PRACTICAL ISSUES IN DATABASE MANAGEMENT

FOR THE THINKING PRACTITIONER

Fabian Pascal
DATABASE DEBUNKINGS
www.dbdebunk.com
DATABASE

- A database is a set of axioms.
- The response to a query is a theorem.
- The process of deriving the theorem from the axioms is a proof, which
  - is made by manipulating symbols according to agreed mathematical rules.
  - can only be as sound and consistent as the rules are.

  — H. Darwen

ASSERTIONS OF FACT

Employee has employee number 100, is named Spencer, works in department E21, was hired on 06-19-1980, earns a salary of $26,150

Employee has employee number 110, is named Lucchessi, works in department A00, was hired on 05-16-1958, earns a salary of $38,170

Employee with employee number (EMP#), named (ENAME), works in department (DEPT#), was hired on (HIREDATE), earns a salary of (SALARY)

<table>
<thead>
<tr>
<th>EMP#</th>
<th>ENAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spencer</td>
<td>E21</td>
<td>06-19-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchessi</td>
<td>A00</td>
<td>05-16-1958</td>
<td>38170</td>
</tr>
</tbody>
</table>
A DBMS is a deductive logic system
  Derives new facts from database facts
  The derived facts are facts if and only if:
    database assertions are true (facts)
    derivation rules are sound and consistent

DATA MODEL = SEMANTICS

- Reality --> Logical Models
  - Types
  - Organization (structure)
  - Integrity
  - Manipulation

- General

- Formal (scientific)
  - As simple as possible (but not simpler!)
THE RELATIONAL DATA MODEL

- Complete
  - Domains
  - R-Tables
  - Integrity Constraints
  - Manipulation

- Formal
  - Predicate Logic
  - Set Theory
  - Dependency Theory

DATA TYPES AND COMPLEXITY

CAREFUL WHAT YOU WISH FOR

CHAPTER 1

pp. 1-24
"The company was using a [SQL] RDBMS ... to handle data transactions for its trading applications. However, the applications required arbitrary data types, which is nearly impossible for relational systems, according to experts."

— TRADE MAGAZINE ARTICLE

"Object-oriented DBMSs ... support the storage and processing of any type of data, such as text, graphics, diagrams, video, audio, and user-defined data."

— VENDOR PRESS RELEASE
THE REAL WORLD (cont'd)

"There is an increasing need of enterprises for complex and function-related data ... this is [t]he most serious challenge so far ... to the relational database management system (RDBMS) model, [which] understands only simple types of data ... [the] solution ... is arriving in the form of Universal Server[s] ... RDBMS[s extended with] support for complex data types."

TRADE MAGAZINE ARTICLE

DATATYPES

- THE ISSUE
- FUNDAMENTALS
- PRACTICAL IMPLICATIONS
- CONCLUSIONS AND RECOMMENDATIONS
THE ISSUE

The data type concept is one of the least understood by database practitioners. This is both a cause and a consequence of the failure by SQL and its commercial dialects to implement relational domains, which are nothing but data types of arbitrary complexity.

Consequently, blame is being misplaced on the relational approach for the SQL products’ lack of support for so-called "complex" types, which permits proponents of the object approach to claim with impunity that object DBMSs are superior in this respect to relational DBMSs.

FUNDAMENTALS

1. "Simple" Types
2. System-defined Types
3. User-defined Types
4. DBMS Type support
5. Type "Atomicity"
6. "Complex" Types
Assertions of Fact

Employee has employee number 100, is named Spenser, works in department E21, was hired on 6/19/1980, earns a salary of $26,150

Employee has employee number 110, is named Lucchesi, works in department A00, was hired on 5/16/1956, earns a salary of $38,170

Employee with employee number (EMP#), named (ENAME), works in department (DEPT#), was hired on (HIREDATE), earns a salary of (SALARY)

<table>
<thead>
<tr>
<th>EMP#</th>
<th>ENAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spenser</td>
<td>E21</td>
<td>06-19-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchesi</td>
<td>A00</td>
<td>05-16-1956</td>
<td>38170</td>
</tr>
</tbody>
</table>

Type Constituents

- Name
- Possible Representation(s)
  - Actual
  - Declared
- Type constraints (optional)
- Operators (applicable to type's values)
**Type: TEMPERATURE**

