Databases

A **database** is a collection of **data** stored in an organised and logical way. Data are stored in **tables** and tables are made up of **records** (rows) which can have 1 more **attributes** (columns). An example of a table is given here:

Student ID	First Name	Surname	DateOfBirth	FormTutor
712	Bart	Simpson	1/4/10	Principal Skinner
423	Lisa	Simpson	20/5/12	Mrs Krabapple
917	Ralph	Wiggum	16/6/10	Mrs Krabapple
124	Nelson	Muntz	14/9/09	Principal Skinner

Entity

Each table contains information about an **entity**. A database entity is an object, person, item or thing about which you want the data stored. Examples of database entities are:

Person entity	Object entity	Item entity
✓ Customer	✓ Book	✓ Sale transaction
✓ Employee	✓ Car	✓ Appointment
✓ Student	✓ House	
✓ Teacher		

Data

Data are atomised facts, values and observations that are stored in a database. That is they cannot be broken up further. Data can be stored as any data type.

Field	Student ID	First Name	Height	Date of Birth	Had Flu Vaccination?
Date Type	*Integer/ number*	*Text/ string*	*Real/ float*	*date*	*Boolean – Yes/no or true/false*
Record 1	712	Bart	1.35	1/4/2010	True
Record 1	423	Lisa	1.16	20/5/2012	True
Record 1	917	Ralph	1.05	16/6/2010	False

Database Index

A database index allows for quick speed of retrieval of data from searches of tables. The index is a separate file that has a sorted column of values that link to records in a table.

Record

A record is a single row in a table that can have data stored as 1 or more fields (columns). A record needs to be uniquely identifiable and needs an entity identifier which in this example is Student ID. A table contains multiple records. The following example contains 4 records.

	StudentID	FirstName	Surname	DateOfBirth	FormTutor
Records	712	Bart	Simpson	1/4/10	Principal Skinner
	423	Lisa	Simpson	20/5/12	Mrs Krabapple
	917	Ralph	Wiggum	16/6/10	Mrs Krabapple
	124	Nelson	Muntz	14/9/09	Principal Skinner

The **Student ID** field contains unique values for each record; this means that each value is different. The **Surname** field does not contain unique values. For instance, *Simpson* appears twice.

Field

Fields / attributes form the columns of the database table and refer to the characteristics of a record. For instance, the fields of the table below include:

- ✓ Student ID
- ✓ First name
- ✓ Surname
- ✓ Date of Birth
- ✓ Form tutor

Fields

Student ID	First Name	Surname	Date of Birth	Form Tutor
712	Bart	Simpson	1/4/10	Principal Skinner
423	Lisa	Simpson	20/5/12	Mrs Krabapple
917	Ralph	Wiggum	16/6/10	Mrs Krabapple
124	Nelson	Muntz	14/9/09	Principal Skinner

Data Redundancy

Data redundancy occurs when the same data are stored in multiple places and so we have repeating data. As a result more space is needed to store the same values several times which is not efficient. In the table below notice how the Author Name fields are repeated.

BookID	Title	FirstName	Surname
1	Fantastic Beasts and Where to Find Them	J.K.	Rowling
2	Harry Potter and the Chamber of Secrets	J.K.	Rowling
3	Harry Potter and Order of the Phoenix	J.K.	Rowling
4	The BFG	Roald	Dahl
5	Going Solo	Roald	Dahl
6	Danny Champion of the World	Roald	Dahl
7	War Horse	Michael	Morpurgo
8	Private Peaceful	Michael	Morpurgo

Data Inconsistency

Data inconsistency occurs when data pertaining to the same object are in fact stored in a different format. For instance, JK. Rowling and Joanne Rowling refer to the same person, but the database may record these as two separate authors.

BookID	Title	FirstName	Surname
1	Fantastic Beasts and Where to Find Them	JK	Rowling
2	Harry Potter and the Chamber of Secrets	Joanne	Rowling
3	Harry Potter and Order of the Phoenix	Joanne	Rowling
4	The BFG	Roald	Dahl
5	Going Solo	Roald	Dahl
6	Danny Champion of the World	Roald	Dahl
7	War Horse	Michael	Morpurgo
8	Private Peaceful	Michael	Morpurgo

Relational Databases

Complex databases can be made up of multiple tables linked together by shared values called a key. These relational databases make it easier to search and find information that you want. Relational databases reduce the amount of duplication (redundancy) of data and reduces inconsistencies in the data.

