UNIVERSITÉ DE MONS-HAINAUT
FACULTÉ DES SCIENCES
INSTITUT D'INFORMATIQUE

Research report

in

Computer Science, speciality : Databases

by Raphaël ASTIER

Subject:

On Data Exchange Problems

Supervised by:
Pr. Jozef WIJSEN

Introduction

We refer here a problem which his exposed in all generality in [3]. This is the problem of data exchange between a source database and a target database, according to some dependencies.

The Fagin et al framework ([3]) is the following :

$$\text{Source database schema } \mathbf{S} \longrightarrow \text{Target database schema } \mathbf{T}$$

with sets of dependencies Σ_s, Σ_t on $\mathbf{S}$, $\mathbf{T}$ respectively, and a set of dependencies Σ_{st} being a material representation of the way the data is transferred.
Then this gives the definition of a *data exchange setting* : $(\mathbf{S}, \mathbf{T}, \Sigma_s, \Sigma_t, \Sigma_{st})$, and of the *data exchange problem* :

> Given a (ground) source database $\mathbf{I}$ over the schema $\mathbf{S}$ (that is an instance over $\mathbf{S}$), satisfying Σ_s, finding a target database $\mathbf{J}$ satisfying Σ_t and such that $< \mathbf{I}, \mathbf{J} >$ satisfies Σ_{st}.

Such a $\mathbf{J}$ will be called a *solution* for $\mathbf{I}$ relative to this data exchange setting.

We don't consider here the problem of "data integration" where several source databases may intervene in the data transfer.

Example.

We consider a ground instance $\mathbf{I}$ of a source database schema $\mathbf{S}$ constituted by 3 tables which represent only the first three academic years in UMH's computer science institute, and a target database $\mathbf{T}$ which merges the contents of these 3 tables in one table, according to the filtering described by Σ_{st}.

Year1	Name	Address
	Alain	2, rue des érables
	Sylvie	25, bvd des italiens
	Christian	3, impasse sans nom

Year2	Name	Address	Project
	Jérome	21, quai Anatole	yes
	Alexandre	12, rue du marchepied	no
	Sylvie	3, rue des lilas	yes

Year3	Name	Address	Project
	Judith	7, impasse l'Abbé	yes
	Christophe	1, rue des amandiers	no

$$\Sigma_{st} = \begin{cases} \mathsf{Year1}(x, y) \rightarrow \exists\ z,\ \mathbf{T}(x, \texttt{"1"}, z) \\ \mathsf{Year2}(x, y, z) \rightarrow \mathbf{T}(x, \texttt{"2"}, z) \\ \mathsf{Year3}(x, y, z) \rightarrow \mathbf{T}(x, \texttt{"3"}, z) \end{cases}$$

In this paper, we consider special case where $\Sigma_{st} = Q$ (only one dependency) and which has no existential quantifier (that is "rule-based conjunctive query" or "full tuple generating dependency", see section **3** and definition 4.1 further).

In that case, the condition "$< \mathbf{I}, \mathbf{J} >$ satisfies Σ_{st}" is replaced by $\mathbf{J} = Q(\mathbf{I})$ (see definition 3.1). For this $\mathbf{J}$, being a solution means $\mathbf{J}$ satisfies Σ_t.
In [3], we found theory on the existence and computation of solutions ("universal solutions"). But the dependencies Σ_s on the source are not taken in account: it doesn't matter what they are.
Here we take in account Σ_s to give an algorithm to decide if for all $\mathbf{I}$ over $\mathbf{S}$ satisfying Σ_s, we have that $\mathbf{J} = Q(\mathbf{I})$ satisfies Σ_t. $(\star)$

Returning to our example, assume that Name is a primary key in each relation name Year1, Year2, Year3, that is, assume we have functional dependencies:
Name→ Address, in Year1 relation, and
Name→ Address, Project, in Year2 and Year3 relations.

In terms of first order logical sentences, (used in section **4**), that means:
$\Sigma_s = \{\mathsf{Year1}(x, y),\ \mathsf{Year1}(x, y') \rightarrow y = y',$
$\mathsf{Year2}(x, y, z),\ \mathsf{Year2}(x, y', z') \rightarrow y = y',$
$\mathsf{Year2}(x, y, z),\ \mathsf{Year2}(x, y', z') \rightarrow z = z',$
$\mathsf{Year3}(x, y, z),\ \mathsf{Year3}(x, y', z') \rightarrow y = y',$
$\mathsf{Year3}(x, y, z),\ \mathsf{Year3}(x, y', z') \rightarrow z = z'\}.$

In this paper, we consider only full dependencies, so replace for the moment the first dependency of Σ_{st} by: $\mathsf{Year1}(x, y) \rightarrow \mathbf{T}(x, \texttt{"1"}, \texttt{"no"})$.
Then, our algorithm will answer for $(\star)$ in this example:
- "NO", if $\Sigma_t = \{\mathsf{Name} \rightarrow \mathsf{Project}\}$,
- "YES", if $\Sigma_t = \{\mathsf{Name},\ \mathsf{YearLevel} \rightarrow \mathsf{Project}\}$.

Show briefly the starting point of the problem, that we solve in this paper.
Notice first that the chosen instance $\mathbf{I}$ of the source database verifies Σ_s. Notice also that $Q(\mathbf{I})$ contains the name "Sylvie" in years one and two, respectively with "project" fixed at "no" and "yes". Then $Q(\mathbf{I})$ does not satisfy $\Sigma_t = \{\mathsf{Name} \rightarrow \mathsf{Project}\}$.
Now obviously when $\Sigma_t = \{\mathsf{Name},\ \mathsf{YearLevel} \rightarrow \mathsf{Project}\}$ we have $Q(\mathbf{I})$ satisfies Σ_t.
All the difficulty is to prove this is true for every $\mathbf{I}$ satisfying Σ_s.

1 Preliminaries

We assume enumerable sets of variables and constants, respectively denoted **Var** and **Dom**, and such that $\mathbf{Var} \cap \mathbf{Dom} = \emptyset$.

A *symbol* is either a variable or a constant and we define $\mathbf{Sym} = \mathbf{Var} \cup \mathbf{Dom}$.

A *free-tuple* is a vector $\langle e_1, e_2, \ldots, e_n \rangle$, where $e_1, e_2, \ldots, e_n$ are symbols, and the *arity* of such free-tuple u is denoted $\#u = n$. For such a free-tuple u, we write $u[i] = e_i$ for its i-th coordinate ($i \in \{1, \ldots, n\}$). We write also $\mathrm{Var}(u)$, $\mathrm{Dom}(u)$ and $\mathrm{Sym}(u) = \{e_1, \ldots, e_n\}$ for its respective sets of variables, constants, and symbols.

A *S-atom* is of the form $S(e_1, e_2, \ldots, e_n)$ where $e_1, e_2, \ldots, e_n$ are symbols and where S is a relation name (or if one prefer: "predicate"). More generally, such elements are called *atoms*.

A *database* $\mathbf{D}$ is a finite set of atoms. The *schema* of a database $\mathbf{D}$ is the set of relation names that appear in $\mathbf{D}$, and is denoted $\mathrm{Schem}(\mathbf{D})$. We also write $\mathrm{Var}(\mathbf{D})$, $\mathrm{Dom}(\mathbf{D})$ and $\mathrm{Sym}(\mathbf{D})$ the respective sets of variables, constants, and symbols occuring in $\mathbf{D}$.

An atom or a database without variables is called *ground*. A ground atom is also called a *fact*, and a ground free-tuple is called a *tuple*.

A *substitution* is a function from **Sym** to **Sym** which is identity on **Dom**. A *valuation* is a substitution from **Sym** to **Dom**. For a substitution θ which is not a valuation, in general we specify a set V of variables such that θ is the identity on $\mathbf{Sym} \backslash V$.

Let V a set of variables. A *renaming* for the variables in V is a substitution ρ on V, such that $\rho(V) \subseteq \mathbf{Var}$ and which is injective on V.

For θ a substitution, we define : $\begin{cases} \theta(\langle e_1, e_2, \ldots, e_n \rangle) = \langle \theta(e_1), \theta(e_2), \ldots, \theta(e_n) \rangle, \\ \theta(S(e_1, e_2, \ldots, e_n)) = S(\theta(e_1), \theta(e_2), \ldots, \theta(e_n)), \\ \theta(\mathbf{D}) = \{\theta(L) \mid L \in \mathbf{D}\}. \end{cases}$ **(1.0)**

Lemma 1.1 *Let* $\mathbf{A}$, $\mathbf{B}$ *be databases, and* μ *a substitution.*

1. $\big\{\mathrm{Var}(\mathbf{A}),\ \mathrm{Dom}(\mathbf{A})\big\}$ *is a partition of* $\mathrm{Sym}(\mathbf{A})$.
2. $\mathbf{A} \subseteq \mathbf{B} \Rightarrow \mathrm{Var}(\mathbf{A}) \subseteq \mathrm{Var}(\mathbf{B})$, *idem for* $\mathrm{Dom}(.)$, $\mathrm{Sym}(.)$, *and* $\mu(.)$.
3. $\mathrm{Var}(\mathbf{A} \cup \mathbf{B}) = \mathrm{Var}(\mathbf{A}) \cup \mathrm{Var}(\mathbf{B})$, *idem for* $\mathrm{Dom}(.)$ *and* $\mathrm{Sym}(.)$, *and* $\mu(.)$.
4. $\mathrm{Var}(\mu(\mathbf{A})) \subseteq \mu(\mathrm{Var}(\mathbf{A}))$, $\mathrm{Dom}(\mu(\mathbf{A})) \supseteq \mu(\mathrm{Dom}(\mathbf{A})) = \mathrm{Dom}(\mathbf{A})$ *(the two inclusions becomes equalities when* μ *is a renaming), and* $\mathrm{Sym}(\mu(\mathbf{A})) = \mu(\mathrm{Sym}(\mathbf{A}))$,
5. $\mathrm{Var}(\mathbf{A}) \subseteq \mathrm{Var}(\mathbf{B}) \Rightarrow \mu\big(\mathrm{Var}(\mathbf{A})\big) \subseteq \mu\big(\mathrm{Var}(\mathbf{B})\big)$, *idem for* $\mathrm{Dom}(.)$ *and* $\mathrm{Sym}(.)$.
6. $\mathrm{Var}(\mathbf{A}) \subseteq \mathrm{Var}(\mathbf{B}) \Rightarrow \mathrm{Var}(\mu(\mathbf{A})) \subseteq \mathrm{Var}(\mu(\mathbf{B})$, *idem for* $\mathrm{Sym}(.)$, **not** *for* $\mathrm{Dom}(.)$.

Proof. Everything is obvious from **(1.0)**. Just an example for the "**not**" of *6*:
Consider $\mathbf{A} = \{A(e, x)\}$, $\mathbf{B} = \{B(e, y, z)\}$, and let $\mu[x/a, y/b, z/c]$ (a shorthand for $\mu(x) = a$, $\mu(y) = b$, $\mu(z) = c$). Then $\mathrm{Dom}(\mu(\mathbf{A})) = \{a, e\} \not\subseteq \mathrm{Dom}(\mu(\mathbf{B})) = \{b, c, e\}$, and on the other hand $\mathrm{Dom}(\mathbf{A}) = \{e\} \subseteq \mathrm{Dom}(\mathbf{B}) = \{e\}$.

Lemma 1.2 *Let* $\mathbf{I}$ *be a database and* μ_1, μ_2 *two substitutions.*

Assume $\forall\ x \in \mathrm{Var}(\mathbf{I})$, $\mu_1(x) = \mu_2(x)$. *Then* $\mu_1(\mathbf{I}) = \mu_2(\mathbf{I})$.

Proof. Obvious from **(1.0)**.

Lemma 1.3 *Let $\langle\cdot\rangle$ denote vector concatenation. Let μ_1, μ_2 be two substitutions, and consider two concatenated free-tuples $\mathrm{U} = \langle u_1, u_2, \ldots, u_m\rangle$ and $\mathrm{V} = \langle v_1, v_2, \ldots, v_m\rangle$ (where for $i \in \{1, \ldots, m\}$, u_i and v_i are free-tuples of same arity). Then we have:*

$$\mu_1(\mathrm{U}) = \mu_2(\mathrm{V}) \Longleftrightarrow \forall\ 1 \leqslant i \leqslant m,\ \mu_1(u_i) = \mu_2(v_i).$$

Proof. Obvious from **(1.0)**.

Definition 1.4 *A* homomorphism *from database $\mathbf{C}$ to database $\mathbf{D}$ is a substitution θ for the variables in $\mathbf{C}$ such that $\theta(\mathbf{C}) \subseteq \mathbf{D}$. If such a homomorphism from $\mathbf{C}$ to $\mathbf{D}$ exists, then $\mathbf{C}$ is said to be* homomorphic *to $\mathbf{D}$, and we write $\mathbf{C} \preccurlyeq \mathbf{D}$. The relation $\preccurlyeq$ is a partial preorder (misses antisymmetry). As usual we denote $\succcurlyeq$ for its inverse relation and $\sim$ its associated equivalence relation, which is defined by:*

$$\mathbf{C} \sim \mathbf{D} \Longleftrightarrow \mathbf{C} \preccurlyeq \mathbf{D} \text{ and } \mathbf{C} \succcurlyeq \mathbf{D}.$$

Lemma 1.5 *Let $\mathbf{C}$, $\mathbf{D}$ be databases.*

1. *If $\mathbf{C}$ is ground, then $\mathbf{C} \preccurlyeq \mathbf{D} \Leftrightarrow \mathbf{C} \subseteq \mathbf{D}$.*
2. *Let μ a substitution. Then $Schem\big(\mu(\mathbf{D})\big) = Schem(\mathbf{D})$.*
3. *If $\mathbf{C} \preccurlyeq \mathbf{D}$, then $Schem(\mathbf{C}) \subseteq Schem(\mathbf{D})$.*
4. *Let ρ a renaming on the variables of $\mathbf{D}$. Then $\mathbf{D} \sim \rho(\mathbf{D})$.*

Proof.
1. Obvious, we take θ as the identity.
2. Obvious from (**1.0**).
3. Obvious from *2*.
4. We have $\mathbf{D} \preccurlyeq \rho(\mathbf{D})$ (using substitution ρ), and we have $\mu(\rho(\mathbf{D})) \subseteq \mathbf{D}$ (using substitution μ, which is ρ^{-1} on $\rho(\mathbf{D})$). Then also $\rho(\mathbf{D}) \preccurlyeq \mathbf{D}$. Note that if $\mathbf{D}$ is ground, then ρ is the identity.

Note. In general between two equivalent databases $\mathbf{C} \sim \mathbf{D}$, it doesn't exist a renaming between their respective sets of variables. For example:
$\mathbf{C} = \{T(a), T(x)\}$ and $\mathbf{D} = \{T(a)\}$, are equivalent but without renaming between $\{x\}$ and $\emptyset$.
The same for $\mathbf{C} = \{T(x), T(y)\}$ and $\mathbf{D} = \{T(x)\}$.

2 Unifiers and mgu's[1]

On the set of substitutions, we have a partial preorder defined by:

$$\nu \hookrightarrow \mu \Longleftrightarrow \exists \text{ substitution } \theta \text{ and } \mu = \theta \circ \nu.$$

If $\nu \hookrightarrow \mu$, we say that ν is *more general* than μ.
We write $\hookleftarrow$ the inverse relation of $\hookrightarrow$, and $\approx$ the associated equivalence relation.

Now let u and v two free-tuples of same arity n.
An *unifier* for u and v is a substitution θ on $\mathrm{Var}(u) \cup \mathrm{Var}(v)$ such that $\theta(u) = \theta(v)$.
A *mgu* ("most general unifier") for u and v is a unifier which is more general than each unifier for u and v.

[1] We adapt and enlarge here definitions and results from [1], p. 293-294.

Lemma 2.6 *Let u and v be two free-tuples of same arity.*

1. *A unifier, hence a mgu, for u and v may not exists. But if a unifier exists, then a mgu exists. Moreover, in that case, such mgu θ can be computed in such way that: $\theta\big(\mathrm{Sym}(u) \cup \mathrm{Sym}(v)\big) \subseteq \mathrm{Sym}(u) \cup \mathrm{Sym}(v)$.*
2. *Unifiers and mgus for u and v are not unique. For instance, if θ is a unifier, for each substitution ν, then $\nu \circ \theta$ is a unifier. If moreover θ is mgu, then for each renaming ρ on the variables of $\theta\big(\mathrm{Var}(u) \cup \mathrm{Var}(v)\big)$, then $\rho \circ \theta$ is also a mgu.*
3. *If θ and θ' are mgu for u and v, then $\theta \approx \theta'$.*
4. *If θ is a mgu for u and v and θ' some substitution with $\theta \approx \theta'$, then θ' is also a mgu.*
5. *Let $\theta \approx \theta'$ be two substitutions, and $\mathbf{D}$ a database. Then there exists a renaming ρ for the variables in $\theta(\mathbf{D})$ such that $\rho(\theta(\mathbf{D})) = \theta'(\mathbf{D})$.*

Proof.
1. For example the two tuples $\langle a \rangle$ and $\langle b \rangle$ do not have a unifier (and so no mgu). The computation of a mgu θ for two free-tuples u and v which admit a unifier is exactly similar to that one given in [1], p. 294. *Note:* Even, the construction of [1] guarantees the absence of new variables. The only thing we make use here (in the annexed low-level proof of proposition 3.6) is that we can choose the value of $\theta(x)$ for each $x \in \mathrm{Var}(u) \cup \mathrm{Var}(v)$ as a symbol from u or v.

