RELATIONAL DATA ANALYSIS
(or NORMALISATION)

Lecture Notes:

1. What is Relational Data Analysis ?

Relational Data Analysis, or Normalisation, is a technique used extensively for database design, it is based on work developed by Codd, which originally aimed to automate the process of data design. This did not happen, but the technique, is widely used by analysts. Codd developed 3 stages of normalisation known as 1st, 2nd and 3rd normal form. These have been extended to include 4th and 5th normal form, however we will only be looking at the first 3 as these are the most widely used.

It is a *'bottom up'* technique, based on an analysis of individual data items and their relationships, (deduced from the way they are grouped together in documents). This is distinct from Logical Data Structuring (or Entity Relationship Modelling) which is a *'top down'* approach - it starts from a high level and then begins to look at the underlying detail. Logical Data Structuring is an intuitive, subjective technique, whereas Relational Data Analysis is a formal, mathematically based technique - it is based on relational algebra and calculus.

2. Where does Relational Data Analysis Fit in SSADM ?

If we were doing a "proper" SSADM analysis, we wouldn't do Relational Data Analysis straight after Logical Data Structuring. Logical Data Structuring comes in Stage 2 of SSADM, "Specification of Requirements", whereas Relational Data Analysis comes in Stage 4, "Data Design". However, for teaching purposes, it assists understanding if we look at the two techniques one after the other.

3. Steps in Relational Data Analysis

3.1 Extract Data from Data Source & Represent it in Unnormalised Form

The data source will usually be an input or an output of the system, whether a screen, a form, a format or a report. Let's take a student module registration form, (from a system to administer a modular degree scheme).

Student Name: Geoff Crane		**Reg #:** 123456789
Course: Biochemistry		**Year:** 3
Module Code	**Module Name**	
GN 301 GN 302 GN 303 . . . etc	Introduction to Genetic Engineering Advanced Genetic Engineering Social Consequences of Genetic Eng . . . etc	
Project Details		
Project Code: PR370/94		
Project Title: Building a Group of Friends		
Project Supervisor: Frank N. Stein		

To represent that data in unnormalised form, simply write the *name* of each data item, showing where there are repeating groups. I do this by enclosing them in brackets, which may be nested if there are repeating groups in the repeating groups. So, for the above document:

UNF
STUDENT
Student Number
Course
Year
Student Name
(Module Code
Module Name)
Project Code
Project Title
Project Supervisor

When you have your data in UNF you need to pick a key. This should ideally be unique on the data source, but you may have to use a combination of items to get a unique identifier. In theory, the key can be anything, but it makes sense to have a "reasonable" key, and to avoid textual keys. So here, although Project Code is unique on the document, and could in theory be used as the key, it makes more sense to use Student Number, as student is what we are storing data about we have underlined it.

3.2 Move from Unnormalised Form to First Normal Form

To go to 1NF, take out the repeating groups, *taking the key with them*, and remove them to separate "relations", and identify new key(s) for them as well, so that they too are uniquely identifiable.

So for our example:

UNF	1NF
STUDENT	**STUDENT**
Student Number	Student Number
Course	Course
Year	Year
Student Name	Student Name
(Module Code	Project Code
Module Name)	Project Title
Project Code	Project Supervisor
Project Title	
Project Supervisor	**STUDENT_MODULE**
	Student Number
	Module Code
	Module Name

3.3 Move from First Normal Form to Second Normal Form

To go from 1NF to 2NF, just look at the relations with more than one key. Check that each data item within them *depends on all keys*. If it doesn't, remove it, together with the key(s) on which it does depend, to a new relation.

In our example, we only have one relation with a compound key:

Student Number
Module Code
Module Name

To work out whether the data item (in this case), depends on both keys, try thinking about the actual data values. If the data item depends on a key, its value will change if there is a different value for the key. So, looking at this relation, we can consider the value of Module Name with different keys;

Student Number:	123456789
Module Code:	GN301
Module Name:	Introduction to Genetic Engineering

With a different Student Number, we would get the following values:

Student Number:	987654321
Module Code:	GN301
Module Name:	Introduction to Genetic Engineering

With a different Module Code, we would get the following values:

Student Number:	987654321
Module Code:	GN302
Module Name:	Advanced Genetic Engineering

Therefore, we can see that Module Name depends *only* on Module Code, so our 2NF looks like this:

UNF	1NF	2NF
STUDENT	**STUDENT**	**STUDENT**
Student Number	Student Number	Student Number
Course	Course	Course
Year	Year	Year
Student Name	Student Name	Student Name
(Module Code	Project Code	Project Code
Module Name)	Project Title	Project Title
Project Code	Project Supervisor	Project Supervisor
Project Title		
Project Supervisor	**STUDENT_MODULE**	
	Student Number	**STUDENT_MODULE**
	Module Code	Student Number
	Module Name	Module Code
		MODULE
		Module Code
		Module Name

We are left with a relation containing only the codes for student and module, but this is fine; we may well find data associated with such a relation on another document, (such as a record of student marks), and in any case, we do need to know which student is taking which module.

3.4 Move From Second Normal Form to Third Normal form

To go to 3NF, we examine all relations, to check that the data items each contains depends on the key(s), rather than on another data item. The test is a similar one as to 2NF - would the data item's value change if the value of another data item was changed ?