- Possible:
  - degrees Fahrenheit
  - degrees Celsius
  - hot/warm/cool

- Declared: degrees Fahrenheit
- Actual: 4 bytes
- Operators: set applicable to degrees Fahrenheit

---

**"SIMPLE" TYPES**

- Numbers
- Strings
SYSTEM-DEFINED TYPES

<table>
<thead>
<tr>
<th>INTEGER</th>
<th>CHARACTER(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declared: INTEGER</td>
<td></td>
</tr>
<tr>
<td>(Actual: 2 bytes)</td>
<td></td>
</tr>
<tr>
<td>Type constraint: &gt;=-231 AND &lt;=231</td>
<td></td>
</tr>
<tr>
<td>Operators: set applicable to integers</td>
<td></td>
</tr>
<tr>
<td>Declared: CHAR(n)</td>
<td></td>
</tr>
<tr>
<td>(Actual: n bytes)</td>
<td></td>
</tr>
<tr>
<td>Type constraint: n&gt;=0 AND n&lt;=255</td>
<td></td>
</tr>
<tr>
<td>Operators: set applicable to character strings</td>
<td></td>
</tr>
</tbody>
</table>

USER-DEFINED TYPES

<table>
<thead>
<tr>
<th>EMP#</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declared: CHAR(3)</td>
<td></td>
</tr>
<tr>
<td>Type constraint: [A-Z][0-9][0-9]</td>
<td></td>
</tr>
<tr>
<td>Operators: set applicable to employee numbers ( selector, =, ..., but not &lt;, &gt;)</td>
<td></td>
</tr>
<tr>
<td>Declared: INTEGER</td>
<td></td>
</tr>
<tr>
<td>Type constraint: =&gt;15000 AND &lt;=100000</td>
<td></td>
</tr>
<tr>
<td>Operators: set applicable to salary values ( selector, =, +, &lt;, &gt;, ..., but not SQROOT)</td>
<td></td>
</tr>
</tbody>
</table>
RESULT TYPES

Project (PROJ#) has staff size (STAFF)

| STAFF |

Declared: INTEGER
Type constraint: >= 0 AND <= 450
Operators: set applicable to STAFF values (selector, =, +, /, ...)

| RESULTS |

Monadic
Dyadic (STAFF_RATIO: DECIMAL(4,2))
Triadic
N-adic

TRUTH-VALUED TYPE

TRUTH-VALUED

Declared: CHAR(1)
(Actual: SMALLINT)
Type constraint: T/F
Operators: set applicable to T/F values
DATE

Declared: CHAR(10)
Type constraint: [Jan, Feb, ..., Dec] [1-31] [0-...]
Operators: set applicable to date values (selector, =, <, >, ..., but not *, /)

SELECT *
FROM employees
WHERE hiredate = DATE('Nov-21-1973');

SELECT *
FROM employees
WHERE hiredate = DATE('Nov', 21, 1973)
COMPLEX REPRESENTATIONS

"Values of any type are designatable, but such designations might be long-winded when it comes to pictures, videos, sound recordings, engineering drawings, and the like ... e.g. two-dimensional arrays of pixels, where each pixel is represented by three numbers designating amounts of red, green, and blue."

— H. Darwen

---

COMPLEX OPERATORS

"Different instances of ["complex"] types require widely different types of processing [operators], and about the only thing these types have in common is that they are hard to deal with in today's DBMS products."

— C. J. Date
"COMPLEX" TYPES

- User-defined
  - Representations
  - Operators
- Encapsulation
  - Programmed
  - DBMS user-extendible

PRACTICAL IMPLICATIONS

1. R. Domains vs. Object Classes
2. Database Design
   - Relational structure vs. Object Manipulation
3. DBMS Design
   - SQL "Domains"
   - "Universal" DBMS
RELATIONAL REPRESENTATION

Employee with employee number (EMP#) has name (ENAME), works in department (DEPT#), was hired on (HIREDATE), earns a salary of (SALARY), has fingerprint (EFP)

<table>
<thead>
<tr>
<th>EMP#</th>
<th>ENAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
<th>EFP</th>
</tr>
</thead>
<tbody>
<tr>
<td>180</td>
<td>Spencer</td>
<td>721</td>
<td>06 19 1988</td>
<td>26150</td>
<td></td>
</tr>
<tr>
<td>260</td>
<td>Plaza</td>
<td>831</td>
<td>34 11 1977</td>
<td>27250</td>
<td></td>
</tr>
</tbody>
</table>

SELECT emp#, ename
FROM employees
WHERE efp = [EFPx]

OBJECT REPRESENTATION?

