Structured System Analysis and Design Method - SSADM

An Introduction to

Relational Data Analysis

to

Third Normal Form

Contents

Introduction

Relational Data Analysis (RDA, also known as normalisation) is a formal technique which examines the existing data in an organisation to produce a simplified, flexible, and efficient model of the data groups. Relational data analysis is carried out for the following reasons.

- To define the detailed organisation of data within the information system without reference to the processing which uses the data.
- To produce a standard representation by reducing the complexity of the data in three stages (called “forms”).
- To ensure that this representation presents the data in a structure that maximises its flexibility and usability while eliminating redundancy and duplication.

As a result of completing data analysis one is able to verify the completeness of the system design prior to building a physical system, viz.:

- The data model which results from data analysis can be compared with that produced during logical data structuring to ensure that all processing and data storage requirements can be met.
- The output from data modelling can be used to build physical records for inclusion in a file or database management system.
- The individual components identified in data analysis can be used for approximate sizing of the system.

Relational Data Analysis

Data analysis provides a means of reducing the complexity of the information held within a system. It causes the logical groups of data to be identified and indicates how these groups relate to each other. A methodical approach to data analysis is followed which, in three steps, will produce a simple group of related data items. An alternative name for this process is ‘normalisation’.

The steps of normalisation are shown below. The steps are described later under “RDA Technique” whilst the terminology used is explained in “Explanation of Terms”.

Unnormalised Form.	The unnormalised form (UNF) is the initial list of identifiable data which is to be analysed.
First Normal Form.	The first normal form (FNF) is produced by removing repeating data groups from within the unnormalised form to create new, simpler data groups.
Second Normal Form.	Second normal form (SNF) considers keys comprising more than one element and ensures that the data elements are dependent on the complete key.
Third Normal Form.	Third normal form (TNF) ensures that data elements are dependant on the key element and not on other non-key elements.
Rationalisation.	During rationalisation all TNF groups are considered together to produce a set of data groups in which no two groups have the same key. Data groups not relevant to the processing requirements are discarded, meaningful names are allocated to the groups, and foreign keys are identified.

There are normal forms beyond TNF (Boyce-Codd Normal Form, Fourth Normal Form, Fifth Normal Form). However, in all but a few rare and pathological cases, data in TNF is already in these higher normal forms. For this reason, analysis to these forms is not normally carried out in SSADM.

Comparing Relational Data Analysis and Logical Data Structuring

Before considering the differences between the relational data structure and the LDM it is necessary to describe how they are constructed. The LDM is developed from the chosen BSO and is the result of a subjective understanding by the analyst of the data in the system. On the other hand, the relational structure diagram is the product of an objective approach, starting from source documentation. However, it may be incomplete, as it may not have been possible to analyse sufficient source documents etc. to identify all the required data groups.

Relational Structure Diagram

The aim of relational data analysis produce a complex but flexible data structure. However, the number of entities and access paths identified normally make the final structure impractical to implement. The relational structure diagram alone may be used as it stands to build the required system because it gives a comprehensive view of the data structure. These can be mapped directly onto a relational database management system. However, access times may be slow because of the inherent flexibility of the normalised structure. It will provide a greater than absolutely necessary number of data groups and access paths between them.

Logical Data Model

The LDM has fewer data groups. It has, therefore, a simpler structure, but it lacks flexibility. The LDM alone may be used as it stands to build the required system. However, this is unlikely to lead to an efficient database. This is because the internal structure of each LDM entity may be complex i.e. may not be in third normal form. This can lead to duplication of data within the database with the resultant increase in storage space and possibility of data incompatibility. The LDM may not be flexible enough to deal with a variety of demands, leading to either a restriction on functionality or a requirement for extensive processing of the data.

RDA and SSADM

RDA is not unique to SSADM and can be used in any environment where efficient data structuring is required. However, the differences between RDA and LDS are such that both can be useful in data analysis. For this reason, SSADM, with its philosophy of comparing complementary views of the system, does both. RDA is used in Stage 3 of SSADM and optionally in Stage 1:

Stage 3. The principal use of RDA in SSADM is step 340 to enhance the required LDM. The inputs and outputs of the required system are analysed and the resulting normalised data groups are used to produce an LDS-style diagram. This structure is compared with that in the required system LDM, produced in Step 320. The differences are resolved to produce the final required system LDM.

Stage 1. RDA may also be used in Stage 1, Step 140 when analysing the current system data, under the following circumstances:

- When producing the LDS it is sometimes difficult to obtain a clear view of the data required to support the processing requirements. In this case it may be useful to

apply RDA to one or two of the major reports and/or data stores, but not in any great detail, to provide a basis from which the remaining analysis is developed.
- Where a high proportion of the current system input/output forms is beyond the control of the project and will remain unchanged in the required system.
- If the current system is wholly or partially a computer system, its file structures may be analysed.

However, RDA is optional in Stage 1 and should only be performed when it is useful.

Inputs to Relational Data Analysis in SSADM

If the technique is used in Stage 1 the sources will be from the current system:
- Current system file structures.
- Report layouts.
- Input and output formats.

In Stage 3 the aim is to produce the most rigorous definition of the data content of all the entities within the final data structure. All relevant sources must be input to the analysis. The major sources are:
- I/O Structures from step 330.
- Current file structures (computer systems only).
- The required system LDM (produced in Step 320), which includes the LDS and Entity Descriptions.

Outputs From Relational Data Analysis in SSADM

The expected output from step 340:
- List of sources used in RDA.
- Rationalised TNF relations.
- Data dictionary.
- Required system LDM.

