Normalisation

**Introduction**

The process of reorganising data in a database so that:

* there is no data redundancy (no unnecessary duplications of data)
* The data is logically grouped so that related data is stored together

…is called normalisation.

By carrying out the process of normalisation on a database, we can be assured that the data can be efficiently updated, queried and stored.

There are 3 stages of normalisation:

1st Normal Form

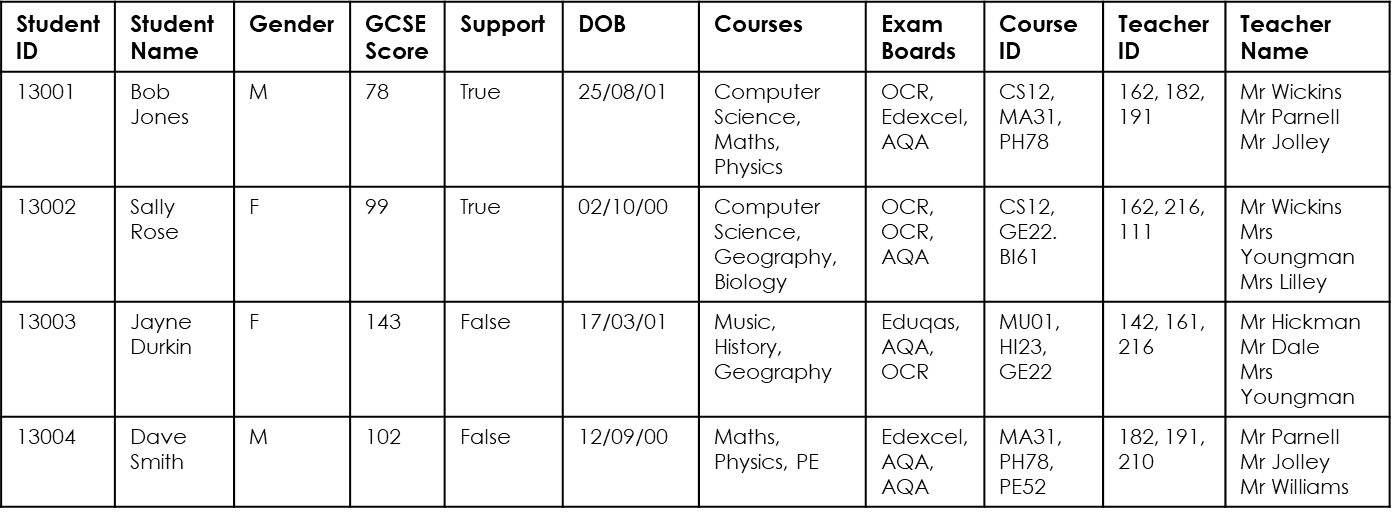
2nd Normal Form

3rd Normal Form

**1st Normal Form**

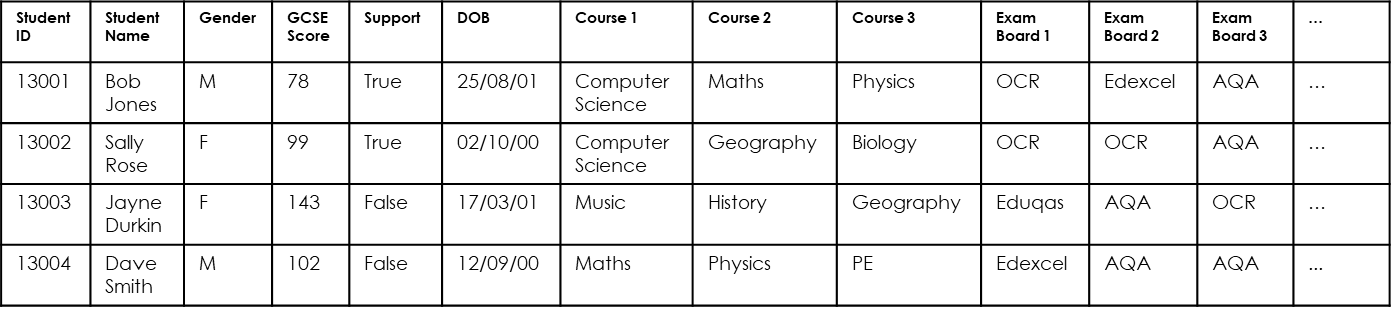
“A database will be in 1nf it is has no repeating attributes or grouped attributes”.

