MISSING VALUES IN A RELATIONAL DATA BASE

Nicolau Reinhard*

WP 966-78 January 1978
MISSING VALUES IN A RELATIONAL DATA BASE

Nicolau Reinhard*

WP 966-78 January 1978

Professor Reinhard from the University of Sao Paulo, Brazil, was a Visting Fellow at the MIT Sloan School during the Fall Term 1977.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Review and Scope of the Paper</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>A Relational Data Model for Missing Values</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Extended Relational Operations</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>Degradation of a Relation Due to Missing Values</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>Example</td>
<td>26</td>
</tr>
<tr>
<td>7</td>
<td>Conclusions</td>
<td>27</td>
</tr>
<tr>
<td>8</td>
<td>References</td>
<td>29</td>
</tr>
<tr>
<td>9</td>
<td>Appendix</td>
<td>31</td>
</tr>
</tbody>
</table>
Abstract

Although missing (or null) values are found in most databases they are not formally recognized by existing DBMS.

The correct manipulation of those values is then left to the applications programs.

This paper extends and formalizes a model suggested by Codd (4) for the explicit recognition of missing values by the relational algebra. It also explores the implications of the model and discusses useful applications.

Key words

Relational Data Base, Relational Algebra, missing values, null values
ACKNOWLEDGEMENTS

I would like to thank Prof. Stuart E. Madnick (Sloan School of Management - MIT) for his helpful suggestions and advice.

This research was partially supported by the Fundação de Amparo à Pesquisa do Estado de São Paulo (Brasil) under Grant number 05-Economia-76/0056
1. **Introduction**

"The biggest problem that has not been faced by the developers (of Data Base Management Systems) is the pragmatic problem that data, unlike programs, can be plagued with errors, omissions and inconsistencies." (Gosden (6)).

In fact, many systems are built based on the assumption of perfect and complete data and cannot "fail soft" when some data elements are not available or incorrect.

In this paper we will consider the special case of missing (or null) values in a relational data base.

This situation can occur if the data collection process was incomplete, or if some data failed to pass the integrity tests. (Hammer (8)).

Hardware and software failures, as well as faulty systems operation can also be the cause of missing data. Grant (7) presented another interpretation: the missing value indicating a domain which does not apply to the tuple. Example: In the relation EMPLOYEE the domain MAIDENNAME does not apply to male employees. According to Smith (11) the relation EMPLOYEE can can be considered as a generalization of the relations MALE EMPLOYEE and FEMALE EMPLOYEE. These relations have different domains. The "doesn't apply" missing value, therefore, can always be eliminated by proper partition of the relation, and will not be considered in this paper.
When the user's view of a relation does not include all domains, any insertion of a new tuple may require the DBMS to add null values for the domains which are hidden from this user, in order to make the new tuple compatible with the global Data Model. (Date (5)).

Simply ignoring these incomplete data can produce erroneous results for some queries, while others may not even be affected.

Therefore, an effective DBMS should be able to accept and operate with these missing values.

2. Review and Scope of the Paper

The existence of missing values is formally ignored by many DBMS. Therefore the Data Base manager or applications programmer has to select an adequate representation for these missing values and design programs to manipulate them properly.

Some systems use data compression, encoding and other techniques to make better use of storage space. Maxwell and Severance (10) present the performance characteristics of such methods and special ways of storing missing values.

These techniques are concerned only with the physical storage of missing values and do not address the logical implications of having them in the data base.

In Codd (2) proposes the explicit recognition of missing values in the relational calculus. He uses a three-valued
logic whose values could be interpreted as "true", "false" and "maybe" (or unknown).

Some of these concepts were already presented by CODASYL in its 1962 report: An Information Algebra (1) but without further development.

Sublanguage ALPHA (2) has a construct that allows the user to specify the retrieval of tuples for which the qualification expression evaluates to "true", or those tuples for which it evaluates to "maybe".

An indication on how this can be implemented in a relational algebra is presented in (Codd (4)). Each occurrence of a missing value is considered as a placeholder for a possibly distinct value. This interpretation, however, cannot be applied consistently to both input and output relations, and no provision is made for handling "maybe" tuples in the input, a necessary condition for processing correctly certain queries that require the storage of intermediate results.

This paper extends the model presented in (4) by

A) giving a precise and uniform interpretation to the "certain" (or true) and "maybe" parts of all involved relations.

B) redefining the relational algebra operations to be consistent with this interpretation ("certain" and "maybe"
tuples, for example, have to be treated differently by some operations).

The paper also explores some implications and uses of the model, such as measures for the degradation of a relation due to missing values and sensitivity analysis of queries with respect to missing values.

3. A Relational Data Model for Missing Values

Let \( R \) be a relation on domains \( d_1, d_2, \ldots, d_n \) and having a "missing value" in the \( k \)-th domain of a tuple \( t \in R \).

