# SOME CLASSES OF MULTILEVEL RELATIONAL STRUCTURES

(Extended Abstract)

Dirk Van Gucht
Indiana University at Bloomington

Patrick C. Fischer
Vanderbilt University

## 1. Introduction.

The original relational database model proposed by Codd permitted complex structures to be entries in a component of a tuple [Cod]. However, Codd recommended that only atomic data values be permitted. The relational model with this restriction to first normal form (1NF) has gained wide acceptance.

One of the earliest suggestions that 1NF was too restrictive came from Makinouchi in 1977 [Mak]. While his treatment was fairly informal, he showed that a multivalued dependency (MVD) could be treated as a set-valued functional dependency (FD) if one were willing to consider unnormalized relations. Thus, relaxing the 1NF restriction could more faithfully model some database applications. Similar suggestions were later made by Bancilhon et. al. [BRS], Schek and Pistor [SP], Gonnet [Gon], Kambayashi et. al. [KTT], Ozsoyoglu, Ozsoyoglu and Matos [OO,OMO], Macleod [Mac], Kuper and Vardi [KV], Freitag and Appelrath [FA], and Clifford and Tansel [CT].

Jaeschke and Schek introduced the NEST and UNNEST operators to restructure relations from 1NF to unnormalized form [JS]. While they considered nesting only over single attributes, they established some important algebraic results. Thomas and Fischer generalized the model to allow multilevel, multiattribute nesting [FT,Tho,TF]. More recently, Schek, and Schek and Scholl described a similar model [Sch,SS]. Thomas and Fischer established many of the results in [JS] for a more generalized model and showed that no information carried by a flat (1NF) relation is lost by multilevel nesting, even if unnesting is done later in other than strict LIFO order. They also began the study of the effects of nesting on FDs and MVDs; later Saxton and Van Gucht contributed to this effort [FSTV].

In a previous paper we were able to solve the problem: given a flat relation and a one-level nesting scheme (nests permitted only over sets of basic attributes), can one determine in polynomial time whether the order of nesting affects the result [FV1]? This involved studying weak MVDs, which were first defined in [JS] and generalized in [Tho]. We then obtained the "top down" version of this result, i.e., given an unnormalized one-level structure, can one determine in polynomial time whether the unnested version of this structure can be renested in any order to obtain the original structure? Characterizations were given in terms of set-theoretic properties and in terms of families of MVDs and weak MVDs [FV2]. Later, we were able to provide a polynomial time algorithm for the problem: given a one-level structure, does there exist some order of nesting which will transform the underlying flat relation to this structure [FV3]?

In this paper, we are able to remove the one-level restriction and solve (in reverse order) all three of the above problems for the general case. Thus we can determine in polynomial time:

1) whether a given multilevel structure is a nested relation (reachable by some nesting sequence); 2) whether a given multilevel structure is a permutable nested relation (reachable from its unnested form by any renesting sequence); and 3) given a flat relation and an arbitrary scheme, whether or not every nesting sequence for that scheme will produce the same instance when applied to the relation.

We also show that the class of permutable nested relations is the largest subclass of the nested relations which is closed under unnesting. Finally, we compare this class to the hierarchically-organized relations of Delobel [Del] and Lien [Lie] and to the hierarchical structures studied by Bancilhon et. al. [BRS], Abiteboul and Bidoit [AB], Ozsoyoglu and Yuan [OY], and Roth et. al. [RKS].

## 2. Basic Concepts.

Our formalism for schemes is very similar to the normal form of Hull and Yap [HY]. However, we always do set formation (collection) whenever we do aggregation (composition), and we do not use generalization (classification). Thus, our schemes need not have different kinds of non-leaf nodes.

Let U denote the universe of attributes. A scheme S is a directed rooted tree for which the leaf nodes are distinct elements of U. The root node is denoted by R(S). The set of leaf nodes (attributes) of S is denoted by att(S), and the set of non-leaf (interior) nodes of S by int(S). The set of nodes which define proper subschemes of S is called sub(S); hence, sub(S) = int(S) - {R(S)}. The set of nodes which have only leaf nodes as children is called the set frn(S) of frontier nodes of S. If R(S) ∈ frn(S) we will call S a flat scheme. We extend the notions att, int, sub, and frn to subschemes of S, e.g., given a node M ∈ int(S), att(M) means att(T(M)) where T(M) is the subtree consisting of M and its descendants.

For a node M ∈ int(S), the set of children of M, denoted by C(M), is partitioned into attributes A(M) = C(M) ∩ att(S) and higher order nodes H(M) = C(M) ∩ int(S). If we apply C, A or H to a scheme S, we will use the root node R(S); hence, C(S) means C(R(S)), etc. Finally, if Y is a set of nodes, we will label a node $Y^*$ only if $C(Y^*) = Y$.

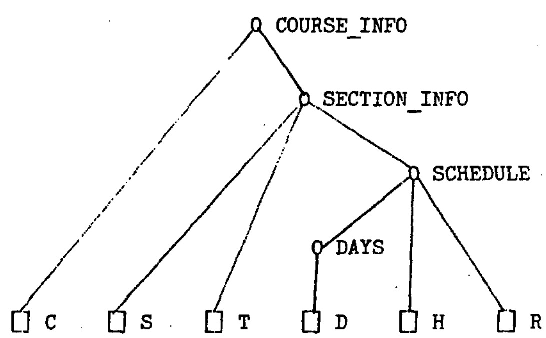
A structure consists of a scheme S and an instance of the root R(S), where an instance is defined recursively as follows:

1. an instance of a leaf node A is a single atomic value from dom(A), the set of possible values of attribute A;

2. an instance of an interior node M of S is a finite set of tuples, where the number of components of a tuple is |C(M)|, and for each N ∈ C(M), a tuple has an entry which is an instance of N, i.e., an atomic value from dom(N) if N ∈ A(M) and a set of tuples over the children of N if N ∈ H(M).

