Importance of relational database normalisation in ecological projects.

F. H. Yung, P. J. de Tores and S. A. Halse

Abstract

The concepts of database theory have been developed since the middle 1960s using three main approaches, namely Hierarchical, Network and Relational. The Relational model is definitely superior and is now universally accepted as the standard in database technology. Biological scientists have been slow to appreciate the power of this model. Often scientists use the software only as an electronic storage of data in the form of spread sheets. Ironically this amounts to using only about one-fifth of the capability of Relational database technology.

This paper reports the experience gained from three ecological projects. It concludes that the application of Normalisation principles allows Relational databases to become a much more effective tool for ecologists. It is best to consult an information scientist at the design stage of ecological projects. It advocates that an introductory unit in this technology should be included in the undergaduate curriculum of most biological science courses.

Introduction

The developments of database theory started in the middle 1960s. Earlier approaches favoured the Hierarchical and Network models. This was partly due to the lack of power of early computers. The pioneering work by the mathematician E. F. Codd, the originator of the Relational database model, began in 1968 (Codd, 1970).

The Relational model is more demanding on computer resources. As the power of mainframe computers and PCs grew, the performance limitations diminished quickly. This opened the way for the Relational model to become the most attractive practical option.

The Relational model, with the robust mathematical foundation layed by Codd, is now considered superior (Date, 1986) and universally accepted as the standard in database technology. It is the most important development in the history of database technology and fuelled the spectacular information technology boom in this century.

Various industries have been quick to take advantage of this new technology. Nowadays most of the advanced computer aided design packages include a Relational database in their core. Artificial intelligence systems require information storage and retrieval more than number crunching and therefore rely heavily on database technology. As a result of the PC boom, research scientists in most organisations have at least one PC for his/her exclusive use. Many of the Relational database softwares are affordable, for example, SIR, dBase III, Paradox, MS Access and Oracle.

However, biological scientists must be wary of the temptation to think that "off the shelf" relational database softwares can automatically design an effective Relational database. It is a common misconception. To date these softwares alone can not automatically design a proper Relational database, which the tables are Normalised. The designer needs to consciously apply the principles of Normalisation. A thorough understanding of the mathematical foundation of the Normalisation theory takes at least

one semester in a computer science course and is obviously not a recommended course for all biologist to take. However, an introductory course especially designed for him as a user would be a distinct advantage. The logical approach for the biologist is to consult an information scientist, just as he/she would consult a biometrician.

If an information scientist is not consulted, a poor or inefficiently designed database is likely to result.

This paper first gives a brief account of the Normalisation principles in Relational database theory. By analysing the database of one ecological project and drawing from the experience of two others, it concludes and advocates that when designing such ecological projects an information scientist should be consulted before data collection. A well designed database should reflect every feature of the data collection procedure in the field.

The basic principle of database Normalisation

Some of the relevant terms in the database Normalisation process are:

Repeated group: **(Mike, this seems to be more of an explanation of the 1NF and is explained more clearly in the descriptions for the tables in figure 2A?? Maybe best to revamp this table - and now referred to (below) as Figure 1).**

Key: In a table, a Key is a group of fields whose values together is unique in that table. Other fields are called non-key attributes.

Partial Dependence: If a non-key attribute depends on only some and not all the key fields, then this is called a Partial Dependence.

Transitive Dependence: If a non-key attribute can be determined by knowing the values of some other non-key attributes, then this is called a Transitive Dependence.

Database Normalisation is a theoretical process that tests whether a table (for example, a simple table consisting of only rows and columns, without repeated groups, where all rows are equivalent) has any undesirable dependence between its columns, namely the partial and transitive dependence. If a table has these types of data dependence, then it should be non-loss decomposed (i.e. broken down into two or more tables without loss of information), until all these types of dependence are removed.

An example of a table in the first normal form (1NF) is shown in Figure 1 **(see above ?)**. A table in the 1NF may contain both types of dependence described above. After removing both types of dependence, the resultant table is said to be in the Third Normal Form (3NF) see Kendall and Kendall (1992).

