Conjunctive Queries
So far

• The main languages for querying relational databases are:
  - Relational Algebra (RA)
  - Domain Relational Calculus (DRC)
  - Tuple Relational Calculus (TRC) (under the active domain semantics)

RA = DRC = TRC

• Evaluation is decidable, and highly tractable in data complexity
  - Foundations of the database industry
  - The core of SQL is equally expressive to RA/DRC/TRC

• Satisfiability, equivalence and containment are undecidable
  - Perfect query optimization is impossible
A Crucial Question

Are there interesting sublanguages of RA/DRC/TRC for which perfect query optimization is possible?

Conjunctive Queries

= \{\sigma, \pi, \bowtie\}-fragment of relational algebra

= relational calculus without \(\neg\), \(\forall\), \(\vee\)

= simple SELECT-FROM-WHERE SQL queries (only AND and equality in the WHERE clause)
Syntax of Conjunctive Queries (CQ)

\[ Q(x) := \exists y \ (R_1(v_1) \land \cdots \land R_m(v_m)) \]

- \( R_1, \ldots, R_m \) are relations
- \( x, y, v_1, \ldots, v_m \) are tuples of variables
- each variable mentioned in \( v_i \) appears either in \( x \) or \( y \)
- the variables in \( x \) are free called distinguished or output variables

It is very convenient to see conjunctive queries as rule-based queries of the form

\[ Q(x) \ :- \ R_1(v_1), \ldots, R_m(v_m) \]

this is called the **body** of \( Q \) that can be seen as a set of atoms
Conjunctive Queries: Example 1

List all the airlines

\[ \pi_{\text{airline}} \text{ Flight} \]

\{z \mid \exists x \exists y \text{ Flight}(x,y,z)\}

\[ Q(z) \iff \text{Flight}(x,y,z) \]

<table>
<thead>
<tr>
<th>Flight</th>
<th>origin</th>
<th>destination</th>
<th>airline</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>LHR</td>
<td></td>
<td>BA</td>
</tr>
<tr>
<td>LHR</td>
<td>EDI</td>
<td></td>
<td>BA</td>
</tr>
<tr>
<td>LGW</td>
<td>GLA</td>
<td></td>
<td>U2</td>
</tr>
<tr>
<td>LCA</td>
<td>VIE</td>
<td></td>
<td>OS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Airport</th>
<th>code</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>Vienna</td>
<td></td>
</tr>
<tr>
<td>LHR</td>
<td>London</td>
<td></td>
</tr>
<tr>
<td>LGW</td>
<td>London</td>
<td></td>
</tr>
<tr>
<td>LCA</td>
<td>Larnaca</td>
<td></td>
</tr>
<tr>
<td>GLA</td>
<td>Glasgow</td>
<td></td>
</tr>
<tr>
<td>EDI</td>
<td>Edinburgh</td>
<td></td>
</tr>
</tbody>
</table>
Conjunctive Queries: Example 2

List the codes of the airports in London

**Flight** | **origin** | **destination** | **airline**
---|---|---|---
VIE | LHR | BA |
LHR | EDI | BA |
LGW | GLA | U2 |
LCA | VIE | OS |

**Airport** | **code** | **city**
---|---|---
VIE | Vienna |
LHR | London |
LGW | London |
LCA | Larnaca |
GLA | Glasgow |
EDI | Edinburgh |

\[ \pi_{\text{code}} (\sigma_{\text{city='London'}} \text{Airport}) \]

\{x \mid \exists y \text{ Airport}(x,y) \land y = \text{London}\}

**Q(x)** \( \iff \) \text{Airport}(x,y), \( y = \text{London} \)

{LHR, LGW}
Conjunctive Queries: Example 2

List the codes of the airports in London

<table>
<thead>
<tr>
<th>Flight</th>
<th>origin</th>
<th>destination</th>
<th>airline</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>LHR</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LHR</td>
<td>EDI</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LGW</td>
<td>GLA</td>
<td>U2</td>
<td></td>
</tr>
<tr>
<td>LCA</td>
<td>VIE</td>
<td>OS</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Airport</th>
<th>code</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td></td>
<td>Vienna</td>
</tr>
<tr>
<td>LHR</td>
<td></td>
<td>London</td>
</tr>
<tr>
<td>LGW</td>
<td></td>
<td>London</td>
</tr>
<tr>
<td>LCA</td>
<td></td>
<td>Larnaca</td>
</tr>
<tr>
<td>GLA</td>
<td></td>
<td>Glasgow</td>
</tr>
<tr>
<td>EDI</td>
<td></td>
<td>Edinburgh</td>
</tr>
</tbody>
</table>

\[ \pi_{\text{code}}(\sigma_{\text{city}=\text{London}} \text{ Airport}) \]

\[ \{x \mid \exists y \text{ Airport}(x,y) \land y = \text{London}\} \]

