ENTITY-RELATIONSHIP CONSISTENCY
FOR RELATIONAL SCHEMAS

BY

J. Makowsky, V.M. Markowitz and N. Rotics

Technical Report #392
December 1985
ENTITY-RELATIONSHIP CONSISTENCY
FOR RELATIONAL SCHEMAS

J. Makowsky, V. M. Markowitz and N. Rotics

Technion - Israel Institute of Technology

Abstract.

We investigate the significance of requiring from a relational database schema to comply with an entity-relationship structure. Relational database schemas here consist of a traditional relational schema together with key and inclusion dependencies. Such a schema is said to be entity-relationship (ER) consistent, if it is possible to translate it into an entity-relationship schema. An algorithm is presented which decides if a schema is ER-consistent and its complexity is discussed. ER-consistency allows us to carry over precisely all the advantages of the ER-oriented design principles to the relational model and makes possible the direct use of ER oriented query and update languages within the relational model. ER-consistency carries out on a higher level the data independence objective of the relational model by making independent the information structure from its logical data representation. Moreover, ER-consistency expresses information structure normalization just as relational normal forms represent data representation normalization. We propose an Entity-Relationship Normal Form for ER-consistent relational schemas and present the corresponding normalization procedure.

Introduction.

One of the main objectives of the relational model is communicability [4], which means offering the user a data model which is easy to understand, use and communicate about. Regretfully, this objective is only partially fulfilled by the relational model since it conceals much of the semantic structure of the real world. Consequently, semantically oriented models have been developed, with the entity-relationship model (ERM) [2] emerging as the most popular. ERM reflects a natural, although limited, view of the world: entities are qualified by their attributes and interactions between entities are expressed by relationships. Codd pointed out that the semantic data models in general, and ERM in particular, lack both a well defined instance level and, therefore, a well defined data manipulation language. The ERM has been mostly accepted as an early stage data base design tool. Once the design stage ends, the entity-relationship schema, represented by an entity-relationship diagram (ERD) is translated into a relational scheme and its role is therewith ended (cf. [14]).
**ER consistency for relational schemas**

We propose to take this translation more seriously and try to capture all the information about the database design contained in the original ERS. This leads us to relational schemas together with key and inclusion dependencies. Such relational schemas (obtained by this translation) are said to be trivially entity-relationship (ER) consistent. Conversely, a relational schema translatable to an ERD is said to be ER-consistent. For ER-consistent relational schemas we propose an *Entity-Relationship Normal Form*.

ER-consistency offers several advantages:

(i) it encourages an ER oriented design which is, as mentioned above, easy and natural;

(ii) it offers a clear cut separation of structural and nonstructural dependencies;

(iii) it makes possible the direct use of ER oriented query languages in relational environments.

We propose ER-consistency as the schema design principle replacing the relational normalization, as recommended by [1]: "the goal of the theory is not to tell us how to normalize but rather to develop rules and directives for the proper design and use of database schemes".

Our work emerged from, and is a continuation of, [10]. Prior to [10], ER to relational structure mappings were presented in numerous papers, starting with [2]. The ERM has been mostly accepted as an early stage database design tool. Accordingly, these mappings are informal and do not attempt to characterize anyhow the relational translates of ERD's. The reverse mapping has been investigated in [7], in a rather informal way. Unlike [7], [10] gives precedence to the structural information provided by the keys over the inclusion dependencies. In [10] the generalization abstraction is not considered, and the relational attributes naming is assumed to be well-behaved, not arbitrary, as it really is. Unlike [10] our approach is graph oriented, allowing both a graph characterization of ERDs, accurate and concise, and to relate ERD digraphs to relational based graphs, such as the inclusion dependency digraph. Moreover, we attempt not to rely on the above mentioned well behaviour of
attribute names.

The first section of the paper reviews concepts of the relational model necessary to our presentation. Next, in section 2, we give the definition of ER diagrams (schemas). In section 3 we present the mapping of ER diagrams to relational schemas and the characterization of these relational schemas. In section 4 we propose a reverse mapping from relational schemas to ER diagrams and investigate when and whether this mapping is possible. In section 5 we discuss the significance of ER consistency and its relation to relational normal forms. We propose an Entity-Relationship Normal Form for ER-consistent relational schemas and present the corresponding normalization procedure. We close the paper by summarizing the results, drawing some conclusions and outlining directions for further research.

1. The Relational Model.

A relational schema (RS) is a pair \((R, D)\) where \(R\) is a set of relational schemes, \(R = (R_1, ..., R_k)\), and \(D\) is a set of dependencies over \(R\). A relational scheme is a named set of attributes, \(R_i(A_i); A_i\) is also called the arity of \(R_i\).

On the semantic level, every attribute is assigned a domain. A database state of \(R\) is defined as \(r = <D_1, ..., D_m, r_1 ... r_k>\), where \(r_i\) is assigned a subset of the cartesian product of the domains corresponding to its attributes.

In the present paper we are concerned with two kinds of dependencies, one inner relational, and one inter relational.

A functional dependency (FD) over \(R_i\), is a statement of the form \(X \rightarrow Y\), where both \(X\) and \(Y\) are subsets of \(A_i\); an FD \(X \rightarrow Y\) is valid in a state \(r\) iff for any two tuples of \(r_i\), \(t\) and \(t'\), if \(t[X] = t'[X]\) then \(t[Y] = t'[Y]\).

