DUNEDIN NEW ZEALAND

Modifications to Smith's Method for Deriving Normalised Relations from a Functional Dependency Diagram

Nigel Stanger

The Information Science Discussion Paper Series

Number 99/23
December 1999
ISSN 1172-6024

University of Otago

Department of Information Science

The Department of Information Science is one of six departments that make up the Division of Commerce at the University of Otago. The department offers courses of study leading to a major in Information Science within the BCom, BA and BSc degrees. In addition to undergraduate teaching, the department is also strongly involved in postgraduate research programmes leading to MCom, MA, MSc and PhD degrees. Research projects in spatial information processing, connectionist-based information systems, software engineering and software development, information engineering and database, software metrics, distributed information systems, multimedia information systems and information systems security are particularly well supported.

The views expressed in this paper are not necessarily those of the department as a whole. The accuracy of the information presented in this paper is the sole responsibility of the authors.

Copyright

Correspondence

This paper represents work to date and may not necessarily form the basis for the authors' final conclusions relating to this topic. It is likely, however, that the paper will appear in some form in a journal or in conference proceedings in the near future. The authors would be pleased to receive correspondence in connection with any of the issues raised in this paper, or for subsequent publication details. Please write directly to the authors at the address provided below. (Details of final journal/conference publication venues for these papers are also provided on the Department's publications web pages: http://www.otago.ac.nz/informationscience/pubs/publications.html). Any other correspondence concerning the Series should be sent to the DPS Coordinator.

Department of Information Science
University of Otago
P O Box 56
Dunedin
NEW ZEALAND

Fax: +64 3 479 8311
email: dps@infoscience.otago.ac.nz
www: http://www.otago.ac.nz/informationscience/

Modifications to Smith's method for deriving normalised relations from a functional dependency diagram

Nigel Stanger*

December 1999

Abstract

Smith's method (Smith, 1985) is a formal technique for deriving a set of normalised relations from a functional dependency diagram (FDD). Smith's original rules for deriving these relations are incomplete, as they do not fully address the issue of determining the foreign key links between relations. In addition, one of the rules for deriving foreign keys can produce incorrect results, while the other rule is difficult to automate. In this paper are described solutions these issues.

1 Introduction

A *functional dependency diagram* (FDD) is a means of graphically modelling the dependencies within a collection of attributes (Date, 1995, pp. 294–295). *Smith's method* (Smith, 1985) is a formal technique for deriving a set of normalised relations from an FDD. As part of this process, Smith defines two rules for deriving foreign keys (referred to by the author as the *target bubble rule* and the *domain flag rule* respectively). There are three major issues with these rules:

1. the target bubble rule does not always produce all possible foreign keys, even in relatively simple FDDs;
2. the target bubble rule can in certain situations produce 'foreign keys' that violate the relational definition of a foreign key; and
3. the domain flag rule is difficult to automate, because information that is important to the correct derivation of foreign keys cannot be expressed using Smith's original FDD notation.

In this paper are described new rules for deriving foreign keys from an FDD. In addition, the author proposes some minor modifications to Smith's original FDD notation to facilitate the process of deriving foreign keys.

*Address correspondence to: N. Stanger, Department of Information Science, University of Otago, P.O. Box 56, Dunedin, New Zealand. Fax: +64-3-479-8311. Email: nstanger@infoscience.otago.ac.nz

Smith's original method is summarised in Section 2. The issues with deriving foreign keys are then described in more detail in Section 3. In Section 4, modifications to Smith's FDD notation and two new foreign key derivation rules are proposed to address these issues. An example of the new rules in use is presented in Section 5, and the paper is concluded in Section 6. It is assumed that readers are familiar with the concepts and terminology of relational dependency theory (Armstrong, 1974; Beeri, Fagin and Howard, 1977; Date, 1995).

2 Overview of Smith's method

As previously stated, a functional dependency diagram (FDD) is a graphical representation of the functional dependencies within a collection of attributes. Smith derives FDDs from a set of plain English *dependency-list statements*, such that shown in Figure 1.

> Anticipated design engineering work is organized into JOB_NO engineering job numbers. Each JOB_NO has one TYPE_JOB (i.e., '1' = Basic Release, '2' = Sustaining, . . .), one RESP_ENGR responsible engineer (entered as an employee number), and one DUE_DATE planned due date.

Figure 1: Example of a dependency-list statement (Smith, 1985, Figure 1)

2.1 Smith's FDD notation

In Smith's FDD notation, attributes (Smith refers to these as 'fields') are placed within *bubbles*. Multiple attributes may be placed within the same bubble to simplify the diagram (see Figure 2).

Figure 2: (a) A bubble that contains a single attribute; (b), (c) bubbles that contain multiple attributes

A functional or *single-valued* dependency $A \rightarrow B$ is represented by a single-headed arrow between the corresponding bubbles. The bubble at the start of the arrow is called a *prime-key bubble*, as shown in Figure 3(a). The bubble at the end of the arrow is called a *target bubble*.

