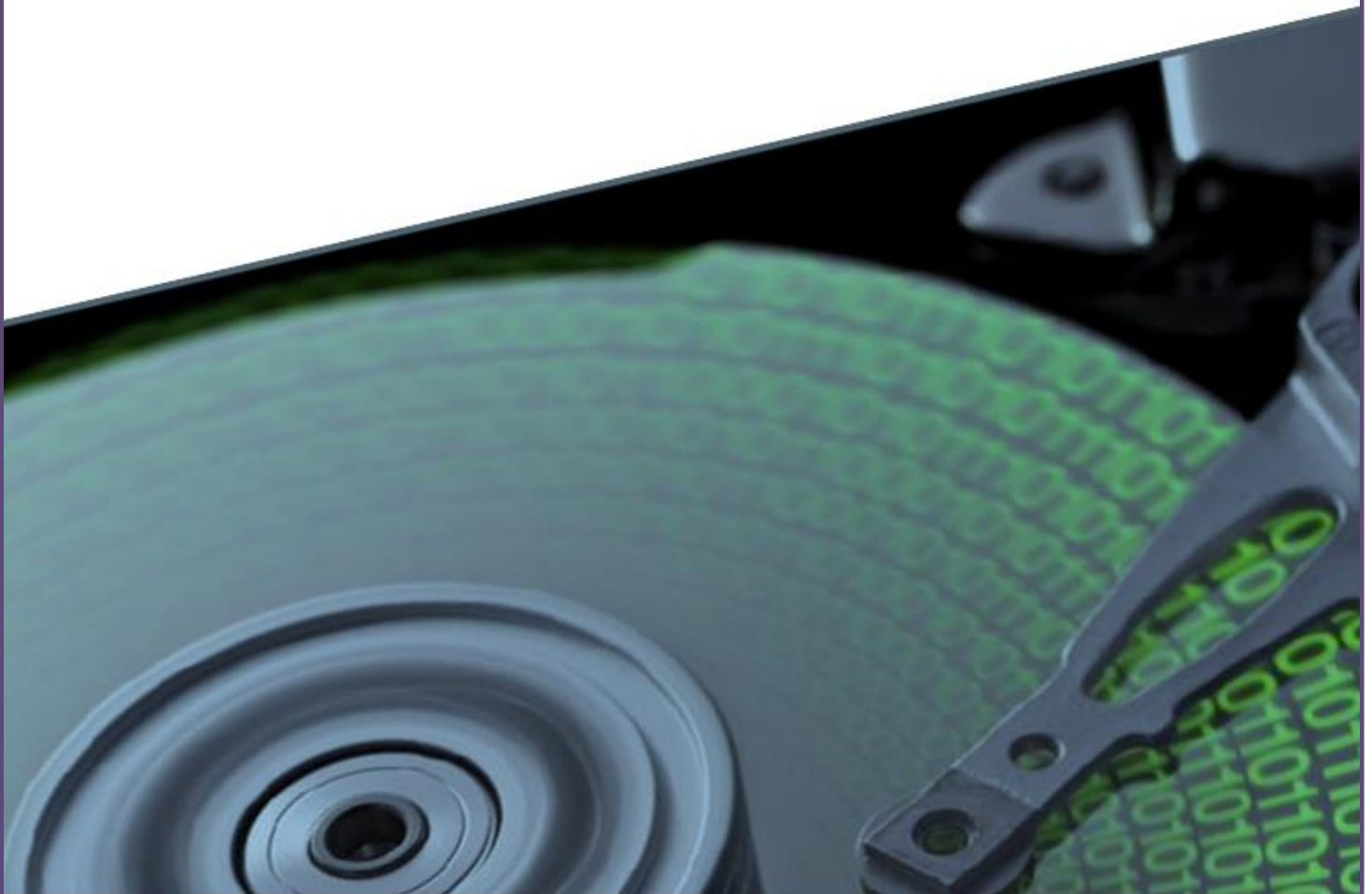


N5



National 5

Database Design & Development



Database design and development

Candidates develop knowledge, understanding and practical problem-solving skills in database design and development, through a range of practical and investigative tasks. This allows candidates to apply computational-thinking skills to analyse, design, implement, test, and evaluate practical solutions, using a range of development tools such as SQL. Tasks involve some complex features (in both familiar and new contexts), that require some interpretation by candidates.

What you need to know?

Database design and development	
Analysis	Identify the end-user and functional requirements of a database problem that relates to the implementation at this level.
Design	<p>Describe and identify the implications for individuals and businesses of the Data Protection Act 1998:</p> <ul style="list-style-type: none"> ◆ prior consent of data subject ◆ accuracy of data ◆ data used for limited, specifically stated purposes ◆ data kept safe and secure <p>Describe and exemplify entity-relationship diagrams with two entities indicating:</p> <ul style="list-style-type: none"> ◆ entity name ◆ attributes ◆ relationship (one-to-many) <p>Describe and exemplify a data dictionary:</p> <ul style="list-style-type: none"> ◆ entity name ◆ attribute name ◆ primary and foreign key ◆ attribute type: <ul style="list-style-type: none"> — text — number — date — time — Boolean ◆ attribute size ◆ validation: <ul style="list-style-type: none"> — presence check — restricted choice — field length — range <p>Exemplify a design of a solution to the query:</p> <ul style="list-style-type: none"> ◆ multiple tables ◆ fields ◆ search criteria ◆ sort order

Implementation	<p>Implement relational databases with two linked tables, to match the design with referential integrity.</p> <p>Describe, exemplify and implement SQL operations for pre-populated relational databases, with a maximum of two linked tables:</p> <ul style="list-style-type: none"> ◆ select: <ul style="list-style-type: none"> — from — where: <ul style="list-style-type: none"> ○ AND, OR, <, >, = ○ order by with a maximum of two fields ◆ insert ◆ update ◆ delete ◆ equi-join between tables <p>Read and explain code that makes use of the above SQL.</p>
Testing	<p>Describe and exemplify testing:</p> <ul style="list-style-type: none"> ◆ SQL operations work correctly at this level
Evaluation	<p>Evaluate solution in terms of:</p> <ul style="list-style-type: none"> ◆ fitness for purpose ◆ accuracy of output

Analysis

Before designing and implementing a database it is important to consider the end-user and functional requirements of the database to ensure it is designed and implemented correctly.