2. If θ is a unifier for u and v, obviously $\nu \circ \theta$ for ν a substitution is also a unifier for u and v. Assume now that θ is a mgu, and ρ a renaming on $\theta\big(\mathrm{Var}(u) \cup \mathrm{Var}(v)\big)$. We have:

$*$ $\theta(u) = \theta(v) \Rightarrow (\rho \circ \theta)(u) = (\rho \circ \theta)(v)$,

$*$ if ν is a unifier for u and v, a substitution μ such that $\nu = \mu \circ \theta$. Let ρ^{-1} the inverse renaming of ρ, which acts on the variables in $\rho\Big(\theta\big(\mathrm{Var}(u) \cup \mathrm{Var}(v)\big)\Big)$, and consider the substitution $\mu' = \mu \circ \rho^{-1}$. Then: $\nu = \mu' \circ (\rho \circ \theta)$.

Hence we have proved that $\rho \circ \theta$ is a mgu for u and v.

3. Because θ is a mgu, and θ' is a unifier, we have $\theta \hookrightarrow \theta'$. Symmetrically we have $\theta' \hookrightarrow \theta$. Hence by definition of equivalence $\approx$ we have $\theta \approx \theta'$.

4. Since $\theta \approx \theta'$, we have $\theta' = \mu \circ \theta$ for some substitution μ. Then θ' is also unifier for u and v, and we have to verify that θ' is "most general". Let ν a unifier for u and v. We have $\theta' \hookrightarrow \theta$ (because $\theta \approx \theta'$), and $\theta \hookrightarrow \nu$ (because θ is most general). So by transitivity $\theta' \hookrightarrow \nu$ and then we conclude θ' is an mgu.

5. With the assumption $\theta \approx \theta'$, we have ν and ν' substitutions such that $\theta' = \nu \circ \theta$ and $\theta = \nu' \circ \theta'$. For $y = \theta(x) \in \theta\big(\mathrm{Sym}(\mathbf{D})\big)$, we define ρ by $\rho(y) = \nu(y) \;\; (= \theta'(x))$.
We have then a function ρ from $\theta\big(\mathrm{Sym}(\mathbf{D})\big)$ to $\theta'\big(\mathrm{Sym}(\mathbf{D})\big)$.
We claim that this function is injective. Let $y = \theta(x), y' = \theta(x') \in \theta\big(\mathrm{Sym}(\mathbf{D})\big)$, such that $\rho(y) = \rho(y')$. We have $\theta'(x) = \theta'(x') \Rightarrow \nu'(\theta'(x)) = \nu'(\theta'(x'))$, that is $y = \theta(x) = \theta(x') = y'$.
Now, we notice by definition that this function is identity on the constants of $\theta\big(\mathrm{Sym}(\mathbf{D})\big)$, and then we can extend this function to a substitution ρ onto the enumerable set $\mathbf{Sym}$.

It remains to verify that $\rho(\theta(\mathbf{D})) = \theta'(\mathbf{D})$. In fact if $S(e_1, \ldots, e_n)$ is an atom of $\mathbf{D}$, using **(1.0)** and the definition of ρ we have: $\rho(\theta(S(e_1, \ldots, e_n))) = \theta'(S(e_1, \ldots, e_n))$, which proves our assertion.

3 Conjunctive queries

We write conjunctive queries in the form of rules (left-to-right as source-to-target[2]):

$$Q : S_1(u_1), S_2(u_2), \ldots, S_n(u_n) \longrightarrow T(v)$$

where $T(v), S_i(u_i)$'s are atoms and where each variable that occurs in v must occur at least once in some of the u_i's. Notice that some S_i's may be equal, but with different u_i's.
The term *rule* will be used as a shorthand for "rule-based conjunctive query." The semantics of rules on ground databases can be found in [1], page 41. It is straightforward to extend these semantics to non-ground databases (see below the output of a rule).
The *output schema* of the above rule is $\{T\}$, and the *source schema* is $\{S_1, \ldots, S_n\}$.
We denote $\mathbf{S} = \{S_1(u_1), S_2(u_2), \ldots, S_n(u_n)\}$, and $\mathbf{T} = \{T(v)\}$ the source and target databases, and $H = T(v)$ the target T-atom of the rule (this is also the "head" of the rule).

Then the above rule can be simply written as: $\quad Q : \mathbf{S} \longrightarrow H, \quad$ or even $\quad Q : \mathbf{S} \longrightarrow \mathbf{T}$.

Using these notations, previous assertion is then: $\mathrm{Var}(H) = \mathrm{Var}(T(v)) = \mathrm{Var}(\mathbf{T}) \subseteq \mathrm{Var}(\mathbf{S})$.

Definition 3.1 *Let $Q : \mathbf{S} \longrightarrow T(v)$ be a rule, and $\mathbf{I}$ a database.*
We define $Q(\boldsymbol{I}) = \{\mu(T(v)) \mid \mu$ *is a substitution on* $\mathrm{Var}(\mathbf{S})$ *and* $\mu(\boldsymbol{S}) \subseteq \boldsymbol{I}\}$
$= \{\mu(T(v)) \mid \mu$ *is a homomorphism from* $\boldsymbol{S}$ *to* $\boldsymbol{I}\}$.

Lemma 3.2 *Let $Q : \mathbf{S} \longrightarrow T(v)$ be a rule.*

1. *If $\mathbf{I}$ is a ground database, then $Q(\mathbf{I})$ is ground, and we may replace "substitution" by "valuation" in definition of $Q(\mathbf{I})$.*
2. *Monotonicity: if $\mathbf{I}$, $\mathbf{J}$ are databases, $\mathbf{I} \preccurlyeq \mathbf{J} \Rightarrow Q(\mathbf{I}) \preccurlyeq Q(\mathbf{J})$. Of course when $\mathbf{I}$, $\mathbf{J}$ are ground, we have the same with inclusions.*
3. *For $\mathbf{I}$ a database and ν a substitution, we have $\nu(Q(\mathbf{I})) \subseteq Q(\nu(\mathbf{I}))$.*

Proof.
1. Notice that obviously: $\mathbf{I}$ is ground $\Leftrightarrow \mathrm{Var}(\mathbf{I}) = \emptyset \Leftrightarrow \mathrm{Sym}(\mathbf{I}) = \mathrm{Dom}(\mathbf{I})$.
We have by 1.1.*2*: $\mu(\mathbf{S}) \subseteq \mathbf{I} \Rightarrow \mathrm{Var}(\mu(\mathbf{S})) \subseteq \mathrm{Var}(\mathbf{I})$, and recalling $\mathrm{Var}(\mathbf{T}) \subseteq \mathrm{Var}(\mathbf{S})$, then by 1.1.*6*, we get $\mathrm{Var}\big(\mu(\mathbf{T})\big) \subseteq \mathrm{Var}\big(\mu(\mathbf{S})\big)$. But since $\mathrm{Var}(T(v)) = \mathrm{Var}(\mathbf{T})$, obviously: $\mathrm{Var}\big(\mu(T(v))\big) = \mathrm{Var}\big(\mu(\mathbf{T})\big)$, hence:

$$\mathrm{Var}\big(Q(\mathbf{I})\big) = \bigcup_{\substack{\mu \text{ substitution} \\ \text{and } \mu(\mathbf{S}) \subseteq \mathbf{I}}} \mathrm{Var}\big(\mu(T(v))\big) = \bigcup_{\substack{\mu \text{ substitution} \\ \text{and } \mu(\mathbf{S}) \subseteq \mathbf{I}}} \mathrm{Var}\big(\mu(\mathbf{T})\big) = \emptyset.$$

Therefore $Q(\mathbf{I})$ is ground.
Now the fact that we may replace "substitution" by "valuation" in definition of $Q(\mathbf{I})$ is obvious, since variables outside $\mathrm{Var}(\mathbf{S})$ are irrelevant.

2. Note: this result is monotonicity of conjunctive queries, which is proved in [1], 4.2.2 p. 42 for ground databases. For complete reference, we verify this here with general databases.

[2]Cf. Introduction about the "framework" of Fagin.

If $\mathbf{I} \preccurlyeq \mathbf{J}$, we have a substitution ν such that $\nu(\mathbf{I}) \subseteq \mathbf{J}$. We search for a substitution ν' such that $\nu'(Q(\mathbf{I})) \subseteq Q(\mathbf{J})$. Take $\nu' = \nu$. We have:

$\begin{cases} \nu(Q(\mathbf{I})) = \{\nu(\mu(T(v))) \mid \mu \text{ substitution on } \mathrm{Var}(\mathbf{S}) \text{ and } \mu(\mathbf{S}) \subseteq \mathbf{I}\}, \\ Q(\mathbf{J}) = \{\mu'(T(v)) \mid \mu' \text{ substitution on } \mathrm{Var}(\mathbf{S}) \text{ and } \mu'(\mathbf{S}) \subseteq \mathbf{J}\}. \end{cases}$

Let $T(w) = \nu(\mu(T(v)))$ an element of $\nu(Q(\mathbf{I}))$. Define the substitution μ' such that μ' equals $\nu \circ \mu$ on $\mathrm{Var}(\mathbf{S})$, and is identity elsewhere. As $\mathrm{Var}\,(T(v)) \subseteq \mathrm{Var}(\mathbf{S})$, we have $T(w) = \mu'(T(v))$. But we have also μ' on $\mathrm{Var}(\mathbf{S})$ by definition, and $\forall\ x \in \mathrm{Var}(\mathbf{S}),\ \mu'(x) = (\nu \circ \mu)(x)$, then, using lemmas 1.3 and 1.1 we get: $\mu'(\mathbf{S}) = \nu(\mu(\mathbf{S})) \subseteq \nu(\mathbf{I}) \subseteq \mathbf{J}$.
All these conditions show that $T(w) \in Q(\mathbf{J})$, and then we have proved $\nu(Q(\mathbf{I})) \subseteq Q(\mathbf{J})$ as desired.

3. We have $\mathbf{I} \preccurlyeq \nu(\mathbf{I})$, so if we set $\mathbf{J} = \nu(\mathbf{I})$, we have substitution ν such that $\nu(\mathbf{I}) \subseteq \mathbf{J}$. Then previous proof *2* gives us: $\nu(Q(\mathbf{I})) \subseteq Q(\mathbf{J}) = Q(\nu(\mathbf{I}))$.

Lemma 3.3 *Let $Q : \mathbf{S} \longrightarrow \mathbf{T}$ be a rule, and $\mathbf{I}$ a database.*

1. *If $Q(\mathbf{I}) \neq \emptyset$ then the schema of $\mathbf{I}$ contains that one of $\mathbf{S}$.*
2. *If $\mathbf{S} = \{S(u)\}$ (only one relation name), then $Q(\mathbf{S}) = \mathbf{T}$ $\big(= \{T(v)\}\big)$.*
3. *Let ρ a renaming on $\mathrm{Var}(\mathbf{S})$. Recall that $Var\,(\mathbf{T}) \subseteq \mathrm{Var}(\mathbf{S})$.*
Consider the rule $Q_\rho :\ \rho(\mathbf{S}) \longrightarrow \rho(\mathbf{T})$. Then for all database $\mathbf{I}$, we have $Q_\rho(\mathbf{I}) = Q(\mathbf{I})$.

Proof.
1. If $Q(\mathbf{I}) \neq \emptyset$, it exists μ a substitution on $\mathbf{S}$ with $\mu(\mathbf{S}) \subseteq \mathbf{I}$. Then $\mathbf{S} \preccurlyeq \mathbf{I}$ and the result follows from lemma 1.5.*3*.

2. Straightforward using **(1.0)**. (In this case of only one relation name in $\mathbf{S}$, the only possibility for μ substitution on $\mathrm{Var}\,(\mathbf{S})$ to satisfy $\mu(\mathbf{S}) \subseteq \mathbf{S}$ is that μ equals identity).

3. Show first $Q(\mathbf{I}) \subseteq Q_\rho(\mathbf{I})$. Let $y \in Q(\mathbf{I})$, $y = \mu(T(v))$, with μ substitution on $\mathrm{Var}(\mathbf{S})$ such that $\mu(\mathbf{S}) \subseteq \mathbf{I}$. Define a substitution μ' like this: $\forall\ x \in \mathbf{Sym}$, $\mu'(\rho(x)) = \mu(x)$. For this definition being valid, we have to verify that $\forall\ x, x'$, $\rho(x) = \rho(x') \Rightarrow \mu(x) = \mu(x')$, and this is clear since ρ is injective. Then $\mu' \circ \rho = \mu$ and with **(1.0)** we have $y = \mu'(T(\rho(v)))$, with μ' $\big(\text{on } \mathrm{Var}(\rho(\mathbf{S}))\big)$ such that $\mu'(\rho(\mathbf{S})) = \mu(\mathbf{S}) \subseteq \mathbf{I}$, hence $y \in Q_\rho(\mathbf{I})$.
Show now $Q(\mathbf{I}) \supseteq Q_\rho(\mathbf{I})$. Let $y \in Q_\rho(\mathbf{I})$, then $y = \mu'(T(\rho(v)))$, with μ' substitution $\big(\text{on } \mathrm{Var}(\rho(\mathbf{S}))\big)$ such that $\mu'(\rho(\mathbf{S})) \subseteq \mathbf{I}$. It suffices to consider the substitution $\mu = \mu' \circ \rho$.

Definition 3.4 *Let $Q : \mathbf{S} \longrightarrow H = T(v)$ be a rule and $\boldsymbol{D}$ a database with schema $\{T\}$.*
Let V be the set of variables that occur in Q. $\big($Recall $Var\,(H) \subseteq \mathrm{Var}(\mathbf{S})$, hence $V = \mathrm{Var}(\mathbf{S})\big)$.
For every $L \in \mathbf{D}$, let ρ_L be a renaming for the variables in V, such that:

$* \ \forall\ L \in \mathbf{D},\ \rho_L(V) \cap \mathrm{Var}(\mathbf{D}) = \emptyset,$
$* \ \forall\ L_1, L_2 \in \mathbf{D},\ L_1 \neq L_2 \Longrightarrow \rho_{L_1}(V) \cap \rho_{L_2}(V) = \emptyset.$

Consider the set $\boldsymbol{\mathcal{U}} = \{\mu$ substitution $\mid \mu(L) = \mu(\rho_L(H))\ \forall\ L \in \mathbf{D}\}$. Assume there exists $\theta \in \boldsymbol{\mathcal{U}}$ such that θ is more general than every elements of $\boldsymbol{\mathcal{U}}$ (a "most general element").
Let then $\mathbf{C} = \cup\{\rho_L(\mathbf{S}) \mid L \in \mathbf{D}\}$. Then the database $\theta(\mathbf{C})$ is called a Q-source for $\mathbf{D}$.[3]

[3] See also ANNEX, remark-examples 6.2.

Remark 3.5 Write $\boldsymbol{D} = \{T(v_1), T(v_2), \ldots, T(v_m)\}$ for some $m \geqslant 1$, and write then ρ_i the previous renamings ($i = 1, \ldots, m$). Write $\mathrm{v} = \langle v_1, \ldots, v_m \rangle$ and $\mathrm{w} = \langle \rho_1(v), \ldots, \rho_m(v) \rangle$ the concatenate free-tuples.

1. Using **(1.0)** and 1.*3* we have $\boldsymbol{\mathcal{U}} = \{\mu \mid \mu \text{ unifier for } \mathrm{v} \text{ and } \mathrm{w}\}$. Then, using 2.6.*1*: $\boldsymbol{\mathcal{U}} \neq \emptyset \Leftrightarrow \exists$ a mgu for v and w ; and in that case the most general elements of $\boldsymbol{\mathcal{U}}$ are exactly the mgu's for v and w. Also notice that Q-source does not exist when $\boldsymbol{\mathcal{U}} = \emptyset$.

2. We will often use the symbolism $Q^{-1}(\boldsymbol{D})$ for a Q-source of $\mathbf{D}$ (an easy shortand). We can justify this because a Q-source is unique up to renaming of the variables, since lemma 2.6 (points *3* and *5*).