Q(x) :- Airport(x,London)
Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

<table>
<thead>
<tr>
<th>Flight</th>
<th>origin</th>
<th>destination</th>
<th>airline</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>LHR</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LHR</td>
<td>EDI</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LGW</td>
<td>GLA</td>
<td>U2</td>
<td></td>
</tr>
<tr>
<td>LCA</td>
<td>VIE</td>
<td>OS</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Airport</th>
<th>code</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>Vienna</td>
<td></td>
</tr>
<tr>
<td>LHR</td>
<td>London</td>
<td></td>
</tr>
<tr>
<td>LGW</td>
<td>London</td>
<td></td>
</tr>
<tr>
<td>LCA</td>
<td>Larnaca</td>
<td></td>
</tr>
<tr>
<td>GLA</td>
<td>Glasgow</td>
<td></td>
</tr>
<tr>
<td>EDI</td>
<td>Edinburgh</td>
<td></td>
</tr>
</tbody>
</table>

$$\pi_{\text{airline}}((\text{Flight} \bowtie_{\text{origin}=\text{code}} (\sigma_{\text{city}='\text{London'}} \text{ Airport})) \bowtie_{\text{destination}=\text{code}} (\sigma_{\text{city}='\text{Glasgow'}} \text{ Airport}))$$

$$\{z \mid \exists x \exists y \exists u \exists v \text{ Airport}(x,u) \land u = \text{London} \land \text{ Airport}(x,u) \land u = \text{London} \} \land \text{ Flight}(x,y,z)$$
Conjunctive Queries: Example 3

List the airlines that fly directly from London to Glasgow

<table>
<thead>
<tr>
<th>Flight</th>
<th>origin</th>
<th>destination</th>
<th>airline</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>LHR</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LHR</td>
<td>EDI</td>
<td>BA</td>
<td></td>
</tr>
<tr>
<td>LGW</td>
<td>GLA</td>
<td>U2</td>
<td></td>
</tr>
<tr>
<td>LCA</td>
<td>VIE</td>
<td>OS</td>
<td></td>
</tr>
</tbody>
</table>

Airport

<table>
<thead>
<tr>
<th>code</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIE</td>
<td>Vienna</td>
</tr>
<tr>
<td>LHR</td>
<td>London</td>
</tr>
<tr>
<td>LGW</td>
<td>London</td>
</tr>
<tr>
<td>LCA</td>
<td>Larnaca</td>
</tr>
<tr>
<td>GLA</td>
<td>Glasgow</td>
</tr>
<tr>
<td>EDI</td>
<td>Edinburgh</td>
</tr>
</tbody>
</table>

\[ Q(z) ::= \text{Airport}(x,\text{London}), \text{Airport}(y,\text{Glasgow}), \text{Flight}(x,y,z) \]
Homomorphism

- Semantics of conjunctive queries via the key notion of homomorphism.

- A substitution from a set of symbols $S$ to a set of symbols $T$ is a function $h : S \rightarrow T$, i.e., $h$ is a set of mappings of the form $s \mapsto t$, where $s \in S$ and $t \in T$.

- A homomorphism from a set of atoms $A$ to a set of atoms $B$ is a substitution $h : \text{terms}(A) \rightarrow \text{terms}(B)$ such that:
  1. $t$ is a constant $\Rightarrow h(t) = t$
  2. $R(t_1, \ldots, t_k) \in A \Rightarrow h(R(t_1, \ldots, t_k)) = R(h(t_1), \ldots, h(t_k)) \in B$

$\text{terms}(A) = \{ t \mid t \text{ is a variable or constant that occurs in } A \}$
Exercise: Find the Homomorphisms

\[ S_1 = \{P(x,y), P(y,z), P(z,w)\} \]

\[ S_2 = \{P(x,y), P(y,z), P(z,x)\} \]

\[ S_3 = \{P(x,y), P(y,x)\} \]

\[ S_4 = \{P(x,x)\} \]

\[ S_5 = \{P(x,y), P(y,x), P(y,y)\} \]
Semantics of Conjunctive Queries

- A **match** of a conjunctive query $Q(x_1,\ldots,x_k) :- \text{body}$ in a database $D$ is a homomorphism $h$ such that $h(\text{body}) \subseteq D$

