On no information nulls in relational databases

Hans-Joachim Klein

Universität Kiel

- The problem
- Some approaches
- POSS–semantics
- Query evaluation
- Application of results to SQL

... The usual operational distinction between the primary key and other candidate keys (if any) is that no tuple is allowed to have an undefined value for any of the primary key components, whereas any other components may have an undefined value. ...

E.F. Codd in
Further Normalization of the Relational Model, 1971

$\boxed{\text{SQL Standard}}$

null value (null): A special value, or mark, that is used to indicate the absence of any data value

E.F. Codd [1975, 1979]:	*value existent but at present unknown*
	$(\|\omega\| = 1)$ *'null value'*

- *duplicate elimination rule*

$$\mathbf{R}\ \begin{array}{|cc|}\hline A & B \\ \hline 1 & \omega \\ \hline \end{array} \cup \mathbf{S}\ \begin{array}{|cc|}\hline A & B \\ \hline 1 & \omega \\ \hline \end{array} = \mathbf{R} \cup \mathbf{S}\ \begin{array}{|cc|}\hline A & B \\ \hline 1 & \omega \\ \hline \end{array}$$

$\downarrow$ [J. Biskup, TODS 1983]
$\hookrightarrow |\omega| \geq 1$

- *three-valued logic* (Łukasiewicz/Kleene)

$\wedge$	T	F	U
T	T	F	U
F	F	F	F
U	U	F	U

$\vee$	T	F	U
T	T	T	T
F	T	F	U
U	T	U	U

$\neg$	T	F	U
	F	T	U

... Faced with these unpalatable options, the designers of SQL fell back to the principle 'trust the user'. Providing support for explicit nulls and three-valued logic in the database system gives users the tools to represent missing data with minimal cost and without requiring defensive coding in every application. At the same time, supporting the NOT NULL constraint at the column level allows users to avoid the anomalies associated with the null values where this is considered important. Since none of the options for representing missing information is flawless, and since users are paying the bills, it seems appropriate that users should be able to choose the appropriate approach they find least troublesome. ...

Don Chamberlin in
A Complete Guide to DB2 - Universal Database,
Morgan Kaufmann, 1998

null $\hat{=}$ unknown

ORDER

	Ordno	Date	Works_on
	1	03-nov	null
	3	03-nov	E2
●	4	04-nov	null

Query: 'Get the data of each order o such that the employee working on o is not working on further orders with the same date of entry as o.'

```
SELECT *
FROM ORDER x
WHERE Works_on <> ALL
                  (SELECT Works_on
                   FROM ORDER y
                   WHERE y.Date = x.Date
                     and y.Ordno <> x.Ordno)
```

SQL answer: ● (sure information)

```
SELECT *
FROM ORDER x
WHERE NOT EXISTS (SELECT *
                  FROM ORDER y
                  WHERE y.Date = x.Date
                    and y.Ordno <> x.Ordno
                    and y.Works_on = x.Works_on)
```

SQL-answer: ORDER (maybe information)

null	$\hat{=}$	not exists

ORDER

Ordno	Date	Works_on
1	03-nov	null
3	03-nov	E2
4	04-nov	null

Query: 'Get the data of each order o such that there is no employee working on o and on further orders with the same date of entry as o.'

```
SELECT *
FROM ORDER x
WHERE NOT EXISTS (SELECT *
                  FROM ORDER y
                  WHERE y.Date = x.Date
                    and y.Ordno <> x.Ordno
                    and y.Works_on = x.Works_on)
```

SQL answer: ORDER (sure information)

```
SELECT *
FROM ORDER x
WHERE Works_on <> ALL
                  (SELECT Works_on
                   FROM ORDER y
                   WHERE y.Date = x.Date
                     and y.Ordno <> x.Ordno)
```

SQL answer: {(4, 04-nov, null)} (incomplete w.r.t. sure information)

Our advice is to take care that you use the SQL definition of a word rather than the standard English definition.

