Equality-Generating Dependencies

**Definition:** An equality-generating dependency (egd) is a formula of relational calculus of the form:

\[ \forall x_1,\ldots,x_n(\varphi(x_1,\ldots,x_n) \rightarrow x_i = x_j), \]

where \( \varphi(x_1,\ldots,x_n) \) is a conjunction of atomic formulas (i.e., \( \varphi \) is a conjunctive query).

**Examples:**

- \[ \forall x_1,x_2,x_3(R(x_1,x_2) \land P(x_2,x_3) \land T(x_2) \rightarrow x_2 = x_3) \]
  - This is an egd, but not a FD.

- \[ \forall x_1,x_2,x_3(R(x_1,x_2) \land R(x_1,x_3) \rightarrow x_2 = x_3) \]
  - This is both an egd and a FD, namely \( A_1 \rightarrow A_2 \).
Tuple-Generating Dependencies

**Definition:** A tuple-generating dependency (tgd) is a formula of relational calculus of the form:

\[ \forall x_1, \ldots, x_n (\varphi(x_1, \ldots, x_n) \rightarrow \exists y_1, \ldots, y_m \psi(x'_1, \ldots, x'_k, y_1, \ldots, y_m)), \]

where

- \( \varphi(x_1, \ldots, x_n) \) and \( \psi(x'_1, \ldots, x'_k, y_1, \ldots, y_m) \) are conjunctions of atomic formulas
- The variables \( x'_1, \ldots, x'_k \) are among the variables \( x_1, \ldots, x_n \).

**Note:** In effect, a tuple-generating dependency asserts that one conjunctive query (namely, the one defined by \( \varphi(x_1, \ldots, x_n) \)) is contained in another conjunctive query (namely, the one defined by \( \exists y_1, \ldots, y_m \psi(x'_1, \ldots, x'_k, y_1, \ldots, y_m) \)).
Schema Mappings & Data Exchange

- **Schema Mapping** \( M = (S, T, \Sigma) \)
  - *Source* schema \( S \), *Target* schema \( T \)
  - A set \( \Sigma \) of high-level, declarative assertions (constraints) that specify the relationship between \( S \)-instances and \( T \)-instances.

- **Data Exchange** via the schema mapping \( M = (S, T, \Sigma) \)
  Transform a given *source* instance \( I \) to a *target* instance \( J \), so that \((I, J)\) satisfy the specifications \( \Sigma \) of \( M \).
Definition: Schema Mapping  \( M = (S, T, \Sigma) \)

- The existence-of-solutions problem \( \text{Sol}(M) \): (decision problem)
  Given a source instance \( I \), is there a solution \( J \) for \( I \)?

- The data exchange problem associated with \( M \): (function problem)
  Given a source instance \( I \), construct a solution \( J \) for \( I \), provided a solution exists.
The relationship between source and target is given by formulas of relational calculus, called

Source-to-Target Tuple Generating Dependencies (s-t tgds)

\[ \forall x (\varphi(x) \rightarrow \exists y \psi(x, y)), \text{ where} \]

- \( \varphi(x) \) is a conjunction of atoms over the source;
- \( \psi(x, y) \) is a conjunction of atoms over the target;
- \( x \) and \( y \) are tuples of variables.

**Example:**

\( (\text{Student}(s) \land \text{Enrolls}(s,c)) \rightarrow \exists t \exists g (\text{Teaches}(t,c) \land \text{Grade}(s,c,g)) \)

(here, we have dropped the universal quantifiers in front of s-t tgds)
Schema Mapping Specification Language

- s-t tgds assert that: some **conjunctive** query over the source is **contained** in some other **conjunctive** query over the target.

\[(\text{Student}(s) \land \text{Enrolls}(s,c)) \rightarrow \exists t \exists g \ (\text{Teaches}(t,c) \land \text{Grade}(s,c,g))\]

- s-t tgds generalize the main specifications used in data integration:
  - They generalize LAV (**local-as-view**) specifications:
    \[P(x) \rightarrow \exists y \ \psi(x, y), \text{ where } P \text{ is a source schema.} \]
    **Note:** Copy, projection, and decomposition are LAV s-t tgds.
  - They generalize GAV (**global-as-view**) specifications:
    \[\varphi(x) \rightarrow R(x), \text{ where } R \text{ is a target relation} \]
    (they are equivalent to full tgds: \(\varphi(x) \rightarrow \psi(x)\),
    where \(\varphi(x)\) and \(\psi(x)\) are conjunctions of atoms).
    **Note:** Copy, projection, and join are GAV s-t tgds.
Target Dependencies