Similar to analysing programs, it is important to identify the **inputs** and **outputs** for a database.

The inputs for a database are the fields (single piece of information) to be held in the database. E.g. Name, Address etc.

The outputs are the results of any queries and the information to be displayed from these queries.

Example:

Active Scotland run sporting events at an organised gathering every July. Athletes compete in a range of sports and pay an entry fee which is different for each sport. Competitors in any sport that costs more than £50 to enter receive a free lunch. The database should be used to identify all competitors eligible for a free lunch.

CODE	FIRST NAME	SURNAME	DATE OF BIRTH	LOCATION	SPORT CODE	SPORT	SPORT FEE
1	Chris	Hunter	07/09/1982	Northern Ireland	SP005	Hockey	£38.50
2	Steve	Florence	03/07/1969	South	SP003	Badminton	£12.50
3	Michael	Danson	12/07/1978	South	SP002	Athletics	£35.00
4	Michael	Gilchrist	19/07/1987	South West	SP001	Aquatics	£20.00
5	Jo	Grainger	14/08/1983	Scotland	SP008	Rugby 7s	£45.00
6	Michael	Casey	04/08/1978	London	SP001	Aquatics	£20.00
7	Kerry	Fallon	05/08/1985	Yorkshire	SP002	Athletics	£35.00
8	Alistair	McMillan	26/07/1988	South East	SP008	Rugby 7s	£67.00
9	Colin	Farah	04/05/1982	Midlands	SP005	Hockey	£38.50
10	Alan	Pennie	24/07/1984	Scotland	SP002	Athletics	£35.00
11	Chris	Brown	01/02/1987	London	SP002	Athletics	£35.00
12	Ally	Cooper	05/06/1978	Midlands	SP003	Badminton	£12.50
13	Tanni	Grey	01/01/1986	South West	SP003	Badminton	£12.50
14	Denise	Clark	25/10/1980	Wales	SP005	Hockey	£38.50
15	Graeme	Burton	05/08/1987	Scotland	SP004	Boxing	£55.00
16	Chris	Dale	19/05/1986	London	SP002	Athletics	£35.00
17	Cameron	McClatchey	30/11/1987	South West	SP008	Rugby 7s	£67.00
18	Ellen	McConnell	12/03/1978	Scotland	SP007	Netball	£11.00
19	Mark	Barlett	12/07/1981	North East	SP004	Boxing	£55.00
20	Colin	Renwick	12/08/1981	Scotland	SP009	Squash	£5.00

Inputs:

- Code
- First Name
- Surname
- Date of Birth
- Location
- Sport Code
- Sport
- Sport Fee

Outputs:

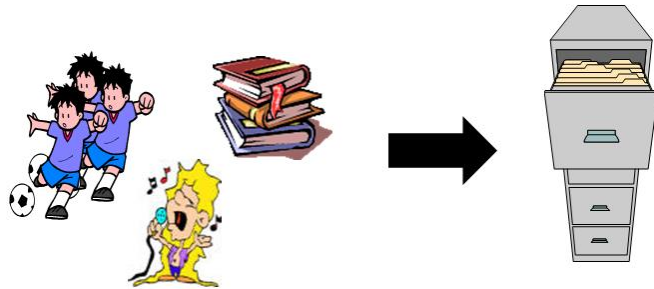
The database should use a query to produce a list of competitors who have participated in any sport that costs more than £50.

Design

Database Structure

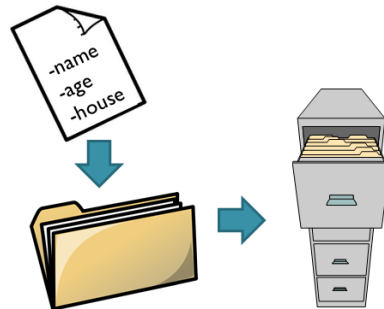
A database is a collection of **information** on a particular topic that is **structured** in some way.

For example you could have a database on football teams, CD collection or books.



A database is organised into:

- Fields
- Records
- Files



Field

A field is a **single piece of information** about a person or thing in the database.

A database about pupils in a school might store information such as:

First Name

Surname

Date of Birth

Year Group

House

Each item is stored in a separate Field.

So, we need 5 fields to store these details

Record

A record is a **collection of fields** about a particular person or thing.

A record would be created for each pupil in a school.

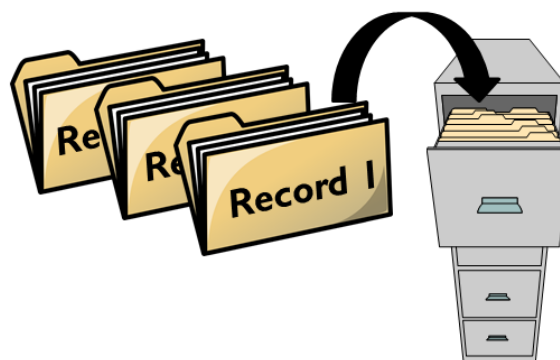
*A class with 20 pupils
would need 20 records*

	Record 1	Record 2	Record 3
First Name	Peter	Jennifer	Cara
Surname	Jones	Smith	Brown
Date of Birth	15/03/99	21/10/98	03/01/00
Year Group	4	5	3
House	Arran	Bute	<u>Cumbræ</u>

File

A file is a **collection of records** on a particular topic.

A file would contain a record for **every** pupil in a school.



School Database File

Field Types

Each field in a database must be assigned a field type.

Field types specify what sort of data the field will hold. There are several field types available.

Basic Field Types:

Fields containing both letters and numbers should have the **TEXT** field type.

Phone numbers should also be **TEXT** as a **NUMBER** field would drop the zero at the start.

Field Type	Example
Text	ABCDEF
Number	123456
Date	01/12/13
Time	15:40

Fields can also be used to store more complex information

Other Field Types:

Field Type	Example
Boolean	Yes / No
Graphic	
Object	
Link	www.google.com

Flat File Database Problems

Flat file databases are simple to create and easy to understand.

However, they can cause problems that make them unreliable.



Errors in the database can happen because of:

- **Data Duplication**
- **Modification errors**

Data Duplication

The database below contains a large amount of data duplication.

Many of the books below belong to the same publisher meaning that the publisher information has had to be entered several times.

Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Co	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Road, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	14 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

When data is duplicated, it can also lead to **inconsistency**. Several of the publisher details below have been entered incorrectly, even although they refer to the same publisher.

Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Co	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Road, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	14 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

Modification Errors

Insertion problems occur when data has to be included more often than is necessary. In the

Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Co	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Road, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	14 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

example below, the publisher details must be entered every time a new book is added.

It is also not possible to store details of a publisher without storing details about a book.

Deletion problems occur when deleting a single record also removes important data that is

Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Co	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Road, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	14 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

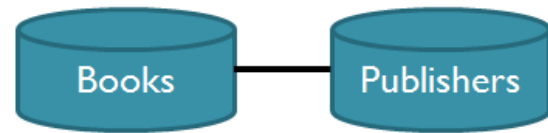
still required.

Removing *The Narrow Escape* book will also remove the only record of *London Publishing Ltd*

Relational Databases

Relational databases overcome the problems caused by flat file databases. Data is entered and stored **once**, removing errors and inconsistency caused by data duplication.

Relational databases use **linked tables** (more than one database table joined together).



The design of a relational database is mainly concerned with three things:

- Entities
- Attributes
- Relationships

An **entity** represents a person, object or event

- e.g. Staff Member, DVD, Booking

Any system can be represented as a collection of one or more 'objects', 'things' or entities

Each entity has a set of **attributes** which describe examples or **instances** of that entity.

e.g. The product entity below has the attributes, **product ID, Description, Price, Supplier ID**

Product
Product ID
Description
Price
Supplier ID

Supplier
Supplier ID
Name
Address
Telephone

Primary Key

All entities must have a primary key. A primary key is used to uniquely identify each record in a database.



Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Company	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Street, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	15 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

The value entered in the primary key field should **never** be repeated in another record.

Common primary keys are:

DVLA	Car Registration	SA60 ABC
Government	National Insurance Number	MN 23 84 96 K
Banks	Account Number	02994812
Online Retail	Order Number	EX782990
NHS	NHS Number	943 476 5919

Primary keys narrow down search results to a single record.

Normalisation

Normalisation involves removing the **repeating items**.

Book Title	Author	ISBN	Publisher Name	Publisher Address
The Haunted House	Hugo First	999-2-7712-3343-2	Glasgow Book Company	25 Square Road, Glasgow
Easy Money	Robin Banks	229-8-3312-0022-1	Glasgow Book Company	25 Square Road, Glasgow
Spectacles	Seymore Withem	777-3-4429-3321-0	Oxford Publishing	25 High Street, Oxford
Swimming the Channel	Frances Near	190-7-9100-2232-1	Oxford Publishing	25 High Street, Oxford
Horse Racing	Willie Winnit	987-5-3234-1212-2	Oxford Publishing	25 High Street, Oxford
Danger	Luke Out	889-1-2321-2222-6	Oxford Publishing	25 High Street, Oxford
Falling Down	Eileen Dover	972-2-3342-1001-8	Penguin Books	15 Main Place, London
Simple Dentistry	Phil McCavity	569-1-4544-2233-9	Penguin Books	15 Main Place, London
Carpet Laying	Walter Wall	122-1-7712-3312-2	Penguin Books	15 Main Place, London
The Narrow Escape	Justin Thyme	766-3-2210-9023-8	London Publishing Ltd	1 Straight Street, London

From the 10 records above, there are actually only 4 different publishers.



Book Title	Author	ISBN
The Haunted House	Hugo First	999-2-7712-3343-2
Easy Money	Robin Banks	229-8-3312-0022-1
Spectacles	Seymore Withem	777-3-4429-3321-0
Swimming the Channel	Frances Near	190-7-9100-2232-1
Horse Racing	Willie Winnit	987-5-3234-1212-2
Danger	Luke Out	889-1-2321-2222-6
Falling Down	Eileen Dover	972-2-3342-1001-8
Simple Dentistry	Phil McCavity	569-1-4544-2233-9
Carpet Laying	Walter Wall	122-1-7712-3312-2
The Narrow Escape	Justin Thyme	766-3-2210-9023-8

We now have separate entities
for Books and Publishers.



Publisher Name	Publisher Address
Glasgow Book Company	25 Square Road, Glasgow
Oxford Publishing	25 High Street, Oxford
Penguin Books	15 Main Place, London
London Publishing Ltd	1 Straight Street, London

But there is no obvious primary key for our new entity.



Adding an ID field to our
publishers entity gives us a
unique field to use as the
primary key.

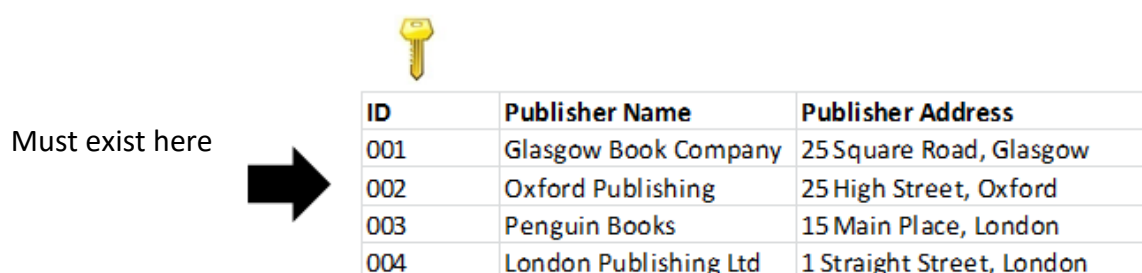
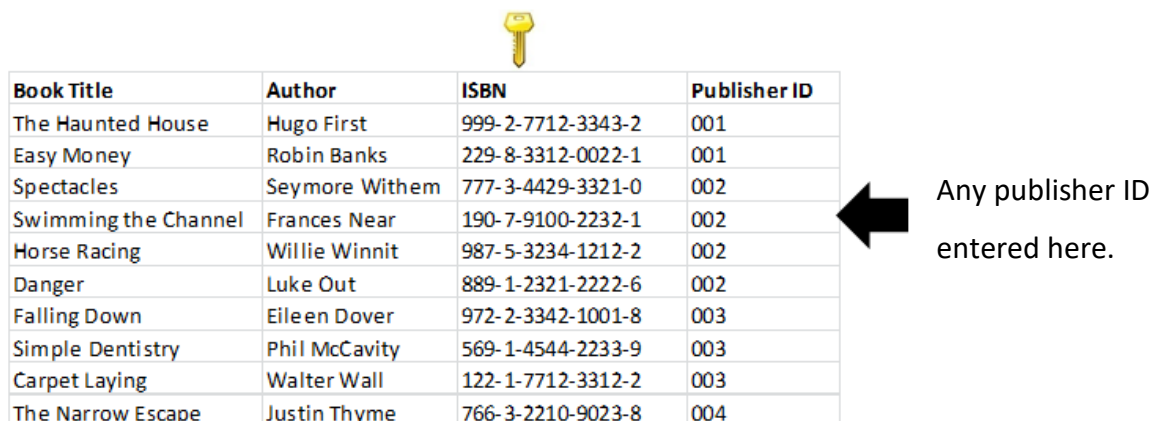
ID	Publisher Name	Publisher Address
001	Glasgow Book Company	25 Square Road, Glasgow
002	Oxford Publishing	25 High Street, Oxford
003	Penguin Books	15 Main Place, London
004	London Publishing Ltd	1 Straight Street, London

