Relational Databases vs. First-Order Logic

Based on

“Equality and Domain Closure in First-Order Databases”

and

“Towards a Logical Reconstruction of Relational Database Theory”

by Raymond Reiter
Part I

Introduction to Relational Databases
# Introduction to Relational Databases

## Two different approaches: Model and Proof Theoretic

<table>
<thead>
<tr>
<th>Model Theoretic</th>
<th>Proof Theoretic</th>
</tr>
</thead>
<tbody>
<tr>
<td>• The database is a model of some set of integrity constraints</td>
<td>• The database is a set of first-order formulæ</td>
</tr>
<tr>
<td>• A query is a formula to be evaluated with respect to the above model</td>
<td>• A query is a set of formulæ to be proven</td>
</tr>
<tr>
<td></td>
<td>• Integrity constraints are considered in terms of consistency</td>
</tr>
</tbody>
</table>
Introduction to Relational Databases
Model theoretic approach

- **Relational language** $R$
  \[ R = (\mathcal{U}, \mathcal{RS}), \text{where} \]
  \[ \text{alphabet } \mathcal{U}, \text{such that } \| \mathcal{U} \| < \aleph_0 \]
  \[ \text{arbitrary wwf 's } \mathcal{RS} \subseteq \mathcal{P}(\mathcal{U}) \]

- **Relational interpretation** $I$
  \[ I = (D, K, E), \text{where} \]
  \[ \text{domain } D \neq \{\} \]
  \[ \text{mapping of constants } K : \mathcal{U} \rightarrow D \]
  \[ \text{mapping of predicates } E : \mathcal{U} \rightarrow D^n, \text{such that } E(\approx) = \{(d, d) | d \in D\} \]
  \[ E(P) \text{ is called } \text{extension of } P \text{ in the interpretation } I \]
Introduction to Relational Databases
Model theoretic approach

- **Relational Database DB**

  \[ DB = (R, I, IC), \text{ where} \]

  \( IC \) is a set of wwf's called **integrity constraints**, \( IC = \{ \varphi | \varphi \equiv \forall x_1 \forall x_2 \ldots \forall x_n (P(x_1, x_2, \ldots, x_n) \rightarrow \tau_1(x_1) \wedge \tau_2(x_2) \wedge \ldots \wedge \tau_n(x_n)) \} \)

  for every predicate \( P \) of \( \mathcal{H} \), with the exception of \( \approx(\_, \_) \)

  \( \tau_i \) are types called **domains** of \( P \)

  \( E(P) \) are called **relations**

- No need for the concept of “safe” and “unsafe” queries, since \( \text{adom}(I) \)
  has only finitely many elements
Introduction to Relational Databases
Model theoretic approach

Example
A relational language $R=(A,W)$ where the only predicates and constants of $A$ are the following:

- Predicates: $TEACHER(_), COURSE(_), STUDENT(_), TEACH(_), ENROLLED(_), \approx(_,_)$
- Simple Types: $TEACHER(_), COURSE(_), STUDENT(_)$
- Constants: $A, B, C, a, b, c, d, CS100, CS200, P100, P200$

Then the following table defines an interpretation of $R$ with domain $D=${$A, B, C, a, b, c, d, CS100, CS200, P100, P200$}
Problems

- **Disjunctive information**
  - Can lead to indefinite answers to queries
  - Example: Student “a” is enrolled in class P200 or CS200
  
  Solution: Split initial interpretation into two distinct interpretations $I_1$ and $I_2$, where in $I_1$ the relation ENROLLED contains the tuple $(a,P200)$ and $I_2$ the tuple $(a,CS200)$

  Any case such as the above could not be solved by treating a relation like $ENROLLED(a,P200) \lor ENROLLED(a,CS200)$ as an integrity constraint, since that would require that at least one of the tuples be present in the relation, and we do not know which is the case.
Problems (cont.)

- **Null values**
  - Their need arises when we want a relation to reference a value that does not belong to our set of constants
  - Example: Student “a” may be enrolled in class P200 or CS200

Solution: Add a new entity \( \omega \) and allow tuples such as \((a, \omega)\). Now, \(\text{ENROLLED}(a,\omega)\) means that student a is either

- enrolled in one of the known classes (previous disjunctive case)
- Enrolled in some “unknown” class, one not in the domain D. Then, \( \omega \) takes the form of a **truth value** in a 3-valued truth system. This is the only case which properly “extends” our existing model in a way that equality and joins keep their intended meaning in relational algebra.
The proof theoretic approach stems from the previous model theoretic approach.