Summary

By performing data analysis to third normal form an unambiguous set of data groups is created showing clearly all the data elements and their interrelationships. This has the following benefits:
- The analysis produces a data requirement which is independent of any processing.
- Logical and physical requirements are separated.
- A set of rules ensures the analysis is complete.
- It operates on real sources of information.
    - Input/Output documents.
    - Existing files.
    - Screen formats.
    - Report layouts.
- It provides a foundation for defining and structuring the physical system.

Explanation Of Terms

A number of terms are used in the technique. They are explained in this section:

Table Of Data Elements (Data Group)

Relational Data Analysis uses a standard tabular format to represent the information within the system. The analysis considers the data as being grouped into two dimensional tables. This does not mean that all data occurring in the system will be in tabular form; however, it may be expressed in this way to assist the use of the technique (figure 1).

Personal Number	Name	Rank	Service
690193T	Biggles	Sqn Ldr	RAF
C020198W	Hornblower	Lt Cdr	RN
209429K	Tanner	Mr	Civilian
123456X	Smith	Flt LT	RAF
C456789X	Smith	Lt	RN

Figure 1: Table of Data Elements

Data Element

Each of the entries in the table is known as a data element, e.g. in figure 1, Personal No, Name, Rank and Service are data elements. In RDA it is the name of the element which is important. For analysis purposes the elements can be listed as in figure 2 in a format which allows progressive analysis to occur.

Personal Number
Name
Rank
Service

Figure 2: Unnormalised Form — List of Data Elements

Key Data Elements

To be able to insert, delete or update information in the table it must be possible to address a row in the table uniquely. Thus one or more of the data elements, which when grouped together, should provide a unique reference to the row of data. This key can take one of three formats: Simple, Compound, or Hierarchic.

Simple Key.

If a row can be uniquely identified by the data in one data element that elements is known as a simple key. In figure 1 knowing someone’s Personal Number allows us to extract the rest of the data in the row. In a list of data items having a simple key the key data element is underlined. Thus figure 2 would become figure 3.

Personal Number
Name
Rank
Service

Figure 3: Unnormalised Form — Simple Key

Compound Key.

A compound key exists when two circumstances occur:

1. More than one data element is required to uniquely identify a row in the table

and

2. Each of the data elements in the key is also the *simple* key of *another* data group.

This can be demonstrated in figure 4, showing a table of postings for an individual. As an individual may have been allocated to several units over a period of time, both Personal Number and Unit Code are required to uniquely identify an entry. Each of these 2 data items is the Simple Key *of another data grouping* (Personnel and Units).

Personal Number	Unit Code	Date Posted In	Date Posted Out
690193T	1234	1 May 82	2 Jun 86
C020198W	A634	21 Apr 80	6 Aug 84
690193T	3421	25 Dec 78	30 Apr 82

Figure 4: Table of data elements

In a list of the data elements each element which makes up the compound key will be underlined as shown in figure 5.

Personal Number
Unit Ident Code
Date Posted In
Date Posted Out

Figure 5: Unnormalised Form — Compound Key

Hierarchic Key.

An hierarchic key (also sometimes called a *composite key*) is another multi-element identifying feature. In this case the rule is:

> More than one data element is required to uniquely identify a row in the table, but not all the data elements included in the hierarchic key are unique identifiers in their own right.

Figure 6 demonstrates this. Both Unit Code and Sub-Unit names are required to identify a row. The sub unit ‘Admin’ is, however, only unique within Unit Code.

Unit Code	Sub-Unit	Description of Duties
1234	Admin	etc
1234	Engineering	etc
A634	Admin	etc
A634	Operations	etc

Figure 6: Table of Data Elements

In a list of data elements such element which makes up the key will be underlined. Furthermore, the hierarchic elements are enclosed in brackets as in figure 7.

<u>(Unit)</u>
<u>(Sub-Unit</u>)
Description of Duties

Figure 7: Unnormalised Form — Hierarchic Key

Key-Only Data Groups

A key only data group is one in which all the elements form a part of the key.

Foreign Keys

The term foreign key is used to describe a data element which occurs as a non-key element in one data group and *is also a simple key of another data group.*

More than one foreign key may occur in a data group. In figure 8 Unit Ident Code in the Task Data Group is a foreign key because it also exists as the key of the Unit Data Group. Data elements which are identified as foreign keys during RDA are marked with an asterisk '*'.

Task Data Group	**Unit Data Group**
<u>Task ID</u>	<u>Unit Ident Code</u>
Description	Unit function
Man hours	Size
* Unit Ident Code	Location
	Title

Figure 8: Unnormalised Form — Foreign Key

Repeating Group

A repeating group of data fields occurs when a single key identifies multiple values of the same field, e.g.:

Personal Number	Name	Rank	Unit Served At	Date Posted in	Date Posted out
690193T	Biggles	Sqn Ldr	Hendon	1 May 82	2 Jun 86
			Halton	25 Dec 78	30 Apr 82
			Cranwell	13 Jun 77	24 Dec 78
C020198W	Hornblower	Lt Cdr	Sultan	21 Apr 80	6 Aug 84

Figure 9: Unnormalised Table of Posting History — Repeating Groups

In figure 9 the data elements Unit Served At and the Dates are repeating groups within the key Personal Number. On the list of data elements repeating groups are enclosed in a bracket with the letter R as shown in figure 10.

Personal Number
Name
Rank
Unit Served at ⎫
Date Posted In ⎬ R
Date Posted Out ⎭

Figure 10: Unnormalised Form — Repeating Groups

Nesting.

