Chapter 6
Dual Tableaux for Relational Databases

6.1 Introduction

We present a calculus TRL of typed relations introduced in [MO04] which is intended to be a formal tool both for representing relational databases [Cod70] and also for reasoning with them. Typed relations are heterogeneous relations, i.e., the objects related with a relation may range over different domains. Three features of this calculus distinguish it from the calculus of ordinary relations in the Tarski-style.

First, associated with each relation is its type, which is a finite subset of a set whose members are interpreted as attributes. In this way we cope with the fact that database relations are determined by (finite) subsets of a set of attributes. Therefore, the relations of the calculus are relative in the sense suggested in [Orł88] and [DO02].

Second, as with ordinary relations, each typed relation has an arity, which is the cardinality of its type. However, for any $n \geq 1$, the order of the elements in the n-tuples belonging to a relation does not matter. This reflects the well-known property of database relations that the order of the attributes in the data table is immaterial. Tuples are treated as mappings that assign to an attribute an element of its domain.

Third, the calculus is comprised of relations of various arities and some operations may act on relations of not necessarily the same arity.

The basic operations on typed relations are intersection, complement, product, and projection. The other typical operations used in relational databases, namely, union, complement of a relation with respect to another relation, natural join, division, and selection are definable in terms of these basic operations. Analogous to the logics of binary relations, formulas of the logic of typed relations are built with typed relational terms and a string of object variables and/or constants of the appropriate length. We present a dual tableau for this logic and prove its completeness.

Next, following [BO97], we discuss a relational representation of database dependencies. All the typical dependencies can be expressed in terms of indiscernibility relations in the set of tuples of a relation in a relational database. Along these lines, any relation generated from indiscernibility relations with the operations from algebras of binary relations may be regarded as a generalized database dependency. This allows us to apply a rule extension of the dual tableau of logic $\mathsf{RL}(1, 1')$ to the verification of these dependencies and their implications. Since indiscernibility relations are equivalence relations we have to add the rules reflecting reflexivity, symmetry, and transitivity to the rules of the system for $\mathsf{RL}(1, 1')$. The dual tableau obtained in this way is sound and complete.

6.2 The Calculus of Typed Relations

Let Ω be an infinite set whose elements are referred to as *attributes*. To each $a \in \Omega$ there is associated a non-empty set D_a called the *domain of attribute a*. Types of relations, usually denoted by capital letters $A, B, \ldots$, etc., are finite subsets of Ω, including the empty set; clearly, if A and B are types, then $A \cup B$, $A \cap B$, and $A - B$ are types. $A \uplus B$ denotes the union of disjoint sets A' and B' obtained from A and B, respectively, by renaming their elements, if necessary. Consequently, card(A) = card(A'), card(B) = card(B'), and $A' \cap B' = \emptyset$. In particular, $A \uplus A$ is the union of two disjoint sets each of which has the same cardinality as A. This understanding of $\uplus$ allows us to assume that $\uplus$ is commutative and associative and that $A \uplus \emptyset = A$. It is a common practice in database systems to rename attributes as needed. To enable renaming, we assume that for every attribute $a \in \Omega$, there are infinitely many attributes a_i such that $D_{a_i} = D_a$. When forming $A \uplus B$, if $a' \in A'$ and $b' \in B'$ correspond to $a \in A$ and $b \in B$, respectively, it is necessary that $D_{a'} = D_a$ and $D_{b'} = D_b$. The set of all types will be denoted by $\mathbb{T}_\Omega$. Our definition of the disjoint union involves renaming implicitly.

Let $D_A = \bigcup\{D_a : a \in A\}$; then in particular, $D_\emptyset = \emptyset$. A tuple of type A is a map $u : A \to D_A$ such that for every $a \in A$, $u(a) \in D_a$. The collection of all tuples of type A is called the relation 1^A; for each $a \in \Omega$ the collection of tuples of type $\{a\}$ is denoted by 1^a. Let $1^\emptyset = \{e\}$, where e is the empty tuple. For each $a \in \Omega$, $D_a \neq \emptyset$; therefore $1^a \neq \emptyset$. Consequently, $1^A \neq \emptyset$ for all $A \in \mathbb{T}_\Omega$.

The above definitions imply that $1^{A \uplus B} = \{t : \exists u \in 1^A, \exists v \in 1^B$ such that if $a \in A$ then $t(a) = u(a)$ and if $b \in B$ then $t(b) = v(b)\}$. Observe that we have defined $\uplus$ so that $1^{A \uplus B} = 1^{B \uplus A}$, $1^{A \uplus (B \uplus C)} = 1^{(A \uplus B) \uplus C}$, and if $B \subseteq A$, then $1^A = 1^{B \uplus (A - B)}$. We often denote tuples $t \in 1^{A \uplus B}$ by uv, and say $t = uv$, where $u \in 1^A$ and $v \in 1^B$. Clearly, uv is a mapping, $uv : A \uplus B \to D_{A \uplus B}$, where $D_{A \uplus B} = D_A \cup D_B$. Our notation uv is analogous to the relational database notation for unions of sets of attributes: uv is the union of two mappings, where a mapping is understood as a set of pairs satisfying the well known functionality requirements. Thus $uv = vu$; similarly, $(uv)w = u(vw)$. Finally, for any $A \in \mathbb{T}_\Omega$, and for any $u \in 1^A$, $ue = eu = u$.

A relation R is said to be *of type A* whenever it is a subset of 1^A.

The basic operations on typed relations are as follows. Let $A, B \in \mathbb{T}_\Omega$:

- Intersection ($\cap$)
  Let $R, S \subseteq 1^A$; then $R \cap^A S \overset{\text{df}}{=} \{u \in 1^A : u \in R \text{ and } u \in S\}$.

- Projection (Π)

  Let $B \subseteq A$ and let $R \subseteq 1^A$; then $\Pi_B^A R \overset{\text{df}}{=} \{u \in 1^B : \exists v \in 1^{A-B}$ such that $uv \in R\}$.
- Product ($\times$)

  Let $R \subseteq 1^A$ and $S \subseteq 1^B$; then $R \times^{A \uplus B} S \overset{\text{df}}{=} \{uv \in 1^{A \uplus B} : u \in 1^A, v \in 1^B$, $u \in R$, and $v \in S\}$.
- Complement ($-$)

  Let $R \subseteq 1^A$; then $-^A R \overset{\text{df}}{=} (1^A - R) = \{u \in 1^A : u \notin R\}$.

We define the constant 0^A as follows: $0^A \overset{\text{df}}{=} -^A 1^A$. Clearly, $0^A = \emptyset$ for all $A \in \mathbb{T}_\Omega$. Union, $R \cup^A S$, and complement of S with respect to R, $R -^A S$, can easily be defined in terms of the above operations. Other operations are typically used in databases; we give their set theoretic definition and the corresponding expression in terms of the above four basic operations:

- Natural join ($\rhd\lhd$)

  Let $R \subseteq 1^A$ and $S \subseteq 1^B$; then $R \rhd\lhd^{A \cup B} S \overset{\text{df}}{=} \{uvw \in 1^{A \cup B} : u \in 1^{A-(A\cap B)}$, $v \in 1^{A \cap B}$, $w \in 1^{B-(A\cap B)}$, $uv \in R$, and $vw \in S\}$.

  The corresponding term of the calculus of typed relations is:

$$(R \times^{A \uplus (B-(A\cap B))} 1^{B-(A\cap B)}) \cap^{A \uplus (B-(A\cap B))} (S \times^{B \uplus (A-(B\cap A))} 1^{A-(B\cap A)}).$$

- Division ($\div$)

  Let $B \subseteq A$, let $R \subseteq 1^A$ and $S \subseteq 1^B$, $S \neq 0^B$; then $R \div_B^A S \overset{\text{df}}{=} \{t \in 1^{A-B} :$ $\forall s \in S, ts \in R\}$.

  The representation in the calculus of typed relations is:

$$\Pi_{A-B}^A R -^{A-B} (\Pi_{A-B}^A ((\Pi_{A-B}^A R \times^{(A-B)\uplus B} S) -^A R)).$$

A general notion of selection operation, namely *select S in R* is defined for any $B \subseteq A$, $S \subseteq 1^B$ and $R \subseteq 1^A$ as follows. Its application to such S and R yields the tuples $ut \in R$ such that $u \in S$. We give its set theoretic definition and an equivalent formulation in terms of the four basic operations:

- Selection (σ)

  Let $B \subseteq A$, let $R \subseteq 1^A$ and let $S \subseteq 1^B$; then $\sigma_B^A(S, R) \overset{\text{df}}{=} \{ut \in 1^A : u \in 1^B$, $t \in 1^{A-B}$, $u \in S$, and $ut \in R\}$.

