SQL Nulls: Algebra and Logic

Enrico Franconi Sergio Tessaris

Free University of Bozen-Bolzano, Italy

http://www.inf.unibz.it/∼franconi

Venezia, 26 June 2012

SQL Nulls (Motivation)

- Peculiarity of SQL Nulls
  - 3-valued logics in the `WHERE` clause

R :

1	2
a	b
b	**N**

```
SELECT * FROM R
WHERE R.1 = R.1 AND R.2 = R.2 ;
```

$\Rightarrow$

```
1 | 2
--+--
a | b
(1 row)
```

- NOT the identity query
  - with Null values as constants
  - with Null values as existentials

SQL Nulls: Algebra and Logic (Summary)

- Characterising SQL with a Null Relational Algebra
  - Properly extends the classical Relational Algebra with Null values
    - The Null value is in the syntax
    - The Null value is part of the domain
- A Null Relational Calculus (first-order logic)
  - Properly extends the classical Relational Calculus with Null values
    - The Null value is in the syntax
    - The Null value is *not* part of the domain

- Theorem: the Null Relational Algebra is equally expressive as the domain independent Null Relational Calculus
  - Properly extends the Codd's theorem

Consequences (Summary)

- Theorem: the Null Relational Algebra/Calculus can be *linearly* embedded in the classical Relational Algebra/Calculus
  - Horizontal Decomposition
- SQL Nulls do not introduce "incompleteness" in databases
  - The semantic of SQL Nulls is "missing information"
- The Null Relational Calculus can express precisely the integrity constraints as defined in the SQL:1999 standard

Classical Relational Algebra ($\mathcal{RA}$)

Atomic relation - R

Constant singleton - $\langle v \rangle$

Selection - $\sigma_{i=v}\ e$ - $\sigma_{i=j}\ e$

Projection - $\pi_{i_1,\ldots,i_k} e$

Cartesian product - $e \times e'$

Union/Difference - $e \cup e'$, $e - e'$

Derived operators -

$\sigma_{i<>v}\ e$

$\sigma_{i<>j}\ e$

$e \underset{\substack{i_1=k_1 \\ \cdots \\ i_\ell=k_\ell}}{\bowtie} e'$

$e \cap e'$

Null Relational Algebra ($\mathcal{RA}^{\mathbf{N}}$)

Null singleton - $\langle \mathbf{N} \rangle$

Selection - $\sigma_{i=j}\ e = \{t \in e \mid\ t[i] = t[j] \wedge t[i] \neq \mathbf{N} \wedge t[j] \neq \mathbf{N}\}$

Derived operators -

$$\sigma_{i<>j}\ e \equiv \sigma_{i=i}\ \sigma_{j=j}\ e - \sigma_{i=j}\ e$$

$$\sigma_{\text{isNull}(i)}\ e \equiv e - \sigma_{i=i}\ e$$

$$\sigma_{\text{isNotNull}(i)}\ e \equiv \sigma_{i=i}\ e$$

$$\sigma_{i=\mathbf{N}}\ e \equiv e - e$$

$$\sigma_{i<>\mathbf{N}}\ e \equiv e - e$$

$$e \underset{\substack{i_1=k_1\\ \cdots\\ i_\ell=k_\ell}}{⟕} e' \equiv (e \underset{\substack{i_1=k_1\\ \cdots\\ i_\ell=k_\ell}}{\bowtie} e') \cup (e - \pi_{1,\dots,m}(e \underset{\substack{i_1=k_1\\ \cdots\\ i_\ell=k_\ell}}{\bowtie} e')) \times (\underbrace{\langle \mathbf{N} \rangle \times \cdots \times \langle \mathbf{N} \rangle}_{n-\ell})$$

Provably equivalent to SQL `SELECT-FROM-WHERE` with 3-valued logic and set operators. *(well...)*

Null Relational Calculus ($\mathcal{FOL}^{\varepsilon}$)

- The Null Relational Calculus extends the classical relational calculus in order to take into account the possibility that some of the arguments of a relation might not exist.
- It is a first-order logic language (*domain calculus*):
  - with an uninterpreted term **N** in the syntax representing the null value,
  - and where an n-ary predicate denotes a set of tuples over *subsets* of the arguments instead of just their whole set of arguments – i.e., a n-tuple is represented as a partial function from the arguments to the domain.
- If n-tuples were just *total* functions, then a n-ary predicate would denote a set of classical n-tuples (1NF).

Example

The legal database states of the formula $R(a, b) \wedge R(b, \mathbf{N})$ are such that the denotation of R includes the tuples $\{1 \mapsto a, 2 \mapsto b\}$ and $\{1 \mapsto b\}$.

Horizontal Decomposition (1NF)

Theorem

The Null Relational Calculus can be linearly embedded in the classical Relational Calculus, with a bijective translation preserving the legal database states of the formulae.

- *Horizontal Decomposition*

Example

The formula $\exists x. R(a, x) \wedge R(x, \mathbf{N}) \wedge R(\mathbf{N}, \mathbf{N})$ over the signature R is translated as the $\mathcal{FOL}$ formula $\exists x. \widetilde{R}_{\{1,2\}}(a, x) \wedge \widetilde{R}_{\{1\}}(x) \wedge \widetilde{R}_{\{\}}$ over the decomposed signature $\widetilde{\mathcal{R}}$, and vice-versa.