Foreign Key

Foreign keys are used to **link** entities in a relational database.



A **foreign key** is an attribute in one entity that is the **primary key** of another entity.



Our database is now in First Normal Form (1NF) with two tables.

This can be written as:

Books (Book Title, Author, ISBN, *Publisher ID)

Publisher (ID, Publisher Name, Publisher Address)

Primary Key - Underlined fields

*Foreign Key - * fields*

Relationships

A relationship is a significant real world association between entities.

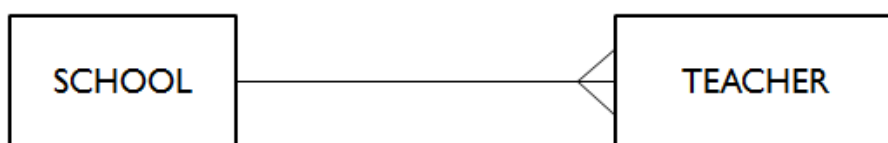
It describes how the two entities are related and can be thought of a connection between them.

Entity Relationship Diagrams

An entity-relationship diagram is a graphical representation of the entities in a system. It is used to illustrate the relationship that exists between two or more entities.

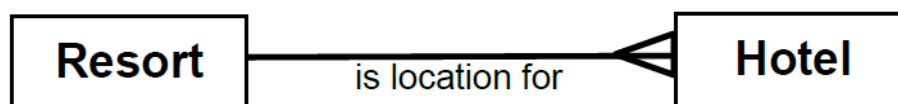
One-to-many

In this type of relationship, every instance in one entity is linked with **several** instances in another entity.



This entity relationship diagram shows a one-to-many relationship (*crow's feet at the many end*).

Each school can have several teachers but each teacher belongs to just one school.



Each resort can have several hotels but each hotel belongs to just one resort.

Validation

Validation is used to make sure that the data entered into fields is in the correct format (does it make sense).

The database can be set up to automatically check data using validation rules:

- **Presence Check**
- **Restricted Choice**
- **Length Check**
- **Range Check**


Presence Check

A presence check is used to make a field a 'required field'.

This means that for each record, data **must** be entered into that field'

Presence Check
↓

First Name	Surname	House
Peter	Jones	Arran
Jennifer	Smith	
Cara	Brown	Cumbræ

 *Field must be filled in*

Restricted Choice

A restricted choice makes users select information for a field from a pre-defined list.

Restricted Choice
↓

First Name	Surname	House
Peter	Jones	<div><div>▼</div><div>Arran Bute Cumbræ Lomond Kintyre</div></div>

This means that only certain pieces of information can be entered into a field.

Length Check

A length check makes sure that the number of characters entered into a field is of a set length.

This is useful in fields containing codes or phone numbers that are always the same size.

Length Check



First Name	Surname	Phone
Peter	Jones	01292123456
Jennifer	Smith	01292654321
Cara	Brown	01292123



Field must be 11 characters

Range Check

A range check makes sure that the value entered into a field falls with upper and lower limits.

This is useful in number fields where the expected range is known.

Range Check



First Name	Surname	Age
Peter	Jones	46
Jennifer	Smith	16
Cara	Brown	14



Field must be 11 to 18

Search and Sort

Searching and Sorting are operations that can be performed on a database in order to arrange information in different ways.

Search reduces the number of records displayed to those matching a set criteria.



Sort arranges displayed records alphabetically (A-Z), numerically (1-10) or chronologically (Jan-Dec).

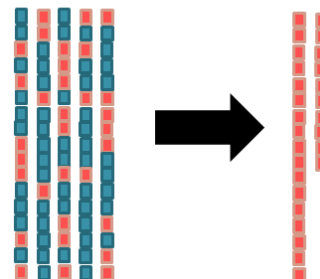


Simple Search

A simple search is when a search criteria uses **one field** to **find** matching records.

Only records that contain data **matching the criteria** are displayed.

Criteria: Gender = "female"

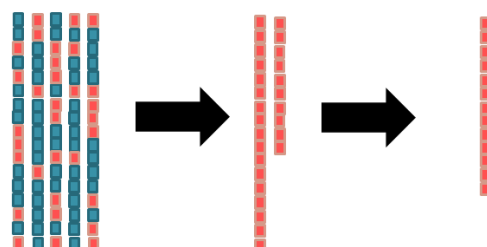


This **reduces** the overall **number of records** displayed.

Complex Search

A complex search is when a search criteria uses **more than one field** to find matching records.

Criteria: Gender = "female" **AND** Age > 14



Simple Sort

Records can be arranged in two ways:



A simple sort is when records are arranged in order using **one** field.

First Name	Surname	House	Age
Peter	Jones	Arran	15
Jennifer	Smith	Bute	16
Cara	Brown	Cumbræ	14
Adam	White	Kintyre	16
Lucy	Campbell	Lomond	13

Sort in **Ascending** order by **House**

First Name	Surname	House	Age
Jennifer	Smith	Bute	16
Adam	White	Kintyre	16
Peter	Jones	Arran	15
Cara	Brown	Cumbræ	14
Lucy	Campbell	Lomond	13

Sort in **Descending** order by **Age**

Complex Sort

A complex sort is when records are arranged in order using **more than one** field.