Repeating groups may be nested or non-nested. Figures 9 and 10 represent an un-nested repeating group. Nested repeating groups are shown in figures 11 and 12. Here, within the row identified by Service Number, we have many Units and for each Unit many entries of Dept.

Personal Number	Name	Unit Served at	Dept	Months in Dept	Date Posted In	Date Posted Ou
690193	Biggles	Hendon	Admin	20	1 May 82	2 Jun 86
			Eng	15		
			Supply	14		
		Halton	Admin	40	25 Dec 78	30 Apr 82
		Cranwell	Supply	9	13 Jun 78	24 Dec 78
			Admin	9		
C020198W	etc.					

Figure 11: Table of Data Elements — Nested Repeating Groups

The unnormalised form would now become:

<u>Service No</u>
Name
Unit served at
Dept
Months } R } R
Date Posted In
Date Posted Out

Figure 12: Unnormalised List of Data Elements — Nested Repeated Groups

Hidden Data

Implied Keys.

An implied key is one which, although a key element of a data group, may not actually appear in the source being analysed but will emerge from analysis. An example would be in figure 13 where job description and pre-employment training are not directly dependant on the key Personal Number. They are attributes of the job itself and the missing key would be employment code.

<u>Personal Number</u>
Name
Job Description
Pre employment training

Figure 13: Unnormalised List of Data Items — Implied Keys

Implied Data.

Implied data occurs when the order in which information is entered in a table has some implied sequence e.g. if in Figure 11 above, the entries are stored in rank order, then hidden data of the relative order of each rank must exist, and should be inserted.

Relations

The formal SSADM documentation uses the term ‘relation’ to describe the table of data items. This term has not been used here because of the inevitable confusion between relations and the relationships on an LDS.

For future reference, a *relation* is the list of data items within the data group which can be uniquely identified by a key. A *relationship* is the connection between these data groups. This is shown diagramatically by lines with crows-feet connecting the data groups.

RDA Technique

The process of relational data analysis can be undertaken as a number of steps which progressively simplify the structure of the document, screen format, or file which is being analysed. The stages of data analysis using third normal form are summarised below. The steps are expanded in the following paragraphs.

Unnormalised Form.

The unnormalised form is that in which the raw data is presented as a list for analysis. The stage involves the following tasks:

- Producing a Table of data items.
- Producing the UNF list of data elements.
- Identify repeating groups with brackets.
- Selecting a key.

First Normal Form.

The first normal form exists when new data groups are produced by removing repeating groups.

Second Normal Form.

The second normal form is achieved by removing part key dependencies.

- Consider groups with compound keys.
- Identify data items not dependent on the whole compound key.
- Form new data groups.

Third Normal Form.

The third normal form is achieved by removing data items uniquely identified by a non-key item in a data group.

Rationalisation.

The TNF is rationalised in the following ways:

- Merge groups with a common key.
- Discard irrelevant groups.
- Assign meaningful names to the data groups.
- Review relationships.
- Identify foreign keys.

Unnormalised Form – Create a List of Data Items

The information to be analysed is converted into tabular form. While this stage is not mandatory, it does help to identify repeating groups and, where a test document is being considered, it helps to check that all data elements have been identified. Sample data should be included in each column. In doing this certain points should be observed:

- *Unique Identifier.* Each row must have some unique value which differentiates it from all other rows. This means that no two rows should be the same. One or more columns together will result in a simple, compound, or hierarchic key being identified.
- *No Hidden Order/Sequence.* If there is significance in the order of rows or columns then it means that there is some hidden data in the relation. This may simply take the

form of an entry serial number or a field which indicates the relative seniority. The hidden data should be included in the table.

- *Column Names.* Column names refer to the data fields and therefore each column in the one table must have a unique name. In the table below the first two columns both contain the personal numbers. To be unique the extra words 'Leader' and 'Team Member' must be added.

Leader Personal Number	Team Member Personal Number	Project Code
2234	6142	4
2234	8192	6
3216	6114	10

Figure 14 - Unnormalised Table

Produce UNF List.

Enter the column names in the first column (Data Items) of the RDA form. The order of entering the data is not important.

Identify Repeating Groups

Identify repeating groups by enclosing them in brackets with the letter R.

Select a Key.

Select a key for the unnormalised form by underlining it on the RDA form. The benefit of RDA is that any item could be chosen as a key, and subsequent stages of analysis will force the best key to emerge. When choosing the keys it is worth considering the following points:

- Choose a unique primary key which identifies something tangible within the system environment.
- The key must not repeat within the UNF list. Choose the smallest number of columns which make the entry in the table unique.
- Choose a numeric identifier (such as a personal number) in preference to an alphabetical identifier (e.g. name) where the chance of it repeating is greater.

Data Analysis of Reports.

When performing data analysis on reports the analyst should consider a number of factors.

Headings. Report headings are often ignored. However, they may indicate static data which may well be an important key, e.g. 'Annual Report of Accidents' could indicate a static data item 'Report Type' which means there will be monthly and weekly reports as well.

Matrix of Information. There may well be a matrix of information within a report which indicates nested repeating groups. When identified, it is important to identify the sequence of nesting e.g.:

Accident Type	Person Type	Qty
Fatal Accident	Passenger	10
	Crew	2
Injured	Passenger	1
	Crew	0

This would emerge as:

Category of Accident } R
Passenger Type } R
Qty

Figure 15: UNF Format of Report

Figure 16 (overleaf) shows an example table of data. This example will be used throughout the remainder of this document. The UNF data item list of this table is shown in figure 17.