In addition to source-to-target dependencies, we also consider target dependencies, since, after all, the target schema may have its own integrity constraints:

- **Target Tgds**: $\varphi_T(x) \rightarrow \exists y \psi_T(x, y)$

  Dept (did, dname, mgr_id, mgr_name) → Mgr (mgr_id, did)
  (a target inclusion dependency constraint)

- **Target Equality Generating Dependencies (egds)**:
  $\varphi_T(x) \rightarrow (x_1 = x_2)$

  (Mgr (e, d_1) ∧ Mgr (e, d_2)) → (d_1 = d_2)
  (a target key constraint)
Data Exchange Framework

Schema Mapping $\mathbf{M} = (\mathbf{S}, \mathbf{T}, \Sigma_{st}, \Sigma_t)$, where

- $\Sigma_{st}$ is a set of source-to-target tgds
- $\Sigma_t$ is a set of target tgds and target egds
Underspecification in Data Exchange

- **Fact:** Given a source instance, multiple solutions may exist.

- **Example:**
  Source relation $E(A,B)$, target relation $H(A,B)$
  \[ \Sigma: \quad E(x,y) \rightarrow \exists z \ (H(x,z) \land H(z,y)) \]
  Source instance $I = \{E(a,b)\}$
  **Solutions:** Infinitely many solutions exist
  - $J_1 = \{H(a,b), H(b,b)\}$  
  - $J_2 = \{H(a,a), H(a,b)\}$  
  - $J_3 = \{H(a,X), H(X,b)\}$  
  - $J_4 = \{H(a,X), H(X,b), H(a,Y), H(Y,b)\}$  
  - $J_5 = \{H(a,X), H(X,b), H(Y,Y)\}$

**constants:**
- a, b, ...

**variables (labelled nulls):**
- X, Y, ...
Universal Solutions in Data Exchange

**Definition** (FKMP 2003): A solution is universal if it has homomorphisms to all other solutions (thus, it is a “most general” solution).

- **Constants**: entries in source instances
- **Variables** (labeled nulls): other entries in target instances
- **Homomorphism** $h: J_1 \rightarrow J_2$ between target instances:
  - $h(c) = c$, for constant $c$
  - If $P(a_1,\ldots,a_m)$ is in $J_1$, then $P(h(a_1),\ldots,h(a_m))$ is in $J_2$.

**Claim**: Universal solutions are the *preferred* solutions in data exchange.
Universal Solutions in Data Exchange

\[ \Sigma \]

Schema \( S \)  
Schema \( T \)

Universal Solution

\[ J \]

Solutions

\[ J_1 \]
\[ J_2 \]
\[ J_3 \]

Homomorphisms

\[ h_1 \]
\[ h_2 \]
\[ h_3 \]
Source relation $S(A,B)$, target relation $T(A,B)$

$\Sigma : E(x,y) \rightarrow \exists z \ (H(x,z) \land H(z,y))$

Source instance $I = \{E(a,b)\}$

**Solutions:** Infinitely many solutions exist

- $J_1 = \{H(a,b) , H(b,b)\}$ is not universal
- $J_2 = \{H(a,a), H(a,b)\}$ is not universal
- $J_3 = \{H(a,X) , H(X,b)\}$ is universal
- $J_4 = \{H(a,X), H(X,b) , H(a,Y), H(Y,b)\}$ is universal
- $J_5 = \{H(a,X) , H(X,b) , H(Y,Y)\}$ is not universal
The Existence-of-Solutions Problem

**Question:** What can we say about the existence-of-solutions problem $\text{Sol}(M)$ for a fixed schema mapping $M = (S, T, \Sigma_\text{st}, \Sigma_t)$ specified by s-t tgds and target tgds and egds?

**Answer:** Depending on the target constraints in $\Sigma_t$:
- $\text{Sol}(M)$ can be trivial (solutions always exist).
  ...
- $\text{Sol}(M)$ can be in PTIME.
  ...
- $\text{Sol}(M)$ can be undecidable.
Algorithmic Problems in Data Exchange

**Proposition:** If $M = (S, T, \Sigma_{st}, \Sigma_t)$ is a schema mapping such that $\Sigma_t$ is a set of **full (GAV) target tgds**, then:

- Solutions always exist; hence, $\text{Sol}(M)$ is trivial.