According to Codd (4), this missing value is a placeholder for and can be replaced by any non-null value.

We will define this missing value as a random variable that can assume any value of the \( k \)-th domain \( d_k \), constituted by the values

\[
d_k = v_1, v_2, \ldots, v_{m_k},
\]

where \( m_k \) is assumed to be finite.

Each value \( v_i \) of \( d_k \) has a probability \( g_i \) of being selected in a sample of \( d_k \). Since these events are mutually exclusive and exhaustive we have

\[
\sum_{i=1}^{m_k} g_i = 1
\]
Having no additional information about the missing value, we assume that all values \( v_i \) of domain \( d_k \) are possible, and no one is certain, that is

\[
0 < g_i < 1 \quad \text{for } i=1, 2, \ldots, m_k \quad (m_k > 1)
\]

We can now define a **sample of a tuple** \( t \in R \) as the sequence of values

\[
\bar{t} = (\bar{s}_1 \bar{s}_2 \ldots \bar{s}_n)
\]

where \( \bar{s}_i \) is a sample of the \( i \)-th domain of tuple \( t \).

If the \( i \)-th domain is the value \( v_i \) which is not a missing value, its samples will be defined as having always the value \( v_i \).

For each tuple \( t_i \in R \) we also define a variable \( p_i \) to be called its **presence probability**, restricted to the interval \( 0 < p_i \leq 1 \).

Tuples provided by the user (resulting from data collection) have presence probability equal to 1.

The relational algebra operations to be defined in section 4, however, can generate tuples with presence probability

\[
0 \leq p_i \leq 1.
\]

Let \( \bar{R} \) be a relation formed by one sample from each tuple \( t_i \) of \( R \) and their associated presence probabilities \( p_i \).
\[ p_1 \bar{t}_1 \]
\[ p_2 \bar{t}_2 \]
\[ \quad \cdots \quad \]
\[ \quad \cdots \quad \]
\[ p_m \bar{t}_m \]

With each row \( i \) or \( R \) we associate a random variable that can take the values \( \bar{t}_i \) and \( \phi \) (the empty set), with probabilities \( p_i \) and \( (1-p_i) \) respectively.

A sample of relation \( \bar{R} \) is obtained by taking a sample from each of these variables in relation \( \bar{R} \).

Obviously tuples with \( p_i = 0 \) should not be stored in \( R \), since they can only produce the sample \( \phi \).

Let \( R_c \) be the relation formed by the tuples of \( R \) having presence probabilities equal to 1, and consider the relation \( R_m \) constituted by the remaining tuples of \( R \). Therefore, every sample of \( R \) contains a sample from each of the tuples of \( R_c \), but not necessarily a sample from each of \( R_m \). In fact, some samples of \( R \) may contain only samples from \( R_c \).

For this reason we will call \( R_c \) the "certain" part of \( R \) and \( R_m \) will be defined as its "maybe" part.

A relation \( S \) having no missing values and such that \( S_m = \phi \) can produce only one sample: the relation \( S \) itself.
We will assume that all random variables in the model are independent, i.e., the outcome of any sampling does not depend on any previous sample.

The actual values of the probabilities may be different in each instance of the data base. In this paper, however, we are interested only in determining whether an event is impossible (p=0), certain (p=1) or possible, but not certain (0<p<1). No assumption is made about the distribution of these values. Any probability in the last class will be represented as p and no distinction will be made between values in this class. This assumption does not affect the results of our analysis.

Impossible events are not stored in the data base. Therefore tuples can only have presence probability 1 or p. Example.

Consider the relation $T = (T_c, T_m)$ defined on domains $d_1 = \{u,v\}$ and $d_2 = \{1,2\}$ with

\[
\begin{array}{c|c}
T_c & T_m \\
\hline
v & 1 \\
1 & u \theta \\
\theta & 1 \\
\end{array}
\]

where the symbol "\theta" represents a missing value.

The following relations are samples of $T$:

\[
\begin{array}{ccccccc}
\bar{T}_1 & \bar{T}_2 & = & \bar{T}_3 & \bar{T}_4 & \bar{T}_5 \\
v & 1 & v & 1 & v & 1 & v & 1 \\
u & 1 & u & 1 & v & 1 & u & 2 \\
u & 2 & u & 2 \\
\end{array}
\]

-7-
4. **Extended Relational Operations**

4.1. **Extended and "Normal" Relational Operations**

Let R and S be relations as defined in section 3 and a relational algebra operation \( O \). If R and S have compatible domains we can apply this operation to any pair of samples \( R_i, S_j \) of R and S obtaining the relation \( T_{ij} \). Let \( T \) be the set of all relations \( T_{ij} \) that can be obtained by applying \( O \) to different pairs of samples of R and S.

The set \( T \) will be defined as the correct result of \( R \circ S \).