Consider the following flat-file database table which contains data about the courses taken by four A-Level students:



In its current form there are several fields which contain grouped attributes. Grouping attributes like this makes searching for specific data quite difficult. For example, if you wanted to find all students that take Computer Science, you couldn’t run the query SELECT \* FROM students WHERE Courses = ‘Computer Science’, because each entry in the courses field contains several pieces of data.

One alternative would be to reconstruct the table as follows…



So here shows an example of the same database where the grouped attributes have now been moved into separate columns with each field now containing a single piece of data. This means that data in the table is more easily queried (e.g.: you can run the query SELECT \* FROM students WHERE Course 1 = ‘Computer Science’).

However, there is still an issue with the structure of this database. In its current form there are repeated attributes (course 1, course 2, course 3 etc). Tables with repeated attributes like this are also problematic.

Imagine if some students dropped one or more subjects. The database would begin to have several empty (NULL) fields.

Worse still, imagine if some students decided that they would take more than 3 courses. We would have to completely restructure the database, which would be very challenging if already filled with lots of data.

The process of normalisation seeks to solve these issues and we will now look at the first step in normalising this databases so that its design enables the database to be fully efficient.

**“A database will be in ‘1st Normal Form’ it is has no repeating attributes or grouped attributes”.**

So, to get the previous database into 1st normal form we simply need to increase the number of records the table has so that attributes are no longer grouped or repeated.

A close up of a piece of paper

Description automatically generatedThis is demonstrated in the table below which is now in 1st normal form:

This table be easily queried as each field contains a single piece of data and if a student were to either drop a course or enrol in a new course, a record could be deleted/created. There would be no instances of empty fields and no need to restructure the table in the future.

It is still not perfect though as this table contains a lot of redundant data (same data stored in more than one place in a database)…introducing second normal form!

**2nd Normal Form**

“A database will be in 2nf it is already in 1nf and is in a state where all of its non key attributes (columns) are dependent on / related to the primary key”.

So as previously stated, this database (which is in first normal form) is not as efficient as it could be as there is a lot of reductant data. For example, ‘Bob Jones’ has been manually written into this table on three separate occasions which is not best practice because:

1. Humans are prone to making mistakes and one incorrectly spelt entry would cause the data to be unsearchable

A close up of a white background

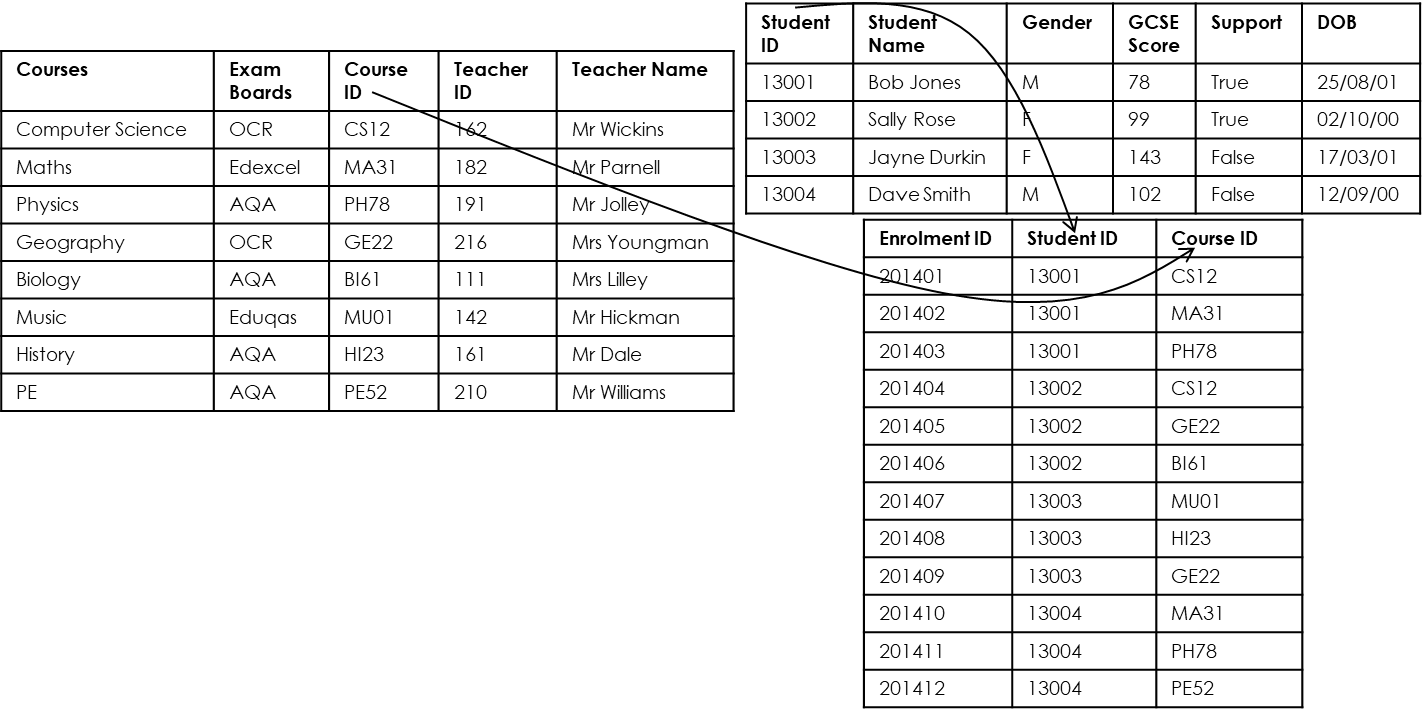
Description automatically generated2. More storage is required to store the duplicate data

To overcome this problem, we can split this table up so that each entity has its own table and then link them together to form a relational database.