There are higher Normal Forms, namely the 4NF, and 5NF, but they are not discussed here as the 3NF is sufficient to illustrate the essence of the process. The interested reader can refer to Date (1986) for formal discussions.

To illustrate why Normalisation is important in designing a database, the simple database example in Fig 2A is used.

Table Q, P, QP are the Quadrats, Plant Species and Quadrat Counts tables, respectively. Each is normalised and in the 3NF. The characteristic feature of these tables is that each

table contains only the relevant information about that entity. For example, the Quadrats table does not contain information about the plant species found in the quadrats. To demonstrate the disadvantages of using Unnormalised tables we merge table Q and QP together into a table QQP shown in Fig 1B. That is, we use only the table QQP and P as our database tables. QQP is in the 1NF only. To test the behaviour of the table QQP, we must consider three operations: "updating", "deleting" and "inserting" data.

Updating: If the soil index of Q1 needs changing from high to medium, then the first three rows in QQP need to be changed consistently, otherwise there will be contradicting data.

Deleting: To make a correction that requires deleting the row involving Q3 would result in deleting all the information about that quadrat, therefore deleting more information than required. The problem can be partially solved by putting a null value into the field "count" and therefore by-pass the problem in an *ad hoc* manner and not delete the record. However, this is not desirable as it introduces an extra rule for the user to remember making the database more problematic to use.

Inserting: To add a new quadrat (Q5) into table QQP, would result in not having a valid "plant species recorded" or "count" value to enter. A null value could not be inserted in the "plant species recorded" because it is a key field.

Therefore, using Unnormalised tables without making mistakes usually requires the user to remember an extra set of rules making life miserable. Furthermore, if the rules are broken there is the potential of a loss of information. For a properly designed database using Normalised tables, these rules (Data Integrity Rules and Business Rules) are easily taken care of automatically.

Prey species trapping database - a case study

In this case study the data semantics of a real ecological project are examined. The project is designed to measure native fauna (prey) response to the different levels of predator density reduction.

The project's database stores the trapping data for captures of reptiles, amphibians, small and medium size mammals. There are up to 13 trapping sites (grids) within each of 4 treatments - each treatment representing a different level of predator density reduction.

The case study highlights the disadvantages of using Unnormalised tables and demonstrates the advantages gained if Normalised tables are used.

The Unnormalised table

If we simply list the data fields we need, we have a table with the structure similar to that shown in the "captures table" in Figure 3. The table is not in 1NF. Each row is a capture record.

Difficulties in the use of the table arose from the requirement to uniquely mark each animal caught. As there is no single method applicable to all groups of animals, a range of marking methods was employed. For example, medium size mammals (such as possums and bandicoots) were identified by a Trovan tag (an small inert passive transponder implanted under the skin). Small mammals (such as a marsupial mouse) were identified by an ear punch and reptiles and amphibians identified by toe clipping. Similarly, a different set of physical measurements was recorded for each group.

For small and medium size mammals, there are different degrees of dependence on the key fields, leading to transitive dependence, an unwanted dependece explained above.

Every capture event required data entry in the fields "date", "trapping session", "grid", "species", "animal ID", "new or recapture", and "weight". Further, for a particular animal group (i.e. reptile, amphibian, small mammal or medium mammal) data entry was required for a subset of the remaining fields. Specifically, a reptile capture required entry in the additional "snout-vent length", "head-tail length", and "toe clip number" fields. An amphibian required entry of data in the additional "snout-vent length" and "toe clip number" fields only. A small mammal required entry in the additional "sex", "head-body length", "tail length", "ear punch number" and "pouch" and a medium size mammal required entry of data in the additional fields of "Trovan", "sex", "head length" " ", "pes length" and "pouch".

For any particular capture record many of the fields were left blank. If one of these fields was a key field, such as "sex", and was left blank as in the case for reptiles, the data base software would (as is the case for most Relational database softwares) prompt the operator with an error message advising that an entry is required. In this case a "dummy" entry was necessary.

