Documenting commutative diagrams of relationships to eliminate sources of redundancy in relational data design - Part One - Methodology.

John Cartmell

September 1, 2016

DRAFT

Abstract

We illustrate a shortcoming in relational data design methodology and propose as a remedy the modelling of certain relationship identities which we call relationship scope constraints. Part Two of this paper formulates a mathematical justification for this methodology.

1 Introduction

E.F.Codd's meta theory, presented as the relational model of data [Cod70], is fully formed – the meta concepts of table, column and primary key are defined as is that of a foreign key enabling one table to cross reference the rows of another. His is a theory of *what data is* and this theory has come to underpin the majority of corporate databases. Each such database, in accord with Codd's prescriptions, holds a meta-description of its own units of storage – the tables, columns and keys – what their names are and how they fit together to enable navigation through the data; this description is the core of what is described as a relational schema. The development of the relational model of data was strongly influenced by the predicate calculus representation of formal logic but arguably this meta-mathematics that influenced Codd has been overtaken by later 20th century meta-mathematics in the form of type theory and category theory; these are more diagrammatic in form and lead not to the relational model of data but to versions of the binary entity relationship model. It is these other meta-mathematical disciplines that influence this paper and lead to meaningful improvements in relational design methodology. Paradoxically, each such improvement in relational design methodology undermines the pre-eminence enjoyed by the relational model.

Codd has described various tests of goodness of a schema applicable, it must be remembered, only with cognisance to the possibilities among the data that it is designed to hold i.e. the intended usage. In the first instance three tests were described and successively a schema is said to be is 1st normal form, 2nd normal form or 3rd normal form depending on its success in passing the tests. A process for fixing deficient schemas is described as normalisation of the schema. Normalisation is therefore a method for converting or transforming one relational schema into another deemed more suitable for the purpose at hand. We can visualise so:

$$\begin{array}{c}\text{relational}\\ \text{schema}\end{array} \xRightarrow{normalise} \begin{array}{c}\text{relational}\\ \text{schema}\end{array}$$

In formal logical terms a relational schema presents a ‘theory of what is’ and normalisation is the process of improving a theory by (i) tightening the theory to better fit the facts and (ii) removing redundancy from the presentation so that the primitives are appropriate as units of storage.

Subsequently, the relations of Codd’s model are more abstractly presented, as either entities or as n-ary relationships, in Chen’s entity-relationship model of data described in [Che76] ; in the approach of Chen there is emphasis on a diagrammatic representation of the model. Chen describes a method for constructing a relational schema (in the sense of Codd) from an entity-relationship schema (ER-schema). He states that normalisation of the relational schema might be required after construction from an ER-schema – though why this might be is not explained. This yields a design process which is a combination of automatic transformation followed by normalisation:

$$\text{er schema} \xrightarrow{\textit{automatic transform}} \text{relational schema} \xrightarrow{\textit{manually normalise}} \text{relational schema}$$

This is said by Chen to be a top-down way to develop a relational schema in contrast to the Codd approach which he describes as bottom up. Note that some authors have mistakenly claimed that the second step within this workflow (the normalisation step) will be unnecessary if the ER-schema is itself in normal form. We will give a significant example where this is not so and in so doing illustrate the concept of relationship scope that was introduced in [ACE97] but is generally absent from current day presentations of ER modelling; we show here that we can mobilise this concept of the scope of relationships to narrow the process gap from ER modelling to fully normalised relational schema.

Within the context of software design methodologies, the normalisation step is represented as a manual process; to perform normalisation additional information is required comprising certain meta-knowledge about the facts that are the subject matter of the data. In relational theory, this additional required knowledge is in part represented as a set of integrity constraints comprising functional dependencies (FDs) and inclusion dependencies (INDs). If these integrity constraints are modelled as an enrichment of the relational schema then we can automatically normalise[1]:

$$\text{relational schema +FDs+INDs} \xrightarrow{\textit{automatically normalise}} \text{relational schema}$$

But normalisation to defined normal forms is not exactly the endpoint - elimination of reduction is the point. In [LV00] it is shown that if normalisation takes into account both FDs and INDs then redundancy is eliminated but this result depends on a definition of redundancy given in the first place in terms of FDs and INDs. We can give a simple example in which FDs and INDs do not provide sufficient information about the relational schema for the normalisation process to eliminate elementary redundancy in the schema and not strictly limited to redundancy as defined in [LV00].

The principal goal of this paper is to provide an answer to how the additional constraint knowledge required for normalisation can be represented in ER-terms rather than relationally

[1] this doesn’t take account of 4th and 5th normal forms

such as in terms of FDs and INDs and to go beyond current relational theory in a methodology for removing elementary redundancy. It is for this we employ the concept of the scope of a relationship; if the notion of an ER-schema is extended by the concept of relationship scope then we can automatically convert from the scope enriched ER-schema to a relational schema in normal form.

We propose a design process which follows the left and lower edges of this square:

$$\begin{tikzcd}
\text{ER-schema} \arrow[r, "\textit{automatically transform}"] \arrow[d, swap, "\textit{define scopes}"] & \text{relational schema} \arrow[d, "\textit{normalisation}"] \\
\text{scope enriched ER-schema} \arrow[r, swap, "\textit{automatically transform}"] & \text{relational schema}
\end{tikzcd}$$

in preference to the upper and right edges. We illustrate that by following this approach sources of redundancy are eliminated.