As stated earlier, **“a database will be in 2nd Normal Form it is already in 1st Normal Form and is in a state where all of its non key attributes (columns) are dependant on / related to the primary key”**.

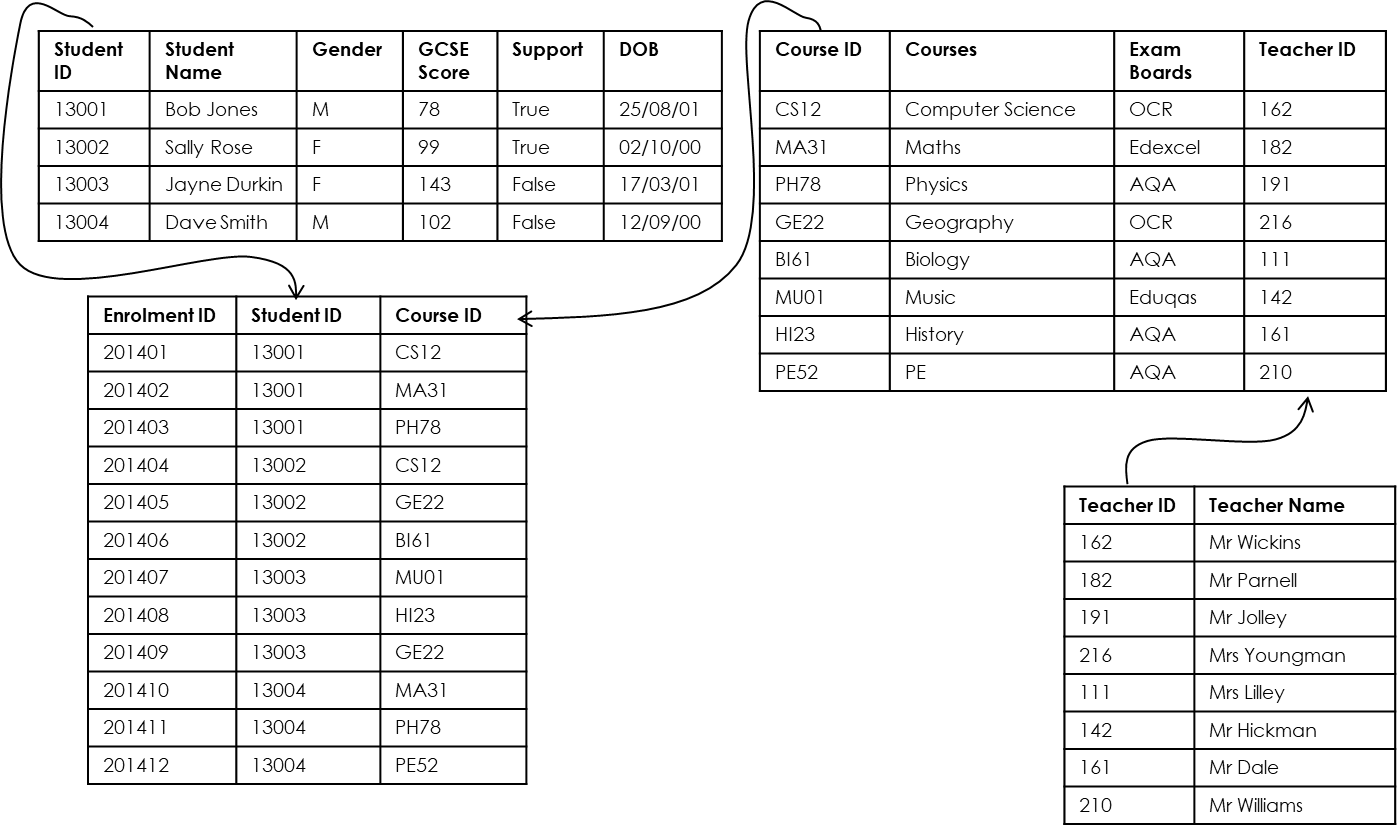
In other words, each table should hold data about an entity, with each column within the table depending on the primary key.