Sort in **Descending** order by **Age** and then
Ascending order by **House**

First Name	Surname	House	Age
Jennifer	Smith	Bute	16
Ben	Green	Bute	16
Adam	White	Kintyre	16
Peter	Jones	Arran	15
Eve	Smith	Bute	15
Andrew	Jones	Arran	14
Chloe	McDonald	Kintyre	14
Daniel	Morris	Kintyre	14
Lucy	Campbell	Lomond	14

Data Dictionary

A **data dictionary** is used to record details about the database. It provides a description of the constraints or rules that apply to each of the attributes of each entity in the system.

A data dictionary is simply a large table that stores **metadata** – in other words, it stores data about data.

The structures needed to store data in the database are **planned** using a data dictionary.

Entity: Resort					
Attribute Name	Key	Type	Size	Required	Validation
resortID	PK	Number		yes	
town		Text	20	yes	
resortType		Text	20	yes	Restricted choice: coastal, city, island
trainStation		Boolean		yes	
Entity: Hotel					
Attribute Name	Key	Type	Size	Required	Validation
hotelRef	PK	Text	4	yes	length=4
hotelName		Text	20	yes	
phoneNumber		Text	11	yes	length=11
resortID	FK	Number		yes	Existing resortID from Resort table
starRating		Number		yes	Range: >=1 and <=5
seasonStartDate		Date		no	
swimmingPool		Boolean		yes	
mealPlan		Text	17	yes	Restricted choice: RO, BB, HB, FB
checkInTime		Time		yes	Range: >=14:00 and <=16:00

- Each row provides details about **one attribute** in the system
- Each column **specifies a rule** or restriction that applies to the attributes.

Each row of the data dictionary is completed by stating the appropriate value or rule for each column

Where no rules apply, the column value is left empty.

- **PK/FK** (Is the field a primary key / foreign key – a field can be both)
- **Data Types** (Text, Number, Boolean, Date/time Object, Link)
- **Size** (Maximum number of characters /digits)
- **Required** (Mandatory or Optional. Primary key must be required)
- **Validation** (Presence/Length/Range checks, restricted choice)

Queries

Queries are used to:

- **search** the tables in order to retrieve sets of data that match the requirements
- **sort** the data in the tables into ascending or descending order
- **update** information in multiple records

Before creating a query, the developer should plan:

- what tables are needed
- what fields to use
- field order for a complex sort – the most important field must nearest the left of the query window

Planning the design of a query before creating the SQL code is good practice. This gives the developer time to think carefully about the fields that are required, which in turns, helps them to identify the table or tables that will be needed.

It also allows developers to consider the purpose of the query (search and/or sort), together with any required search criteria and/or sort order. Planning ahead helps to reduce the errors that developers may otherwise encounter when working with the SQL code.

Example

A relational database is used by a travel agency to store details of Scottish holiday resorts and hotels in each resort. The resort and hotel details are arranged in two separate tables called **Resort** and **Hotel**. The structure of the tables is shown below:

	Field Name
🔑	resortID
	town
	resortType
	trainStation

	Field Name
🔑	hotelRef
	hotelName
	phoneNumber
	resortID
	starRating
	seasonStartDate
	swimmingPool
	mealPlan
	checkInTime

Example 1

Design a query to list the town name and train station details of all resorts that have a train station.

Field(s)	town, trainStation
Table(s)	Resort
Search criteria	trainStation = true
Sort order	

Example 2

Design a query to list the hotel name and phone number, together with the star rating and swimming pool details for all hotels with a swimming pool that have a rating of at least 4 stars.

Field(s)	hotelName, phoneNumber, starRating, swimmingPool
Table(s)	Hotel
Search criteria	swimmingPool = true AND starRating >= 4
Sort order	

Example 3

Design a query to list the hotel name and its star rating, together with the town, resort type and check-in time, of all hotels that allow check in before 15:00. These details should be displayed so that the hotel with the highest rating is listed first; hotels with the same star rating should be listed in alphabetical order of town name.

Field(s)	hotelName, starRating, town, resortType, checkInTime
Table(s)	Hotel, Resort
Search criteria	checkInTime < 15:00
Sort order	starRating DESC, town ASC

Data Protection Act

The Data Protection Act is used to control how **information that is stored on computer** is handled.

It gives **legal protection** to anyone whose personal data is held on computer by a company or organisation.

It is the **responsibility of the company** holding the personal data to ensure that all the requirements of the law are met.

The **data subject** is the person whose data is being stored. The data subject must provide consent for their data to be stored.

The **data user** or data processor is usually an employee of the company who can access and use the information stored about the data subject.

The data controller is the person who is wholly responsible for holding the data. They are usually the company boss/manager or owner.



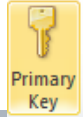
All data must be:	Data Subject has the right to:
<ul style="list-style-type: none"> • Fairly and lawfully processed. • Only used for specified purposes (the reason given). • Not shown to anyone except the subject • Adequate, relevant and not excessive. • Accurate and kept up to date. • Kept safe and secure to prevent hacking. • Not kept longer than is necessary • Not transferred to other countries without adequate protection 	<ul style="list-style-type: none"> • See data held about themselves • Know the purpose of the data being held about them • Have errors in the data corrected. • Be compensated if Act is broke. • Prevent direct marketing (junk mail) by writing to the data controller. • Prevent the processing of data that is likely to cause damage or distress

Implementation

When building a database it is important to ensure it matches the design (data dictionary).

Implementing Primary Keys

Open each table in design view. Select a field and click the Primary Key option.



TutorGroup		
	Field Name	Data Type
	TGCode	Text
	Tutor	Text
	Room	Text
	Day	Text
	Time	Text

Student		
	Field Name	Data Type
	StudentNo	Number
	Forename	Text
	Surname	Text
	Address	Text
	Town	Text
	Postcode	Text
	TGCode	Text

Setting Field Size and Required

Select each field in turn (for each table) and change the size, required and indexed settings to match the data dictionary requirements.

General	
Lookup	
Field Size	30
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	Yes
Allow Zero Length	Yes
Indexed	Yes (No Duplicates)
Unicode Compression	No
IME Mode	No Control
IME Sentence Mode	None
Smart Tags	

Size:

Set the maximum number of characters allowed here. (Text fields only)

Required (Presence Check):

Set required to Yes or No.