A key dependency (KD) over \(R_i\), is the functional dependency \(K_i \rightarrow A_i\), where \(K_i\) is a subset of \(A_i\) and there is no subset of \(K_i\) with this property; \(K_i\) is called key. The set of KIDs associated with some relational schema is denoted \(K\).
An inclusion dependency (IND) over $R_i$ is a statement of the form $R_i[X] \subseteq R_j[Y]$, where $X$ and $Y$ are subsets of $A_i$ and $A_j$, respectively, and $|X| = |Y|$. An IND $R_i[X] \subseteq R_j[Y]$, is valid in a state $\tau$, iff $\tau_i[X]$ is a subset of $\tau_j[Y]$. The set of INDs associated with some relational schema is denoted $I$.

Provided the domains are sets of interpreted values which are restricted conceptually and operationally, two attributes are said to be compatible if they are associated with a same domain. Compatible attributes with different names express, generally, different roles played by a domain within a database. Two sets of attributes, $A_i$ and $A_j$, are said to be compatible iff there is a one-to-one correspondence of attributes between $A_i$ and $A_j$. The attribute compatibility graph is the graph $G_{AC} = (V, E)$, where $V = A$ and $(A_{i_k}, A_{j_m})$ is an edge of $E$ iff $A_{i_k}$ and $A_{j_m}$ are compatible; then, two attributes are said to be compatible if they belong to a same connected component of the associated attribute compatibility graph.

If the domains are sets of uninterpreted values, an alternative way of defining attribute compatibility is either by explicitly specifying the attribute compatibility graph or by using the set of specified INDs as follows: given a set of INDs, $I$, over $R$, the attribute compatibility graph is the digraph $G_{AC} = (V, E)$, where $V = A$ and $(A_{i_k}, A_{j_m})$ is an edge of $E$ iff there is some IND in $I$, $R_i[X_1 \cdots X_m] \subseteq R_j[Y_1 \cdots Y_m]$ s.t. $A_{i_k} = X_k$ and $A_{j_m} = Y_k$.

Given a relation scheme $R_i$, a correlation key (CK) in $R_i$ is a subset of $A_i$, $CK_i$, which is a key in some relation, other than $R_i$. For some relation scheme $R_i$, $CK_i = \cup CK_i$.

Given a set of KDS, $K$, over $R$, the associated key correlation graph (KG) is a digraph $G_K = (V, E)$, where $V = R$ and $R_1 \rightarrow R_2$ is an edge of $E$ iff

(i) $K_1 = K_2$; or (ii) $K_2 \subseteq CK_1$ and there is no $R_3$ s.t. $K_2 \subseteq CK_3$ and $K_3 \subseteq CK_1$.

A set of INDs, $I$, is said to be cyclic if either $R_i[X] \subseteq R_i[Y]$ for $X$ and $Y$ different; or there are $R_1$, $R_n$ s.t. $R_1[X_1] \subseteq R_2[Y_2]$, $R_2[X_2] \subseteq R_3[Y_3]$, ..., $R_n[X_n] \subseteq R_1[Y_1]$.

An IND $R_i[X] \subseteq R_2[Y]$, is said to be key-based [11], if $Y = K_2$. 
ER consistency for relational schemas

A set of INDs, $I$, is said to be bounded [11], if whenever $R_1[X_1] \subseteq R_2[X_2]$, and $R_1[X_1] \subseteq R_3[X_3]$, then there is $R_4$ s.t. $R_2[X_2] \subseteq R_4[X_4]$, and $R_3[X_3] \subseteq R_4[X_4]$, are implied by $I$.

An inclusion dependency $R_1[X] \subseteq R_2[Y]$ is said to be typed [6], if $X = Y$.

Given a set of INDs, $I$, over $R$, the associated IND graph (IG) is the digraph $G_I = (V, E)$, where $V = R$ and $R_1 \rightarrow R_2$ is an edge of $E$ iff $R_1[X] \subseteq R_2[Y]$ is in $I$.

**Proposition 1.1** [11]: A set of INDs, $I$, is acyclic iff the associated IG digraph is a dag.

**Proposition 1.2** [6]: Given a set of typed INDs, $I$, every IND $R_i[X] \subseteq R_j[Y]$ is implied by $I$ iff either it is trivial, or $X = Y$ and there is a path from $R_i$ to $R_j$ in the associated IG digraph, corresponding to a sequence of INDs of $I$, $R_i[W] \subseteq \cdots \subseteq R_j[W]$, s.t. $X \subseteq W$.

**Proposition 1.3** [6]: Let $I$ and $K$ be sets of INDs and KDs, respectively, in a relation schema $(R, K, I)$; then $(I \cup K)^+ = I^+ \cup K^+$.

2. The Entity-Relationship Diagram.

An Entity-Relationship Schema, called the Entity-Relationship Diagram (ERD), is a finite labeled digraph $G_{ER} = (V, H)$, where

- $V$ is the disjoint union of four subsets of vertices, $S, A, E$ and $R$:
  
  (i) $S$ is the set of s-vertices; (ii) $A$ is the set of a-vertices;
  
  (iii) $E$ is the set of e-vertices; (iv) $R$ is the set of r-vertices;

  s-vertices and e-vertices are represented graphically by rectangles, a-vertices and r-vertices are represented graphically by circles and diamonds, respectively;

- $H$ is the set of directed edges, where an edge can be of one of the following forms:
  
  (i) $E_i \rightarrow A_j$; (ii) $R_i \rightarrow A_j$; (iii) $R_i \rightarrow E_j$; (iv) $E_i \rightarrow E_j$; (v) $A_k \rightarrow S_j$.