3. We have by 1.1.*3*: $Q^{-1}(\mathbf{D}) = \theta(\bigcup_{L \in \mathbf{D}} \rho_L(\mathbf{S})) = \bigcup_{L \in \mathbf{D}} \theta(\rho_L(\mathbf{S}))$.

Proposition 3.6 *Let $Q : \mathbf{S} \longrightarrow H = T(v)$ be a rule. We consider:*

- $\mathbf{I}$ *a database whose schema contains that one of* $\mathbf{S}$, [4]
- *the database* $\mathbf{J} = Q(\mathbf{I})$,
- $\mathbf{D}$ *a database whose schema is* $\{T\}$ *(the output schema of* Q*),*
- μ *a substitution on* $Var(\mathbf{D})$ *such that* $\mu(\mathbf{D}) \subseteq \mathbf{J}$.

Then a Q-source for $\mathbf{D}$ exists, and we have a substitution α such that:

$$\begin{cases} \imath) & \alpha(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}, \\ \imath\imath) & \mu(\mathbf{D}) \subseteq Q(\alpha(Q^{-1}(\mathbf{D}))). \end{cases}$$

If moreover $\mathbf{I}$ is ground, then $\mathbf{J}$ is ground, and α may be chosen as a valuation.

Proof. We keep notations of definition 3.4: the renamings ρ_L for $L \in \mathbf{D}$, the set $V = \mathrm{Var}(\mathbf{S})$ of variables that occur in Q, and $\boldsymbol{\mathcal{U}} = \{\mu \text{ substitution} \mid \mu(L) = \mu(\rho_L(H)) \ \forall\ L \in \mathbf{D}\}$.

For every $L \in \mathbf{D}$, we have $\mu(L) \in \mu(\mathbf{D}) \subseteq Q(\mathbf{I})$, hence by definition 3.1, for every $L \in \mathbf{D}$, we can assume a substitution ν_L such that $\nu_L(\mathbf{S}) \subseteq \mathbf{I}$ and $\mu(L) = \nu_L(H)$.

For every $L \in \mathbf{D}$, the substitution ρ_L^{-1} is well-defined since ρ_L is injective, and therefore we can define substitution on $\rho_L(V)$ by $\phi_L(u) = \nu_L(\rho_L^{-1}(u))$. That is, for every variable $w \in V$, $\phi_L(\rho_L(w)) = \nu_L(w)$. Now define $\phi = \cup\{\phi_L \mid L \in \mathbf{D}\}$, the substitution which equals ϕ_L on each set of variables $\rho_L(V)$. ϕ is well-defined since these sets of variables are disjoints by construction of renamings ρ_L. Then, $\forall\ L \in \mathbf{D}$ we have $\nu_L(H) = \phi_L(\rho_L(H)) = \phi(\rho_L(H))$.
It follows $\forall\ L \in \mathbf{D}$, $\phi(\rho_L(H)) = \mu(L)$. Consider substitution $\omega = \phi \cup \mu$, which equals μ on $\mathrm{Var}(\mathbf{D})$, equals ϕ on sets of variables $\rho_L(V)$, and is identity elsewhere. ω is well-defined since by construction of renamings, the sets $\rho_L(V)$ are disjoints from $\mathrm{Var}(\mathbf{D})$.
Then, $\forall\ L \in \mathbf{D}$, $\omega(L) = \mu(L)$, and also $\omega(\rho_L(w)) = \phi(\rho_L(w))$ for every variable $w \in V$.
Since $\mathrm{Var}(H) \subseteq V$, this last equality gives us in particular $\omega(\rho_L(H)) = \phi(\rho_L(H))$, and then: $\forall\ L \in \mathbf{D}$, $\omega(\rho_L(H)) = \omega(L)$, and so $\omega \in \boldsymbol{\mathcal{U}}$.
Therefore using 3.5.1, we have θ a most general element of $\boldsymbol{\mathcal{U}}$ (that is a mgu for v and w), hence we have a Q-source for $\mathbf{D}$.
Then substitution α comes naturally: since θ is more general than ω, we can assume a substitution α such that $\omega = \alpha \circ \theta$.
Now we prove points $\imath$) and $\imath\imath$).

[4] We may discard this assumption: if $\mathrm{Schem}(\mathbf{S}) \not\subseteq \mathrm{Schem}(\mathbf{I})$, then $\mathbf{J} = Q(\mathbf{I}) = \emptyset$, and "$\exists\ \mu$ such that $\mu(\mathbf{D}) \subseteq \mathbf{J}$" implies $\mathbf{D} = \emptyset$, then $Q^{-1}(\mathbf{D}) = \emptyset$, and assertions of the proposition are in this case trivial ones.

i) Fix $L \in \mathbf{D}$. Using definition of α, we have $\omega(\rho_L(\mathbf{S})) = \alpha(\theta(\rho_L(\mathbf{S})))$. On the other hand, we have seen, for every $w \in V$, $\omega(\rho_L(w)) = \phi_L(\rho_L(w))$, and then from 1.2, we get $\omega(\rho_L(\mathbf{S})) = \phi_L(\rho_L(\mathbf{S}))$. But for every $w \in V$, we have also $\phi_L(\rho_L(w)) = \nu_L(w)$, and then $\phi_L(\rho_L(\mathbf{S})) = \nu_L(\mathbf{S})$. By transitivity, $\alpha(\theta(\rho_L(\mathbf{S}))) = \nu_L(\mathbf{S})$. Now by definition of ν_L we have $\nu_L(\mathbf{S}) \subseteq \mathbf{I}$, and then, using 3.5.3, we deduce $\alpha(Q^{-1}(\mathbf{D})) = \bigcup_{L \in \mathbf{D}} \alpha\big(\theta(\rho_L(\mathbf{S}))\big) \subseteq \mathbf{I}$.

ii) Fix again $L \in \mathbf{D}$. We have seen $\mu(L) = \phi(\rho_L(H)) = \omega(\rho_L(H)) = \alpha(\theta(\rho_L(H)))$ (the last equality by definition of α). Let β the substitution which equals $\theta \circ \rho_L$ on V, and is identity elsewhere. Since $\mathrm{Var}(H) \subseteq V$ and $T(v) = H$, we have $\mu(L) = \alpha(\beta(T(v)))$. But, by 1.2 and 3.5.3, we have also $\beta(\mathbf{S}) = (\theta \circ \rho_L)(\mathbf{S}) \subseteq Q^{-1}(\mathbf{D}) = \bigcup_{L \in \mathbf{D}} \theta(\rho_L(\mathbf{S}))$.

Then, from definition 3.1, we have $\beta(T(v)) \in Q(Q^{-1}(\mathbf{D}))$, so $\mu(L) \in \alpha(Q(Q^{-1}(\mathbf{D})))$.

Since this is true for any $L \in \mathbf{D}$, we have then shown $\mu(\mathbf{D}) \subseteq \alpha\big(Q(Q^{-1}(\mathbf{D}))\big)$.

Therefore the result follows from lemma 3.2.*3*: $\alpha\big(Q(Q^{-1}(\mathbf{D}))\big) \subseteq Q(\alpha(Q^{-1}(\mathbf{D})))$.

For conclusion of this proposition, it remains to see what happens if $\mathbf{I}$ is ground.
The fact that $Q(\mathbf{I})$ is ground follows from lemma 3.2.*1*.

Now we can define substitution α' which equals α on $\mathrm{Var}(Q^{-1}(\mathbf{D}))$, and which equals some arbitrary constant on other variables.
Then $\alpha'(Q^{-1}(\mathbf{D})) = \alpha(Q^{-1}(\mathbf{D}))$, so *i)* and *ii)* remain true for α', and since $\alpha(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}$ with $\mathbf{I}$ ground, the values of α' on $\mathrm{Var}(Q^{-1}(\mathbf{D}))$ are constants. Then α' is a valuation.
And therefore α can actually be selected as a valuation (as α').

4 Constraints

We adapt here definitions and lemmas from [1], p. 216-219 (dependencies) and p. 173-177 (chase), but also from [2], section 3. Notice that here "dependency" is a synonym for "constraint". We consider here only dependencies which are full and single-head. This includes all classical dependencies (except inclusion dependencies) such as functional, multi-valued, and join dependencies. As in [2], dependencies may contain constants.

Definition 4.1 *A full (single-head) dependency σ is a first-order logic sentence, that we write:*

$$\sigma : \ \mathbf{B} \rightarrow K$$

where the "body" $\mathbf{B} = \{B_1(v_1), \ldots, B_m(v_m)\}$ is a database, and the head K is:

- *either $T(w)$ a T-atom with $\mathrm{Var}(T(w)) \subseteq \mathrm{Var}(\mathbf{B})$, in this case we call it a "ftgd" (full tuple generating dependency),*
- *either $e_i = f_j$ with $e_i, f_j \in \mathrm{Var}(\mathbf{B}) \cup \mathbf{Dom}$, in this case we call it a "fegd" (full equality generating dependency).*

Notice that in fact a ftgd is nothing but a rule.
As a shorthand for full single head dependency, we will use "dependency", or "constraint".

Definition 4.2 *Let σ be a constraint, with previous notations. We say that a ground database $\mathbf{I}$ satisfies σ, and we write $\mathbf{I} \models \sigma$, when for each substitution μ such that $\mu(\mathbf{B}) \subseteq \mathbf{I}$, we have:*

- *if σ is ftgd, then $\mu(T(w)) \in \mathbf{I}$,*
- *if σ is fegd, then $\mu(e_i) = \mu(f_j)$.*

Let Σ a set of constraints. We say that a ground database $\mathbf{I}$ satisfies Σ, and we write $\mathbf{I} \models \Sigma$, if for all $\sigma \in \Sigma$, we have $\mathbf{I} \models \sigma$.

Remark 4.3

1. We only consider satisfaction for **ground** databases.

2. In the preceding definition, we can impose that substitutions μ are on $\mathrm{Var}(\mathbf{B})$ since others variables are irrelevant.
Moreover, since $\mathbf{I}$ is supposed to be ground, we may replace μ "substitution" by μ "valuation".

3. When there is no substitution μ such that $\mu(\mathbf{B}) \subseteq \mathbf{I}$, for example when $\mathrm{Schem}(\mathbf{B}) \not\subseteq \mathrm{Schem}(\mathbf{I})$ (cf. lemma 1.5.*2*), then the definition says that $\mathbf{I} \models \sigma$. A particular example of this is when $\mathbf{B} \neq \emptyset$ and $\mathbf{I} = \emptyset$. Then $\mathbf{I} \models \sigma$. Assume now $\mathbf{B} = \emptyset$. By our definition, σ must be a ftgd $\sigma : \{\} \to T(w)$, where w is a tuple (no variables). Then let $\mathbf{I}$ be a ground database. We have obviously $\mathbf{I} \models \sigma \iff \mathbf{I} \supseteq \{T(w)\}$, (hence any ground database $\mathbf{I}$ satisfying σ is nonempty).

4. Let σ be a ftgd, and $\mathbf{I}$ be a ground database. Using definition 3.1 and previous point 2, we have: $\sigma(\mathbf{I}) = \{\mu(T(w)) \mid \mu \text{ is a substitution on } \mathrm{Var}(\mathbf{B}) \text{ and } \mu(\boldsymbol{B}) \subseteq \boldsymbol{I}\}$. Then obviously:

$$\mathbf{I} \models \sigma \iff \sigma(\mathbf{I}) \subseteq \mathbf{I}.$$

One can also write this is equivalent to: $\mathbf{I} \cup \sigma(\mathbf{I}) = \mathbf{I}$, which is exactly – in case of ground databases, and except that here $\mathbf{B}$ need not to be unirelational – definition 6 of [2] concerning satisfaction of constraints.

5. Let now σ be a fegd. Then $\mathbf{I} \models \sigma \iff$ there is no valuation μ such that $\mu(\mathbf{B}) \subseteq \mathbf{I}$ and $\mu(e_i) \neq \mu(f_j)$. On the contrary, $\mathbf{I} \not\models \sigma \iff$ there is a valuation μ such that $\mu(\mathbf{B}) \subseteq \mathbf{I}$ and $\mu(e_i) \neq \mu(f_j)$. Again, this is exactly – in case of ground databases, and except that here $\mathbf{B}$ need not to be unirelational – definition 6 of [2].

Lemma 4.4 *Let $\sigma : \mathbf{B} \to K$ be a dependency, and $\mathbf{J}$ be a ground database.*
Assume that for all μ valuation such that $\mu(\mathbf{B}) \subseteq \mathbf{J}$, there exists a (necessarily ground) database $\mathbf{J}_\mu$ satisfying σ and such that $\mu(\mathbf{B}) \subseteq \mathbf{J}_\mu \subseteq \mathbf{J}$. Then $\mathbf{J}$ also satisfy σ.

Proof. In case where there is no valuation μ such that $\mu(\mathbf{B}) \subseteq \mathbf{J}$, the result is obvious since previous remark 4.3.3.
Assume there exists valuation(s) μ such that $\mu(\mathbf{B}) \subseteq \mathbf{J}$. We want to verify that for such a valuation we have:

- if σ is ftgd, then $\mu(T(w)) \in \mathbf{J}$,
- if σ is fegd, then $\mu(e_i) = \mu(f_j)$.

But by hypothesis we have $\mu(\mathbf{B}) \subseteq \mathbf{J}_\mu$, and J_μ satisfy σ, then by definition of satisfaction:
if σ is ftgd, then $\mu(T(w)) \in \mathbf{J}_\mu$, and if σ is fegd, then $\mu(e_i) = \mu(f_j)$.
Now since $\mathbf{J}_\mu \subseteq \mathbf{J}$, the verification is done.

Note. If $\mathbf{J}_\mu \subseteq \mathbf{J}$ databases, with $\mathbf{J}$ ground, then $\mathbf{J}_\mu$ is necessarily ground since in that case, using 1.1.*2*, we have $\mathrm{Var}(\mathbf{J}_\mu) \subseteq \mathrm{Var}(\mathbf{J}) = \emptyset$.

Definition 4.5 *Applying dependencies (direct adaptation of definition 7 from [2]).*

We consider here the set of ground databases $\mathcal{G}$ partially ordered by inclusion, and we extend it with $\square$ an artificial element which contains (for the extended inclusion order) any element of $\mathcal{G}$ (as in [2], def. 7). For convenience $\square$ is also called "ground database" (but the notion of satisfaction is not defined for $\square$).

Let $\sigma : \mathbf{B} \to K$ be a constraint (with previous notations), and $\mathbf{I} \neq \square$ be a ground database. We set $\boldsymbol{\mathcal{W}} = \{\mu$ valuations such that $\mu(\mathbf{B}) \subseteq \mathbf{I}\}$.

- ***a)*** *In case where σ is a ftgd, we set the result of* applying σ to $\mathbf{I}$ *to be the (ground) database $\mathbf{I}' = \mathbf{I} \cup \sigma(\mathbf{I})$. Notice that if $\boldsymbol{\mathcal{W}} = \emptyset$, we have $\sigma(\mathbf{I}) = \emptyset$, then $\mathbf{I}' = \mathbf{I}$, and on the other hand we have already said (remark 4.3.3) that in this case $\mathbf{I} \models \sigma$.*
- ***b)*** *In case where σ is a fegd, we set the result of applying σ to $\mathbf{I}$ to be the (ground) database $\mathbf{I}' = \begin{cases} \square \text{ if } \mathbf{I} \not\models \sigma, \\ \mathbf{I} \text{ if } \mathbf{I} \models \sigma \end{cases}$ (for instance when $\boldsymbol{\mathcal{W}} = \emptyset$).*

Lemma 4.6 *Let $\mathbf{I} \neq \square$ be a ground database and σ a dependency.*

- ***a)*** *If $\mathbf{I} \models \sigma$, then applying σ on $\mathbf{I}$ produces $\mathbf{I}$.*
- ***b)*** *Let $\mathbf{I}'$ be the result of applying σ on $\mathbf{I}$. We have $\mathbf{I} \subseteq \mathbf{I}'$.*

Proof.
a) This is obvious for σ a fegd, and this is a direct consequence from remark 4.3.4 for σ a ftgd.
b) Obvious with previous definition.

Definition 4.5 (continuation) Chasing.

Now let Σ be a set of constraints. A chasing sequence $(\mathbf{I}_k)$ *of $\mathbf{I}$ by Σ is a set of databases where $\mathbf{I}_0 = \mathbf{I}$, and where each $\mathbf{I}_{k+1}$ is obtained from $\mathbf{I}_k$ by applying some dependency from Σ. We say that a chasing sequence is finite when we cannot find a dependency $\sigma \in \Sigma$ which produces a new database when applied to the last element of the sequence. In that case, the length of this sequence is trivially the number of distinct databases in the sequence, and is written:* $\text{length}\big((\mathbf{I}_k)_{k=0,\ldots,n}\big)$ *(where $\mathbf{I}_k \subsetneq \mathbf{I}_{k+1}$ for all $k = 0, \ldots, n$).*

Note. In [1], p. 174-175, the chasing procedure is defined on tableau queries (without constants), and extended in ex. 8.27 p. 190 to tableau queries with constants: if the chasing procedure tries to identify two distinct constants, the result is a new tableau query denoted T_{false} which corresponds to the query producing empty output whatever input instance. In [2] (def. 7), the summary of a tableau query is dropped, hence the chase procedure applies to general tableaux (including constants and variables). Again when the chase tries to identify two distinct constants, then the resulting tableau is materialized by a new symbol $\square$ which is assumed to be a top order element of the set of tableaux for the partial preorder $\preccurlyeq$.