We will now define an "extended" operation \( O' \) with the following property: Applying this operation \( O' \) to the relations R and S produces a relation \( T' \) such that:

1. Any relation \( T_{ij} \in T \) is a sample of \( T' \).
2. For any sample relation \( T' \) of \( T' \) all tuples of \( T' \) are in some relation \( T_{ij} \in T \) (except for the noise-tuples as will be discussed in section 4.2).
3. Furthermore, all tuples having a sample in every tuple \( T_{ij} \in T \) (and only those) belong to the certain part of \( T' \).

As a result we have that

4. If C is the set of tuples in the certain part of \( T' \) that have no missing values then \( C \subseteq T_{ij}, V_{ij} \) such that \( T_{ij} \in T \)

5. If R and S have no missing values or maybe tuples,
then $T'$ too, will have no missing values or maybe tuples and its only sample will be the only relation in $T$. Therefore, the relational algebra operation $O$ can be considered to be a special case of the "extended" operation $O'$.

Example.

Consider the relations $R = (R_c, \phi)$ and $S = (S_c, \phi)$ defined on domains $d_1 = \{u, v\}$ and $d_2 = \{1, 2\}$ with

$$
\begin{array}{cc}
R_c & S_c \\
v & 2 & v & 2 \\
@ & 1 & u & @ \\
u & 2
\end{array}
$$

Samples of $R$ and $S$ are:

$$
\begin{array}{cccc}
\overline{R}_1 & \overline{R}_2 & \overline{S}_1 & \overline{S}_2 \\
v & 2 & v & 2 \\
u & 1 & v & 1 \\
u & 2 & u & 2
\end{array}
$$

Consider the operations $T = R \cap S$ performed on samples of $R$ and $S$. The possible outcomes for $T$ are:

$$
\begin{array}{cccc}
\overline{T}_{11} & \overline{T}_{12} & \overline{T}_{21} & \overline{T}_{22} = \overline{T}_{12} \\
v & 2 & v & 2 \\
u & 1 & u & 2
\end{array}
$$
According to section 4.2.2. $T' = R \cap S$ will be defined as $T' = (T'_c, T'_m)$ with

\[
\begin{array}{cc}
T'_c & T'_m \\
v 2 & u 1 \\
& u 2 \\
\end{array}
\]

Note that $T'_c$ is in all samples of $T$, but none of these samples contain both $(u 1)$ and $(u 2)$.

4.2. Definition of the "Extended" Relational Algebra Operations

According to Codd (1), an algebra having the operations union, intersection, difference, projection, cartesian product, join, restriction, and division is relationally complete.

It has been shown that the division can be defined from the remaining operations.

We will extend these operations to handle the data model defined in section 3.

4.2.1. UNION $T' = R \cup S$

Given relations $R$ and $S$ defined on the same domains, we define $T'$ as the relation formed by the tuples of $R$ and the tuples of $S$. Since there is a 1 to 1 correspondence between tuples of $R$ and $S$ on one side and tuples of $T'$ on the other, any sample of $R$ and $S$ has its corresponding sample in $T'$ and vice-versa.
Reduction of "similar" tuples (see appendix ) eliminates the storage of redundant information.

4.2.2. \textbf{INTERSECTION} \( T' = R \cap S \)

Consider relations \( R \) and \( S \) defined on the same domains and tuples \( r \in R \) and \( s \in S \), with presence probabilities \( p_r \) and \( p_s \).

Any samples \( \overline{r} \) of \( r \) and \( \overline{s} \) of \( s \) are equal only if its domains are pairwise equal. For non-null (non-empty) samples of \( \overline{r} \) and \( \overline{s} \) we have

\[
P(\overline{r} = \overline{s}) = P(r_1 = s_1).P(r_2 = s_2). \ldots . P(r_n = s_n)
\]

where \( r_i, s_i, i=1,2,\ldots,n \) are the random variables associated with the \( i \)-th domain of \( r \) and \( s \).

For any \( 0 < i < n \), if \( r_i \) and \( s_i \) are not missing values then \( P(r_i = s_i) = 1 \) if \( r_i = s_i \) and

\[
P(r_i = s_i) = 0 \quad \text{otherwise}
\]

The random variable \( y_i = r_i \) generates this sample.

If only one of the elements, say \( r_i \), is a missing value, then

\[
P(r_i = s_i) = p
\]

and the random variable \( y_i = s_i \) generates this event. Conversely is \( s_i \) is the only missing value, then \( y_i = r_i \) produces the sample for which \( r_i = s_i \).
If both \( r_i \) and \( s_i \) are missing values, then any value of the \( i \)-th domain can be in the intersection of \( r \) and \( s \) with presence probability \( p \). The random variable \( y_i = \emptyset \) generates these samples.

Table 1 summarizes these results.

