Normalisation

This is where it gets a mite tricky. ***Skip these two pages if you're taking AQA.***

Normalisation Breaks Down Tables into the Smallest Possible Units

Normalisation is completed during the **design stage** of a relational database.

1) It creates a **logical structure** of related tables and helps to create a flexible, efficient and easy-to-query database.
2) Normalisation ensures data consistency and integrity and gets rid of repeated data (**data redundancy**).
3) Normalisation is a **staged** process consisting of first (**1NF**), second (**2NF**) and third (**3NF**) normal form.

You've got to know the **rules of normalisation** and be able to identify and explain which normal form data is in.

Before a table gets to 1NF, it is in **unnormalised** form — **0NF**.

A table is in **1NF** if:

- every data value in a field is **atomic** — i.e. the data cannot be broken down any further.
- there is a **primary key**.
- there are no **repeating fields** within a table.

A table is in **2NF** if:

- the table follows the rules of **1NF**.
- each table has a **single** primary or composite key.
- there are no **partial key dependencies** — every non-key field must be directly related to the whole primary key (see below for an example).

A table is in **3NF** if:

- the table is in **2NF** (and therefore 1NF)
- there are no **non-key** field **dependencies** — there should be no non-key fields dependent upon any other non-key fields (see below for an example).

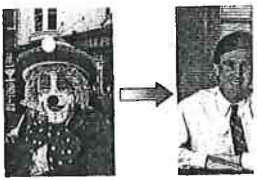
Note — you only get partial key dependencies if the table uses a composite key.

Here's an Example

Each student in a school takes 3 AS level courses. The database is structured like this:

STUDENT (<u>StudentID</u>, Forename, Surname, AS_ID, AS_Title, TeacherID, Teacher_Name, AS_ID, AS_Title, TeacherID, Teacher_Name, AS_ID, AS_Title, TeacherID, Teacher_Name)

This is not in 1 NF form because there are **repeating fields** — AS_ID, AS_Title, Teacher_Name and TeacherID are all repeated 3 times.


Before
Afte

1NF

STUDENT (<u>StudentID</u>, Forename, Surname, <u>*AS_ID*</u>)

COURSE (<u>AS_ID</u>, AS_Title, TeacherID, Teacher_Name)

There are no repeating fields or groups of fields, so it's in 1 NF form r Each student will have 3 records, 1 for each AS they are taking.

The fields StudentID and AS_ID form a composite primary key for the STUDENT table — each record can be uniquely identified from these two fields combined.

But there are partial key dependencies — forename and surname are dependent on the StudentID field but are not uniquely linked to AS_ID. So they're only dependent on part of the primary key (which is StudentID and AS_ID combined, remember).

2NF

STUDENT (<u>StudentID</u>, Forename, Surname)

COURSE (<u>AS_ID</u>, AS_Title, TeacherID, Teacher_Name)

ENROLLED (<u>*StudentID*</u>, <u>*AS_ID*</u>)

The tables are now in 2NF as there are no partial key dependencies. But there is still a non-key dependency as the Teacher_name field is dependent upon TeacherID (which is "non-key") rather than AS_ID.

3NF

STUDENT (<u>StudentID</u>, Forename, Surname)

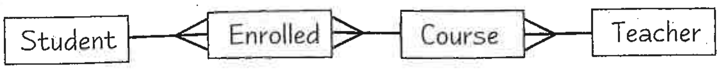
COURSE (<u>AS_ID</u>, AS_Title, *TeacherID*)

ENROLLED (<u>*StudentID*</u>, <u>*AS_ID*</u>)

TEACHER (<u>TeacherID</u>, Teacher_Name)

The tables are now in 3NF and provide a logical basis for the database structure.

The database structure can be shown as:


Student
Enrolled
Course
Teacher