J. Melton, A.R. Simon: *Understanding the New SQL: A Complete Guide.* Morgan Kaufmann, 1993

Query semantics (informal)

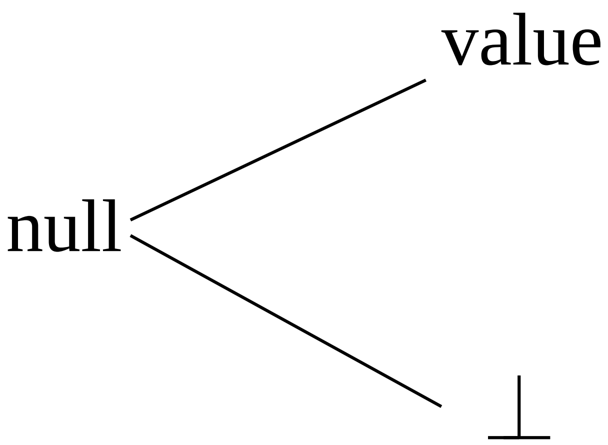
q query

$\|q\|_*$: the set of objects for which we can conclude, from the information available in the system, that they are in the external interpretation of q

$\|q\|^*$: the set of objects for which we cannot rule out the possibility of belonging to the external interpretation of q

external interpretation of q: answer to q in that (unknown) possible world corresponding to the 'reality'

[W. Lipski, 1979]



Sure information: Valid for q in every possible *world*

Possible information: Valid for q in at least one possible *world*

$\boxed{\textit{Maybe and maybe may be not maybe}}$

R

A	B
1	null

```
SELECT x.A
FROM   R x
WHERE NOT EXISTS         \
        (SELECT *         \
         FROM   R y       /  maybe \
         WHERE y.B = 1)  /          \
  AND NOT EXISTS         \           \   and   --->  not maybe
        (SELECT *         \          /
         FROM   R z       /  maybe  /
         WHERE z.B <> 1) /         /
```

Proposals not guaranteeing sure information in answers:

E.F. Codd [1975,1979,1986]	*duplicate elimination rule*
SQL	*switch 3VL/Boolean logic*
Gottlob/Zicari [1988]	*'unk', symbolic equality*
V.S. Lakshmanan [1989]	*intuitionistic logic, syntactical identity*
C. Zaniolo [1982, 1984]	*'no information'–interpretation, classes*
Roth/Korth/Silberschatz [1985]	
Levene [1991]	
....	

C. Zaniolo: Database Relations with Null Values. *JCSS* 28, 1984

Information ordering ($\geq$): t, t' tuples on α;

$$t' \geq t \Leftrightarrow_{def} (\forall A \in \alpha)(t(A)\ defined \quad \Rightarrow \quad t'(A) = t(A))$$

Examples: $(1, 2, 3) \geq (1, ni, 3), \quad (1, ni, 3) \geq (ni, ni, 3)$

$$S \geq R \Leftrightarrow_{def} (\forall t \in R)(\exists s \in S)(s \geq t)$$

Example:

S

A	B	C
1	2	3

$\geq$

R

A	B	C
1	2	ni
ni	2	3

$$S \equiv R \Leftrightarrow_{def} S \geq R \wedge R \geq S$$

Example:

S

A	B	C
1	2	3
ni	2	3

$\equiv$

R

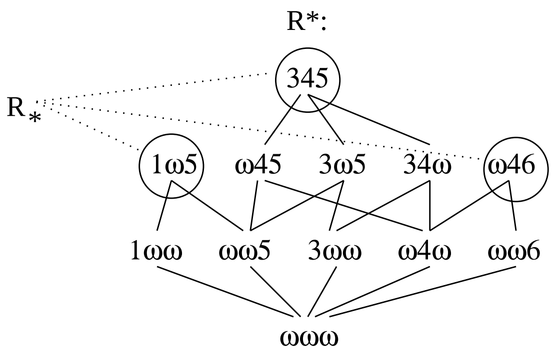
A	B	C
1	2	3
ni	ni	3

$\hat{R}$: equivalence class containing R

Maximal representative: $R^* = \{t \mid (\exists t' \in R)(t' \geq t)\}$

Minimal representative: $R_* = \{t \in R \mid \neg(\exists t' \in R)(t' \neq t \wedge t' \geq t)\}$

R: {(3,4,5), (1,ω,5), (ω,4,5), (3,ω,ω), (ω,4,6)}



ordinary
set theory [A.M. Keller, IPL 86]

