseteq q2 \wedge q1 \subseteq q2$

- Complexity of Query Containment
  - Conjunctive Queries (CQ), Union of CQs: NP-complete
  - CQ with comparisons ($=, <, \neq$): Π^P_2 -complete
  - FOL, recursive queries: Undecidable

Note: containment and equivalence are properties of the queries, not of the database!

Query containment examples

- q1(X,Z) :- p(X,Y,Z)
- q2(X,Z) :- p(X,X,Z)

  q2 $\subseteq$ q1

- q3(X,Y) :- p(X,Z), p(Z,Y), p(X,W)
- q4(X,Y) :- p(X,Z), p(Z,Y)

  q3 $\subseteq$ q4

  q4 $\subseteq$ q3

Query containment is useful

- Query Minimization
  - q1(x) ← r(x,y) ^ r(y,z) ^ r(z,w) ^ r(w,u) ^ r(w,x) ^ r(x,x)
  - q2(x) ← r(x,x)
  - q1(x) = q2(x)
- Reuse previous results (materialized views)
  - q1(x,w) ← r1(x,y) ^ r2(y,z) ^ r3(z,w)
  - q2(x,w) ← r1(x,y) ^ r2(y,z) ^ r3(z,x) ^ r4(x,w)
  - q2(x,w) = q1(x,x) ^ r4(x,w)
- Data Integration !!!

Testing Query Containment

- Two approaches:
  1. Homomorphism/Containment mappings
  2. Canonical databases

  (For CQs both approaches are essentially the same)

- CQ Containment is NP-complete, but since often queries are small, it is not a problem

[based on slide by Ullman]

Conjunctive query evaluation as homomorphism

Assume D is a relational database over a single relation R, and q is a conjunctive query

A tuple X is an answer to query q over database D, X∈q(D) *iff* there exists a homomorphism h from q to D:

- h is a *function* over the variables and constants occurring in q (i.e., it maps to a single element)
- h is the identity on the constants of q
- h maps variables to constants in D
- for each conjunct, if R(X1,..., Xn) ∈ q,
  then R(h(X1), ..., h(Xn)) ∈D

[based on slide by Abiteboul, Manolescu, Rigaux, Rousset, Senellart]

Conjunctive query evaluation: example

q(X,Z) :- r(X,Y), r(Y,Z)

D = { r(1,2), r(1,3), r(2,3), r(2,4) }

(1,3) $\in$q(D)

h = {X$\rightarrow$1, Y$\rightarrow$2, Z$\rightarrow$3}

(1,4) $\in$q(D)

h = {X$\rightarrow$1, Y$\rightarrow$2, Z$\rightarrow$4}

[Chandra & Merlin,1977]

Conjunctive Query Containment: Homomorphism Theorem

$q1 \subseteq q2$ *iff* h(q2)=q1 [q2--CM-->q1]

there exists a homomorphism h from q2 to q1

(i.e., can map all of q2 into q1)

- In this context, the homomorphism is also called a containment mapping
- Note that the containment mapping is in the opposite direction of the containment: it goes from the containing CQ to the contained CQ

Containment Mappings

A mapping from the variables of CQ q2 to the variables of CQ q1, such that:

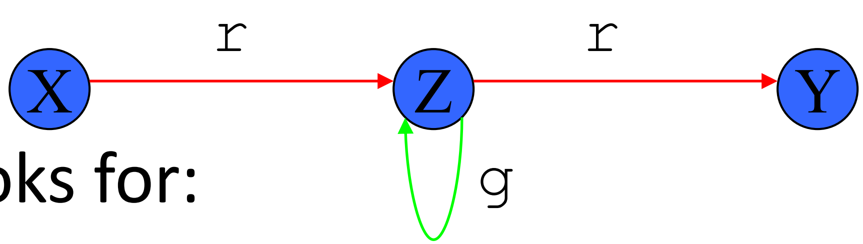
1. head(q2) maps to head(q1)
2. Each subgoal of Q2 maps to some subgoal of Q1 with the same predicate.

