



4.10.1 Relational Database

Databases

A **database** is an organised collection of data.

A **flat file** database is a database where all the data is stored in one table.

A **relational database** stores data in multiple tables with relationships between them

The data in a database is stored in **tables**.

Each entry/row in the table is a **record**.

Each column in the table is a **field/attribute**.

customerID	firstName	lastName	phone	country
1	John	Smith	020-4034-4930	USA
2	Harriet	Jones	523-4231-9970	UK
3	Billy	Baggins	060-2095-4937	FR
4	Fred	Tomlin	159-6344-4932	UK
5	Rebecca	Taylor	628-1135-6133	SP
6	David	Roberts	941-6274-7330	USA
7	Rohey	Doe	824-3334-1932	USA





Flat File Database

A **flat file** database is a database where all the data is stored in one table.

They are very simple to implement but:

- Can have a lot of repeating data.
- Are slower to query as the database grows.
- Are difficult to maintain as the database becomes larger.

Student ID	First name	Surname	Teacher	Subject
1	John	Smith	Mr. Bear	Mathematics
2	Harriet	Jones	Mr. Bear	Mathematics
3	Billy	Baggins	Ms. Peacock	English
4	Fred	Tomlin	Mr. Bear	Mathematics
5	Rebecca	Taylor	Mr. Bear	Mathematics
6	David	Roberts	Ms. Peacock	English
7	Rohey	Doe	Ms. Peacock	English

The teachers and their subjects are repeated multiple times in the table.





Relational Database

A **relational database** is a database which is created using multiple tables with relationships between those tables.

An **entity** is an object or concept about which data is stored. In a relational database, an entity is usually represented by a table.

For example, a database for a site that stores details of customers and downloaded movies may have 3 tables. One for the customers, one for the movies and the other for storing details about the downloads.

There are 3 entities: customer, movie and download

customer			movie		download		
customerID	firstName	lastName	movieID	movie	customerID	movieID	date
1	John	Smith	1	Alien	1	2	02/06/22
2	Harriet	Jones	2	The Hobbit	1	3	08/06/22
3	Billy	Baggins	3	Troy	2	1	14/07/22
4	Fred	Tomlin	4	Avatar	4	6	05/08/22
5	Rebecca	Taylor	5	Terminator	5	6	11/09/22
6	David	Roberts	6	Shrek	7	3	12/10/22
7	Rohey	Doe	7	Thor	7	7	25/10/22

Benefits of relational databases

- Reduced data redundancy.
- Improved data consistency.
- Fields can be more easily added or removed from tables.
- Security can be improved by controlling access to different table.
- Easier to query.





Primary and Composite Keys

A **primary key** is a field that can be used to uniquely identify a record in a table.

A **composite key** is a primary key that is made up of two or more fields.

The *customerID* in the customer table is a primary key as it is a unique identifier for a customer.

movieID is a primary key for the movie table

The download table has a composite key. Its primary key is the combination of the *customerID* and *movieID* fields.

customer			movie		download		
customerID	firstName	lastName	movieID	movie	customerID	movieID	date
1	John	Smith	1	Alien	1	2	02/06/22
2	Harriet	Jones	2	The Hobbit	1	3	08/06/22
3	Billy	Baggins	3	Troy	2	1	14/07/22
4	Fred	Tomlin	4	Avatar	4	6	05/08/22
5	Rebecca	Taylor	5	Terminator	5	6	11/09/22
6	David	Roberts	6	Shrek	7	3	12/10/22
7	Rohey	Doe	7	Thor	7	7	25/10/22

Foreign Keys

A **foreign key** is a field in one table that links to a primary key in another table.

The foreign key – primary key pairings form the relational links between the entities.

The *customerID* field in the download table is a foreign key that links to the primary key in the customer table.

The *movieID* field in the download table is a foreign key that links to the primary key in the movie table.

customer			download			movie	
customerID	firstName	lastName	customerID	movieID	date	movieID	movie
1	John	Smith	1	2	02/06/22	1	Alien
2	Harriet	Jones	1	3	08/06/22	2	The Hobbit
3	Billy	Baggins	2	1	14/07/22	3	Troy
4	Fred	Tomlin	4	6	05/08/22	4	Avatar
5	Rebecca	Taylor	5	6	11/09/22	5	Terminator
6	David	Roberts	7	3	12/10/22	6	Shrek
7	Rohey	Doe	7	7	25/10/22	7	Thor





Entity-relationship Diagram

An **entity-relationship diagram** is a visual description of relationships between entities.

There are three types of relationship that can exist between entities:

- 1 **One-to-one** – For example, an employee may have one office and that office is only used by that employee.



- 2 **One-to-many** – For example, customer may make many downloads, but each download will be by one customer.



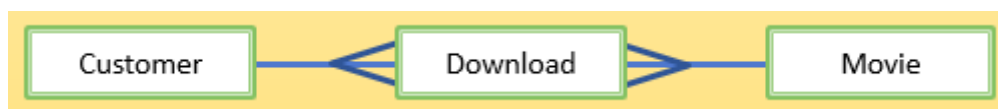
- 3 **Many-to-many** – For example, a customer may purchase many movies, and a movie may have been purchased by many customers.





You can work out the entity relationship diagram for a database by finding the foreign keys.

For a one-to-many relationship, the table containing the foreign key is the 'many' side, while the table containing the referenced primary key is the 'one' side.



customer			download			movie	
customerID	firstName	lastName	customerID	movieID	date	movieID	movie
1	John	Smith	1	2	02/06/22	1	Alien
2	Harriet	Jones	1	3	08/06/22	2	The Hobbit
3	Billy	Baggins	2	1	14/07/22	3	Troy
4	Fred	Tomlin	4	6	05/08/22	4	Avatar
5	Rebecca	Taylor	5	6	11/09/22	5	Terminator
6	David	Roberts	7	3	12/10/22	6	Shrek
7	Rohey	Doe	7	7	25/10/22	7	Thor

Entity Descriptions

Details of a database can be written in simple notation. The table name is given with the attributes after in brackets. The primary key, or fields making up a composite key, are underlined.

customer (customerID, firstName, lastName)

movie (movieID, movie)

download (customerID, movieID, date)

customer			movie		download		
customerID	firstName	lastName	movieID	movie	customerID	movieID	date
1	John	Smith	1	Alien	1	2	02/06/22
2	Harriet	Jones	2	The Hobbit	1	3	08/06/22
3	Billy	Baggins	3	Troy	2	1	14/07/22
4	Fred	Tomlin	4	Avatar	4	6	05/08/22
5	Rebecca	Taylor	5	Terminator	5	6	11/09/22
6	David	Roberts	6	Shrek	7	3	12/10/22
7	Rohey	Doe	7	Thor	7	7	25/10/22