<table>
<thead>
<tr>
<th></th>
<th>( a )</th>
<th>( b )</th>
<th>( \emptyset )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>( 1, a )</td>
<td>( 0, \emptyset )</td>
<td>( p, a )</td>
</tr>
<tr>
<td>b</td>
<td>( 0, \emptyset )</td>
<td>( 1, b )</td>
<td>( p, b )</td>
</tr>
<tr>
<td>( \emptyset )</td>
<td>( p, a )</td>
<td>( p, b )</td>
<td>( p, \emptyset )</td>
</tr>
</tbody>
</table>

The pair \( (x, y) \) in the row \( k \), column \( 1 \) represents:
- \( y \)- the random variable that generates all samples of the intersection of \( r_k \) and \( s_k \).
- \( x \)-probability of these events

**TABLE 1**

Therefore we can define \( t = rns \) with presence probability \( p_t \) as

\[
t = (y_1 \ y_2 \ \ldots \ y_n)
\]

\[
p_t = p_r \cdot p_s \cdot x_1 \cdot x_2 \cdot \ldots \cdot x_n
\]

where domains \( y_1, y_2, \ldots, y_n \) and probabilities \( x_1, x_2, \ldots, x_n \) are defined according to Table 1.
\( r \cap S \) will be defined as
\[
\begin{align*}
\text{r} \cap S &= (r \cap s_1) \cup (r \cap s_2) \cup \ldots \cup (r \cap s_m)
\end{align*}
\]
where \( s_1, s_2, \ldots, s_m \) are the tuples of \( S \)

\( R \cap S \) can then be defined by
\[
T' = R \cap S = (r_1 \cap S) \cup (r_2 \cap S) \cup \ldots \cup (r_k \cap S)
\]
where \( r_1, r_2, \ldots, r_k \) are the tuples of \( R \).

4.2.3. DIFFERENCE \( T' = R - S \)

Consider \( r \in R, \) \( s \in S \) with presence probabilities \( p_r \) and \( p_s \) and \( \bar{r}, \bar{s} \) samples of \( r \) and \( s \) respectively.

\[
\begin{align*}
\bar{r} - \bar{s} &= \emptyset \text{ if a) } \bar{r} = \bar{s} \neq \emptyset \text{ or b) } r = \emptyset \\
\bar{r} - \bar{s} &= \bar{r} \neq \emptyset \text{ if c) } \emptyset \neq \bar{r} \neq \bar{s} \neq \emptyset \\
&\quad \text{ or d) } \bar{s} = \emptyset \text{ and } \bar{r} \neq \emptyset
\end{align*}
\]

The events a), b), c) and d) are mutually exclusive and exhaustive, therefore
\[
\begin{align*}
\hat{p}(r - s = r \neq \emptyset) &= p_r p_s \cdot P(r \neq s \mid r \neq \emptyset) + p_r (1 - p_s)
\end{align*}
\]

1) If \( r \cap s = \emptyset \) then \( P(r \neq s \mid r \neq \emptyset) = 1 \) and
\[
\begin{align*}
P(r - s = r \neq \emptyset) &= \hat{p}_r p_s + p_r (1 - p_s) = p_r \text{ for all } \bar{r},
\end{align*}
\]
that is, tuple \( r \) with presence probability \( p_r \) generates all samples of \( r - s \).
II) Now consider the case \( r \cap s \neq \phi \).

a) If \( r \) and \( s \) have no missing values then
\[
p(r \neq s \mid r \neq 0) = 1 - p_s
\]
and
\[
P(r - s = r \neq \phi) = p_r \cdot p_s \cdot (1 - p_s) + p_r \cdot (1 - p_s)
\]
If \( p_s = 1 \) then \( P(r - s = r \neq \phi) = 0 \) and therefore \( r - s = \phi \).
If \( p_s < 1 \) then \( P(r - s = r \neq \phi) = p \) and tuple \( r \) is the only non-null sample of \( r - s \).

b) If \( r \) and \( s \) have missing values then
\[
P(r \neq s \mid r \neq \phi) = p \text{ for all samples of } r \cap s
\]
and
\[
P(r \neq s \mid r \neq \phi) = 1 \text{ otherwise.}
\]
Therefore tuple \( r \) with presence probability \( p \) generates all samples of \( r - s \).

c) If only \( r \) has missing values then
\[
P(r \neq s \mid r \neq \phi) = 0 \text{ for the sample } \overline{r} = s,
\]
and
\[
P(r \neq s \mid r \neq \phi) = 1 \text{ otherwise.}
\]
Tuple \( r \) with presence probability \( p \) generates all samples of \( r - s \), but also the "noise tuple" \( s \), not contained in \( r - s \).

These results can be summarized in a decision table (Table 2).
The values $p_r$, $p_s$, $p_t$ are the presence probabilities of tuples $r$, $s$ and $t$ respectively.

Note that decision rule 5 generates the "noise tuple" $s$.