Personal No	Name	Rank	Current Employ Code	Pre Employ Training	Job Desc	Unit	Mths in Dept	Dept	Dept Strength	Posted In	Posted Out
690193T	Biggles	Sqn Ldr	A3345	Systems Course	S Anal	Hendon	Admin	20	10	1 May 82	2 Jun 86
							Eng	15	5		
							Supply	14	15		
						Halton	Admin	40	3	25 Dec 78	30 Apr 82
						Cranwell	Supply	9	23	13 Jun 78	24 Dec 78
C020198	etc										

Figure 16: Table of Data Elements for Analysis.

UNF	FNF	SNF	TNF
Personal No			
Rank			
Name			
Curr Empl Code			
Pre-Employ Trg			
Job Desc			
Unit ⎫			
Dept ⎫ ⎪			
Months in Dept } R ⎪			
Dept Str ⎭ } R			
Posted In ⎪			
Posted Out ⎭			

Figure 17: RDA Form Entry — Unnormalised Data

First Normal Form - Remove Repeating Groups

The first normal form rules are concerned with removing all repeating groups including those which are nested. For non-nested lists this is achieved by forming new data groups building *compound keys* comprising of the list's primary key and a unique identifying item within the repeating group.

If the list contains nested repeating groups a hierarchy of multi-field keys is built up. Within each nesting level the key of the level above is increased in size by the unique identifier of the next level. Within each level all the data items identified by the key are included in the new data group. Thus figure 17 becomes figure 18 after applying first normal form rules.

UNF	FNF	SNF	TNF
Personal No	Personal No		
Rank	Rank		
Name	Name		
Curr Empl Code	Curr Empl Code		
Pre-Employ Trg	Pre-Employ Trg		
Job Desc	Job Desc		
Unit ⎫			
Dept ⎫ ⎪			
Months in Dept } R ⎪			
Dept Str ⎭ } R	Personal No		
Posted In ⎪	Unit		
Posted Out ⎭	Posted In		
	Posted Out		
	Personal No		
	(Unit)		
	(Dept)		
	Months in Dept		
	Dept Str		

Figure 18: First Normal Form - Repeating Groups Removed

Introduction of Hierarchic Keys.

Sometimes when removing repeating groups the resulting compound key is made up of a large number of data elements.

UNF		**FNF**
Order No		Order No
etc 1		etc 1
Product Code	⎤	
etc 2	⎥	Order No
Delivery Date	⎤ ⎬ R	Product Code
etc 3	⎥ R ⎥	etc 2
Delivery Address	⎤ R ⎥ ⎥	
etc 4	⎦ ⎦ ⎦	Order No
		Product Code
		Delivery Date
		etc 3
		Order No
		Product Code
		Delivery Date
		Delivery Address
		etc 4

Figure 19: First Normalised Form — Order Form

Figure 19 would result from a multi line order form can be simplified if a hierarchic key, Order Line No, were introduced to replace the compound key of Product Code, Delivery Date and Delivery Address. This would result in a FNF grouping of:

(Order No)
(Order Line No)
etc 4

Second Normal Form – Remove Part-Key Dependencies

The second normal form ensures that all items within a data group having compound or hierarchic keys depend on all elements of that key. This is achieved by considering in turn each FNF data group having a compound or hierarchic key. FNF data groups with simple keys are automatically in SNF.

For reach item in the data group ask the question 'Can this data be uniquely identified by part of the compound key or does it require the whole key? '

If part key dependencies are identified, form new data groups. The data items which do not depend upon the whole key are listed in groups having the correct key. These new groups, and what is left of the original data group, are entered in the SNF column of the RDA form.

If there are nested repeating groups, the compound keys formed in generation the FNF data groups may be more complex than is necessary. It may be that only part of a compound key is sufficient to identify a row in a table. If that is the case, the superfluous attributes should be removed from the key, but kept in the data group as foreign keys.

In Figure 18 there are two groups having compound keys: Personal No/Unit and Personal No/(Unit/Dept). The length of time spent in the department, 'Months in Dept', depends on both elements of the key. However, the department strength 'Dept Str' depends only on (Unit/Dept), not on Personal No. Therefore, a new data group is formed as in figure 20.

UNF		FNF	SNF	TNF
Personal No		Personal No		
Rank		Rank		
Name		Name		
Curr Empl Code		Curr Empl Code		
Pre-Employ Trg		Pre-Employ Trg		
Job Desc		Job Desc		
Unit				
Dept				
Months in Dept } R				
Dept Str	} R	Personal No		
Posted In		Unit		
Posted Out		Posted In		
		Posted Out		
		Personal No	Personal No	
		(Unit)	(Unit)	
		(Dept)	(Dept)	
		Months in Dept	Months in Dept	
		Dept Str		
			(Unit)	
			(Dept)	
			Dept Str	

Figure 20: Second Normal Form — Part Key Dependencies Removed

Third Normal Form – Remove Attributes Not Dependent on the Key

Second normal form considered data items and their relationship with the group's key. Third normal form identifies items which depend not on the key but on some other data item.

Consider each SNF data group in turn. For each non-key item in the data group, compare it with each other non-key item and ask the question, 'For a given value of the one can we just have one value of the other?'

Where an item is found to depend on another data item a new data group is formed. One of these related data items will be the key, and it will remain in the original data group. All items dependent on this new key will be included in the new data group.