After Chen's 1976 paper, coming into and through the 1980's, came the development, concurrently, of Computer Aided Software Engineering (CASE) tools, including Meta-CASE tools, and semi-formalised and, in some instances, standardised official methodologies and notations, supporting structured systems analysis and development. Universally in the methodologies from this time the terms entity and relationship introduced in Chen's paper were retained within a logical modelling phase and Chen's transformation step into relational database design, inclusive of a normalisation step, is likewise retained. Though the terms and the overall shape of the process is retained the concepts behind these terms are adjusted. Most noticeably 'relationships' are now 'binary relationships' and at an early stage in these methodologies many-many relationships are eliminated in favour of many-one relationships. At this point there has been a conceptual *volte face* for a many-one binary relationship, implementation considerations aside, is a thinly disguised pointer between records of a file, such as in a VSAM file system, or a link between records in the network database model and it can be conceptualised, abstractly, as a function between sets of like-typed entities - leading some authors to describe a functional model of data [BF79],[Shi81]. The entity-relationship diagrams of these software analysis methods and the accompanying CASE tools that emerged in the 80's bear more resemblance to notation that preceded the work of Codd and Chen such as Bachman's data structure diagrams than to the diagrams of Chen. Among the many, and as summarised in [RE89], there are three variants of binary entity relationship diagram that stand out, those found, respectively, in SSADM/Barker-Ellis (now adopted by Oracle), in Clive Finkelstein and James Martin's Information Engineering, and in IDEF.

Chen's paper introduced the idea of entities being dependent on binary relationships with others for both their identification and their existence:

> Theoretically, any kind of relationship may be used to identify entities. For simplicity, we shall restrict ourselves to the use of only one kind of relationship: the binary relationships with 1:n mapping in which the existence of the n entities on one side of the relationship depends on the existence of one entity on the other side of the relationship. For example, one employee may have n (= 0, 1, 2, . . .)

> dependants, and the existence of the dependants depends on the existence of the corresponding employee. This method of identification of entities by relationships with other entities can be applied recursively until the entities which can be identified by their own attribute values are reached. For example, the primary key of a department in a company may consist of the department number and the primary key of the division, which in turn consists of the division number and the name of the company.

In many cases, software methodologies and supporting CASE tools introduced an intermediate step between the ER model and the relational model naming the intermediary model the physical design model and the starting model the logical model. This shifted the problem slightly but didn't make it go away. I shall call such an automatic transformation between logical and physical models the Chen transform. It is described in section 3.

$$\begin{tikzcd}
\text{logical er schema} \arrow[r, Rightarrow, "\substack{\textit{Chen transform}\\\textit{(automatic)}}"] & \text{physical er schema} \arrow[r, Rightarrow, "\substack{\textit{manually}\\\textit{normalise}}"] & \text{physical er schema} \arrow[r, Rightarrow, "\substack{\textit{code}\\\textit{generate}}"] & \text{relational schema}
\end{tikzcd}$$

In the mathematical description in Part Two we shall present a general definition of ER-schema which is general enough to include both purely logical schemas that are to the left of this diagram and the physical schemas to the right. We shall define the term ER model to mean an ER schema and all its intended usages and we shall show that by revising the definition of the Chen transform we can show that for each well-formulated purely logical schema there is a corresponding relational schema in normal form. In this paper we are more concerned with illustration of this as a methodology.

Following PCTE[2][BGMT88],[ECM97], we use the term *composition relationship* for Chen's *binary relationships with 1:n mapping in which the existence of the n entities on one side ... depends on the existence of one entity on the other side* and we use the term *reference relationship* for binary relationships which are neither composition relationships nor their inverses. We shall also describe the inverses of composition relationships as being *dependency relationships*. Earlier than this a similar distinction had been made by the designers of the CAIS specification [Obe88] but in which the two kinds of relationship were distinguished as primary and secondary - their rationale for the distinction [MOPT88] was as follows: *[Entities] and relationships may form a general graph or bowl of spaghetti. However, this raises various practical problems of deletion and garbage collection, longterm naming, and unconnected sub-graphs. CAIS therefore designates certain relationships as primary (and all others as secondary) and requires that all [Entities] and primary relationships in the data base form a single tree structure.* This distinction between composition and reference made by both CAIS and then PCTE served the goal of modelling computer file systems within a database framework, see figure 1 for example. In this paper we shall not assume that all composition relationships are identifying nor, vice-versa, that only composition relationships may be identifying.
In this paper to depict ER-schemas we use a version of the Barker-Ellis notation. Figure 4 is a meta-model of this notation – it is an ER schema describing ER schemas.
In cases where we wish to distinguish composition relationships from reference relationships then we draw the diagram top down: an anonymous root entity type (the 'absolute') is introduced at the top of the diagram, relationships leaving the lower edges of boxes are composition relationships and they always meet the top edge of the box representing the dependent type, reference relationships meet boxes from one side or the other. We note that there is a structural resemblance to diagrams drawn by Bachman[Bac73].

[2]

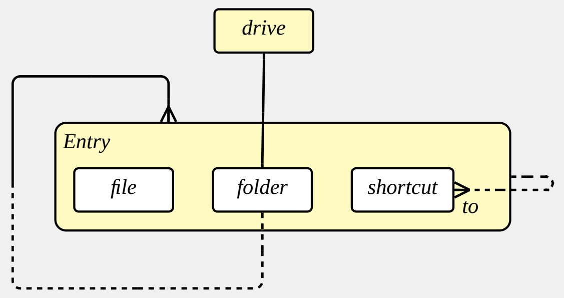
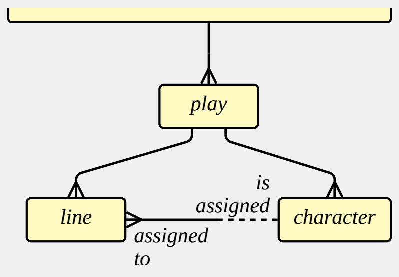


Figure 1: An ER model of folder system modelling the hierarchical structure as a recursive composition relationship and shortcuts as reference relationships.