A multivalued dependency $C \twoheadrightarrow D$ is represented by a double-headed arrow between the corresponding bubbles. If the bubble at the end of the arrow is a prime key bubble, then the bubble at the start of the arrow is called an *uplink-key bubble*, otherwise it is a prime-key bubble. If the bubble at the end of the arrow is not a prime key or an uplink key, then it is known as an *end-key bubble*, as shown in Figure 3(b).

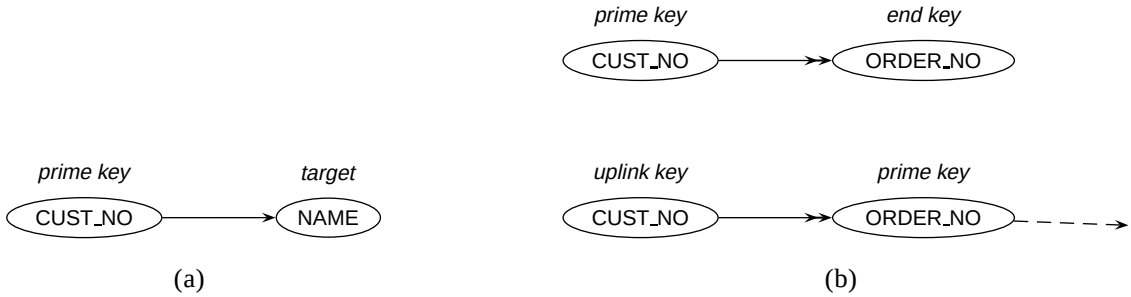


Figure 3: (a) Single- and (b) multivalued dependencies

Attributes may be placed within more than one bubble. 'Multibubbles' are often used to show the linkage of a chain of uplink-key, prime-key and end-key bubbles, as shown in Figure 4(a). Each bubble is independent of the others.

Domain flags are used to tag attributes which belong to the same domain. For example, EMP_NO and DEPT_MGR both belong to the domain 'employee number', as shown in Figure 4(b).

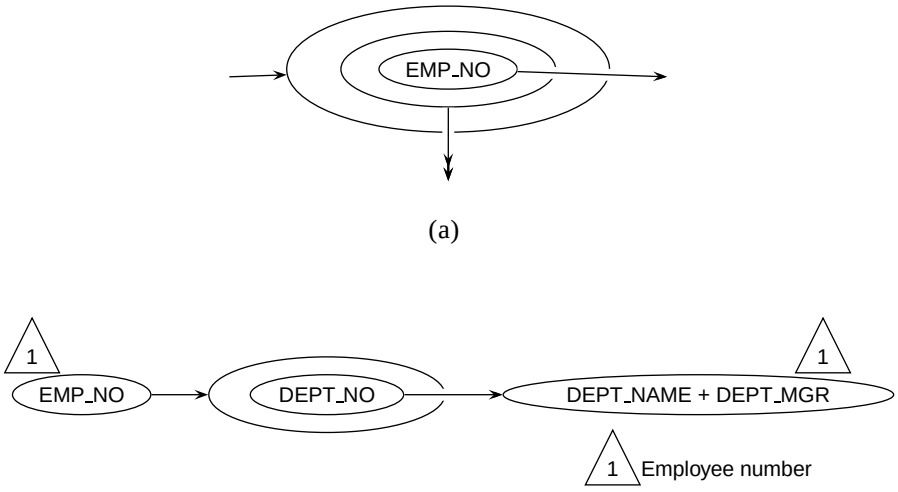


Figure 4: (a) Multiple bubbles and (b) domain flags

2.2 Smith's method for deriving a set of relations from an FDD

2.2.1 Single-valued dependencies composed into relations

All target bubbles of a prime-key bubble, plus all associated uplink-key bubbles (if any) become the attributes of a single relation. The primary key of this relation comprises the concatenation of all attributes within the prime-key bubble plus all attributes within associated uplink-key bubbles. An example is shown in Figure 5 on the following page.

Attributes within a target bubble become foreign keys of the derived relation if they also function as a key bubble of any sort (referred to by the author as the *target bub-*