If S is a flat scheme, a structure over S is called a flat relation or simply a relation.

**Example 1.** Consider a database of class schedules for a given term with attributes C (Course), S (Section), T (Teacher), D (Day), H (Hour) and R (Room). This information is often printed out as a three-level nested structure (although set-theoretic braces are usually omitted). We give a scheme $S_1$ and an instance $s_1$ below. The "header" for the instance $s_1$ corresponds to the leaf nodes of the scheme $S_1$.




Scheme $S_1$

| C | S | T | D | H | R |
|---|---|---|---|---|---|
| CS100 | { 01 | Jones | {{M,W} | 10 | EN239} |
| | | | {F}} | 2 | EN201}} |
| | 02 | Smith | {{M,W,F} | 1 | EN203}} |

Instance $s_1$

An alternative way of labelling the nodes of $S_1$ would be to use the notation in [AB]. Then DAYS would be $D^*$, SCHEDULE would be $(D^*HR)^*$, etc.

We now define the NEST and UNNEST operators. First, let t be a tuple in a structure with scheme S and let $X \subset C(S)$. The X-value of t, denoted t[X], is the restriction of the tuple t to the components associated with X. Now let s be a structure over scheme S and let $Y \subset C(S)$. Then $NEST_Y(s)$ is a new structure s' with scheme S', where S' is obtained by inserting a new child $Y^*$ of R(S) into S and making all of the members of Y children of $Y^*$ in S'. Hence $C(S') = (C(S)-Y) \cup \{Y^*\}$. A tuple $v \in s'$ if and only if there exists a tuple $t \in s$ such that:

1. $v[C(S)-Y] = t[C(S)-Y]$ and
2. $v[Y^*] = \Pi_Y(\{t' \in s \mid t'[C(S)-Y] = t[C(S)-Y]\})$.

Thus, the tuples of s which agree outside of Y are merged into a single tuple of s' with a set-valued entry in the $Y^*$ position.

Let s be a structure with scheme S and let $M \in H(S)$. Then $UNNEST_M(s)$ is a structure s" with scheme S", where S" is obtained by deleting M from S and connecting the children of M to R(S), i.e., $C(S") = (C(S)-\{M\}) \cup C(M)$. A tuple $t \in s"$ if and only if there exists a tuple $v \in s$ such that:

1. $t[C(S)-\{M\}] = v[C(S)-\{M\}]$ and
2. $t[C(M)] \in v[M]$.

We now extend the definition of NEST to describe a sequence of NEST operations. Let S be a scheme. A sequence $P(S) = (Y_1,\ldots,Y_n)$ of sets of nodes of S is called a nesting sequence for S if and only if the sequence is a valid order of nesting to reach the scheme S, i.e.:

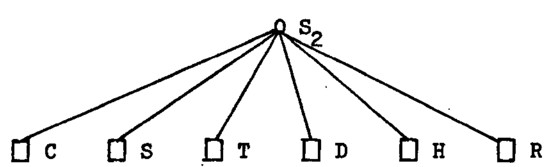
1. for each i, $1 \le i \le n$, $Y_i = C(Y_i^*)$ for some $Y_i^* \in sub(S)$;
2. $\{Y_1^*,\ldots,Y_n^*\} = sub(S)$;
3. for each pair of indices i and j, $1 \le i < j \le n$, either $Y_j^*$ is an ancestor of $Y_i^*$ in S or $Y_i^*$ and $Y_j^*$ are incomparable nodes in S, i.e., neither of them is a descendant of the other.

Let r be a relation over att(S) and let $P(S) = (Y_1,\ldots,Y_n)$ be a nesting sequence for S. The structure $NEST_{P(S)}(r)$ over scheme S is defined as the structure $NEST_{Y_n}(NEST_{Y_{n-1}}\ldots(NEST_{Y_1}(r))\ldots)$.

Since the UNNEST operator commutes [JS,FT,TF], its natural extension deals with unnesting over a set of objects. Let S be a scheme with $Q(S) \subset sub(S)$. We say that Q(S) is an unnesting set of S if and only if whenever a node $M \in sub(S)$ is in Q(S), then all the ancestors of M in sub(S) are also in Q(S). Thus, we may not unnest an object unless higher-level nestings over that object have been unnested. Now let s be a structure over scheme S and Q(S) be an unnesting set of S. Then $UNNEST_{Q(S)}(s)$ is the structure $UNNEST_{M_k}(UNNEST_{M_{k-1}}(\ldots(UNNEST_{M_1}(s))\ldots))$, where the sequence $(M_1,\ldots,M_k)$ satisfies the conditions:

1. $\{M_1,\ldots,M_k\} = Q(S)$ and
2. for each pair of indices i and j, $1 \le i < j \le k$, either $M_j$ is a descendant of $M_i$ in S or $M_i$ and $M_j$ are incomparable nodes in S.