For the definition of $r - S$ we calculate

$t_1 = r - s_1$  where $s_1, s_2, \ldots, s_m$ are
$t_2 = t_1 - s_2$  the tuples of $S$.

\[
\begin{align*}
  \cdots \\
  \cdots \\
  \cdots \\
  t_m &= t_{m-1} - s_m
\end{align*}
\]
and define $r - S = t_m$ with its associated presence probability $p_{t_m}$.

The relation $T' = R - S$ will be defined as

$$T' = (r_1 - S) \cup (r_2 - S) \cup \ldots \cup (r_k - S)$$

where $r_1, r_2, \ldots, r_k$ are the tuples of $R$.

Note that having "noise tuples" is only a problem of representation. Since it is always possible to identify them when they are generated, the "noise tuples" can be recorded separately and the result presented as $T' - N$, where $N$ is the union of all the generated "noise tuples".

4.4.4. **PROJECTION** $T' = R[A]$ where $A$ is a subset of the set of domains of $R$.

Eliminating the domains of $A$ from the tuples of $R$ we have a new relation $T'$ whose tuples and samples have a 1 to 1 correspondence with the tuples and samples of the relation $R$.

Therefore the tuples in the new relation have the same presence probability as their corresponding tuples in $R$.

The projection operation does not recognize missing values. "Similar tuples" in $T'$ can be reduced (see appendix).

4.2.5. **CARTESIAN PRODUCT** $T' = R \times S$

There is also a 1 to 1 correspondence between tuples of $r$ and $s$ on one side and $r \times s$, the cartesian product of $r$ and $s$.
on the other. Therefore, the tuple \( r \times s \) has presence probability \( p_r \cdot p_s \).

This operation, too, does not recognize missing values.

4.2.6. \( \Theta \)-Join \( T' = R[A \ B] S \)

where \( A \) and \( B \) are subsets of the sets of domains of \( R \) and \( S \) respectively, and

\[ \Theta \in \{ =, \neq, \leq, \geq, <, > \} \]

Consider the tuples \( r \in R \), \( s \in S \), with presence probabilities \( p_r \) and \( p_s \) respectively.

If \( r[A] \) and \( s[B] \) have no missing values then \( r \bowtie s \) will be a tuple of \( T' \) if \( r[A] \bowtie s[B] \), with presence probability \( p_r \cdot p_s \), the probability of samples of both, \( r \) and \( s \) being non-null tuples.

If \( r[A] \) and/or \( S[B] \) have missing values, \( r[A] \bowtie s[B] \) cannot be evaluated directly. Since it is difficult to derive a general expression for all \( \Theta \)'s and the result may depend on the position of the missing values in the tuples, we define \( r \bowtie s \) to be in \( T' \) with presence probability \( p \).

This tuple obviously generates all samples of \( r \) and \( s \) for which \( r[A] \bowtie s[B] \), but will also generate the "noise" tuples of \( r \bowtie s \) for which \( r[A] \bowtie s[B] \).

There is a special case in which this noise can be eliminated: If \( \Theta \) is = (the equi-join operation), then
\[ r[A] = s[B] \] for all samples of \( y \), where \( y \) is the tuple \( y = r[A \land s[B]] \).

The tuple \( y \) and its presence probability \( p_y \) are calculated as defined in section 4.2.2.

Substituting \( r[A] \) and \( s[B] \) in \( r \land s \) by \( y \), this tuple will generate all the samples (and only those) for which \( r[A] = s[B] \). The new tuple has presence probability \( p_r \cdot p_s \cdot p_y \).

Eliminating the noise in the other cases requires the enumeration of the values of the domains having a missing value and will not be considered here.

4.2.7. **RESTRICTION** \( T' = R[A \land B] \)

Given a tuple \( r \in R \), if \( r[A] \) and \( r[B] \) have no missing values, \( r[A \land B] \) is evaluated in the usual way and \( r \) will be in \( T' \) if \( r[A \land B] \).

If \((r[A] \text{ and/or } r[B])\) have missing values, we define \( r \) to be in \( T' \) with presence probability \( p \). This will also generate the "noise tuples" of \( r \) for which \( r[A] \notin r[B] \).

5. **Degradation of a relation due to missing values**

5.1. **Definition of a Measure of Degradation**

Missing values and maybe tuples reduce the information content of a Database. This degradation is propagated through relational algebra operations and affects the usefulness of
the outputs resulting from queries of the database. It then becomes important to define a measure of this degradation and to predict the degradation of results of a query without actually carrying out the operations.

Given a probability distribution \( P \) with values \( p_1, p_2, \ldots, p_n \) with \( p_i > 0, i=1,2, \ldots, n \) and \( \sum p_i = 1 \), Information Theory [9] defines the entropy of \( P \) as

\[
H_p = - \sum_{i=1}^{n} p_i \log_2 p_i \quad \text{where } 0 \log_2 0 \text{ is defined as } 0
\]

\( H_p \) is used as a measure of uncertainty.