(i) a play is composed of one or more spoken lines

(ii) a play is composed of one or more characters

(iii) a spoken line is assigned to exactly one character

(iv) each line of a play is assigned to a character of that same play

Figure 2: Composition and reference within the script of a play

The example in figure 2 shows two composition relationships (which have been left unlabelled) and one reference relationship doubly labelled. See figure 3 for another example of the notation.

It has often been noted that there is a disparity, and therefore a conflict, between uses of the term ‘model’ in mathematical logic and use of the same term in other disciplines including database theory, for in database theory and other disciplines, models are ‘theories of what is’ - they are models of conceptual situations and the term is synonymous with ‘theory’, whereas in mathematical logic the term model is used to mean an instantiation or an interpretation of such a theory. In the database sense of the word, models are represented in database schemas; to avoid ambiguity, we will choose to use the term schema synonymously with database model whilst remembering that such a thing is also a theory. We will refer to different kinds of schema as relational schema or ER-schema. By ER-schema, unless stated otherwise, we will mean not the ER-schema in the sense of Chen but a schema of entity types, binary relationships and attributes as meta-modelled in figure 4.

In the terminology of Ellis[Ell82], wherever in an entity model there is a path of single-valued relationships $a \overset{r_1}{\ni\!-} \cdot \overset{r_2}{\ni\!-} \cdot \ldots \overset{r_n}{\ni\!-} b$ then the destination entity type b is said to be in the *logical horizon* of the source entity type a. In programming, equivalently, we might say that it was possible to navigate from one to the other. Now if there are two such navigation paths between

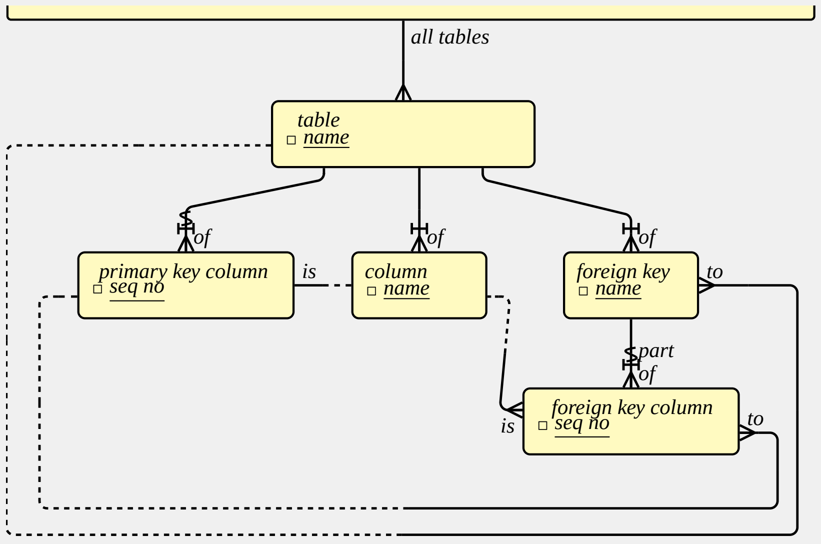


Figure 3: An ER model of Relational Schema. This is a logical ER model and so there is no indication as to how the relationships are implemented.

entity type *a* (the source) and entity type *b* (the destination) then a question naturally arises as to whether following one path is equivalent to following the other i.e whether starting at any entity of type *a* we arrive at the same destination entity of type *b* regardless of which of the two paths we follow. In an abstract mathematical setting, diagrams showing such equivalent paths are said to be *commutative diagrams* and methods of reasoning using such diagrams is the starting point of category theory. Johnson and Dampney [JD94] have emphasised the importance of recognising such commutative diagrams of relationships during entity modelling; in summary, there are identities between joins of derived relationships and these are important and should be documented during the construction of an entity model. Johnson, Dampney and Wood in [JRW02] formulate a description of ER model that goes beyond the view of an ER schema as a directed graph by addition of constraints including commutative diagrams, cartesian products and pullbacks by defining an ER schema as a presentation of category with finite limits and colimits. A similar definition of a data model specification is given by Piessens and Steegman [PS95]. In a further paper, Johnson and Roseburgh [JR02] show the relationship between their formulation of ER models and relational models. These descriptions written in the style of abstract mathematics call for extensions to the notation employed by entity modellers so that ER schemas can be more expressive.

Shlaer and Lang in [SL96] describe alternative paths between two entity types as relationship loops and when they are equivalent say that there are dependencies between the relationships. Kolp and Zimnyi ([KZ95]) instead use the term relationship cycle and identify them as a source of superfluous attributes in the transformation from ER model to relational model. They say: *ER cycles can be sources of superfluous attributes not detected by classical normalization. Hence, the interest of enhanced ER-based design methodologies that remove anomalies due to cycles and inclusion constraints.*

See figure 6 for a notation proposed in[SL96] for the expression of relationship dependen-

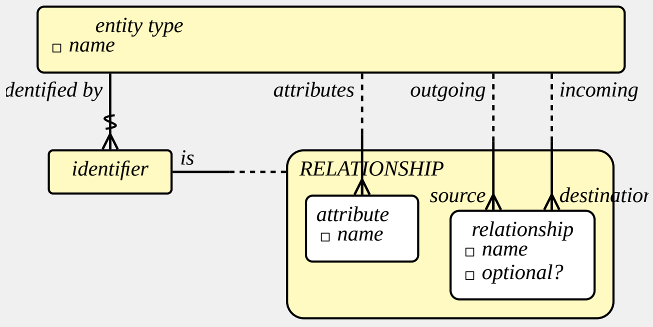


Figure 4: The logical ER meta-model. This is a logical ER model of a logical ER model.