We denote by $Atr(X_i)$ the set $\{A_j | X_i \rightarrow A_j$ is an edge in $H\}$, and by $Ent(R_i)$ the set $\{E_j | R_i \rightarrow E_j$ is an edge in $H\}$. A directed path from $X_i$ to $X_j$ is denoted $X_i \rightarrow X_j$.
ER consistency for relational schemas

Intuitively, s-vertices, e-vertices, a-vertices and r-vertices represent value-sets, entity-sets, attributes of entity-sets or relationship-sets, and relationship-sets, respectively. An entity-set groups entities of a same type, where the entity-type is perceived as the sharing of a same set of attributes. A value-set is a special kind of entity-set, without attributes, and grouping atomic values of a certain type. A relationship represents the interaction between several entities, and relationships of the same type are grouped in a relationship-set. A relationship-set can have attributes, just like an entity-set. An attribute is associated with one or several value-sets and provides an interpretation for that combination of value-sets in the context of an entity-set or relationship-set. An entity-set in a relationship-set may have a role, expressing the function it plays in the relationship.

Every ERD vertex is labeled by the name of the associated value-set, entity-set, relationship-set, or attribute name; e-vertices, r-vertices and s-vertices are uniquely identified by their labels globally, while a-vertices are uniquely identified by their labels only locally, within the set of a-vertices connected to some e-vertex/r-vertex. A subset of the attributes associated with an entity-set $E_i$, $\text{Atr}(E_i)$, is specified as the entity-identifier, $\text{Id}(E_i)$.

ERD edges specify existence constraints:

$(X_i \rightarrow A_j)$ where $X_i$ is either an e-vertex or an r-vertex, express the fact that an attribute value is meaningful only as characterizing an entity or relationship;

$(E_i \rightarrow E_j)$ an entity may be contingent on the existence of other entities, through ISA or ID relationships:

- the ISA relationship expresses a subset relationship between two entity-sets; the corresponding edge is labeled by an ISA label and is called ISA-edge;

- the ID relationship expresses an identification relationship between an entity-set, called weak entity-set, which cannot be identified by its own attributes, but has to be identified by its relationship(s) with other entity-sets; the corresponding edge is labeled by an ID label and is called ID-edge;
A self-explanatory example of an ERD digraph is given in figure 1.

Cardinality constraints [13] are restrictions on the minimum and maximum number of entities from a given entity-set, that can be related, in the context of some relationship-set, to a specific combination of entities from other entity-sets. Every edge $R_i \rightarrow E_j$ is labeled as follows: if it corresponds to a maximum cardinality of one, it is labeled with 1, and is called one-labeled; if it corresponds to a maximum cardinality greater than one, it is labeled with $m, n, ...$, and is called many-labeled; we shall assume that at least one outgoing edge of every r-vertex is many labeled.

Generally value-sets are not represented in ERD digraphs and, alternatively, every edge $E_i \rightarrow A_j$ or $R_i \rightarrow A_j$, may be labeled by the combined type of the value-sets associated with $A_j$. In the present paper we omit the specification of value-sets for the sake of conciseness.

A self-explanatory example of an ERD digraph is given in figure 1.

The reduced ERD, RERD, is the digraph $G'_{ER} = (V', H')$, where $V' = E \cup R$, and

---

Fig. 1 ERD digraph example: identifiers are underlined.
A directed path in an RERD digraph is called an **ISA-path** if all the edges on the path are ISA-edges; a directed path in an RERD digraph is called an **ID-path** if at least one edge on the path is an ID-edge.

The entity-set, relationship-set and attribute compatibility, whose intuition is straightforward, have the following ERD based definitions:

(i) two attributes, \( A_i \) and \( A_j \), are said to be compatible iff the a-vertices representing them are connected to the same s-vertices, that is, are of the same type;

(ii) two entity-sets, \( E_i \) and \( E_j \), are said to be compatible iff there is some e-vertex \( E_k \) such that \( E_i \rightarrow ISA \rightarrow E_k \) and \( E_j \rightarrow ISA \rightarrow E_k \); and

(iii) two relationship-sets, \( R_i \) and \( R_j \), are said to be compatible iff there is a one-to-one correspondence of compatible entity-sets between \( \text{Ent}(R_i) \) and \( \text{Ent}(R_j) \).

ERD digraphs obey the following constraints (with the exception of (ER0) below, the constraints refer only to the *reduced ERD*):

(ER0) for any a-vertex of \( V \), \( A_i \), indegree\( (A_i) = 1 \);

(ER1) for any e-vertex of \( V \), \( E_i \),

(ER1.i) it cannot have both ISA and ID outgoing edges;

(ER1.ii) if \( E_i \) has at least one outgoing ID-edge, or outdegree\( (E_i) = 0 \), then id\( (E_i) \) must be not empty;

(ER1.iii) when \( E_i \) has outgoing ISA-edges, Id\( (E_i) \) must be empty;

(ER1.iv) there are not attributes connected to a same set of value-sets, that is of a same type, neither within an entity-identifier, nor in several different entity-identifiers;

(ER2) for any r-vertex of \( V \), \( R_i \), indegree\( (R_i) = 0 \) and outdegree\( (R_i) \geq 2 \);

(ER3) ERDs are connected dags;

(ER4) there are no parallel or unlabeled edges;
(ER5) two ISA paths having the same start e-vertex, must have the same end e-vertex;

(ER6) let $E_j$ be the end e-vertex of two ID(directed) paths having the same start e-vertex(r-vertex);

(ER6.i) every $X_i \rightarrow E_j$ last edge belonging to these paths, is labeled by the corresponding role of $E_j$; or

(ER6.ii) if roles are not used, two ID(directed) paths having the same start e-vertex(r-vertex), must not have the same end e-vertex.