  The representation in the calculus of typed relations is:

$$(S \times^{B \uplus (A-B)} 1^{A-B}) \cap^A R.$$

Now, we define a binary operation, $\odot_{a,b}$, which is useful to represent entailment. Let $R, S \subseteq 1^A$ and let $a, b \in A, a \neq b$. Then:

$$R \odot_{a,b} S \overset{\text{df}}{=} (1^{A-\{a,b\}} \times \Pi_a^A R) \times \Pi_b^A S.$$

In [IL84] it is shown that the operations of Codd's relational algebra [Cod70] may be defined in terms of intersection, complement, and cylindrification. It follows that Codd's relational algebra can be treated as a disguised version of cylindric set algebra. The approach to relational databases via cylindrification operation has the advantage that all operations are total, since all relations are of the same type. The disadvantage of this approach is that all relations are forced to be of the same arity, and in real-life databases query checking is computationally more efficient if we use relations with varying arities. Moreover, there is no completeness theorem for the cylindrical version of relational database theory. For this reason, we choose to develop a typed calculus, with the four basic operations defined above. Other operations definable in terms of intersection, projection, product, and complement include the update operations (see [Ngu91]) and other joins.

We can easily see that with typed relations we can express all of the fundamental notions of relational database theory: *schema* – a set of attributes, *relation over a schema* – a typed relation, *tuple* and *database* – a set of typed relations. We often write explicitly the types of both the operations and their argument relations, although some typing information may be redundant.

Some arithmetics laws of the calculus of typed relations are listed in the following propositions. The proofs can be found in [MO06].

Proposition 6.2.1 (Properties of Π^A_B). *For all $A, B \in \mathbb{T}_\Omega$ and for all $R, S \subseteq 1^A$, if $B \subseteq A$, then:*

1. $\Pi^A_B(1^A) = 1^B$, $\Pi^A_B(0^A) = 0^B$;
2. $\Pi^A_B(R \cup^A S) = \Pi^A_B R \cup^B \Pi^A_B S$;
3. $\Pi^A_B(R \cap^A S) \subseteq \Pi^A_B R \cap^B \Pi^A_B S$;
4. $-^B \Pi^A_B R \subseteq \Pi^A_B(-^A R)$;
5. $\Pi^A_A R = R$;
6. $\Pi^A_\emptyset R \subseteq 1^\emptyset$ *and if* $R \neq 0^A$, *then* $\Pi^A_\emptyset R = 1^\emptyset$.

Proposition 6.2.2 (Properties of σ^A_B). *For all $A, B \in \mathbb{T}_\Omega$, for all $R \subseteq 1^A$, and for all $S, T \subseteq 1^B$, if $B \subseteq A$, then:*

1. $\sigma^A_B(S \cup^B T, R) = \sigma^A_B(S, R) \cup^A \sigma^A_B(T, R)$;
2. $\sigma^A_B(S, R \cup^A T) = \sigma^A_B(S, R) \cup^A \sigma^A_B(S, T)$;
3. $\sigma^A_B(S \cap^B T, R) = \sigma^A_B(S, R) \cap^A \sigma^A_B(T, R)$;
4. $\sigma^A_B(S, R \cap^A T) = \sigma^A_B(S, R) \cap^A \sigma^A_B(S, T)$;
5. $\sigma^A_B(-^B S, R) \subseteq -^A \sigma^A_B(S, R)$;
6. $\sigma^A_B(0^B, R) = 0^A$;
7. $\sigma^A_B(S, 0^A) = 0^A$.

Proposition 6.2.3 (Properties of $\times$). *For all $A, B, C \in \mathbb{T}_\Omega$, for all $R \in 1^A, S \in 1^B, T \in 1^C$:*

1. $R \times^{A \uplus B} S = S \times^{B \uplus A} R$;
2. $(R \times^{A \uplus B} S) \times^{(A \uplus B) \uplus C} T = R \times^{A \uplus (B \uplus C)} (S \times^{B \uplus C} T)$.

Proposition 6.2.4 (Properties of $\rhd\lhd$). *For all $A, B, C \in \mathbb{T}_\Omega$, for all $R \in 1^A$, $S \in 1^B$, $T \in 1^C$:*

1. $R \rhd\lhd^{A\cup A} R = R$;
2. $R \rhd\lhd^{A\cup B} S = S \rhd\lhd^{B\cup A} R$;
3. $(R \rhd\lhd^{A\cup B} S) \rhd\lhd^{(A\cup B)\cup C} T = R \rhd\lhd^{A\cup(B\cup C)} (S \rhd\lhd^{B\cup C} T)$.

Proposition 6.2.5 (Properties of $\odot_{a,b}$). *For all $A \in \mathbb{T}_\Omega$, for all $a, b \in A$ such that $a \neq b$, and for all $R, S, P, Q \subseteq 1^A$, if $Q \neq \emptyset$, then:*

1. $1^A \odot_{a,b} -^A 1^A = 0^A$;
2. $(1^A \odot_{a,b} -^A 1^A) \odot_{a,b} 1^A = 0^A$;
3. $1^A \odot_{a,b} 1^A = 1^A$;
4. $R \odot_{a,b} 0^A = 0^A = 0^A \odot_{a,b} R$;
5. $R \odot_{a,b} (S \odot_{a,b} P) = (R \odot_{a,b} S) \odot_{a,b} P$.

Let $R, S \subseteq 1^A$ and let $a, b \in A, a \neq b$. Then:

$$R \supset^A_{a,b} S \overset{\text{df}}{=} ((1^A \odot_{a,b} -^A R) \odot_{a,b} 1^A) \cup^A S.$$

Proposition 6.2.6. *Let $R, S \subseteq 1^A$. Then for all $a, b \in A$ such that $a \neq b$ the following hold:*

1. $1^A \supset^A_{a,b} S = S$;
2. *If $R \neq 1^A$, then $R \supset^A_{a,b} S = 1^A$.*

The following theorem shows that entailment in the calculus of typed relations can be expressed in its language.

Theorem 6.2.1. *Let $A, B \subset \Omega$, let $R \subseteq 1^A, S \subseteq 1^B$, $a, b \in A, a \neq b$, and let $C = A \cup B$. Then the following are equivalent:*

1. *$R = 1^A$ implies $S = 1^B$;*
2. $R \times 1^{C-A} \supset^C_{a,b} S \times 1^{C-B} = 1^C$.

6.3 A Logic of Typed Relations

In this section we present a language of typed relations introduced in [MO06] whose intended models are databases. Let Ω be a set of attributes as defined in Sect. 6.2. Let $\mathbb{T}_\Omega$ be a set of types based on Ω. Then the expressions of the language of the logic TRL of typed relations over Ω are built from the following pairwise disjoint sets of symbols:

- $\{e\}$ – the empty tuple;
- $\mathbb{OV}^a_{\mathsf{TRL}}$ – an infinite set of object variables of type a, for each $a \in \Omega$; by $\mathbb{OV}^A_{\mathsf{TRL}}$ we denote the set $\bigcup_{a\in A} \mathbb{OV}^a_{\mathsf{TRL}}$, where $A \in \mathbb{T}_\Omega$;

- $\mathbb{OC}^a_{\mathsf{TRL}}$ – a set of object constants of type a, for each $a \in \Omega$; by $\mathbb{OC}^A_{\mathsf{TRL}}$ we denote the set $\bigcup_{a \in A} \mathbb{OC}^A_{\mathsf{TRL}}$, where $A \in \mathbb{T}_\Omega$;
- $\mathbb{OS}^a_{\mathsf{TRL}} = \mathbb{OV}^a_{\mathsf{TRL}} \cup \mathbb{OC}^a_{\mathsf{TRL}}$; by $\mathbb{OS}^A_{\mathsf{TRL}}$ we denote the set $\bigcup_{a \in A} \mathbb{OS}^a_{\mathsf{TRL}}$, where $A \in \mathbb{T}_\Omega$; we presume $\mathbb{OC}^\emptyset_{\mathsf{TRL}} = \{e\}$;
- $\mathbb{RV}^A_{\mathsf{TRL}}$ – a set of relational variables of type A, for each $A \in \mathbb{T}_\Omega$;
- $\mathbb{RC}^A_{\mathsf{TRL}}$ – a set of relational constants of type A, for each $A \in \mathbb{T}_\Omega$; $0^A, 1^A \in \mathbb{RC}^A_{\mathsf{TRL}}$;
- $\mathbb{OP}_{\mathsf{TRL}}$ – a set of operations of varying arities such that: for every k-ary operation $\# \in \mathbb{OP}_{\mathsf{TRL}}$, $k \geq 1$, there is associated a sequence $\tau(\#) = (A_1, \ldots, A_k, A)$ of $k+1$ elements of $\mathbb{T}_\Omega$, where A_i is the type of the ith argument of #, $i = 1, \ldots, k$, A is the type of the expression obtained by performing the operation #.

