Learning Aims

Relational databases and normalisation

- Describe the use of secondary keys and indexing
- Normalise relations to third normal form
- Understand why databases are normalised

Key Terms

Relational database is a collection of tables in which relationships are modelled by shared attributes.

Indexing is a method used to store the position of each record ordered by a certain attribute. This is used to look up and access data quickly.

Normalisation is a process used to come up with the best possible design for a relational database.

Relational database design

In a relational database, data is held in tables (also called **relations**) and the tables are linked by means of common attributes.

A **relational database** is a collection of tables in which relationships are modelled by shared attributes.

Conceptually then, one row of a table holds one record. Each column in the table represents one attribute.

A table holding data about an entity Book may have the following rows and columns:

BookID	DeweyCode	Title	Author	DatePublished
88	121.9	Mary Berry Cooks the Perfect	Berry, M	2014
123	345.440	The Paying Guests	Waters, S	2014
300	345.440	Fragile Lies	Elliot, L	2015
657	200.00	Learn French with stories	Bibard, F	2014
777	001.602	GCSE ICT	Barber, A	2010
etc				

To describe the table shown above, you would write:

Book (BookID, DeweyCode, Title, Author, DatePublished)

Indexing

In order that a record with a particular primary key can be quickly located in a database, an **index** of primary keys will be automatically maintained by the database software, giving the position of each record according to its primary key.

One or more secondary indexes may be defined when the database is created, for any attribute that is often used as a search criterion.

For example, table both **Author** and **Title** might be defined as **secondary keys**. This would speed up searches on either of these fields, which would otherwise have to be searched sequentially.

Linking database tables

Tables may be linked through the use of a common attribute.

This attribute must be a primary key of one of the tables, and is known as a **foreign key** in the second table.

There are three possible types of relationship between entities:

- one-to-one,
- one-to-many
- many-to-many

Normalisation

Normalisation is a process used to come up with the best possible design for a relational database.

Tables should be organised in such a way that:

- **no data is unnecessarily duplicated** (i.e. the same data item held in more than one table)
- **data is consistent** throughout the database (e.g. a customer is not recorded as having different addresses in different tables of the database). Consistency should be an automatic consequence of not holding any duplicated data. This means that anomalies will not arise when data is inserted, amended or deleted.
- the structure of each table is flexible enough to allow you to enter as many or as few items (for example, components making up a product) as required
- the structure should enable a user to make all kinds of complex queries relating data from
- different tables

There are three basic stages of normalisation known as **first, second and third normal form**.

Three types of normalisation

First normal form

- All **fields** names must be unique.
- Values in fields should be from the same domain.
- Values in **fields** should be atomic.
- No two **records** may be identical.
- Every **table** must have a **primary key**.

Second normal form

- The data must already be in 1NF.
- Remove any partial dependencies.
- Fix any M:M relationships created as a result.

Third normal form

- The data must already be in **2NF**.
- Remove any **transitive dependencies**.

  In other words, ensure that non-key fields are not dependent on each other.

Normalisation – 0NF (flat file before any normalisation)

A database for storing students' details

`STUDENT (name, date of birth, gender, course number, course name, lecturer)`

name	date of birth	gender	course number	course name	lecturer
Tony Gibbons	15/02/1979	M	F451 G403 P202	Computing History of Art Classics	CSA, Craig Sargent AOH, Austin O'Harel CSA, Craig Sargent
Mathew Robinson	14/03/1980	M	G403 Q947 P202	History of Art Textiles Classics	AOH, Austin O'Harel LCO, Linda Cox CSA, Craig Sargent
Claire Matthews	21/05/1974	F	F451 J564 P554	Computing Drama Physics	CSA, Craig Sargent LCO, Linda Cox JHA, James Hayes
Alfred Pillar-Hofman	22/03/1982	M	P202 H544 J390	Classics History English lit	CSA, Craig Sargent SRU, Simon Russel LCO, Linda Cox
James Applegate	01/02/1978	M	Q947 G403 J564	Textiles History of Art Drama	LCO, Linda Cox AOH, Austin O'Harel LCO, Linda Cox
...	...	...	...	...	...

To get to **first normal form (1NF),** a table should follow five rules:

1. All **field** names must be unique.
2. Values in **fields** should be from the same **domain**.
3. Values in **fields** should be atomic.
4. No two **records** can be identical.
5. Each **table** needs a **primary key**.