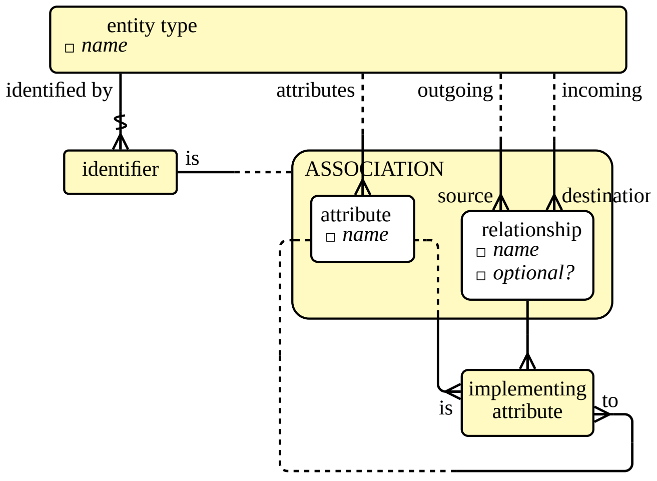


Figure 5: The Physical Entity Relational Meta Model. This is a logical ER model of a physical ER model – in the physical model referential attributes are introduced to represent the implementation of relationships.

cies. This notation enables certain commuting diagrams to be indicated on physical entity relationship diagrams.

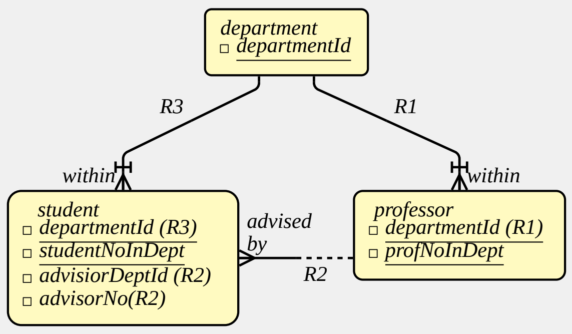


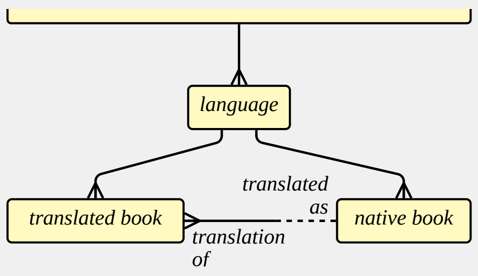
Figure 6: An example of a physical ER model. Physical ER models include attributes for the implementation of relationships and in [SL96] are annotated with the names of the relationships which they implement. If students are advised by professors within the same department then a single departmentId attribute of the student entity is used as the implementation of both R2 and R3 instead of the two shown.

2 Relationship Scope

With reference to points (i) to (iv) of figure 2, point (iv), unlike points (i), (ii) and (iii), expresses information not otherwise represented on the accompanying diagram. In the terminology that we introduce here it expresses a relationship scope constraint. This concept is significant to understanding the domain of discourse given by a model. It is significant in this case that the reference relationships is limited in scope - we can say that the instances of the reference relationship shown in figure 2 are local to the context of individual plays: to spell out what is meant by this – a line of one play is never assigned to a character of a *different* play – we might summarise this by saying that the relationship of line assignment is intra-play not inter-play. Most significantly, whilst the ER diagram notation is able to express the types and cardinalities of reference relationships and how they are articulated, it is unable to express such a constraint as this one in point(iv) of figure 2 – such a constraint as this we shall say is a scope constraint for the relationship. The thesis here is that every reference relationship has a scope, the scope is fundamentally important in data modelling, it can be expressed in words, or, as we see later, in equations between expressible relationships or in a commutative diagram called a scope diagram[3] and present, in the terminology of [SL96], a dependency between the reference relationship and certain other relationships.

Contrast the situation of figure 2 with that of figure 7. There is a similarity in shape but behind this there is a significant difference in that the reference relationship in figure 7 is *not* limited in scope. Rather the point of the relationship *translation* is to cross languages - it establishes an inter-language relationship rather than intra-language relationship. This again shows that the scope of a relationship - the extent to which it is global or local - 'inter' or 'intra' - cannot be deduced from the entity relationship diagram alone. Other means of expression must be used. Gaining an understanding of scope and the means of its expression is an important part of learning entity modelling.

[3]Scope diagrams, as we describe them here, are akin to the commutative diagrams used in category theory to express identities between differently composed morphisms however because relationships can be optional, they are not commutative diagrams but 2-cells in the 2-category of finite sets, partial functions and inclusions.



(i) *a language has one or more native books*

(ii) *a language has one or more translated books*

(iii) *a translated book is a translation of exactly one native book*

(iv) *a native book is translated as zero,one or more translated books*

Figure 7: Translations of a book

Knowledge of a relationship's scope is a significant part of understanding how a relationship is used and failure to respect this aspect of proper usage is what constitutes a scope violation. When talking about the cast members of a performance of King Lear it would be a scope error to suggest that a member of cast play the character Desdemona for Desdemona is a character within the scope of different play namely Othello. The type of entity is correct for 'Desdemona' is indeed a 'character' of a play, but the context is not. It is part of our knowledge of the 'plays part of'/'part played by' relationship (see figure 8) that this is a relationship whose scope is local to the enclosing 'play' context. We can say that it is intra-play rather than inter-play.