Constraints (ER0) to (ER6) define proper ER structures. An attribute characterizes a single entity-set or relationship-set, therefore an a-vertex is connected by a single edge to a single e-vertex or r-vertex, as expressed by constraint (ER0). Note that directed cycles could consist only of ISA and ID edges. Constraint (ER3) above guarantees that such cycles do not exist; the meaning of this constraint is that an entity-set will neither be defined as depending on identification on itself, nor be defined as a proper subset of itself. Constraint (ER5) above guarantees entity-set compatibility.

We shall assume in the following that roles are not required, that is, two ID(directed) paths having the same start e-vertex(r-vertex), must not have the same end e-vertex (ER6.ii).


The following mapping is an extension of the known ER to relational schema mapping (cf. [14]).

Mapping ER Diagram into Relational Schemas Procedure ($T_s$).

Input: $G_{ER} = (V,H)$, an ERD digraph as defined above;

Output: the relational schema $(R,K,I)$ interpreting $G_{ER}$;

(0) initially $R,K,I$, are empty;

(1) prefix all the labels of a-vertices belonging to entity-identifiers by the label of the corresponding e-vertex; consequently, the labels of the a-vertices belonging to entity-identifiers are assured to be globally unique within the ERD digraph;
ER consistency for relational schemas

(2) compute, for every e-vertex and every r-vertex, the set of key a-vertices:

(2.i) for every e-vertex $E_i$, $\text{Key} (E_i) = \text{Id}(E_i) \cup \{ \text{Key}(E_j) | E_i \rightarrow E_j \}$;

(2.ii) for every r-vertex $R_i$, $\text{Key} (R_i) = \bigcup \{ \text{Key}(E_j) | R_i \rightarrow E_j, \text{is many-labeled} \}$;

*note*: constraint (ER3) guarantees the finiteness of this step;

(3) for every e-vertex/r-vertex $X_i$:

(3.i) define relation-scheme $R_i$;

(3.ii) $K_i := \text{Key}(X_i)$;

(3.2) $A_i := \text{Atr}(X_i) \cup K_i$;

(3.3) $R := R \cup R_i(A_i)$;

(3.4) $K := K \cup (K_i \rightarrow A_i)$;

(4) for every edge $X_i \rightarrow E_j$, where $X_i$ is an e-vertex/r-vertex and $E_j$ is an e-vertex:

$I := I \cup (R_i[K_i] \subseteq R_j[K_j])$, where relational schemes $R_i$ and $R_j$ correspond to $X_i$ and $E_j$ respectively;

(5) build the attribute compatibility graph, $G_{AC}$, following the ERD based definition of compatibility (see section 2);

*note*: only the ERD attribute compatibility has a relational correspondent, and accordingly, the entity-set and relationship-set compatibility is lost by the mapping.

An exemplification of the above mapping is given in figure 2, for the ERD digraph of figure 1, section 2.

Lemma 3.1

Let $(R, K, I)$ be the relational schema translate of the ERD digraph $G_{ER}$; the IG digraph $G_I$ associated with $(R, K, I)$, and the corresponding RERD digraph $G_{ER}'$, are isomorphic.

Proof:

(i) a vertex of $G_{ER}'$ is either an e-vertex or an r-vertex; by translation $T_a$ (step 3), to every such vertex corresponds exactly one relation scheme which, in $G_I$, is associated with exactly one vertex (definition of IG digraph);
ER consistency for relational schemas

relational-schemes (R - keys are underlined):

COUNTRY (COUNTRY.NAME)
CITY (CITY.NAME,COUNTRY.NAME)
STREET (STREET.NAME,CITY.NAME,COUNTRY.NAME)
PERSON (PERSON.ID,NAME)
EMPLOYEE (PERSON.ID,EMPN,SalARY)
LIVES (STREET.NAME,CITY.NAME,COUNTRY.NAME,PERSON.ID)

inclusion dependencies (I):

CITY[COUNTRY.NAME] ⊂ COUNTRY[COUNTRY.NAME]
STREET[CITY.NAME,COUNTRY.NAME] ⊂ CITY[CITY.NAME,COUNTRY.NAME]
LIVES[STREET.NAME,CITY.NAME,COUNTRY.NAME] ⊂ STREET[STREET.NAME,CITY.NAME,COUNTRY.NAME]
LIVES[PERSON.ID] ⊂ PERSON[PERSON.ID]
EMPLOYEE[PERSON.ID] ⊂ PERSON[PERSON.ID]

Fig.2 Translation of ERD digraph of Fig.1

(ii) let $X \rightarrow Y$ be an edge of $G_{ER}$, such that $X$ and $Y$ correspond, by translation $T_s$, to
relation schemes $R_i$ and $R_j$ respectively; also by $T_s$ (step 4), $X \rightarrow Y$ iff
$R_i[K_j] \subset R_j[K_j]$, which is associated in $G_f$ with the edge $R_i \rightarrow R_j$.

The following proposition provides the characterization of the relational scheme
translates of r-vertices and e-vertices.

**Proposition 3.2.**

Let $(R,K,I)$ be a relational schema translate of an ERD digraph $G_{ER}$. An IND of $I$, $R_i[K_j] \subset R_j[K_j]$, corresponds to an edge of $G_{ER}$ which is:

ID-labeled iff $K_i$ is not included in $CK_i$;

ISA-labeled iff $K_i = CK_i = K_j$;
many-labeled iff \( K_i \subseteq CK_i \), and \( K_j \subseteq K_i \);

one-labeled iff \( K_i \subseteq CK_i \), \( K_j \) not included in \( K_i \) and \( K_j \subseteq CK_i \).