- The answer to $Q(x_1,\ldots,x_k) :- \text{body}$ over $D$ is the set of $k$-tuples

  $$Q(D) := \{(h(x_1),\ldots,h(x_k)) \mid h \text{ is a match of } Q \text{ in } D\}$$

- The answer consists of the witnesses for the **distinguished variables** of $Q$
Conjunctive Queries: Example

List the airlines that fly directly from London to Glasgow

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

{x \mapsto \text{LGW}, y \mapsto \text{GLA}, z \mapsto \text{U2}}
Complexity of CQ

Theorem: It holds that:

- BQE(CQ) is NP-complete (combined complexity)
- BQE[D](CQ) is NP-complete, for a fixed database D (query complexity)
- BQE[Q](CQ) is in LOGSPACE, for a fixed query Q ∈ CQ (data complexity)

Proof:

(NP-membership) Consider a database D, and a Boolean CQ Q :- body

Guess a substitution h : terms(body) → terms(D)

Verify that h is a match of Q in D, i.e., h(body) ⊆ D

(NP-hardness) Reduction from 3-colorability

(LOGSPACE-membership) Inherited from BQE[Q](DRC)
NP-hardness

(NP-hardness) Reduction from 3-colorability

3COL
Input: an undirected graph $G = (V,E)$
Question: is there a function $c : V \rightarrow \{R,G,B\}$ such that $(v,u) \in E \Rightarrow c(v) \neq c(u)$?

Lemma: $G$ is 3-colorable iff $G$ can be mapped to $K_3$, i.e., $G \xrightarrow{\text{hom}} \begin{array}{c} \text{hom} \\ \end{array}
\begin{array}{c} \begin{array}{c} \circ \\ \circ \\ \circ \\ \end{array} \\ \end{array}
$

therefore, $G$ is 3-colorable iff there is a match of $Q_G$ in $D = \{E(x,y), E(y,z), E(z,x)\}$

the Boolean CQ that represents $G$
Complexity of CQ

Theorem: It holds that:

- \( \text{BQE}(\text{CQ}) \) is NP-complete (combined complexity)
- \( \text{BQE}[D](\text{CQ}) \) is NP-complete, for a fixed database \( D \) (query complexity)
- \( \text{BQE}[Q](\text{CQ}) \) is in LOGSPACE, for a fixed query \( Q \in \text{CQ} \) (data complexity)

Proof:

(NP-membership) Consider a database \( D \), and a Boolean CQ \( Q : - \text{body} \)

Guess a substitution \( h : \text{terms(body)} \rightarrow \text{terms}(D) \)

Verify that \( h \) is a match of \( Q \) in \( D \), i.e., \( h(\text{body}) \subseteq D \)

(NP-hardness) Reduction from 3-colorability

(LOGSPACE-membership) Inherited from \( \text{BQE}[Q](\text{DRC}) \)
What About Optimization of CQs?

SAT(CQ)

Input: a query $Q \in \text{CQ}$

Question: is there a (finite) database $D$ such that $Q(D)$ is non-empty?

EQUIV(CQ)

Input: two queries $Q_1, Q_2 \in \text{CQ}$

Question: $Q_1 \equiv Q_2$? or $Q_1(D) = Q_2(D)$ for every (finite) database $D$?

CONT(CQ)

Input: two queries $Q_1, Q_2 \in \text{CQ}$

Question: $Q_1 \subseteq Q_2$? or $Q_1(D) \subseteq Q_2(D)$ for every (finite) database $D$?
Canonical Database

• Convert a conjunctive query $Q$ into a database $D[Q]$ - the canonical database of $Q$

• Given a conjunctive query of the form $Q(x) :- \text{body}$, $D[Q]$ is obtained from body by replacing each variable $x$ with a new constant $c(x) = x$

• E.g., given $Q(x,y) :- R(x,y), P(y,z,w), R(z,x)$, then $D[Q] = \{R(x,y), P(y,z,w), R(z,x)\}$

• **Note:** The mapping $c : \{\text{variables in body}\} \rightarrow \{\text{new constants}\}$ is a bijection, where $c(\text{body}) = D[Q]$ and $c^{-1}(D[Q]) = \text{body}$
Satisfiability of CQs