Example 2. The flat relation associated with the structure $s_1$ in Example 1 above is given as the structure $s_2$ with scheme $S_2$ below. One can readily observe that $s_2$ can be obtained from $s_1$ using the unnesting set $sub(S_1) = \{(ST(D^*HR)^*)^*, (D^*HR)^*, D^*\}$. Conversely, the nesting sequence (D, $D^*HR$, $ST(D^*HR)^*$) will transform $s_2$ into $s_1$.




Scheme $S_2$

| C | S | T | D | H | R |
|---|---|---|---|---|---|
| CS100 | 01 | Jones | M | 10 | EN239 |
| CS100 | 01 | Jones | W | 10 | EN239 |
| CS100 | 01 | Jones | F | 2 | EN201 |
| CS100 | 02 | Smith | M | 1 | EN203 |
| CS100 | 02 | Smith | W | 1 | EN203 |
| CS100 | 02 | Smith | F | 1 | EN203 |

Instance $s_2$

As in [Tho] we extend the definitions of FDs and MVDs in the natural way: tuple entries which are non-atomic are equal if they are equal as sets. We similarly extend the definition of weak multivalued dependencies (WMVDs) given in [FV1]. Let X, Y, Z partition C(S) for some scheme S. Then a structure s satisfies the WMVD X -w-> Y|Z if whenever s contains tuples of the form:

| | X | Y | Z |
|---|---|---|---|
| t1 | x | y' | z |
| t2 | x | y | z |
| t3 | x | y | z' |

then s also contains a tuple of the form:

| | | | |
|---|---|---|---|
| t4 | x | y' | z' |

3. The Classes NL, NR, and PNR.

Between relations and arbitrary structures there are many classes of structures. A normalization-lossless structure is a structure that can be fully unnested to a relation and recreated from this relation using only legal NEST and UNNEST operations. This was the main class studied in [Tho,TF]; we shall denote it by NL. Our classes will all be subclasses of NL.

A nested relation is a structure which can be obtained from a relation using only legal NEST operations, i.e., s is a nested relation over scheme S if and only if there exists a relation r over att(S) and a nesting sequence P(S) for S such that $s = NEST_{P(S)}(r)$. We denote the class of nested relations over scheme S by NR(S) and the class of all nested relations by NR. Nested relations can be used at various levels in a DBMS, at the conceptual level to model applications better, at the user interface level to provide for more readable output and at the physical level to store data more efficiently.

We say that a structure s is a permutable nested relation over the scheme S if there exists a relation r over att(S) such that for any nesting sequence P(S) for S, $NEST_{P(S)}(r) = s$. We denote the class of permutable nested relations over scheme S by PNR(S) and denote the class of all permutable nested relations by PNR. Obviously, PNR is contained in NR.

We now give characterizations for when a given structure belongs to NR(S) or PNR(S). Furthermore we show that the classes NL, NR and PNR are distinct and that PNR is the largest subclass of NR which is closed under unnesting.

We begin with a characterization for nested relations.

Theorem 1. Let s be a structure over scheme S and let $P(S) = (Y_1,\ldots,Y_n)$ be a nesting sequence for S. Let $s_n = s$, $S_n = S$, and $s_i = UNNEST_{Y^*_{i+1}}(s_{i+1})$ for $0 \leq i < n$, and let $S_i$ be the scheme of $s_i$ for $0 \leq i < n$. Thus, $s_0 = UNNEST_{sub(S)}(s)$ and $S_0$ is a flat scheme with $C(S_0) = att(S)$. Then:

$s = NEST_{P(S)}(s_0)$ if and only if

$s_i$ satisfies the FD $(C(S_i)-Y^*_i) \dashrightarrow Y^*_i$

for $1 \leq i \leq n$.

Proof. The proof will be by induction on $k = |sub(S)|$, the number of subschemes of S.

Basis: If $k = 0$, then s is a flat relation and the theorem is vacuously true.

Induction Step: Let s be a structure over S such that $|sub(S)| = k+1$ and let $P(S) = (Y_1,\ldots,Y_k,Y_{k+1})$ be a nesting sequence for S. Then the definitions of $s_{k+1} = s$, $s_k$, ..., $s_0$ are given above. Now let $t = s_k$ and let the instances $t_k$, $t_{k-1}$, ..., $t_0$ and the schemes $T_k$, $T_{k-1}$, ..., $T_0$ be defined by applying the same definitions to t and the truncated nesting sequence $Q(S_k) = (Y_1,\ldots,Y_k)$. Clearly, $t_j = s_j$ and $T_j = S_j$ for $0 \leq j \leq k$. Since $T_k$ has k subschemes, we may apply the induction hypothesis; hence we have:

$t_k = NEST_{Q(T_k)}(t_0)$ if and only if

$t_i$ satisfies the FD $(C(T_i)-Y^*_i) \dashrightarrow Y^*_i$

for $1 \leq i \leq k$.

Restating this gives:

$s_k = NEST_{Q(S_k)}(s_0)$ if and only if

$s_i$ satisfies the FD $(C(S_i)-Y^*_i) \dashrightarrow Y^*_i$

for $1 \leq i \leq k$.

By a result in [TF] (cf. also [FSTV]) characterizing when $NEST_Y(UNNEST_{Y^*}(s)) = s$ we have:

$s_{k+1} = NEST_{Y_{k+1}}(s_k)$ if and only if

$s_{k+1}$ satisfies the FD $(C(S_{k+1})-Y^*_{k+1}) \dashrightarrow Y^*_{k+1}$.