R :

1	2
a	b
b	**N**

$\Longrightarrow$

$\widetilde{R}_{\{1,2\}}$:

1	2
a	b

$\widetilde{R}_{\{1\}}$:

1
b

Codd's theorem with Null values

- There exists a polynomial reduction from the membership problem of a tuple in the answer of a $\mathcal{RA}^{\mathbf{N}}$ expression over a database instance with null values
- into the satisfiability problem of a closed safe-range $\mathcal{FOL}^{\varepsilon}$ formula over an homomorphic database ;

- and there exists a polynomial reduction from the satisfiability problem of a closed safe-range $\mathcal{FOL}^{\varepsilon}$ formula over a database instance with null values
- into the emptiness problem of the answer of a $\mathcal{RA}^{\mathbf{N}}$ expression over an homomorphic database.

Equivalence of Algebra and Calculus

Theorem

Let e be a $\mathcal{RA}^{\mathbf{N}}$ expression of arity n, and t a n-tuple as a total function with values taken from the set $\mathcal{C} \cup \{\mathbf{N}\}$. There is a function $\Omega(e, t)$ translating e with respect to t into a closed safe-range $\mathcal{FOL}^{\varepsilon}$ formula, such that for any instance $\mathcal{I}$ with the Standard Name Assumption:

$$t \in e(\mathcal{I}^{\mathbf{N}}) \quad \text{if and only if} \quad \mathcal{I}^{\varepsilon} \models_{\mathcal{FOL}^{\varepsilon}} \Omega(e, t).$$

Theorem

Let φ be a safe-range $\mathcal{FOL}^{\varepsilon}$ closed formula. There is a $\mathcal{RA}^{\mathbf{N}}$ expression e, such that for any instance $\mathcal{I}$ with the Standard Name Assumption:

$$\mathcal{I}^{\varepsilon} \models_{\mathcal{FOL}^{\varepsilon}} \varphi \quad \text{if and only if} \quad e(\mathcal{I}^{\mathbf{N}}) \neq \emptyset.$$

Our first motivating example

R :

1	2
a	b
b	**N**

```
SELECT * FROM R
WHERE R.1 = R.1 AND R.2 = R.2 ;
```

$\Rrightarrow$

```
1 | 2
--+--
a | b
(1 row)
```

Example

- Checking whether the tuple $\langle a, b\rangle$ or the tuple $\langle b, \mathbf{N}\rangle$ are in the answer of the $\mathcal{RA}^{\mathbf{N}}$ query $\sigma_{1=1}\ \sigma_{2=2}\ R$
- corresponds to check whether the tuple $\langle a, b\rangle$ or the tuple $\langle b, \mathbf{N}\rangle$ are in the answer of the $\mathcal{RA}$ query $\sigma_{1=1}\ \sigma_{2=2}\ \sigma_{\text{isNotNull}(1)}\ \sigma_{\text{isNotNull}(2)}\ R$;
- and it corresponds to check the satisfiability over the database instance of the $\mathcal{FOL}^{\varepsilon}$ closed safe-range formula $R(a, b)$ for the tuple $\langle a, b\rangle$, or of the formula **false** for the tuple $\langle b, \mathbf{N}\rangle$.

More Examples

R :

1	2
a	b
b	**N**

S :

1	2
a	a
a	**N**
N	a
N	**N**

- UNIQUE constraint for $R.1$: $\sigma_{1=3}\ \sigma_{2\neq 4}\ (R \times R) = \emptyset$;
- NOT-NULL constraint for $R.1$: $\sigma_{\text{isNull}(1)}\ R = \emptyset$;
- UNIQUE constraint for $S.1$: $\sigma_{1=3}\ \sigma_{2\neq 4}\ (S \times S) = \emptyset$,
- FOREIGN KEY constraint from $S.2$ to $R.1$: $\pi_2\sigma_{\text{isNotNull}(2)}\ S - \pi_1\sigma_{\text{isNotNull}(1)}\ R = \emptyset$.

- UNIQUE constraint for $R.1$
  - ($\mathcal{RA}^{\mathbf{N}}$): $\sigma_{1=3}\ \sigma_{2\neq 4}\ (R \times R) = \emptyset$.
  - ($\mathcal{FOL}^{\varepsilon}$): $\forall x, y, z.\ R(x, y) \land R(x, z) \rightarrow y = z$.
- FOREIGN KEY from $S.2$ to $R.1$:
  - ($\mathcal{FOL}^{\varepsilon}$): $\forall x, y.\ S(x, y) \rightarrow \exists z.\ R(y, z)$
  - ($\mathcal{RA}^{\mathbf{N}}$): $\pi_2\sigma_{\text{isNotNull}(2)}\ S - \pi_1\sigma_{\text{isNotNull}(1)}\ R = \emptyset$.

Interesting Facts

- SQL Nulls do not introduce incompleteness in databases
  - The semantic of SQL Nulls is "missing information"
- The Null Relational Calculus captures exactly the integrity constraints as defined in the SQL:1999 standard - without having the Null value in the domain