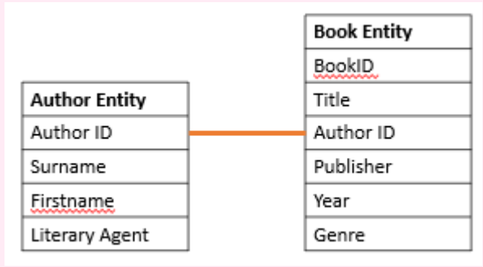
Primary Key

All tables have a field that is the primary key and uniquely identifies each record. This is also known as entity identifier

Foreign Key

These are primary keys that are held as fields in other tables to cross reference tables. They allow tables to be linked together.

For instance, in a book database with two tables including Author table and Book table, AuthorID is primary key in Author table and is used to cross-reference with the AuthorID in the book table which is the foreign key so the two tables can be linked.



Primary key — Author Table

AuthorID	FirstName	Surname	LiteraryAgent
1	Joanne	Rowling	Neil Blair
2	Roald	Dahl	David Higham Associates
3	Michael	Morpurgo	David Higham Associates

Foreign key — Book Table

BookID	AuthorID	Title	YearPublished	Publisher	Genre
1	1	Fantastic ...	2001	Bloomsbury	Fantasy
2	1	... Chamber of Secrets	1998	Bloomsbury	Fantasy
3	1	... Order of the Phoenix	203	Bloomsbury	Fantasy
4	2	The BFG	1982	Penguin	Fantasy
5	2	Going Solo	1986	Jonathan Cape	Autobiography
6	2	Danny Champion ...	1975	Jonathan Cape	Children
7	3	War Horse	1982	Kaye & Ward	Historical fiction
8	3	Private Peaceful	2003	HarperCollins	Historical fiction

Structured Query Language

We will use this book table in the examples that follow.

Book ID	Title	Author	Year Published	Publisher	Genre
1	Fantastic Beasts and Where to Find Them	JK Rowling	2001	Bloomsbury	Fantasy
2	Harry Potter and the Chamber of Secrets	JK Rowling	1998	Bloomsbury	Fantasy
3	Harry Potter and Order of the Phoenix	JK Rowling	2003	Bloomsbury	Fantasy
4	The BFG	Roald Dahl	1982	Penguin	Fantasy
5	Going Solo	Roald Dahl	1986	Jonathan Cape	Autobiography
6	Danny Champion of the World	Roald Dahl	1975	Jonathan Cape	Children
7	War Horse	Michael Morpurgo	1982	Kaye & Ward	Historical fiction
8	Private Peaceful	Michael Morpurgo	2003	HarperCollins	Historical fiction

Select

To retrieve data from the table

To retrieve all records data from the table we can use the SELECT statement with the wild card operator *.

```
SELECT *
FROM tableName
```

Examlpe

```
SELECT *
FROM book
```

RETRIEVED DATA

1	Fantastic Beasts ..	JK Rowling	2001	Bloomsbury	Fantasy
2	..Chamber of Secrets	JK Rowling	1998	Bloomsbury	Fantasy
3	.. Order of the Phoenix	JK Rowling	2003	Bloomsbury	Fantasy
4	The BFG	Roald Dahl	1982	Penguin	Fantasy
5	Going Solo	Roald Dahl	1986	Jonathan Cape	Autobiography
6	Danny Champion ..	Roald Dahl	1975	Jonathan Cape	Children
7	War Horse	Michael Morpurgo	1982	Kaye & Ward	Historical fiction
8	Private Peaceful	Michael Morpurgo	2003	HarperCollins	Historical fiction

We can also choose the fields that we wish to retrieve:

```
SELECT field1, field2, …
FROM tableName
```

EXAMPLE

```
SELECT Author, Title
FROM book
```

RETRIEVED DATA

Fantastic Beasts and Where to Find Them	JK Rowling
Harry Potter and the Chamber of Secrets	JK Rowling
Harry Potter and Order of the Phoenix	JK Rowling
The BFG	Roald Dahl
Going Solo	Roald Dahl
Danny Champion of the World	Roald Dahl
War Horse	Michael Morpurgo
Private Peaceful	Michael Morpurgo