"It is odd that so many [object proponents] tend to use employees, departments, and so forth as examples of object classes. An object class is a type, of course, and so those [proponents] are forced to define a "collection" for those employees. What is more, those "collections" typically omit the all-important attribute names, so they are not relational tables. As a consequence, they do not lend themselves very well to the formulation of ad hoc queries, declarative integrity constraints, and so forth -- a fact that advocates of the approach themselves often admit, apparently without being aware that it is precisely the lack of attribute names that causes the problems."

- C.J. Date
DESIGN CHOICES

- Employee Type
  - Component representation
  - One-column table

- EMPLOYEES Table
  - no components
  - six-column table (one "complex")

THE FILM RIGHTS COMPANY

- Movie rights sales
  - Contracts(transactions)
  - "What rights, for what films, in what regions, have been sold to what customers, for what price?"
NO FREE LUNCH

- Tackling Complexity
- Structure and Integrity (RM)
- Manipulation (OO)

THE PLIGHT OF NASA

"... struggling with ... how to capture and analyze the [terabytes] of data beamed down to earth daily from orbiting satellites ... [a] problem is the way in which the raw data must be assigned to tables in order to be processed. This process inherently requires a degree of rationalization and some predisposition toward the ultimate use of the data. This is difficult because the scientist may not know ahead of time what analysis to run on the data. This lack of knowledge severely limits the usefulness of the system."

— TRADE MAGAZINE ARTICLE
FORMAL DATA MODEL

- Atomicity
- Selectivity
- Addressability

THINGS TO REFER TO

"The most appropriate design choices will emerge if careful consideration is given to the distinction between (a) declarative sentences in human language and (b) the vocabulary used in the construction of such sentences ... it is [rows in relational tables] that stand for those sentences, and it is domain values [in columns] that stand for particular elements typically nouns in that vocabulary. To say it slightly differently: Domains (types) give us values that represent things we might wish to refer to, [relational tables] give us ways of referring to those things in utterances about them."

- C.J. Date
DBMS DESIGN

1. SQL "Domains"
2. "Universal" DBMS

SQL SYSTEM-DEFINED "DOMAINS"

- Numbers
- Character Strings
- Bit Strings
- Dates
- Times
- Timestamps
- Year-Month Intervals
- Day-Time Intervals
SQL "DOMAINS"

CREATE DOMAIN emp# AS CHAR(3);

CREATE TABLE employees
(emp#, emp#,
ename, ename, ...);

SQL DBMS "COMPLEX DOMAINS"

- IMAGE
- TEXT
- BINARY
- VARBINARY
- (XML !)

-39-

-40-
Domain Impersonation

- Representation-based comparability
- No user-defined domains
- No operators (system- or user-defined) for system-defined "complex" types
- No type constraints
- No truth-valued domain

Universal DBMS

- User-defined DBMS Type Extensions
  - Problem resolution?
  - Integrity, data independence violations?
  - Catalog integration?
  - Performance optimization?
  - Upgrades, maintenance, support?
CONCLUSIONS

<table>
<thead>
<tr>
<th>Representations</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>* At least one stored</td>
<td>* Multiple possible representations</td>
</tr>
<tr>
<td>* Of arbitrary complexity</td>
<td>* Not necessarily stored</td>
</tr>
<tr>
<td>* Can have components</td>
<td>* Operator set (encapsulated)</td>
</tr>
<tr>
<td></td>
<td>* &quot;Atomic&quot; by definition</td>
</tr>
</tbody>
</table>

RECOMMENDATIONS

- No panacea
  - "Universal" misleading
  - ODBMS = DBMS "building-kit"
- No casual deployment to avoid structuring
  - Structure/integrity vs. manipulation
  - Dependence on programming
  - DBMS extendability
"The company was using a [SQL] RDBMS ... to handle data transactions for its trading applications. However, the applications required arbitrary data types, which is nearly impossible for relational systems, according to experts."

- TRADE MAGAZINE ARTICLE

"Object-oriented DBMSs ... support the storage and processing of any type of data, such as text, graphics, diagrams, video, audio, and user-defined data."