Containment Mapping: Example1 (1)

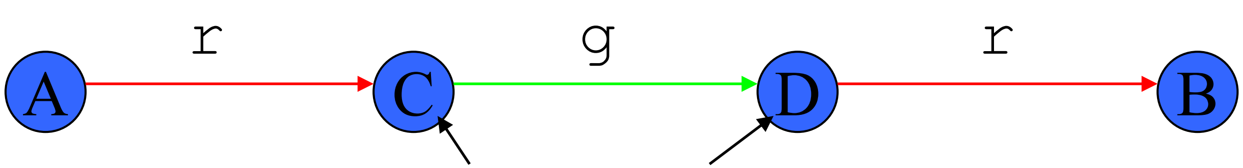
Q1: `p(X,Y):-r(X,Z) & g(Z,Z) & r(Z,Y)`

Q2: `p(A,B):-r(A,C) & g(C,D) & r(D,B)`

Q1 looks for:



Q2 looks for:



Since C=D is possible,
expect Q1 $\subseteq$ Q2

[based on slide by Ullman]

Containment Mapping: Example1 (2)

Q1: `p(X,Y) :-r(X,Z) & g(Z,Z) & r(Z,Y)`

Q2: `p(A,B) :-r(A,C) & g(C,D) & r(D,B)`

Containment mapping m from Q2 to Q1:

m(A)=X; m(B)=Y; m(C)=m(D)=Z

=> Q1 $\subseteq$ Q2

[based on slide by Ullman]

Containment Mapping: Example1 (3)

Q1: `p(X,Y):-r(X,Z) & g(Z,Z) & r(Z,Y)`

Q2: `p(A,B):-r(A,C) & g(C,D) & r(D,B)`

Q2 $\subseteq$ Q1?

- No containment mapping from Q1 to Q2.
  - g(Z,Z) can only be mapped to g(C,D).
    - No other *g* subgoals in Q2.
    - Z must map to both C and D --- impossible (not a function)
- Thus, Q2 $\not\subseteq$ Q1 , Q1 properly contained in Q2.

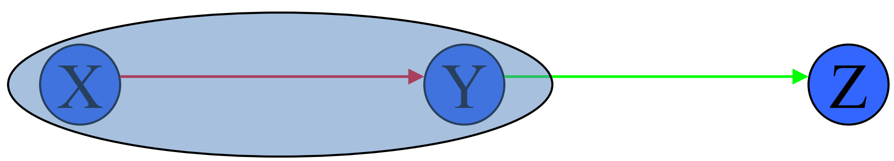
[based on slide by Ullman]

Containment Mapping: Example2 (1)

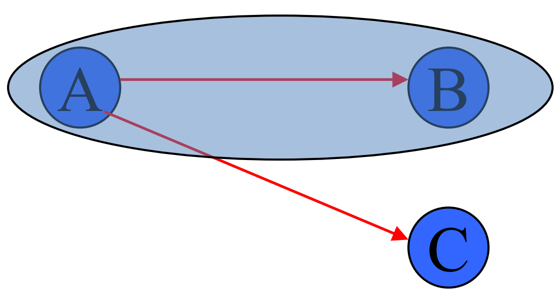
Q1: `p(X,Y):-r(X,Y) & g(Y,Z)`

Q2: `p(A,B):-r(A,B) & r(A,C)`

Q1 looks for:



Q2 looks for:



Containment Mapping: Example2 (2)

Q1: `p(X,Y):-r(X,Y) & g(Y,Z)`

And not every subgoal need be a target.

Q2: `p(A,B):-r(A,B) & r(A,C)`

Notice two subgoals can map to one.

Containment mapping from Q2 to Q1:

m(A)=X; m(B)=m(C)=Y

=> Q1 $\subseteq$ Q2

[based on slide by Ullman]

Containment Mapping: Example2 (3)

Q1: `p(X,Y):-r(X,Y) & g(Y,Z)`
Q2: `p(A,B):-r(A,B) & r(A,C)`

- No containment mapping from Q1 to Q2.
  - g(Y,Z) cannot map anywhere, since there is no *g* subgoal in Q2.
- Thus, Q2 $\not\subseteq$ Q1 , Q1 properly contained in Q2.

[based on slide by Ullman]

Conjunctive Query Containment with Constants

- CQ's are often allowed to have constants in subgoals.
  - Corresponds to selection in relational algebra
- CM's and CM test are the same, but:
  - A variable can map to *one* variable or *one* constant
  - A constant can only map to itself
- Example:

Q2: `p(X) :- e(X,Y)`

Q1: `p(A) :- e(A,10)`

CM from Q2 to Q1 maps X->A and Y->10
Thus, Q1 $\subseteq$ Q2.