Here we apply the chase procedure to databases, hence not only to tableaux (unirelational databases), but only on **ground** databases. As we will see, all lemmas and theorems from [2] section 3.2, apply almost directly in our case.

Definition-theorem 4.7 *Let Σ be a set of constraints and $\mathbf{I} \neq \square$ be a ground database.*

1. *All chasing sequences of $\mathbf{I}$ by Σ are finite.*
2. *All chasing sequences of $\mathbf{I}$ by Σ, which are maximal with respect to length, terminate with the same last element. We write this one $Chase_\Sigma(\mathbf{I})$.*

Proof. This is particular case from [2] (lemma 6 and theorem 2). It suffices to replace "$\preccurlyeq$" by "$\subseteq$" and "$\sim$" by "$=$" (and of course to be convinced this is legal). Point **2** is the Church-Rosser property. In fact theorem 2 follows from lemma 6 and also from lemmas 7 and 8 which are true in our special case.

Example 4.8 We take $\mathbf{I} = \emptyset$.

Let $\Sigma = \{\sigma_0, \sigma_1\}$, with $\sigma_0 : \{\} \to T(a, b)$, and $\sigma_1 : T(x, y) \to x = y$.
Then the only maximal chasing sequence is: $\mathbf{I} = \{\} \subsetneq \{T(a, b)\} \subsetneq \square$. And $\text{Chase}_\Sigma(\mathbf{I}) = \square$.
Notice that if we have taken $\emptyset$ instead of $\square$ in definition 4.5, then the chasing procedure never terminates in this example.

Lemma 4.9 *Let Σ be a set of constraints and $\mathbf{I} \neq \square$ a ground database.*

1. $\mathbf{I} \subseteq \text{Chase}_\Sigma(\mathbf{I})$.
2. *If $\mathbf{I} \models \Sigma$, then $\text{Chase}_\Sigma(\mathbf{I}) = \mathbf{I}$ (and all chasing sequences are of length zero).*
3. *If $\text{Chase}_\Sigma(\mathbf{I}) \neq \square$, then $\text{Chase}_\Sigma(\mathbf{I}) \models \Sigma$.*
4. *If $\mathbf{J} \neq \square$ is a ground database which satisfy Σ and such that $\mathbf{I} \subseteq \mathbf{J}$, then $\text{Chase}_\Sigma(\mathbf{I}) \neq \square$ and $\text{Chase}_\Sigma(\mathbf{I}) \subseteq \mathbf{J}$.*

Proof.
1. Direct consequence from lemma 4.6 b). (Or lemma 6 point 1 from [2]).
2. Direct consequence from lemma 4.6 a).
3. This results immediately from lemma 6 point 2 of [2].
4. We prove this result explicitly here, since it is slightly different from lemma 7 of [2]. (But this is the same idea).
Let $\mathbf{I} = \mathbf{I}_0 \subseteq \mathbf{I}_1 \subseteq \mathbf{I}_2 \subseteq \ldots \subseteq \mathbf{I}_k \subseteq \mathbf{I}_{k+1} \subseteq \ldots$ a chasing sequence starting from $\mathbf{I}$. Raisoning by induction, we assume $\mathbf{I}_k \neq \square$ and $\mathbf{I}_k \subseteq \mathbf{J}$. (this is true for $\mathbf{I} = \mathbf{I}_0$). It suffices to verify the same for $\mathbf{I}_{k+1}$. Let $\sigma : \mathbf{B} \to K$, $\sigma \in \Sigma$, be the dependency which produces $\mathbf{I}_{k+1}$ from $\mathbf{I}_k$. If σ is a fegd with K replaced by $e_i = f_j$, by remark 4.3.5: $\mathbf{I}_k \not\models \sigma$ signify that $\exists$ valuation μ such that $\mu(D) \subseteq \mathbf{I}_k$ and $\mu(e_i) \neq \mu(f_j)$. As $\mathbf{I}_k \subseteq \mathbf{J}$ we have same valuation μ such that $\mu(\mathbf{B}) \subseteq \mathbf{J}$, and $\mu(e_i) \neq \mu(f_j)$, that is: $\mathbf{J} \not\models \sigma$, which is impossible since by hypothesis $\mathbf{J} \models \Sigma$. Hence in case σ a fegd, we have $\mathbf{I}_{k+1} = \mathbf{I}_k$.
Now assume σ is a ftgd, so that by definition $\mathbf{I}_{k+1} = \mathbf{I}_k \cup \sigma(\mathbf{I}_k)$.
We have $\mathbf{I}_k \subseteq \mathbf{J}$ and then $\sigma(\mathbf{I}_k) \subseteq \sigma(\mathbf{J})$ (lemma 3.2.2). On the other hand, since $\mathbf{J} \models \sigma$, we have also $\sigma(\mathbf{J}) \subseteq \mathbf{J}$ (remark 4.3.4), and therefore $\mathbf{I}_{k+1} \subseteq \mathbf{J}$, which proves our assertion.

Definition 4.10 *Let C be a set of constants. A* constants'renaming *for the constants in C is a one to one function f from C to $f(C)$, that we extend to an injective function from* **Dom** *to* **Dom**. *Given f a constants'renaming, we define $f(X)$ for X a tuple, a ground atom or a ground database, exactly in the same way of* **(1.0)**.

Lemma 4.11 *Let* $\mathbf{I}$ *and* $\mathbf{J}$ *be ground databases, both distinct from* $\square$.
Assume there exists a constants'renaming f *from* $\mathrm{Dom}(\mathbf{I})$ *to* $\mathrm{Dom}(\mathbf{J})$*, that we extend to an injective function on* $\mathbf{Dom}$ *and such that* $f(\mathbf{I}) = \mathbf{J}$. *For convenience we write in that case:* $\mathbf{I} \simeq \mathbf{J}$.

1. *Let* $Q : \mathbf{S} \to T(v)$ *be a rule. Assume that* f *is identity on* $\mathrm{Dom}(Q)$*, that is on all constants occuring in* Q. *Then* $Q(\mathbf{I}) \simeq Q(\mathbf{J})$.
2. *Let* $\sigma : \mathbf{B} \to K$ *be a constraint. Assume* f *is identity on every constant occuring in* σ. *Then* $\mathbf{I} \models \sigma \Longleftrightarrow \mathbf{J} \models \sigma$.
3. *Let* Σ *be a set of constraints. Assume* f *is identity on every constant occuring in the* Σ*'s constraints. Then* $Chase_{\Sigma}(\mathbf{I}) \simeq Chase_{\Sigma}(\mathbf{J})$.[5]

Proof.
1. Recall definition 3.1 with lemma 3.2: $Q(\mathbf{I}) = \{\nu(T(v)) \mid \nu \text{ valuation with } \nu(\mathbf{S}) \subseteq \mathbf{I}\}$.
Then $f(Q(\mathbf{I})) = \{f(\nu(T(v))) \mid \nu \text{ valuation with } \nu(\mathbf{S}) \subseteq \mathbf{I}\}$, and we will prove:
$f(Q(\mathbf{I})) = Q(\mathbf{J})$, where $Q(\mathbf{J}) = \{\mu(T(v)) \mid \mu \text{ valuation with } \mu(\mathbf{S}) \subseteq \mathbf{J}\}$.

- Inclusion '$\subseteq$'. For $f(\nu(T(v))) \in f(Q(\mathbf{I}))$, we define μ which equals $f \circ \nu$ on $\mathrm{Var}(\mathbf{S}) \cup \mathrm{Dom}(Q)$, which equals some constant on the others variables and is identity on others constants. Then $f(\nu(T(v))) = \mu(T(v))$ and μ is a valuation (here we use the hypothesis: f is identity on $\mathrm{Dom}(Q)$. Moreover $\nu(\mathbf{S}) \subseteq \mathbf{I} \Longrightarrow \mu(\mathbf{S}) = f(\nu(\mathbf{S})) \subseteq f(\mathbf{I}) = \mathbf{J}$.

- Inclusion '$\supseteq$'. Let $\mu(T(v))$ be an element $Q(\mathbf{J})$. We define ν which equals $f^{-1} \circ \mu$ on $\mathrm{Var}(\mathbf{S})$, and which equals some constant on the others variables and is identity on $\mathbf{Dom}$. First we verify $\mu(T(v)) = f(\nu(T(v)))$. Let x an element of $\mathrm{Sym}(T(v))$. If $x \in \mathrm{Var}(\mathbf{S})$, since $\mu(\mathbf{S}) \subseteq \mathbf{J}$ we have $\mu(x) \in \mathrm{Dom}(\mathbf{J})$, then $f(f^{-1}(\mu(x))) = \mu(x)$. If x is a constant, then $x \in \mathrm{Dom}(Q)$, hence $\mu(x) = x = \nu(x)$, but also $f(x) = x$, then $\mu(x) = f(\nu(x))$.
Now ν is a valuation, and we have $\mu(\mathbf{S}) \subseteq \mathbf{J} \Longrightarrow \nu(\mathbf{S}) = f^{-1}(\mu(\mathbf{S})) \subseteq f^{-1}(\mathbf{J}) = \mathbf{I}$.

It remains to see that f maps $\mathrm{Dom}(Q(\mathbf{I}))$ to $\mathrm{Dom}(Q(\mathbf{J}))$, but this is clear since we have $f(Q(\mathbf{I})) = Q(\mathbf{J})$ and f is identity in $\mathrm{Dom}(Q)$.

2. Let $\sigma : \mathbf{B} \to K$ be a constraint with f identity on $\mathrm{Dom}(\sigma)$.

First case. σ is a fegd: $\mathbf{B} \to e_i = f_j$, where $e_i, f_j \in \mathrm{Var}(\mathbf{B}) \cup \mathbf{Dom}$.
Assume $\mathbf{I} \models \sigma$. We will prove $\mathbf{J} \models \sigma$. Let μ be a valuation such that $\mu(\mathbf{B}) \subseteq \mathbf{J}$. We want to verify $\mu(e_i) = \mu(f_j)$.
As in *1*, consider valuation ν which equals $f^{-1} \circ \mu$ on $\mathrm{Var}(\mathbf{B})$, and which equals some constant on the others variables and is identity on $\mathbf{Dom}$. Then $\nu(\mathbf{B}) = f^{-1}(\mu(\mathbf{B})) \subseteq f^{-1}(\mathbf{J}) = \mathbf{I}$, hence $\nu(e_i) = \nu(f_j)$ since $\mathbf{I} \models \sigma$. So we have $f^{-1}(\mu(e_i)) = f^{-1}(\mu(f_j))$ and by applying f to this equality we get $\mu(e_i) = \mu(f_j)$.

Assume now $\mathbf{J} \models \sigma$ and we want to prove $\mathbf{I} \models \sigma$. Let ν be a valuation such that $\nu(\mathbf{B}) \subseteq \mathbf{I}$. As in *1*, define valuation μ which equals $f \circ \nu$ on $\mathrm{Var}(\mathbf{B}) \cup \mathrm{Dom}(\sigma)$, equals some constant on the others variables and is identity on others constants. (Here $\mathrm{Dom}(\sigma) = \mathrm{Dom}(\mathbf{B})$).
Then $\mu(\mathbf{B}) = f(\nu(\mathbf{B})) \subseteq f(\mathbf{I}) = \mathbf{J}$, and then $\mu(e_i) = \mu(f_j)$, that is $f(\nu(e_i)) = f(\nu(f_j))$. We apply then f^{-1} to this last equality and get $\nu(e_i) = \nu(f_j)$.

[5] Or equality if one (then both) equals $\square$.

Second case. σ is a ftgd.

Assume $\mathbf{I} \models \sigma$. We have already seen in remark 4.3.4 this is equivalent to $\sigma(\mathbf{I}) \subseteq \mathbf{I}$. We want to verify if also $\sigma(\mathbf{J}) \subseteq \mathbf{J}$. But using *1*, we get $\sigma(\mathbf{J}) = f(\sigma(\mathbf{I})) \subseteq f(\mathbf{I}) = \mathbf{J}$.
Conversely assume $\mathbf{J} \models \sigma$, that is $\sigma(\mathbf{J}) \subseteq \mathbf{J}$. Then $\sigma(\mathbf{I}) = f^{-1}(\sigma(\mathbf{J})) \subseteq f^{-1}(\mathbf{J}) = \mathbf{I}$.

3. It suffices to verify that applying one step of chasing to both ground databases $\mathbf{I}_k$ and $\mathbf{J}_k$ such that $\mathbf{I}_k \simeq \mathbf{J}_k$, produce two new ground databases $\mathbf{I}_{k+1}$ and $\mathbf{J}_{k+1}$ such that $\mathbf{I}_{k+1} \simeq \mathbf{J}_{k+1}$.
Let σ be the constraint of this chasing step.
If σ is a fegd, we have simultaneously $[\mathbf{I}_{k+1} = \mathbf{I}_k$ or $\mathbf{I}_{k+1} = \square]$ and $[\mathbf{J}_{k+1} = \mathbf{J}_k$ or $\mathbf{J}_{k+1} = \square]$, since with *2*: $\mathbf{I}_k \models \sigma \Leftrightarrow \mathbf{J}_k \models \sigma$.
If σ is a ftgd, we have $\mathbf{I}_{k+1} = \mathbf{I}_k \cup \sigma(\mathbf{I}_k)$ and $\mathbf{J}_{k+1} = \mathbf{J}_k \cup \sigma(\mathbf{J}_k)$. By hypothesis $\mathbf{I}_k \simeq \mathbf{J}_k$ (with function f), then using *1* with rule σ, we have also $\sigma(\mathbf{I}_k) \simeq \sigma(\mathbf{J}_k)$ (with same function f). Thus we deduce using this injective function f that $\mathbf{I}_{k+1} \simeq \mathbf{J}_{k+1}$.

Examples 4.12

1. We see here that the lemma is not true if we do not assume f identity on the constants occuring in the body of Q.
We set $\mathbf{I} = \{R(b, c)\}$, $\mathbf{J} = \{R(a, c)\}$ and $Q : R(a, x) \rightarrow R(x, x)$.
Then $\mathbf{I} \simeq \mathbf{J}$ with $f : \{b \rightarrow a,\ c \rightarrow c,\ a \rightarrow a' \neq a\}$, but $Q(\mathbf{I}) = \emptyset$ and $Q(\mathbf{J}) = \{R(c, c)\}$. Hence $Q(\mathbf{I}) \not\simeq Q(\mathbf{J})$.

2. We see here that the lemma is not true if we do not assume f identity on the constants occuring in the head of Q.
We set $\mathbf{I} = \{R(a, b)\}$, $\mathbf{J} = \{R(e, f)\}$ and $Q : R(x, y) \rightarrow R(a, x)$.
Then $\mathbf{I} \simeq \mathbf{J}$ with $f : \{a \rightarrow e,\ b \rightarrow f,\ e \rightarrow e' \neq e,\ f \rightarrow f' \neq f\}$, but $Q(\mathbf{I}) = \{R(a, a)\}$ and $Q(\mathbf{J}) = \{R(a, e)\}$. Hence $Q(\mathbf{I}) \not\simeq Q(\mathbf{J})$.

5 Algorithm

Recall the general problem from introduction that we specialize here :

- one rule $Q : \mathbf{S} \rightarrow \mathbf{T}$, which represents our source-to-target rules, and where we denote $\text{Schem}(\mathbf{S}) = \{S_1, S_2, \ldots, S_n\}$ as the source schema, and $\text{Schem}(\mathbf{T}) = \{T\}$ as the target schema,
- on source and target schemas we consider respectively Σ_s and Σ_t, sets of constraints, and here we assume for Σ_t that all of its bodies'schemas are $\{T\}$.

We want to decide the following question:

For all $\mathbf{I}$ ground databases such that $\mathbf{I} \models \Sigma_s$, do we have $Q(\mathbf{I}) \models \Sigma_t$?

We provide now an algorithm for deciding this question.
The algorithm consists in repeating the following procedure for all constraints in Σ_t successively. If for one the procedure says `'no'`, then stop: the answer of our algorithm is `'no'`. Else (that is always obtain `'yes'` in procedures) then our algorithm says `'yes'`.

Notice that procedure exits when it asserts 'yes' or 'no'.