The proof of the above proposition is based directly on the mapping \( T_s \). An illustration of the above proposition is given in figure 3.

**Proposition 3.3.**

Let \((R, K, I)\) be the relational schema translate of an ERD digraph as defined above. The corresponding IND set, \( I \), is typed, key-based, acyclic and bounded.

Proof:

(i) the translation \( T_s \) generates only key-based and typed INDs;

(ii) since the IG graph associated with the relational schema translate and the corresponding RERD digraph are isomorphic (lemma 3.1), the IG graph inherits the acyclicity property (ER3) of the respective ERD digraph; consequently, the IND set \( I \) is acyclic (Proposition 1.1);

---

<table>
<thead>
<tr>
<th>keys</th>
<th>correlation keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>S(B) B -</td>
</tr>
<tr>
<td>B[S]</td>
<td></td>
</tr>
<tr>
<td>ISA</td>
<td>S(B) B -</td>
</tr>
<tr>
<td>B[S]</td>
<td></td>
</tr>
<tr>
<td>ln</td>
<td>S(B) B -</td>
</tr>
<tr>
<td>T[S]</td>
<td>T(A) A -</td>
</tr>
</tbody>
</table>

Fig. 3 Exemplification for Proposition 3.2
ER consistency for relational schemas

(iii) suppose \( R_1[A] \subseteq R_2[A] \) and \( R_1[A] \subseteq R_3[A] \); by (ER0) and (ER6), it can be shown that \( R_1 \) can be the translate only of an e-vertex, and both INDs correspond to ISA edges; as above, the IG graph inherits the property (ER5) of the corresponding ERD digraph, which expresses the boundness property for the typed and key-based INDs we are considering.


Now we shall investigate the question of the range of the translation \( T_s \), and when and whether a reverse translation \( T_s^R \) is defined.

Let \((R,K,I)\) be the translate relational schema of ERD digraph \( G_{ER} \). We have seen above that the INDs generated by translation \( T_s \) are key-based, and that the corresponding IG digraph \( G_I \) and the source RERD digraph \( G'_{ER} \) are isomorphic. Moreover, for a translate relational schema, \( G_I \) is a subgraph of \( G_K \): both have the same set of vertices; and the set of edges of \( G_I \) is a subset of the set of edges of \( G_K \).

These observations suggest to base the reverse translation from relational schemas to ERDs, \( T_s^R \), on the structural information provided by the key set. Given a relational schema \((R,K,I)\), our procedure derives a candidate ERD digraph from \( G_K \), and verifies whether it is a proper ERD digraph.

Mapping Relational Schemas into ER Diagrams (\( T_s^R \)).

Input: relational schema \((R,K,I)\) and the associated attribute compatibility graph \( G_{AC} \), where \( I \) is acyclic and in any key of \( K \), there are no two compatible attributes.

Output: either an ERD digraph \( G_{ER} \), or a fail message;

(a) rename to a common name all the key attributes associated with vertices belonging to a same connected component of \( G_{AC} \);

(b) construct the digraph \( G_K \) associated with the input relational schema;

(c) derive candidate RERD \( G'_{ER} = (V'_{ER}, H'_{ER}) \), from \( G_K \):
ER consistency for relational schemas

(2.1) \( V'_{ER} := V_K \), and \( H'_{ER} := H_K \);

(2.2) label every edge in \( H'_{ER} \), \( X_i \rightarrow Y_j \), corresponding to edge \( R_i \rightarrow R_j \) in \( G_K \), as follows:

(ID-label) \( K_i \) not included in \( CK_i \);

(ISA-label) \( K_i = K_j = CK_i \);

(many-label) \( K_i \subseteq CK_i \), and \( K_j \subseteq K_i \);

(one-label) \( K_i \subseteq CK_i \), \( K_j \) not included in \( K_i \) and \( K_j \subseteq CK_i \).

(2.3) split \( V'_{ER} \) into two disjunctive sets of e-vertices and r-vertices, where vertex \( X \) is an r-vertex iff every outgoing edge is either many- or one-labeled;

(3) for every edge \( X_i \rightarrow X_j \) of \( G'_{ER} \), of one of the following three forms, corresponding to edge \( R_i \rightarrow R_j \) of \( G_K \), verify:

\((E_i-ID\rightarrow E_j)\) if \( R_i[K_j] \subseteq R_j[K_j] \) is not logically implied by \((K, l)\), then mark missing IND;

\((E_i-ISA\rightarrow E_j)\) if \( R_i[K_j] \subseteq R_j[K_j] \) is not logically implied by \((K, l)\), then:

if \( E_j - ISA \rightarrow E_i \) is in \( H'_{ER} \), then remove \( E_i - ISA \rightarrow E_j \) from \( G'_{ER} \);

if \( E_j - ISA \rightarrow E_i \) is not in \( H'_{ER} \), then mark missing IND;

\((R_i \rightarrow E_j)\) if \( R_i[K_j] \subseteq R_j[K_j] \) is not logically implied by \((K, l)\), then:

if there is \( E_k \) s.t. \( R_i \rightarrow E_k \) and \( E_j - ISA \rightarrow E_k \) are in \( H'_{ER} \), then remove \( R_i \rightarrow E_j \) from \( G'_{ER} \);

if there is no \( E_k \) s.t. \( R_i \rightarrow E_k \) and \( E_j - ISA \rightarrow E_k \) are in \( H'_{ER} \), then mark missing IND;