So for our example, we can see that Project Title and Project Supervisor would change if Project Code were altered. These data items should therefore be removed to another relation, using the data item they depend on as their key. This data item should be left in the original relation, and should be marked as a foreign key, thus:

UNF	1NF	2NF	3NF
STUDENT	**STUDENT**	**STUDENT**	**STUDENT**
Student Number	Student Number	Student Number	Student Number
Course	Course	Course	Course
Year	Year	Year	Year
Student Name	Student Name	Student Name	Student Name
(Module Code	Project Code	Project Code	***Project Code #***
Module Name)	Project Title	Project Title	**STUDENT_MODULE**
Project Code	Project Supervisor	Project Supervisor	Student Number
Project Title			Module Code
Project Supervisor	**STUDENT_MODULE**	**STUDENT_MODULE**	
	Student Number	Student Number	**MODULE**
	Module Code	Module Code	Module Code
	Module Name		Module Name
		MODULE	
		Module Code	**PROJECT**
		Module Name	Project Code
			Project Title
			Project Supervisor

The relations defined in 3NF are said to be normalised.

Now try and do this with the following examples:

Drug Card

Patient No: 923 **Surname:** Moneybags **Forename:** Maurice

Ward No: 10 **Ward Name:** Barnard

Drugs Prescribed:

Date	Drug Code	Drug Name	Dosage	Time
20/5/94	CO2355P	Cortisone	2 pills 3 x day after meals	14 days
20/5/94	MO3416T	Morphine	Injection every 4 hours	5 days
25/5/94	MO3416T	Morphine	Injection every 8 hours	3 days
26/5/94	PE8694N	Penicillin	1 pill 3 x day	7 days

Staff Allocation Sheet

Project Code: 3411 **Project Description:** New Accounts
Customer Number: 3475 **Customer Name:** British Bakers

Staff No	Name	Grade	No of Days
34	Bloggs	S. Prog	12
12	Jones	Analyst	3
23	Brown	Manager	9
45	Williams	Teaboy	32

Invoice

Invoice Number: 3412 **Date of Invoice:** 23/10/94
Customer Number: 3475 **Customer Name:** British Bakers

Customer Address: Bread House, Albert Sq, London, W14 3RT

Proj Desc	Start Date	Finish Date	Man Days	Cost
New Accounts	12/8/94	11/11/94	222	£13,000
Delivery System	3/3/94	31/11/94	53	£34,000

Total Cost: £55,000

References:

Ashworth, C., Goodland, M., "SSADM, a Practical Approach", 1990

Self Directed Study:

1) If we don't finish the Drug Card, Staff Allocation, and Invoice, in the lecture and tutorial, do them in your own time.

Drug Card Solution

UNF

PATIENT
Patient No
Surname
Forename
Ward No
Ward Name
{Drug Code
Date
Drug Name
Dosage
Time}

1NF

PATIENT
Patient No
Surname
Forename
Ward No
Ward Name

PATIENT_DRUG
Patient No
Drug Code
Date
Drug Name
Dosage
Time

2NF

PATIENT
Patient No
Surname
Forename
Ward No
Ward Name

PATIENT_DRUG
Patient No
Drug Code
Date
Dosage
Time

DRUG
Drug Code
Drug Name

3NF

PATIENT
Patient No
Surname
Forename
Ward No#

WARD
Ward No
Ward Name

PATIENT_DRUG
Patient No
Drug Code
Date
Dosage
Time

DRUG
Drug Code
Drug Name

Staff Allocation Sheet Solution

UNF

PROJECT
Proj Code
Proj Desc
Cust No
Cust Name
{Staff No
Staff Name
Grade
No of Days}

1NF

PROJECT
Proj Code
Proj Desc
Cust No
Cust Name

PROJECT_STAFF
Proj Code
Staff No
Staff Name
Grade
No of Days

2NF

PROJECT
Proj Code
Proj Desc
Cust No
Cust Name

PROJECT_STAFF
Proj Code
Staff No
No of Days

STAFF
Staff No
Staff Name
Grade

3NF

PROJECT
Proj Code
Proj Desc
Cust No#

CUSTOMER
Cust No
Cust Name

PROJECT_STAFF
Proj Code
Staff No
No of Days

STAFF
Staff No
Staff Name
Grade

Invoice Solution

UNF

INVOICE
Invoice No
Date of Invoice
Cust No
Cust Name
Cust Address
{Proj Desc
Start Date
Finish Date
Man Days
Cost}
Total Cost

1NF

INVOICE
Invoice No
Date of Invoice
Total Cost
Cust No
Cust Name
Cust Address

INVOICE_PROJ
Invoice No
Proj Desc
Start Date
Finish Date
Man Days
Cost

2NF

INVOICE
Invoice No
Date of Invoice
Total Cost
Cust No
Cust Name
Cust Address

INVOICE_PROJ
Invoice No
Proj Desc

PROJECT
Proj Desc
Start Date
Finish Date
Man Days
Cost

3NF

INVOICE
Invoice No
Date of Invoice
Total Cost
Cust No #

CUSTOMER
Cust No
CustName
Cust Address

INVOICE_PROJ
Invoice No
Proj Code

PROJECT
Proj Desc
Start Date
Finish Date
Man Days
Cost