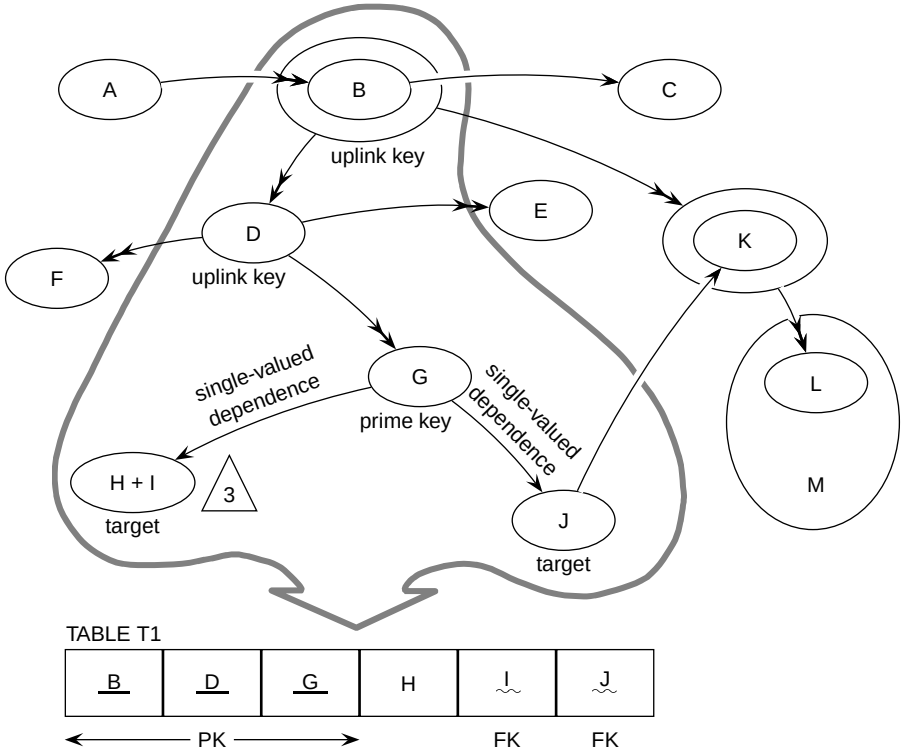


Figure 5: Deriving a relation from a single-valued dependency (Smith, 1985, p. 830)

ble rule), or are tagged with a domain flag (the *domain flag rule*). These rules shall be revisited in Section 3.

2.2.2 End-key dependencies composed into relations

All attributes of an end-key bubble, its prime-key bubble and all associated uplink-key bubbles become the primary key of a single relation. An example is shown in Figure 6.

2.2.3 Isolated bubbles composed into relations

An *isolated bubble* is one that has no arrows pointing either to or from it. All attributes within an isolated bubble become the primary key of a single relation.

2.2.4 Practicable diagrams

Smith defines a *practicable dependency diagram* as one that does not produce a relation with a primary key comprising three or more attributes. To correct an impracticable FDD, the diagram is modified by adding *surrogate keys* (Date, 1995, pp. 368–369) to break the offending relation(s) into two or more sub-relations (see Figure 7).

2.2.5 Additional guidelines

Smith also stated several additional guidelines for designing FDDs that generally produce a 'better' design (Smith, 1985, pp. 831–832). These guidelines do not have any impact on the foreign key issues and are therefore not described here.

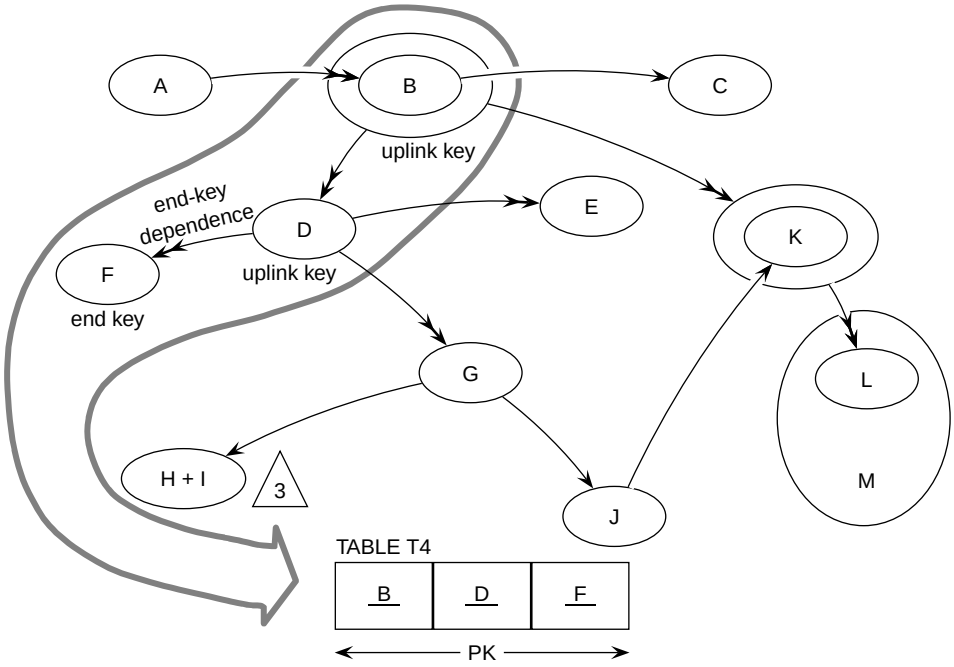
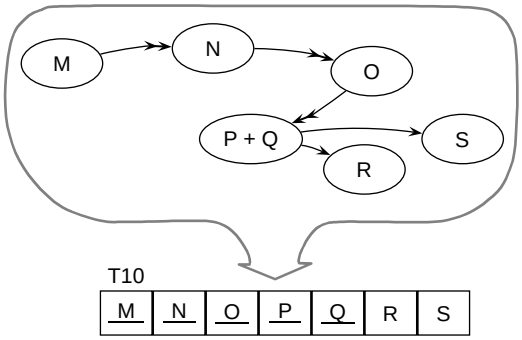
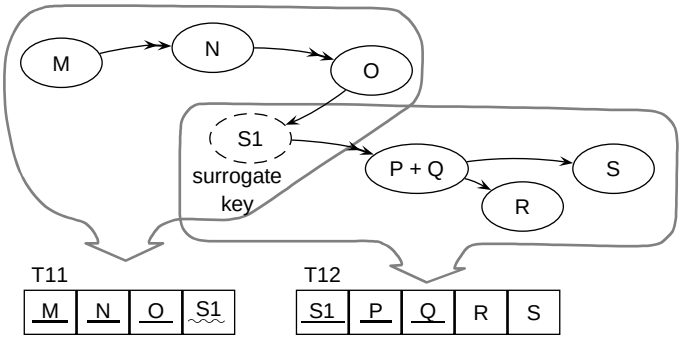