In the case of small mammals, the information contained in the "sex" field is also given in the "animal ID" field. However, this was a deliberate feature, designed to maximise use of the standard numbering system for small mammals. For example, for any particular trapping grid, each small mammal is given a unique ID, such as Sd Wea M09. In this case the ID identifies the animal as a small dasyurid marsupial, *Sminthopsis dolicura* (Sd), caught at Wearne grid (Wea), and is male (M), number 9 (09). As there are only 99 unique numbers available for *Sminthopsis dolicura* at Wearne grid, the extra character for sex (in this case M) was added to effectively double the number of unique IDs available.

Similarly there was dependence between the fields "new or recapture" and "date". For recaptures (animals caught for a second or subsequent time) the "date" field was also required to make the record unique, as there may be many records for any particular animal. Therefore it was necessary to make both these fields key fields.

Data entry was required to obey a complicated set of Business Rules. Consequently, the table was cumbersome and suffered from the disadvantages described in the QQP table (Fig. 2). The Business Rules were not automatically enforced by the system and had to be enforced during data entry.

The table was not in the 1NF and the major drawbacks of this design were:

1. Entering data in only a subset of fields from a large table;
2. Multiple entry of data - each capture event at a particular grid required entry of "date", "trapping session", and "grid" although these would be identical in many cases;
3. When coding a query to extract information, it was necessary to remember all the Business Rules. The query could be as cumbersome as data entry and more complicated than necessary.

Use of queries, using only the necessary fields, alleviated the problems in 1 - i.e. a query was generated to enter data for each group of animals. For example, the reptile data entry query had only the fields "date', "trapping session", "grid", "species", "new or recapture", "weight", "snout-vent length", "head-tail length" and "toe clip number". None of the small or medium mammal fields was visible when reptile data were entered. However this did not facilitate information extraction querying.

Realising all these difficulties from the start, the tables were Normalised.

The Normalised design

The outline of the final design is shown in Figure 4. The software used was MS Access V2. The design consists of 4 Normalised capture tables. The Forms facility is used as the front end. This Forms facility uses the relationships to link more than one table at a time and displays the selected fields, with Referential Integrity rules enforced. This makes the database system more user-friendly.

The first Form enabled the data entry operator to enter the fields "trapping session", "grid" and "date". These data are then automatically entered in the relevant table. Subsequent entries for the same trapping session, at a given grid and date do not require re-entry of these data. The species caught is then selected from a look up list and the relevant capture table accessed, displaying all the fields for that group of animals.

It must be emphasised that the Normalised tables are the foundation of a database. If the tables are not Normalised, using softwares such as MS Access or Paradox is not going to solve the problems.

Other properly designed databases

We have also reviewed two other projects.

One project concerns the survey of waterbird usage of wetlands including nesting and brooding information. This database contains more than two megabytes of data with the largest file containing about 54,000 lines. The four database tables were initially only partly Normalised into the 2NF. In view of that, the first author carried out the Normalisation to the full, ie all the tables in the 3NF. The initial four tables have to be decomposed further into six tables and the system was written in Paradox for Windows on a 486 PC. After that, the user enjoys much more efficient extraction of information, as reported in our example above.

The other project is a river health monitoring project involving 200 observation sites. Data collected includes the gross information about each site, the habitats within each site, chemicals and invertebrates found in each habitat. The project leader particularly appreciates the power of Relational database technology and sought input from the first author at the early stage when the field sheet was designed. So the data semantics was also analysed and the database was built with properly Normalised tables on a 486 PC before field data collection. Seven tables contain the collected data.

The final design is shown in Fig 5. Data in the tables are as follows:

Sites	contains gross information about the sites.
Rounds	contains information about the four data taking rounds.
Site Round	solely to represent the relationship between tables Site & Round.
Site Chem	contains the chemical data of the sites, measured while on site.
Lab Chem	contains the chemical data of the sites, which cannot be measured on site, but determined in the laboratory
Habitats	contains the data about the habitats
Invertebrates	contains the abundance data of the invertibrates identified.

Obviously, there is a many-to-many relationship between tables Sites and Rounds. The proper design technique is to form a third table Site Round, hence the many-to-many relationship becomes two one-to-many relationships making Data Integrity easily enforced in most softwares. This design feature is also reflected in Fig 1A (?) above,

except in which the design was arrived at in a more natural way because of data semantics.