In order for every wff that is satisfiable in the m.t.a to be provable in the p.t.a., we need a **domain closure axiom**

\[ \forall x \left( \approx(x, A) \lor \approx(x, B) \lor \ldots \lor \approx(x, Z) \right), \text{where } D = \{ A, B, \ldots, Z \} \]

To compensate for negations of equality, which are satisfiable in the m.t.a, but unprovable in the p.t.a, we need **unique name axioms** (i.e. no allii), so that for each two constants c, c' of the domain

\[ \neg \approx(c, c') \]
Introduction to Relational Databases
Proof theoretic approach

- **Relational theory** $T \subseteq \mathcal{L}$, contains
  - domain closure axiom
  - unique name axioms
  - equality axioms
    - $\forall x (\approx(x,x))$
    - $\forall x (\approx(x,y) \rightarrow \approx(y,x))$
    - $\forall x \forall y \forall z (\approx(x,y) \wedge \approx(y,z) \rightarrow \approx(x,z))$
    - Leibnitz substitution of equal terms in wwf’s
  - a set $\Delta \subseteq T$, and for each predicate $P$ distinct for the equality predicate a set $C_p$ of n-tuples of constants
    \[ C_p = \{ \bar{c} \mid P(\bar{c}) \in \Delta \}. \text{The set } \{ P(\bar{c}) \mid \bar{c} \in \Delta \} \text{is called the extention of } P \in T \]
  - **Completion axioms** for each predicate $P$ of the form
    \[ \forall x_1 ... \forall x_n [P(x_1, ..., x_n) \rightarrow \approx(x_1, c_1^{(1)}) \wedge \approx(x_n, c_n^{(1)}) \vee ... \vee \approx(x_1, c_1^{(r)}) \wedge ... \wedge \approx(x_n, c_n^{(r)})] \]
    assuming that $C_p = \{ (c_1^{(1)}, ..., c_n^{(1)}), ..., (c_1^{(r)}, ..., c_n^{(r)}) \}$
Introduction to Relational Databases
Proof theoretic approach

- Completion axioms consist of a proof theoretic way of describing the contents of a relation (database table)
- Then, a **relational database** is defined as
  \[ \text{DB} = (R,T,IC) \]
- The above definitions suffice to prove that
  - If \( T \) is as relational theory then \( T \) has a unique model \( I \) which is a relational interpretation for \( R \)
  - If \( I \) is a relational interpretation of \( R \) then there is a relational theory \( T \) of \( R \) such that \( I \) is the only model of \( T \)

And the model / proof theoretic approach analogy is complete
Introduction to Relational Databases
Proof theoretic approach

Problems

• **Disjunctive information**
  
  - The original constraints of the m.t.a. are in inadequate due to the equality predicate
  
  - Example: Foo supplies a or Foo supplies b

Simply adding in IC the wwf

\[ SUPPLIES(Foo,a) \lor SUPPLIES(Foo,b) \]

in a theory that has the completion axiom

\[ \forall x \forall y [SUPPLIES(x,y) \rightarrow \approx(x,Acme) \land \approx(y,a) \lor \approx(x,Foo) \land \approx(y,b)] \]

whose contrapositive is

\[ \forall x \forall y [\neg \approx(x,Acme) \lor \neg \approx(y,a)] \land [\neg \approx(x,Foo) \lor \neg \approx(y,b)] \rightarrow \neg SUPPLIES(x,y) \]

leads to proving \[ \neg SUPPLIES(Foo,a) \] which is inconsistent
Problems (cont.)

- Need to “enrich” the completion axioms
- This leads to a **generalized relational theory** of T, by redefining the sets $C_p$
  $$C_p = \{ \hat{c} | \text{for some positive ground clause } A_1 \lor \ldots \lor A_r \text{ of } \Delta$$
  and some $i, 1 \leq i \leq r, A_i \text{ is } P(\hat{c}) \}$$
- A wwf of $\Psi$ is called a **positive ground clause** of R iff it has the form $A_1 \lor \ldots \lor A_r$ where each $A_i$ is a ground atomic formula whose predicate is distinct from the equality predicate.
Problems (cont.)