Combining these two equivalences yields:

$s = NEST_{P(S)}(s_0)$ if and only if the k+1 FDs hold. This completes the induction step; hence the theorem is established. ■

The FDs in Theorem 1 track some of the semantic information associated with the order of nesting the data. One may also study different views of the same data in this manner (cf. [V]).

Theorem 1 implies that the structure $s_1 = \text{UNNEST}_{sub(S)-int(M)}(s)$ satisfies the FD $(att(S)-att(M)) \dashrightarrow M$ where $M = Y_1^*$. The converse is true also:

**Lemma 1.** For any $M \in frn(S)$, if the FD $(att(S)-att(M)) \dashrightarrow M$ is satisfied by $\text{UNNEST}_{sub(S)-int(M)}(s)$, then there exists a nesting sequence P(S) with C(M) as its <u>first</u> member such that $\text{NEST}_{P(S)}(r) = s$, where $r = \text{UNNEST}_{sub(S)}(s)$ is the fully unnested form of s.

**Proof.** One assumes that of the nesting sequences which produce s from r, k is the earliest position in which the $\text{NEST}_M$ operation occurs. Theorem 1 can then be invoked to show that k = 1. We omit details. ■

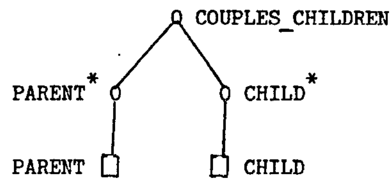
Theorem 1 and Lemma 1 can be used to produce a polynomial-time recognition algorithm for membership in NR(S).

**Algorithm NR.** Let s be a structure over scheme S.
<u>if</u> frn(S) = {R(S)},
<u>then</u> s is a relation, hence s ∈ NR(S)
<u>else</u>
  find a node M ∈ frn(S) such that
  $\text{UNNEST}_{sub(S)-int(M)}(s)$ satisfies the FD
  (att(S)-att(M)) --> M;
  <u>if</u> such an M exists,
  <u>then</u> s ∈ NR(S) if and only if s ∈ NR(S')
    where S' is the same scheme as S but with
    M interpreted as an attribute rather than
    a subscheme of S'
  <u>else</u> s is not in NR(S)
<u>end</u> NR

Algorithm NR runs in polynomial time with respect to the size of the fully unnested relation associated with s. We cannot claim polynomial time with respect to the size of the structure s since the unnested relation may have size exponential in the size of s. One can, however, keep the space complexity polynomial in the size of s by choosing appropriate data structures.

We now consider the permutable nested relations. The following example shows that PNR is properly contained in NR.

**Example 3.** Consider the structure $s_3$ over scheme $S_3$:




Scheme $S_3$

| PARENT | CHILD |
|---|---|
| $\{p_1,p_2\}$ | $\{c_1,c_2\}$ |
| $\{p_1,p_3\}$ | $\{c_3\}$ |
| $\{p_4,p_5\}$ | $\{c_4\}$ |

Instance $s_3$

Let $s_3' = \text{UNNEST}_{\{PARENT^*,CHILD^*\}}(s_3)$. The reader can verify that $s_3 = \text{NEST}_{(PARENT,CHILD)}(s_3')$, which implies that $s_3 \in NR(S_3)$. However, $s_3 \neq \text{NEST}_{(CHILD,PARENT)}(s_3')$; hence $s_3$ is not a member of $PNR(S_3)$.

| PARENT | CHILD |
|---|---|
| $p_1$ | $c_1$ |
| $p_1$ | $c_2$ |
| $p_2$ | $c_1$ |
| $p_2$ | $c_2$ |
| $p_1$ | $c_3$ |
| $p_3$ | $c_3$ |
| $p_4$ | $c_4$ |
| $p_5$ | $c_4$ |

Instance $s_3'$

In order to characterize when a structure s with scheme S belongs to PNR(S) we need to introduce the notion of a generalized FD. Let $v_1$, $v_2$ be tuples of s and let $M \in H(S)$. We say that $v_1$ and $v_2$ <u>overlap</u> on M, denoted $v_1[M]$ ovp $v_2[M]$ if and only if either:

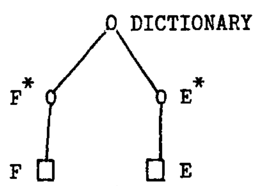
1. $v_1[M] \cap v_2[M] \neq \emptyset$, or
2. there exist tuples $t_1 \in v_1[M]$ and $t_2 \in v_2[M]$ such that $t_1[A(M)] = t_2[A(M)]$, and for all $N \in H(M)$, $t_1[N]$ ovp $t_2[N]$.

Let s be a structure over scheme S and let $V,Z \subseteq C(S)$ and $W \subseteq H(S)$. Then s satisfies the <u>generalized functional dependency</u> (GFD) V<W> --> Z if and only if for any two tuples $t_1$, $t_2 \in s$, whenever $t_1[V] = t_2[V]$ and for all $M \in W$ $t_1[M]$ ovp $t_2[M]$, we have $t_1[Z] = t_2[Z]$.

When $W = \emptyset$, a GFD is simply an ordinary FD. In the one-level case a GFD becomes a "SFD" as given in [FV3].

To get an intuitive view of permutable nested relations and GFDs, consider the following example (somewhat simplified) of Zaniolo and Melkanoff [ZM] (also studied in [BK]).

**Example 4.** Suppose there is a dictionary $s_4$ over scheme $S_4$ of corresponding technical terms in two different languages, French (F) and English (E). Assume that an object might be described by more than one word in each language. No word, however, describes more than one object.