Other relationships look simpler, but data semantics can still be rather subtle. The table Site Chem is at the top of a tree structure, reflecting the data collection process closely. For instance, a site contains different habitats, in each of which invertebrates were identified.

You may wonder why tables Site Chem and Lab Chem are not combined into one. This is also because of the data collection process. Certain physical and chemical parameters of the water such as temperature, conductivity and pH are measured on the spot when the observer is at the site. Other properties such as turbidity, alkalinity, nitrogen and phosphorus contents are not known until the water sample is analysed in the laboratory weeks later. Therefore, it is more convenient to store these two types of data in separate tables. This is a good example that a user friendly database simulates the data collection process reasonably closely. Any logical property of the relationships that contradicts the data collection process is a potential problem area.

To date, no major change was required, except minor enhancements. This system is now used by four separate organisations in Western Australia. Data sets have also been extracting smoothly for statistical analysis and posted interstate for various mathematical modelling.

Conclusion

It is clear that for any non-trivial ecological project involving data collection, there is likely to be a significant time loss during data entry, maintenance and querying if appropriate Relational database technology is not used. The Business Rules, which should have been automated with minimal programming, may become a significant burden. There are numerous anecdotal accounts claiming that using such a badly designed system is off-putting for the data entry operator. A query can take longer than necessary resulting in user anxiety and low morale.

The Relational database technology should be used whenever possible. Tables should be properly Normalised by the database designer, who has a clear understanding of the meanings of all the fields and the data collection process. It is logical to consult an information scientist at the beginning of the project and is as important as a consultation with the biometrician.

Here we further advocate that a short introductory unit of the Relational database technology should be included in most biological science courses. This may suffice to prepare him to be an efficient user in his later career.

It is appropriate to point out that database Normalisation is not the panacea. To design, construct and implement a database system successfully, it is essential to use a sound phased Systems Analysis procedure. With this procedure, satisfactory communication between the ecologist and the information scientist can be checked at the end of each phase. This guarantees that the final database fulfils all the requirements of the ecological project.

Acknowledgment

The authors wish to thank Mr Nicholas Lander for encouraging this work; Julie A Raines and R I T Prince for beneficial discussions in ecology and sharing their experience in using databases; Irene G H Yung, Stephanie H T Yung and Suzanne Rosier for editing; Nicole Dennis for typing Fig ? **(can we do Fig 4 on Powerpoint???)**.

This study was funded by the Department of Conservation & Land Management and (???).

References

Codd E. F. (1970). A Relational Model of Data for Large Shared Data Banks, Communications of ACM, Vol 13, No 6, June 1970.

Date C. J. (1986). An Introduction to Database System, Vol 1, 4 th Ed., 19-20, 99-100.

Kendall K. & Kendall J. (1992), Systems Analysis & Design, 2 nd Ed., Englewood Clifs.

LIST OF REVIEWERS SUGGESTED ???

Ecologist:

Dr A H Burbidge, Principal Senior Research Scientist, Dept of CALM, P.O. Box 51, Wanneroo, phone 4055109.

Dr Neil Gibson, Senior Research Scientist, Dept of CALM, P.O. Box 51, Wanneroo phone 4055139

Information Scientists:

Dr Colin Pearce, Branch Manager, Information Management Branch, Operations HQ, Dept of CALM, Hayman Rd, phone 3340343

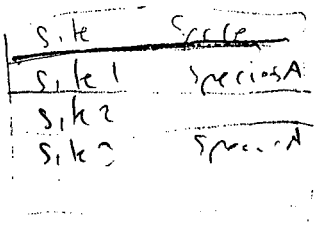
Mr Warwick Boardman, Application Manager, Information Management Branch, Operations HQ, Dept of CALM, Hayman Rd, phone 3340472.

Mr James Staddle, Former Head of Section, Personnel Information Management System Project Group, W A Government, phone 4487845, 2222555 work (W A Government).

PdeT with record

Figure 1: An example of a table with repeated groups ----- show how should be flat and 1NF????

sample site	species recorded
site 1	species 1
	species 2
	species 3



Appendix. ~~Some important terms in the database Normalisation process~~

Repeated Group: For example, in an improper table with blank cells like

(where to put this?)