UNION: $\hat{R}\tilde{\cup}\hat{S} =_{def} \hat{\{} r \mid r \in R \text{ or } r \in S \hat{\}}$

DIFFERENCE: $\hat{R}\tilde{-}\hat{S} =_{def} \hat{\{} t \mid t \in R \text{ and } (\forall s \in S)(\neg(s \geq t)) \hat{\}}$

SELECTION: $\hat{R}[A\theta k] =_{def} \hat{\{} r \mid r \in R \text{ is A-total } \wedge r[A]\theta k \hat{\}}$

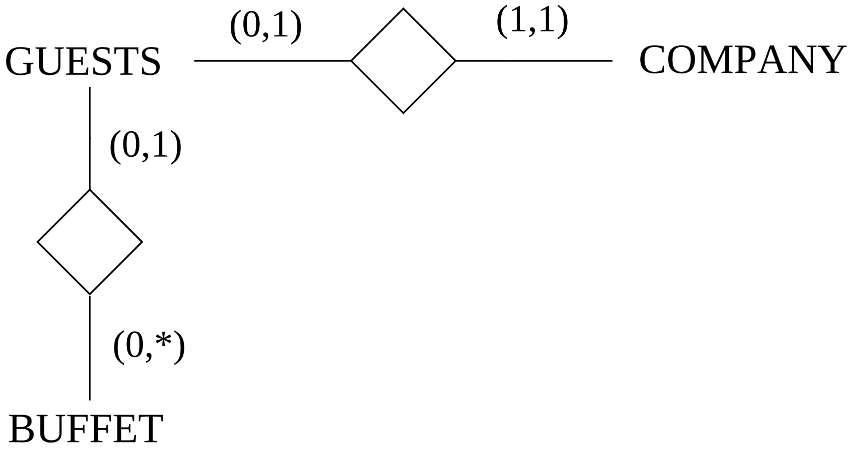
$$\text{R} = \{(a, b)\}, \qquad \text{S} = \{(a, ni)\}$$

$$S - ((R \cup S)[B \neq b])$$

$$\hat{\{}(a, ni)\hat{\}}\tilde{-}((\hat{\{}(a, b), (a, ni)\hat{\}})[B \neq b]) \quad =$$

$$\hat{\{}(a, ni)\hat{\}} \quad \tilde{-} \quad \emptyset \qquad = \quad \hat{\{}(a, ni)\hat{\}}$$

(not sure)



PARTY

Gname	Tel#	Cname	Contribution
Tim	346537	Anna	null
Gina	739012	null	cake
Petty	660890	null	null

GUESTS

Gname	Tel#
Tim	346537
Gina	739012
Petty	660890

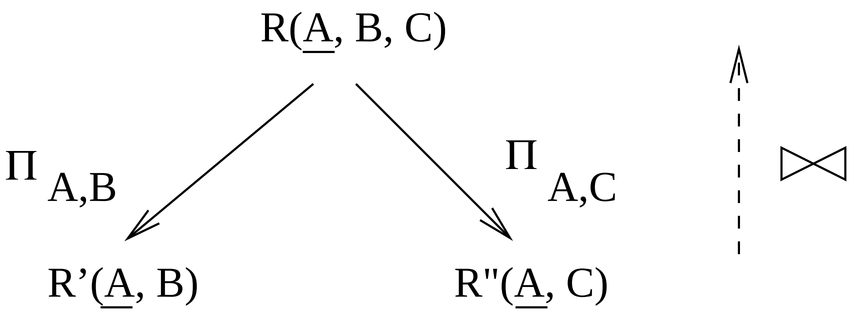
COMP_BUF

Gname	Cname	Contribution
Tim	Anna	null
Gina	null	cake
Petty	null	null

```
SELECT *
FROM GUESTS x
WHERE EXISTS (SELECT *
              FROM COMP_BUF y
              WHERE y.Gname = x.Gname)
```

```
SELECT *
FROM GUESTS x
WHERE EXISTS (SELECT *
              FROM COMP_BUF y
              WHERE y.Gname = x.Gname
                AND y.Contribution = 'cake' )
```