It would be a scope error in a conversation about antipodeans to assert that the New Zealand born physicist Ernest Rutherford could have been a native of Nelson in Lancashire, a place just 20 miles away from where he was Professor of Physics. Not only would it be a factual error – it would be non-nonsensical and this is because of the grasp we have of the meta-relationship between relationships 'country of birth' and 'place of birth' – that the latter is a more detailed version of the former. The assertion would violate the scope of the 'place of birth' relationship (see figure 9).

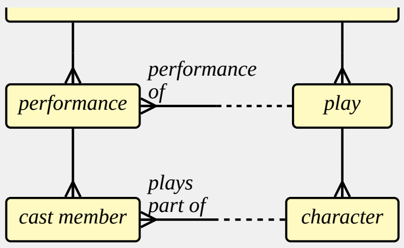


Figure 8: Model of the Performance of a Play - the relationship 'plays part of' is dominated by the relationship 'performance of'.

Likewise it would be a scope error to think that a local telephone call could be made between different countries or that the hydrogen of a water molecule could be covalently bonded to the oxygen of a *different* molecule or that the captain of one team in a cricket match might be scheduled to bat for the opposing team. All of these errors are characterised as failures to respect the scopes of relationships.

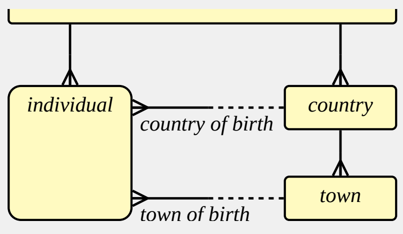


Figure 9: The relationship ‘place of birth’ is dominated by ‘country of birth’.

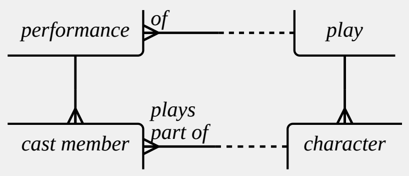
In accord with a fundamental principle of information theory the more constrained in scope a relationship is then the less the information needed to express its individual instances. From this it follows that the definition of relationship scopes is intimately connected to specifying information requirements for representing or communicating relationship instances and, in particular, for representing them in databases, relational or otherwise. For example if we wish to propose that, in the context of a performance of King Lear, we give someone the role of ‘the fool’ then we do not have to say ‘the fool in King Lear’ we simply have to say ‘the fool’ for our shared knowledge of the scope of the relationship ‘plays the part of’/‘part played by’ implies the rest.

In the final section of this paper we shall see how these observations translate into relational data design.

2.1 Diagrams Expressing Scopes

In hierarchical ER-schemas, relationships are classified as being either ‘composition’ relationships or ‘reference’ relationships. Of these, it is the reference relationships that have scopes whereas the composition relationships enable definition of scopes by modelling nested localities i.e. possible contexts. In mathematical notation it is possible to include the scope constraint as a more general kind of type constraint than can be expressed in an entity model, namely a ‘dependent type constraint’. In entity modelling this is not possible and every relationship defined in a model should have a scope constraint specified for it. There is no standard way of doing this but an accompanying diagram, one per associative relationship is a satisfactory way of doing so. Figure 10 is an example of such a diagram. In fact it is a pullback diagram and it is a typical instance of such occurring very naturally within a database schema.

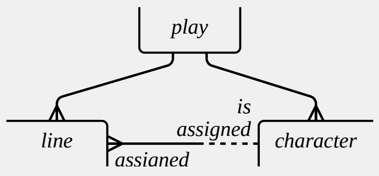
Similarly the diagram in figure 11 can be interpreted as the scope constraints for relationship ‘assigned to’ within the context of the entity model of figure 2. The text on the right of the figure explains the constraint expressed by the scope diagram - what it says seems obvious but this is so only if we *know* this model and this relationship i.e. providing we understand its proper usage. In this example shown in figure 11 there is a single entity type at the top of the diagram - therefore we call the diagram a scope triangle rather than a scope square. If we allow of the use of identity relationship in a scope square and allow it to be drawn horizontally then any scope triangle can be re-expressed as a scope square as illustrated by the redrafting



Whenever a cast member plays the part of a character then the performance the cast member is part of is a performance of the play the character is a part of.

Figure 10: A scope constraint diagram for relationship ‘plays part of’. In such a diagram it is the lower (horizontal) relationship which is the subject of the constraint. We have given an alternative explanation of the constraint to the right of the diagram but it is wordy and difficult to follow - for this reason we prefer to draw the diagram and have this mean the very same thing.

of figure 11 as figure 12.



Whenever a line of a play is assigned to a character then the play the character is part of is the same play as the line is part of.

Figure 11: A scope constraint diagram for relationship ‘assigned to’. This is a scope triangle because rooted at a single entity type.

Some relationships may be unconstrained in their scope in the sense that they are global in their reach. We have given an example of such a relationship, ‘translation of’, in figure 7. The scope of this relationship, the fact that it is unconstrained, is expressed by the relationship scope diagram in figure 13. By way of explanation - what this diagram says is :

> The absolute of the language of a translated book is the absolute of the language of the native book it is a translation of.

In other words it says that the relationship ‘translation of’ is such that two absolutes are equal - which is to say nothing at all about the relationship because *a priori* all absolutes are equal since absolute is unique of its type.

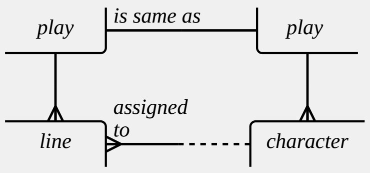


Figure 12: The identity relationship ‘same as’ used to express the scope triangle of figure 11 as a square.