Validation Rules

Add the validation rules for the fields indicated on the data dictionary

General	
Lookup	
Display Control	Combo Box
Row Source Type	Value List
Row Source	"Multiple Choice";"Short Answer";"Extended Response";"Practical"
Bound Column	1
Column Count	1
Column Heads	No

Restricted Choice

General		Lookup
Field Size	Long Integer	
Format	General Number	
Decimal Places	Auto	
Input Mask		
Caption		
Default Value		
Validation Rule	>=0 And <=100	
Validation Text	Value must be 0 - 100	
Required	No	

Range Check

General		Lookup
Field Size	Long Integer	
Format		
Decimal Places	Auto	
Input Mask		
Caption		
Default Value		
Validation Rule	Len([StudentNo])=6	
Validation Text	Student No must be 6 digits	
Required	No	

Length Check

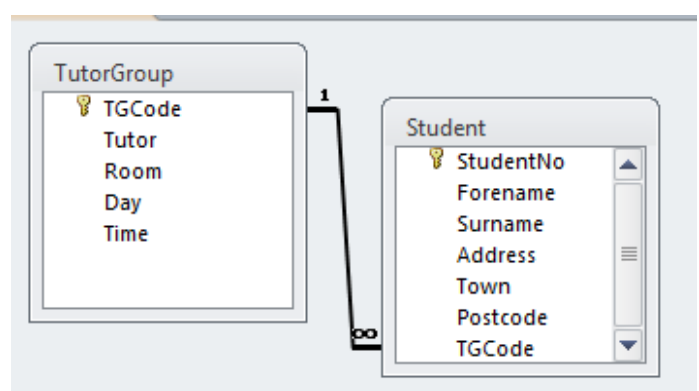
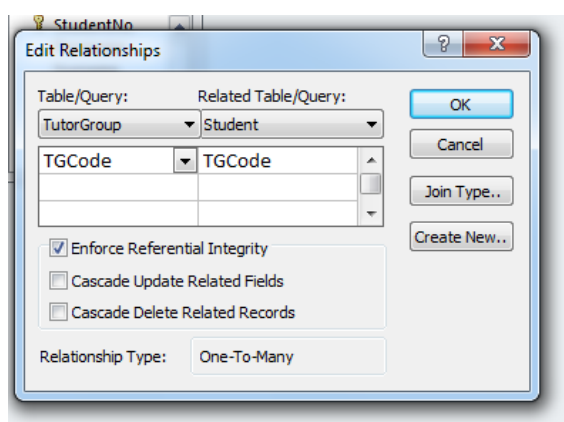
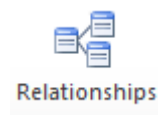
Creating Relationships (Implementing Relational Database)

Make sure all tables are closed.

Open the relationships option (from the Database Tools menu).

Add the two tables and create the relationships shown in the entity-relationship diagram.

Click on the **primary key** in the main table and drag your mouse over to the foreign key in the other table. Tick the box to **Enforce Referential Integrity**. Click create.



Creating Queries

SQL stands for **Structured Query Language** and it is used for creating queries.

Queries are used to:

- **search** the tables in order to retrieve sets of data that match the requirements
- **sort** the data in the tables into ascending or descending order
- **add, remove** and **change** information in the database

The main SQL statements you will use are:

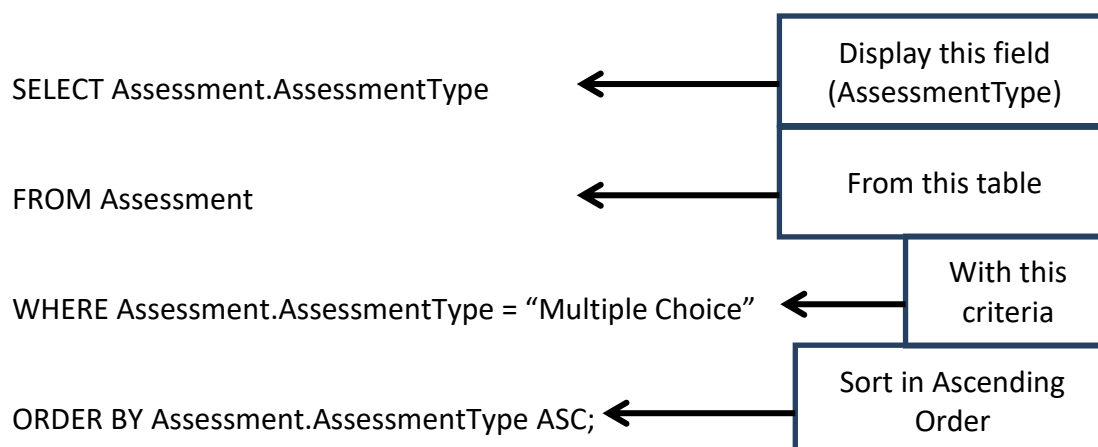
SELECT – Choose fields to be displayed as part of a query