- There is a **Datalog program** $\pi$ over the target $T$ that can be used to compute universal solutions as follows:
  
  Given a source instance $I$,
  
  1. Compute a universal solution $J^*$ for $I$ w.r.t. the schema mapping $M^* = (S, T, \Sigma_{st})$ using the **naïve chase** algorithm.
  2. Run the **Datalog program** $\pi$ on $J^*$ to obtain a universal solution $J$ for $I$ w.r.t. $M$.

  **Note:** The Datalog program $\pi$ is initialized by setting all of its predicates to the values of the relations in $J^*$.

- Consequently, universal solutions can be computed in polynomial time.
The Naïve Chase Algorithm

**Naïve Chase Algorithm** for $M^* = (S, T, \Sigma_{st})$ : given a source instance $I$, build a target instance $J^*$ that satisfies each s-t tgd in $\Sigma_{st}$
- by introducing new facts in $J$ as dictated by the RHS of the s-t tgd and
- by introducing new values (variables) in $J$ each time existential quantifiers need witnesses.

**Example:** $M = (S, T, \Sigma_{st})$ (here $\Sigma_t = \emptyset$)

$\Sigma_{st}$: $E(x, y) \rightarrow \exists z(F(x, z) \land F(z, y))$

The naïve chase returns a relation $F^*$ obtained from $E$ by adding a new node between every edge of $E$.
- If $E = \{(1, 2)\}$, then $F^* = \{(1, N), (N, 2)\}$ Universal solution for $E$
- If $E = \{(1, 2), (2, 3), (1, 4)\}$, then $F^* = \{(1, M), (M, 2), (2, N), (N, 3), (1, U), (U, 4)\}$ Universal solution for $E$
The Naïve Chase Algorithm

Example: Collapsing paths of length 2 to edges

\[ \mathbf{M} = (\mathbf{S}, \mathbf{T}, \Sigma_{\mathit{st}}) \quad (\text{here } \Sigma_{\mathit{t}} = \emptyset) \]

\[ \Sigma_{\mathit{st}} : \quad \mathcal{E}(x,z) \land \mathcal{E}(z,y) \rightarrow \mathcal{F}(x,y) \quad \text{(GAV mapping)} \]

- \( E = \{ (1,3), (2,4), (3,4) \} \)
  \( F^* = \{ F(1,4) \} \) Universal Solution for \( E \)

- \( E = \{ (1,3), (2,4), (3,4), (4,3) \} \)
  \( F^* = \{ (1,4), (2,3), (3,3), (4,4) \} \) Universal solution for \( E \)
**Algorithmic Problems in Data Exchange**

\[ \mathbf{M} = (\mathbf{S}, \mathbf{T}, \Sigma_{\text{st}}, \Sigma_{\text{t}}) \] is a schema mapping such that \( \Sigma_{\text{t}} \) is a set of full (GAV) target tgds:

- Universal solutions can be computed in polynomial time using Naïve chase for \( \Sigma_{\text{st}} \) + Datalog program extracted from \( \Sigma_{\text{t}} \)

**Example:** \( \mathbf{M} = (\mathbf{S}, \mathbf{T}, \Sigma_{\text{st}}, \Sigma_{\text{t}}) \)

\[ \Sigma_{\text{st}}: \ E(x,y) \rightarrow \exists z(F(x,z) \land F(z,y)) \]
\[ \Sigma_{\text{t}}: \ F(u,w) \land F(w,v) \rightarrow F(u,v) \]

1. The naïve chase returns a relation \( F^* \) obtained from \( E \) by adding a new node between every edge of \( E \).
2. The Datalog program \( \pi \) computes the **transitive closure** of \( F^* \).
Algorithmic Problems in Data Exchang

**Proposition:** If $M = (S, T, \Sigma_{str}, \Sigma_t)$ is a schema mapping such that $\Sigma_t$ is a set of **full target tgds** and **target egds**, then:

- Solutions need not always exist.
- The existence-of-solutions problem $\text{Sol}(M)$ is in PTIME, and may be PTIME-complete.