A CM from Q1 to Q2 would have to map constant 10 to variable Y; hence no such mapping exists.

[based on slides by Ullman]

Canonical Databases

- General idea: test Q1 $\subseteq$ Q2 by checking that Q1(D_1) $\subseteq$ Q2(D_1),…, Q1(D_n) $\subseteq$ Q2(D_n), where D_1,…,D_n are the canonical databases.

- For the standard CQ case, we only need one canonical DB: the frozen Q1.

- But in more general forms of queries, larger sets of canonical DB's are needed.

[based on slide by Ullman]

Canonical Database (for CQs)

- Canonical database = “frozen query”
  - For each variable of Q, create a corresponding, unique constant
  - Frozen CQ is a DB with one tuple formed from each subgoal of Q, with constants in place of variables
- Example: p(X,Y) :- r(X,Z), g(Z,Z), r(Z,Y)
  - use lower-case letters as constants corresponding to variables
  - Canonical database (“frozen CQ”):
    - Relation R for predicate r = {(x,z), (z,y)} (or r={(1,3), (3,2)})
    - Relation G for predicate g = {(z,z)} (or g={(3,3)})

Query Containment for Conjunctive Queries (and CQ and Datalog)

Method of Canonical Databases

1. Create a canonical database D ("frozen" body of q1)
2. Compute q2(D)
3. If q2(D) contains the "frozen" head of q1, then q1 $\subseteq$ q2, otherwise not.

Containment Mapping: Example1 (2)

Q1: `q(X,Y):-r(X,Z) & g(Z,Z) & r(Z,Y)`

Q2: `q(A,B):-r(A,C) & g(C,D) & r(D,B)`

D_{Q1} = { R= {(x,z) (z,y)} , G={(z,z)}

Q2(D_{Q1}) = {(x,y)}

Frozen-head(Q1) = (x,y) ∈ Q2(D_{Q1})

=> Q1 $\subseteq$ Q2

[based on slide by Ullman]

CQ/Datalog Query Containment: Example

q1 is the CQ:

path(X,Y) :- arc(X,Z), arc(Z,W), arc(W,Y)

q2 is the result of path in the following recursive Datalog program:

path(X,Y) :- arc(X,Y)

path(X,Y) :- path(X,Z), path(Z,Y)

Is q1 $\subseteq$ q2 ?

Intuitively, q1 = paths of length 3; q2 = paths of length 1 or more,

1. Freeze q1 , say with 0, 1, 2, 3 as constants for X, Z, W, Y, respectively

   D = {arc(0, 1), arc(1, 2), arc(2, 3)}

   Frozen head of q1 is path(0, 3).
2. Compute q2(D)   Ext(path) = { (0,1), (1,2), (2,3), (0,2), (1,3), (0,3)}
3. Since frozen head of q1, path(0, 3), is in q2(D) then q1 $\subseteq$ q2

Essentially, we proved that for any set of answer tuples satisfying q1, they also satisfy q2, that is, q1 $\subseteq$ q2

[based on slides by Ullman]

Homomorphism Theorem: Proof
Q2 CM Q1 => Q1 $\subseteq$ Q2

- Assume there is a CM *m* from Q2 to Q1, then we must show that for any db *D*, Q1(*D*) $\subseteq$ Q2(*D*)
- Suppose *t* is a tuple in Q1(*D*); we must show *t* is also in Q2(*D*)

Q2: q(X,Y) :- … p(Y,Z) …

CM m

t=(a,b)∈Q1 Q1: q(U,V) :- … p(V,W) …

Valuation s

Database D: q(a,b) :- … p(a, c) …

Compose m and s => t=(a,b) ∈Q2 => Q1 $\subseteq$ Q2

[based on slide by Ullman]

Homomorphism Theorem: Proof
Q1 $\subseteq$ Q2 => Q2 CM Q1

- Assume Q1 $\subseteq$ Q2, we must show there is a CM from Q2 to Q1
- Consider canonical database D as the frozen Q1

Q1: q(A,B) :- … p(A,C) …

Freeze

Canonical db: q(a,b) :- … p(a, c) … t=(a,b)∈Q1

Valuation s

Q2: q(X,Y) :- … p(X,Z) … t=(a,b)∈Q2

Compose s and "unfreeze" => Q2 CM Q1

[based on slides by Ullman]