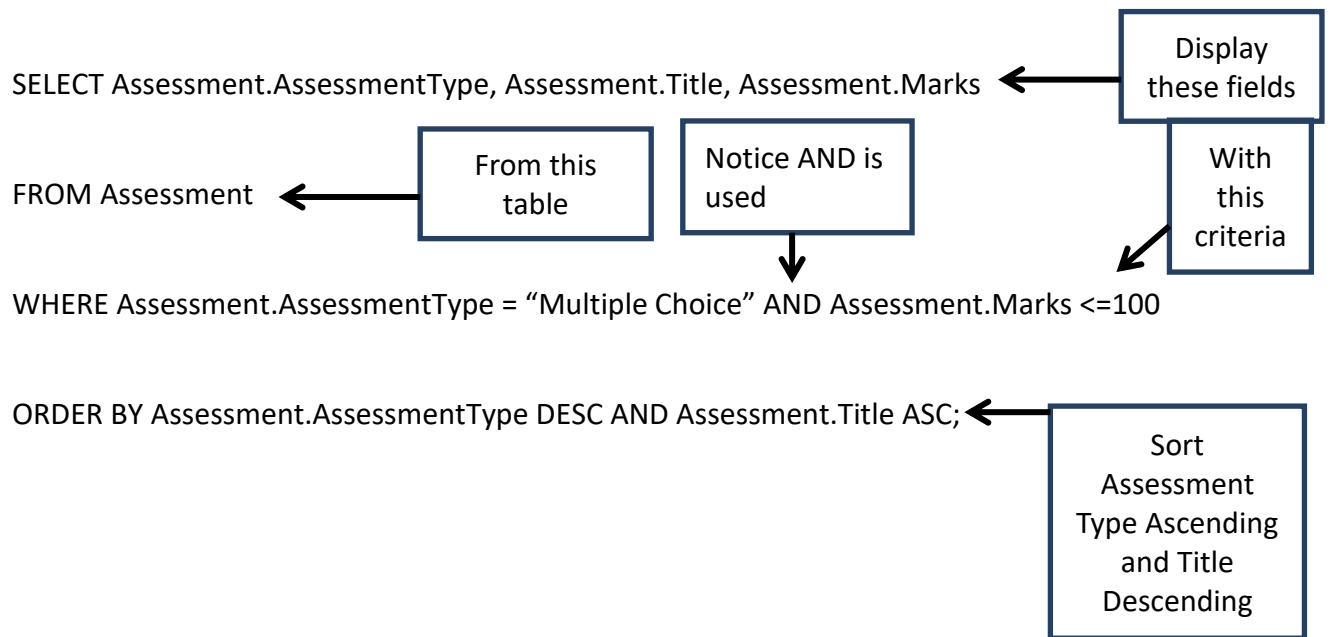
INSERT – Add new data to a table (add records)

UPDATE – Amend data in a table (update records)

DELETE – Remove data from a table (remove records)

Select Queries

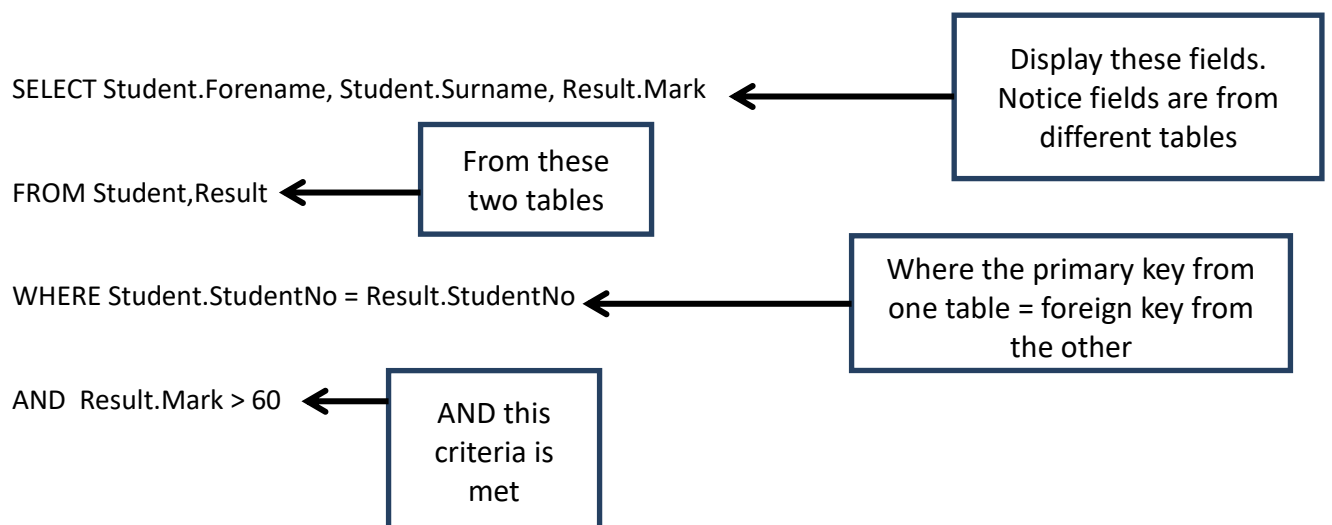




When creating a query there can be multiple fields used for the criteria and >, <, =, >=, <= can be used in the criteria. E.g.

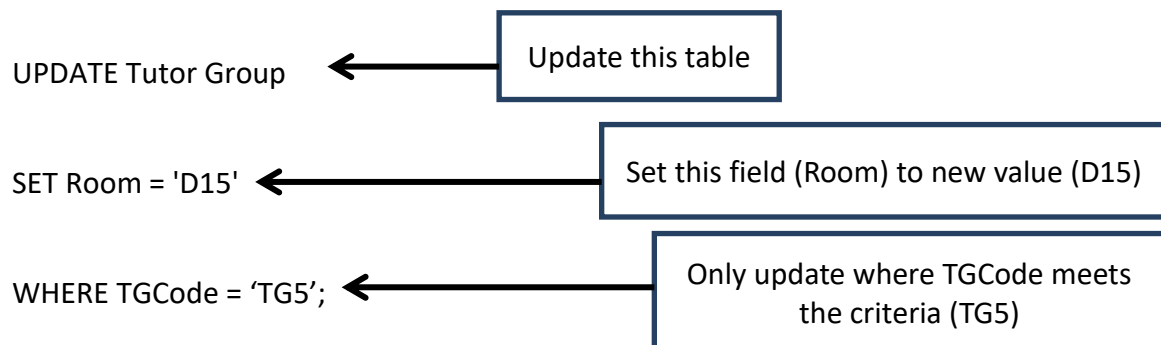
WHERE Assessment.AssessmentType = "Multiple Choice" **AND** Assessment.Marks <=100
 OR
 Assessment.AssessmentType = "Multiple Choice" **OR** Assessment.Marks >50

Select Queries (Equi-Join)