We presume: $\mathbb{OP}_{\mathsf{TRL}} \supseteq \{\Pi^A_B, \cup^A, \cap^A, -^A, \times^{A \uplus C} : A, B, C \in \mathbb{T}_\Omega, B \subseteq A\}$, where $\tau(\Pi^A_B) = (A, B)$, $\tau(\cup^A) = \tau(\cap^A) = (A, A, A)$, $\tau(-^A) = (A, A)$, and $\tau(\times^{A \uplus C}) = (A, C, A \uplus C)$. It follows that $\tau(\div^A_B) = (A, B, A-B)$, $\tau(\sigma^A_B) = (B, A, A)$, and for any $D \in \mathbb{T}_\Omega$, $\tau(\triangleright\triangleleft^{A \cup D}) = (A, D, A \cup D)$.

Assumptions concerning the elements of $\mathbb{OS}^{A \uplus B}_{\mathsf{TRL}}$ and $\mathbb{OS}^\emptyset_{\mathsf{TRL}}$, analogous to the corresponding assumptions on the set of tuples, are assumed to hold. It follows from the definitions that $\mathbb{OS}^{A \uplus B}_{\mathsf{TRL}} = \mathbb{OS}^{B \uplus A}_{\mathsf{TRL}}$ and $\mathbb{OS}^{(A \uplus B) \uplus C}_{\mathsf{TRL}} = \mathbb{OS}^{A \uplus (B \uplus C)}_{\mathsf{TRL}}$. As with the notation defined for the tuples, if u denotes a variable from $\mathbb{OV}^{A \uplus B}_{\mathsf{TRL}}$, it may be replaced by an expression vw, where $v \in \mathbb{OV}^A_{\mathsf{TRL}}$ and $w \in \mathbb{OV}^B_{\mathsf{TRL}}$. The set of atomic terms $\mathbb{RA}^A_{\mathsf{TRL}}$ is defined as the set $\mathbb{RV}^A_{\mathsf{TRL}} \cup \mathbb{RC}^A_{\mathsf{TRL}}$. We assume that for all A, $\mathbb{OS}^A_{\mathsf{TRL}} \neq \emptyset$ and $\mathbb{OS}^\emptyset_{\mathsf{TRL}} = \{e\}$.

For each $A \in \mathbb{T}_\Omega$, the set of terms of type A, $\mathbb{RT}^A_{\mathsf{TRL}}$, is the smallest set such that:

- $\mathbb{RA}^A_{\mathsf{TRL}} \subseteq \mathbb{RT}^A_{\mathsf{TRL}}$;
- If $\# \in \mathbb{OP}_{\mathsf{TRL}}$ is such that $\tau(\#) = (A_1, \ldots, A_m, A)$ and $F_i \in \mathbb{RT}^{A_i}_{\mathsf{TRL}}$, $i = 1, \ldots, m$, then $\#(F_1, \ldots, F_m) \in \mathbb{RT}^A_{\mathsf{TRL}}$.

A formula in the language of the logic TRL of typed relations over Ω is an expression of the form $F(u)$, where $F \in \mathbb{RT}^A_{\mathsf{TRL}}$ and $u \in \mathbb{OS}^A_{\mathsf{TRL}}$, for some $A \in \mathbb{T}_\Omega$.

A TRL-*model* for the language of the logic TRL of typed relations over Ω is a system $\mathcal{M} = \{\{A : A \in \mathbb{T}_\Omega\}, \{U^A : A \in \mathbb{T}_\Omega\}, e, m\}$, where U^A is a non-empty set of tuples of type A, $U^\emptyset = \{e\}$, and m is a meaning function subject to the following conditions:

- If $u \in \mathbb{OC}^A_{\mathsf{TRL}}$, then $m(u) \in U^A$ and if $u = vw$, then $m(u) = m(v)m(w)$; in addition, $m(e) = e$;
- If $R \in \mathbb{RA}^A_{\mathsf{TRL}}$, then $m(R) \subseteq U^A$; in particular, $m(0^A) = \emptyset$ and $m(1^A) = U^A$;
- If $\# \in \mathbb{OP}_{\mathsf{TRL}}$ and $\tau(\#) = (A_1, \ldots, A_k, A)$, then $m(\#)$ is a k-ary operation acting on relations of types $A_1, \ldots, A_k$ and returning a relation of type A: $m(\#) : 2^{U^{A_1}} \times \cdots \times 2^{U^{A_k}} \to 2^{U^A}$;
- m extends to all the compound relational terms as follows:
  if $F = \#(F_1, \ldots, F_k)$, then $m(F) = m(\#)(m(F_1), \ldots, m(F_k))$.

It is easy to see that each database over Ω determines a TRL-model.

A valuation in a TRL-model $\mathcal{M} = \{\{A : A \in \mathbb{T}_\Omega\}, \{U^A : A \in \mathbb{T}_\Omega\}, e, m\}$ is a function $v\colon \bigcup\{\mathbb{OS}^A_{\mathsf{TRL}}\colon A \in \mathbb{T}_\Omega\} \to \bigcup\{U^A\colon A \in \mathbb{T}_\Omega\}$ such that:

- If $u \in \mathbb{OV}^A_{\mathsf{TRL}}$, then $v(u) \in U^A$;
- If $u \in \mathbb{OC}^A_{\mathsf{TRL}}$, then $v(u) = m(u)$;
- If $u \in \mathbb{OS}^A_{\mathsf{TRL}}$ and $w \in \mathbb{OS}^B_{\mathsf{TRL}}$, then $v(uw) = v(u)v(w)$.

It follows from the definition of tuple that $v(ut) = v(tu)$, $v((ut)w) = v(u(tw))$, and $v(ue) = v(eu) = v(u)$.

Let $F \in \mathbb{RT}^A_{\mathsf{TRL}}$ and let $u \in \mathbb{OS}^A_{\mathsf{TRL}}$. The formula $F(u)$ is said to be satisfied in a TRL-model $\mathcal{M}$ by a valuation v whenever $v(u) \in m(F)$. We say that the formula $F(u)$ is true in the model $\mathcal{M}$ if and only if it is satisfied by all valuations in $\mathcal{M}$. Therefore if $u \in \mathbb{OV}^A_{\mathsf{TRL}}$, then $F(u)$ is true in the model $\mathcal{M}$ if and only if $m(F) = U^A$. We say that $F(u)$ is TRL-valid if it is true in all TRL-models.

6.4 Dual Tableau for the Logic of Typed Relations

We present a dual tableau for the language of typed relations whose semantics is determined by the class of TRL-models.

Decomposition rules have the following forms:

For all $F, G \in \mathbb{RT}^A_{\mathsf{TRL}}$, $B \subseteq A$, $u \in \mathbb{OS}^A_{\mathsf{TRL}}$, and $w \in \mathbb{OS}^B_{\mathsf{TRL}}$,

$$(\cup)\ \frac{(F \cup^A G)(u)}{F(u), G(u)} \qquad (-\cup)\ \frac{-^A(F \cup^A G)(u)}{-^A F(u) \mid -^A G(u)}$$

$$(\cap)\ \frac{(F \cap^A G)(u)}{F(u) \mid G(u)} \qquad (-\cap)\ \frac{-^A(F \cap^A G)(u)}{-^A F(u), -^A G(u)}$$

$$(-)\ \frac{-^A -^A F(u)}{F(u)}$$

$$(\Pi)\ \frac{(\Pi^A_B F)(w)}{F(wt), (\Pi^A_B F)(w)} \qquad t \in \mathbb{OS}^{A-B}_{\mathsf{TRL}} \text{ is any object symbol}$$

$$(-\Pi)\ \frac{-^B(\Pi^A_B F)(w)}{-^A F(wt)} \qquad t \in \mathbb{OV}^{A-B}_{\mathsf{TRL}} \text{ is a new object variable}$$

For all $F \in \mathbb{RT}^A_{\mathsf{TRL}}$, $G \in \mathbb{RT}^B_{\mathsf{TRL}}$, $v \in \mathbb{OS}^A_{\mathsf{TRL}}$, $w \in \mathbb{OS}^B_{\mathsf{TRL}}$, and $z \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$,

$$(\times)\ \frac{(F \times^{A \uplus B} G)(vw)}{F(v) \mid G(w)} \qquad (-\times)\ \frac{-^{A \uplus B}(F \times^{A \uplus B} G)(vw)}{-^A F(v), -^B G(w)}$$

