Proceedings of the NACCQ 2000
Wellington NZ www.naccq.ac.nz

Database Design and the Reality of Normalisation

Dave Kennedy

Institute of Technology
Christchurch Polytechnic
Te Whare Runanga O Otautahi
kennedyd@chchpoly.ac.nz

ABSTRACT

What is normalisation all about? Why do we teach it? How do we teach it? How can we explain normalisation to our students so that they will understand it?

This paper presents a method of teaching normalisation that, experience has shown, students can understand.

The paper also considers the broader questions of:

♦ Why is normalisation important?
♦ Where does it fit in the process of database design?
♦ How important is it in the "real world"?

Database design can be done using an entity relationship diagram (ERD) - a top down approach or by normalisation of sets of data - a bottom up approach.

The question is, What do real database designers do?

♦ What methodologies do they use?
♦ How important is normalisation?
♦ What normalisation rules do they use i.e. how far do they take it?
♦ How important is denormalisation?

This paper presents a summary of findings, from interviews with database designers, that should help us in our teaching of Database design.

Keywords
Normalisation, database design, dependency diagram

1. INTRODUCTION

"I use a mostly ERD approach to database design but I don't do it unaware of normalisation"

Database designer

What is normalisation all about? Why do we teach it? How do we teach it? How can we explain normalisation to our students so that they will understand it?

This paper presents a method of teaching normalisation that, experience has shown, students can understand. The paper also considers the broader questions of:

♦ Why is normalisation important?
♦ Where does it fit in the process of database design?
♦ How important is it in the "real world"?

The teaching method I use is based on Rob & Coronel (1997). The reality check was made by interviewing three industry people involved in relational database design.

2. NORMALISATION

2.1 What is Normalisation

Normalisation is a set of rules which can be used to modify the way data is stored in tables. (Rob & Coronel, Lecture #5, Date, Date & Fagin).

Normalisation The process of converting complex data structures into simple, stable data structures (McFadden & Hoffer).

There are rules for 1NF, 2NF, 3NF, BCNF, 4NF, 5NF and domain-Key NF. Most textbooks mention 5NF and DKNF only in passing and note that they are not particularly applicable to the design process (Rob & Coronel, pg303, Pratt & Adamski, pg 161, Howe p87). Normalisation is really about the "formalisation of simple ideas" (Date & Fagin). All too often the simplicity is lost in esoteric terminology and papers are "often excessively concerned with the formalism and provide very little practical insight" (Date & Fagin).

2.2 Why Normalisation

Normalisation is about designing a "good" database i.e. a set of related tables with a minimum of redundant data and no update, delete or insert anomalies.

Normalisation is a "bottom up" approach to database design. The designer interviews users and collects documents - reports etc. The data on a report can be listed and then normalised to produce the required tables and attributes.

Normalisation is also used to repair a "bad" database design, i.e. given a set of tables that exhibit update, delete & insert anomalies the normalisation process can be used to change this set of tables to a set that do not have problems.

Another approach to database design is to use Entity-Relationship Diagrams (ERD). This is a "top-down" approach. An Entity is a thing about which we wish to store data. An ERD models the entities, their attributes and the relationships between them. The ERD "rules" are:

1. Each entity has its own table
2. M-M relationships are resolved by creating a composite (bridge) entity which has, at least, the primary keys of its parent entities
3. The aim is to minimise data redundancy.

If we follow the ERD approach and then check the resultant tables against the normalisation rules we usually only need to go to 3NF. If we blindly normalise sets of data then often we will need to go to 4NF.