Figure 6: Deriving a relation from an end-key dependency (Smith, 1985, p. 831)



(a) Not practicable



(b) Practicable

Figure 7: Correcting an impracticable FDD (Smith, 1985, p. 832)

3 Issues with deriving foreign keys

3.1 Non-derivable foreign keys

There are some valid foreign keys that cannot be derived using the target bubble rule. Consider Smith's first example, shown in Figure 8; it is clear that CLASS + SECTION in relation R222 is a foreign key to CLASS + SECTION in relation R212. This cannot be derived using the target bubble rule, however, because the bubble containing CLASS and SECTION is not a target bubble. Similarly, STUDENT in R11a is a foreign key to STUDENT in R121, but cannot be derived from the diagram using the target bubble rule because neither of the two STUDENT bubbles are target bubbles.

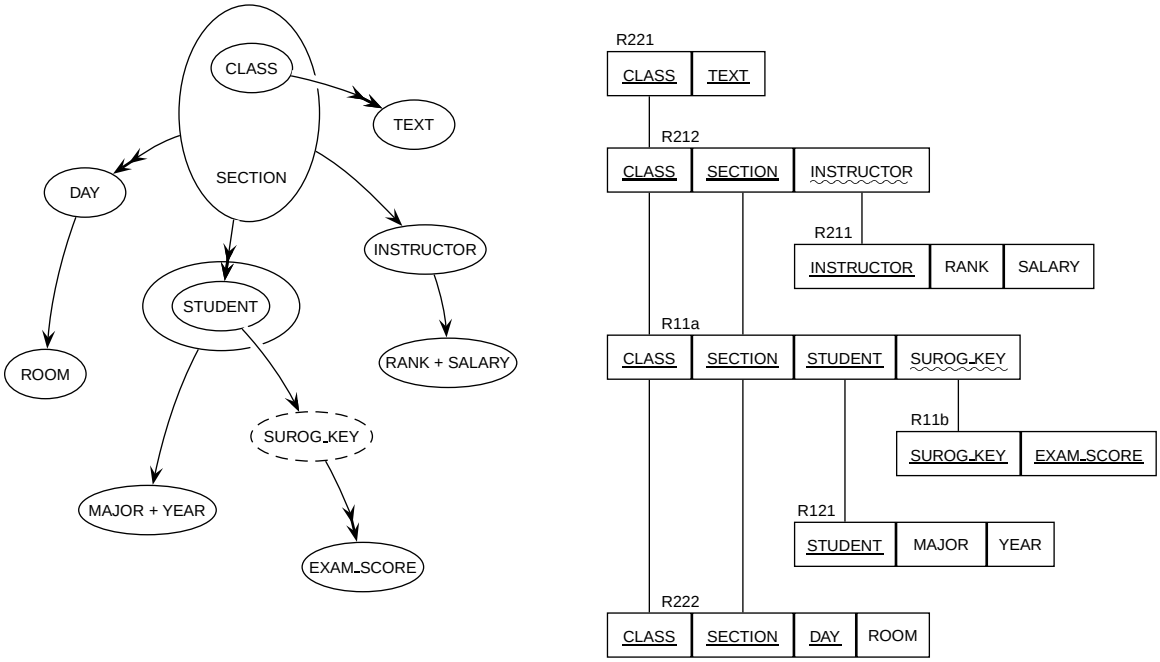


Figure 8: Foreign keys that cannot be derived using the existing rules (Smith, 1985, Figure 3)

3.2 Invalid foreign keys

The target bubble rule can sometimes produce invalid foreign keys. The target bubble rule states that attributes within a target bubble become foreign keys of the resultant relation if they also function as a key bubble. Applying this rule to Smith's second example (Smith, 1985, Figures 4–7) results in several attributes being identified as foreign keys when they are not, such as those highlighted in Figure 9 on the next page.