• **Null Values**
  
  - As in the m.t.a. we add a new entity $\omega$, which is now taken to be a **constant**. (Skolem constant)
  
  - As such, the domain closure axiom must be expanded to include $\omega$
  
  - The only thing that distinguishes the new constant from all the others is its absence in any unique name axiom (since it may represent one of those)
  
  - The resulting theory $T'$ is called a **generalized relational theory with null values**
Part II

Equality and Domain Closure in Relational Databases
Equality and Domain Closure in Relational Databases

- From the two different approaches in the first-order logic reconstruction of relational databases, the later (proof theoretic) has been shown to be far more fruitful.

- However, it is clear that the equality relation presents severe computational problems to any automated first-order theorem provers, due to the complexity that it introduces in the generalized relational theories.

- **Goal**: Devise a “natural” class of databases that can be described by a simpler relational theory.

- We assume a **function-free** first-order language to describe our databases.
Formal Preliminaries (Entities)

- Constants: \( c_1, c_2, \ldots \)
- Variables: \( x_1, x_2, \ldots \)
- Logical connectives: \( \land, \lor, \rightarrow, \neg \)
- Predicates: \( P, Q, R, ..., \approx \)
  - Subset of unary predicates called **simple types**
  - Remaining predicates called **common predicates**
- A set of **types**, where
  - A simple type is a type
  - *If \( \tau_1 \) and \( \tau_2 \) are types then so are \( \tau_1 \Theta \tau_2 \), where \( \Theta \in \{ \land, \lor, \rightarrow, \neg \} \)*
- Quantifiers: \( \forall, \exists \)
Formal Preliminaries (Syntax of formulæ)

- Terms \subseteq Variables \cup Constants
- Literals: \( P(t_1, \ldots, t_r) \), where \( P \) is a predicate and \( t_i \) are terms
  - Type literals if \( P \) is a simple type predicate
  - Common literals if \( P \) is a common predicate
- \( \tau \)-wwf's
  - A type literal is a \( \tau \)-wwf
  - If \( W_1 \) and \( W_2 \) are \( \tau \)-wwf's then so are \( W_1 \Theta W_2 \)
  - If \( W \) is a \( \tau \)-wwf then so are \( \forall x W, \exists x W \)
- Typed \( \text{wwf}'s \) (\( \text{twwf}'s \))
  - A common literal is a twwf
  - If \( W_1 \) and \( W_2 \) are twwf's then so are \( W_1 \Theta W_2 \)
  - If \( W \) is a \( \tau \)-wwf then so are \( \forall x/t W, \exists x/t W \)
Equality and Domain Closure in Relational Databases

Formal Preliminaries (Databases)

- A **database** is made up of two disjoint components
  - A **type database (TDB)**, where all information about types resides
  - A **principal database (PDB)**, which contains information about all the other predicates
- So DB=TDB ∪ PDB
- No wwf in DB contains any existential quantifier in its PNF
- PDB is a set of twwf's which contains the aforementioned equality axioms
- TDB is a **τ-complete** finite set of τ-wwf's (for each constant c and type τ, either T ⊢ τ(c) or T ⊢ ¬τ(c))
- TDB is formally a set of formulæ in the **monadic predicate calculus** and hence **decidable**
- If τ is a type, we define |τ|_{TDB} = { c | c is a constant and TDB ⊢ τ(c) }
Equality and Domain Closure in Relational Databases

Example

- TDB

\[
\begin{align*}
Axioms \\
\forall x \mathit{CALCULUS}(x) \lor \mathit{CS}(x) \lor \mathit{HISTORY}(x) \rightarrow \mathit{COURSE}(x) \\
\forall x \neg (\mathit{COURSE}(x) \land \mathit{TEACHER}(x)) \\
\forall x \neg (\mathit{TEACHER}(x) \land \mathit{STUDENT}(x))
\end{align*}
\]

Simple type instances

\[
\begin{align*}
|\mathit{CALCULUS}| &= \{C100, C200\} \\
|\mathit{CS}| &= \{CS100, CS200, CS300\} \\
|\mathit{HISTORY}| &= \{H100, H200\} \\
|\mathit{TEACHER}| &= \{A, B, C\} \\
|\mathit{STUDENT}| &= \{a, b, c, d\}
\end{align*}
\]
Equality and Domain Closure
in
Relational Databases

Example (cont.)