It can be shown that \( H_p \) is minimal (\( H_p \min =0 \)) if and only if \( P \) has only one non-zero value and that \( H_p \) is maximal if \( p_1 = p_2 = \ldots = p_n = 1/n \). In this case

\[
H_p \max = - \sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = \log_2 n \quad (1)
\]

In section 3, a relation \( R \) was defined as a random variable whose values were called samples of \( R \). According to (9) the maximum entropy of this distribution is

\[
H_R = \log_2 N_R \quad , \text{where } N_R \text{ is the number of samples of } R
\]

If the user asks questions of the type:

- What are the sample of relation \( R \)?
- Is there a sample of \( R \) containing all the tuples \( t_1, t_2, \ldots, t_k \) ? \((K>1)\)
- How many tuples are there in a sample of \( R \)?
then $H_R$ (or simply $N_R$) can be used as a measure of the degradation of $R$.

Consider $R$ having $K$ missing values and $m$ tuples in the "maybe" part. For the sake of simplicity assume that all $D$ domains of $R$ have the same number of values $n$.

It is easy to show that $N_R$ is minimal when all $K$ missing values are in the same tuple in the maybe part of $R$ (this obviously requires $K < D$). In this case we have

$$N_{\text{min}} = (n^K + 1) \cdot 2^{m-1}$$

since the tuple containing $K$ missing values has $n^K$ non-null samples and 1 null sample. The remaining $m-1$ tuples in the maybe part of $R$ have 2 samples each. Since these samples are assumed to be independent (see section 3) the total number of samples is the product of these values.

It can also be shown that $N_R$ is maximal when all missing values are in the certain part of $R$.

$$N_{\text{max}} = n^K \cdot 2^m$$

We will use $N_{\text{max}}$ as an approximation for the number of samples in any relation $R$ having $K$ missing values and $m$ tuples in the maybe part. This is an overestimation by a factor not exceeding 2. In fact

$$F = N_{\text{max}} / N_{\text{min}} = 2 \cdot n^K / (n^K + 1) < 2$$
5.2 Degradation due to the application of the extended relational algebra operations

Consider relations A and B having respectively $K_A$ and $K_B$ missing values, $m_A$ and $m_B$ tuples in the maybe part, and $n_A$ and $n_B$ tuples in both the certain and maybe parts.

5.2.1. **Union** $C = A \cup B$

According to the definition of union in 4.2.1

$$N_C \leq N_A \cdot N_B \quad \text{and} \quad K_C = K_A + K_B$$

5.2.2. **Intersection** $C = A \cap B$

Given tuples $t_A \in A, t_B \in B$, according to 4.2.2.

$t_A \cap t_B$ generates the samples $t_A$ and $\emptyset$ if $t_A = t_B$ and $t_A$ is in the maybe part of A, or if $t_A$ contains no missing values, $t_B$ has missing values and $t_A$ is a sample of $t_B$. Since there are no duplicate tuples in A, relation A can have at most $n^i$ such tuples for each tuple of B with $i$ missing values. The same reasoning can be applied to the tuples of B. Therefore

$$N_C \leq 2^{(m_A + m_B)} \cdot (2^n)^{(K_A + K_B)}$$

Table 1 of 4.2.2. also indicates that $K_C = \min (K_A, K_B)$

5.2.3. **Difference** $C = A - B$

Given tuples $t_A \in A, t_B \in B$, according to 4.2.3.
a. \( t_A - B \) generates the 2 samples \( t_A \) and \( \emptyset \) if \( t_A \) is in the maybe part of \( A \), has no missing values and \( t_A \cap B = \emptyset \).

b. \( t_A - B \) generates \( n^i + 1 \) samples if \( t_A \) has \( i \) missing values and \( t_A \cap B \neq \emptyset \).

c. \( t_A - t_B \) generates the 2 samples \( t_A \) and \( \emptyset \) if \( t_A \) has no missing values, \( t_B \) has \( i \) missing values and \( t_A \) is a sample of \( t_B \). There can be at most \( n^i \) such tuples in \( A \) for each tuple \( t_B \) in \( B \) having \( i \) missing values. Therefore

\[
N_C \leq 2^m_A \cdot (2^n)^m_B \cdot (n + 1)^K_A \cdot (2^n)^K_B \approx N_A \cdot 2^N_B
\]

\[
K_C = K_A
\]

5.2.4. **Projection** \( C = A [a] \), where \( a \) is the set of domains on which \( A \) is projected

\[
N_C \leq N_A \quad \text{and} \quad K_C \leq K_A
\]

5.2.5. **Cartesian Product** \( C = A \times B \)

Since each tuple in \( A \) generates \( n_B \) tuples in \( C \) and each tuple in \( B \) generates \( n_A \) tuples in \( B_C \), the number of samples of \( C \) is

\[
N_C = (N_A \cdot n_B) \cdot (N_B \cdot n_A)
\]

and

\[
K_C = (K_A \cdot n_B) + (K_B \cdot n_A)
\]
5.2.6. $\theta$-Join $C = A \{a \theta b\} B$

Given tuples $t_A \in A$, $t_B \in B$, if $t_A$ has missing values in $t_A \{a\}$, then according to 4.2.6. in principle all tuples $t_B \in B$ could be such that $t_A \{a \theta b\} t_B$ and therefore $t_A \wedge t_B$ becomes an element of the maybe part of $C$.

