SBI101-5: Extract a data structure into third normal form using a formalised process.

I will use data from the PROMs Wales project to illustrate how data in 3rd normal form looks like. But first, let's define what database normalisation means.

Database Normalisation

The following definitions and examples have been retrieved from https://www.studytonight.com/dbms (https://www.studytonight.com/dbms).

Database (DB) normalisation is a multi-step systematic approach for organising data inside a DB.

It accomplishes two main purposes:

- It **minimises redundant data**
- It ensures that **data dependancies** are represented in a **logical way**

Main issues caused by a lack of normalisation:

- **Data redundancy** (inefficient use of memory space)
- **Insertion anomaly** (it occurs when certain attribute cannot be inserted into the DB without the presence of other attributes)
- **Updation anomaly** (e.g. partial update)
- **Deletion anomaly** (it occurs when a record is deleted that may contain certain attributes tht shouldn't be deleted)

Normalisation rules

There are **5 rules** of DB normalisation:

- **First Normal Form (1NF)**
- **Second Normal Form (2NF)**
- **Third Normal Form (3NF)**
- **Boyce & Codd Normal Form (BCNF)**
- **Fourth Normal Form (4NF)**

1NF

Table requirements:
1) It should only have **atomic** valued attributes/columns
2) Values stored in a column should be of the **same domain**
3) All the columns in a table should have **unique** names
4) The **order** in which data is stored does **not** matter

roll_no	name	subject
101	Akon	OS, CN
103	Ckon	Java
102	Bkon	C, C++

The example above shows a table **not** in **1NF** (the 'subject' attribute is **not** atomic)

In order to normalise the table into **1NF**, the 'subject' attribute needs to be made atomic:

roll_no	name	subject
101	Akon	OS
101	Akon	CN
103	Ckon	Java
102	Bkon	C
102	Bkon	C++

2NF

Table requirements:
1) It should be in the **1NF**
2) It should **not** have **Partial Dependency** (Partial Dependency means that a non-primary attribute is functionally dependent on a **part** of a candidate key)

score_id	student_id	subject_id	marks	teacher
1	10	1	70	Java Teacher
2	10	2	75	C++ Teacher
3	11	1	80	Java Teacher

In the table above, the primary key is composite ('student_id' + 'subject_id'), however the attribute 'teacher' depends only on one component of the primary key (i.e. 'subject_id' as each teacher is associated to the subject they teach). Thus, in this case there is **Partial Dependency** and the table is **not** in **2NF**.

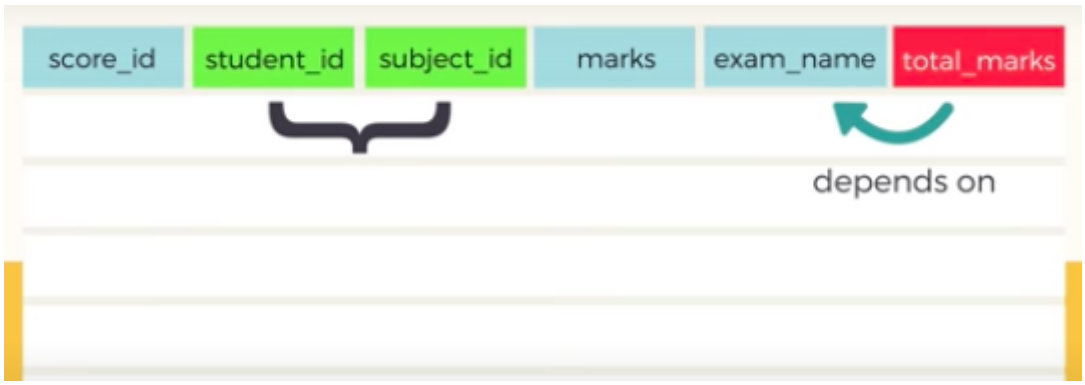
By splitting the data into two tables (as shown below), the data can be **normalised** into **2NF**:

subject_id	subject_name	teacher
1	Java	Java Teacher
2	C++	C++ Teacher
3	Php	Php Teacher

score_id	student_id	subject_id	marks
1	10	1	70
2	10	2	75
3	11	1	80

3NF

Table requirements:
1) It should be in the **2NF**
2) It should **not** have **Transitive Dependency** (Transitive Dependency is an indirect relationship between values in the same table that causes a **functional dependency**)



If in the Score table we add as additional information 'exam_name' and 'total_marks' we would end up with a table **not** normalised into the **3NF**, because the attribute 'total_marks'(in red) depends on the attribute 'exam_name' (e.g. the final total marks might be higher for a theory exam rather than for a practical exam), which in turn depends on the composite primary key "student_id" + "subject_id" (in green). This is an example of **Transitive Dependency**.