(4) remove extraneous edges from \( G'_{ER} \):

\((E_i-ISA\rightarrow E_j)\) remove \( E_i - ISA \rightarrow E_j \) from \( G'_{ER} \) whenever there is an ISA-path, other than \( E_i - ISA \rightarrow E_j \) from \( E_i \) to \( E_j \);

\((R_i \rightarrow E_j)\) remove \( R_i \rightarrow E_j \) from \( G'_{ER} \) whenever there is an e-vertex \( E_k \) s.t. \( R_i \rightarrow E_k \) and \( E_k - ISA \rightarrow E_j \) are in \( H'_{ER} \);
(5) complete RERD to a full ERD:

for every e-vertex $E_i$ corresponding to relation scheme $R_i(A_i)$:

$$\text{Atr}(E_i) = A_i - \{A_j | A_j \subseteq \text{Id}(E_j), \text{where } E_i \rightarrow E_j\};$$

and

$$\text{Id}(E_i) = \text{Key}(R_i) - \{A_j | A_j \subseteq \text{Id}(E_j), \text{where } E_i \rightarrow E_j\};$$

for every r-vertex $R_i$ corresponding to relation scheme $R_i(A_i)$:

$$\text{Atr}(R_i) = A_i - \{A_j | A_j \subseteq \text{Id}(E_j), \text{where } R_i \rightarrow E_j\};$$

(6) verify that the resulting ERD digraph, $G_{ER}$, is proper, that is, constraints (ER0) to (ER6) hold.

Several remarks concerning $T^R_S$ are in order:

(i) since implication for unrestricted sets of INDs and FDs is undecidable [6], we must require the input set of INDs to be acyclic; implication for acyclic INDs and FDs has exponential lower bound [5], therefore step (3) of $T^R_S$ is highly complex; restricting further the set of input INDs to typed and key-based INDs, which we know to be the only ones we are interested in, then the complexity of this step could be reduced to polynomial (Propositions 1.2 and 1.3);

(ii) step (2) of $T^R_S$ is based on lemma 3.1 and proposition 3.2;

(iii) in step (3) apparently missing INDs are marked; this feature of $T^R_S$ highlights the precedence given in $T^R_S$ to the keys over the inclusion dependencies: we assume that the choice of the key for a relational scheme is not arbitrary, but it represents either an entity or a relationship; accordingly, an attribute set $A_j$ compatible with some key $K_i$, refers to the same entity-set or relationship-set which is represented by $K_i$ and we shall expect that either $R_j[A_j] \subseteq R_i[K_i]$ or $R_i[K_i] \subseteq R_j[A_j]$ will hold;

(iv) the failure of $T^R_S$ does not mean necessarily that the relational schema has no associated ERD, but could be the result of a specific key choosing; such failures could be limited by extending the input set of keys with several, rather than one, candidate keys for every relational scheme;
Theorem. 

(9) the restriction put on the input set of keys is analogous to the constraints (ER1.iv) and (ER6.ii) (when roles are not considered); the following example illustrates the significance of constraint (ER6); let E(A), F(AB), G(AC) and R(ABC) be relational schemes with keys A, AB, AC and ABC, respectively, and let R(AB) \subseteq F(AB), R(AC) \subseteq G(AC), F[A] \subseteq E[A] and G[A] \subseteq E[A] be INDs; the candidate ERD, which does not obey (ER6), would be

![ERD diagram](image)

but the relationship key for R should be ABAC;

an extension of \( T^R_5 \) in the sense of [10] could cope with the removal of these restrictions; the extension is straightforward but tedious.

Entity-Relationship Consistency for relational schemas is defined as follows.

Definition.

A relational schema \((R,K,I)\) is said to be **ER-consistent** if there is a mapping \( S \) from the set of relational schemas to the set of ERDs, such that

\[ T_S \circ S(R,K,I) \text{ and } (R,K,I) \text{ are related in the following way:} \]

(i) attributes are the same, up to a renaming for compatible key attributes;

(ii) relation schemes are the same, up to the renaming of compatible key attributes;

(iii) \( T_S \circ S(I) \) is a consequence of \((K,I)\).

Theorem.


Sketch of the proof: The if direction follows from the specification (Input/Output of \( T^R_5 \)) and correctness of \( T^R_5 \) (whose proof is out of the scope of this paper). For the other direction, it suffices to show that \( T^R_5 \) succeeds on \( T_S \circ S(R,K,I) \) rather than on \((R,K,I)\). The latter is true because \( S(R,K,I) \) is an ERD and therefore satisfies constraints (ER9) to (ER8).
ER consistency for relational schemas

Note that $T^R_4 T^P_4 (G_{ER}) = G_{ER}$, but $T^P_4 T^R_4 (R,K,I) \neq (R,K,I)$, that is, $T^P_4$ is only the left inverse of $T^R_4$:

\[
\begin{array}{c}
(T^R_4, R,K,I) \\
\downarrow \\
G_{ER} \\
\uparrow \\
(T^P_4, R',K',I')
\end{array}
\]

The acyclicity, boundness and key-basing properties imposed on the set of INDs have been proposed in [11] in order to restrict the excessive, and possibly harmful, power of INDs. These properties, considered in [11] as necessary characteristics of well-designing, are captured in a precise manner by ER-consistency. Moreover, ER oriented schema design makes possible the expression of the explicit inclusion dependencies as ERD inherent [13] constraints, that is, constraints that are part of the schema structure.

5. ER-Consistency vs Relational Normalization.

The relational model has intended to achieve data independence by providing a clear boundary between the logical and physical aspects of database management [4]. ER-consistency carries out the same objective on a higher level by making independent the actual information from its logical data representation.