Scheme $S_4$

| F | E |
|---|---|
| $\{f_1\}$ | $\{e_1,e_2\}$ |
| $\{f_2,f_3\}$ | $\{e_3,e_4\}$ |
| $\{f_4\}$ | $\{e_5\}$ |

Instance $s_4$

Let $s_4' = \text{UNNEST}_{\{F^*,E^*\}}(s_4)$. The reader can verify that $s_4 = \text{NEST}_{(F,E)}(s_4') = \text{NEST}_{(E,F)}(s_4')$, and therefore $s_4 \in \text{PNR}(S_4)$.

| F | E |
|---|---|
| $f_1$ | $e_1$ |
| $f_1$ | $e_2$ |
| $f_2$ | $e_3$ |
| $f_2$ | $e_4$ |
| $f_3$ | $e_3$ |
| $f_3$ | $e_4$ |
| $f_4$ | $e_5$ |

Instance $s_4'$

If we compare the structures in Examples 3 and 4 we notice a subtle difference. In $s_3$, the value $p_1$ appears in the $\text{PARENT}^*$ component of the first ($t_1$) and second ($t_2$) tuples, i.e., $t_1[\text{PARENT}^*] \cap t_2[\text{PARENT}^*] \neq \emptyset$. On the other hand, in $s_4$ the analogous set intersections are empty. It turns out that a recursive extension of this intersection property, namely the definition of a GFD, is the crucial difference between nested relations and permutable nested relations. As can be verified, $s_4$ satisfies the GFDs $\langle E^*\rangle \text{ --> } F^*$ and $\langle F^*\rangle \text{ --> } E^*$. The structure $s_3$, however, violates the GFD $\langle \text{PARENT}^*\rangle \text{ --> CHILD}^*$ although it satisfies the ordinary FD $\text{PARENT}^* \text{ --> CHILD}^*$ and the GFD $\langle \text{CHILD}^*\rangle \text{ --> PARENT}^*$.

In the next theorem we can characterize the class PNR(S) in terms of satisfaction of FDs in a partially unnested version of that structure. Furthermore, we see the correspondence between satisfaction of these FDs and satisfaction of GFDs in the original structure.

**Theorem 2.** Let s be a structure over scheme S. Then the following statements are equivalent:

1. $s \in \text{PNR}(S)$;
2. for all $M \in \text{sub}(S)$, $\text{UNNEST}_{\text{sub}(S)-\text{int}(M)}(s)$ satisfies the FD $(\text{att}(S)-\text{att}(M)) \text{ --> } M$;
3. for all $M \in H(S)$,
   a. s satisfies the GFD $A(S)\langle H(S)-M\rangle \text{ --> } M$ and
   b. for all tuples $t \in s$, $t[M] \in \text{PNR}(T(M))$.

**Sketch of Proof.** The implication (1) => (2) follows directly from Theorem 1 by choosing, for each $M \in \text{sub}(S)$, a nesting sequence which nests up to M before performing any other nests. The implication (2) => (1) involves consideration of the effect of unnesting on FDs (cf. [FSTV]). The proof of the equivalence of (2) and (3) is quite technical and is omitted. ∎

Part 3 of Theorem 2 permits the construction of an algorithm to check membership in PNR(S) by working only with the given structure. However, Part 2 immediately leads to an algorithm which is easier to understand, i.e., for each $M \in \text{sub}(S)$ perform the appropriate unnesting and check for satisfaction of the related FD. Both algorithms run in polynomial time with respect to the size of the fully unnested relation associated with s. The comments with respect to Algorithm NR also apply here.

It is obvious that NL is closed under the UNNEST operator, and Thomas showed that NR is not closed under UNNEST [Tho]. The latter fact may also be verified by considering the nested relation $s_3$ of Example 3 above. It can easily be shown by using Theorem 1 that the structure $\text{UNNEST}_{\text{PARENT}^*}(s_3)$ is not a nested relation. The class PNR is dealt with in the next theorem.

Theorem 3. The class PNR is the largest subclass of the nested relations which is closed under the UNNEST operator.

Proof. We first show that PNR is closed under UNNEST. Suppose s ∈ PNR, then s ∈ PNR(S), where S is the scheme of s. Let $Y^* \in H(S)$, let s' = $UNNEST_{Y^*}(s)$, and let S' be the scheme of s'. We need to show s' ∈ PNR(S'). For any nesting sequence Q(S') for S', s = $NEST_Y(NEST_{Q(S')}(r))$, where r = $UNNEST_{sub(S)}(s)$, since (Q(S'), Y) is a nesting sequence for S and s ∈ PNR(S). Applying $UNNEST_{Y^*}$ to both sides of the equation yields s' = $NEST_{Q(S')}(r)$ as desired.

Now suppose some class B ⊂ NR is closed under UNNEST. We have to show that B ⊂ PNR. Let s ∈ B and let S be its scheme. We need to prove that s ∈ PNR(S). We will use induction on |sub(S)|.

Induction Hypothesis: For any structure s over scheme S such that $0 \leq |sub(S)| \leq k$, if s ∈ B then s ∈ PNR.

Basis: If k = 0, then s is a relation and s ∈ PNR regardless of whether s ∈ B.