**Proof:** Reduction from **Path Systems**.
Recall the Datalog program:
$T(x) :- A(x)$
$T(x) :- R(x,y,z), T(y), T(z)$.

**Fact:** The following problem is P-complete:
Given sets $A$ and $B$, and a ternary relation $R$, is $B \cap T \neq \emptyset$, where $T$ is the semantics of the above Datalog program.
Question:

What about arbitrary target tgds and egds?
Theorem (K ... , Panttaja, Tan - 2006):
There is a schema mapping $M = (S, T, \Sigma^*_{st}, \Sigma^*_{t})$ such that:

- $\Sigma^*_{st}$ consists of a single source-to-target tgd;
- $\Sigma^*_{t}$ consists of one egd, one full target tgd, and one (non-full) target tgd;
- The existence-of-solutions problem $\text{Sol}(M)$ is undecidable.

**Hint of Proof:**
Reduction from the **Embedding Problem for Finite Semigroups**:
Given a finite partial semigroup, can it be embedded to a finite semigroup?
The Embedding Problem & Data Exchange

Reducing the **Embedding Problem for Semigroups** to **\( \text{Sol}(M) \)**

- \( \Sigma_{st} : R(x,y,z) \rightarrow R'(x,y,z) \)

- \( \Sigma_t : \)
  - \( R' \) is a **partial function**:
    \( R'(x,y,z) \land R'(x,y,w) \rightarrow z = w \)
  
  - \( R' \) is **associative**
    \( R'(x,y,u) \land R'(y,z,v) \land R'(u,z,w) \rightarrow R'(x,u,w) \)
  
  - \( R' \) is a **total function**
    \( R'(x,y,z) \land R'(x',y',z') \rightarrow \exists w_1 \ldots \exists w_9 \)
    
    - \( R'(x,x',w_1) \land R'(x,y',w_2) \land R'(x,z',w_3) \)
    - \( R'(y,x',w_4) \land R'(y,y',w_5) \land R'(x,z',w_6) \)
    - \( R'(z,x',w_7) \land R'(z,y',w_8) \land R'(z,z',w_9) \)
The Existence-of-Solutions Problem

**Summary:** The existence-of-solutions problem

- is **undecidable** for schema mappings in which the target dependencies are arbitrary tgds and egds;
- is in **PTIME** for schema mappings in which the target dependencies are **full** tgds and egds.

**Question:** Are there classes of target tgds **richer** than full tgds and egds for which the existence-of-solutions problem is in **PTIME**?
Algorithmic Properties of Universal Solutions

**Theorem** (FKMP 2003): Schema mapping $\mathbf{M} = (\mathbf{S}, \mathbf{T}, \Sigma_{st}, \Sigma_t)$ such that:

- $\Sigma_{st}$ is a set of source-to-target tgds;
- $\Sigma_t$ is the union of a weakly acyclic set of target tgds with a set of target egds.

Then:

- Universal solutions exist if and only if solutions exist.
- $\text{Sol}(\mathbf{M})$ is in PTIME.
- A *canonical* universal solution (if a solution exists) can be produced in polynomial time using the chase procedure.
Weakly Acyclic Sets of Tgds

Weakly acyclic sets of tgds contain as special cases:

- **Sets of full tgds**
  \[ \varphi_T(x,x') \rightarrow \psi_T(x), \]
  where \( \varphi_T(x,x') \) and \( \psi_T(x) \) are conjunctions of target atoms.

- **Acyclic sets of inclusion dependencies**
  Large class of dependencies occurring in practice.
Weakly Acyclic Sets of Tgds: Definition

- **Position graph** of a set $\Sigma$ of tgds:
  - **Nodes:** $R.A$, with $R$ relation symbol, $A$ attribute of $R$
  - **Edges:** for every $\varphi(x) \rightarrow \exists y \psi(x, y)$ in $\Sigma$, for every $x$ in $x$ occurring in $\psi$, for every occurrence of $x$ in $\varphi$ in $R.A$:
    - For every occurrence of $x$ in $\psi$ in $S.B$, add an edge $R.A \rightarrow S.B$
    - In addition, for every existentially quantified $y$ that occurs in $\psi$ in $T.C$, add a **special edge** $R.A \rightarrow T.C$

- $\Sigma$ is **weakly acyclic** if the position graph has no cycle containing a **special edge**.

- A tgd $\theta$ is **weakly acyclic** if so is the singleton set $\{\theta\}$.
Weakly Acyclic Sets of Tgds: Examples

- **Example 1:** \{ D(e,m) \rightarrow M(m), \ M(m) \rightarrow \exists \ e \ D(e,m) \} is weakly acyclic, but cyclic.