- VENDOR PRESS RELEASE
REAL WORLD REVISITED (cont'd)

"There is an increasing need of enterprises for complex and function-related data ... this is [t]he most serious challenge so far ... to the relational database management system (RDBMS) model, [which] understands only simple types of data ... [the] solution ... is arriving in the form of Universal Server[s] ... RDBMS[s extended with] support for complex data types."

— TRADE MAGAZINE ARTICLE

MISSING INFORMATION

WHAT YOU DON'T KNOW CAN HURT YOU

CHAPTER 10

pp. 225-47
"The SQL Language Reference" manual contradicts itself right away with the definition of NULL. NULL is not equivalent to zero or a blank. Empty strings have a null value. In subsequent sections "Null and search conditions" the manual doesn't distinguish NULL and zero-length strings. It goes on to explain that

SELECT ... WHERE x IS NULL

returns null rows as well as empty strings. The problem is, the converse is not true. Such a statement is guaranteed to return nothing

SELECT ... WHERE x = "

This is an nasty exception! Some further confusion in the manual can be found in the following: If any item in an expression contains a null value, then the result of evaluating the expression is null. However, this doesn't seem to apply to NULL string. These expressions simply treat NULL as empty strings.

@length(x) || ||""

Furthermore, x IS NULL never results in NULL. It seems the vendor has major confusion on NULL."
THE ISSUE

As attested to by the volume of writings and the heat of the debate on the subject without an end in sight, how to treat missing information has possibly been the thorniest aspect of database management.

Users are left between a rock and a hard place: They can either rely on SQL’s badly flawed version of three-valued logic based on NULLs and risk hard to interpret database answers and/or hard to detect errors in query results, or take upon themselves the burden of what is a complex database function, which belongs in the DBMS.

FUNDAMENTALS

1. Meaningless Assertions
2. Empty Assertions
3. Many-Valued Logic
4. Missing Information as Metadata
5. DBMS Support
Employee identified by employee number (EMP#), named (NAME) works in department (DEPT#), was hired on (HIREDATE), earns a salary of (SALARY).

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spenser</td>
<td>E21</td>
<td>06-19-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchesi</td>
<td>A00</td>
<td>05-16-1988</td>
<td>38170</td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>12-05-1963</td>
<td>37950</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>33800</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
<td>33760</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td>39180</td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td>17250</td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>05-30-1980</td>
<td>15340</td>
</tr>
<tr>
<td>310</td>
<td>Setright</td>
<td>D31</td>
<td>09-12-1964</td>
<td>15900</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
<th>COMMISSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spenser</td>
<td>E21</td>
<td>06-19-1980</td>
<td>26150</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>Lucchesi</td>
<td>A00</td>
<td>05-16-1988</td>
<td>38170</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>12-05-1963</td>
<td>37950</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>33800</td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
<td>33760</td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td>39180</td>
<td></td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td>17250</td>
<td></td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>05-30-1980</td>
<td>15340</td>
<td>4780</td>
</tr>
<tr>
<td>310</td>
<td>Setright</td>
<td>D31</td>
<td>09-12-1964</td>
<td>15900</td>
<td>3200</td>
</tr>
</tbody>
</table>
### ENTITY SUBTYPE/SUPERTYPE

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIRDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spencer</td>
<td>E21</td>
<td>06-15-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchesi</td>
<td>A00</td>
<td>05-15-1958</td>
<td>31710</td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>12-05-1963</td>
<td>37950</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>38000</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
<td>33760</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td>39180</td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td>17250</td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>03-30-1980</td>
<td>15340</td>
</tr>
<tr>
<td>310</td>
<td>Serright</td>
<td>D31</td>
<td>09-12-1964</td>
<td>15900</td>
</tr>
</tbody>
</table>

### EMPTY ASSERTIONS

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIRDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spencer</td>
<td>D21</td>
<td>06-15-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchesi</td>
<td>A00</td>
<td>05-15-1958</td>
<td>31710</td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>12-05-1963</td>
<td>37950</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>38000</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
<td>33760</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td>39180</td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td>17250</td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>03-30-1980</td>
<td>15340</td>
</tr>
<tr>
<td>310</td>
<td>Serright</td>
<td>D31</td>
<td>09-12-1964</td>
<td>15900</td>
</tr>
</tbody>
</table>
MANY-VALUED LOGIC