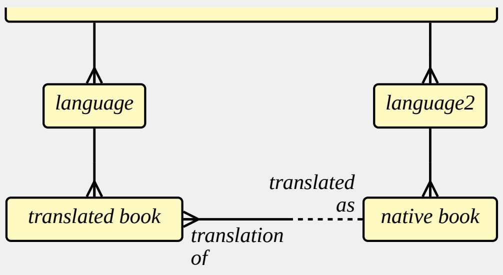


Figure 13: An example of a scope diagram for a relationship which is unconstrained in scope - the scope diagram is rooted at absolute.

3 Chen’s Transformation

Chen presents the transformation process from ER to relational by way of an example. He gives an example ER model and proceeds to say that from it that certain relations can ‘*easily be derived*’[4].

In terms of the binary ER model the transformation illustrated by Chen can be summarised thus:

I For each entity type on the diagram, a table is instantiated to represent the entity type.

II For each attribute of each entity type, a column is instantiated within the table instantiated to represent the entity type. Specifically *identifying* attributes are instantiated as primary key columns.

III For all identifying relationships, primary key columns of the table representing the source of the relationship are instantiated – one per primary key column of the table representing the destination entity type.

[4]The verb *migrate* is often used in descriptions of this process; for example I found a Wikipedia article describing a foreign key as a key that had migrated to another entity and I found a description elsewhere stating:

1. Identify and define the primary key attributes for each entity
2. Validate primary keys and relationships
3. Migrate the primary keys to establish foreign keys

The term ‘migrate’ is inappropriate because key columns do not migrate anywhere - they stay where they are - what happens is that for each primary key column and for each relationship a corresponding foreign key column is instantiated.

IV For all non-identifying relationships, columns of the table representing the source entity type of the relationship are instantiated one per primary key column of the table representing the destination entity type.

Another way of looking at the matter, rather than speaking of cascading and migrating keys, is based simply on the observation that the columns in the physical representation on an entity type a correspond to the attributes of the entity type a union the set of tuples $\langle r_1, ...r_n, p\rangle$ where $n \geq 1$ and where $a \overset{r_1}{\ni\!-} \cdot \overset{r_2}{\ni\!-} \cdot \ldots \overset{r_n}{\ni\!-} b$ is a path of single-valued relationships, where r_i is identifying for each $i > 1$ and where p is an identifying attribute of the destination entity type b of the relationship r_n. This observation suggests a formal mathematical definition of the Chen transform and this is the approach we follow in Part Two.

4 The Shortcomings of the Simple Chen Transformation

4.1 The Relational Meta Schema Example

Now let us apply this Chen transformation (I)–(V) to the meta-schema for relational databases (figure 3).
First some background. Since the scheme is data (sometimes said to be meta-data since it is data about the structure of data) then this schema can be held in a database (indeed, this was prescribed in articles by Codd in what became known as Codd's rules). The tables, columns and keys used to hold this data themselves have a description which itself is a schema. We shall refer to this here as the relational meta-schema. Different software vendors represent this relational meta-schema in different ways. One example which can be found online [MyS15] is for the MYSQL imno database engine.

An abstract representation of the relational meta-schema using the entity relationship notation we have already seen in figure 3.

Applying the Chen transformation to the model in figure 3 we get:

```
TABLE(TABLE-NAME)
COLUMN(TABLE-NAME,COLUMN-NAME)
PRIMARY-KEY-COLUMN(TABLE-NAME,INDEX-NO,IS-TABLE-NAME,IS-COLUMN-NAME)
FOREIGN-KEY(TABLE-NAME,NAME,TO-TABLE-NAME)
FOREIGN-KEY-COLUMN(TABLE-NAME,FOREIGN-KEY-NAME,INDEX-NO,
IS-TABLE-NAME, IS-COLUMN-NAME, TO-TABLE-NAME, TO-COLUMN-NAME)
```

We see that the generated table definitions have additional columns in two tables and these tables are not in normal form (for description of normal forms see [Ken83].) The generated definitions (relations) that we have obtained are not in normal form:

- PRIMARY-KEY-COLUMN is not in 1st normal form because the column 'IS-TABLE-NAME' is a duplicate column since its value in all cases will be identical to those of the TABLE-NAME column. We need remove the IS-TABLE-NAME column to remove the redundancy and to obtain a definition satisfying the 1st-normal form rule.

- The FOREIGN-KEY-COLUMN is not in 1st-normal form for, again, the column 'IS-TABLE-NAME' is a duplicate whose values are the same as those of TABLE-NAME. Even with this removed the table is not in 2nd normal form for the column TO-TABLE-NAME is redundant since its value in all cases can be obtained from the parent FOREIGN-KEY entity. We can remove the TO-TABLE-NAME column. Note that in this example when we normalise the table we pay for it the losing the possibility of expressing the referentiality constraint representing the 'TO' relationship[5].

4.2 What's Gone Wrong?

We might wonder whether something is wrong with out ER-model (figure 3) but that is not the case. What is wrong is that the transformation has not taken account of certain commutative diagrams among the primitive relationships of the ER model. There are identities among joins of the primitive relationships and these are the causes of the redundancies. There are three commutative diagrams in all that are the source of the problem. One of these is shown in figure 14. All of the three are indicated in figure 15 using scope annotations for each of the reference relationships. In each scope annotation the commutative diagram is expressed as an identity between two paths. The relationship which is the subject of the scope constraint is denoted by a tilde symbol (~).