A foreign key is a set of attributes that act as a link between associated relations. A foreign key links to a candidate key of some relation in the database; typically this candidate key is also the primary key of the relation being linked to (Date, 1995, p. 117). *Referential integrity* states that the value of a foreign key must be identical to a key value in the linked relation, or it must be null (Date, 1995; Elmasri and Navathe, 1994). In Figure 9 the highlighted 'foreign keys' do not reference candidate keys; rather they are referencing only part of the primary key of the referenced relation.

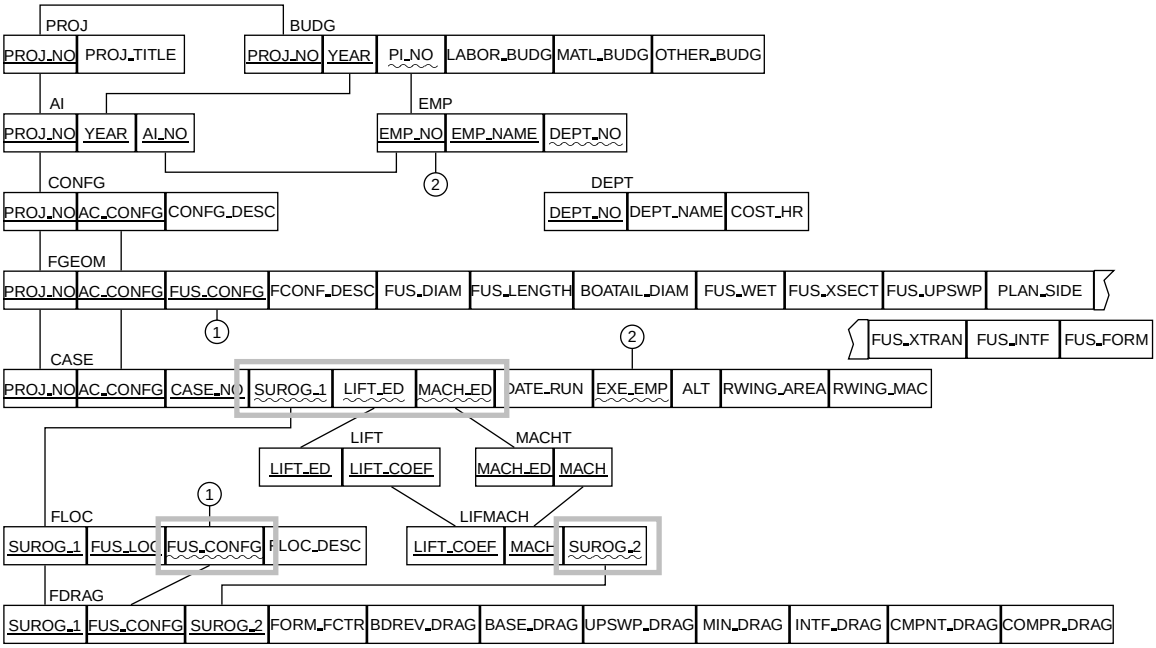


Figure 9: Derivation of invalid foreign keys (Smith, 1985, Figure 7)

This issue arises because the term ‘prime-key bubble’ can be applied to bubbles at the start of both single- and multivalued dependencies. Applying the target bubble rule to a target bubble that is also the prime-key bubble of a multivalued dependency produces the type of ‘foreign keys’ shown in Figure 9. The only effective way to address this issue is to alter Smith’s bubble terminology (see Section 4).

3.3 Automation of the domain flag rule

Smith’s method was an interesting candidate for the author’s research into automated translations among different data modelling representations (Stanger and Pascoe, 1997a; Stanger, 1999). However, the domain flag rule is difficult to automate in its current form. The domain flag rule states that attributes within a target bubble become foreign keys of that relation if they are tagged with a domain flag. When translating a collection of domain flags into relational form, it is required to know which of the tagged attributes is the ‘target’ attribute for the purposes of generating the correct foreign key references (that is, the attribute that the foreign keys will reference). Smith’s FDD notation cannot identify the ‘target’ attribute of a domain flag, so automation of this rule is problematic.

Smith gave no explanation of how to properly treat domain flags, yet in his examples, domain flags are translated correctly. This is possibly because Smith expected the process to be carried out manually and used the dependency-list statements to resolve ambiguities. This is, however, not particularly amenable to automation.

4 Proposed modifications

Smith’s target bubble rule needs to be replaced. To facilitate this change, the author has modified Smith’s original terminology for bubble types. Smith uses the term *prime key* to denote the start bubble of a single-valued dependency, and sometimes the start bubble

of a multivalued dependency. The term 'prime' implies that the prime key attributes are sole determinants of the target attributes. Thus it seems rather counter-intuitive that a prime-key bubble can be part of an uplink key chain, as it is no longer the sole determinant. The following bubble terminology is therefore proposed:

Single-key bubble: the start bubble of a single-valued dependency.

Target bubble: the end bubble of a single-valued dependency.

Multi-key bubble: the start bubble of a multivalued dependency.

End-key bubble: the end bubble of a multivalued dependency.