Induction Step: Let s ∈ B and assume |sub(S)| = k+1. Since B is closed under unnesting, for each $Y^* \in H(S)$ $UNNEST_{Y^*}(s) \in B$. Therefore, by the induction hypothesis $UNNEST_{Y^*}(s) \in PNR$. We now show that $NEST_Y(UNNEST_{Y^*}(s)) = s$. Since s ∈ B and B ⊂ NR, we know that there exists a nesting sequence for S, i.e., some P(S) = $(Y_1,\ldots,Y_n)$, such that s = $NEST_{P(S)}(UNNEST_{sub(S)}(s))$. Then Y must appear in P(S) since $Y^* \in H(S)$, i.e., Y = $Y_m$ for some $1 \leq m \leq n$. By Theorem 1 we know that $s_m = UNNEST_{\{Y_n^*,\ldots,Y_{m+1}^*\}}(s)$ satisfies the FD $(C(S_m)-Y^*) \rightarrow Y^*$, where $S_m$ is the scheme of $s_m$. It follows from a result in [FSTV] (regarding nesting on a subset of the left hand side of a FD) and the fact that $Y^* \in H(S)$ (hence $Y^*$ does not participate in later nesting) that s satisfies the FD $(C(S)-Y^*) \rightarrow Y^*$. From this we conclude that s = $NEST_Y(UNNEST_{Y^*}(s))$, again using a lemma of [TF] as in the proof of Theorem 1. Recapitulating, we have for each $Y^* \in H(S)$, $UNNEST_{Y^*}(s) \in PNR$ and s = $NEST_Y(UNNEST_{Y^*}(s))$. Since any nesting sequence for S must end with a NEST operation on a set Y such that $Y^* \in H(S)$, it follows that s ∈ PNR(S); hence s ∈ PNR. ∎

## 4. Hierarchical Structures and Nesting.

Since many real-world data applications are hierarchical in nature, the class of hierarchical structures has been studied intensively by several researchers (cf. [AB,OY,RKS]). A hierarchical structure has the property that for the structure and each of its substructures, the flat attributes of the associated scheme form a key for that structure or substructure. In order to study hierarchical structures in the context of nested relations, we introduce the notion of the hierarchical nest operator HNEST.

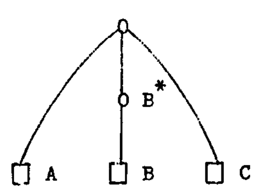
Let s be s structure over scheme S and let $X \subseteq A(S)$, $Y \subseteq C(S)$ and $X \cap Y = \emptyset$. Then: $HNEST_{Y:X}(s) = NEST_Y(\Pi_{XY}(s) \,|X|\, \Pi_{XZ}(s))$, where Z = C(S)-XY. Performing an HNEST operation is equivalent to enforcing the MVD X ->-> Y|Z, with X, Y, Z defined as above, thus producing a new instance s' with scheme S, then doing $NEST_Y(s')$.

Example 5. Let $s_5$ be the flat relation:

| A | B | C |
|---|---|---|
| a | $b_1$ | $c_1$ |
| a | $b_2$ | $c_2$ |

Instance $s_5$

The structure $s_5' = HNEST_{B:A}(s_5)$ over scheme $S_5'$ is shown below:




Scheme $S_5'$

| A | B | C |
|---|---|---|
| a | $\{b_1,b_2\}$ | $c_1$ |
| a | $\{b_1,b_2\}$ | $c_2$ |

Instance $s_5'$

We can make some observations about the properties of $HNEST_{Y:X}(s)$:

1. For any X, $HNEST_{Y:X}(s)$ and $NEST_Y(s)$ have the same scheme.

2. $HNEST_{Y:X}(s)$ satisfies the FD $X \rightarrow Y^*$ since the MVD X ->-> Y is enforced before the nest is performed (cf. [Mak,FSTV]).

3. $UNNEST_{Y^*}(HNEST_{Y:X}(s)) = s$ if and only if s satisfies the MVD X ->-> Y.

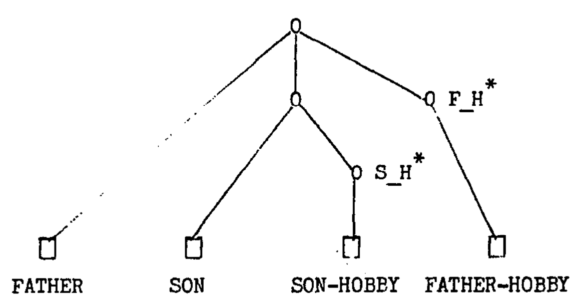
In order to define hierarchical structures we need some additional notions. Let S be a scheme and let M ∈ int(S). Let anc(M) be the set of proper ancestors (excluding M) of M in S. We define K(M) to be the union of A(N) for all N ∈ anc(M). Now we extend the use of a nesting sequence to permit a sequence of HNEST operations, i.e., if P(S) is a nesting sequence for S, and r is a relation over att(S), then $HNEST_{P(S)}(r) = HNEST_{Y_n:X_n}(HNEST_{Y_{n-1}:X_{n-1}} \ldots (HNEST_{Y_1:X_1}(r))\ldots)$, where for $1 \leq i \leq n$, $X_i = K(Y_i^*)$.

A structure s over scheme S is a <u>hierarchical structure</u> over S if and only if there exists a nesting sequence P(S) for S and a relation r over att(S) such that $s = HNEST_{P(S)}(r)$.

Example 6. The structure $s_6$ over the scheme $S_6$ below is an example of a hierarchical structure. It can be created by the application of three HNEST operations to the appropriate flat relation:

1. $HNEST_{SON\text{-}HOBBY:FATHER,SON}$
2. $HNEST_{SON,S\_H^*:FATHER}$
3. $HNEST_{FATHER\_HOBBY:FATHER}$

Note that the third HNEST could be performed at any time but the first must precede the second.