We fix $\sigma : \mathbf{D} \to K$ a dependency in Σ_t.
We define the set $\mathcal{C} = \mathrm{Dom}(Q) \cup \mathrm{Dom}(\sigma) \cup_{\sigma_i \in \Sigma_s} \mathrm{Dom}(\sigma_i)$, that is the set of all constants occuring from the fixed context.

Note : When Q-source for $\mathbf{D}$ exists:
$\mathrm{Dom}(Q^{-1}(\mathbf{D})) = \mathrm{Dom}(\cup_{L \in \mathbf{D}} \theta(\rho_L(\mathbf{S}))) \subseteq \mathrm{Dom}(\mathbf{S}) \subseteq \mathrm{Dom}(Q) \subseteq \mathcal{C}$.

Procedure (for the dependency σ).

1. Calculate the Q-source $Q^{-1}(\mathbf{D})$. If the mgu needed for this does not exist, then say `'yes'` and exit.
2. Fix a set $\mathcal{D}$ of constants one-to-one with $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$ and disjoint from $\mathcal{C}$. Let $a \in \mathbf{Dom}$ be some arbitrary constant. Consider the (finite) set $\mathcal{V}$ of all valuations which equal a on variables outside $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$, and take their values on $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$ in the following set: $\mathcal{D} \cup \mathcal{C}$.
   For all $\nu \in \mathcal{V}$, calculate $\mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big)$. If all these Chases result in $\square$, then say `'yes'` and exit.
3. For each preceding valuation ν such that $\mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big) \neq \square$, verify if $Q\big(\mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big)\big) \models \sigma$. If not (for one), then say `'no'` and exit, else say `'yes'` and return.

Theorem 5.1 *Previous algorithm gives the correct answer.*

Proof. We will prove the following (with dependency σ of the preceding procedure):
- if procedure says `'yes'`, then $\forall$ **I** ground database satisfying Σ_s, we have $Q(\mathbf{I}) \models \sigma$ **(6.1.1)**
- if procedure says `'no'`, then $\exists$ **I** ground database satisfying Σ_s, such that $Q(\mathbf{I}) \not\models \sigma$ **(6.1.2)**

Now let $\Sigma_t = \{\sigma_1, \ldots, \sigma_s\}$. Assume **(6.1.1)** and **(6.1.2)** are true with σ_j's.

In case where procedure says `'no'` for some σ_j, $j \in \{1, \ldots, s\}$, then $\exists$ **I** ground database satisfying Σ_s, such that $Q(\mathbf{I}) \not\models \sigma_j$, so by definition of satisfaction: such that $Q(\mathbf{I}) \not\models \Sigma_t$.

Then in that case, our algorithm says `'no'`, and the answer to the question is actually `'no'`.

In case where procedure says `'yes'` for all σ_j, $j = 1, \ldots, s$, we have:
$\forall\, j \in \{1, \ldots, s\}$, $\forall$ **I** ground database, $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \sigma_j$,
that is by definition of satisfaction:
$\forall$ **I** ground database, $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \Sigma_t$.
So in that case, our algorithm says `'yes'`, and the answer to the question is actually `'yes'`.

Then, to prove the theorem, it suffices to prove **(6.1.1)** and **(6.1.2)**.

Proof of **(6.1.2)**.

The only case where procedure says `'no'` is when there exists valuation $\nu \in \mathcal{V}$ such that $\mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big) \neq \square$ and such that $Q\big(\mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big)\big) \not\models \sigma$.
Therefore, considering ground database $\mathbf{I} = \mathrm{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big)$, we have $\mathbf{I} \models \Sigma_s$ (use lemma 4.9.*3*), but $Q(\mathbf{I}) \not\models \sigma$.

Proof of **(6.1.1)**.

Recall $Q : \mathbf{S} \to T(v)$ the rule we consider.
We assume procedure says 'yes' and we want to prove the assertion:

$$\forall\ \mathbf{I} \text{ ground database, } \mathbf{I} \models \Sigma_s \Longrightarrow Q(\mathbf{I}) \models \sigma.$$

Notice first that if there exists no ground database $\mathbf{I}$ such that $\mathbf{I} \models \Sigma_s$, then this assertion is obviously true. We call this eventuality Σ_s *unsatisfiable*. As we will see in the following case 2, this corresponds to the eventuality :

"for all $\nu \in \mathcal{V}$, we have $\text{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big) = \square$."

In the following we consider $\mathbf{I}$ a ground database satisfying Σ_s (and then assume it exists).

Case 1. Q-source $Q^{-1}(\mathbf{D})$ doesn't exist (the first `'yes'` of the procedure).
First observe that we can reformulate the first part of proposition 3.6 in the following manner:

> Given a rule $Q : \mathbf{S} \to T(v)$, and a database $\mathbf{D}$ with schema $\{T\}$, if we have a database $\mathbf{A}$ such that $Q(\mathbf{A}) \succcurlyeq \mathbf{D}$, then a Q-source for $\mathbf{D}$ exists.

Hence in this case 1, we deduce from this : there is no database $\mathbf{A}$ such that $Q(\mathbf{A}) \succcurlyeq \mathbf{D}$.
Then for our ground database $\mathbf{I}$, there is no valuation μ such that $\mu(\mathbf{D}) \subseteq Q(\mathbf{I})$, and then from remark 4.3.3, we deduce $Q(\mathbf{I}) \models \sigma$.

Case 2. $Q^{-1}(\mathbf{D})$ exists (all remaining `'yes'` in the procedure).

We use $\mathcal{D}$ a set of constants one-to-one with $\text{Var}\big(Q^{-1}(\mathbf{D})\big)$, disjoint from $\mathcal{C}$, and an arbitrary constant $a \in \mathbf{Dom}$. We consider the (finite) set $\mathcal{V}$ of all valuations which equal a on variables outside $\text{Var}\big(Q^{-1}(\mathbf{D})\big)$, and take their values in $\mathcal{D} \cup \mathcal{C}$ on $\text{Var}\big(Q^{-1}(\mathbf{D})\big)$.

We have assumed $\mathbf{I} \models \Sigma_s$, and we want to prove $\mathbf{J} = Q(\mathbf{I}) \models \sigma$.
For this we take μ a valuation such that $\mu(\mathbf{D}) \subseteq \mathbf{J}$.
Then we are in the situation of proposition 3.6, with $\mathbf{I}$ ground database.
Therefore we have seen we can choose a valuation α such that: $\begin{cases} \alpha(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}, \\ \mu(\mathbf{D}) \subseteq Q(\alpha(Q^{-1}(\mathbf{D}))). \end{cases}$

Now, since $I \models \Sigma_s$, using lemma 4.9.*4*, we get $\text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big) \subseteq \mathbf{I}$ (it is understood this Chase is not $\square$), and then using 4.9.*1* and monotonicity of Q we have:

$$\mu(\mathbf{D}) \subseteq Q(\alpha(Q^{-1}(\mathbf{D}))) \subseteq Q\big(\text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big)\big) \subseteq Q(\mathbf{I}) = \mathbf{J}.$$

We set $\mathbf{J}_\mu = Q\big(\text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big)\big)$.

Consider the set $\alpha\big(Q^{-1}(\mathbf{D})\big)$. Since is $\mathbf{I}$ is arbitrary, this set can contain arbitrary constants, like constants from $\text{Dom}(Q)$, $\text{Dom}(\sigma_i)$ where $\sigma_i \in \Sigma_s$, or even from $\text{Dom}(\sigma)$.
But since our definitions of $\mathcal{D}$, $\mathcal{C}$ and $\mathcal{V}$, it exists a valuation $\nu_\alpha \in \mathcal{V}$ and a constants'renaming f such that :

- $f\big(\nu_\alpha\big(Q^{-1}(\mathbf{D})\big)\big) = \alpha\big(Q^{-1}(\mathbf{D})\big)$
- f is identity on $\mathcal{C}$,
- $\nu_\alpha\big(\text{Sym}\big(Q^{-1}(\mathbf{D})\big)$ exactly map (in one-to-one way by f) to $\alpha\big(\text{Sym}\big(Q^{-1}(\mathbf{D})\big)\big)$,
- f is injective on $\mathbf{Dom}$.

Then by lemma 4.11.*3* we have $\text{Chase}_{\Sigma_s}\big(\nu_\alpha(Q^{-1}(\mathbf{D}))\big) \simeq \text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big)$, and since the second is not $\square$, also the first is not.
Then we see here that condition "for all $\nu \in \mathcal{V}$, $\text{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big) = \square$" implies it doesn't exist $\mathbf{I}$ ground database which satisfies Σ_s, case already discussed.

Now, again using lemma 4.11, our hypothesis $Q\big(\text{Chase}_{\Sigma_s}\big(\nu_\alpha(Q^{-1}(\mathbf{D}))\big)\big) \models \sigma$ implies that we also have $\mathbf{J}_\mu = Q\big(\text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big)\big) \models \sigma$.

Conclusion: lemma 4.4 with $\mathbf{J}_\mu$ gives us that $\mathbf{J} \models \sigma$.

Examples 5.2 Notice that the following may be obvious but I consider it advisable.

1. Copy case. We consider the rule $Q : S(x_1, \ldots, x_n) \to T(x_1, \ldots, x_n)$

On the source we have a set of dependencies $\Sigma_s = \{\sigma_1, \ldots, \sigma_p\}$ (we assume their bodies are on schema $\{S\}$), and on the target a set of dependencies $\Sigma_t = \{\tau_1, \ldots, \tau_r\}$, (where we also assume their bodies are on schema $\{T\}$).
We assume that none of the dependencies in Σ_s (resp. Σ_t) is a logical consequence of other dependencies in Σ_s, (resp. Σ_t). Then:

"$\forall$ $\mathbf{I}$ ground database, $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \Sigma_t$" is true $\iff$ $\begin{cases} \text{- either } \Sigma_s \text{ is unsatisfiable,} \\ \text{- either } p \geqslant r \text{ and } \sigma_i = \tau_i \text{ where} \\ \text{we replace letter } T \text{ by letter } S. \end{cases}$

In fact the meaning of the right hand of this equivalence (when Σ_s is satisfiable) is: "the dependencies in Σ_t must appear in Σ_s modulo the replacement of letter T by letter S.

Sketch proof (easy):
$\Leftarrow$) Assume Σ_s is satisfiable and let $\mathbf{I}$ be ground database such that $\mathbf{I} \models \Sigma_s$. Then $Q(\mathbf{I})$ is a ground database where we replace letter S by letter T. Then $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \Sigma_t$.
$\Rightarrow$) Assume Σ_s is satisfiable. Let τ_i a dependency of Σ_t which does not appear in Σ_s. It suffices to be convinced that we can find a ground database $\mathbf{I}$ such that $\mathbf{I} \models \Sigma_s$ but $\mathbf{I} \not\models Q^{-1}(\tau_i)$, where $Q^{-1}(\tau_i)$ is a shorthand for the dependency τ_i where we have replaced letter T by letter S. Then obviously $Q(\mathbf{I}) \not\models \tau_i$, hence $Q(\mathbf{I}) \not\models \Sigma_t$, and this contradicts the hypothesis.

And now, what about our algorithm, when we are in case of previous equivalence ?
For simplicity of exposure, assume $\Sigma_s = \{\sigma\}$ and $\Sigma_t = \{\tau\}$ where:
$\sigma : \mathbf{D}_s \to K_s$, with $\mathbf{D}_s = \{S(\vec{x_1}), \ldots, S(\vec{x_m})\}$ and K_s replaced by either $e_i = f_j$, either $S(\vec{y})$,
$\tau : \mathbf{D}_t \to K_t$, with $\mathbf{D}_t = \{T(\vec{x_1}), \ldots, T(\vec{x_m})\}$ and K_t replaced by either $e_i = f_j$, either $T(\vec{y})$.
We have $Q^{-1}(\mathbf{D}_t) = \mathbf{D}_s$, and on the other hand Σ_s and Σ_t are simultaneously satisfiable or unsatisfiable. Assume they are satisfiable. We have with obvious notations: $\forall\ \nu_\alpha \in \mathcal{V}$,
$Q(\text{Chase}_{\Sigma_s}(\nu_\alpha(Q^{-1}(\mathbf{D}_t)))) = Q(\text{Chase}_{\sigma}(\nu_\alpha(\mathbf{D}_s))) = \text{Chase}_{\tau}(\nu_\alpha(Q(\mathbf{D}_s))) = \text{Chase}_{\tau}(\nu_\alpha(\mathbf{D}_t))$
and this last one satisfy τ (4.9.3). Then our algorithm actually says `'yes'`.

2. Let $Q : S(x, y) \to T(x, y)$ be the rule, and $\sigma : \mathbf{D} = \{\} \to T(a, b)$ be the dependency on the target (we may consider that the schema of the empty database is $\{T\}$).
Then $Q^{-1}(\mathbf{D}) = \emptyset$, and $\mathcal{V} = \{\nu\}$, where ν is the valuation which equals arbitrary constant a on **Var**. We have also $\mu(\mathbf{D}) = \emptyset$ and $\nu(Q^{-1}(\mathbf{D})) = \alpha(Q^{-1}(\mathbf{D})) = \emptyset$.
Then $\text{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big) = \text{Chase}_{\Sigma_s}\big(\alpha(Q^{-1}(\mathbf{D}))\big) = \text{Chase}_{\Sigma_s}(\emptyset)$, which depends on Σ_s.

We have already seen in 1, for such copy-case that the answer to the question is `'yes'` if and only if either $\Sigma_s = \{\sigma_s\}$, where $\sigma_s : \{\} \to S(a,b)$, or either if Σ_s is unsatisfiable.

Verify correctness of our algorithm here.

- If $\Sigma_s = \emptyset$, the answer is `'no'`. On the other hand (our algorithm): $\text{Chase}_{\Sigma_s}(\emptyset) = \emptyset$, then $Q(\text{Chase}_{\Sigma_s}(\emptyset)) = \emptyset$, which doesn't satisfy σ (4.3.3). Then our algorithm says `'no'`.

- If $\Sigma_s = \{\sigma_s\}$, the answer is `'yes'`. On the other hand (our algorithm): $\text{Chase}_{\Sigma_s}(\emptyset) = \{S(a,b)\}$, $Q(\{S(a,b)\}) = \{T(a,b)\}$ which satisfies σ. Then our algorithm says `'yes'`.

- If $\Sigma_s = \{\sigma'_s\}$ where $\sigma'_s : \{\} \to S(e,f)$. Then the answer is `'no'`, and we have $\text{Chase}_{\Sigma_s}(\emptyset) = \{S(e,f)\}$, and $Q(\{S(e,f)\}) = \{T(e,f)\}$ which doesn't satisfy σ, then our algorithm says `'no'`.

- If $\Sigma_s = \{\sigma_s, \sigma_1\}$, where $\sigma_1 : S(x,y) \to x = y$. The answer is `'yes'` ($\Sigma_s$ is unsatisfiable), and we have $\text{Chase}_{\Sigma_s}(\emptyset) = \square$, so that our algorithm says `'yes'`.

3. We give here example of not copy-case. Let $Q : S(x,y) \to T(x,x)$ be the rule, and consider only one dependency respectively on source and target: $\Sigma_s = \{\sigma_s\}$ and $\Sigma_t = \{\sigma_t\}$.

We set $\sigma_s : S(x,y) \to x = a$, and $\sigma_t : \mathbf{D} = \{T(x,y)\} \to K$, with $K = \begin{cases} \text{either } x = y \\ \text{either } T(x,x) \\ \text{either } T(a,a). \end{cases}$

It is clear that we have: $\forall$ **I** ground database, $\mathbf{I} \models \Sigma_s \Longrightarrow Q(\mathbf{I}) \models \Sigma_t$.

Verify that our algorithm says `'yes'`.

First we calculate Q-source for $\mathbf{D}$. We search mgu θ such that $\theta(T(x,y)) = \theta(T(x_1,x_1))$, where we use renaming $\rho_1(Q) : S(x_1,y_1) \to T(x_1,x_1)$.
We have $x \equiv x_1 \equiv y$, and we may choose x for a representative of this class.
Then $Q^{-1}(\mathbf{D}) = \theta(\rho_1(\{S(x,y)\})) = \theta(\{S(x_1,y_1)\}) = \{S(x,y_1)\}$.
Now we search for valuations $\nu \in \mathcal{V}$.
We have $\text{Dom}(Q) = \emptyset$, $\text{Dom}(\sigma_s) = \{a\}$ and $\text{Dom}(\sigma_t) = \{a\}$ or $\emptyset$.
Then $\mathcal{C} = \text{Dom}(Q) \cup \text{Dom}(\sigma_s) \cup \text{Dom}(\sigma_t) = \{a\}$.
We need a set of constants $\mathcal{D}$ disjoint from $\mathcal{C}$ and one to one with $\text{Var}(Q^{-1}(\mathbf{D})) = \{x, y_1\}$.
Set $\mathcal{D} = \{e, f\}$. Then the (finite) set of valuations ν is defined by:
$\forall\ z \in \mathbf{Var} \backslash \text{Var}(Q^{-1}(\mathbf{D}))$, $\nu(z) = a$,
$\nu(x), \nu(y_1) \in \mathcal{D} \cup \mathcal{C} = \{e, f\} \cup \{a\}$.
We have then here nine possibilities, we give all possible $\nu(Q^{-1}(\mathbf{D}))$:
$\{S(e,f)\}$, $\{S(e,a)\}$, $\{S(f,e)\}$, $\{S(a,e)\}$, $\{S(f,a)\}$, $\{S(a,f)\}$, $\{S(a,a)\}$, $\{S(e,e)\}$, $\{S(f,f)\}$.