Isolated bubble: a bubble with no attached dependencies (identical to Smith's definition).

Bubbles may only be of one type. Smith did not enforce this restriction, for example, target bubbles could also be prime-key bubbles, although he did state that the multiple bubbles could be used to clarify such situations. Since the new terminology requires every bubble to be of a single type, multiple bubbles become essential.

Having made these changes, it is now possible to replace Smith's target bubble rule with the following:

Key bubble rule: Let B be a bubble of any type, and R_B be the derived relation to which this bubble contributes. If the attributes of B form the entire contents of a single-key bubble S ($S \neq B$, contributing to a derived relation R_S), then the attributes contained by S become a foreign key of R_B that refers to R_S.

Smith's domain flag rule could remain unchanged, but it cannot be fully automated in its current form. Consequently, the author has introduced the notation shown in Figure 10 to indicate that the tagged attribute is the 'target' attribute for that domain flag. The domain flag rule can now be redefined as:

New domain flag rule: Let B be a bubble of any type containing an attribute A that is tagged with a domain flag, and let R_B be the derived relation to which this bubble contributes. The domain flag is 'targeted' on another attribute D that is the sole attribute contained by a single-key bubble S ($S \neq B$, derived relation R_S). Attribute A becomes a foreign key of R_B that refers to attribute D of R_S.

Figure 10: 'Target' attribute domain flag notation

5 Example

It was originally planned to use Smith's examples to illustrate the new rules in action. However, upon closer examination, it was discovered that neither of the two examples are particularly useful. The first example (that of a University database) is not really

complete enough illustrate the new rules. By contrast, the second example (a drag prediction database) is far too complex, and has so many complicated dependencies among attributes that it is arguable whether a relational implementation is the best solution.

Instead, an example devised by the author will be used. Consider a database that stores assessment marks for a course. An entity-relationship diagram representing this database is shown in Figure 11.

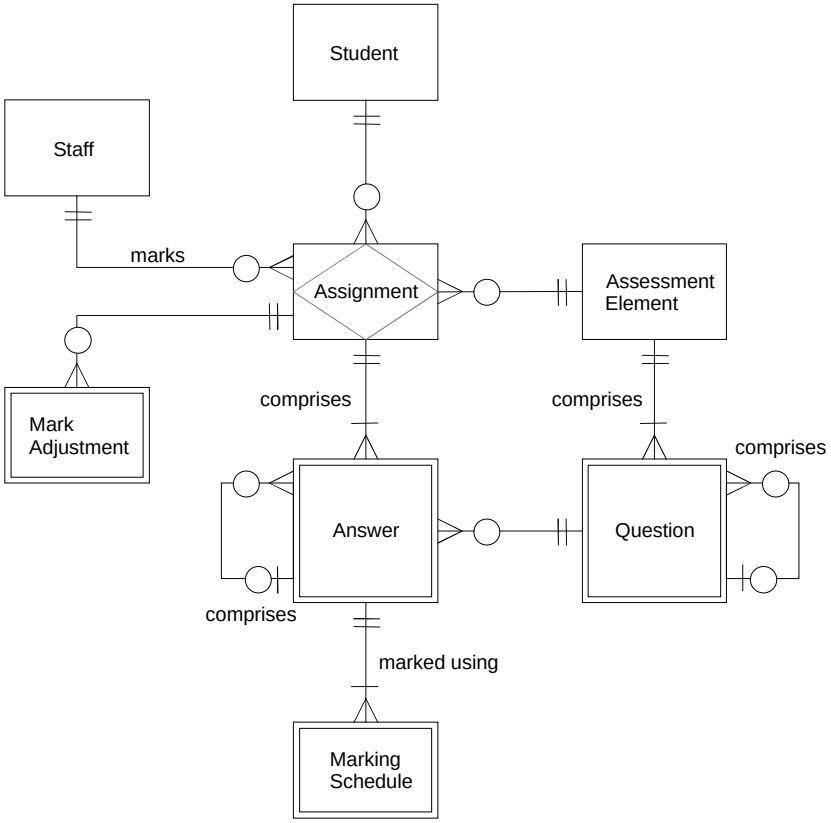


Figure 11: E-R description of the assessment marks viewpoint (normalised)

The final result for the course is determined by the results of a collection of assessment elements (such as practical exercises and examinations), each of which comprises a collection of questions. Each question may or may not comprise a collection of subquestions.

Students individually complete several assessment elements during the course, submitting each as an assignment that is marked by a single staff member. As with assessment elements, an assignment comprises a collection of answers (corresponding to questions), which in turn comprise a collection of sub-answers.

The total mark for an assignment is broken down into a collection of marks for each individual answer. Each answer is marked according to a marking schedule that specifies a set of marking criteria and the marks allocation for each criterion. Marks may be adjusted at a later date for reasons of illness or technical difficulties.