Specific rules have the following forms:

For all $F \in \mathbb{RT}^A_{\mathsf{TRL}}$, $u_i \in \mathbb{OS}^{A_i}_{\mathsf{TRL}}$, $A = A_1 \uplus \ldots \uplus A_n$, $1 \le i \le n$,

$$(e)\quad \frac{F(u_1 \dots u_{i-1}(eu_i) \dots u_n)}{F(u_1 \dots u_{i-1}u_i \dots u_n),\, F(u_1 \dots u_{i-1}(eu_i) \dots u_n)}$$

$$(e')\quad \frac{F(u_1 \dots u_{i-1}u_i \dots u_n)}{F(u_1 \dots u_{i-1}(eu_i) \dots u_n),\, F(u_1 \dots u_{i-1}u_i \dots u_n)}$$

$$(\pi)\quad \frac{F(u_1 \dots u_n)}{F(u_{\pi(1)} \dots u_{\pi(n)})}$$

where π is a permutation on $\{1, \dots, n\}$

$$(\uplus)\quad \frac{F(u_1 \dots u_n)}{F(u_1 \dots u_{i-1}v_1v_2u_{i+1} \dots u_n),\, F(u_1 \dots u_n)}$$

$A_i = B_1 \uplus B_2$,
$v_1 \in \mathbb{OS}^{B_1}_{\mathsf{TRL}}$, $v_2 \in \mathbb{OS}^{B_2}_{\mathsf{TRL}}$ are any object symbols

The rules (e) and (e') reflect the interpretation of e as the empty tuple, the rule (π) reflects the fact that objects can be permuted without changing the meaning of a formula, and the rule $(\uplus)$ reflects the fact that any variable can be split into components in such a way that its type is preserved. The language includes variables of the empty type so that the rule $\uplus$ holds for all types.

A set of TRL-formulas is said to be TRL-axiomatic whenever it includes either of the sets of the following forms:

For any $u \in \mathbb{OS}^{A}_{\mathsf{TRL}}$, $R \in \mathbb{RA}^{A}_{\mathsf{TRL}}$, and $F \in \mathbb{RT}^{A}_{\mathsf{TRL}}$, $F \neq 0^A$,

(Ax1) $\{R(u), (-^A R)(u)\}$;
(Ax2) $\{1^A(u)\}$;
(Ax3) $\{-^A 0^A(u)\}$;
(Ax4) $\{(\Pi^A_{\emptyset} F)(e)\}$.

Rules for the defined operations may be given explicitly using their set theoretic formulation. We present some examples below.

Let $F \in \mathbb{RT}^{A}_{\mathsf{TRL}}$, $G \in \mathbb{RT}^{B}_{\mathsf{TRL}}$, $B \subseteq A$, $w \in \mathbb{OS}^{B}_{\mathsf{TRL}}$, $t \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$, then:

$$(\sigma)\quad \frac{(\sigma^A_B(G, F))(wt)}{G(w) \mid F(wt)} \qquad (-\sigma)\quad \frac{-^A(\sigma^A_B(G, F))(wt)}{-^B G(w),\, -^A F(wt)}$$

if, in addition, $B \neq \emptyset$, then:

$$(\div)\quad \frac{(F \div^A_B G)(t)}{-^B G(w),\, F(tw)} \qquad w \in \mathbb{OV}^{B}_{\mathsf{TRL}} \text{ is a new object variable}$$

$$(-\div)\quad \frac{-^{A-B}(F \div^A_B G)(t)}{G(w),\, -^{A-B}(F \div^A_B G)(t) \mid -^A F(tw),\, -^{A-B}(F \div^A_B G)(t)}$$

$w \in \mathbb{OS}^{B}_{\mathsf{TRL}}$ is any object symbol

Let $F \in \mathbb{RT}^{A}_{\mathsf{TRL}}, G \in \mathbb{RT}^{B}_{\mathsf{TRL}}, u \in \mathbb{OS}^{A-(A\cap B)}_{\mathsf{TRL}}, v \in \mathbb{OS}^{A\cap B}_{\mathsf{TRL}}, w \in \mathbb{OS}^{B-(A\cap B)}_{\mathsf{TRL}}$, then:

$$(\triangleright\triangleleft) \quad \frac{(F \triangleright\triangleleft^{A\cup B} G)(uvw)}{F(uv) \mid G(vw)} \qquad\qquad (-\triangleright\triangleleft) \quad \frac{-^{A\cup B}(F \triangleright\triangleleft^{A\cup B} G)(uvw)}{-^{A}F(uv), -^{B}G(vw)}$$

As usual, a TRL-set is a finite set of TRL-formulas such that the first-order disjunction of its members is true in all TRL-models. Correctness of a rule is defined in a similar way as in the relational logics of classical algebras of binary relations (see Sect. 2.4).

Proposition 6.4.1.

1. *The* TRL*-rules are* TRL*-correct;*
2. *The* TRL*-axiomatic sets are* TRL*-sets.*

Proof. By way of example, we show correctness of the rules $(-\Pi)$ and (e).

$(-\Pi)$ Let X be a finite set of TRL-formulas and let t be such that $t \neq w$ and it does not occur in X. Assume $X \cup \{-^{B}(\Pi^{A}_{B}F)(w)\}$ is a TRL-set. Suppose $X \cup \{-^{A}F(wt)\}$ is not a TRL-set. Then, there exist a TRL-model $\mathcal{M}$ and a valuation v in $\mathcal{M}$ such that for every $\varphi \in X \cup \{-^{A}F(wt)\}$, $\mathcal{M}, v \not\models \varphi$, which means that $v(w)v(t) \in m(F)$, where $v(w) \in 1^{B}$ and $v(t) \in 1^{A-B}$. Since $X \cup \{-^{B}(\Pi^{A}_{B}F)(w)\}$ is a TRL-set, $\mathcal{M}, v \models -^{B}(\Pi^{A}_{B}F)(w)$. Thus, for every $u \in 1^{A-B}$, $v(w)u \notin m(F)$, a contradiction. Now, assume that $X \cup \{-^{A}F(wt)\}$ is a TRL-set. Then for every TRL-model $\mathcal{M}$ and for every valuation v in $\mathcal{M}$, either there exists $\varphi \in X$ such that $\mathcal{M}, v \models \varphi$ or $\mathcal{M}, v \models -^{A}F(wt)$. By the assumption on variable t, it follows that either there exists $\varphi \in X$ such that $\mathcal{M}, v \models \varphi$ or for every $u \in 1^{A-B}$, $v(w)u \notin m(F)$. Hence, $X \cup \{-^{B}(\Pi^{A}_{B}F)(w)\}$ is a TRL-set.

(e) Let X be a finite set of TRL-formulas. If $X \cup \{F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n)\}$ is a TRL-set, then so is $X \cup \{F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n), F(u_1 \ldots u_{i-1}u_i \ldots u_n)\}$. Assume $X \cup \{F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n), F(u_1 \ldots u_{i-1}u_i \ldots u_n)\}$ is a TRL-set. Suppose $X \cup \{F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n)\}$ is not a TRL-set. Then, there exist a TRL-model $\mathcal{M}$ and a valuation v in $\mathcal{M}$ such that for every $\varphi \in X \cup \{F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n)\}$, $\mathcal{M}, v \not\models \varphi$, so $v(u_1) \ldots v(u_{i-1})v(eu_i) \ldots v(u_n) \notin m(F)$. Since $v(eu_i) = v(u_i)$, we obtain $v(u_1) \ldots v(u_{i-1})v(u_i) \ldots v(u_n) \notin m(F)$. However, by the assumption $\mathcal{M}, v \models F(u_1 \ldots u_{i-1}u_i \ldots u_n)$. Hence, $v(u_1) \ldots v(u_{i-1})v(u_i) \ldots v(u_n) \in m(F)$, a contradiction. □

The notions of a TRL-proof tree, a closed branch of such a tree, a closed TRL-proof tree, and TRL-provability are defined as in Sect. 2.4.

Due to Proposition 6.4.1, we obtain:

Proposition 6.4.2. *Let φ be a* TRL*-formula. If φ is* TRL*-provable, then it is* TRL*-valid.*

Below we list the completion conditions that are specific for the logic TRL. The completion conditions for the rules $(\cap)$, $(-\cap)$, and $(-)$ are as in Sect. 2.5.