Pairs of tuples $t_A \wedge t_B$ such that $t_A \{a \theta b\} t_B$ and $t_A$ and/or $t_B$ are in the maybe part of $A$ or $B$ respectively, become a tuple of the maybe part of $C$. Therefore

$$N_C \leq 2 \cdot 2 \cdot 2$$

and

$$N_C = K_A \cdot n_B + K_B \cdot n_A$$

5.2.7. Restriction $C = A[a_1 \theta a_2]$

Any tuple $t_A \in A$ having a missing value in $a_1$ and/or $a_2$ is defined in 4.2.7. as having $t_A \{a_1 \wedge a_2\} = \text{maybe}$, and is therefore a tuple of the maybe part of $C$. Thus

$$N_C \leq 2 \cdot m_A \cdot (n + 1) \cdot K_A \cdot N_A$$

5.3 Comments

In Section 4.2 the correct result of $T = R \bowtie S$ was defined
as the set $T = \{T_{ij}\}$ of all relations that can be obtained by applying operation $O$ to pairs of samples $R_i$ of $R$ and $S_j$ of $S$. Thus, if $R$ and $S$ have respectively $N_R$ and $N_S$ samples, the set $T$ can have at most $N_T = N_R \cdot N_S$ different relations.

The ratio between this number and the number of samples of the relation produced by an operation can be used as a measure of the degradation introduced by that operation.

The previous section presented upper limits to the degradation of relations resulting from the application of relational algebra operations. A more realistic analysis would consider the actual distribution of the values in each domain of the relation, the position of the missing values, etc.

The scope of this analysis, however, is only to present the order of magnitude of the degradation and discuss its implications.

While some operations (union and projection) produce relations that are no more degraded than their inputs, the cartesian product linearly increases the degradation of its output over the degradation of the input relations.

The important result, however, is that the remaining operations can produce relations with a degradation that is an exponential function of the degradation of the input relations.

This exponential increase in degradation stems from the representation of the missing values and their interpretation as independent random variables.
The operation $\cap (v_1 \cup v_2)$, for example, produces the samples $\emptyset, v_1, v_2$. Representing this result as $(v_1)$ with both tuples in the maybe part of the resulting relation, ignores that the samples $(v_1)$ and $(v_2)$ are mutually exclusive.

Capturing this information requires the use of additional symbols or the explicit storage of these dependencies among tuple samples and is equivalent, in the extreme case, to the enumeration of all possible samples.

The implementor has therefore to make the tradeoff between the amount of stored information and the acceptable degradation of the result.

5.4. Another measure of degradation

If, however, the user is only interested in knowing which tuples can (or have to) be in a sample of a relation $R$, then the measure of degradation above is not adequate.

In this case, it is more appropriate to measure the degradation of the relation by the number of its noise-tuples.

According to Section 4, only the operations difference, restriction and $\theta$-join (excluding the equi-join) can produce noise-tuples. Since only missing values in the input can produce noise-tuples in the output, the maximum number of noise-tuples produced is not greater than the number of missing values in the
input for the operations difference and restriction. For the 
\( \emptyset \)-join it is a linear function of the number of missing values
in the input.

It should be noted, however, that these noise-tuples can always be identified as such.

It seems that most data base queries are in this latter category (concerned only with individual tuples in the result, rather than combinations of tuples), for which the presented model is quite adequate.

6. Examples

Consider the relations \( R, S, \) and \( T \) defined on domains
\( d_1 = \{u, v\} \) and \( d_2 = \{1, 2\} \) having the following tuples:

\[
\begin{array}{cccccc}
  R_c & R_m & S_c & S_m & T_c & T_m \\
  u \emptyset & \emptyset & u \emptyset & \emptyset & u \emptyset & \emptyset \\
  u 1 & u 1 & v 2 & v 1 & \emptyset 1 & v 1 \\
  \emptyset 2 & v 1 & \\
\end{array}
\]

The query

\[ V = x \mid x \in T \Leftrightarrow (x \in R \vee x \in S) \] (\( V \) is the exclusive or) can be evaluated as follows:

\[ V = T \cap [(R-S) \cup (S-R)] = T \cap (V_3 \cup V_4) = T \cap V_2 \]
where \( V_3 = R - S \)
\( V_4 = S - R \)
\( V_2 = V_3 \cdot V_4 \)

Applying the extended relational algebra operations the resulting relations will be:

\[
\begin{array}{cccc}
V_{3c} & V_{3m} & V_{4c} & V_{4m} \\
\phi & u @ & v 1 & u @ \\
@ 2 & & & \\
V_{2c} & V_{2m} & V_c & V_m \\
v 1 & u @ & v 1 & u @ \\
@ 2 & & & v 2
\end{array}
\]

Note that relations \( V_3, V_4 \) and \( V \) generate the "noise tuple" \((u 1)\), a sample of tuple \((u @)\), as a result of the existence of the tuple \((u @)\) both in \( R_c \) and \( S_c \).