Scheme $S_6$

| FATHER | SON | SON-HOBBY | FATHER-HOBBY |
|---|---|---|---|
| $f_1$ | $s_1$ | $\{h_1,h_2\}$ | $\{h_2,h_4\}$ |
| | $s_2$ | $\{h_3\}$ | |
| $f_2$ | $s_3$ | $\{h_1,h_3\}$ | $\{h_1\}$ |
| | $s_4$ | $\{h_5\}$ | |

Instance $s_6$

We denote the class of hierarchical structures over scheme S by HNR(S) and the class of all hierarchical structures by HNR.

Theorem 4. Let s be a structure over scheme S. Then the following statements are equivalent:

1. s ∈ HNR(S);
2. for all M ∈ sub(S), $UNNEST_{sub(S)-int(M)}(s)$ satisfies the FD K(M) --> M, where K(M) is defined as above;
3. for all M ∈ H(S),
   a. s satisfies the FD A(S) --> M (hence A(S) is a key for s by the union rule for FDs ), and
   b. for all tuples t ∈ s, t[M] ∈ HNR(T(M)).

Discussion. This theorem is the analogue for HNR(S) of Theorem 2, which characterizes PNR(S). The proof is similar in approach to that of Theorem 2 and is omitted. ∎

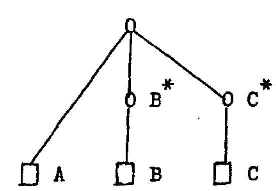
The second item in Theorem 4 states that the concatenation of the flat attributes along the path from the root node to any node forms a key for the corresponding unnested structure. The third item corresponds to the definition for hierarchical structures given in [RKS]. It may also be used to construct a polynomial-time algorithm to determine membership in HNR(S).

Corollary 1. The class HNR is contained in PNR.

Proof. Follows immediately from Theorems 2 and 4 by a simple induction on |sub(S)|. ∎

The fact that the containment is proper is easily shown by the following example:

Example 7. The structure $s_7$ is clearly in PNR but fails to be in HNR because the FD A --> $B^*$ is violated.




Scheme $S_7$

| A | B | C |
|---|---|---|
| a | $\{b_1\}$ | $\{c_1\}$ |
| a | $\{b_2\}$ | $\{c_2\}$ |

Instance $s_7$

The class HNR is closed under UNNEST. This was first shown by Roth, Korth and Silberschatz [RKS]. The proof can be simplified using the HNEST characterization given above.

## 5. Associated Flat Relations.

The previous theorems have been "top down", i.e., by analyzing a structure some properties of the structure have been determined. One may also consider a "bottom up" approach by focussing attention on the flat relations associated with a class of structures. Recall that any structure unnests to a unique relation, but relations may or may not renest to the same structure.

The class PNR(S) induces a subclass PR(S) of the flat relations which are obtainable by unnesting structures in PNR(S), i.e., a relation $r \in PR(S)$ if for some $s \in PNR(S)$, $r = UNNEST_{sub(S)}(s)$. Hence, r may be renested to s using any legal nesting sequence. We now characterize PR(S).

Theorem 5. Let S be a scheme and let r be a relation over att(S). Then the following statements are equivalent:

1. $r \in PR(S)$;
2. r satisfies the WMVDs
   ((att(S)-att(M))-att(N)) -w-> att(M)
   for each pair M, N of incomparable interior nodes of S;
3. r satisfies the WMVDs
   ((att(S)-att(M))-att(N)) -w-> att(M)
   for each pair M, N of interior nodes of S which are siblings.

The relation $s'_4$ in Example 4 illustrates this theorem. In this simple case items 2 and 3 above both state that the WMVDs $\emptyset$ -w-> E and $\emptyset$ -w-> F must hold in $s'_4$.

We now define the class HR(S) of hierarchically organized relations (over scheme S), first studied by Delobel and Lien [Del,Lie]. We say a relation $r \in HR(S)$ if for some $s \in HNR(S)$, $r = UNNEST_{sub(S)}(s)$. We claim that our definition is equivalent to the one used by Delobel.

Theorem 6. Let S be a scheme and let r be a relation over att(S). Then

1. $r \in HR(S)$, if and only if
2. r satisfies the MVDs
   $(K(M) \cup A(M))$ ->-> att(N) for all $M \in int(S)$ and all $N \in H(M)$, where K(M) is defined as before.

## 6. Open Problems.

We conclude this paper by mentioning a few open problems.

1. Find a characterization of NL. What is the time complexity of a recognition problem for NL? Is it at least demonstrably in NP?

2. In this paper we did not allow null values or empty sets. How does the theory change if we allow these (cf. [AB])?

3. Are there deeper semantics for GFDs, i.e., can they also say something about the meaning of a database or do they only reflect the way the data has been restructured (cf. [FV3,V])?

4. Do WMVDs have any role other than indicating when NEST operations commute? Are they of any interest in a database design algorithm (cf. [FV1,V])?

## References


[AB] S. Abiteboul, N. Bidoit, "Non First Normal Form Relations to Represent Hierarchically Organized Data", Proc. Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1984, 191-200.

[BRS] F. Bancilhon, P. Richard, M. Scholl, "On Line Processing of Compacted Relations", Proc. 8th Int'l Conf. on Very Large Data Bases, 1982, 263-269.