Decomposition



GUESTS_COMP

Gname	Tel#	Cname
Tim	346537	Anna
Gina	739012	null
Petty	660890	null
Frank	730286	null

GUESTS

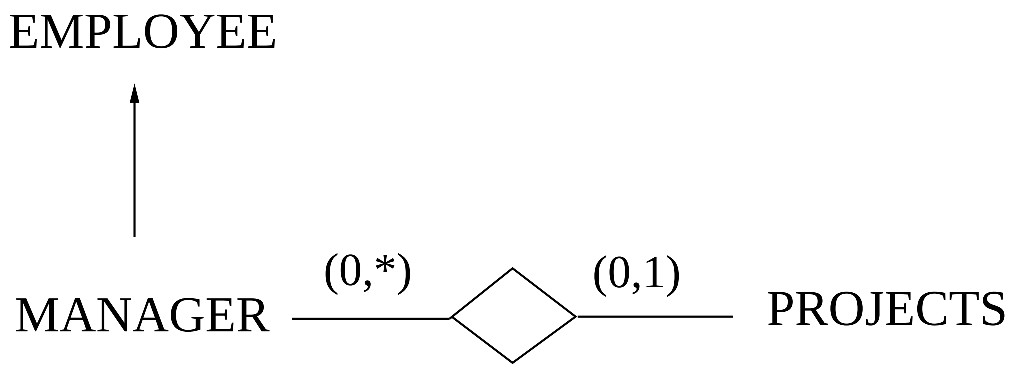
Gname	Tel#
Tim	346537
Gina	739012
Petty	660890
Frank	730286

COMPANY

Gname	Cname
Tim	Anna
Gina	null
Petty	null

null $\hat{=}$ not exists $\hookrightarrow$ problems with CWA!!

```
SELECT x.Gname
FROM GUESTS x
WHERE EXISTS (SELECT *
              FROM COMPANY y
              WHERE y.Gname = x.Gname)
```



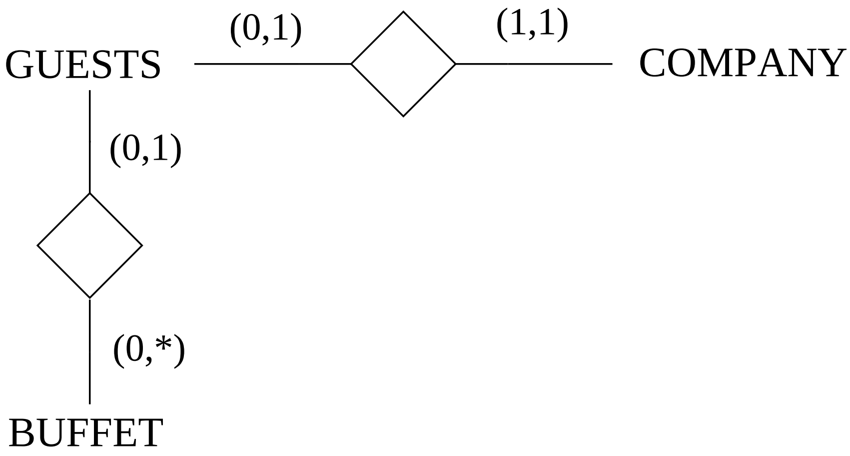
EMPLOYEES

Name	Tel#
Joe	448
Ben	375
Jane	897
Peter	993

PROJMAN

Name	Proj#
Joe	P2
Ben	P3
Jane	null

```
SELECT *
FROM EMPLOYEES x
WHERE EXISTS (SELECT *
              FROM PROJMAN y
              WHERE y.Name = x.Name)
```



PARTY

Gname	Tel#	Cname	Contribution
Tim	346537	Anna	null
Gina	739012	null	cake
Petty	660890	null	null

GUESTS

Gname	Tel#
Tim	346537
Gina	739012
Petty	660890

COMP_BUF

Gname	Cname	Contribution
Tim	Anna	null
Gina	null	cake
Petty	null	null

```
SELECT *
FROM GUESTS x
WHERE EXISTS (SELECT *
              FROM COMP_BUF y
              WHERE y.Gname = x.Gname)
```

```
SELECT *
FROM GUESTS x
WHERE EXISTS (SELECT *
              FROM COMP_BUF y
              WHERE y.Gname = x.Gname
                AND y.Contribution <> 'cake' )    (!!)
```

Database semantics

(cf. E. Sciore [1980], D. Maier [1983])

Concept: set of attributes of a relation type with attributes classified either as *mandatory* or as *optional* (with respect to the concept).