- Value not valid (e.g. unprevented integrity violation): unknown
- Value not supplied (refusal to answer a question): unknown
- Value does not exist (e.g. some employees do not earn a commission): inapplicable
- Value undefined: e.g. (e.g. calculated column with 0 divisor): inapplicable
- Value is empty set (e.g. outer-joins): inapplicable

GUARANTEED CORRECTNESS

- Relational model
  - First Order Predicate Logic (2VL)
    - Accurate representation
    - Correct derivation
"[Even if] it would be possible to define a 3VL that is logically self-consistent ... in a [database] system based on [it], certain conclusions will follow that are 'logically incorrect' in the real world"

- C.J. Date

- Propositions true in 3VL not necessarily true in the real world:
- Tables with missing values are not relational tables
- Normalization rules break down;
- Missing value marks not typed;
- No criteria for comparing missing values;
PROHIBITIVE COMPLEXITY

<table>
<thead>
<tr>
<th></th>
<th>2VL</th>
<th>3VL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n=2)</td>
<td>4</td>
<td>27</td>
</tr>
<tr>
<td>Monadic operators</td>
<td>16</td>
<td>19,683</td>
</tr>
<tr>
<td>Dyadic operators</td>
<td>16</td>
<td>19,683</td>
</tr>
</tbody>
</table>

- Counterintuitiveness
- Implementation errors
- Interpretation errors

USER DISCIPLINE

"Suppose employee Joe is not a salesperson and so does not qualify for a commission [value inapplicable]. Then Joe's commission [would] be misrepresented as 'value unknown' ... One simple consequence of such misrepresentation is that Joe's total compensation (salary plus commission) will incorrectly evaluate to 'unknown' instead of to just the salary value".

- C.J. Date
### METADATA

"The kind of data we are concerned with here is most commonly encountered in audit trails in which the source of the data, data entry operator ID, time of entry, and so on are captured. Extending an audit trail concept to include metadata about missing information would seem natural. To accomplish this goal, we would need to capture [for each table] ... the primary key [value] of the row involved, a code classifying the data entry operator's belief, and an identifier for the column involved (but not a column value). This metadata can be recorded in separate lookup tables. While this solution can be implemented manually, why should not RDBMS products support audit trails in which the information to be captured is declaratively specified by the user?"

— David McGovern

---

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Speiser</td>
<td>E21</td>
<td>06-30-1980</td>
<td>26150</td>
</tr>
<tr>
<td>110</td>
<td>Lucchessi</td>
<td>A00</td>
<td>05-16-1958</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>12-05-1963</td>
<td>37950</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>33800</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>07-05-1979</td>
<td>11760</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td></td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td></td>
</tr>
<tr>
<td>290</td>
<td>Parke</td>
<td>D31</td>
<td>05-30-1980</td>
<td>15340</td>
</tr>
<tr>
<td>310</td>
<td>Seright</td>
<td>D31</td>
<td>09-12-1964</td>
<td>15900</td>
</tr>
</tbody>
</table>

- Existence of salary
- Knowledge of salary amount
### DBMS SUPPORT: MANIPULATION

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spencer</td>
<td>F21</td>
<td>06-19-1980</td>
</tr>
<tr>
<td>110</td>
<td>Lucchessi</td>
<td>A00</td>
<td>05-16-1958</td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>02-05-1963</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>05-30-1980</td>
</tr>
<tr>
<td>310</td>
<td>Setright</td>
<td>D31</td>
<td>09-17-1964</td>
</tr>
</tbody>
</table>