We can sort the output of our SELECT statement by using the ORDER BY clause. ASC and DESC refer to sorting ascending and descending alphabetically or numerically of a specified field.

```
ORDER BY fieldname ASC|DESC
```

EXAMPLE SORT ASCENDING

```
SELECT Author, Title
FROM book
ORDER BY Title ASC
```

Danny Champion of the World	Roald Dahl
Fantastic Beasts and Where to Find Them	JK Rowling
Going Solo	Roald Dahl
Harry Potter and the Chamber of Secrets	JK Rowling
Harry Potter and Order of the Phoenix	JK Rowling
Private Peaceful	Michael Morpurgo
The BFG	Roald Dahl
War Horse	Michael Morpurgo

EXAMPLE SORT DESCENDING

```
SELECT Author, Title
FROM book
ORDER BY Title DESC
```

War Horse	Michael Morpurgo
The BFG	Roald Dahl
Private Peaceful	Michael Morpurgo
Harry Potter and Order of the Phoenix	JK Rowling
Harry Potter and the Chamber of Secrets	JK Rowling
Going Solo	Roald Dahl
Fantastic Beasts and Where to Find Them	JK Rowling
Danny Champion of the World	Roald Dahl

WHERE CLAUSE

We can filter our selection using the `WHERE` clause

```
WHERE fieldname operator value
```

Operator	Description
=	Value equal to
!=	Value not equal to
<	Value less than
>	Value greater than
<=	Value less than or equal to
>=	Value greater than or equal to

SELECT USING WHERE CLAUSE

EXAMPLE 1 – SELECT BOOKS WRITTEN SINCE 2000

```
SELECT Title, Author, yearPublished
FROM book
WHERE YearPublished > 2000
```

Fantastic Beasts and Where to Find Them	JK Rowling	2001
Harry Potter and Order of the Phoenix	JK Rowling	2003
Private Peaceful	Michael Morpurgo	2003

EXAMPLE 2 – SELECT BOOKS WRITTEN BY MICHAEL MORPURGO

```
SELECT Title, Author
FROM book
WHERE Author = "Michael Morpurgo"
```

Notice how the author name is in speech marks because it is a string datatype.

War Horse	Michael Morpurgo
Private Peaceful	Michael Morpurgo

EXAMPLE 3 – SELECT BY DATE

```
WHERE Date < #1/1/2010#
```

For data type date you need to use #. Eg

BOOLEAN OPERATORS

We can use Boolean and relational operators with the WHERE clause if we have multiple conditions that need to be met.

Operator	Description
OR	Allows us to combine multiple conditions. Any of the conditions can be true for the overall expression to return true
AND	Allows us to combine multiple conditions. All conditions need to be true for the overall expression to return true
NOT	Reverses the value of a condition. If it is true it will be false and vice versa

EXAMPLE – SELECT ALL BOOKS WRITTEN BY MICHAEL MORPURGO SINCE 2016

```
SELECT Title, Author FROM book
WHERE Author="Michael Morpurgo"
AND YearPublished > 2000
```

Private Peaceful	Michael Morpurgo

UPDATE - TO UPDATE RECORDS IN A DATABASE

To make changes to a record that is already in a table we can use the `UPDATE` statement.

EXAMPLE 1: Update the book table to change the genre of all fields to Children

```
UPDATE book
SET Genre="Children"
```

EXAMPLE 2: Update the book table to change the author name from JK Rowling to Joanne Rowling.

```
UPDATE book
SET Author="Joanne Rowling"
WHERE Author="JK Rowling"
```

Book ID	Title	Author	Year Published	Publisher	Genre
1	Fantastic Beasts .	Joanne Rowling	2001	Bloomsbury	Children
2	Harry Potter ..	Joanne Rowling	1998	Bloomsbury	Children
3	Harry Potter ..	Joanne Rowling	2003	Bloomsbury	Children
4	The BFG	Roald Dahl	1982	Penguin	Children
5	Going Solo	Roald Dahl	1986	Jonathan Cape	Children
6	Danny .	Roald Dahl	1975	Jonathan Cape	Children
7	War Horse	Michael Morpurgo	1982	Kaye & Ward	Children
8	Private Peaceful	Michael Morpurgo	2003	HarperCollins	Children

INSERT INTO - ADDING NEW RECORDS

INSERT INTO is a commonly used command in SQL for adding new records to database tables. To insert all attributes for a table we can use:

```
INSERT INTO table
VALUES(value1, value2,…)
```