In figure 21 the 'Personal No' data group has been transferred forward to SNF. At TNF each item is considered and it can be seen that job description and pre-employment training (carried out to prepare for the job) are in fact dependent on current employment code. Therefore:

- A new data group is formed having ‘Current Employment Code’ as its key.
- ‘Pre-employment Training’ and ‘Job Description’ are data elements in the new data group.
- ‘Current Employment Code’ remains in the original ‘Personal No’ data group.

UNF	**FNF**	**SNF**	**TNF**
Personal No	Personal No		Personal No
Rank	Rank		Rank
Name	Name		Name
Curr Empl Code	Curr Empl Code		Curr Empl Code
Pre-Employ Trg	Pre-Employ Trg		
Job Desc	Job Desc		Empl Code
Unit			Pre-Employ Trg
Dept			Job Desc
Months in Dept } R			
Dept Str } R	Personal No		Personal No
Posted In	Unit		Unit
Posted Out	Posted In		Posted In
	Posted Out		Posted Out
	Personal No	Personal No	Personal No
	(Unit)	(Unit)	(Unit)
	(Dept)	(Dept)	(Dept)
	Months in Dept	Months in Dept	Months in Dept
	Dept Str		
		(Unit)	(Unit)
		(Dept)	(Dept)
		Dept Str	Dept Str

Figure 21: Third Normal Form — Inter-Data Item Dependence Removed

Inter-Key Dependency.

In producing the TNF it is important to consider all items for interdependency, including the simple key and each element of a compound key. This allows one to pose the question does item A depend on item B or B on A? This resolves any problems caused by initially selecting the wrong key.

Rationalisation

Once the TFN relations have been obtained, rationalisation needs to take place to ensure that result of RDA is as simple as possible, coherent, and meaningful. Rationalisation is carried out in two stages:

- On the TNF of a simple table of data. The results are recorded in the last column of the Relational Data Analysis Form.
- On the complete system. The results are recorded on the consolidated Rationalised TNF Relations form..

In either case rationalisation includes the following steps:

Merge Groups Having a Common Key.

All the data groups which result from TNF are examined. If groups are identified as having the same key, they are merged. Take care that the true meaning of the key is clear.

Discard Irrelevant Groups.

With a view to the functional requirements of the system, any data group not serving a purpose is discarded. In particular data groups comprising a key but no data element may not be required.

Assign Meaningful Names.

Meaningful names within the context of the overall system are assigned to the merged groups. Furthermore, data item names are rationalised. The following problems must be resolved:

- Data items having the same name but different meanings.
- Data items having different names but the same meaning.

This step ensures that a single, unambiguous meaning can be assigned to each datum.

Review Relationships.

The new data groups resulting from the steps above are reviewed to ensure that interdependencies have not been re-introduced. This may require the new data groups to undergo RDA again

Identify Foreign Keys.

Data items in one data group which are the key of a data group in their own right are marked with an ‘*’.

In the example form being analysed, the only action which can be completed for rationalisation is to assign meaningful names to the data groups and to identify the foreign keys. Empl Code is the simple key of a data group. It also appears as an entry in the group with the key Personal No, albeit with a different name. This name must be changed and marked with an ‘*’, as shown in figure 22.

Personal No Rank Name * Empl Code	PERSON
Empl Code Pre-Emply Trg Job Description	JOB
Personal No Unit Posted In Posted Out	UNITS SERVED
Personal No (Unit Dept) Months	TIME IN DEPARTMENT
(Unit Dept) Strength	DEPARTMENT

Figure 22: Rationalised Data Groups

Producing the Relational Structure Diagram

The relations produced by relational data analysis are used to produce an relational structure diagram, which is similar to the LDS produced in Stage 1. It is a graphical representation of the results of relational data analysis. The relational structure diagram and the logical data structure are very similar ways of modelling the same information. Entities on an LDS correspond to normalised relations and similarly the relationships on an LDS correspond to the relationships between normalised data groups.

Steps in Creating the Relational Structure Diagram

There are five steps to be considered in order to construct the relational structure diagram. The starting point is the collection of rationalised data groups that have been produced from the relational data analysis. An example, demonstrating the rules to be applied when producing a relational structure diagram, is at Appendix C. It is important to remember that the basic shape of the relational structure diagram will be similar to that of the required systems LDS.

The five rules or steps are:

Rule 1. Make the TNF data groups into entities.
Rule 2 In hierarchic keys mark the elements that exist as simple keys.
Rule 3. Ensure all masters of groups with compound keys are present.
Rule 4. Start drawing the model by adding groups with a compound key.
Rule 5. Identify groups with foreign keys and add these groups to the model.

Rule 1 - Make the TNF data groups into entities.

Each rationalised TNF data group is considered to be an entity. A name is selected for each data group and a soft box drawn around each name when it is added to the model.

The entities are connected by lines. These represent the relationships between data items that form the key/foreign key of the connected groups. The connections are made according to the type of key relationship which exists, as described in the next four rules.

Rule 2 - In hierarchic keys mark the elements which exist as simple keys

When the key is hierarchic, each part of the hierarchic key is considered. If any of the hierarchic data key elements exist as a simple key in another entity it is marked (with *) within the hierarchic key (figure 23). The entity with the hierarchic key will become the detail of the entity with the simple key (figure 24). Where more than one element of a hierarchic key has been marked with an * the entity will become the detail of several masters. Remember that not all the elements of a hierarchic key can appear as simple key entities.

Entity 0	**Entity 1**
(*Key A	Key A
Data item 0)	Data item 2
Data item 1	Key C

Figure 23: Rationalised TNF Data Groups after Rule 1.