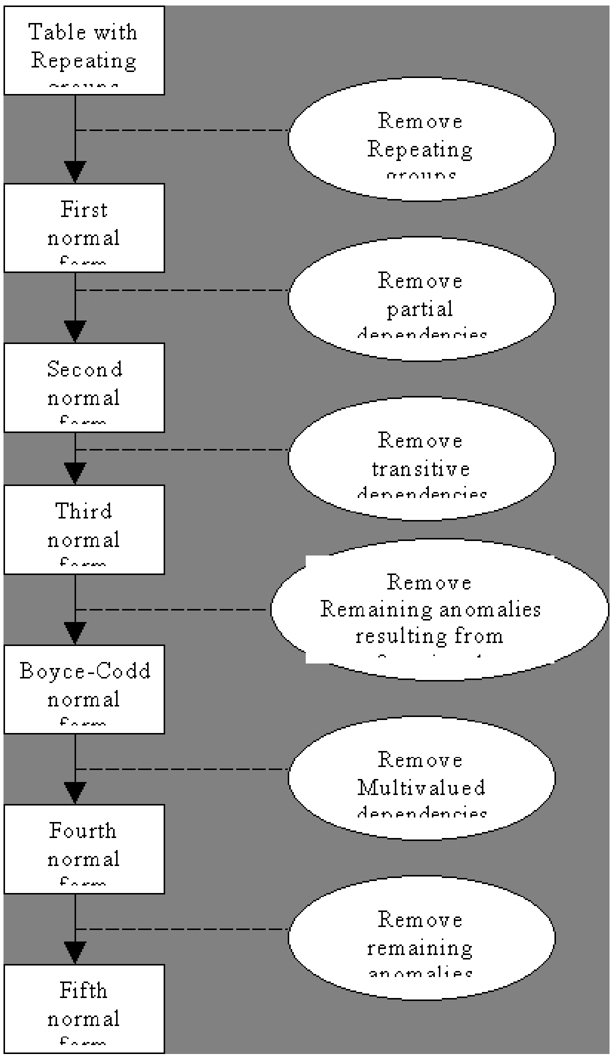

Figure 1. Steps in Normalisation

When teaching normalisation I often find myself thinking "There are 2 or 3 entities involved here and there are various M-M relationships between them - If I sorted that out first then I wouldn't have to go through this complicated normalisation stuff".

Rob & Coronel, pg 226 and Harrington suggest that the best approach is a combined, iterative methodology. Database design is not just about normalisation although normalisation is a useful aid in the process of database design (Date).

2.3 Normalisation - The How

There are many textbooks and websites that attempt to explain the process and the rules of normalisation. There are many different ways to explain normalisation. Some are easier to understand than others. I have found the following (based on Rob & Coronel, McFadden & Hoffer) to be the most understandable.

See Figure 1. Steps in normalisation

2.4 Definitions

1NF	First Normal Form
	atomic values
	Primary Key
	no repeating groups
2NF	Second Normal Form
	There are no partial dependencies
3NF	Third Normal Form
	There are no transitive dependencies
BCNF	Boyce-Codd Normal Form
	Every determinant is a candidate key
4NF	Fourth Normal Form
	There are no multi-valued dependencies

I find these to be the most easily understood definitions of 1NF - 4NF. It is worth noting the Date & Fagin conditions for 4NF and 5NF viz. 4NF - the relation is in BCNF and some key is simple. 5NF - the relation is in 3NF and every key is simple. A simple key is a single attribute key.

2.5 Normalisation - a Teaching Method.

1. Create a table and insert representative data - with as much redundancy as possible
2. Identify the Primary Key
3. Draw a dependency diagram
4. Remove partial dependencies
5. Remove transitive dependencies
6. Check for BCNF i.e. remove any other dependencies that are not candidate keys
7. Remove multi-valued dependencies

Example Table 1

Normalise the following relation to BCNF.
Classes (Staff#, StaffName, {ClassCode {StudID, StudName, Grade, ClassPos}})

A staff member can teach more than one class.
A student can be in more than one class.
A class is taught by only one staff member and can have many students.
Grade is the student's final grade in a specific class.
ClassPos is the student's final position in a specific class - and is unique for that class.

Staff#	StaffName	ClassCode	StudID	StudName	Grade	ClassPos
S101	Smith	PR203A	1000	Peter	B	5
S101	Smith	PR203A	1010	Raewyn	A	2
S101	Smith	DB100B	1000	Peter	B	7
S101	Smith	DB100B	1020	Sue	A	5
S102	Jones	SF100C	1010	Raewyn	A	3
S102	Jones	SF100C	1020	Sue	B	8

etc
The Primary key is ClassCode + StudID.