EXAMPLE

```
INSERT INTO book
VALUES ('Boy', 'Roald Dahl', 1984, 'Penguin',
'Autobiography')
```

Sometimes we do not enter data into every field. Instead we can explicitly state which fields we would like to add the data to.

```
INSERT INTO table (field1, field2,…)
VALUES(value1, value2,…)
```

The values correspond to the fields in the table i.e.:

- ✓ Field 1: Book ID
- ✓ Field 2: Title
- ✓ Field 3: Author
- ✓ Field 4: YearPublished
- ✓ Field 5: Publisher
- ✓ Field 6: Genre

EXAMPLE

```
INSERT INTO book (Title, Author, YearPublished,
Publisher, Genre) VALUES ('Boy', 'Roald Dahl', 1984,
'Penguin', 'Autobiography')
```

DELETING RECORDS

To delete a record we specify which record(s) from which table we wish to remove.

```
DELETE FROM table WHERE condition
```

EXAMPLES

Remove all books

```
DELETE FROM book
DELETE * FROM book
```

The `WHERE` clause is used to filter records so that we do not apply a statement to a whole table.

Remove all books written by JK Rowling:

```
DELETE FROM book WHERE Author='JK Rowling'
```

Remove all books written by Michael Morpurgo and written before 2000

```
DELETE FROM book WHERE Author='Michael Morpurgo' AND
YearPublished < 2000
```

SELECT ATTRIBUTES FROM MULTIPLE TABLES

So far we have looked at a database made up of a single table. databases can be made up of multiple tables. We can link tables together using primary keys and foreign keys. We can use SQL statements to select data from multiple tables. When selecting the data from multiple tables we need to specify the name of the table from which each attribute we are wishing to retrieve.

We will use the following database table as an example case study.

Primary key — Author Table

AuthorID	FirstName	Surname	LiteraryAgent
1	Joanne	Rowling	Neil Blair
2	Roald	Dahl	David Higham Associates
3	Michael	Morpurgo	David Higham Associates

Foreign key — Book Table

BookID	AuthorID	Title	Surname	YearPublished	Publisher
1	1	Fantastic Beasts and Where to Find Them	2001	Bloomsbury	Fantasy
2	1	Harry Potter and the Chamber of Secrets	1998	Bloomsbury	Fantasy
3	1	Harry Potter and Order of the Phoenix	203	Bloomsbury	Fantasy
4	2	The BFG	1982	Penguin	Fantasy
5	2	Going Solo	1986	Jonathan Cape	Autobiography
6	2	Danny Champion of the World	1975	Jonathan Cape	Children
7	3	War Horse	1982	Kaye & Ward	Historical fiction
8	3	Private Peaceful	2003	HarperCollins	Historical fiction

We need to specify that we only wish to select the records where the primary key and foreign key match.

EXAMPLES

Retrieve data book title and author surname

```
SELECT book.Title, author.Surname
FROM author, book
WHERE author.AuthorID=book.AuthorID
```

Fantastic Beasts and Where to Find Them	JK Rowling
Harry Potter and the Chamber of Secrets	JK Rowling
Harry Potter and Order of the Phoenix	JK Rowling
The BFG	Roald Dahl
Going Solo	Roald Dahl
Danny Champion of the World	Roald Dahl
War Horse	Michael Morpurgo
Private Peaceful	Michael Morpurgo

Retrieve book title and author surname where genre is *fantasy*

```
SELECT book.title, author.surname
FROM author, book
WHERE author.AuthorID=book.AuthorID
AND book.Genre="Fantasy"
```

Fantastic Beasts and Where to Find Them	JK Rowling
Harry Potter and the Chamber of Secrets	JK Rowling
Harry Potter and Order of the Phoenix	JK Rowling
The BFG	Roald Dahl

Retrieve book title and author surname where genre is fantasy and sort in descending order Title

```
SELECT book.title, author.surname
FROM author, book
WHERE author.AuthorID=book.AuthorID
AND book.Genre="Fantasy"
ORDER BY title DESC
```

The BFG	Roald Dahl
Harry Potter and Order of the Phoenix	JK Rowling
Harry Potter and the Chamber of Secrets	JK Rowling
Fantastic Beasts and Where to Find Them	JK Rowling