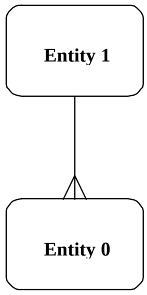

Figure 24: Relational Structure Diagram — Hierarchic Key

Where two or more elements of a hierarchic key exist as the prime key of another entity, these entities can be considered as part of a compound key and subject to Rule 4. See the example in Appendix C.

Rule 3 - All Masters of Compound Keys Must be Present.

Examine all groups with compound keys and ensure that each element of the compound key occurs as the simple or hierarchic key of another entity. If an element is part of a compound key but is not the sole key of another entity then:

a. Create a new data group (entity) with the element as its key. This will more than likely be a key-only entity.
b. Make this new entity a master of each entity that has the element as part of its compound key.
c. In all other data groups where it appears as a non-key element mark the item as a foreign key (with *). See Rule 5.

Entity 1	**Entity 2**	**Entity 3**
Key A	Key B	Key A
Data item 2	Data item 3	Key B
Key C	Data item 4	Key C
		Data item 5

Figure 25: Rationalised Data Groups

Note that this rule does not apply to hierarchic keys.

Inspection of Entity 3 in figure 25 shows that the Key C does not exist as a simple key elsewhere, although it does exist as a data item in Entity 1. Rule 3 requires that a new entity, Entity 4, is formed with Key C as its key. As Key C exists as a data item in Entity 1 it must be marked as a foreign key in Entity 1. The entities now appear as shown in figure 26.

Entity 1	**Entity 2**	**Entity 3**	**Entity 4**
Key A	Key B	Key A	Key C
Data item 2	Data item 3	Key B	
*Key C	Data item 4	Key C	
		Data item 5	

Figure 26 - Insertion of Missing Element of a Compound Key

Rule 4 - Data Groups With a Compound Key Become Details

Start to create the structure by making each data group that has a compound key the detail entity of those data groups that have either a single or multiple element of the compound key as their total key. It is thus possible to allocate multiple elements of a detail's compound key to a single master. Each element of the detail key is allocated once only.

It is often easiest to start with the entity whose compound key has the greatest number of data key elements, and comparing these with other entities which have some of these elements as their compound key (see figure 27). The aim is to find the largest common set of elements, as this will produce the least number of relationships on the entity model. (The rationalised TNF data groups in figure 27 could occur because there were two levels of nesting in the unnormalised form.)

Entity 1	**Entity 2**	**Entity 3**	**Entity 4**	**Entity 5**
<u>Key A</u>	<u>Key B</u>	<u>Key A</u>	<u>Key C</u>	<u>Key A</u>
Data item 2	Data item 3	<u>Key B</u>		<u>Key B</u>
*Key C	Data item 4	<u>Key C</u>		Data item 6
		Data item 5		

Figure 27: Rationalised TNF Data Groups

Entity 3 has 3 keys: Key A, B and C. Each of these keys exists as a simple key in Entities 1, 2 and 4 respectively. Entity 3 becomes the detail of each of these 3 entities. Similarly Entity 5 becomes the detail of Entities 1 and 2. This is represented in figure 28 (the non-ideal relationship).

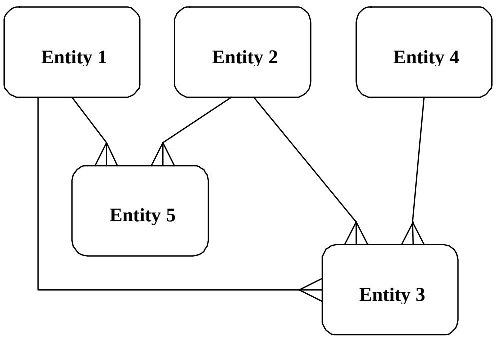

Figure 28: Non-Ideal Relationships

There are too many relationships in figure 28. Some of these can be eliminated by examining the entities with compound keys in order to identify similarities in the keys. The compound key of Entity 5 appears as part of the key of Entity 3. Therefore, the relationship of Entity 3 to Entities 1 and 2 is not direct but indirect through Entity 5. The relational structure diagram can now be drawn as in figure 29, producing the least number of relationships on the entity model. This model is the preferred solution.

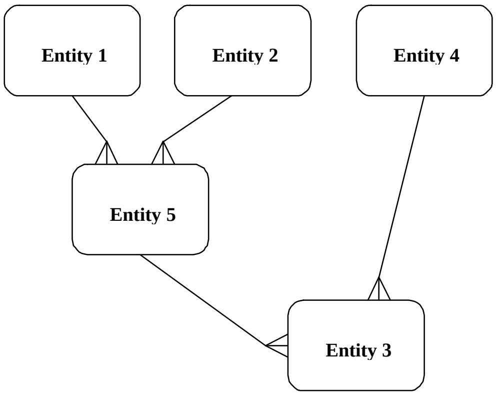

Figure 29: Relational structure diagram - Compound Keys.

Rule 5 - Foreign Keys Indicate Masters

Non-key data items will have been marked as foreign keys during RDA or as the result of applying Rule 3. The entity that contains the foreign key becomes a detail of the entity having the data item (foreign key) as its simple key.

Entity 1	**Entity 4**
Key A	Key C
Data item 1	
* Key C	

Figure 30: Rationalised TNF Data Groups

The relations of figure 30 are represented as shown figure 31.

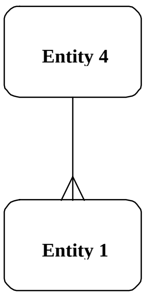

Figure 31: Relational structure diagram - Foreign Key Data Item.

Example

Appendix C contains an example of how to convert rationalised TNF data groups to a relational structure diagram.