[BK] C. Beeri, M. Kifer, "Comprehensive Approach to the Design of Relational Database Schemes", Proc. 10th Int'l Conf. on Very Large Data Bases, 1984, 196-207.

[CT] J. Clifford, A. Tansel, "On an Algebra For Historical Relational Databases: Two Views", Proc. ACM SIGMOD Int'l Conf. on Management of Data, 1985, 247-265.

[Cod] E.F. Codd, "A Relational Model for Large Shared Data Banks", Comm. ACM 13,6 (June 1970), 377-387.

[Del] C. Delobel, "Normalization and Hierarchical Dependencies in the Relational Data Model", ACM Trans. on Database Systems 3,3 (September 1978), 201-222.

[FSTV] P.C. Fischer, L.V. Saxton, S.J. Thomas, D. Van Gucht, "Interactions between Dependencies and Nested Relational Structures", J. Computer System Sciences 31 (1985), to appear.

[FT] P.C. Fischer, S.J. Thomas, "Operators for Non-First-Normal Form Relations", Proc. IEEE Computer Software and Applications Conference, 1983, 464-475.

[FV1] P.C. Fischer, D. Van Gucht, "Weak Multivalued Dependencies", Proc. Third ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1984, 266-274.

[FV2] P.C. Fischer, D. Van Gucht, "Structure of Relations Satisfying Certain Families of Dependencies", in Proc. 2nd Symposium on Theoretical Aspects of Computer Science, K. Mehlhorn, Ed., Springer-Verlag, Berlin, 1985, 132-142.

[FV3] P.C. Fischer, D. Van Gucht, "Determining When A Structure is a Nested Relation", Proc. 11th Int'l Conf. on Very Large Data Bases, 1985, 171-180.

[FA] J. Freitag, H.J. Appelrath, "Modelling IR by S-NF$^2$ Relations", Computing 85: A Broad Perspective of Current Developments, G. Bucci and G. Valle (eds.), Elsevier Science Publishers, B.V. (North-Holland) 1985.

[Gon] G. Gonnet, "Unstructured Data Bases", Proc. Second ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1983, 117-124.

[HY] R. Hull, C.K. Yap, "The Format Model", Journal of the ACM 31,3 (July 1984), 86-96.

[JS] G. Jaeschke, H.-J. Schek, "Remarks on the Algebra on Non First Normal Form Relations", Proc. ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1982, 124-138.

[KTT] Y. Kambayashi, K. Tanaka, K. Takeda, "Synthesis of Unnormalized Relations Incorporating More Meaning", Information Sciences 29 (1983), 201-247.

[KV] G.M. Kuper, M.Y. Vardi, "A New Approach to Database Logic", Proc. Third ACM SIGACT-SIGMOD Symposium on the Principles of Database Systems, 1984, 86-96.

[Lie] Y. E. Lien, "Hierarchial Schemata for Relational Databases", ACM Trans. on Database Systems 6,1 (March 1981), 48-69.

[Mac] I.A. Macleod,"A Model for Integrated Information Systems", Proc. Ninth Int'l Conf. on Very Large Data Bases, 1983, 280-289.

[Mak] A. Makinouchi, "A Consideration of Normal Form of Not-Necessarily-Normalized Relations in the Relational Data Model", Proc. 5th Int'l Conf. on Very Large Data Bases, 1977, 447-453.

[OMO] Z. Ozsoyoglu, V. Matos, Z.M. Ozsoyoglu, "Extending Relational Albebra and Relational Calculus for Set-Valued Attributes and Aggregate Functions", Tech. Report, Case Western Reserve University, 1983.

[OO] Z.M. Ozsoyoglu, G. Ozsoyoglu, "An Extension of Relational Algebra for Summary Tables", Proc. Statistical Database Workshop, 1983, 202-211.

[OY] Z.M. Ozsoyoglu, L.Y. Yuan, "A Normal Form for Nested Relations", Proc. Fourth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems, 1985, 251-260.

[RKS] M.A. Roth, H.F. Korth, A. Silberschatz, "Theory of Non-First-Normal-Form Relational Databases", Tech. Report TR-84-36, University of Texas at Austin, 1984.

[Sch] H.J. Schek, "Towards a Basic Relational NF$^2$ Algebra Processor", Proc. of the International Conference on Foundations of Data Organization, Kyoto, Japan, 1985, 173-182.

[SP] H.-J. Schek, P. Pistor, "Data Structures for an Integrated Data Base Management and Information Retrieval System", Proc. 8th Int'l Conf. on Very Large Data Bases, Mexico, 1982, 197-207.

[SS] H.J. Schek, M.H. Scholl, "An Algebra for the Relational Model with Relation-Valued Attributes", TR DVSI-1984-T1, Technical University of Darmstadt, West Germany, 1984.

[Tho] S.J. Thomas, "A Non-First-Normal Form Relational Database Model", Ph.D. Dissertation, Vanderbilt University, 1983.

[TF] S.J. Thomas, P.C. Fischer, "Nested Relational Structures", Theory of Databases, P.C. Kanellakis, Ed., JAI Press, 1985, to appear.

[V] D. Van Gucht, "Theory of Unnormalized Relational Structures", Ph.D. Dissertation, Vanderbilt University, 1985.

[ZM] C. Zaniolo, M.A. Melkanoff, "On the Design of Relational Database Schemata", ACM Trans. on Database Systems 6,1 (March 1981), 1-47.