- **Example 2:** \{ E(x,y) \rightarrow \exists \ z \ E(y,z) \} is not weakly acyclic.
Theorem (FKMP): Schema mapping $M = (S, T, \Sigma_{st}, \Sigma_t)$ such that:
- $\Sigma_{st}$ is a set of source-to-target tgds;
- $\Sigma_t$ is the union of a weakly acyclic set of target tgds with a set of target egds.

There is an algorithm, based on the chase procedure, so that:

- Given a source instance $I$, the algorithm determines if a solution for $I$ exists; if so, it produces a canonical universal solution for $I$.

- The running time of the algorithm is polynomial in the size of $I$.

- Hence, the existence-of-solutions problem $\text{Sol}(M)$ for $M$, is in PTIME.
Chase Procedure for Tgds and Egds

Given a source instance I,
1. Use the naïve chase to chase I with $\Sigma_{st}$ and obtain a
target instance $J^*$.
2. Chase $J^*$ with the target tgds and the target egds in $\Sigma_t$ to obtain a target instance $J$
as follows:
   2.1. For target tgds introduce new facts in $J$ as dictated by the RHS of the
        s-t tgd and introduce new values (variables) in $J$ each time existential
        quantifiers need witnesses.
   2.2. For target egds $\phi(x) \rightarrow x_1 = x_2$
        2.2.1. If a variable is equated to a constant, replace the variable by that
                constant;
        2.2.2. If one variable is equated to another variable, replace one
                variable by the other variable.
        2.2.3 If one constant is equated to a different constant, stop and repor
            “failure”.
Weak Acyclicity and the Chase Procedure

Note: If the set of target tgd}s is not weakly acyclic, then the chase may never terminate.

Example: $E(x, y) \rightarrow \exists z \ E(y,z)$ is not weakly acyclic

- $E(1,2)$  \Rightarrow
- $E(2,X_1)$  \Rightarrow
- $E(X_1,X_2)$  \Rightarrow
- $E(X_2, X_3)$  \Rightarrow
- ... infinite chase
The Existence of Solutions Problem

**Summary:** The existence-of-solutions problem

- is undecidable for schema mappings in which the target dependencies are arbitrary tgds and egds;
- is in PTIME for schema mappings in which the set of the target dependencies is the union of a weakly acyclic set of tgds and a set of egds.

**Note:**

- These are *data complexity* results.
- The *combined complexity* of the existence-of-solutions problem is 2EXPTIME-complete (weakly acyclic sets of target tgds and egds).
# The Complexity of the Existence of Solutions Problem

**$M = (S, T, \Sigma_{st}, \Sigma_t)$**

<table>
<thead>
<tr>
<th>$\Sigma_t$:</th>
<th>Existence-of-Solutions Problem</th>
<th>Existence-of-Universal Solutions Problem</th>
<th>Computing a Universal Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full target tgds</td>
<td>Trivial</td>
<td>Trivial</td>
<td>PTIME</td>
</tr>
<tr>
<td>Full target tgds + egds</td>
<td>PTIME</td>
<td>PTIME</td>
<td>PTIME</td>
</tr>
<tr>
<td>Weakly acyclic set of target tgds + egds</td>
<td>PTIME (It can be PTIME-complete)</td>
<td>PTIME (Univ. solutions exist if and only if solutions exist)</td>
<td>PTIME</td>
</tr>
<tr>
<td>Target tgds + egds</td>
<td>Undecidable, in general</td>
<td>Undecidable, in general</td>
<td>No algorithm exists, in general</td>
</tr>
</tbody>
</table>
Outline – Progress Report

- Schema Mappings and Data Exchange: Overview

- Solutions in Data Exchange
  - Universal Solutions
  - Universal Solutions via the Chase

- Query Answering in Data Exchange
Question: What is the semantics of target query answering?