Producing the Enhanced Data Model

After relational data analysis there are two views of the required system's data:

a. The logical data structure produced in Step 320.
b. The relational structure diagram.

Either of those views could be used as the basis of the design of the system. However, a more comprehensive and flexible logical design can be obtained if both views are amalgamated. This third view is called the Enhanced Data Model

The Enhanced Data Model

The production of the enhanced data model requires the careful examination of both the normalised data model and the LDM, particularly to identify their advantages and disadvantages. From this examination a workable, comprehensive and flexible logical design is produced, ready for use as input to the logical design process carried out in Stage 5 of SSADM. The examination will frequently identify the need to compromise between the views of the required system's data, as identified separately by the normalised model and the LDM. The compromise view will need to balance the requirement to have a simple structure and content, yet be sufficiently flexible to permit the model to be implemented. Ideally this compromise view, which merges the results of RDA with the LDM, gives a data model having the following characteristics:

- Completeness — it meets all processing requirements.
- Flexibility — it provides a sufficient degree of flexibility.
- Simplicity — it has the simplest possible structure while being both complete and flexible.

The more complex the final enhanced data model, the more difficult it is to convert it to a physical design (though modern RDBMSs ameliorate much of this chore). Furthermore, a very complex structure will increase the data management overheads during data access and, therefore, both the response time and the resource requirements. Hence a satisfactory balance must be obtained between a complex structure which represents the pure TNF design and practical processing restraints which produce an impure but simpler design.

Producing the Enhanced Data Model

The first step is to convert the list of rationalised TNF data groups into a relational structure diagram. The resultant structure should have the same basic form as the required LDS, although further entities may be identified.

The LDS and the results of RDA are merged using the guide lines below:

Multiple Masters

Every LDS entity with multiple masters should appear in the results of RDA and vice versa. If one does not, either:

- There is an error in one of the structures. After resolving the fault the data group concerned is added to the enhanced data model.

or

- There is no error. If the entity does not appear on the LDS then it is likely that the missing normalised data group is embedded in an LDS entity (i.e. the LDS entity is not in third normal form). Consideration of the processing requirements usually identifies which structure is to be included in the enhanced data model. Alternatively, if the data group is missing from the TNF model it should be added to the enhanced data model.

Compatibility

The two structures are compared for compatibility and every data group that appears in both is transferred to the enhanced data model. If any group appears in only one structure, either:

- There is an error in one of the structures. After resolving the fault the data group concerned is added to the enhanced data model.

or

- There is no error. This situation usually arises when a normalised data group does not appear in the LDM because it is not yet required for processing requirements, although it may be in the future. Alternatively, if the data group is missing from the normalised structure it should be added to the enhanced data model.

Access Paths

Any difference in access paths between the two structures is resolved by considering the processing requirements.

Revalidation

The enhanced data model produced is revalidated against the processing requirement.

Supporting Documentation.

The entity descriptions must now be modified to reflect the exact data content of each entity on the enhanced data model. The Data Store/Entity Cross-Reference must be amended to reflect the changes in the data model.

Conclusion

The technique of relational data analysis produces by stages a set of data groups in which repeating groups have been removed, i.e. the data items depend fully on the data group key, the number data groups has been minimised and a data dictionary of meaningful data group and item names has been produced. The resulting relations and relationships can then be shown graphically on the relational structure diagram. This model is then be compared with the required system LDM in order to produce an enhanced logical data model.

Appendix A: Relational Data Analysis Working Paper Form

Purpose

Used for the normalisation of data from its initial unnormalised form to the logical groupings of third form (TNF). One column of the form is used for each stage of the normalisation procedure.

Completion Instructions

Source Name	Enter the reference of the source being analysed.
UNF (Unnormalised Form)	Enter the name of every data item.
Level	Used for identifying repeating groups. Use the number or bracket method.
1NF (First Normal Form)	⎫ Use one column for each step in the
2NF (Second Normal Form)	⎬ normalisation process. Keys are underlined
3NF (Third Normal Form)	⎭
Result	After any rationalisation the result are entered here.
Relation	Enter the meaningful name given to the data group.
Attributes	This column shows all the data items which make up the group.

RDA Working Paper

Project		**Author**		**Date**		**Version**		**Status**		**Page**		**of**	

Source Name	

UNF		**1NF**	**2NF**	**3NF**	**Result**	
Attributes	Level				Relation	Attributes

Appendix B: Entity Description — Part 1

Purpose.

Used for recording information about the entities represented on the entity models. These are the Logical Data Structure (LDS), TNF Entity Model and the Required system LDS. One form is used for each entity. The form is used in the earlier stages of SSADM to show the entity content. It is later used to record the rationalised results of Relational Data Analysis and the contents of the final required system entities.

Completion Instructions

Entity Name and ID	The entity number and name.
Location	Where the entity was found
Occurrences — Average	The average number of occurrences of the entity within the system.
Occurrences - Maximum	The maximum number of occurrences of the entity within the system.
Description	A brief description the entity.
Primary Key	Place a tick alongside the entry if the data item is the primary key (or forms a part of the primary key).
Foreign Key	Enter an tick alongside the entry if the data item is a foreign key.

(If the data item is not part of the key, nor a foreign key, leave these two columns blank.)

Notes	Any information which may be of use in the future, e.g.: Supplementary information Cross references Names of people from whom the information has been obtained etc.

More information on the data structure and its compnents is recorded on the Entity Description form (Part 2), the Relationship Description form, and the Attribute/Data Item Description form.