Normalisation – 1NF

A database for storing students' details

STUDENT (name, date of birth, gender, course number, course name, lecturer initials, lecturer name)

name	date of birth	gender	course number	course name	lecturer initials	lecturer name
Tony Gibbons	15/02/1979	M	F451	Computing	CSA	Craig Sargent
Tony Gibbons	15/02/1979	M	G403	History of Art	AOH	Austin O'Harel
Tony Gibbons	15/02/1979	M	P202	Classics	CSA	Craig Sargent
Mathew Robinson	14/03/1980	M	G403	History of Art	AOH	Austin O'Harel
Mathew Robinson	14/03/1980	M	Q947	Textiles	LCO	Linda Cox
Mathew Robinson	14/03/1980	M	P202	Classics	CSA	Craig Sargent
Claire Matthews	21/05/1974	F	F451	Computing	CSA	Craig Sargent
Claire Matthews	21/05/1974	F	J564	Drama	LCO	Linda Cox
Claire Matthews	21/05/1974	F	P554	Physics	JHA	James Hayes
Alfred Pillar-Hofman	22/03/1982	M	P202	Classics	CSA	Craig Sargent
Alfred Pillar-Hofman	22/03/1982	M	H544	History	SRU	Simon Russel
Alfred Pillar-Hofman	22/03/1982	M	J390	English lit	LCO	Linda Cox
James Applegate	01/02/1978	M	Q947	Textiles	LCO	Linda Cox
James Applegate	01/02/1978	M	G403	History of Art	AOH	Austin O'Harel
James Applegate	01/02/1978	M	J564	Drama	LCO	Linda Cox

To get to **first normal form (1NF),** a table should follow five rules:

1. All **field** names must be unique.
2. Values in **fields** should be from the same **domain**.
3. Values in **fields** should be atomic.
4. No two **records** can be identical.
5. Each **table** needs a **primary key**.

The solution still needs work, but let's go with it for now and see if we can find a better one as we further **normalise** the **database**.

The final rule has been met, so our table is now in **first normal form (1NF)**.

Normalisation – 2NF

A database for storing students' details

STUDENT (<u>student number</u>, name, date of birth, gender)
COURSE (<u>course number</u>, course name, lecturer initials, lecturer name)
STUDENT_TAKES_COURSE(<u>student number</u>, <u>course number</u>)

student number	name	date of birth	gender
001	Tony Gibbons	15/02/1979	M
002	Mathew Robinson	14/03/1980	M
003	Claire Matthews	21/05/1974	F
004	Alfred Pillar-Hofman	22/03/1982	M
005	James Applegate	01/02/1978	M
...	...	...	...

student number	course number
001	F451
001	G403
001	P202
002	G403
002	Q947
002	P202
003	F451
...	...

course number	course name	lecturer initials	lecturer name
F451	Computing	CSA	Craig Sargent
G403	History of Art	AOH	Austin O'Harel
P202	Classics	CSA	Craig Sargent
Q947	Textiles	LCO	Linda Cox
J564	Drama	LCO	Linda Cox
P554	Physics	JHA	James Hayes
H544	History	SRU	Simon Russel
J390	English lit	LCO	Linda Cox

Which courses does Mathew Robinson take?

To get to **second normal form (2NF)**, a table should follow two rules:

1. The data is already in **1NF**.
2. Any **partial dependencies** have been removed.

Once we spot a **many-to-many** relationship, we can fix it by:

- Creating a linking table.
- Assigning the **primary keys** from the two initial **tables** as the **composite key** for the new linking **table**.
- Flipping the **M:M** crows-feet relationship to become two separate **1:M** relationships joined by the new **table.**

3NF

Normalisation – 3NF

A database for storing students' details

STUDENT (student number, name, date of birth, gender)
COURSE (course number, course name, lecturer initials, lecturer name)
STUDENT_TAKES_COURSE(student number, course number)
LECTURER (lecturer initials, lecturer name)

All our tables are now in **3NF**.

In moving from **1NF** to **3NF**, we have gone from one single table (a flat file) to a four-table **relational database**.

We have removed repeating data, and each table now serves a single purpose.

student number	name	date of birth	gender
001	Tony Gibbons	15/02/1979	M
002	Mathew Robinson	14/03/1980	M
003	Claire Matthews	21/05/1974	F
004	Alfred Pillar-Hofman	22/03/1982	M
005	James Applegate	01/02/1978	M
...	...	...	...

student number	course number
001	F451
001	G403
001	P202
002	G403
002	Q947
002	P202
003	F451
...	...

course number	course name	lecturer initials
F451	Computing	CSA
G403	History of Art	AOH
P202	Classics	CSA
Q947	Textiles	LCO
J564	Drama	LCO
P554	Physics	JHA
H544	History	SRU
J390	English lit	LCO

lecturer initials	lecturer name
CSA	Craig Sargent
AOH	Austin O'Harel
LCO	Linda Cox
JHA	James Hayes
SRU	Simon Russel