In Figure 12 on the next page is shown a functional dependency diagram for the example, based on the following set of dependencies:

- *student_id* $\rightarrow$ *name, password*
- *staff_id* $\rightarrow$ *name, password*

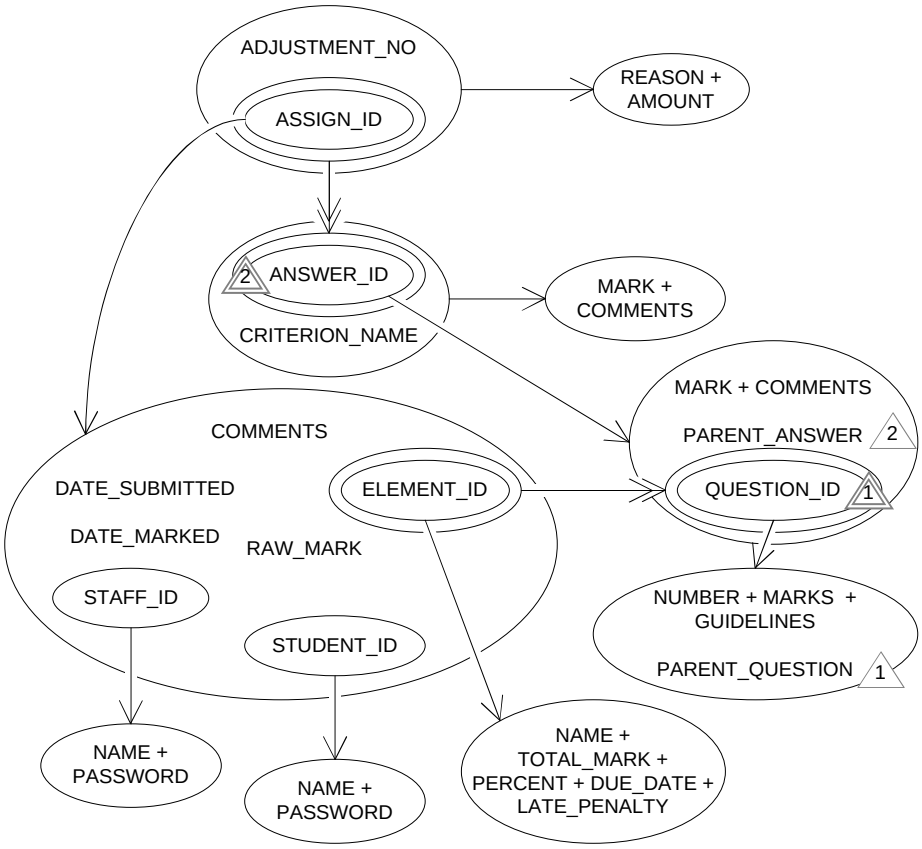


Figure 12: Functional dependency description of the assessment marks viewpoint

- *element_id* $\rightarrow$ *name, total_mark, percent, due_date, late_penalty*
- *element_id* $\twoheadrightarrow$ *question_id*
- *question_id* $\rightarrow$ *number, marks, guidelines, parent_question (question_id)*
- *assign_id* $\rightarrow$ *date_submitted, date_marked, raw_mark, comments, student_id, staff_id, element_id*
- *assign_id* $\twoheadrightarrow$ *answer_id*
- *answer_id* $\rightarrow$ *mark, comments, question_id, parent_answer (answer_id)*
- *assign_id, adjustment_no* $\rightarrow$ *reason, amount*
- *answer_id, criterion_name* $\rightarrow$ *mark, comments*

The last two functional dependencies include the *embedded* multivalued dependencies (Date, 1995, p. 341) *assign_id* $\twoheadrightarrow$ *adjustment_no* and *answer_id* $\twoheadrightarrow$ *criterion_name* respectively. The relations corresponding to this set of dependencies are in at least fourth normal form.

Applying Smith's original foreign key rules to the FDD shown in Figure 12, the following set of relations can be derived (primary keys are underlined):

1. Staff(staff_id, name, password)
2. Student(student_id, name, password)

3. Element(element_id, name, total_mark, percent, date_due, late_penalty)

4. Assignment(assign_id, element_id, student_id, staff_id, date_submitted, raw_mark, comments)
   element_id is a foreign key to Element (target bubble rule)
   student_id is a foreign key to Student (target bubble rule)
   staff_id is a foreign key to Staff (target bubble rule)

5. Adjustment(assign_id, adjustment_no, reason, amount)
   assign_id should be a foreign key to Assignment, but this cannot be derived because none of the bubbles containing *assign_id* are target bubbles.

6. Question(question_id, number, marks, guidelines, parent_question)
   parent_question is a foreign key to Question (domain flag rule)

7. Answer(answer_id, question_id, mark, comments, parent_answer)
   question_id is a foreign key to Question (target bubble rule)
   parent_answer is a foreign key to Answer (domain flag rule)

8. Criterion(answer_id, criterion_name, mark, comments)
   answer_id should be a foreign key to Answer, but this cannot be derived because none of the bubbles containing *answer_id* are target bubbles.