Now we have to calculate the $\text{Chase}_{\Sigma_s}\big(\nu(Q^{-1}(\mathbf{D}))\big)$ for all $\nu \in \mathcal{V}$.
For those $\nu(Q^{-1}(\mathbf{D}))$ where the first component of the atom is not a, then the Chase is $\square$, and for the other ones the resulting Chase are themselves. Then, those who are not $\square$ are $\{S(a,a)\}$, $\{S(a,e)\}$, $\{S(a,f)\}$.
Therefore the output by Q of these last three ones is $\{T(a,a)\}$, which satisfy Σ_t, and our algorithm says `'yes'`.

Examples 5.3 Here we give examples of ideas which would have greatly simplify the algorithm, but which unfortunately do not work, or stay for the moment unproved.

Let us define a *skolem* valuation: this is a valuation which is injective on **Var**, and such that the values it assigns for the variables in Q, or Σ_s, or Σ_t, are not values of $\mathcal{C}$.

First idea. Replace all the valuations $\nu \in \mathcal{V}$ by only one *skolem* valuation on $Q^{-1}(\mathbf{D})$.

This doesn't work as we will see in the following example:

Let $Q : S(x,y) \to T(x,y)$ be the rule, $\Sigma_s : S(x,y) \to x = y$ and $\sigma : \mathbf{D} = \{T(x,y)\} \to T(a,a)$. As already seen, in that case: "$\forall$ $\mathbf{I}$ ground database, $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \sigma$" is false. And then our algorithm says `'no'`. But now consider algorithm where we replace "$\forall\ \nu \in \mathcal{V}$...", by "for a fixed *skolem* ν...".
We have $Q^{-1}(\mathbf{D}) = \{S(x,y)\}$. Fix the *skolem* $\nu : \{x \to h, y \to f\}$.
Then $\text{Chase}_{\Sigma_s}(\nu(Q^{-1}(\mathbf{D}))) = \text{Chase}_{\Sigma_s}(\{S(h,f)\}) = \square$. Hence this newer algorithm would say `'yes'`, and then it would be false.
Notice that our correct algorithm consider all valuations on $\{x,y\}$ taking values in (for instance) $\{h,f\} = \mathcal{D}$, and then for $\mathbf{I} \models \Sigma_s$, and α such that $\alpha(Q^{-1}(\mathbf{D})) \subset \mathbf{I}$, we would find $\nu : \{x \to h, y \to h\}$ such that $f(\nu(Q^{-1}(\mathbf{D}))) \sim \alpha(Q^{-1}(\mathbf{D}))$.
For this ν we have $\text{Chase}_{\Sigma_s}(\nu(Q^{-1}(\mathbf{D}))) = \{S(h,h)\} \models \Sigma_s$, and $Q(\text{Chase}_{\Sigma_s}(\nu(Q^{-1}(\mathbf{D})))) = \{T(h,h)\} \not\models \sigma$.

Second idea. Here we try to stay "most general" as far as possible, that is: we *specialize* with *skolem* only on last step, that is on $Q(\text{Chase}_{\Sigma_s}(Q^{-1}(\mathbf{D})))$. (We use definition 8 of [2] for the Chase on non-ground databases, here $Q^{-1}(\mathbf{D})$). For the moment, the correctness of this other algorithm remains unproved.

Let $Q : S(x,y) \to T(x,y)$ be the rule, $\Sigma_s = \{\sigma_1, \sigma_2\}$, where:
$\begin{cases} \sigma_1 : S(x,a) \to S(x,b) \\ \sigma_2 : S(x,a) \to S(c,x) \end{cases}$ and $\sigma : \mathbf{D} = \{T(x,a)\} \to T(x,b)$.
As already seen (5.2.1), in that case: "$\forall$ $\mathbf{I}$ ground database, $\mathbf{I} \models \Sigma_s \Rightarrow Q(\mathbf{I}) \models \sigma$" is true.

On the other hand we have:
- $Q^{-1}(\mathbf{D}) = \{S(x,a)\}$,
- $\text{Chase}_{\Sigma_s}(Q^{-1}(\mathbf{D})) = \{S(x,a), S(x,b), S(c,x)\}$,
- $Q(\text{Chase}_{\Sigma_s}(Q^{-1}(\mathbf{D}))) = \{T(x,a), T(x,b), T(c,x)\}$.

Now choose a valuation *skolem* on $Q(\text{Chase}_{\Sigma_s}(Q^{-1}(\mathbf{D})))$, that is a valuation which replaces variable x by some constant not in $\{a,b,c\}$. For instance take $\nu : \{x \to e\}$.
Then $\nu(Q(\text{Chase}_{\Sigma_s}(Q^{-1}(\mathbf{D})))) = \{T(e,a), T(e,b), T(c,e)\}$ which satisfy σ, so our newer algorithm would say correct answer `'yes'`, and greatly faster in comparison to our algorithm.

Comparison with our algorithm.
Let $\mathcal{D} = \{e\}$ one to one with $\text{Var}(Q^{-1}(\mathbf{D})) = \{x\}$ and disjoint from $\mathcal{C}$.
We consider valuations ν which take their values on $\text{Var}(Q^{-1}(\mathbf{D}))$ in $\mathcal{D} \cup \mathcal{C} = \{a,b,c,e\}$.
Let $\mathbf{I} = \text{Chase}_{\Sigma_s}(\nu(Q^{-1}(\mathbf{D})))$.

First one $\nu : \{x \to a\}$. Then:
$\nu(Q^{-1}(\mathbf{D})) = \{S(a,a)\}$,
$\mathbf{I} = \{S(a,a), S(a,b), S(c,a), S(c,b), S(c,c)\}$,
$Q(\mathbf{I}) = \{T(a,a), T(a,b), T(c,a), T(c,b), T(c,c)\} \models \sigma$.

Second one $\nu : \{x \to u\}$, where u is either b, c, or e (identical cases). Then:
$\nu(Q^{-1}(\mathbf{D})) = \{S(u, a)\}$,
$\mathbf{I} = \{S(u, a), S(u, b), S(c, u)\}$,
$Q(\mathbf{I}) = \{T(u, a), T(u, b), T(c, u)\} \models \sigma$.

Hence our algorithm says also `'yes'`.

ANNEX

6 Auxiliary results

Corollary 6.1 *(of proposition 3.6).*
Let $Q : \mathbf{S} \longrightarrow \mathbf{T}$ be a rule. Let $\mathbf{D}$ be a database whose schema is $\{T\}$.
We assume a Q-source for $\mathbf{D}$ exists.

1. $Q(Q^{-1}(\boldsymbol{D})) \succcurlyeq \mathbf{D}$.
2. *If $\mathbf{I}$ is a database such that $Q(\mathbf{I}) \succcurlyeq \mathbf{D}$, then $\mathbf{I} \succcurlyeq Q^{-1}(\mathbf{D})$.*
3. *For every database $\mathbf{I}$, if $Q(Q^{-1}(\mathbf{D})) \succcurlyeq Q(\mathbf{I}) \succcurlyeq \mathbf{D}$, then $Q(Q^{-1}(\mathbf{D})) \sim Q(\mathbf{I})$.*
   ("Minimality" of $Q(Q^{-1}(\mathbf{D}))$).

Proof.
1. By definition $Q(Q^{-1}(\mathbf{D})) = \{\mu(T(v)) \mid \mu$ substitution on $\mathrm{Var}(\mathbf{S})$ and $\mu(\mathbf{S}) \subseteq Q^{-1}(\mathbf{D})\}$, and by 3.5.3: $Q^{-1}(\mathbf{D}) = \cup\{\theta(\rho_L(\mathbf{S})) \mid L \in \mathbf{D}\}$ (θ a mgu, cf. definition 3.4).
We will show that $\theta(\boldsymbol{D}) \subseteq Q(Q^{-1}(\mathbf{D}))$ (which proves $Q(Q^{-1}(\mathbf{D})) \succcurlyeq \mathbf{D}$).
Let $\theta(L) \in \theta(\mathbf{D})$, with $L \in \mathbf{D}$. From definition of θ, we have $\theta(L) = \theta(\rho_L(T(v)))$.
Define substitution μ_L which equals $\theta \circ \rho_L$ on $\mathrm{Var}(\mathbf{S})$, and is identity elsewhere.
Since $\mathrm{Var}(T(v)) \subseteq \mathrm{Var}(\mathbf{S})$, we have $\theta(L) = \mu_L(T(v))$, but, using lemma 1.2, we also have $\mu_L(\boldsymbol{S}) = \theta(\rho_L(\mathbf{S})) \subseteq Q^{-1}(\mathbf{D})$. Hence $\theta(L) \in Q(Q^{-1}(\mathbf{D}))$, which proves our statement.

2. Let $\mathbf{I}$ a database such that $Q(\mathbf{I}) \succcurlyeq \mathbf{D}$. Set $\mathbf{J} = Q(\mathbf{I})$. We have then μ substitution on $\mathrm{Var}(\mathbf{D})$ such that $\mu(\mathbf{D}) \subseteq \mathbf{J} = Q(\mathbf{I})$. Then we apply proposition 3.6 which give us substitution α such that $\alpha(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}$. Of course if we consider substitution α' which equals α on $\mathrm{Var}(Q^{-1}(\mathbf{D}))$ and is identity elsewhere, then from 1.2 we also have $\alpha'(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}$. This proves $\mathbf{I} \succcurlyeq Q^{-1}(\mathbf{D})$.

3. Assume $Q(Q^{-1}(\mathbf{D})) \succcurlyeq Q(\mathbf{I}) \succcurlyeq \mathbf{D}$. The second comparison implies by *2*: $\mathbf{I} \succcurlyeq Q^{-1}(\mathbf{D})$, so by monotonicity of Q we get $Q(\mathbf{I}) \succcurlyeq Q(Q^{-1}(\mathbf{D})$, and then we conclude $Q(Q^{-1}(\mathbf{D}) \sim Q(\mathbf{I})$.

Remark-examples 6.2 Let $Q : \mathbf{S} \longrightarrow \mathbf{T}$ be a rule and $\mathbf{D}$ a database whose schema is $\{T\}$.

1. We give an obvious example where Q-source doesn't exist.
2. Q-source for $\mathbf{D}$ exists doesn't imply: $\forall\ \nu$ valuation, Q-source exists for $\nu(\mathbf{D})$.
3. Q-source for $\mathbf{D}$ exist does not imply: $\forall\ \mathbf{I}$ database, $Q(\mathbf{I}) \succcurlyeq \mathbf{D}$.
4. Q-source for $\mathbf{D}$ exists $\Longleftrightarrow \exists\ \mathbf{I}$ database such that $Q(\mathbf{I}) \succcurlyeq \mathbf{D}$.
5. Q-source for $\mathbf{D}$ exists with mgu θ implies Q-source for $\theta(\mathbf{D})$ exists.

Proof.
1. Set $\mathbf{D} = \{T(a, x)\}$ and $Q : S(x, y) \rightarrow T(b, y)$. Then we search for a unifier μ such that $\mu(T(a, x) = \mu(T(b, x_1))$. (Recall $Q_{\rho_1} : S(x_1, y_1) \rightarrow T(b, y_1)$). Then μ doesn't exist (and $\boldsymbol{\mathcal{U}} = \emptyset$ see definition 3.4).
2. - 3. Set $Q : S(y) \rightarrow T(a, y, y)$, and $\mathbf{D} = \{T(x, x', a)\}$.
Then Q-source exists for $\mathbf{D}$, since it exists unifier μ such that $\mu(T(a, y, y)) = \mu(T(x, x', a))$ (we have necessarily $\mu(x) = \mu(x') = \mu(y) = a$) and this μ is the only possible unifier, and then is a mgu. Also $Q^{-1}(\mathbf{D}) = \mu(\{S(y)\}) = \{S(a)\}$.
But if we take ν valuation such that $\nu(x) = e$, $\nu(x') = f$, then $\nu(\mathbf{D}) = \{T(e, f, a)\}$, and we do not have unifier μ' such that $\mu'(T(a, y, y)) = \mu'(T(e, f, a))$ and then there is no mgu between $\nu(\mathbf{D})$ and $T(a, y, y)$, hence Q-source doesn't exist for $\nu(\mathbf{D})$.

Now if we take $\mathbf{I} = \{S(b)\}$, then $Q(\mathbf{I}) = \{T(a,b,b)\}$, and we cannot find substitution ν such that $\nu(\mathbf{D}) \subseteq Q(\mathbf{I})$. Hence for this (ground) database $\mathbf{I}$ we do not have $Q(\mathbf{I}) \succcurlyeq \mathbf{D}$.
4. $\Leftarrow$) This is first part of proposition 3.6.
$\Rightarrow$) Use $\mathbf{I} = Q^{-1}(\mathbf{D})$ and previous corollary, point 1.
5. It suffices to show there exists a database $\mathbf{I}$ such that $Q(\mathbf{I}) \succcurlyeq \theta(\mathbf{D})$, since previous point. But in previous corollary we have seen $\theta(\mathbf{D}) \subseteq Q(Q^{-1}(\mathbf{D}))$, hence obviously $\theta(\mathbf{D}) \preccurlyeq Q(\mathbf{I})$ with $\mathbf{I} = Q^{-1}(\mathbf{D})$.

7 A very low-level proof of proposition 3.6

Here we give in details explanations of how to construct the substitution α (in fact a slightly different α) of proposition 3.6. We produce directly an α on $\mathrm{Var}(Q^{-1}(\mathbf{D}))$. This is very low-level (and then less elegant), since it describes all possible configurations for the two free-tuples we want to unify to obtain a Q-source. It came for me in hands first.

Lemma 7.1 *Let $u = \langle e_1, \ldots, e_n \rangle$ and $v = \langle f_1, \ldots, f_n \rangle$ two free-tuples with same arity n. Assume it exists θ a mgu for u and v. Let $x, x' \in \mathrm{Sym}(u)$, such that $x \neq x'$ and $\theta(x) = \theta(x')$. Then we have $k \in \mathbb{N}^*$, and $g_0, g_2, \ldots, g_k \in \mathrm{Sym}(u) \cup \mathrm{Sym}(v)$, with $x = g_0$, $x' = g_k$, such that:*

$$\begin{gathered} \theta(x) = \theta(g_1) \\ \theta(g_1) = \theta(g_2) \\ \theta(g_2) = \theta(g_3) \\ \vdots \\ \theta(g_{k-1}) = \theta(x') \end{gathered}$$

and with $\forall\, i \in \{0, \ldots, k-1\}$, $g_i \in \mathrm{Sym}(u)$ $\big($resp. $\mathrm{Sym}(v)\big) \Rightarrow g_{i+1} \in \mathrm{Sym}(v)$ $\big($resp. $\mathrm{Sym}(u)\big)$. The integer k is called the length *of the* path *between x and x'. In the particular case where x' also in $\mathrm{Sym}(v)$ $\big($or when x also in $\mathrm{Sym}(v)\big)$, then this length equals one, with $x = g_0, x' = g_1$.*

Notes.
1. Of course x and x' cannot be both constants. So at least x or x' must be in $\mathrm{Var}(u)$.
2. We have a graphical interpretation of this lemma (see also figures).
We associate to each symbol a vertex (labeled with the symbol), and split this vertices into two regions: left for $\mathrm{Sym}(u)$ and right for $\mathrm{Sym}(v)$. We put an horizontal edge between each pair (e_i, f_i) $\big($this symbolize $\theta(e_i) = \theta(f_i)\big)$.
Now for $h_i, h_j \in \mathrm{Sym}(u) \cup \mathrm{Sym}(v)$, we add an edge between h_i and h_j if and only if $h_i = h_j$. Then the interpretation is the following (which is straightforward to verify):

$$\begin{array}{l} \exists\ g_i\text{'s with } g_0 = x,\ g_k = x' \\ \text{and} \left\{ \begin{array}{c} \theta(x) = \theta(g_1) \\ \theta(g_1) = \theta(g_2) \\ \theta(g_2) = \theta(g_3) \\ \vdots \\ \theta(g_{k-1}) = \theta(x') \end{array} \right. \\ \text{and } g_i \in \mathrm{Sym}(u) \Rightarrow g_{i+1} \in \mathrm{Sym}(v), \\ [\text{resp. } g_i \in \mathrm{Sym}(v) \Rightarrow g_{i+1} \in \mathrm{Sym}(u)] \end{array} \quad \Longleftrightarrow \quad \begin{array}{l} \text{there exists a path in the graph} \\ \text{between } x \text{ and } x'. \end{array}$$

Figure 1. Here $\mathrm{Sym}(u) \cap \mathrm{Sym}(v) = \emptyset$.
We have $x \neq x'$, $y \neq y'$, but $\theta(x) = \theta(x')$ and we see the path between x and x'.