Relational normal forms have been developed in order to decrease both the impact of the side effects when changing relations and the data redundancy in relations. The main cause of lack of normalization is the embedding of data about independent real-world facts into one relation. Real-world facts are described by natural language sentences. We shall call a fact elementary if it is described by a single verb declarative natural language sentence. The schema design based on entities and relationships plus the principle of taking care that a relationship-type would always model an elementary fact, ensure, together, a certain level of information normalization.
ER consistency for relational schemas

ER-consistent schemas favor the realization of many of the relational normalization objectives. An example should make this clear. Let SALE(DEPT, ITEM, FLOOR) be a relational scheme with DEPT, ITEM as key and associated with the FD DEPT → FLOOR. The scheme is not normalized, namely not in 2NF, because of the partial functional dependency of attribute FLOOR on a non-key attribute subset. Suppose that the key above is only ITEM; then the scheme is also not normalized, now not in 3NF, because of the transitive dependency of attribute FLOOR on key ITEM. In both situations the relational scheme represents the embedding of independent relationships, among DEPT and FLOOR, and DEPT and ITEM, respectively.

Relational normal forms are data representation normalizations, while ER-consistency is an expression of information normalization. Therefore, ER-consistency is not equivalent to any of the normal forms, such as 2NF, 3NF, etc. Thus, SALE in the above example becomes normalized if both DEPT and ITEM are candidate keys. As another example, take the step of collapsing relations with equivalent keys in the normalization procedures (cf [14]). The equivalence of keys could represent a one-to-one relationship, such as an ISA relationship, therefore the merging of such relations might lead to the embedding of independent facts, precisely what normal forms try to avoid. By achieving a certain level of semantic separation, the relational normal forms overlap with ER-consistency, but do not ensure it.

Similarly, ER-consistency does not ensure relational normalization as illustrated by the following example: let SUPPLY(DN, IN, SN), with key DN, IN, be the translate of the SUPPLY r-vertex of

![Diagram]

the additional constraint expressed by the FD IN → SN, makes the relation scheme not to be in 2NF.
We propose for the ER-consistent relational schemas a normal form called *Entity-Relationship Normal Form (ERNF)*.

First we present the mapping of ERD digraphs into ERNF relational schemas. This mapping, $T_n$, is a modified version of the mapping $T_s$ of section 3. Given an ERD digraph $G_{ER}$, we associate with every e-vertex $E_i$ a dummy a-vertex denoted $%E_i$. The relational schema is of the form $(R,K,I,J)$ where $(R,K,I)$ is as before and $J$ is a set of pairs of the form $(X,Y)$, such that every pair corresponds to the relational scheme translate of an e-vertex $E_i$ which has no outgoing ISA edges, $X$ corresponds to the dummy a-vertex $%E_i$ and $Y$ corresponds to $\text{Id}(E_i)$.

*Mapping ER Diagrams into ERNF Relational Schemes Procedure ($T_n$).*

Input: $G_{ER} = (V,H)$, an ERD digraph;

Output: the ERNF relational schema $(R,K,I,J)$;

(0) initially $R,K,I,J$ are empty;

(1) for every e-vertex $E_i$ compute the set of a-vertices:

$$\text{Key}(E_i) = %E_i \cup \{\text{Key}(E_j) \mid E_i \rightarrow E_j\};$$

(2) for every e-vertex $E_i$ that does not have any outgoing ISA edge:

$$J := J \cup (%E_i, \text{Id}(E_i))$$

(3-5): steps 2ii-4 of $T_s$ of section 3.

An exemplification of the above mapping is given in figure 4, for the ERD digraph of figure 1, section 2.

Several remarks concerning the above mapping are in order:

(i) we do not have to rename, as we did $T_s$ (step 1), the labels of the a-vertices belonging to entity-identifiers in order to make them globally unique within the ERD digraph;

(ii) note that the dummy attribute has almost the same meaning as the surrogate attribute; recall that a surrogate is intended to represent an aggregated object, that is an entity, within a given database state [3]; a surrogate is information free and the user can do no more than cause its deletion or generation; for a given
relational-schemes \((R\text{-}keys\text{\ are\ underlined})\):

\[
\begin{align*}
\text{COUNTRY} & \quad (\%\text{COUNTRY}, \%\text{NAME}) \\
\text{CITY} & \quad (\%\text{CITY}, \%\text{COUNTRY}) \\
\text{STREET} & \quad (\%\text{STREET}, \%\text{CITY}, \%\text{COUNTRY}) \\
\text{PERSON} & \quad (\%\text{PERSON}, \%\text{ID}) \\
\text{EMPLOYEE} & \quad (\%\text{PERSON}, \%\text{EMPN}, \%\text{SALARY}, \%\text{ID}) \\
\text{LIVES} & \quad (\%\text{STREET}, \%\text{CITY}, \%\text{COUNTRY}, \%\text{PERSON})
\end{align*}
\]

inclusion dependencies \((I)\):

\[
\begin{align*}
\text{CITY}[\%\text{COUNTRY}] & \subseteq \text{COUNTRY}[\%\text{COUNTRY}] \\
\text{STREET}[\%\text{CITY}, \%\text{COUNTRY}] & \subseteq \text{CITY}[\%\text{CITY}, \%\text{COUNTRY}] \\
\text{LIVES}[\%\text{STREET}, \%\text{CITY}, \%\text{COUNTRY}] & \subseteq \text{STREET}[\%\text{STREET}, \%\text{CITY}, \%\text{COUNTRY}] \\
\text{LIVES}[\%\text{PERSON}] & \subseteq \text{PERSON}[\%\text{PERSON}] \\
\text{EMPLOYEE}[\%\text{PERSON}] & \subseteq \text{PERSON}[\%\text{PERSON}]
\end{align*}
\]