9. Assign_Answer(assign_id, answer_id)
   assign_id should be a foreign key to Assignment and *answer_id* should be a foreign key to Answer, but these cannot be derived because neither of the bubbles involved are target bubbles.

10. Element_Question(element_id, question_id)
    element_id should be a foreign key to Element and *question_id* should be a foreign key to Question, but these cannot be derived because neither of the bubbles involved are target bubbles.

Using the new rules defined in Section 4, the following set of relations can be derived:

1. Staff(staff_id, name, password)

2. Student(student_id, name, password)

3. Element(element_id, name, total_mark, percent, date_due, late_penalty)

4. Assignment(assign_id, element_id, student_id, staff_id, date_submitted, raw_mark, comments)
   element_id is a foreign key to Element (key bubble rule)
   student_id is a foreign key to Student (key bubble rule)
   staff_id is a foreign key to Staff (key bubble rule)

5. Adjustment(assign_id, adjustment_no, reason, amount)
   assign_id is a foreign key to Assignment (key bubble rule)

6. Question(question_id, number, marks, guidelines, parent_question)
   parent_question is a foreign key to Question (new domain flag rule)

7. Answer(answer_id, question_id, mark, comments, parent_answer)
   question_id is a foreign key to Question (key bubble rule)
   parent_answer is a foreign key to Answer (new domain flag rule)

8. Criterion(answer_id, criterion_name, mark, comments)
   answer_id is a foreign key to Answer (key bubble rule)

9. Assign_Answer(assign_id, answer_id)
   assign_id is a foreign key to Assignment (key bubble rule)
   answer_id is a foreign key to Answer (key bubble rule)

10. Element_Question(element_id, question_id)
    element_id is a foreign key to Element (key bubble rule)
    question_id is a foreign key to Question (key bubble rule)

It can be seen from this example that the key bubble rule has allowed the derivation of six foreign keys that could not be identified using the original target bubble rule (in relations Adjustment, Criterion, Assign_Answer and Element_Question). The new domain flag rule has not produced any additional foreign keys, but this is to be expected as this rule was intended primarily to support the automation of Smith's method (Stanger and Pascoe, 1997a; Stanger and Pascoe, 1997b; Stanger and Pascoe, 1997c).

6 Conclusion

Smith's method is a technique that allows the derivation of normalised relations from a functional dependency diagram. Smith's original rules for deriving foreign keys were difficult to automate, failed to produce some foreign keys and could also produce invalid foreign keys under certain conditions. In this paper, new rules that address these issues were defined to replace Smith's original rules. These new rules allow the derivation of all foreign keys, and do not produce invalid foreign keys. To facilitate these rule changes, some modifications were also made to Smith's bubble terminology. These changes have resulted in a robust method for deriving relations from functional dependency diagrams that can be easily automated.

References

Armstrong, W. (1974). Dependency structures of data base relationships, *in* J. L. Rosenfeld (ed.), *IFIP Congress '74 (Information Processing '74)*, North-Holland, Stockholm, Sweden, pp. 580–583.

Beeri, C., Fagin, R. and Howard, J. H. (1977). A complete axiomatization for functional and multivalued dependencies in database relations, *in* D. C. Smith (ed.), *1977 ACM SIGMOD International Conference on Management of Data*, ACM, Toronto, Canada, pp. 47–61.

Date, C. (1995). *An Introduction to Database Systems*, sixth edn, Addison-Wesley, Reading, Massachusetts.

Elmasri, R. and Navathe, S. B. (1994). *Fundamentals of Database Systems*, second edn, Benjamin / Cummings, Redwood City, California.

Smith, H. C. (1985). Database design: Composing fully normalized tables from a rigorous dependency diagram, *Communications of the ACM* **28**(8): 826–838.

Stanger, N. (1999). *Using Multiple Representations Within a Viewpoint*, PhD thesis, Department of Information Science, University of Otago, Dunedin, New Zealand.

Stanger, N. and Pascoe, R. (1997a). Environments for viewpoint representations, *in* R. Galliers, S. Carlsson, C. Loebbecke, C. Murphy, H. Hansen and R. O'Callaghan (eds), *Fifth European Conference on Information Systems (ECIS'97)*, Vol. I, Cork Publishing, Cork, Ireland, pp. 367–382.

Stanger, N. and Pascoe, R. (1997b). Exploiting the advantages of object-oriented programming in the implementation of a database design environment, *Information Science Discussion Paper 97/08*, Department of Information Science, University of Otago, Dunedin, New Zealand.
URL: ⟨*http://divcom.otago.ac.nz/infosci/publctns/complete/papers/dp9708ns.zip*⟩

Stanger, N. and Pascoe, R. (1997c). Exploiting the advantages of object-oriented programming in the implementation of a database design environment, *Joint 1997 Asia Pacific Software Engineering Conference and International Computer Science Conference (APSEC'97/ICSC'97)*, IEEE Press, Hong Kong.
URL: ⟨*http://divcom.otago.ac.nz/infosci/darc/publications/APSEC97.pdf*⟩