**SAT(CQ)**

**Input:** a query $Q \in CQ$

**Question:** is there a (finite) database $D$ such that $Q(D)$ is non-empty?

**Theorem:** A query $Q \in CQ$ is always satisfiable - $SAT(CQ) \in O(1)$-time

**Proof:** Due to its canonical database - $Q(D[Q])$ is trivially non-empty
Equivalence and Containment of CQs

**EQUIV(CQ)**

**Input:** two queries \( Q_1 \in CQ \) and \( Q_2 \in CQ \)

**Question:** \( Q_1 \equiv Q_2 \)? or \( Q_1(D) = Q_2(D) \) for every (finite) database \( D \)?

**CONT(CQ)**

**Input:** two queries \( Q_1 \in CQ \) and \( Q_2 \in CQ \)

**Question:** \( Q_1 \subseteq Q_2 \)? or \( Q_1(D) \subseteq Q_2(D) \) for every (finite) database \( D \)?

\[
Q_1 \equiv Q_2 \iff Q_1 \subseteq Q_2 \text{ and } Q_2 \subseteq Q_1
\]

\[
Q_1 \subseteq Q_2 \iff Q_1 \equiv (Q_1 \land Q_2)
\]

...thus, we can safely focus on CONT(CQ)
Homomorphism Theorem

A query homomorphism from \(Q_1(x_1,\ldots,x_k) :\text{body}_1\) to \(Q_2(y_1,\ldots,y_k) :\text{body}_2\) is a substitution \(h : \text{terms(body}_1) \to \text{terms(body}_2)\) such that:

1. \(h\) is a homomorphism from \(\text{body}_1\) to \(\text{body}_2\)
2. \((h(x_1),\ldots,h(x_k)) = (y_1,\ldots,y_k)\)

**Homomorphism Theorem:** Let \(Q_1\) and \(Q_2\) be conjunctive queries. It holds that:

\[ Q_1 \subseteq Q_2 \iff \text{there exists a query homomorphism from } Q_2 \text{ to } Q_1 \]
Homomorphism Theorem: Example

\[ Q_1(x,y) :\ R(x,z),\ S(z,z),\ R(z,y) \]

\[ Q_2(a,b) :\ R(a,c),\ S(c,d),\ R(d,b) \]

We expect that \( Q_1 \subseteq Q_2 \). Why?
Homomorphism Theorem: Example

$Q_1(x,y) \triangleq R(x,z), S(z,z), R(z,y)$

$Q_2(a,b) \triangleq R(a,c), S(c,d), R(d,b)$

$h = \{a \mapsto x, b \mapsto y, c \mapsto z, d \mapsto z\}$

- $h$ is a query homomorphism from $Q_2$ to $Q_1 \Rightarrow Q_1 \subseteq Q_2$
- But, there is no homomorphism from $Q_1$ to $Q_2 \Rightarrow Q_1 \subset Q_2$
Homomorphism Theorem: Proof

Assume that \( Q_1(x_1, \ldots, x_k) : \text{body}_1 \) and \( Q_2(y_1, \ldots, y_k) : \text{body}_2 \)

\[ (\Rightarrow) \quad Q_1 \subseteq Q_2 \quad \Rightarrow \quad \text{there exists a query homomorphism from } Q_2 \text{ to } Q_1 \]

- Clearly, \((c(x_1), \ldots, c(x_k)) \in Q_1(D[Q_1]) - \text{ recall that } D[Q_1] = c(\text{body}_1)\)
- Since \( Q_1 \subseteq Q_2 \), we conclude that \((c(x_1), \ldots, c(x_k)) \in Q_2(D[Q_1])\)
- Therefore, there exists a homomorphism \( h \) such that \( h(\text{body}_2) \subseteq D[Q_1] = c(\text{body}_1)\) and \( h((y_1, \ldots, y_k)) = (c(x_1), \ldots, c(x_k))\)
- By construction, \( c^{-1}(c(\text{body}_1)) = \text{body}_1\) and \( c^{-1}((c(x_1), \ldots, c(x_k))) = (x_1, \ldots, x_k)\)
- Therefore, \( c^{-1} \circ h \) is a query homomorphism from \( Q_2 \) to \( Q_1 \)
Homomorphism Theorem: Proof

Assume that \( Q_1(x_1, \ldots, x_k) :\text{body}_1 \) and \( Q_2(y_1, \ldots, y_k) :\text{body}_2 \)