Table 1

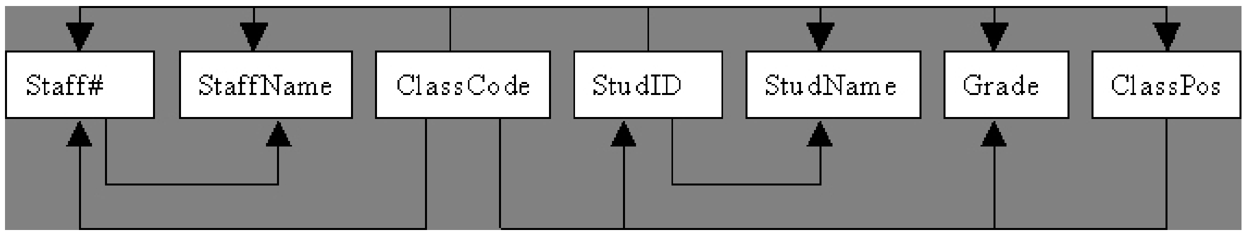

Figure 2. Dependency Diagram

2NF Remove the partial dependencies

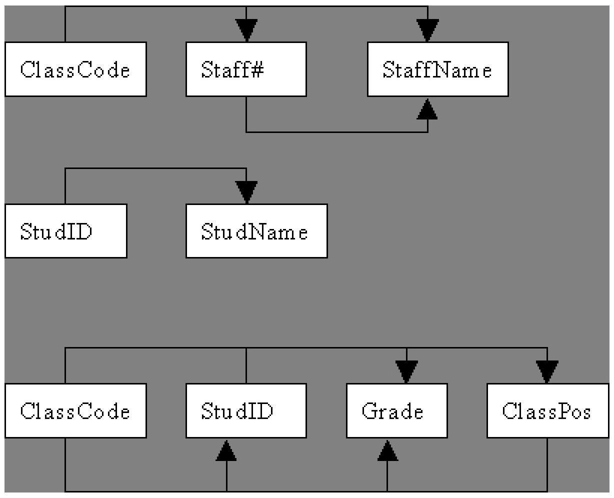

Figure 3. 2NF Dependency Diagram

3NF & BCNF Remove the transitive dependency

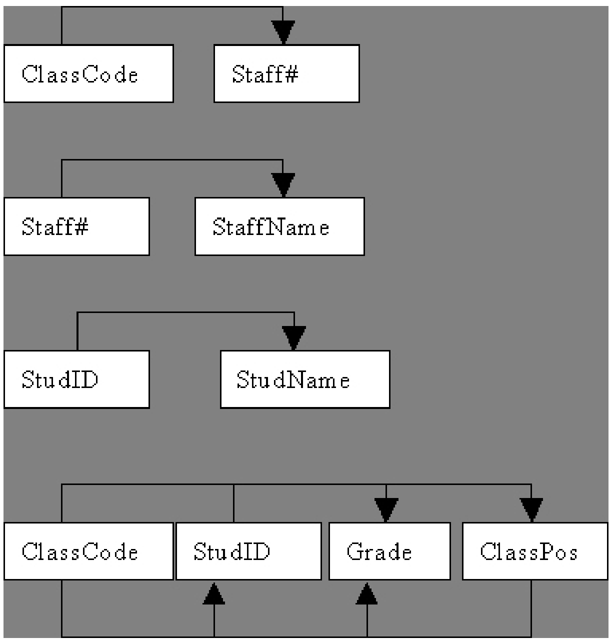

Figure 4. 3NF & BCNF Dependency Diagram

This relation satisfies BCNF because ClassCode + ClassPos is a candidate key

2.6 Normalisation - The Reality

Both Pratt and Date note that in reality we usually eliminate multiple repeating groups at the 1NF stage. This, in effect, is what the ERD composite entity creation does too.

5NF and DKNF are of little practical significance (Howe, Rob & Coronel) and if the ERD and normalization are done in conjunction then usually 3NF is sufficient to produce a good database design (Harrington, Rob & Coronel).

The question is, What do real database designers do?

- What methodologies do they use?
- How important is normalisation?
- What normalisation rules do they use i.e. how far do they take it?
- How important is denormalisation?

2.6.1 Method

Three database designers, from three different companies, were interviewed. They were asked the following questions:

1. What process/methodologies do you use in designing a relational database?
2. How far do you take the normalisation process?
   Why?
   How?
3. Is normalisation important?
4. Is de-normalisation a consideration?
   When?
   Why?
   How?

2.6.2 Summary of Responses

Question 1
What process/methodologies do you use in designing a relational database?