Salesman	Customer
salesman 1	customer 1
	customer 2
	customer 3

the field Customer forms a Repeated Group. To remove it, the value of Salesman must be repeated in the other two rows, ie a proper Table with only rows and columns without other structures. Such a proper Table is said to be in the First Normal Form (1NF) and is also said to be flat.

Figure 2A: An example of a simple Relational database consisting of Normalised tables in 3NF (Key fields are in italics and bold).

Table P: Plant species

Plant species code	Plant species name	Type
P1	Black boy	indigenous
P2	Jarrah	indigenous
P3	Marri	indigenous
P4	Cape Tulip	introduced
P5	Cape Weed	introduced

Table Q: Quadrats

Quadrat no.	Habitat	Soil index
Q1	Tuart Forest	high
Q2	Banksia woodland	low
Q3	Banksia woodland	medium
Q4	Coastal Heath	low

Table QP: Quadrat counts

Quadrat no.	***Plant species recorded***	Count
Q1	P1	120
Q1	P2	200
Q1	P5	40
Q2	P1	98
Q2	P2	84
Q3	P2	104
Q4	P3	30

Figure 2B: Combining Tables Q and QP into table QQP (1NF only)

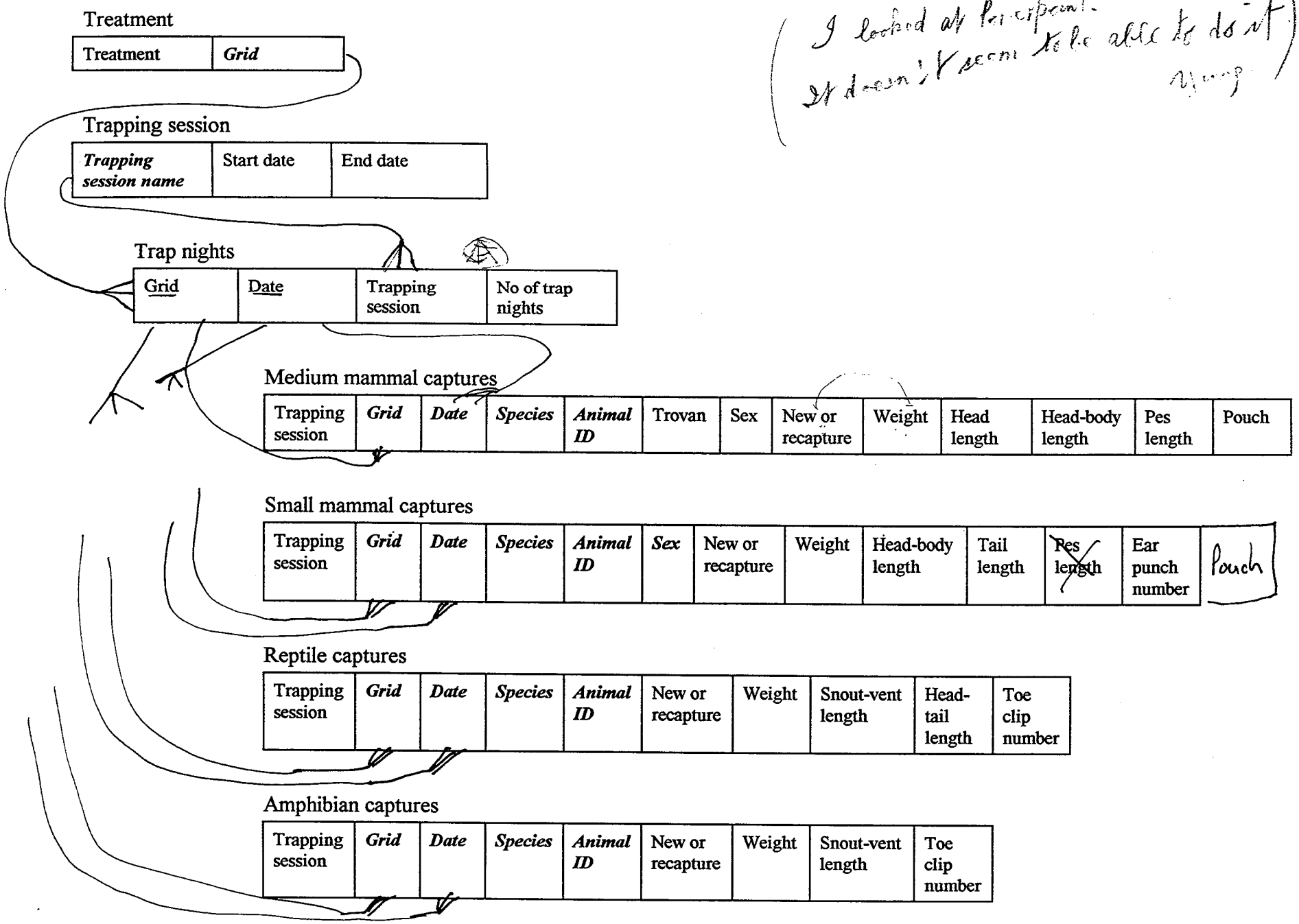
Quadrat no.	Habitat	Soil index	***Plant species recorded***	Count
Q1	Tuart Forest	high	P1	120
Q1	Tuart Forest	high	P2	200
Q1	Tuart Forest	high	P5	40
Q2	Banksia woodland	low	P1	98
Q2	Banksia woodland	low	P2	84
Q3	Banksia woodland	medium	P2	104
Q4	Coastal Heath	low	P3	30