For all $F, G \in \mathbb{RT}^{A}_{\mathsf{TRL}}$, $B \subseteq A$, $u \in \mathbb{OS}^{A}_{\mathsf{TRL}}$, and $w \in \mathbb{OS}^{B}_{\mathsf{TRL}}$, Cpl($\Pi$) (resp. Cpl($-\Pi$)) If $(\Pi^A_B F)(u) \in b$ (resp. $(-^B \Pi^A_B F)(u) \in b$), then for every $t \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$, $F(ut) \in b$ (resp. for some $t \in \mathbb{OV}^{A-B}_{\mathsf{TRL}}$, $-^A F(ut) \in b$), obtained by an application of the rule (Π) (resp. ($-\Pi$));

For all $F \in \mathbb{RT}^A_{\mathsf{TRL}}$, $G \in \mathbb{RT}^B_{\mathsf{TRL}}$, $v \in \mathbb{OS}^A_{\mathsf{TRL}}$, $w \in \mathbb{OS}^B_{\mathsf{TRL}}$, and $z \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$,

Cpl($\times$) (resp. Cpl($-\times$)) If $(F \times^{A \uplus B} G)(vw) \in b$ (resp. $-^{A \uplus B}(F \times^{A \uplus B} G)(vw) \in b$), for some $v \in \mathbb{OS}^A_{\mathsf{TRL}}$ and $w \in \mathbb{OS}^B_{\mathsf{TRL}}$, then either $F(v) \in b$ or $G(w) \in b$ (resp. both $-^A F(v) \in b$ and $-^B G(w) \in b$), obtained by an application of the rule ($\times$) (resp. ($-\times$));

For all $F \in \mathbb{RT}^A_{\mathsf{TRL}}$, $u_i \in \mathbb{OS}^{A_i}_{\mathsf{TRL}}$, $A = A_1 \uplus \ldots \uplus A_n$, $1 \leq i \leq n$,

Cpl(e) If $F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n) \in b$, for $n \geq 1$ and $1 \leq i \leq n$, then $F(u_1 \ldots u_{i-1} u_i \ldots u_n) \in b$, obtained by an application of the rule (e);
Cpl(e') If $F(u_1 \ldots u_{i-1}(u_i) \ldots u_n) \in b$, for $n \geq 1$ and $1 \leq i \leq n$, then $F(u_1 \ldots u_{i-1}(eu_i) \ldots u_n) \in b$, obtained by an application of the rule (e');
Cpl(π) If $F(u_1 \ldots u_n) \in b$, then for any permutation π of the indices $1, \ldots, n$, $F(u_{\pi(1)} \ldots u_{\pi(n)}) \in b$, obtained by an application of the rule (π);
Cpl($\uplus$) If $F(u_1 \ldots u_n) \in b$, for $u_i \in \mathbb{OS}^{A_i}_{\mathsf{TRL}}$, $1 \leq i \leq n$, and $A_i = B_1 \uplus B_2$, then for all $v_1 \in \mathbb{OS}^{B_1}_{\mathsf{TRL}}$, $v_2 \in \mathbb{OS}^{B_2}_{\mathsf{TRL}}$, $F(u_1 \ldots u_{i-1} v_1 v_2 u_{i+1} \ldots u_n) \in b$, obtained by an application of the rule ($\uplus$).

The notions of a complete TRL-proof tree and an open branch of a TRL-proof tree are defined as in RL-logic (see Sect. 2.5). Due to the forms of the rules of TRL-dual tableau, we can prove that whenever a branch of a TRL-proof tree contains both of the formulas $R(u)$ and $(-^A R)(u)$, for $u \in \mathbb{OS}^A_{\mathsf{TRL}}$ and for an atomic $R \in \mathbb{RA}^A_{\mathsf{TRL}}$, then the branch can be closed. Thus, the closed branch property can be proved.

Let b be an open branch of a TRL-proof tree. A branch structure

$$\mathcal{M}^b = \{\{A : A \in \mathbb{T}_\Omega\}, \{U^A : A \in \mathbb{T}_\Omega\}, e^b, m^b\}$$

is defined as follows:

- $U^A = \mathbb{OS}^A_{\mathsf{TRL}}$, for any $\emptyset \neq A \in \mathbb{T}_\Omega$;
- $U^\emptyset = \{e\}$ and $e^b = m^b(e) = e$;
- $m^b(c) = c$, for every $c \in \mathbb{OC}^A_{\mathsf{TRL}}$;
- $m^b(R) = \{u \in \mathbb{OS}^A_{\mathsf{TRL}} : R(u) \notin b\}$, for every $R \in \mathbb{RA}^A_{\mathsf{TRL}}$;
- For every $\# \in \mathbb{OP}_{\mathsf{TRL}}$, $m^b(\#)$ is defined as in TRL-models;
- m^b extends to all the compound relational terms as in TRL-models.

It is easy to see that the branch structure defined above is a TRL-model. Hence, the branch model property holds. Let v^b be a valuation in $\mathcal{M}^b$ such that $v^b(u) = u$, for every $u \in \mathbb{OS}_{\mathsf{TRL}}$.

Proposition 6.4.3 (Satisfaction in Branch Model Property). *For every open branch b of a* TRL*-proof tree and for every* TRL*-formula φ, if $\mathcal{M}^b, v^b \models \varphi$, then $\varphi \notin b$.*

Proof. The proof is by induction on the complexity of formulas. The atomic case can be proved as in Sect. 2.5, it uses the closed branch property. By way of example, we show the proposition for some compound formulas that are specific for TRL-logic.

Let $\varphi = (\Pi^A_B F)(w)$. Assume $\mathcal{M}^b, v^b \models (\Pi^A_B F)(w)$. Then there exists $u \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$ such that $wu \in m^b(F)$, and by the induction hypothesis, $F(wu) \notin b$. Suppose $(\Pi^A_B F)(w) \in b$. Then by the completion condition Cpl(Π), for every $u \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}$, $F(wu) \in b$, a contradiction.

Let $\varphi = -^{A \uplus B}(F \times^{A \uplus B} G)(uw)$. Assume $\mathcal{M}^b, v^b \models -^{A \uplus B}(F \times^{A \uplus B} G)(uw)$. Then $u \notin m^b(F)$ or $w \notin m^b(G)$. Suppose $-^{A \uplus B}(F \times^{A \uplus B} G)(uw) \in b$. Then, by the completion condition Cpl($-\times$), $-^A F(u) \in b$ and $-^B G(w) \in b$. Thus, by the induction hypothesis, $u \in m^b(F)$ and $w \in m^b(G)$, a contradiction. □

As usual, Proposition 6.4.3 enables us to prove:

Proposition 6.4.4. *Let φ be a TRL-formula. If φ is TRL-valid, then it is TRL-provable.*

Finally, by Propositions 6.4.2 and 6.4.4, we have:

Theorem 6.4.1 (Soundness and Completeness of TRL). *For every TRL-formula φ, the following conditions are equivalent:*

1. *φ is TRL-valid;*
2. *φ is TRL-provable.*

Some other relational approaches to databases can be found in [DM01, Mac00, Mac01].

Theorem 6.4.2.

1. *Each of the Boolean algebra identities is TRL-provable;*
2. *Each of the expressions in Propositions 6.2.1, …, 6.2.5 is TRL-provable.*

Example. Consider the following property: if R is a relation of type A and $B \subseteq A$, then $R \subseteq (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B}$. To prove that this property holds in all TRL-models it suffices to demonstrate that the formula $(R \rightarrow^A (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B})(u)$ is TRL-provable, where $F \rightarrow^A G$ is defined as $-^A F \cup^A G$. If $u \in \mathbb{OV}^A_{\mathsf{TRL}}$, then $(F \rightarrow^A G)(u)$ is TRL-valid iff for all TRL-models, $m(F) \subseteq m(G)$. Figure 6.1 presents a TRL-proof of the formula

$$(-^A R \cup^A (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B})(u).$$

Figure 6.2 shows TRL-provability of the formula:

$$(T \times^{A_1 \uplus A_2} \sigma^{A_2}_B(S, R) \rightarrow^{A_1 \uplus A_2} \sigma^{A_1 \uplus A_2}_B(S, T \times R))(u).$$

$$\begin{tikzcd}
& \underline{(-^A R \cup^A (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B})(u)} \arrow[d, "(\uplus) \text{ with } v \in \mathbb{OS}^A_{\mathsf{TRL}} \text{ and } w \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}"] & \\
& \underline{(-^A R \cup^A (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B})(vw)} \arrow[d, "(\cup)"] & \\
& (-^A R)(vw), \underline{(\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B}(vw)}, \ldots \arrow[dl] \arrow[dr] \arrow[d, phantom, "(\times)"] & \\
(-^A R)(vw), \underline{(\Pi^A_B R)(v)}, \ldots \arrow[d, "(\Pi) \text{ with } w \in \mathbb{OS}^{A-B}_{\mathsf{TRL}}"] & & \begin{array}{c}(-^A R)(vw), 1^{A-B}(w), \ldots \\ \text{closed}\end{array} \\
\begin{array}{c}(-^A R)(vw), R(vw), (\Pi^A_B R)(v), \ldots \\ \text{closed}\end{array} & &
\end{tikzcd}$$