- PDB

\[(\forall x/\text{CALCULUS}) \text{TEACH}(A, x)\]
\[(\forall x/\text{CS}) \text{TEACH}(B, x)\]
\[(\forall x/\text{TEACHER})(\forall y/\text{COURSE})(\forall z/\text{STUDENT}) \text{TEACH}(x, y)\]
\[\land \text{ENROLLED}(z, y) \rightarrow \text{TEACHER-OF}(z, x)\]

<table>
<thead>
<tr>
<th>TEACH</th>
<th>ENROLLED</th>
</tr>
</thead>
<tbody>
<tr>
<td>C H100</td>
<td>a C100</td>
</tr>
<tr>
<td>C H200</td>
<td>a CS100</td>
</tr>
<tr>
<td></td>
<td>b C200</td>
</tr>
<tr>
<td></td>
<td>b CS200</td>
</tr>
<tr>
<td></td>
<td>b CS300</td>
</tr>
<tr>
<td></td>
<td>c C100</td>
</tr>
<tr>
<td></td>
<td>c H200</td>
</tr>
<tr>
<td></td>
<td>d H100</td>
</tr>
</tbody>
</table>
### Equality and Domain Closure in Relational Databases

#### Formal Preliminaries (Queries)

- A query is any expression of the form
  \[
  <x_1/\tau_1, \ldots, x_n/\tau_n| (\Theta_1 y_1/\theta_1) \cdots (\Theta_m y_m/\theta_m) W(x_1, \ldots, x_n, y_1, \ldots, y_m)>
  \]
  where \(\tau_i, \theta_i\) are types, \(W\) is a quantifier – free twwf whose only free variables are \(x_1, \ldots, x_n, y_1, \ldots, y_m\), and which contains only constants occurring \(\in DB\).

- Types only appear in the **index** of a query (the sequence of \(x_i\) variables)

#### Examples:

- Who are a's teachers?
  \[
  <x/TEACHER|TEACHER - OF (a, x)>
  \]

- Who teaches all history courses?
  \[
  <x/TEACHER|(\forall y/HISTORY) TEACHES(x, y)>
  \]

- Who teaches calculus but not history?
  \[
  <x/TEACHER|(\exists y/CALCULUS)(\forall z/HISTORY) TEACH(x, y) \land \neg TEACH(x, z)>
  \]
Equality and Domain Closure
in
Relational Databases

Formal Preliminaries (Semantics of Queries)

- **Answers** to queries
  - Defining an n-tuple as an answer to Q iff
    
    
    \[
    DB \vdash \tau_1(c_1) \land \ldots \land \tau_n(c_n) \land (\Theta y/\theta) W(c, y)
    \]
    
    poses problems in the case of disjunctive information
  - Correct:
    
    \[
    \text{a set of } n-\text{tuples } \{c^{(1)}, \ldots, c^{(m)}\} \text{ is an answer to } Q \iff
    DB \vdash \lor_{i \leq m} [\tau(c^i) \land (\Theta y/\theta) W(c^{(i)}, y)]
    \]
    
    where \( \tau(c) \) denotes \( \tau_1(c_1) \land \ldots \land \tau_n(c_n) \)

- We define \( \{c^{(1)}, \ldots, c^{(m)}\} \equiv c^{(1)} + \ldots + c^{(m)} \)
Equality and Domain Closure in Relational Databases

Formal Preliminaries (Semantics of Queries cont.)

- An answer to $Q$ is **minimal** iff
  
  \[
  \text{for no } i, 1 \leq i \leq m, \text{ is } c^{(1)} + \ldots + c^{(i-1)} + c^{(i+1)} + \ldots + c^{(m)} \text{ an answer to } Q
  \]

- An answer to $Q$ is **definite** if $m=1$

- An answer to $Q$ is **indefinite** if $m>1$

  An indefinite answer means that $x$ is either one of the tuples $c^{(i)}$, but it is impossible to say which

- The **value** of a query $||Q||_{DB}$ is the set of minimal answers to $Q$
Equality and Domain Closure
in
Relational Databases

Reduction of arbitrary queries to existential in closed databases

- If \( Q = <x/\tau,z/\psi| (\Theta y/\theta W)(x,y,z) \) we define the quotient of \( ||Q|| \) by \( \psi, \Delta_\psi ||Q|| \) as follows:

  \[
  c^{(1)} + \ldots + c^{(m)} \in \Delta_\psi ||Q|| \text{ iff }
  \]
  \[
  \text{for all } a_i \in |\psi|, 1 \leq i \leq m, (c^{(1)}, a_1) + \ldots + (c^{(m)}, a_m) \text{ is an answer of } Q
  \]
  \[
  \text{for no } i, 1 \leq i \leq m \text{ does } c^{(1)} + \ldots + c^{(i-1)} + c^{(i+1)} + \ldots + c^{(m)} \text{ have the later property}
  \]

- Then, \( ||<x/\tau| (\forall y/\theta) W(x,y)>|| = \Delta_\theta ||<x/\tau,y/\theta|W(x,y)>|| \)

- So we can strip of any leading universal quantifiers until we remain with a twwf that has only existential quantifiers (if any).
Equality and Domain Closure in Relational Databases

Reduction of arbitrary queries to existential in closed databases (cont.)