Below is an example of turning the previous flat file data into a relational database by splitting it up into tables about each entity involved:



But looking at each table you can see that the ‘courses’ table is still not in second normal form.

A teacher could in theory teach several courses and so a teacher is not dependant on the course ID. It is however dependant on the Teacher ID (a teacher can’t have more than one ID).

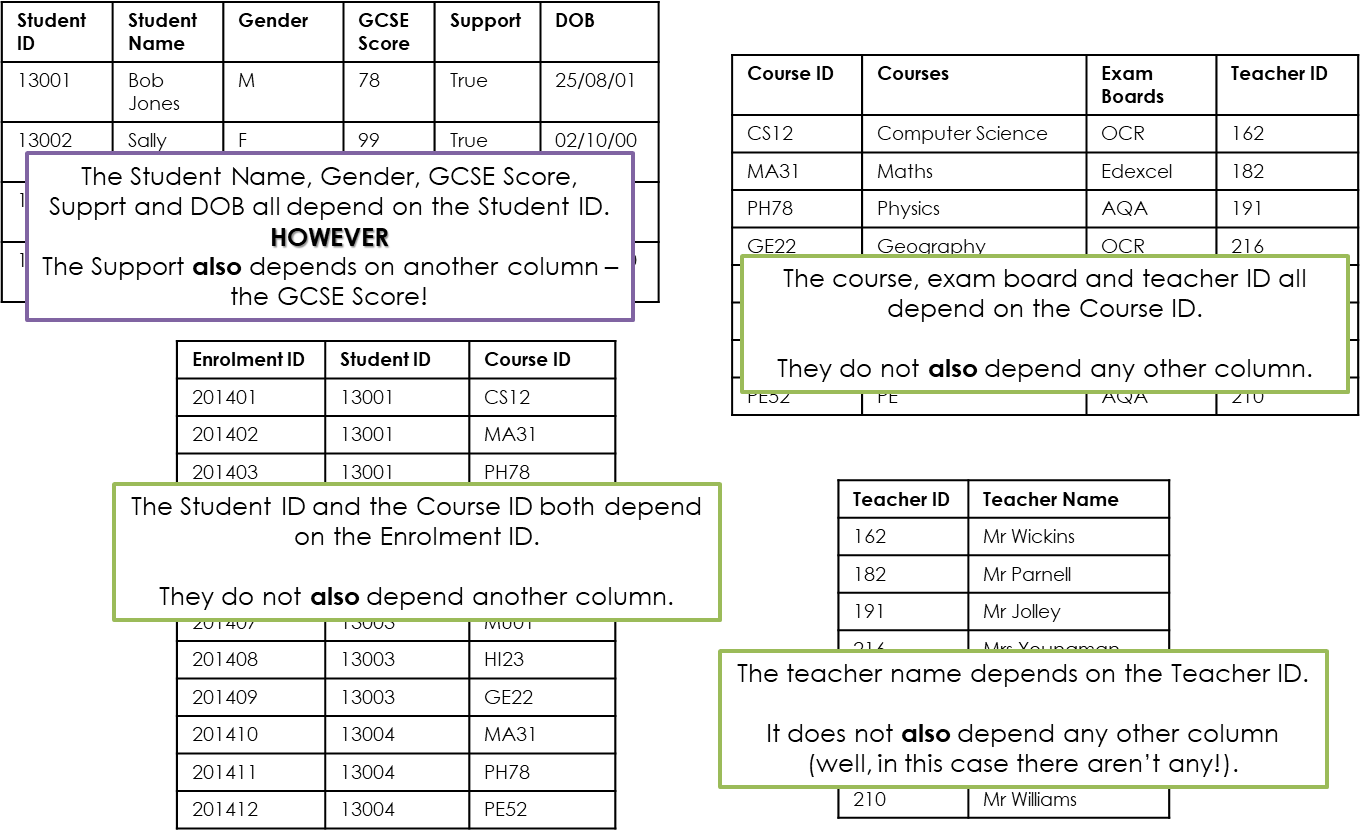
So we can split this table further!