Fig. 6.1 A TRL-proof of the formula $(-^A R \cup^A (\Pi^A_B R) \times^{B \uplus (A-B)} 1^{A-B})(u)$

Its equivalent form is:

$$(-^{A_1 \uplus A_2}(T \times^{A_1 \uplus A_2} \sigma^{A_2}_B(S, R)) \cup^{A_1 \uplus A_2} \sigma^{A_1 \uplus A_2}_B(S, T \times R))(u).$$

6.5 Relational Representation of Database Dependencies

Let a universe Ω of attributes be given and let A be a finite subset of Ω. We recall that a database relation over A, R^A, is a set of tuples of type A. We shall often omit the index A in the name of a database relation. Given a database relation R, by AT_R we mean the underlying set of attributes. Following the usual database notation, for a subset B of A and for a tuple $t \in R^A$ we write $t[B]$ for a restriction of t (as a mapping) to set B.

Given a database relation R over A and a subset B of A, we define an *indiscernibility relation in R* as follows. For any $t, u \in R$, $(t, u) \in ind(B)$ iff $t[B] = u[B]$.

Example. Consider relation R defined in Table 6.1. Indiscernibility relation *ind*(a) determined by attribute a consists of the following pairs of tuples:

$$ind(a) = \{(t_1, t_1), (t_2, t_2), (t_3, t_3), (t_4, t_4), (t_1, t_4), (t_2, t_3), (t_3, t_2)\}.$$

Indiscernibility relation *ind*(bc) determined by attributes b and c consists of the following tuples:

$$ind(bc) = \{(t_1, t_1), (t_2, t_2), (t_3, t_3), (t_4, t_4), (t_1, t_2), (t_2, t_1)\}.$$

$$\begin{tikzcd}[column sep=small]
& \underline{(-^{A_1\uplus A_2}(T \times^{A_1\uplus A_2} \sigma_B^{A_2}(S,R)) \cup^{A_1\uplus A_2} \sigma_B^{A_1\uplus A_2}(S, T\times R))(u)} \arrow[d, "(\cup)"] & & \\
& \underline{-^{A_1\uplus A_2}(T \times^{A_1\uplus A_2} \sigma_B^{A_2}(S,R))(u)},\ (\sigma_B^{A_1\uplus A_2}(S, T\times R))(u) \arrow[d, "(\uplus)\text{ with } z \in \mathbb{OS}^{A_1}_{\mathsf{TRL}} \text{ and } t \in \mathbb{OS}^{A_2}_{\mathsf{TRL}}"] & & \\
& \underline{-^{A_1\uplus A_2}(T \times^{A_1\uplus A_2} \sigma_B^{A_2}(S,R))(zt)},\ (\sigma_B^{A_1\uplus A_2}(S, T\times R))(u), \ldots \arrow[d, "(-\times)"] & & \\
& -^{A_1}T(z),\ \underline{-^{A_2}\sigma_B^{A_2}(S,R)(t)},\ (\sigma_B^{A_1\uplus A_2}(S, T\times R))(u), \ldots \arrow[d, "(\uplus)\text{ with } v \in \mathbb{OS}^{B}_{\mathsf{TRL}} \text{ and } x \in \mathbb{OS}^{A_2-B}_{\mathsf{TRL}}"] & & \\
& -^{A_1}T(z),\ \underline{-^{A_2}\sigma_B^{A_2}(S,R)(vx)},\ (\sigma_B^{A_1\uplus A_2}(S, T\times R))(u), \ldots \arrow[d, "(-\sigma)"] & & \\
& -^{A_1}T(z),\ -^{B}S(v),\ -^{A_2}R(vx),\ \underline{(\sigma_B^{A_1\uplus A_2}(S, T\times R))(u)}, \ldots \arrow[d, "(\uplus)\text{ with } v \in \mathbb{OS}^{B}_{\mathsf{TRL}} \text{ and } q \in \mathbb{OS}^{A_1\uplus A_2-B}_{\mathsf{TRL}}"] & & \\
& -^{A_1}T(z),\ -^{B}S(v),\ -^{A_2}R(vx),\ \underline{(\sigma_B^{A_1\uplus A_2}(S, T\times R))(vq)}, \ldots \arrow[dl] \arrow[dr] \arrow[d, phantom, "(\sigma)"] & & \\
\begin{array}{c} -^{A_1}T(z),\ -^{B}S(v), \\ -^{A_2}R(vx),\ S(v), \ldots \\ \text{closed} \end{array} & & -^{A_1}T(z),\ -^{B}S(v),\ -^{A_2}R(vx),\ \underline{(T \times^{A_1\uplus A_2} R)(vq)}, \ldots \arrow[d, "\begin{array}{l}(\uplus)\text{ with } z \in \mathbb{OS}^{A_1}_{\mathsf{TRL}} \\ \text{and } x \in \mathbb{OS}^{A_2}_{\mathsf{TRL}}\end{array}"] & \\
& & -^{A_1}T(z),\ -^{B}S(v),\ -^{A_2}R(vx),\ \underline{(T \times^{A_1\uplus A_2} R)(vzx)}, \ldots \arrow[d, "(\pi)"] & \\
& & -^{A_1}T(z),\ -^{B}S(v),\ -^{A_2}R(vx),\ \underline{(T \times^{A_1\uplus A_2} R)(zvx)}, \ldots \arrow[dl] \arrow[dr] \arrow[d, phantom, "(\times)"] & \\
& \begin{array}{c} -^{A_1}T(z),\ -^{B}S(v), \\ -^{A_2}R(vx),\ T(z), \ldots \\ \text{closed} \end{array} & & \begin{array}{c} -^{A_1}T(z),\ -^{B}S(v), \\ -^{A_2}R(vx),\ R(vx), \ldots \\ \text{closed} \end{array}
\end{tikzcd}$$

Fig. 6.2 A TRL-proof of $(-^{A_1\uplus A_2}(T \times^{A_1\uplus A_2} \sigma_B^{A_2}(S,R)) \cup^{A_1\uplus A_2} \sigma_B^{A_1\uplus A_2}(S, T\times R))(u)$

Table 6.1 A database relation

	Attributes		
Tuples	a	b	c
t_1	1	2	3
t_2	4	2	3
t_3	4	5	3
t_4	1	5	6

In the following proposition we recall some basic properties of the indiscernibility relations.

Proposition 6.5.1. *For any database relation R and for all $A, B \subseteq AT_R$, the following conditions are satisfied:*

1. $ind(AT_R) = \{(t,t) : t \in R\}$, $ind(\emptyset) = R \times R$;
2. $ind(A \cup B) = ind(A) \cap ind(B)$;
3. $ind(A) \cup ind(B) \subseteq ind(A) ; ind(B)$;
4. *If* $A \subseteq B$, *then* $ind(B) \subseteq ind(A)$;
5. $ind(A) = \bigcap\{ind(a) : a \in A\}$.

Let a database relation R be given. Various attribute dependencies in R can be defined in terms of indiscernibility relations. We recall the standard definitions of those dependencies and their relational representation developed in [Orł87]. Let $A, B, C \subseteq AT_R$.

Functional Dependency

$A \rightarrow B$ holds in R iff for all tuples $t, u \in R$, if $t[A] = u[A]$, then $t[B] = u[B]$.

Proposition 6.5.2. *The following conditions are equivalent:*

1. *Functional dependency* $A \rightarrow B$ *holds in* R;
2. $ind(A) \subseteq ind(B)$.

Multivalued Dependency

$A \rightarrow\rightarrow B$ holds in R iff for all tuples $t, u \in R$, if $t[A] = u[A]$, then there exists $t' \in R$ such that the following conditions are satisfied:

- $t'[A \cup B] = t[A \cup B]$;
- $t'[AT_R - (A \cup B)] = u[AT_R - (A \cup B)]$.

Proposition 6.5.3. *The following conditions are equivalent:*

1. *Multivalued dependency* $A \rightarrow\rightarrow B$ *holds in* R;
2. $ind(A) \subseteq ind(A \cup B) ; ind(AT_R - (A \cup B))$.