In order to **normalise** the table into **3NF**, we need to split the data so to remove the transitive dependency:

Score Table: In 3rd Normal Form

score_id	student_id	subject_id	marks	exam_id

The new Exam table

exam_id	exam_name	total_marks
1	Workshop	200
2	Mains	70
3	Practicals	30

BCNF

Table requirements:
1) It should be in the **3NF**
2) For each **Functional Dependency (X → Y)**, **X** should be a **super key** (this means that X must be a primary attribute if Y is a primary attribute)

student_id	subject	professor
101	Java	P.Java
101	C++	P.Cpp
102	Java	P.Java2
103	C#	P.Chash
104	Java	P.Java

The table above satisfies 3NF, although it is **not** in **BCNF**. The reason is because the composite primary key is "student_id" + "subject", but the primary attribute 'subject' depends on the non-primary attribute professor (i.e. professor → subject). In this table, each professor teaches one and only one subject, while each subject can be taught by one or more professors.

In order to satisfy the **BCNF**, the table needs to be decomposed into two distinct ones:

Student Table

student_id	p_id
101	1
101	2
and so on...	

And, **Professor Table**

p_id	professor	subject
1	P.Java	Java
2	P.Cpp	C++
and so on...		

4NF

Table requirements:
1) It should be in the **BCNF**
2) It should **not** have **Multi-Valued Dependency**

For a table to have multi-valued dependency, all following conditions need to be satisfied:

- For a dependency A → B, if for a single value of A, multiple value of B exists, then the table may have multi-valued dependency
- A table should have at-least 3 columns for it to have a multi-valued dependency
- For a relation R(A,B,C), if there is a multi-valued dependency between, A and B, then B and C should be independent of each other.

s_id	course	hobby
1	Science	Cricket
1	Maths	Hockey
2	C#	Cricket
2	Php	Hockey

The table below shows the courses chosen by each student (s_id) and also their hobbies. However the attribute 'course' is **not related** to the attribute 'hobby', the two attributes are independent of each other. This creates **multi-valued dependency**, making the table **not** normalised into the **4NF**.

In order to satisfy the **4NF**, we need to decompose the table into 2 distinct tables:

CourseOpted Table

s_id	course
1	Science
1	Maths
2	C#
2	Php

And, **Hobbies Table**,

s_id	hobby
1	Cricket
1	Hockey
2	Cricket
2	Hockey

Example using PROMs Wales data

The following input tables were used for the creation of the PROMs Mapping Tool:

- **GeraldFile_200313.csv** --> this file contains the relationships between FTID, DW Views, and PROM Tools (plus some extra information)
- **FormLabels_200313_reduced.csv** --> this file contains extra information on Forms (the column with the Welsh labels of the Forms has been removed to avoid character compatibility issues)
- **FormsInfo_200313_reduced.csv** --> this file has been downloaded directly from the Welsh PROMs Server (View FormsInfo); it contains many attributes, however for this example only the attribute FTID and associated (clinical) pathway have been kept

In [42]:

```
GeraldFile <- read.csv("files/GeraldFile_200313.csv",
                       check.names = FALSE,
                       stringsAsFactors = FALSE)
head(GeraldFile,10) #only the first 10 rows of the table are shown
```

A data.frame: 10 × 7

	FTID	Form Name	nForms	DW_views	FTID_versions	PROM_TO(
	<int>	<chr>	<int>	<chr>	<chr>	<ch
1	1	ORTHOPAEDICHIP	1102	PROMsGeneric	v1.0.0.1 v1.1.0.0 v2.0.0.0	GENER
2	1	ORTHOPAEDICHIP	1102	PROMsOrthopaedicHip_v1001	v1.0.0.1	Oxford H Scc
3	1	ORTHOPAEDICHIP	1102	PROMsOrthopaedicHip_v1100	v1.1.0.	Oxford H Scc
4	1	ORTHOPAEDICHIP	1102	PROMsOrthopaedicHip_v1100	v2.0.0.0	Oxford H Scc
5	2	ORTHOPAEDICKNEE	1247	PROMsGeneric	v1.0.0.1 v1.1.0.0 v2.0.0.0	GENER
6	2	ORTHOPAEDICKNEE	1247	PROMsOrthopaedicKnee_v1001	v1.0.0.1	Oxford Kn Scc
7	2	ORTHOPAEDICKNEE	1247	PROMsOrthopaedicKnee_v1100	v1.1.0.0	Oxford Kn Scc
8	2	ORTHOPAEDICKNEE	1247	PROMsOrthopaedicKnee_v1100	v2.0.0.0	Oxford Kn Scc
9	7	GENERIC	7734	PROMsGeneric	v1.0.0.0 v2.0.0.0	GENER
10	8	FOOTANDANKLE	1	Data not currently being extracted [GENERIC_PATHWAY]		GENER

The table above does **not** meet all the criteria for being in **1NF**: the attribute **'FTID_versions'** is **not atomic**.