7. **Conclusions**

The proposed data model allows a **precise** interpretation of the missing values in a relational data base. The extended operations are **consistent** with normal relational algebra operations.

No assumption is made about the missing values other than considering them as independent random variables.

The model allows the data base to "fail soft" when missing values are stored while preserving the correctness of the results of queries.
Some queries require the counting of the tuples in a relation. Given the relation \( T = (T_c, T_m) \), an upper limit for the number of tuples in a sample of \( T \) is \( |T_c| + |T_m| \). The lower limit is given by \( |T'_c| \), where \( T'_c \) is set of tuples in \( T_c \) that have no missing value.

Measures of degradation are proposed and upper limits for this degradation are established when relational algebra operations are performed.

The exponential growth of the degradation for some operations and the generation of noise tuples indicate that the sequence in which (otherwise interchangeable) operations are executed affects the degradation of the resulting relations and should be taken into account in the processing of complex queries.

If this degradation is not acceptable, other representations for relations with missing values could be required (explicit enumeration of the relation's samples, storage of the noise tuples, etc.).

The simulation (i.e., repeated processing of samples of the input relations using normal relational algebra) of a query does not add noise tuples or increase the degradation of the result, but may require considerable additional storage space and computational effort. The presentation of these simulations results to the user would also be a problem, since for any non-
trivial number of missing values in the input there would be a very large number of relations in the output.

In contrast, the model described in Sections 3 and 4 presents the results in a very concise and useful form using only one relation.

The model also provides some useful by-products:

1. Each missing value in the resulting relation and each tuple of its "maybe" part is the consequence of and can be traced back to one or more of the missing values and/or maybe tuples in the input. This permits us to anticipate the effect of "recovering" a missing value (i.e. substituting it by a valid value).

2. Replacing one or more data items by missing values and analyzing the effect on the result of a certain query can be used to determine the sensitivity of this query to those data items.

8. References


11. Smith, J.M. and Smith, D.C.P.

Appendix

REDUCTION OF "SIMILAR" TUPLES

It is possible that two tuples $t_i$ and $t_j$ of $R$ with $i \neq j$ produce the same tuples in a sample of $R$. If $t_i$ and $t_j$ have no missing values and

$$t_{ik} = t_{jk} \text{ for all } 1 \leq k \leq n$$

then $t_i$ and $t_j$ can only produce the same sample $t_i$ in a sample of $R$ (or the empty set, depending on the presence probabilities $p_i$ and $p_j$).

Tuples $t_i$ and $t_j$ will be called "similar" tuples.

The probability that a sample $R$ of $R$ contains the sample $t_i$ (from either $t_i$ or $t_j$) is

$$p' = 1 - (1 - p_i)(1 - p_j)$$

If $p_i$ and/or $p_j$ are equal to 1 then $p' = 1$. Otherwise $p'$ will have value $p$.

"Similar" tuples will be substituted by their equivalent: tuple $t_i$ with presence probability $p'$. Since duplicates are not allowed in a sample of $R$, this substitution will still produce the same set of samples of $R$.

For tuples having missing values a reduction is possible if the user is not interested in knowing the number of tuples.
that can be in a sample of R. Let $t_i$ and $t_j$ be tuples of R such that for all $1 \leq k \leq n$ either $t_{ik} = t_{jk}$ or $t_{jk}$ is a missing value. In this case every sample of $t_i$ is also a sample of $t_j$. We say that $t_i \subset t_j$.

If we eliminate $t_i$ from R, all tuples that are samples of R will also be samples of this new relation and conversely. We lose, however, some information, since some samples of the original relation R may contain a sample of $t_i$ and a (possibly different) sample of $t_j$. 

-33-
<table>
<thead>
<tr>
<th>Date</th>
<th>Due</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEC 7 1984</td>
<td>MAY 30 1990</td>
</tr>
<tr>
<td>DEC 16 1985</td>
<td>JUN 30 1990</td>
</tr>
<tr>
<td>DEC 31 1985</td>
<td>MAR 19 1992</td>
</tr>
<tr>
<td>OCT 9 1986</td>
<td></td>
</tr>
</tbody>
</table>