Embedded Multivalued Dependency

$A \rightarrow\rightarrow B|C$ holds in R iff $A \rightarrow\rightarrow B$ holds in the set $\{t[A \cup B \cup C] : t \in R\}$.

Proposition 6.5.4. *The following conditions are equivalent:*

1. *Embedded multivalued dependency $A \rightarrow\rightarrow B|C$ holds in R;*
2. *$ind(A) \subseteq ind(A \cup B) ; ind((C-(A \cup B)))$.*

Decomposition

(A, B) holds in R iff $A \cup B = AT_R$ and for all $t, u \in R$, if $t[A \cap B] = u[A \cap B]$, then there exists $t' \in R$ such that $t'[A] = t[A]$ and $t'[B] = u[B]$.

Proposition 6.5.5. *The following conditions are equivalent:*

1. *Decomposition (A, B) holds in R;*
2. *$A \cup B = AT_R$ and $ind(A \cap B) \subseteq ind(A) ; ind(B)$.*

Join Dependency

$^*(A_1, \dots, A_n)$ holds in R iff $A_1, \dots, A_n = AT_R$ and R is the join of relations $\{t[A_i] : t \in R\}$, for $i = 1, \dots, n$.

Proposition 6.5.6. *Join dependency $^*(A_1, \dots, A_n)$ holds in R implies $ind(A_1 \cap \dots \cap A_n) \subseteq ind(A_1) ; \dots ; ind(A_n)$.*

A dependency defined by the condition $ind(A_1 \cap \dots \cap A_n) \subseteq ind(A_1) ; \dots ; ind(A_n)$, for $A_1, \dots, A_n$ such that $A_1 \cup \dots A_n = AT_R$, is called a *generalized join dependency*.

We conclude that given a database relation R, any relation generated from relations $ind(a)$, for $a \in AT_R$, with the operations of algebras of binary relations may be viewed as a kind of database dependency.

6.6 Dual Tableau for Database Dependencies

Due to the relational representation of database dependencies we can verify dependencies and implications of dependencies within a relational logic. The logic adequate for this purpose, $\mathsf{RL}_{\mathsf{EQ}}$, is obtained from $\mathsf{RL}(1, 1')$ by restricting its class of models to the models where the meanings of relation variables are assumed to be equivalence relations. This assumption is in agreement with the fact that dependencies are represented with relations generated by indiscernibility relations which in turn are equivalence relations.

A relational logic obtained from $\mathsf{RL}_{\mathsf{EQ}}$ by restricting its class of models to the models whose universes are sets of tuples of a database relation, is referred to as a *relational logic with database models*, $\mathsf{RL}_{\mathsf{DEP}}$.

Given a database relation R, an $\mathsf{RL}_{\mathsf{DEP}}$*-model* is a structure $\mathcal{M} = (R, m)$ such that:

- R is the set of tuples;
- $m(P) \in \{ind_R(a) : a \in AT_R\}$, for every relational variable P;
- $m(1) = R \times R$;
- $m(1') = Id_R$.

Since AT_R is finite for any database relation R, the image under m of the set of relational variables is finite in any database model. Clearly, $\mathsf{RL_{DEP}}$-model is an $\mathsf{RL_{EQ}}$-model. On the other hand, for every $\mathsf{RL_{EQ}}$-model we can construct an $\mathsf{RL_{DEP}}$-model such that the models verify the same formulas, that is we have the following proposition.

Proposition 6.6.1.

1. *For every* $\mathsf{RL_{DEP}}$*-model there exists an* $\mathsf{RL_{EQ}}$*-model* $\mathcal{M} = (U, m)$ *such that* $m(\{P : P \text{ is a relational variable}\})$ *is finite and the models verify the same formulas;*
2. *For every* $\mathsf{RL_{EQ}}$*-model* $\mathcal{M} = (U, m)$*, if* $m(\{P : P \text{ is a relational variable}\})$ *is finite, then there is an* $\mathsf{RL_{DEP}}$*-model such that the models verify the same formulas.*

Proof. The part 1. of the proposition is obvious in view of the corresponding definitions. To prove 2. we construct an $\mathsf{RL_{DEP}}$-model determined by the given $\mathsf{RL_{EQ}}$-model $\mathcal{M} = (U, m)$. We define a database relation R so that $AT_R = m(\{P : P \text{ is a relational variable}\})$, that is the attributes of R are the equivalence relations determined by the meanings of relational variables. It follows that for every relational variable P, $\{m(P)\}$ is an attribute; it will be denoted by a_P. For every $x \in U$ we define a tuple t_x as $t_x(a_P) = \|x\|_{m(P)}$ which assigns an equivalence class of x with respect to the relation $m(P)$ to the attribute a_P. Then we have $(t_x, t_y) \in ind_R(a_P)$ iff $(x, y) \in m(P)$. It is easy to check that the sets of formulas true in these models coincide. □

A dual tableau for the logic $\mathsf{RL_{EQ}}$ consists of the rules and axiomatic sets of $\mathsf{RL}(1, 1')$-dual tableau endowed with the rules which reflect interpretation of relational variables as equivalence relations:

For all object symbols x and y and for any relational variable P,

$$(\text{ref } P)\ \frac{xPy}{x1'y, xPy} \qquad\qquad (\text{sym } P)\ \frac{xPy}{yPx}$$

$$(\text{tran } P)\ \frac{xPy}{xPz, xPy \mid zPy, xPy} \qquad z \text{ is any object symbol}$$

The notions of an $\mathsf{RL_{EQ}}$-set and $\mathsf{RL_{EQ}}$-correctness of a rule are defined as in Sect. 2.4.

Theorem 6.6.1 (Correspondence). *Let* $\mathcal{K}$ *be a class of* $\mathsf{RL}(1, 1')$*-models. Then* $\mathcal{K}$ *is a class of* $\mathsf{RL_{EQ}}$*-models iff the rules (ref P), (sym P), and (tran P) are* $\mathcal{K}$*-correct.*

Proof. Let $\mathcal{K}$ be a class of $\mathsf{RL}(1, 1')$-models.

($\rightarrow$) Assume that $\mathcal{K}$ is the class of $\mathsf{RL_{EQ}}$-models. By way of example, we show correctness of the rule (tran P). Let X be any finite set of formulas. Clearly, if $X \cup \{xPy\}$ is an $\mathsf{RL_{EQ}}$-set, then so are $X \cup \{xPz, xPy\}$ and $X \cup \{zPy, xPy\}$.

Now, assume that $X \cup \{xPz, xPy\}$ and $X \cup \{zPy, xPy\}$ are $\mathsf{RL_{EQ}}$-sets, and suppose $X \cup \{xPy\}$ is not an $\mathsf{RL_{EQ}}$-set. Then, there exist an $\mathsf{RL_{EQ}}$-model $\mathcal{M} = (U, m)$ and a valuation v in $\mathcal{M}$ such that for every $\varphi \in X \cup \{xPy\}$, $\mathcal{M}, v \not\models \varphi$. Thus, $(v(x), v(y)) \notin m(P)$. By the assumption, $\mathcal{M}, v \models xPz$ and $\mathcal{M}, v \models zPy$. Hence, $(v(x), v(z)) \in m(P)$ and $(v(z), v(y)) \in m(P)$, and by transitivity of $m(P)$, $(v(x), v(y)) \in m(P)$, a contradiction. Therefore, the rule (tran P) is $\mathsf{RL_{EQ}}$-correct.

($\leftarrow$) Assume that for all relational variables P the rules (ref P), (sym P), and (tran P) are $\mathcal{K}$-correct. We show that for every relational variable P, and for every $\mathcal{K}$-model $\mathcal{M} = (U, m)$, $m(P)$ is an equivalence relation. By way of example, we show transitivity of the relation $m(P)$. Observe that due to correctness of rule (tran P), for every finite set X of relational formulas the following holds: $X \cup \{xPy\}$ is a $\mathcal{K}$-set iff $X \cup \{xPz, xPy\}$ and $X \cup \{zPy, xPy\}$ are $\mathcal{K}$-sets. Let $X \overset{\text{df}}{=} \{x{-}Pz, z{-}Py\}$. Since $\{xPz, xPy, x{-}Pz, z{-}Py\}$ and $\{zPy, xPy, x{-}Pz, z{-}Py\}$ are $\mathcal{K}$-sets, then so is the set $\{xPy,\ x{-}Pz,\ z{-}Py\}$. Thus, for every $\mathcal{K}$-model $\mathcal{M} = (U, m)$ and for every valuation v in $\mathcal{M}$, if $(v(x), v(z)) \in m(P)$ and $(v(z), v(y)) \in m(P)$, then $(v(x), v(y)) \in m(P)$. Hence, $m(P)$ is transitive in every $\mathcal{K}$-model. □

Proposition 6.6.2.