- If $Q = \langle x/\tau, z/\psi | (\Theta y/\theta W)(x,y,z) \rangle$ we define the projection of $\|Q\|$ with respect to $\psi$, $\pi_\psi \|Q\|$ as follows:

\[
\begin{aligned}
& c^{(1)} + \ldots + c^{(m)} \in \pi_\psi \|Q\| \text{ iff } \\
& \text{there exist constants } a_1, \ldots, a_r \in |\theta| \text{ such that } (c^{(i)} + \ldots + c^{(i-1)} + c^{(i+1)} + \ldots + c^{(m)}) \text{ is an answer to } Q \\
& \text{for no } i, 1 \leq i \leq m \text{ does } c^{(1)} + \ldots + c^{(i-1)} + c^{(i+1)} + \ldots + c^{(m)} \text{ have the later property}
\end{aligned}
\]

- Then, $\|\langle x/\tau | (\exists x/\theta) W(x,y) \rangle\| = \pi_\theta \|\langle x/\tau, y/\theta | W(x,y) \rangle\|$  

- So we can strip of any existential quantifiers preceding universal quantifiers
Reduction of arbitrary queries to existential in closed databases (cont.)

- Examples:
  
  Suppose $\| Q \| = \{(A, a) + (B, b), (A, b), (B, a), (C, a), (C, d), (D, a), (E, a) + (F, a), (E, b)\}$ and $\| \psi \| = \{a, b\}$. Then
  
  $\Delta_\psi \| Q \| = \{A + B, C\}$

  Suppose $Q = \langle x_1/\theta_1, x_2/\theta_2, z/\psi \mid W(x, y, z)\rangle$
  and that $\| Q \| = \{(a, a, A) + (a, a, B) + (a, d, C), (a, b, A) + (a, c, D), (a, b, C)\}$. Then
  
  $\pi_\psi \| Q \| = \{(a, a) + (a, d), (a, b)\}$
Equality and Domain Closure in Relational Databases

Reduction of arbitrary queries to existential in closed databases (cont.)

- For closed databases

\[ DB \vdash (\exists y/\theta)W(y) \text{ iff } DB \setminus DC \vdash (\exists y/\theta)W(y) \]

where \( DC \) is the domain closure axiom, and \( W \) is a quantifier-free twwf

- Therefore, we can dispose of the domain closure axiom.

- The previous theorems are based on the \( \tau \)-completeness of the TDB

- The aforementioned operators are appropriate generalizations of the division and projection operators of the relational algebra

- These operators allow us to compute a set of minimal answers to an arbitrary query by applying them on the minimal answers of an existential query

For example:

\[ ||<x/\tau|(\exists y/\theta)(\forall z/\psi)(\exists w/\rho)W(x,y,z,w)>|| = \]

\[ \pi_{\psi}\Delta_{\psi}||<x/\tau,y/\theta,z/\psi|(\exists w/\rho)W(x,y,z,w)>|| \]

- Their computation is straightforward in the case of definite answers
Special cases

- Let $W$ be a twwf in PNF such that $T=(\forall x/\tau)C(x)$, where $C$ is quantifier-free. Then $W$ is **Horn** iff $C(x)$ has at most one positive literal. A database is Horn if all of its twwf's are Horn.

- **Closed-world databases** are databases with appropriate rules so that the negation of a ground literal is derived upon failure to derive the said literal. Therefore, only positive information need be stored.

- In the case of Horn databases with positive queries, and in the case of closed-world databases, no indefinite answers arise.
Special cases (cont.)

- A database is \( \exists \)-saturated iff it has unique name axioms for all constants in its domain.
- In such a case, we can dispose of all equality axioms but the one of reflexivity.
- This is of great importance since these axioms pose significant computational difficulties.
Conclusion

- Certain types of databases, mainly closed databases and their special cases, exhibit a significant degree of simplification as far as computational feasibility is concerned.

- The limitation on the absence of functional symbols is not that restricting; one can insert functions in the database guised as relations.

- More generally, functions can be shown to pose no problem to the above findings, as long as they are total.

- Partial functions may lead to indefinite answers even if the information is in store.