Entity Description - Part 1

Variant

Project		**Author**		**Date**		**Version**		**Status**		**Page**		**of**	

Entity Name		**Entity ID**	

Location		Occurrences		Average		Max

Description

Synonyms

Attribute Name/ID	**Primary Key**	**Foreign Key**

Rel No	**‘must be’/ ‘may be’**	**‘either’/ ‘or’**	**Link Phrase**	**‘One and only one’/ ‘one or more’**	**Object Entity Name**

Notes

Appendix C: Example Relationship Structure Diagram

This example shows how the relationship structure diagram for the ‘Personnel’ example is produced. At the end of RDA it is sometimes the practice to identify foreign keys and apply names to the data groups. In this example it is assumed this has *not* been done.

Rule 1 - Make the TNF data groups into entities

The data groups have been assigned meaningful names in figure C1.

PERSON	**JOB**	**UNITS SERVED**	**TIME IN DEPARTMENT**	**DEPARTMENT**
Personal No	Empl Code	Personal No	Personal No	(Unit)
Rank	Pre-Emply Trg	Unit	(Unit)	(Dept)
Name	Job Description	Posted In	(Dept)	Strength
Empl Code		Posted Out	Months	

Figure C1: Rationalised TNF Data Groups

Note: As a general rule the number of data elements which constitute a key indicate its level i.e. simple key = level 1, a two element key = level 2 etc. The presence of a foreign key in the list of data items, however, will make an apparent level 1 entity a level 2 entity (master/detail relationship).

Rule 2 - In hierarchic keys mark the elements that exist as simple keys

The key of DEPARTMENT is hierarchic, as ‘Dept’ is not unique. Although a group has not been identified with ‘Unit ID’ as its key, inspection of the original document indicates that one should exist. Mark ‘Unit’ with * and enclose the elements in parentheses (see figure C2).

Note: From now on the hierarchic key (Unit/Dept) can be considered in its entirety wherever else it occurs.

Note: The key of TIME IN DEPARTMENT is also hierarchic.

PERSON	**JOB**	**UNITS SERVED**	**TIME IN DEPARTMENT**	**DEPARTMENT**
Personal No	Empl Code	Personal No	Personal No	(*Unit)
Rank	Pre-Emply Trg	Unit	(*Unit)	(Dept)
Name	Job Description	Posted In	(Dept)	Strength
*Empl Code		Posted Out	Months	

Figure C2: Rule 2 — Hierarchic keys identified.

Rule 3 - All masters of compound keys must be present

There is one entity with a compound key, UNITS SERVED, and two entities with hierarchic keys, TIME IN DEPARTMENT and DEPARTMENT. The elements of the keys which must be considered here are ‘Personal No’ and ‘Unit’. Remember, ‘Dept’ is not unique. The key ‘Personal No’ exists as the simple key of PERSON but there is no group with ‘Unit’ as its key.

A new entity must be introduced with 'Unit' as its key. This will be called UNIT, which, for the moment is a key only group. DEPARTMENT becomes a detail of this entity.

Rule 4 - Add groups with a compound key to the model

1. The UNITS SERVED entity now has a compound key - Personal No/Unit. It therefore becomes a detail of the next level up. It has two masters: PERSON and UNIT.

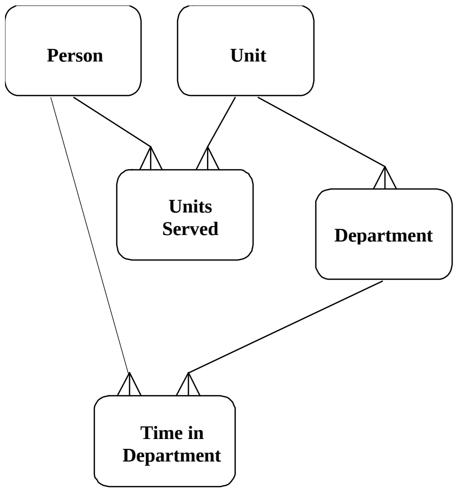

Figure C3: Rule 4 — Entities with compound keys and their masters.

Rule 5 - Identify foreign keys and add the groups to the model

The 'Person' entity contains a foreign key, 'Empl Code', and is, therefore, a detail of 'Job', which has the item as a single key. Add this entity to the model. Also add the relationship which is indicated between 'Unit' and 'Department' (hierarchic key).

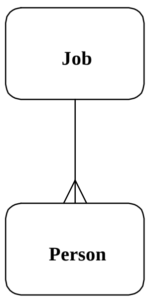

Figure C4: Rule 5 — Foreign keys added.

The final relational structure diagram is therefore:

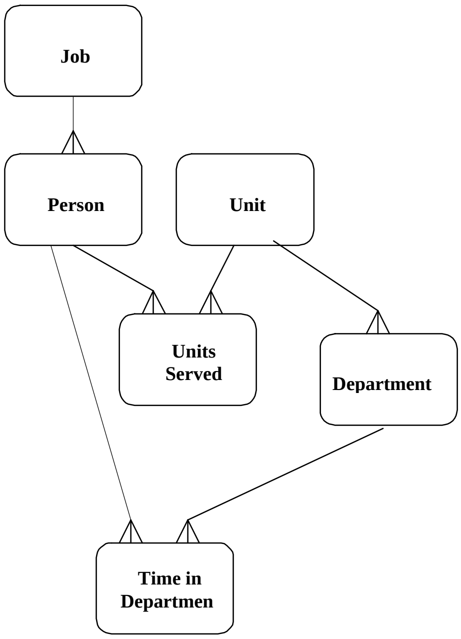

Figure C5: Final relational structure diagram