\[J=\{ (\%\text{COUNTRY}, \%\text{NAME}), (\%\text{CITY}, \%\text{NAME}), (\%\text{STREET}, \%\text{NAME}), (\%\text{PERSON}, \%\text{ID}) \}.
\]

\[\text{Fig. 4 Normalization Mapping of the ERD digraph of Fig. 1.}\]

entity-set \(E_i, \text{Key}(E_i)\), which is a set of dummy attributes, corresponds to the surrogate attribute and \(\text{Key}(E_i)\) consists of more than one dummy attribute only for entity-sets which are ID related to other entity-sets;

(iii) note that extending \(\mathcal{N}_c\) to handle several, rather than one, identifiers per entity-set, is straightforward;

(iv) \(T^R \circ T^c\) can be easily extended in order to obtain \(T^R \circ T^c(G_{ER}) = G_{ER}\).

The normalization procedure for an ER-consistent relational schema is \(T^R \circ T^c\). The following proposition characterises the relationship between an ER-consistent relational schema and its corresponding ER normal form.
**Proposition 5.1**

Let \((R,K,I)\) be an ER-consistent relational schema, \((R^*,K^*,I^*,J^*) = T_s^R \circ T_{\eta_0}(R,K,I)\), its corresponding ERNF schema and let \((R',K',I')\) be the output of the following mapping, \(T_{d}\):

**Input:** \((R^*,K^*,I^*,J^*)\), an ERNF relational schema;

**Output:** the relational schema \((R',K',I')\):

1. **The attributes of \(J^*\)** are renamed in order to become globally unique within the relational schema;

2. **For every pair** \((X,Y)\) of \(J^*\), replace \(X\) by \(Y\) in:
   - **(ii,R)** every relational scheme \(R_i\) of \(R^*\);
   - **(ii,K)** every key \(K_i\) of \(K^*\); and
   - **(ii,I)** every IND of \(J^*\).

Then \((R,K,I)\) and \((R',K',I')\) are related in the following way:

1. **Attributes** are the same, up to a renaming for compatible key attributes;
2. **Relation schemes** are the same, up to the renaming of compatible key attributes;
3. **\(I\)** is a consequence of \((K,I)\).

The proposition, whose proof is straightforward, expresses the following extension of the diagram of the preceding section:
**Conclusion.**

We have introduced the concept of ER-consistency for relational schemas. ER-consistency, which is an expression of information structure normalization, allows to carry over precisely all the advantages of the ER-oriented design principles to the relational model. ER oriented design, expressed by ER-consistency, and the associated ER normal form, simplifies and makes natural the task of keeping independent facts separated and assures a greater adaptability to changes not concerning the structure of the modeled information, such as the cardinality of relationships. We propose ER-consistency as the schema design principle replacing the relational normalization, as recommended by [1].

We have presented two mappings; the first defines the relational interpretation of ERD digraphs; the second attempts to find the ERD digraph corresponding to a relational schema. The range of both mappings can be extended in the sense of [10], by using roles in order to loosen constraint (ER6). For the ER-consistent relational schemas we have proposed an *Entity-Relationship Normal Form*, and we have presented the corresponding normalization procedure.

The ERD edges represent only a part of the INDs belonging to some set I in a relation schema. Generally this restriction is useful in keeping the set of INDs manageable. However, some INDs are not representable because of the limitations of the ERD, such as the impossibility of connecting two r-vertices. An extension of the mappings $T_s$ and $T_r^R$, over ERDs relaxing these restrictions is under work.

ER oriented design, expressed by ER-consistency, simplifies and makes natural the task of keeping independent facts separated and assures a greater adaptability to changes not concerning the structure of the modeled information, such as the cardinality of relationships.

The structural flatness of the relational model which limits the capability of the model in capturing the real-world semantics, is also the source of most difficulties experienced by users in mastering relational query languages. Relations are viewed as tables and users have to be concerned mainly with table manipulations and must be aware of...
the semantics of the table transformations they perform. ER-consistent schemas bring the design process closer to the way people perceive information. The modeled information consists of facts described by natural language sentences. The ERD closely resembles the surface semantics of the corresponding sentence types. Accordingly, a manipulation language over ERD could have constructs based on natural language constructions rules, thus ensuring a closer relationship between the data modeling and the natural structures as perceived by users. A query language over ERD following this principle has been proposed in [9].

Beside navigational paths ERD edges are intended to specify existence constraints: relationships can exist only when the related entities also exist; entities may be either independent or contingent on the existence of other entities through ISA or IS relationships. Existence constraints determine a certain way of update propagation required by the need to enforce these constraints, and ERD favors their precise definition on the schema level [12].

The mappings between ERDs and relational schemas should be completed with the direct and reverse translations of ER oriented operations to operations over the translate relational schemas, which would mean to extend ER-consistency to operations. Operations over ERD are meant to represent certain types of real-world information manipulations, thus ensuring a close relationship between this level of data modeling and the natural structures as perceived by users. The direct operation mapping will define the relational manipulations consistent with the ERD graph transformations. Provided the relational schema is ER-consistent, a (composition of) relational operation should preserve the ER-consistency of the relational schema. Conversely, it is worth investigating when and whether a reverse mapping, \( T_{ER} \), is possible.

References


ER consistency for relational schemas