Definition: The certain answers of a query $q$ over $T$ on $I$

$$\text{certain}(q, I) = \bigcap \{ q(J): J \text{ is a solution for } I \}.$$  

Note: It is the standard semantics in data integration.
**Example:** Source relation $E(A,B)$, target relation $H(A,B)$

$$\Sigma: \ E(x,y) \rightarrow \exists z \ (H(x,z) \land H(z,y))$$

Target conjunctive query \(q(x):- H(x,y)\)

Source instance \(I = \{E(a,b)\}\)

**Solutions:** Infinitely many solutions exist

- \(J_1 = \{H(a,b), H(b,b)\}\)  \(q(J_1) = \{a, b\}\)
- \(J_2 = \{H(a,a), H(a,b)\}\)  \(q(J_2) = \{a\}\)
- \(J_3 = \{H(a,X), H(X,b)\}\)  \(q(J_3) = \{a, X\}\)
- \(J_4 = \{H(a,X), H(X,b), H(a,Y), H(Y,b)\}\)  \(q(J_4) = \{a, X, Y\}\)
- \(J_5 = \{H(a,X), H(X,b), H(Z,Z)\}\)  \(q(J_5) = \{a,X,Z\}\)
- ...

\(\text{certain}(q, I) = \cap \{ q(J): J \text{ is a solution for } I \} = \{a\}\)
Certain Answers Semantics

\[
\text{certain}(q,I) = \bigcap \{ q(J) : J \text{ is a solution for } I \}.
\]
Theorem (FKMP): Schema mapping $M = (S, T, \Sigma_{st}, \Sigma_t)$ such that:

- $\Sigma_{st}$ is a set of source-to-target tgds, and
- $\Sigma_t$ is the union of a weakly acyclic set of tgds with a set of egds.

Let $q$ be a union of conjunctive queries over $T$.

- If $I$ is a source instance and $J$ is a universal solution for $I$, then

  $$\text{certain}(q,I) = \text{the set of all "variable-free" tuples in } q(J).$$

Hence, $\text{certain}(q,I)$ is computable in time polynomial in $|I|$:

1. Compute a canonical universal $J$ solution in polynomial time;
2. Evaluate $q(J)$ and remove tuples with “variables”.

Note: This is a data complexity result ($M$ and $q$ are fixed).
Example: Source relation E(A,B), target relation H(A,B)

\[ \Sigma: \quad \text{E}(x,y) \rightarrow \exists z \ (\text{H}(x,z) \land \text{H}(z,y)) \]

Target conjunctive query \( q(x):- \text{H}(x,y) \)

Source instance \( I = \{\text{E}(a,b)\} \)

Solutions: Infinitely many solutions exist

- \( J_1 = \{\text{H}(a,b), \text{H}(b,b)\} \)
  \( q(J_1) = \{a, b\} \)
- \( J_2 = \{\text{H}(a,a), \text{H}(a,b)\} \)
  \( q(J_2) = \{a\} \)
- \( J_3 = \{\text{H}(a,X), \text{H}(X,b)\} \) universal solution
  - \( q(J_3) = \{a, X\} \)
  - Variable-free part of \( q(J_3) = \{a\} = \text{certain}(q,I) \)
Certain Answers via Universal Solutions

\[ \text{certain}(q, I) = \text{set of null-free tuples of } q(J). \]
Computing the Certain Answers

**Theorem (FKMP):** Schema mapping $M = (S, T, \Sigma_{st}, \Sigma_t)$ such that:
- $\Sigma_{st}$ is a set of source-to-target tgds, and
- $\Sigma_t$ is the union of a weakly acyclic set of tgds with a set of egds.

Let $q$ be a union of conjunctive queries with inequalities ($\neq$).
- If $q$ has at most one inequality per conjunct, then $\text{certain}(q, I)$ is computable in time polynomial in $|I|$ using a disjunctive chase.
- If $q$ is has at most two inequalities per conjunct, then $\text{certain}(q, I)$ can be coNP-complete, even if $\Sigma_t = \emptyset$. 
From Theory to Practice

- Clio Project at IBM Almaden:
  - Semi-automatic schema-mapping generation tool;
  - Data exchange system based on schema mappings.

- Universal solutions used as the semantics of data exchange.

- Universal solutions are generated via SQL queries extended with Skolem functions (implementation of chase procedure), provided there are no target constraints.

- Clio technology is now part of IBM Rational® Data Architect.
Some Features of Clio

- Supports **nested** structures
  - Nested Relational Model
  - Nested Constraints
- Automatic & semi-automatic discovery of attribute correspondence.
- Interactive derivation of schema mappings.
- Performs data exchange
Schema Mappings in Clio

Data exchange process (or SQL/XQuery/XSLT)

Source Schema S

"conforms to"

Mapping Generation

Target Schema T

"conforms to"