Examples:

Concepts of PARTY(Gname, Tel#, Cname, Contribution):

{(Gname,m), (Tel#,o)}
{(Gname,m), (Cname,m)}
{(Gname,m), (Contribution,m)}

Concepts of COMP_BUFFET(Gname, Cname, Contribution):

{(Gname,m), (Cname,m)}
{(Gname,m), (Contribution,m)}

Concepts of PROJMAN(Name, Proj#):

{(Name,m)}
{(Name,m), (Proj#,m)}
{(Proj#,m)}

Definition: Let RT be a relation type on attribute set α with set of concepts $\mathcal{C}$. An ω-tuple t on α is allowed wrt $\mathcal{C}$ $\Leftrightarrow_{def}$

- t is not totally undefined.
- If t is defined in an attribute A then there is at least one concept in $\mathcal{C}$ containing A such that t is defined in all mandatory attributes of this concept.

An ω-relation R of type RT is allowed wrt $\mathcal{C}$ $\Leftrightarrow_{def}$ all tuples of R are allowed wrt $\mathcal{C}$.

Definition: Let R be an ω-relation of type RT with set of concepts $\mathcal{C}$. An ω-relation R is called possible relation to R wrt $\mathcal{C}$ $\Leftrightarrow_{def}$ R' can be obtained from an extension R'' of R by discarding all tuples which are not allowed.

$\hookrightarrow$ possible states ($POSS$)

Examples:

COMP_BUF

Gname	Cname	Contribution
Tim	Anna	null
Gina	null	cake
Petty	null	null

{(Gname,m), (Cname,m)}
{(Gname,m), (Contribution,m)}

Gname	Cname	Contribution
Tim	Anna	null
Gina	null	cake

PROJMAN

Name	Proj#
Joe	P2
Ben	P3
Jane	null

{(Name,m)}
{(Name,m), (Proj#,m)}
{(Proj#,m)}

N. Lerat, W. Lipski: Nonapplicable nulls. *TCS* 46, 1986

- domain relational calculus
- universal relations (allowed relations without optional attributes)
- 2 kinds of variables (X,Y,Z,... x,y,z,...)
- *undefined* added to domains ($\mathit{undefined} = \mathrm{c} \hookrightarrow \mathit{false}$, $\mathit{undefined} = \mathit{undefined} \hookrightarrow \mathit{true}$)

GUEST

Gname	Tel#
Tim	346537
Gina	739012
Frank	660890

COMPANY

Gname	Cname
Tim	null
Gina	null

GUEST

Gname	Tel#
Tim	346537
Gisela	739012
Frank	660890

COMPANY

Gname	Cname
Tim	Anna
Gina	Nino

1

COMPANY

Gname	Cname
Tim	Anna

2

COMPANY

Gname	Cname

3

1: $\{(\text{Tim}, 346537), (\text{Gina}, 739012)\}$

2: $\emptyset$

3: $\{(\text{Tim}, 346537), (\text{Gina}, 739012), (\text{Frank}, 660890)\}$

GUEST

Gname	Tel#
Tim	346537
Gina	739012
Frank	660890

COMPANY

Gname	Cname
Tim	null
Gina	null

```
           /\
          /
         /
   ____/\____
  /          \
```

$(x)/GUEST\ x \wedge\ (\neg(\exists y : COMPANY\ y)\ \vee$

$\emptyset$

$(\exists u : COMPANY\ u)(u.Gname = x.Gname \wedge$

$\emptyset$

$(\exists z : COMPANY\ z)(z.Gname \neq u.Gname)))$

Relational Algebra

R

A	B
1	ω

S

A	B
1	3

T

A	B
1	4
1	3

$$T \backslash (R \cap S)$$

2 versions of operations:

- *sure*
- *maybe**

$$T \backslash_s (R \cap_m S)$$

$\hookrightarrow$ **Switch-Algebra**

Theorem: 3VL-interpretation $\equiv$ Switch-Algebra

$$\boxed{v \theta ALL(S)}$$

sure result $\Longrightarrow$ maybe result for S

Critical cases:

1) $v \equiv null$ $(null \quad \theta \quad ALL(S))$

SQL: *true* if $S \rightarrow \emptyset$
unknown if $S \rightarrow \neq \emptyset$

$\hookrightarrow$ *false* instead of *unknown* (because of possible negation)

Add AND (A IS NOT NULL OR NOT EXISTS(S))
for every attribute A in v

2) *null* occurs in the result of the evaluation of S (e.g.: $5 \leq ALL\{7, 8, null\}$)

result may be *unknown* $\Longrightarrow$ *false* instead of *unknown*

Let $S =$

```
SELECT w
FROM ....
WHERE cond
```

Add AND NOT EXISTS (S') where

S' =

```
SELECT w
FROM ....
WHERE cond AND (B IS NULL OR ...)
```

for every attribute B in w

null	$\hat{=}$	unknown

ORDER

Ordno	Date	Works_on
1	03-nov	null
3	03-nov	E2
4	04-nov	null

Query: 'Get the data of each order o such that the employee working on o is not working on further orders with the same date of entry as o.'

```
SELECT *
FROM ORDER x
WHERE x.Date <> ALL
                (SELECT y.Date
                 FROM ORDER y
                 WHERE y.Ordno <> x.Ordno
                   AND y.Works_on = x.Works_on)
```

```
SELECT *
FROM ORDER x
WHERE x.Date <> ALL
                (SELECT y.Date
                 FROM ORDER y
                 WHERE y.Ordno <> x.Ordno
                   AND (y.Works_on = x.Works_on OR
                     y.Works_on IS NULL OR x.Works_on IS NULL))
  AND x.Date IS NOT NULL                                 \
  AND NOT EXISTS (SELECT z.Date                           \
                  FROM ORDER z                             \
                  WHERE z.Ordno <> x.Ordno                 / redundant
                    AND z.Date IS NULL)                   /
                                                         /
```

EMPLOYEE

Empno	Dep	Sal	Year
1	d2	100k	93
2	d1	110k	null
3	d2	100k	94

```
SELECT x.Dep
FROM EMPLOYEE x
WHERE EXISTS (SELECT *
              FROM EMPLOYEE y
              WHERE y.Sal > x.Sal AND
                    y.Year <> x.year)
```

SQL-answer: $\emptyset$

Every possible answer: $(d2)$

```
SELECT x.Dep, x.Year
....
```

Possible answers:		$(d2, 93)$	$null \hookleftarrow 94$
		$(d2, 94)$	$null \hookleftarrow 93$
	or	$(d2, 93), (d2, 94)$	$null \hookleftarrow \neq 93, 94$

Approximations (*unknown*)

Query semantics for incomplete databases

(POSS) $\hookrightarrow$ Query semantics for complete databases

Representation problem	$\hookrightarrow$	Approximations(1) Different kinds of answers
Computability/Efficiency	$\hookrightarrow$	Approximations(2)

Query evaluation:

Relational Calculus		*Relational Algebra*
ω–interpretation(3VL)	$\hat{=}$	Switch–Algebra (maybe/sure versions of operations)
$\omega_{\{\}}$–interpretation(5VL)	$\hat{=}$	Switch–Algebra (maybe/sure versions of extended operations)

[H.-J. Klein, KRDB99]

Model-theoretic view		*Proof-theoretic view*
3VL–interpretation		Efficient algorithms [R.Reiter,86],[M.Vardi,86]
5VL-interpretation	$\hookrightarrow$	?

[H.-J. Klein, FoMLaDo98]

The meaning of a question addressed to a query system is not to be identified with how the system processes the query (and is not to be identified with a program at any level), but rather is to be identified with the range of answers that the question permits. That is, for a query system and a user to agree on the meaning of a question is for there to be agreement as to what counts as an answer to the question, regardless of how, or if, any answer is produced.

N.D. Belnap, T.B. Steel: *The Logic of Questions and Answers.*
Yale University Press, 1976