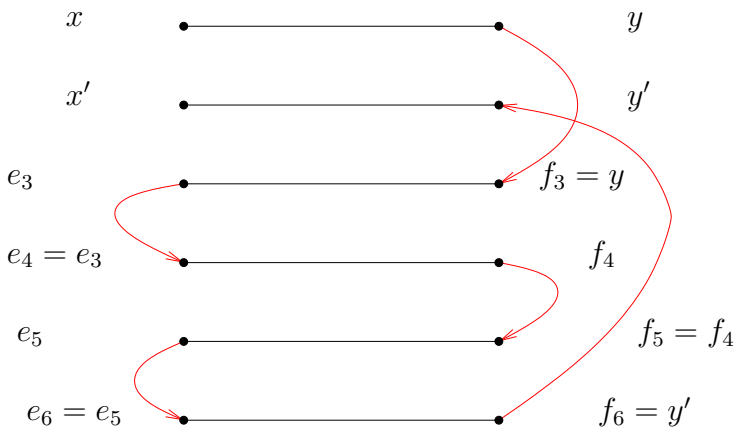
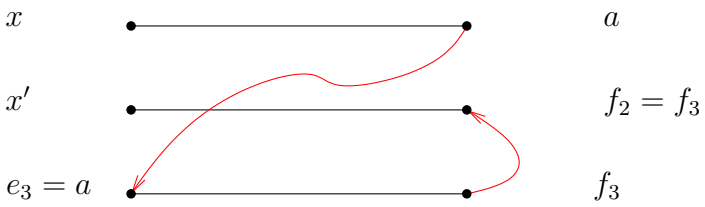


Figure 2. Here $\mathrm{Sym}(u) \cap \mathrm{Sym}(v) \neq \emptyset$.
We have $x \neq x'$, and $\theta(x) = \theta(x')$ by the path that we see between x and x'. A constant a here shows that constants may intervene.



Proof. (lemma 7.1)
So we consider $x, x' \in \mathrm{Sym}(u)$ such that $x \neq x'$ and $\theta(x) = \theta(x')$.
Let y, y' be the corresponding symbols in the free-tuple v, that is: $x = u[i] \Rightarrow y = v[i]$, and $x' = u[j] \Rightarrow y' = v[j]$ (for some $i, j \in \{1, \ldots, n\}$).
We have trivial cases if at least one of these equalities occurs: $x = y', x' = y, y = y'$.
Assume $y = y'$ for example. Then the path announced is simply given by: $\theta(x) = \theta(y)$
$\theta(y) = \theta(x')$.
Now assume that we have none of these trivial cases, and more generally assume there is no path in the associate graph (previously described) between x and x'.
This implies that the variable x' (and then also the symbol y') does not occur in each path containing x (if there exists at least one).
On the other hand x', y' may belong to a collection $\mathcal{P}$ of paths, each one not containing x (here a path is a set of symbols as we identify vertices and their symbol labels).
Necessarily there is no common symbol between $\mathcal{P}$ and paths containing x, otherwise there would be a path from x to x'.
Thus we can define a substitution θ' like this:
$\forall\ e \in \mathbf{Sym} \backslash \mathcal{P}$, $\theta'(e) = \theta(e)$,
$\forall\ e \in \mathcal{P}$, $\theta'(e) = h \in \mathbf{Var}$, with $\boldsymbol{h \neq \theta(x)}$.

Then clearly θ' is a unifier for u and v.
But θ is supposed to be a mgu for u and v, so it is more general than the unifier θ', and we have substitution ν with $\theta' = \nu \circ \theta$.
Then, using this, the equality $\theta(x) = \theta(x')$, and the definition of θ', we get:
$h = \theta'(x') = \nu(\theta(x')) = \nu(\theta(x)) = \theta'(x) = \theta(x)$ which is contradiction.
Hence we conclude that there must exists a path between x and x'.

Remark 7.2 On the preceding lemma.

1. Let $u = \langle x, x' \rangle$, $v = \langle y, y' \rangle$, with all of these four variables different from each other. Then we have a unifier ν for u and v defined by: $\nu(x) = \nu(x') = \nu(y) = \nu(y') = a \in \mathbf{Dom}$. This example shows that for a unifier which is not a mgu, we do not have a path as announced (between x and x'), although $\nu(x) = \nu(x')$. So the condition "mgu" **is required**.
2. Particular case. Assume $\mathrm{Var}\,(u) \cap \mathrm{Var}(v) = \emptyset$. Then the only possibilities for having "diagonal" edges between $\mathrm{Sym}(u)$ and $\mathrm{Sym}(v)$ in the graph, is when $\mathrm{Dom}(u) \cap \mathrm{Dom}(v) \neq \emptyset$. An instance of this is chosen for figure 2, and moreover we have taken a constant belonging to a path between x and x'.
3. General case. For the clearness of the proof we have taken x, x' "on the same side", that is here for instance in $\mathrm{Sym}\,(u)$, but obviously the result is still valid if they are not, it suffices to exchange the roles of x and y (which are linked up by an horizontal edge).

Proposition 7.3 *Recall proposition 3.6: we consider a rule $Q : \mathbf{S} \longrightarrow \mathbf{T}$ and*

- $\mathbf{I}$ *a database whose schema contains that one of* $\mathbf{S}$,
- *the database* $\mathbf{J} = Q(\mathbf{I})$,
- $\mathbf{D}$ *a database whose schema is* $\{T\}$,
- μ *a substitution on* $Var\,(\mathbf{D})$ *such that* $\mu(\mathbf{D}) \subseteq \mathbf{J}$.

Assuming that a Q-source for $\mathbf{D}$ exists, then we have a substitution α on $\mathrm{Var}(Q^{-1}(\mathbf{D}))$ such that: $\begin{cases} \imath) & \alpha(Q^{-1}(\mathbf{D})) \subseteq \mathbf{I}, \\ \imath\imath) & \mu(\mathbf{D}) \subseteq Q(\alpha(Q^{-1}(\mathbf{D}))). \end{cases}$

If moreover $\mathbf{I}$ is ground, then $\mathbf{J}$ is ground, and α may be chosen as a valuation.

Proof.
We use here notations of remark 3.5: we write $\boldsymbol{D} = \{T(v_1), T(v_2), \ldots, T(v_m)\}$ for some $m \geqslant 1$, and write then ρ_i the previous renamings $(i = 1, \ldots, m)$. We write also $\mathrm{V}= \langle v_1, \ldots, v_m \rangle$ and $\mathrm{W}= \langle \rho_1(v), \ldots, \rho_m(v) \rangle$ the concatenate free-tuples such that $\theta(\mathrm{V}) = \theta(\mathrm{W})$.
For all $i \in \{1, \ldots, m\}$, set $V_i = \mathrm{Var}(v_i)$ and $W_i = \mathrm{Var}(\rho_i(v))$ and let $\begin{cases} V = \cup_{i=1}^m V_i = \mathrm{Var}(\mathrm{V}), \\ W = \cup_{i=1}^m W_i = \mathrm{Var}(\mathrm{W}). \end{cases}$
With these notations we have the following facts:

1. Reformulation of 3.5.*3*: $Q^{-1}(\mathbf{D}) = \theta(\cup_{i=1}^m \rho_i(\mathbf{S})) = \cup_{i=1}^m \theta(\rho_i(\mathbf{S}))$.
2. $\forall\, 1 \leqslant i, j \leqslant m,\, i \neq j \Longrightarrow \rho_i\big(\mathrm{Var}(\mathbf{T})\big) \cap \rho_j\big(\mathrm{Var}(\mathbf{T})\big) = \emptyset$. This is obvious since using lemma 1.1.*5* we have $\rho_i\big(\mathrm{Var}(\mathbf{T})\big) \subseteq \rho_i\big(\mathrm{Var}(\mathbf{S})\big)$ for all $i \in \{1, \ldots, m\}$.

3. Free tuples v and w have disjoint sets of variables. *Proof.* The first's variable set is $\mathrm{Var}(\mathbf{D})$, the second is $\mathrm{Var}\big(\cup_{i=1}^m \rho_i(\mathbf{T})\big)$. Then, using lemma 1.1 (*2*, *3* and *4*), we have: $\mathrm{Var}\big(\cup_{i=1}^m \rho_i(\mathbf{T})\big) \subseteq \mathrm{Var}\big(\cup_{i=1}^m \rho_i(\mathbf{S})\big) \subseteq \cup_{i=1}^n \rho_i\big(\mathrm{Var}(\mathbf{S})\big)$, and this last set has empty intersection with $\mathrm{Var}(\mathbf{D})$, since $\forall\ i \in \{1,\dots,m\}$, $\rho_i\big(\mathrm{Var}(\mathbf{S})\big) \cap \mathrm{Var}(\mathbf{D}) = \emptyset$.

4. From 2.6.*1* we deduce: θ is a substitution on $V \cup W$ (that is: elsewhere = identity).

5. We have $V = \mathrm{Var}(\mathbf{D})$, and $W_i = \mathrm{Var}\big(\rho_i(\mathbf{T})\big)$ for all $i \in \{1,\dots,m\}$.

6. Previous point 3 says $V \cap W = \emptyset$, and we deduce from previous point 2 that $W_i \cap W_j = \emptyset$ for all $i \neq j \in \{1,\dots,m\}$, (with 1.1.*4*).

Fix $i \in \{1,\dots,m\}$, and consider the rule $Q_i :\ \rho_i(\mathbf{T}) \longleftarrow \rho_i(\mathbf{S})$.
By lemma 3.3.*3*, we have $Q_i(\mathbf{I}) = Q(\mathbf{I})$.
By hypothesis we have a substitution μ such that $\mu(\mathbf{D}) \subseteq \mathbf{J}$ and we have $\mathbf{J} = Q(\mathbf{I}) = Q_i(\mathbf{I})$.
Then each element $\mu(T(v_i)) \in \mu(\mathbf{D})$ belongs to $Q_i(\mathbf{I})$. Then by definition of $Q_i(\mathbf{I})$ we have a substitution ν_i on $\mathrm{Var}\,(\rho_i(\mathbf{T}))$ such that $\mu(T(v_i)) = \nu_i(T(\rho_i(v)))$ with $\nu_i(\rho_i(\mathbf{S})) \subseteq \mathbf{I}$.

Since $W = \cup_{i=1}^m W_i = \cup_{i=1}^m \mathrm{Var}\big(\rho_i(\mathbf{T})\big)$ is a partition, we can define a substitution ν on W like this: $\forall\ y \in W$, $\exists$ unique $i \in \{1,\dots,m\}$ with $y \in W_i$, and we set $\nu(y) = \nu_i(y)$.

Then, in terms of free-tuples and using **(1.0)**, we rewrite $\mu(T(v_i)) = \nu_i(T(\rho_i(v)))$ under the form $\mu(v_i) = \nu(\rho_i(v))$, and since this relation is valid for all $i \in \{1,\dots,m\}$, using lemma 1.3, we deduce: $\mu(\mathrm{v}) = \nu(\mathrm{w})$.

Now (using lemma 1.1) we define:

$$\boldsymbol{\mathcal{S}} = \cup_{i=1}^m \rho_i\big(\mathrm{Sym}(\mathbf{S})\big) = \cup_{i=1}^m \mathrm{Sym}\big(\rho_i(\mathbf{S})\big) = \mathrm{Sym}\big(\cup_{i=1}^m \rho_i(\mathbf{S})\big),$$

and we will prove the following fact:

$$\forall\ y, y' \in \boldsymbol{\mathcal{S}} \cup \mathbf{Dom},\ \theta(y) = \theta(y') \Longrightarrow \nu(y) = \nu(y') \qquad \mathbf{(7.3.1)}$$

Let then $y, y' \in \boldsymbol{\mathcal{S}} \cup \mathbf{Dom}$, and such that $\theta(y) = \theta(y')$. We will consider all possible cases.

First case. We assume $y, y' \in \boldsymbol{\mathcal{S}}$, with neither y, nor y' constant.

Then (with 1.1) we have $y, y' \in \mathrm{Var}\big(\cup_{i=1}^m \rho_i(\mathbf{S})\big) = \cup_{i=1}^m \mathrm{Var}\big(\rho_i(\mathbf{S})\big) = \cup_{i=1}^m \rho_i\big(\mathrm{Var}(\mathbf{S})\big)$.

Subcase 1. We assume $y \in \rho_i\big(\mathrm{Var}(\mathbf{S})\big) \backslash W_i$ and $y' \in \rho_j\big(\mathrm{Var}(\mathbf{S})\big) \backslash W_j$.

That means y (resp. y') does not belongs to the head of the rule Q_i (resp Q_j).
As ν_i and ν_j are respectively valuations on $\mathrm{Var}(\rho_i(\mathbf{T}))$ and $\mathrm{Var}(\rho_j(\mathbf{T}))$, we have:
$\nu(y) = \nu_i(y) = y$ and $\nu(y') = \nu_j(y') = y'$.
Now $y \in V = \mathrm{Var}(\mathbf{D})$ is impossible since $\mathrm{Var}(\mathbf{D}) \cap \rho_i\big(\mathrm{Var}(\mathbf{S})\big) = \emptyset$, and $y \in W$ is also impossible, else there would be $k \neq i$, such that $y \in W_k$, but (with lemma 1.1 points *2* and *4*), we have $W_k = \mathrm{Var}\,\big(\rho_k(\mathbf{T})\big) \subseteq \mathrm{Var}\,\big(\rho_k(\mathbf{S})\big) \subseteq \rho_k\big(\mathrm{Var}\,(\mathbf{S})\big)$, and by hypothesis this last set has empty intersection with $\rho_i\big(\mathrm{Var}\,(\mathbf{S})\big)$.
Then $y \notin V \cup W$, and the same holds for y'.
But as already noticed, θ is a substitution on $V \cup W$, then $\theta(y) = y$, and $\theta(y') = y'$.
Conclusion: $\nu(y) = y = \theta(y) = z$, and $\nu(y') = y' = \theta(y') = z'$, with $z = z'$.

Subcase 2. We assume $y \in \rho_i\big(\mathrm{Var}(\mathbf{S})\big)\backslash W_i$ and $y' =\in W_j$.

We have seen in first subcase that $\theta(y) = y \notin V \cup W$. On the other hand $y' \in W_j \subseteq V \cup W$. Using lemma 2.6.*1*, we have $\theta(y') \in \mathrm{Sym}(\mathrm{v}) \cup \mathrm{Sym}(\mathrm{w})$. But $\theta(y') = \theta(y) = y$, and y cannot be a constant since ρ_i is a renaming of variables. Then $\theta(y') \in V \cup W$, but $\theta(y)$ not.
Conclusion: that proves this case is empty.

Subcase 3. We assume $y \in W_i$ and $y' \in W_j$.

In that case, the first possibility is that $y = y'$, here nothing to prove, we have $\nu(y) = \nu(y')$. Assume $y \neq y'$. As $\theta(y) = \theta(y')$, we are in situation of lemma 7.1 with free-tuples v and w. Then we have $k \in \mathbb{N}^*$, and $g_0, g_2, \ldots, g_k \in \mathrm{Sym}\,(\mathrm{v}) \cup \mathrm{Sym}\,(\mathrm{w})$, with $y = g_0$, $y' = g_k$, such that:

$$\left|\begin{array}{l} \theta(y) = \theta(g_1) \\ \theta(g_1) = \theta(g_2) \\ \theta(g_2) = \theta(g_3) \\ \vdots \\ \theta(g_{k-1}) = \theta(y') \end{array}\right| \quad \begin{array}{l} \text{with } \forall\ i \in \{0, \ldots, k-1\} \\ g_i \in \mathrm{Sym}\,(\mathrm{v}) \Rightarrow g_{i+1} \in \mathrm{Sym}\,(\mathrm{w}) \\ \text{resp. :} \\ g_i \in \mathrm{Sym}\,(\mathrm{w}) \Rightarrow g_{i+1} \in \mathrm{Sym}\,(\mathrm{v}). \end{array}$$

Recall that we associate a graph to the symbols in $\mathrm{Sym}(\mathrm{v}) \cup \mathrm{Sym}(\mathrm{w})$, such that $\forall\ 1 \leqslant l \leqslant \#\mathrm{v}$, we have an horizontal line between $\mathrm{v}[l]$ and $\mathrm{w}[l]$. These horizontal lines represent the fact $\theta(\mathrm{v}) = \theta(\mathrm{w})$, but here also the fact $\mu(\mathrm{v}) = \nu(\mathrm{w})$.
Recall also that we add edge between $e, f \in \mathrm{Sym}(\mathrm{v}) \cup \mathrm{Sym}(\mathrm{w})$ if and only if $e = f$.
For here, we have already noticed that $V \cap W = \emptyset$, and then remark 7.2.*2* says that the only possibilities for having diagonal edges in the graph is when $\mathrm{Dom}(\mathrm{v}) \cap \mathrm{Dom}(\mathrm{w}) \neq \emptyset$.