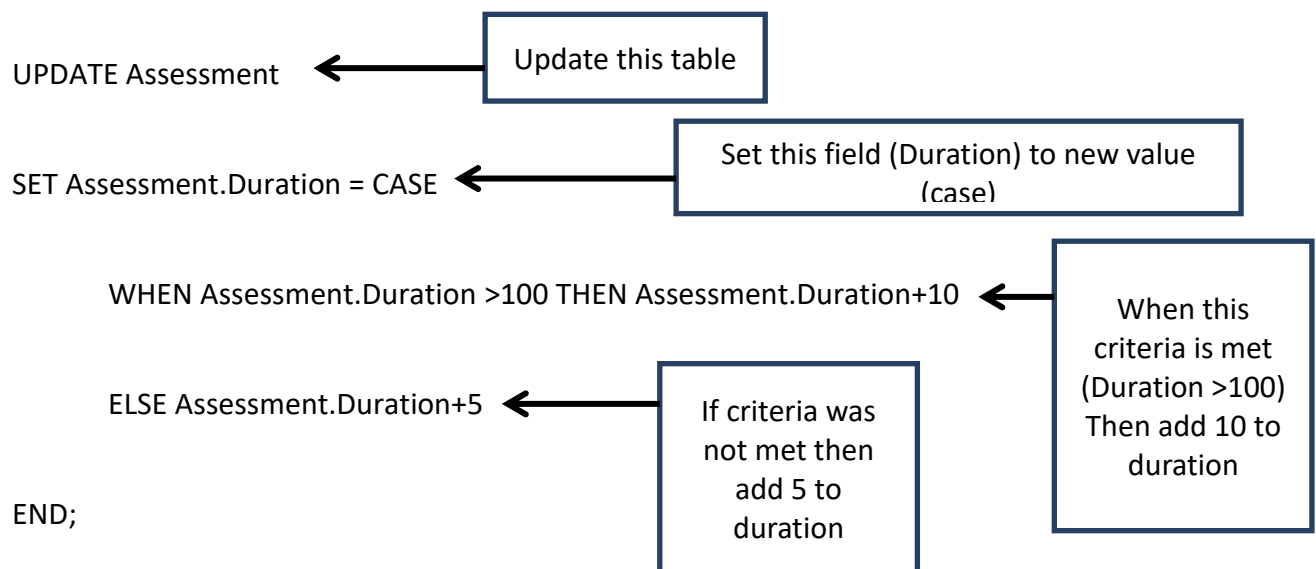


Update Queries

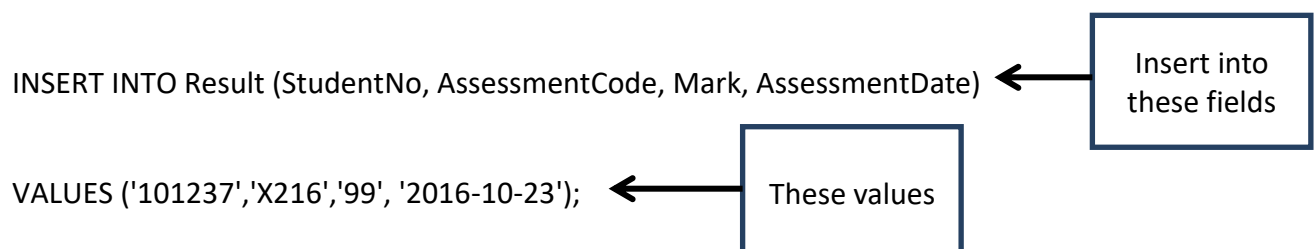
Unconditional



Conditional- will update when a condition is met




Insert Queries



Delete Queries

DELETE FROM Result WHERE Result.StudentNo = "101237";



Delete records from
table (Result) where
field meets criteria
(StudentNo = 101237)

For more help with SQL queries visit <https://www.w3schools.com/sql/>

Testing and Evaluation

It is important to test your database queries to ensure the **expected output** matches the **actual output**. Below is an example of how we should test a database:

Consider the **Customer** table shown below.

custo	foreName	surname	street	town	package	directDebit	paymentDu
101	Lauren	Calder	2 Paisley Street	Wemyss Bay	Premier	<input checked="" type="checkbox"/>	12/04/2016
102	Abigail	Cameron	16 Paisley Street	Wemyss Bay	Standard	<input type="checkbox"/>	12/04/2016
103	Ryan	Collins	17 Dunoon Drive	Gourock	Premier	<input checked="" type="checkbox"/>	19/05/2016
104	Nicole	Rutherford	5A Panama Place	Port Glasgow	Large	<input type="checkbox"/>	13/04/2016
105	Justine	O'Docherty	7 High Street	Kilmacolm	Premier	<input checked="" type="checkbox"/>	18/04/2016
106	Shelby	Sweeney	3 Paisley Road	Port Glasgow	Premier	<input checked="" type="checkbox"/>	26/05/2016
107	Donald	McAndrew	1 Big Hill Avenue	Inverkip	Standard	<input checked="" type="checkbox"/>	01/08/2016
108	Rowan	Hastings	6 Clydeview Crescent	Gourock	Large	<input checked="" type="checkbox"/>	05/05/2016
109	Grant	Donaldson	9 Dunoon Drive	Gourock	Premier	<input type="checkbox"/>	07/04/2016
110	Christine	Flowers	63 Hamilton Drive	Greenock	Large	<input checked="" type="checkbox"/>	19/04/2016
111	Ross	Lambie	12 Paisley Road	Kilmacolm	Standard	<input checked="" type="checkbox"/>	27/03/2016
112	Paul	McGill	3C Cow Lane	Kilmacolm	Premier	<input type="checkbox"/>	10/04/2016
113	Jack	Shields	4 Brookside Close	Port Glasgow	Large	<input type="checkbox"/>	10/05/2016
114	Pauline	Milne	32 High Street	Kilmacolm	Premier	<input checked="" type="checkbox"/>	09/05/2016
115	Margaret	Rice	5 Drumchapel Square	Greenock	Standard	<input type="checkbox"/>	14/04/2016

Query testing

A query is required to display the full name and town of all customers who live in Gourock. The details should be listed in alphabetical order of customer surname. This table shows the output predicted from the query.

Expected output	Forename	Surname	Town
Details of customer listed first	Ryan	Collins	Gourock
Details of customer listed last	Rowan	Hastings	Gourock

Query evaluation

This answer table is the output from the query used to perform the task.

surname	town
Collins	Gourock
Donaldson	Gourock
Hastings	Gourock

Actual output

Comparing the answer table with the predicted output, it is possible to evaluate the query. The query is fit for purpose because it displays details of the three customers who live in Gourock and has arranged the details in ascending order of surname. However, the query output is not accurate because the answer table only shows details of surname and town; the forenames of the customers are missing from the answer table.