1. *The* $\mathsf{RL_{EQ}}$*-rules are* $\mathsf{RL_{EQ}}$*-correct;*
2. *The* $\mathsf{RL_{EQ}}$*-axiomatic sets are* $\mathsf{RL_{EQ}}$*-sets.*

Correctness of the rules (ref P), (sym P), and (tran P) follows from Theorem 6.6.1. Correctness of the remaining rules can be proved as in $\mathsf{RL}(1, 1')$-logic (see Sects. 2.5 and 2.7).

The notions of an $\mathsf{RL_{EQ}}$-proof tree, a closed branch of such a tree, a closed $\mathsf{RL_{EQ}}$-proof tree, and $\mathsf{RL_{EQ}}$-provability are defined as in Sect. 2.4.

The completion conditions that are specific for $\mathsf{RL_{EQ}}$-dual tableau are:

For all object symbols x and y and for any relational variable P,

Cpl(ref P) If $xPy \in b$, then $x1'y \in b$, obtained by an application of the rule (ref P);

Cpl(sym P) If $xPy \in b$, then $yPx \in b$, obtained by an application of the rule (sym P);

Cpl(tran P) If $xPy \in b$, then for every object symbol z, either $xPz \in b$ or $zPy \in b$, obtained by an application of the rule (tran P).

The notions of a complete $\mathsf{RL_{EQ}}$-proof tree and an open branch of an $\mathsf{RL_{EQ}}$-proof tree are defined as in RL-logic (see Sect. 2.5).

Although, the rule (sym P) does not preserve the formulas of the form xPy, for a relational variable P, we can show that the closed branch property holds. The proof is similar to the proof of Proposition 2.8.1.

The branch structure $\mathcal{M}^b = (U^b, m^b)$ determined by an open branch b of an $\mathsf{RL_{EQ}}$-proof tree is defined as in the completeness proof of RL-dual tableau, in particular $m^b(P) = \{(x, y) \in U^b \times U^b : xPy \notin b\}$, for every relational variable P (see Sect. 2.6, p. 44).

Proposition 6.6.3 (Branch Model Property). *For every open branch b of an* $\mathsf{RL_{EQ}}$*-proof tree, $\mathcal{M}^b$ is an* $\mathsf{RL_{EQ}}$*-model.*

Proof. Following the method of proving the branch model property in the completeness proof of $\mathsf{RL}(1,1')$-dual tableau (see Sects. 2.5 and 2.7), we show that $\mathcal{M}^b$ satisfies specific properties of $\mathsf{RL_{EQ}}$-models, namely, we need to prove that every relation $m^b(P)$ is an equivalence relation. For reflexivity, assume that $(x,y) \in m^b(1')$ and suppose that $(x,y) \notin m^b(P)$. Then $x1'y \notin b$ and $xPy \in b$. By the completion condition Cpl(ref P), $x1'y \in b$, a contradiction. For symmetry, assume that $(x,y) \in m^b(P)$ and suppose that $(y,x) \notin m^b(P)$. Then $xPy \notin b$ and $yPx \in b$. By the completion condition Cpl(sym R), $xPy \in b$, a contradiction. For transitivity, assume that $(x,y) \in m^b(P)$ and $(y,z) \in m^b(P)$, that is $xPy \notin b$ and $yPz \notin b$. Suppose that $(x,z) \notin m^b(P)$. Then $xPz \in b$, and by the completion condition Cpl(tran P), either $xPy \in b$ or $yPz \in b$, a contradiction. □

Now, the completeness of the $\mathsf{RL_{EQ}}$-dual tableau can be proved as in $\mathsf{RL}(1,1')$-logic.

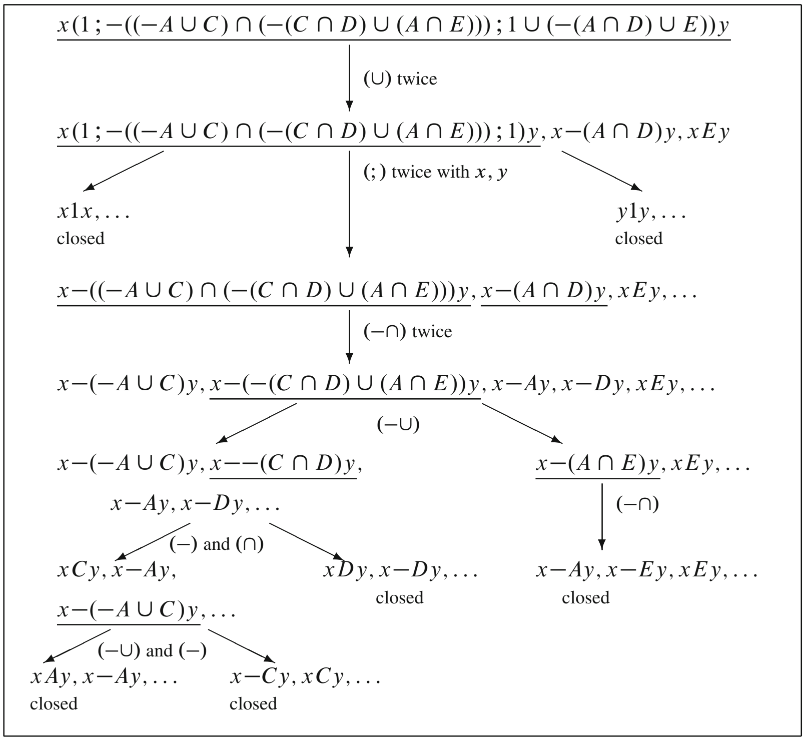
$x(1\,;-((-A\cup C)\cap(-(C\cap D)\cup(A\cap E)))\,;1\cup(-(A\cap D)\cup E))y$
$(\cup)$ twice
$x(1\,;-((-A\cup C)\cap(-(C\cap D)\cup(A\cap E)))\,;1)y, x-(A\cap D)y, xEy$
$(;)$ twice with x, y
$x1x, \ldots$ closed
$y1y, \ldots$ closed
$x-((-A\cup C)\cap(-(C\cap D)\cup(A\cap E)))y, x-(A\cap D)y, xEy, \ldots$
$(-\cap)$ twice
$x-(-A\cup C)y, x-(-(C\cap D)\cup(A\cap E))y, x-Ay, x-Dy, xEy, \ldots$
$(-\cup)$
$x-(-A\cup C)y, x--(C\cap D)y, x-Ay, x-Dy, \ldots$
$x-(A\cap E)y, xEy, \ldots$
$(-\cap)$
$(-)$ and $(\cap)$
$xCy, x-Ay, x-(-A\cup C)y, \ldots$
$xDy, x-Dy, \ldots$ closed
$x-Ay, x-Ey, xEy, \ldots$ closed
$(-\cup)$ and $(-)$
$xAy, x-Ay, \ldots$ closed
$x-Cy, xCy, \ldots$ closed

Fig. 6.3 An $\mathsf{RL_{EQ}}$-proof showing that $\{A \to C, CD \to AE\}$ implies $AD \to E$

Theorem 6.6.2 (Soundness and Completeness of $\mathsf{RL}_{\mathsf{EQ}}$). *For every $\mathsf{RL}_{\mathsf{EQ}}$-formula φ, the following conditions are equivalent:*

1. *φ is $\mathsf{RL}_{\mathsf{EQ}}$-valid;*
2. *φ is $\mathsf{RL}_{\mathsf{EQ}}$-provable.*

Theorems on entailment, model checking, and verification of satisfaction presented in Sects. 2.11, 3.4, and 3.5, respectively, apply to $\mathsf{RL}_{\mathsf{EQ}}$-logic.

Example. We show that the set $\{A \to C, CD \to AE\}$ of functional dependencies implies dependency $AD \to E$. By Propositions 6.5.1(2.) and 6.5.2, it suffices to show that $ind(A) \subseteq ind(C)$ and $ind(C) \cap ind(D) \subseteq ind(A) \cap ind(E)$ entail $ind(A) \cap ind(D) \subseteq ind(E)$. In what follows, for the sake of simplicity, we shall write Z instead of $ind(Z)$ for $Z = A, C, D, E$. We apply the method of verification of entailment presented in Sect. 2.11, thus we verify $\mathsf{RL}_{\mathsf{EQ}}$-validity of the following relational formula:

$$x[1\,;-[(-A \cup C) \cap (-(C \cap D) \cup (A \cap E))]\,;1 \cup (-(A \cap D) \cup E)]y.$$

Figure 6.3 presents its $\mathsf{RL}_{\mathsf{EQ}}$-proof.