In [41]:

```
FormLabels <- read.csv("files/FormLabels_200313_reduced.csv",
                       check.names = FALSE,
                       stringsAsFactors = FALSE)
head(FormLabels,10) #only the first 10 rows of the table are shown
```

A data.frame: 10 × 6

	FTID	Form_Title	Collection method	Tool	Patient-friendly label (English)	Clinician-friendly label
	<int>	<chr>	<chr>	<chr>	<chr>	<chr>
1	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)
2	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)
3	7	GENERIC	In-Clinic	Generic Questionnaire	PROMs Generic Form	Generic Questionnaire
4	8	FOOTANDANKLE	In-Clinic	Pre-op Oxford F&A (in clinic)	Foot and Ankle Form	Foot and Ankle - Oxford Pre op (in-clinic)
5	14	ORTHOPAEDICHIP_POSTOP	In-Clinic	Post-op Oxford Hip (in clinic)	Post Op Orthopaedic Hip Form	Hip - Oxford Post op (in clinic)
6	15	ORTHOPAEDICKNEE_POSTOP	In-Clinic	Post-op Oxford Knee(in clinic)	Post Op Orthopaedic Knee Form	Knee - Oxford Post op (in-clinic)
7	17	GENERIC_POSTOP				
8	19	GENERICQUESTIONS	Remote		PROMs Generic Questions	Generic Questionnaire
9	20	OXFORDHIPSCORE	Remote	Oxford hip score	Oxford Hip Score	Hip - Oxford Form
10	21	OXFORDKNEESCORE	Remote	Oxford knee score	Oxford Knee Score	Knee - Oxford Form

The table above meets the criteria for being in **3NF**:

- It is in **2NF** (and thus, also in 1NF):
  - the **attributes** are **atomic**;
  - **values** stored in the **same column** are of the **same type**;
  - the **columns' names** in the table are **unique**;
  - the **order** in which the data is stored is **irrelevant**
  - There is **no Partial Dependency** (in this example, the key attribute is FTID and all the remaining attributes directly depends on it)

- There is no Transitive Dependency (all the table's attributes depend on the primary key FTID)

In [34]:

```
FormsInfo <- read.csv("files/FormsInfo_200313_reduced.csv",
                      ,check.names = FALSE,
                      ,stringsAsFactors = FALSE)
head(FormsInfo,10) #only the first 10 rows of the table are shown
```

A data.frame: 10 × 2

	FTID	Pathway
	<int>	<chr>
1	200	Non-condition-specific PROMs pathway
2	200	Dermatology pathway - Generic
3	361	Ophthalmology pathway - Cataracts
4	370	Dermatology pathway - Generic
5	200	Dermatology pathway - Generic
6	370	Dermatology pathway - Generic
7	200	Non-condition-specific PROMs pathway
8	200	Non-condition-specific PROMs pathway
9	370	Dermatology pathway - Generic
10	200	Non-condition-specific PROMs pathway

This is raw data from a view of the table 'FormsInfo' stored in the PROMs Welsh database. There are many repetitions (e.g. row 1 and 10 represent the same association of FTID = '200' with Pathway = 'Non-condition-specific PROMs pathway'). Given the **data redundancy** still present at this stage, the data **cannot** be considered **normalised**

The above Input Tables were merged together using the function 'CreateQueryTable_v2.R' to create a unique final query table (qt) from which the data was retrieved for the Shiny App (all the input tables were joined using the attribute 'FTID' as link)

In [4]:

```
source("CreateQueryTable_v2.R")
```

In [38]:

```
qt <- CreatQueryTable("files/GeraldFile_200313.csv",
                      "files/FormsInfo_200313_reduced.csv",
                      "files/FormLabels_200313_reduced.csv")
```

In [40]:

```
head(qt,10) #only the first 10 rows of the dataframe are shown
```

A data.frame: 10 × 9

	FTID	Form_Title	Collection method	Tool	Patient-friendly label (English)	Clinician-friendly label	PROM_TOOL	
	<int>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	
3	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)	About You	
5	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)	EQ5D-5L	
1	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)	Oxford Hip Score	PROM
2	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)	Oxford Hip Score	PROM
4	1	ORTHOPAEDICHIP	In-Clinic	Pre-op Oxford Hip (in clinic)	Orthopaedic Hip Form	Hip - Oxford Pre op (in-clinic)	WPAI	
7	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)	About You	
6	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)	EQ5D-5L	
8	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)	Oxford Knee Score	PROMsC
9	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)	Oxford Knee Score	PROMsC
10	2	ORTHOPAEDICKNEE	In-Clinic	Pre-op Oxford Knee (in clinic)	Orthopaedic Knee Form	Knee - Oxford Pre op (in-clinic)	WPAI	

This final dataframe meets the criteria for being in **1NF**:

- Its **attributes** are **atomic**;

- The **values** stored in the **same column** are of the **same type**;
- The **columns' names** in the table are **unique**;
- The **order** in which the data is stored is **irrelevant**

However, the dataframe is *not* in 2NF as there is **Partial Dependency**: some attributes (i.e. 'Form_Title', 'Collection Method', 'Tool', 'Patient-friendly label (English') and 'Clinician-friendly label') refer to only one part (FTID) of the composite primary key (which is composed by the combination of FTID, PROM_Tool, DW_Views and Pathway).
However, for the purposes of the tool using the final data, it was easier to have all relevant data available in one unique dataframe, rather than having the data scattered in multiple normalised tables.