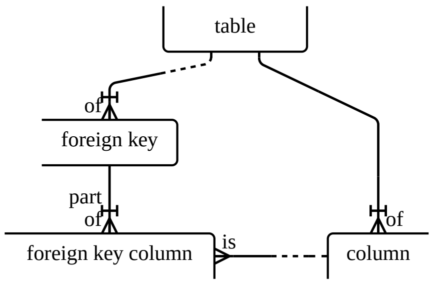


Figure 14: An identity among primitive relationships of the relational meta-model is given by the scope of the 'is' relationship – the cause of a redundant copy of the IS-TABLE-NAME column. Algebraically: is/of = part of/of. For pragmatic diagramming reasons if we use such an equation in the context of a particular relationship as a scope constraint on a diagram then the name of the subject relationship is replaced by a tilde symbol, to get, for example, ~/of = part of/of

We see from this that the general algorithm sketched above needs to be modified to take account of the scopes of relationships for we can see how the relational identities expressed in the scope constraints lead to the duplicate columns that have to be removed to eliminate redundancy and achieve 1st and 2nd normal form. We need the Chen transform to instead yield the physical ER model shown in figure 16. It is clear now that to achieve this we need to start from a logical model that has additional scope constraints documented, as illustrated in figure 15.

[5]This is surprising - as defined by Codd and as implemented in modern relational databases not all binary relationships can have matching foreign key constraints.

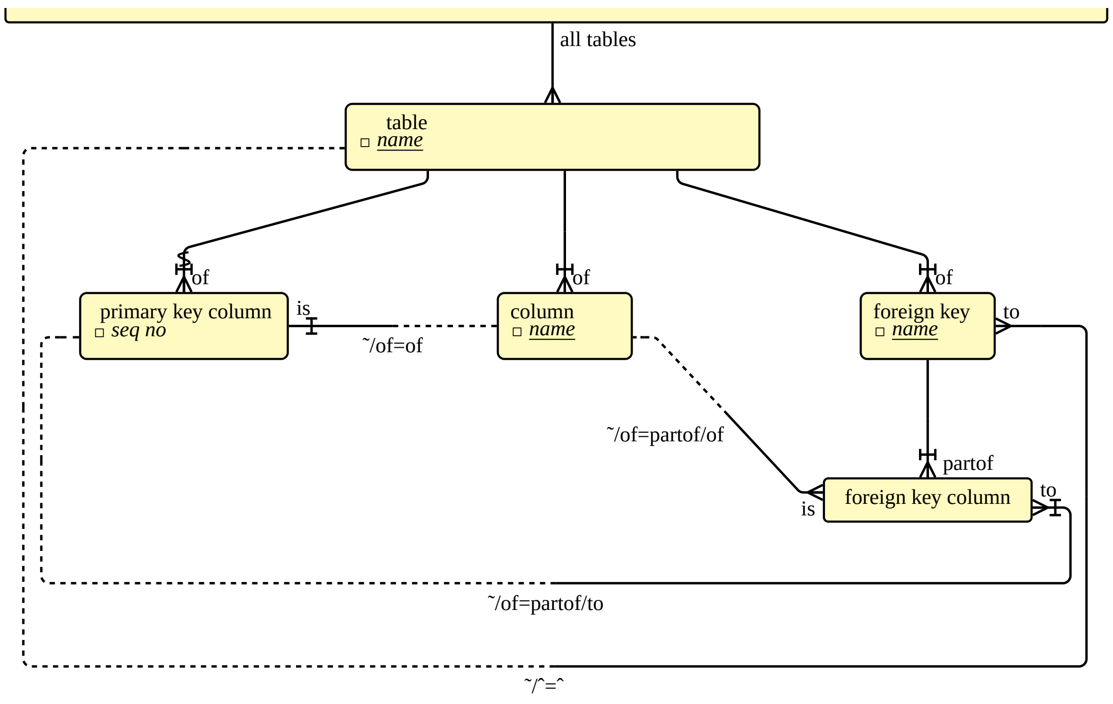


Figure 15: Logical ER model of the Relational Meta Model - showing relationship scope constraints in which tilde(~) denotes the relationship being scoped and the hat symbol (ˆ) denotes the absolute. The annotation ~/ ˆ = ˆ denotes a relationship of global scope.