Designer A

- ♦ identify input sources and output requirements
- ♦ identify data catchment points
- ♦ design input screens
- ♦ build the ERD and resolve M-M relationships
- ♦ use auto generators for Primary Keys most of the time especially for tables with composite keys (except when the table is only a bridge)

Designer B

- ♦ Talk to users
- ♦ Identify requirements
- ♦ Build an ERD and resolve M-M relationships
- ♦ I'm very much an entity person
- ♦ Repeat the above steps - take it back to people and ask about scenarios
- ♦ I hate writing one line of code until the data structures are settled
- ♦ Use composite keys in bridging tables
- ♦ Use composite keys in other tables with caution

Designer C

- ♦ Interview users - what do they want - specifics
- ♦ Define outcomes
- ♦ From outcomes work back to define entities and build an ERD
- ♦ Use this to talk to users again
- ♦ Build a prototype - and use this to talk to users
- ♦ Iterative prototyping - the prototype becomes the documentation - used to promote communication and understanding
- ♦ Resolve M-M relationships
- ♦ Auto generated numeric Primary keys - except for bridge entities

Question 2
How far do you take the normalisation process?
Why?
How?

Designer A

- ♦ The method forces 4NF because you have already resolved M-M relationships
- ♦ Auto generated PKs force 2NF
- ♦ Each entity has own table - forces 3NF
- ♦ BCNF doesn't occur

Designer B

- ♦ ERD method resolves 4NF issues
- ♦ I only go to 3NF

Designer C

- ♦ Payback not there to go beyond 3NF
- ♦ The nature of the data is such that it is not often updated

Question 3
Is normalisation important?

Designer A

- ♦ Often a major issue when you take over someone else's work - often the problems are normalisation problems.
- ♦ I use an ERD approach but I don't do it unaware of normalization

Designer B

- ♦ Important that the design is at least 3NF or you will have problems with changes

Designer C

- ♦ I use normalisation when looking at other people's tables
- ♦ Useful in a trouble-shooting role

Question 4
Is de-normalisation a consideration?
When?
Why?
How?

Designer A

- Important to get the tables right before you de-normalise - only then do you know you are doing it safely
- Usually for performance - especially when response times are an issue - i.e. tables involved in the client, customer interface
- De-normalisation is not done often - maybe 2 or 3 tables in a database of 120 tables
- Needs to be controlled - code required for this

Designer B

- Yes - but not often.
- Usually for performance - often money totals are added to a table
- Use triggers to keep database in sync
- I don't de-normalise when using Access because it doesn't have triggers

Designer C

- I don't de-normalise

3. CONCLUSIONS

Normalisation is an important process that database designers need to know. The database designers that I interviewed use an ERD approach to database design but they are aware of normalisation problems. Their approach usually results in tables normalised to 4NF. Normalisation is most useful when analyzing other people's database designs or when there are problems with a database.

If we are to teach normalization so that student's can understand it then we need to keep it simple. I have found that if they add rows of data to a table and then draw a dependency diagram it improves their understanding of the process.

In the light of the interviews conducted for this research I would suggest allowing student's to draw ER Diagrams in conjunction with the normalisation process.

The normalization process should also eliminate multiple repeating groups at the 1NF stage thus avoiding the need to check for 4NF. It is important to note that in reality database designers add auto generated numeric Primary keys to most tables, which ensures at least 2NF.

4. REFERENCES

Date C. J. (1986) "An Introduction to Database Systems". V 1, Fourth edition, Addison-Wesley.

Date, C. J. & Fagin, R. (1992) "Simple Conditions for Guaranteeing Higher Normal Forms in Relational Databases". ACM TODS, V 17, 3, Sept, pp 465 - 476.

Harrington, J. L.(1998) "Relational Database Design Clearly Explained". AP Professional.

Howe, D. R. ((1983) "Data Analysis for Data Base Design". Edward Arnold.

Lecture #5 (2000) Data Normalisation. Accessed April 17, 2000. <http://phoenix.ucr.edu/mis/mgt230/lecture5/index.html>

McFadden, F. R. and Hoffer, J. A. (1994) "Modern Database Management". Fourth Edition, The Bejamin/Cummings Publishing Company.

Pratt P. J. & Adamski J. J. (1987) "Database Systems: Management and Design". Boyd & Fraser.

Rob P. & Coronel C. (1997) "DataBase Systems, Design, Implementation and Management", Course Technology.