\((\Leftarrow)\ Q_1 \subseteq Q_2 \Leftarrow \) there exists a query homomorphism from \( Q_2 \) to \( Q_1 \)

- Consider a database \( D \), and a tuple \( t \) such that \( t \in Q_1(D) \)
- We need to show that \( t \in Q_2(D) \)
- Clearly, there exists a homomorphism \( g \) such that \( g(\text{body}_1) \subseteq D \) and \( g((x_1, \ldots, x_k)) = t \)
- By hypothesis, there exists a query homomorphism \( h \) from \( Q_2 \) to \( Q_1 \)
- Therefore, \( g(h(\text{body}_2)) \subseteq D \) and \( g(h((y_1, \ldots, y_k))) = t \), which implies that \( t \in Q_2(D) \)
Existence of a Query Homomorphism

**Theorem:** Let $Q_1$ and $Q_2$ be conjunctive queries. The problem of deciding whether there exists a query homomorphism from $Q_2$ to $Q_1$ is NP-complete.

**Proof:**

*(NP-membership)* Guess a substitution, and verify that is a query homomorphism.

*(NP-hardness)* Straightforward reduction from BQE(CQ).

By applying the homomorphism theorem we get that:

**Corollary:** EQUIV(CQ) and CONT(CQ) are NP-complete.
Recap

$L \in \{\text{RA}, \text{DRC}, \text{TRC}\}$

UNDECIDABLE

PSPACE

NP

LOGSPACE

$O(1)$-time
Minimizing Conjunctive Queries

• **Goal:** minimize the number of joins in a query

• A conjunctive query $Q_1$ is *minimal* if there is no conjunctive query $Q_2$ such that:
  1. $Q_1 \equiv Q_2$
  2. $Q_2$ has fewer atoms than $Q_1$

• The task of **CQ minimization** is, given a conjunctive query $Q$, to compute a minimal one that is equivalent to $Q$
Minimization by Deletion

By exploiting the homomorphism theorem we can show the following:

**Theorem:** Consider a conjunctive query $Q_1(x_1,\ldots,x_k) :- \text{body}_1$. If $Q_1$ is equivalent to a conjunctive query $Q_2(y_1,\ldots,y_k) :- \text{body}_2$ where $|\text{body}_2| < |\text{body}_1|$, then $Q_1$ is equivalent to a query $Q_3(x_1,\ldots,x_k) :- \text{body}_3$ such that $\text{body}_3 \subseteq \text{body}_1$.

The above theorem says that to minimize a conjunctive query $Q_1(x) :- \text{body}$ we simply need to remove some atoms from $\text{body}$. 
Minimization Procedure

Minimization(Q(x) :- body)

Repeat until no change

    choose an atom α ∈ body

    if there is a query homomorphism from Q(x) :- body to Q(x) :- body \ {α}

    then body := body \ {α}

Return Q(x) :- body

Note: if there is a query homomorphism from Q(x) :- body to Q(x) :- body \ {α},
then the two queries are equivalent since there is trivially a query homomorphism
from the latter to the former query
Minimization Procedure: Example

Q(x) : - R(x,y), R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

{y \mapsto b}
Q(x) : - R(x,b), R(a,b), R(u,c), R(u,v), S(a,c,d)

{v \mapsto c}
Q(x) : - R(x,b), R(a,b), R(u,c), S(a,c,d)

(a,b,c,d are constants)

minimal query

Note: the mapping x \mapsto a is not valid since x is a distinguished variable
Uniqueness of Minimal Queries

**Natural question:** does the order in which we remove atoms from the body of the input conjunctive query matter?

**Theorem:** Consider a conjunctive query $Q$. Let $Q_1$ and $Q_2$ be minimal conjunctive queries such that $Q_1 \equiv Q$ and $Q_2 \equiv Q$. Then, $Q_1$ and $Q_2$ are isomorphic (i.e., they are the same up to variable renaming)

Therefore, given a conjunctive query $Q$, the result of Minimization($Q$) is unique (up to variable renaming) and is called the *core of $Q*
Recap

• The main relational query languages - RA/DRC/TRC
  – Evaluation is decidable - foundations of the database industry
  – Perfect query optimization is impossible

• Conjunctive queries - an important query language
  – All the relevant algorithmic problems are decidable
  – Query minimization

RA = DRC = TRC*

CQ

*under the active domain semantics