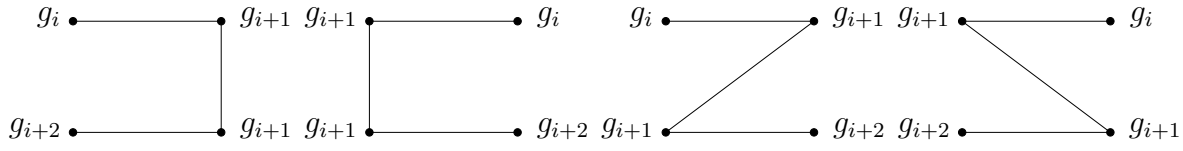
Now we have seen in lemma 7.1 that the previous list of equalities represents a path between y and y' in the graph.

Consider two consecutive equalities ($i \in \{0, \ldots, k-2\}$) in this previous list, which represent portion of this path:

$$\begin{cases} \theta(g_i) = \theta(g_{i+1}) \\ \theta(g_{i+1}) = \theta(g_{i+2}) \end{cases}$$

With respect to previous considerations, there is only four situations they may represent, which we have pictured below. As in lemma 3.7, we have separate $\mathrm{Sym}(\mathrm{w})$ on the left side and $\mathrm{Sym}(\mathrm{v})$ on the right side, and we have mentioned below each picture the corresponding situation for g_i, g_{i+1}, g_{i+2}.

Figure 3. The four possible portions of path between y and y'.



$g_i \in \mathrm{Sym}(\mathrm{w})$	$g_i \in \mathrm{Sym}(\mathrm{v})$	$g_i \in \mathrm{Sym}(\mathrm{w})$	$g_i \in \mathrm{Sym}(\mathrm{v})$
$g_{i+1} \in \mathrm{Sym}(\mathrm{v})$	$g_{i+1} \in \mathrm{Sym}(\mathrm{w})$	$g_{i+1} \in \mathbf{Dom}$	$g_{i+1} \in \mathbf{Dom}$
$g_{i+2} \in \mathrm{Sym}(\mathrm{w})$	$g_{i+2} \in \mathrm{Sym}(\mathrm{v})$	$g_{i+2} \in \mathrm{Sym}(\mathrm{v})$	$g_{i+2} \in \mathrm{Sym}(\mathrm{w})$

In each case, the corresponding equalities, using $\mu(\mathrm{v}) = \nu(\mathrm{w})$, are:

$$\begin{array}{cccc} \nu(g_i) = \mu(g_{i+1}) & \mu(g_i) = \nu(g_{i+1}) & \nu(g_i) = \mu(g_{i+1}) & \mu(g_i) = \nu(g_{i+1}) \\ \mu(g_{i+1}) = \nu(g_{i+2}) & \nu(g_{i+1}) = \mu(g_{i+2}) & \nu(g_{i+1}) = \mu(g_{i+2}) & \mu(g_{i+1}) = \nu(g_{i+2}) \end{array}$$

But now we go through the list of equalities from up to down, starting with $\nu(y) = \mu(g_1)$, and ending with $\mu(g_{k-1}) = \nu(y')$. So in each preceding four cases, we are interested in knowing if the value of the entry point equals the value of the leaving point.
This is clear for the first's two cases. For the last's two ones, fortunately g_{i+1} is a constant, therefore:
- in the third case, $\mu(g_{i+1}) = \nu(g_{i+1}) = g_{i+1}$,
- in the fourth case, $\nu(g_{i+1}) = \mu(g_{i+1}) = g_{i+1}$.

Then there is no breaking point, and we can conclude that $\nu(y) = \nu(y')$, which finishes this third subcase and the first global case.

Second case. We assume $y \in \boldsymbol{\mathcal{S}} \backslash \mathbf{Dom}$, and $y' \in \mathbf{Dom}$.

Then, as already mentioned, $y \in \cup_{i=1}^{m} \rho_i\big(\mathrm{Var}(\mathbf{S})\big)$.

Subcase 1. We assume $y \in \rho_i\big(\mathrm{Var}(\mathbf{S})\big) \backslash W_i$.

As already seen in precedent subcase 1, we have $\theta(y) = y$, but here $\theta(y') = y' \in \mathbf{Dom}$, and with $\theta(y) = \theta(y')$ we see this is an empty case.

Subcase 2. We assume $y \in W_i$ and set $y' = a$.

We have $\theta(y) = \theta(y') = a$, and $y \neq a$. Recall that $\theta(y) = a$ implies $a \in \mathrm{Sym}\,(\mathrm{v}) \cup \mathrm{Sym}\,(\mathrm{v})$, since lemma 2.6.*1*. So again we are in situation of lemma 7.1, (with remark 7.2.*3*), and there is a path between y and a.
The only difference with previous subcase 3 is that the ending point of the path, (that is a), might be on the right, that is in Sym(v). Assume this. Then, with previous notations, the last equality (which is possibly the only one) is: $\nu(g_{k-1}) = \mu(a)$.
Then, returning with first equality, we have $\nu(y) = \mu(a) = a = y' = \nu(y')$, which proves this subcase.

Third case. We assume $y, y' \in \mathbf{Dom}$.

Here it's obvious, $y = \theta(y) = \theta(y') = y' \Longrightarrow y = y' \Longrightarrow \nu(y) = \nu(y')$.

Then the proof of **(7.3.1)** is complete. Now come back to our proposition.

Definition of substitution α.

We have: $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big) = \mathrm{Var}\big(\cup_{i=1}^{m} \theta(\rho_i(\mathbf{S}))\big) = \cup_{i=1}^{m} \mathrm{Var}\big(\theta(\rho_i(\mathbf{S}))\big)$, (with 1.1.*2*), and (with 1.1.*4*), $\mathrm{Var}\big(\theta(\rho_i(\mathbf{S}))\big) \subseteq \theta\big(\mathrm{Var}(\rho_i(\mathbf{S}))\big) = \theta\big(\rho_i(\mathrm{Var}(\mathbf{S}))\big)$, for $i \in \{1, \ldots, m\}$. Then:

$\forall\ z \in \mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$, $\exists\ i \in \{1, \ldots, m\}$, $x \in \mathrm{Var}(\mathbf{S})$ and $y \in \rho_i\big(\mathrm{Var}\,(\mathbf{S})\big)$ with $\begin{cases} y = \rho_i(x) \\ z = \theta(y). \end{cases}$

Using this, we set: $\boldsymbol{\alpha(z)} = \begin{cases} \nu(y) = \nu(\rho_i(x)) = \nu_i(\rho_i(x)) \text{ if } z \in \mathrm{Var}\,\big(Q^{-1}(\mathbf{D})\big), \\ z \text{ elsewhere.} \end{cases}$

To show this define a substitution α on $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$, we have to verify before that produces a well-defined function, i. e. : $\forall\ z, z' \in \mathbf{Sym},\ z = z' \Rightarrow \alpha(z) = \alpha(z')$.

As α is set to be identity outside $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$, the case $z = z' \in \mathbf{Sym}\backslash\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$ is obvious. Now (using 1.1) we have:
$\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big) \subseteq \cup_{i=1}^{m}\theta\big(\rho_i(\mathrm{Var}(\mathbf{S}))\big) \subseteq \cup_{i=1}^{m}\theta\big(\rho_i(\mathrm{Sym}(\mathbf{S}))\big) = \theta\big(\cup_{i=1}^{m}\rho_i(\mathrm{Sym}(\mathbf{S}))\big) = \theta(\boldsymbol{\mathcal{S}})$,
then if $z = z' \in \mathrm{Var}\big(Q^{-1}(\mathbf{D})\big), \exists\, y, y' \in \boldsymbol{\mathcal{S}}$, such that $z = \theta(y),\ z' = \theta(y')$.
Therefore the result follows from **(7.3.1)**.

Proof of points *i)* and *ii)*.

Fix $i \in \{1, \ldots, m\}$. We first prove that $\forall\, x \in \mathrm{Var}(\mathbf{S})$, we have $\alpha\big(\theta(\rho_i(x))\big) = \nu_i\big(\rho_i(x)\big)$.
Let $x \in \mathrm{Var}(\mathbf{S})$. We have $\theta(\rho_i(x)) \in \theta\big(\rho_i(\mathrm{Var}(\mathbf{S}))\big) \supseteq \mathrm{Var}\big(\theta(\rho_i(\mathbf{S}))\big)$.
Assume first that $z = \theta(\rho_i(x)) \in \mathrm{Var}\big(\theta(\rho_i(\mathbf{S}))\big)$, which is a subset of $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$.
Then the definition of α is: $\alpha(z) = \nu_i\big(\rho_i(x)\big)$.
Assume now that $z = \theta(\rho_i(x)) \in \theta\big(\rho_i(\mathrm{Var}(\mathbf{S}))\big)\backslash\mathrm{Var}\big(\theta(\rho_i(\mathbf{S}))\big)$.
That clearly means $z = \theta(y)$ is a constant, say $a \in \mathbf{Dom}$. Notice that $y = \rho_i(x)$ is not a constant since ρ_i is a renaming.
So we have $\theta(y) = \theta(a)\ (= a)$, and $y = \rho_i(x) \in \rho_i\big(\mathrm{Var}(\mathbf{S})\big) \subseteq \rho_i\big(\mathrm{Sym}(\mathbf{S})\big) \subseteq \boldsymbol{\mathcal{S}}$.
Then we apply result **(7.3.1)** to obtain $\nu(y) = \nu(a) = a$.
Finally $\alpha\big(\theta(\rho_i(x))\big) = \alpha(a) = a = \nu(y) = \nu_i(\rho_i(x))$, which proves our statement.
Now we deduce from this, using lemma 1.3, that $\alpha\big(\theta(\rho_i(\mathbf{S}))\big) = \nu_i\big(\rho_i(\mathbf{S})\big)$. Hence:

i) The definition of ν_i give us $\nu_i\big(\rho_i(\mathbf{S})\big) \subseteq \mathbf{I}$. Then we deduce that for all $i \in \{1, \ldots, m\}$, we have: $\alpha\big(\theta(\rho_i(\mathbf{S}))\big) \subseteq \mathbf{I}$, and then $\alpha\big(Q^{-1}(\mathbf{D})\big) = \alpha\big(\cup_{i=1}^{m}\theta(\rho_i(\mathbf{S}))\big) = \cup_{i=1}^{m}\alpha\big(\theta(\rho_i(\mathbf{S}))\big) \subseteq \mathbf{I}$, therefore *i)*.

ii) We want to prove $\mu(\mathbf{D}) \subseteq Q\big(\alpha(Q^{-1}(\mathbf{D}))\big)$.
Recall that we have by definition of queries:
$Q\big(\alpha(Q^{-1}(\mathbf{D}))\big) = \{\nu(T(v)) \mid \nu$ substitution on $\mathrm{Var}\,(\mathbf{S})$ and $\nu(\mathbf{S}) \subseteq \alpha\big(Q^{-1}(\mathbf{D})\big)\}$.
Now consider $\mu(T(v_i)) \in \mu(\mathbf{D})$ (for some $i \in \{1, \ldots, m\}$).
Using definition of ν_i and **(1.0)**, we have $\mu(T(v_i)) = \nu_i\big(T(\rho_i(v))\big) = (\nu_i \circ \rho_i)(T(v))$, and then it remains to verify that $\nu_i \circ \rho_i$ is a substitution on $\mathrm{Var}(\mathbf{S})$ such that $(\nu_i \circ \rho_i)(\mathbf{S}) \subseteq \alpha(Q^{-1}(\mathbf{D}))$.
But we have just already seen $(\nu_i \circ \rho_i)(\mathbf{S}) = \alpha\big(\theta(\rho_i(\mathbf{S}))\big)$ and $\alpha(Q^{-1}(\mathbf{D})) = \cup_{i=1}^{m}\alpha\big(\theta(\rho_i(\mathbf{S}))\big)$.
$\big[$The fact that $\nu_i \circ \rho_i$ is a substitution on $\mathrm{Var}(\mathbf{S})$ is obvious, since $\mathrm{Var}\big(\rho_i(\mathbf{T})\big) \subseteq \mathrm{Var}\,\big(\rho_i(\mathbf{S})\big).\big]$

For conclusion of this proposition, it remains to see what happens if $\mathbf{I}$ is ground.

The fact that $Q(\mathbf{I})$ is ground follows from lemma 3.2.

Now consider α. We have seen:
$\forall\, z \in \mathrm{Var}\big(Q^{-1}(\mathbf{D})\big), \exists\, i \in \{1, \ldots, m\}$ and $y = \rho_i(x) \in \rho_i\big(\mathrm{Var}(\mathbf{S})\big)$ such that $z = \theta(y)$, and by definition $\alpha(z) = \nu(y) = \nu_i(\rho_i(x))$.
But with 1.1.*4*: $\nu_i(\rho_i(x)) \in \nu_i(\rho_i(\mathrm{Var}(\mathbf{S}))) \subseteq \nu_i(\rho_i(\mathrm{Sym}(\mathbf{S}))) = \mathrm{Sym}\big(\nu_i(\rho_i(\mathbf{S}))\big)$, and by definition of ν_i, we have $\nu_i(\rho_i(\mathbf{S})) \subseteq \mathbf{I}$, so with 1.1.*2*: $\mathrm{Sym}\big(\nu_i(\rho_i(\mathbf{S}))\big) \subseteq \mathrm{Sym}(\mathbf{I}) = \mathrm{Dom}(\mathbf{I})$, and therefore α has ground values on $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$.
Then clearly we can choose constants values for α on variables outside $\mathrm{Var}\big(Q^{-1}(\mathbf{D})\big)$, which make it a valuation. Therefore, α can actually be chosen as a valuation.

Note. The statement **(7.3.1)** is no longer true if we take $y, y' \in \mathbf{Sym}$, or even if we take $y, y' \in \mathrm{Sym}(\mathrm{v})$.

Example for **Sym**. Take $Q : S(x, z) \to T(a, z)$, $\mathbf{D} = \{T(y, a)\}$.

Here $\mathrm{v} = \langle y, a\rangle$, $\mathrm{w} = \langle a, y'\rangle$, we have $y \in \mathrm{Sym}(\mathrm{v})$, $y' \in \mathrm{Sym}(\mathrm{w})$, and this is the case where $\mathrm{Dom}(\mathrm{v}) \cap \mathrm{Dom}(\mathrm{w}) = \{a\} \neq \emptyset$.
We have $\theta(y) = \theta(a) = a$, $th(y') = \theta(a) = a$, so $\theta(y) = \theta(y')$, and the path between y and y' is clear through a.
Then also $\nu(a) = \mu(y)$, $\nu(y') = \mu(a)$, and because $y \notin \cup_{i=1}^{m} \mathrm{Var}(\rho_i(\mathbf{T})) = \mathrm{Var}(\{T(a, z))\}$, $\nu(y) = y$, which is not $\nu(y')$ $(= a)$.

Example for $\mathrm{Sym}(\mathrm{v})$. Take $Q : S(z, b) \to T(a, z, z)$, $\mathbf{D} = \{T(y, y', a)\}$.

Here $\mathrm{v} = \langle y, y', a\rangle$, $\mathrm{w} = \langle a, z, z\rangle$, then $y, y' \in \mathrm{Sym}(\mathrm{v})$, with $\mathrm{Dom}(\mathrm{v}) \cap \mathrm{Dom}(\mathrm{w}) = \{a\} \neq \emptyset$.
Here again $\theta(y) = \theta(y')$ via the path through a and z, but $y, y' \notin \cup_{i=1}^{m} \mathrm{Var}(\rho_i(\mathbf{T})) = \mathrm{Var}(\{T(a, z, z))\}$, then $\nu(y) = y \neq y' = \nu(y')$.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases*. Addison-Wesley, 1995.

2. J. Wijsen. *Database Repairing Using Updates*. ACM Transactions on Database Systems, 2005.

3. R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. *Data Exchange: Semantics and Query Answering*. Springer-Verlag, 2003.

4. M. Arenas, P. Barceló, R. Fagin, L. Libkin. *Locally Consistent Transformations and Query Answering in Data Exchange*. PODS 2004, ACM.