**SELECT emp#, name, salary**  
**FROM emp, emp_sal**  
**WHERE emp.emp# = sal_emp.emp#**

### DBMS SUPPORT: META-TABLES

<table>
<thead>
<tr>
<th>EMP#</th>
<th>NAME</th>
<th>DEPT#</th>
<th>HIREDATE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Spencer</td>
<td>F21</td>
<td>06-19-1980</td>
<td>26750</td>
</tr>
<tr>
<td>110</td>
<td>Lucchessi</td>
<td>A00</td>
<td>05-16-1958</td>
<td>37950</td>
</tr>
<tr>
<td>120</td>
<td>O'Connell</td>
<td>A00</td>
<td>02-05-1963</td>
<td>33500</td>
</tr>
<tr>
<td>130</td>
<td>Quintana</td>
<td>C01</td>
<td>07-28-1971</td>
<td>28000</td>
</tr>
<tr>
<td>240</td>
<td>Marino</td>
<td>D21</td>
<td>12-05-1979</td>
<td>15340</td>
</tr>
<tr>
<td>250</td>
<td>Smith</td>
<td>D21</td>
<td>10-30-1969</td>
<td>13600</td>
</tr>
<tr>
<td>260</td>
<td>Johnson</td>
<td>D21</td>
<td>09-11-1975</td>
<td>33750</td>
</tr>
<tr>
<td>290</td>
<td>Parker</td>
<td>D31</td>
<td>05-30-1980</td>
<td>15340</td>
</tr>
<tr>
<td>310</td>
<td>Setright</td>
<td>D31</td>
<td>09-17-1964</td>
<td>13600</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PKVALUE</th>
<th>COLUMN</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>Salary</td>
<td>UNK</td>
</tr>
<tr>
<td>250</td>
<td>Salary</td>
<td>UNK</td>
</tr>
<tr>
<td>260</td>
<td>Salary</td>
<td>UNK</td>
</tr>
</tbody>
</table>

-66-
PRACTICAL IMPLICATIONS

1. SQL NULLs
2. User Options
   1. NULLs and 4VL
   2. NULLs and 3VL
3. 2VL and Metadata
SQL NULLS

- Aggregate functions (SUM(), AVG()) ignore NULLs (except for COUNT());
- Scalar expressions on empty tables evaluate to NULL, instead of 0;
- "NULL=NULL" evaluates to NULL, but is actually invalid in SQL;
  - ORDER BY treats NULLs as equal;
  - Precede or follow "regular" values left to the DBMS vendor;
- "x IS NOT NULL" => "NOT(x IS NULL)";
- SQL's NOT <> natural language 'not';
  - "NOT is not Not!"
- SQL's EXISTS does not behave like 3VL EXISTS;
- For integrity purposes NULL is treated as true;

NULLS (cont'd)

- 3VL
  - Implementation errors
  - No truth-valued domain
    - NULL not typed
    - No full support of all 3VL operators
      - 27 monadic
      - 19,683 dyadic
NULLS AND 4VL

- ADMISSIONS
  - ADM#, PAT_ID, CLASS, SPONSOR, HOSP_ID, THOSP_ID
  - THOSP_ID: 3/95

- DELIVERIES {DEL#, ...}
  - 1/76, 1/78, 5/78, 1/80, 9/80, 10/80, 7/86, 8/89, 4/94, 3/95
NULLS AND 3VL

ADM76: {ADM#, PAT_ID, CLASS, SPONSOR}
ADM95: {ADM#, THOSP_ID}

► Inapplicable values
► Outer-Joins
► 3VL problems

2VL AND METADATA

► Table proliferation
  ► 5 attributes --> 15
  ► 6 attributes --> 36
► Manual meta-table system?
► Manual manipulation modification?
► SQL --> NULLs
CONCLUSIONS

- Imperfect knowledge --> Limited assertability
- Real solution: 0 tolerance for missing values*
  - No DBMS metadata support --> disincentive
- SQL (NULLs)
  - Allows missing values
  - 3VL without operations
  - Flawed implementation

RECOMMENDATIONS

- Avoid/Minimize NULLs in base tables
- Exercise extreme care
  - Formulating queries
  - Interpreting results
"The SQL Language Reference" manual contradicts itself right away with the definition of NULL: NULL is not equivalent to zero or blank. Empty strings have a null value, but in subsequent sections on "null and search conditions" the manual doesn't distinguish NULL and zero-length string. It goes on to explain that

`SELECT ... WHERE x IS NULL`

returns NULL rows as well as empty strings. The problem is, the converse is not true. Such a statement is guaranteed to return nothing.

`SELECT ... WHERE x = ''`

This is a nasty exception! Some further confusion in the manual can be found in the following: if any item in an expression contains a null value, then the result of evaluating the expression is null. However, this doesn't seem to apply to NULL string. These expressions simply treat NULL as empty strings.

`@length('yy' || 'x')`

Furthermore, x IS NULL never results in null! It seems the vendor has major confusion on NULL.