Containment: CQ and UCQ

Theorem: a CQ is contained in a UCQ **iff** it is contained in one of the conjunctive queries of the union

- CQ/UCQ containment is still NP-complete
- Example:

$$Q_1(X,Y) :\text{-}\ Flight(X,Z), Flight(Z,Y)$$
$$Q_1(X,Y) :\text{-}\ Flight(X,Z), Flight(Y,Z), Hub(Z)$$

$$Q_2(X,Y) :\text{-}\ Flight(X,Z), Flight(Z,Y), Hub(Z)$$

Q1 CM Q2
Q2 ⊆Q1

[based on slide by Doan, Halevy, Ives]

Conjunctive Queries with Interpreted Predicates: Sufficient, but not necessary

$$Q_1(A,B) :\text{-}\ R(A,B), S(C,D), C \leq D$$

$$Q_2(X,Y) :\text{-}\ R(X,Y), S(U,V), S(V,U)$$

No containment mapping with interpreted predicates,
but $Q_2 \subseteq Q_1$

- Assume U ⩽ V, then Q1 CM Q2: A→X, B→Y, C→U, D→V proves Q2⊆Q1
- Assume U > V, then Q1 CM Q2: A→X, B→Y, C→V, D→U proves Q2⊆Q1

[based on slide by Doan, Halevy, Ives]

Conjunctive Queries with Comparison: Query refinements

$$Q_1(A,B) :- R(A,B), S(C,D), C \pounds D$$

U ⩽ V

$$Q_2(X,Y) :- R(X,Y), S(U,V), S(V,U)$$

U > V

We consider the refinements of Q_2

$$Q_2(X,Y) :- R(X,Y), S(U,V), S(V,U), U \pounds V$$

$$Q_2(X,Y) :- R(X,Y), S(U,V), S(V,U), V < U$$

- If U ⩽ V, then Q1 CM Q2: A→X, B→Y, C→U, D→V proves Q2⊆Q1
- If U > V, then Q1 CM Q2: A→X, B→Y, C→V, D→U proves Q2⊆Q1

[based on slide by Doan, Halevy, Ives]

[Klug, 1988, van der Meyden, 1992]

Containment of Conjunctive Queries with Interpreted Predicates

Q_1 contains Q_2 if and only if there is a containment mapping from Q_1 to every refinement of Q_2.

Deciding whether Q_1 contains Q_2 is $\tilde{\mathrm{O}}_2^p$-*complete* (*)

(*) $\tilde{\mathrm{O}}_2^p$= coNP^{NP} problems are solvable in coNP time assuming we have a NP oracle

http://en.wikipedia.org/wiki/Polynomial_hierarchy

[based on slide by Doan, Halevy, Ives]

[Levy & Sagiv, 1993]

Containment of Conjunctive Queries with Negation

Q1 $\supseteq$ Q2 *if and only if*

Q1(D) $\supseteq$ Q2(D) for all databases D with at most B constants, where B is the total number of variables and constants in Q2

Deciding whether Q1 contains Q2 is $\tilde{O}_2^p$ - *complete*

[based on slide by Doan, Halevy, Ives]

Containment of Conjunctive Queries with Negation: Levy-Sagiv Test

Test Q1 $\subseteq$ Q2 by:

1. Consider the set of all canonical databases *D* such that the tuples of *D* are composed of only symbols 1,2,…,*n*, where *n* is the number of variables and constants of Q1.
2. If there is such a *D* for which Q1(*D*) $\not\subseteq$ Q2(*D*), then Q1 $\not\subseteq$ Q2.
3. Otherwise, Q1 $\subseteq$ Q2.

[based on slides by Ullman]

Containment of Conjunctive Queries with Negation: Levy-Sagiv Test Example

Q1: `p(X,Y) :- a(X,Z) & a(Z,Y) &`
`NOT a(X,Y)`

Q2: `p(X,Y) :- a(X,Y) & NOT a(Y,X)`

- Try *D* = {a(1,2), a(2,3)}
- Q1(*D*) = {p(1,3)}
- Q2(*D*) = {p(1,2), p(2,3)}
- head(Q1) $\notin$ Q2(D). Thus, Q1 $\not\subseteq$ Q2.

Basic Database Theory Concepts

- Relational data model
- Queries and answers
- Recursive Queries: Datalog
- Query Containment