In Database Systems (DBS) Design, the fundamental design concepts are called

* dependencies and
* normal forms

These two concepts are used in a design process called normalisation. DB design in general, and particularly these concepts, can be difficult to understand for some students.

Relational databases only allow one data structure called a table. **A table has some important properties**. The fixed grid structure

* 1. has a uniform number of columns i.e. each row has a fixed number of columns.
  2. the intersection of a row and a column can only have one data value. i.e. NO REPEATING GROUPS

**Scalable Design: what does this mean?**

Well designed tables are scalable. Logically, a table can have any number of rows. Note, there will be some limit due to the physical memory in the computer.

Bad design suffers from

* Waste of space, (i.e. repeated, duplicated, redundant data) resulting in
* Increase search time and data transfer time from the disk to processing memory,

**Processing anomalies**?

A processing anomaly is an undesirable side effect of an Insert, Update or Delete operation.

**Definition: A key is the minimum number of attributes that distinguishes one row of a table from another**.

**Why minimum number of attributes?**

Because we search tables using access mechanisms such as indexes. Indexes must be as small as possible to be stored and searched efficiently. Think about an index in a text book. It is small, to look up quickly, and then access the correct page of the main data book.

So, we have two types of design. **Database normalisation and Modelling.**

As a (very) general rule of thumb,

**Normalisation is very effective if the system to be computerised exists already i.e. has data already stored in paper forms and the analyst can see redundancy and patterns in the data.**

**Modelling is effective for new build applications.**

There can also be **associations or relationships between different objects**. **Verbs in a specification usually indicate an association between objects**. E.g. Person drives car; Patient attends Doctor; Customer rents Video, Student loans Book. An association can be described by an attribute e.g. qty, date etc.

**Definition: A table is said to be in first normal form (1NF) if it adheres to 3 basic properties of tables**, 1. Fixed columns(grid) structure;2. No repeating groups 3.Key

**To handle different design situations we will add more properties to ensure that our table designs do not suffer any processing anomalies; as we add a new property we will define a higher normal form.**

**So, 1 NF adheres to basic rules, 2 NF extra rule, 3NF another extra rule.**

**Definition: A dependency is where the value that appears in one attribute is determined by another attribute (or is dependent on another). In text books these are called Functional Dependencies (FD). If B is dependent on A, we write**

**A 🡪 B NB: where A can be composite.**

**: FDs corresponds with the objects and the attributes that describe an object. All attributes of an object are functionally dependent on the object identifier (i.e. the key). A FD just describes a pattern of data that is repeated**

This is called **a non loss decomposition** because all the information that was in the original table is now in the new set of tables.

**Definition of non loss**: W**e can always re-combine these separate tables into the original single table without loss of data (rows) or the addition of any previously unknown rows.**

Throughout this course we will see the consequences or knock on effects that result from the use of normalised tables as the basis of data storage. These include:

* data retrieval i.e. how data is retrieved from many tables (see SQL section later), and
* how can we know that the set of tables we form by breaking up a large single table is correct. (we’ll need definitions and rules for correctness, see design 2 next)
* how data is organised on the disk for optimum efficiency

High level description of normalisation

Identify **unnecessary repeated data** (blocks of attributes) and Cut them from the table ( make a new table for them)

Identify the primary key of the new table, and use a copy of that column(s) to make a link,

Effectively this only cuts the non key attributes, and leaves a link on the key attribute(s) of the new table.

**Calculated Fields? Run time Cpu V’s Storage costs**

**NB this is not a strict rule and you as the designer must decide on the key that is relevant and ‘make sense’ from a usability point of view for the given design.**

So, just like a book index, is used to quickly search a large text book, **the database would like to have the primary key as the column that the user uses most frequently to lookup the database.**

**Indexes are used for efficient retrieval of records and for implementing key uniqueness.**

**It is beneficial from a performance perspective to have these index files as small as possible as they may fit into main memory for processing quickly.**

**The designer may opt to introduce a new single attribute key column (identifier) when**

* a large number of attributes make up the primary key,
* the client has no problem with the introduction of new attributes
* the existing key may not be suitable e.g. may not be able to be key in the future as new rows are added over the years.
* Query performance is taken into account i.e. how long it takes to search the database

Normalization is the process of organizing data in a database. This includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependenc

**First Normal Form**

* Eliminate repeating groups in individual tables.
* Create a separate table for each set of related data.
* Identify each set of related data with a primary key.

### Second Normal Form­­­­­

* Create separate tables for sets of values that apply to multiple records.
* Relate these tables with a foreign key.

### Third Normal Form

* Eliminate fields that do not depend on the key.