Figure 3: *** (Key fields are in italics and bold).

Captures Table

Date	Trapping session	***Grid***	***Species***	***Animal ID***	***New or Recap.***	Sex	Trovan	Weight	Snout-vent length	Head-tail length	Toe clip number	Head length	Head-body length	Pes length	Tail length	Ear punch number	Pouch

Figure 4: Main features of the final Normalised design for the trapping database. The rectangular boxes are the Normalised tables with the key fields in bold and italic. Other fields are shown only if illustrative. Relationships between tables are indicated by linking lines. An one-to-many relationship is represented by a line with a branching sign at one end.



Treatment

Treatment	***Grid***

Trapping session

Trapping session name	Start date	End date

Trap nights

Grid	Date	Trapping session	No of trap nights

Medium mammal captures

Trapping session	***Grid***	***Date***	***Species***	***Animal ID***	Trovan	Sex	New or recapture	Weight	Head length	Head-body length	Pes length	Pouch

Small mammal captures

Trapping session	***Grid***	***Date***	***Species***	***Animal ID***	***Sex***	New or recapture	Weight	Head-body length	Tail length	Pes length	Ear punch number	Pouch

Reptile captures

Trapping session	***Grid***	***Date***	***Species***	***Animal ID***	New or recapture	Weight	Snout-vent length	Head-tail length	Toe clip number

Amphibian captures

Trapping session	***Grid***	***Date***	***Species***	***Animal ID***	New or recapture	Weight	Snout-vent length	Toe clip number

Figure 5. Cost-benefit estimates of the Trapping DB, when Normalisation is used. this project goes on for longer than two years

No.	Credit (+manweek)	debit (- manweek)
1	Systems analysis, User participates also: no saving	Information scientist does - systems analysis: 2 (-1.5 mw): -3 mw Normalisation: -1 mw
2	User setting up DB without Normalisation & MS Access Forms, all this work exempted: saved: 2 mw	Information scientist does programming with MS Access Forms (including design) : -2 mw
3	User doing data take-up, all this work exempted: saved: 2 mw	Data take-up (includes improving data defn): -0.2 mw
4	User does beta testing also: no saving	Writes User Guide & does beta testing: -2 mw
	User work saved (sub. tot.) = **4 mw**	Extra work done = **-8.2 mw**
5	data entry: saves 60 %	
6	Update & Querying: saves 60 %	
7	Spin-off (synergy): telemetry data 7 tables were also linked up to this system, saves 40 %	
8	**Summary of advantages**: -clearer meaning for each data field -data easier to maintain with integrity -database system easier to maintain -User can concentrate on biological science **(invaluable)**	

Figure 6. The Relational database model for a river health monitoring project. The notation is the same as in Fig 2.