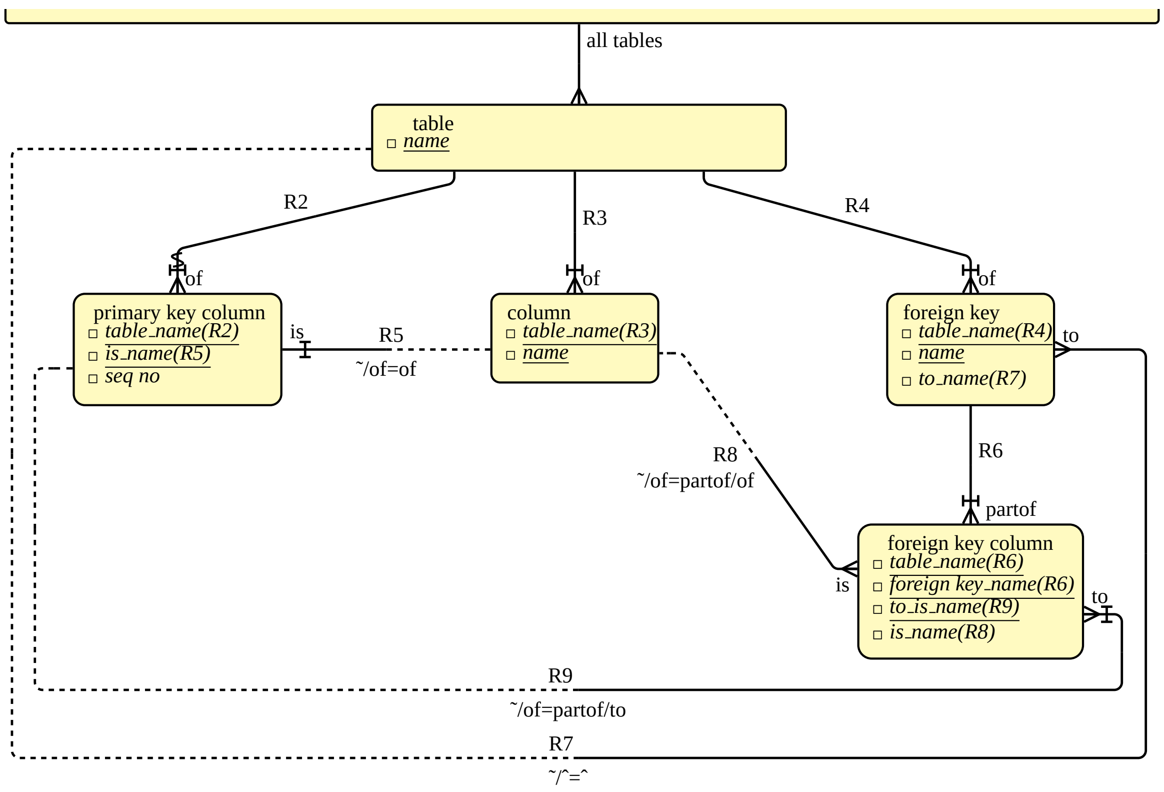


Figure 16: Physical ER model of the Relational Meta Model - showing how a revised algorithm generates it from previous figure.

5 Conclusion

Diagrams of relationships can be used to express integrity constraints for binary entity relationship models of data. They can be incorporated into methodologies for representing the scopes of reference relationships in top-down style Barker-Ellis ER-models. ER-models enriched in this way can be automatically transformed into relational schemas with higher degrees of normalisation (i.e lower levels of redundancy) than exhibited by previous methodologies.
Part Two of this work is a mathematically precise description and justification for this approach.

References

[ACE97] A. Alderson, J. W. Cartmell, and A. Elliott. On the scope of relationships. In *Proceedings of the 8th International Workshop on Software Technology and Engineering Practice (STEP '97) (Including CASE '97)*, STEP '97, pages 279–, Washington, DC, USA, 1997. IEEE Computer Society.

[Bac73] Charles W. Bachman. The programmer as navigator. *Commun. ACM*, 16(11):653–658, November 1973.

[BF79] Peter Buneman and Robert E. Frankel. Fql: A functional query language. In *Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data*, SIGMOD '79, pages 52–58, New York, NY, USA, 1979. ACM.

[BGMT88] Gerard Boudier, Ferdinando Gallo, Regis Minot, and Ian Thomas. An Overview of PCTE and PCTE+. In *Proceedings of the Third ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments*, SDE 3, pages 248–257, New York, NY, USA, 1988. ACM.

[Che76] Peter Pin-Shan Chen. The entity-relationship model — toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, March 1976.

[Cod70] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, June 1970.

[ECM97] ECMA. *ECMA-149: Portable Common Tool Environment (PCTE) — Abstract Specification*. Fourth edition, December 1997.

[Ell82] HC Ellis. A refined model for definition of system requirements. *Database Journal*, 12(3):2–9, 1982.

[JD94] Michael Johnson and C.N.G. Dampney. On the value of commutative diagrams in information modelling. In Maurice Nivat, Charles Rattray, Teodor Rus, and Giuseppe Scollo, editors, *Algebraic Methodology and Software Technology (AMAST93)*, Workshops in Computing, pages 45–58. Springer London, 1994.

[JR02] Michael Johnson and Robert Rosebrugh. Sketch data models, relational schema and data specifications. *Electronic Notes in Theoretical Computer Science*, 61:51–63, 2002.

[JRW02] Michael Johnson, Robert Rosebrugh, and RJ Wood. Entity-relationship-attribute designs and sketches. *Theory and Applications of Categories*, 10(3):94–112, 2002.

[Ken83] William Kent. A simple guide to five normal forms in relational database theory. *Commun. ACM*, 26(2):120–125, February 1983.

[KZ95] Manuel Kolp and Esteban Zimnyi. Relational database design using an er approach and prolog. In Subhash Bhalla, editor, *Information Systems and Data Management*, volume 1006 of *Lecture Notes in Computer Science*, pages 214–231. Springer Berlin Heidelberg, 1995.

[LV00] Mark Levene and Millist W. Vincent. Justification for inclusion dependency normal form. *IEEE Trans. on Knowl. and Data Eng.*, 12(2):281–291, March 2000.

[MOPT88] Robert Munck, Patricia Oberndorf, Erhard Ploedereder, and Richard Thall. An Overview of DOD-STD-1838A (Proposed) the Common APSE Interface Set: Revision. In *Proceedings of the Third ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments*, SDE 3, pages 235–247, New York, NY, USA, 1988. ACM.

[MyS15] MySQL 5.7 Reference Manual, 2015.

[Obe88] Patricia A. Oberndorf. The Common Ada Programming Support Environment (APSE) Interface Set (CAIS). *IEEE Trans. Software Eng.*, 14(6):742–748, 1988.

[PS95] Frank Piessens and Eric Steegmans. Categorical data specifications. *Theory and Applications of Categories*, 1(8):156–173, 1995.

[RE89] Rosemary Rock-Evans. *An Introduction to Data and Activity Analysis*. QED Information Sciences, Inc., Wellesley, MA, USA, 1989.

[Shi81] David W. Shipman. The Functional Data Model and the Data Languages DAPLEX. *ACM Trans. Database Syst.*, 6(1):140–173, March 1981.

[SL96] Sally Shlaer and Neil Lang. Shlaer-mellor method: The ooa96 report. Technical report, Project Technology, Inc., 1996.