Now this database is in second normal form because it was already in 1st Normal Form and is now in a state where (for each table) all of its non key attributes (columns) are dependant on / related to the primary key.

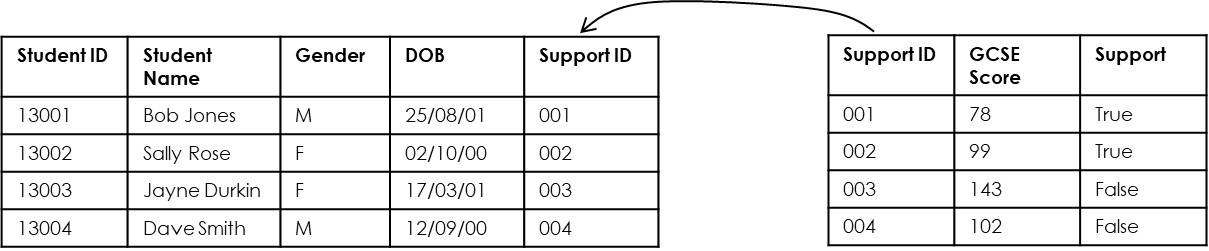
Now, at this stage, quite often a database is also in third normal form…but we must check! So lets take a look at what 3rd normal form is.

**3rd Normal Form**

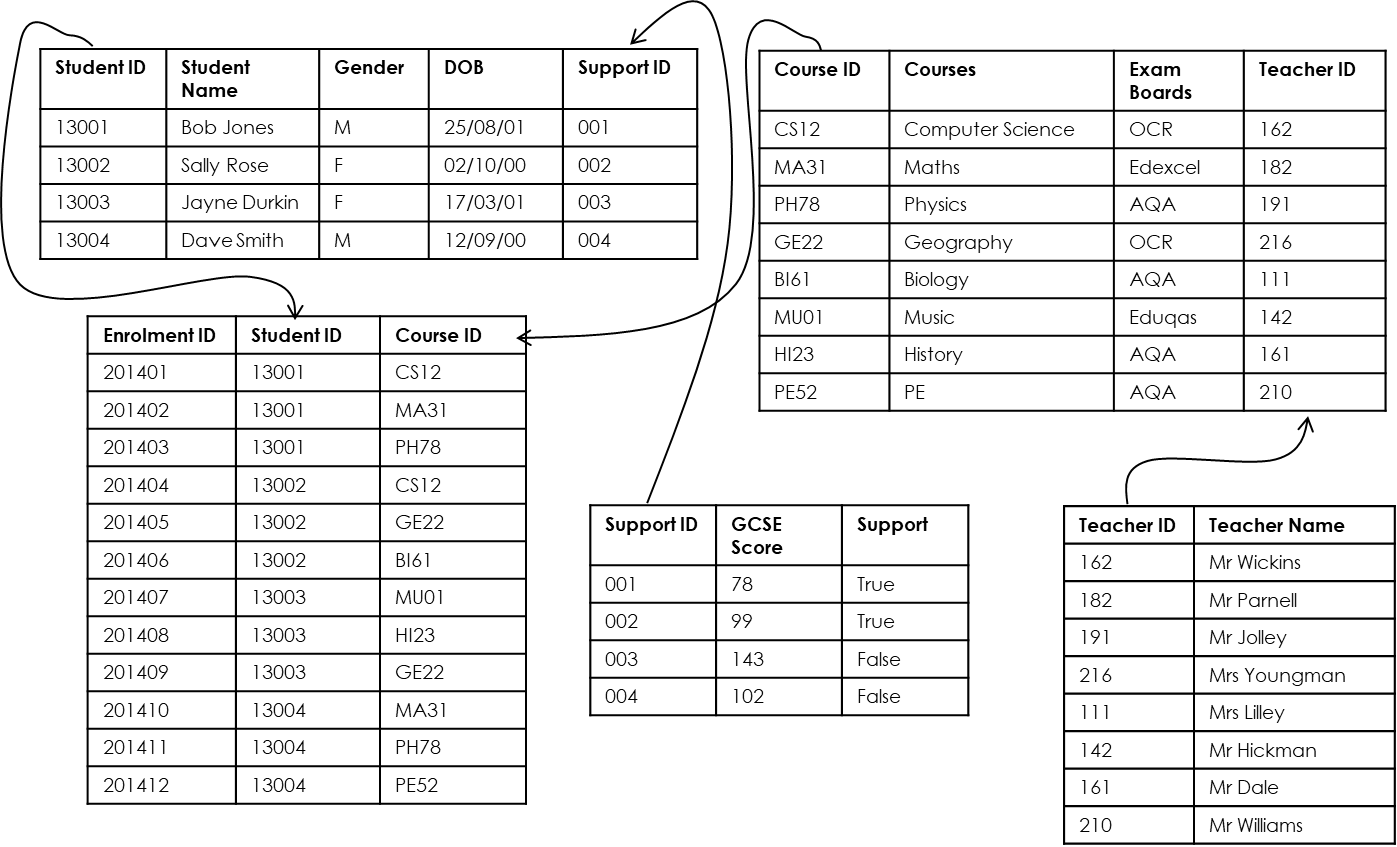
“A database will be in 3nf it is already in 2nf and contains no attributes that (in addition to the primary key) ALSO depend on another attribute in the table”.

“In 3rd NF all of its attributes (columns) are non-transiently dependant on the primary key”.

As stated earlier, “A database will be in 3rd Normal Form it is already in 2nd Normal Form and contains no attributes that (in addition to the primary key) ALSO depend on another attribute in the table”.

Below is an example of getting the previous ‘student table’, which was in second normal form, into third normal form. By splitting the table further, we now can say that each table contains no attributes that (in addition to the primary key) ALSO depend on another attribute in the table as we have removed the Support and GCSE Score fields.

So, our database is now in Third Normal Form…



**Summary**

1NF: “A database will be in 1nf it is has no repeating attributes or grouped attributes”.

2NF: “A database will be in 2nf it is already in 1nf and is in a state where all of its non key attributes (columns) are dependant on / related to the primary key”.

3NF: “A database will be in 3nf it is already in 2nf and contains no attributes that (in addition to the primary key) are ALSO depend on another attribute in the table”.

***Keywords / Key Terms:***

**Relational Database:** *A database which consists of more than one table, linked together with key fields.*

**Normalisation:** *The process of reorganising a database so that there is no data redundancy and so that data is grouped logically so that related data is stored together.*