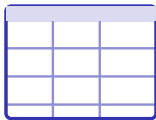


Databases

IGCSE Computer Science ■ Topic 9

IGCSE Computer Science



What we will cover

- 1 Tables, records & fields
- 2 Data types for fields
- 3 The primary key
- 4 Validation
- 5 SQL
- 6 Recap



Records and fields

field (a column)

record (a row)

StudentID	FirstName	DateOfBirth	FormClass	FeesPaid
1	Amy	14/03/2009	10A	TRUE
2	Ben	02/11/2008	10B	FALSE

primary key
(unique)

- A **record** is one row in the table – all the data about one thing (for example one student).
- A **field** is one column in the table – one piece of data that every record has (for example “First name”).

Field data types

text	a name
number	an age
date/time	a birthday
boolean	yes / no

Every field stores **one data type** ,
chosen to fit the data it holds.

- The type decides what the database will **let you do**: sort a date into the right order, do arithmetic on a number, test a Boolean as yes or no.
- **character** holds a single letter such as 'M' – the type most often forgotten.

The field data types

text/alphabetic

letters, digits and symbols – 10A, Amy

integer

a whole number – 42

real

a number with a decimal point – 3.5

Boolean

two states only – TRUE/FALSE, or Yes/No

date/time

stored as a date so it can be sorted and compared correctly

the trap

a phone number is **text**, not an integer – it may start with a zero, and you never do arithmetic on it

Pick the type by **what you will do with the field**, not by what it looks like. That is the single most-tested judgement in this topic.

The primary key

A **primary key** is the field that **uniquely** identifies each record – no two rows share it.

- a StudentID, an ISBN, an order number
- never blank, never repeated

A name is a **poor** key – two people share “Li Wei”. The key must be something guaranteed unique, which is why databases invent an ID field.



Validation in a database

age field: range check 0–120

enter 16 → accepted

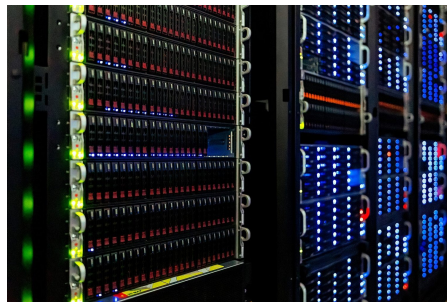
enter 200 → rejected

The database can **reject** bad data as it is entered:

- **range** – a mark 0–100
- **type** – a number, not letters
- **presence** – not left blank
- **format** – a date as dd/mm/yyyy

Validation stops **impossible** data at the door – but it cannot tell whether a *valid* value is the *right* one.

What is a database?



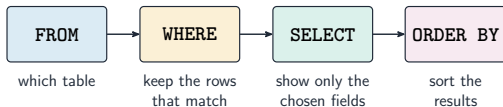
Large modern databases are stored on servers in data centres

A library card catalogue is a database on paper – records you can search and sort by index

Structured Query Language

SQL asks a database questions:

```
SELECT ... FROM ... WHERE ...
```



- **SELECT** the fields (columns)
- **FROM** the table
- **WHERE** a condition is met
- **ORDER BY** to sort

Read it in that order: pick columns, from a table, keeping only rows the WHERE allows.

Working out the output

To find the output of an SQL script, read it in this order:

WHERE – keep only the records that match the condition; 3.

Worked example – a query

List the name and mark of every student who scored above 80, best first.

```
SELECT Name, Mark  
FROM Students  
WHERE Mark > 80  
ORDER BY Mark DESC
```

WHERE filters **rows**; SELECT picks **columns**. DESC sorts high-to-low – leave it off and you get low-to-high.

SELECT picks fields

```
SELECT name  
FROM students  
WHERE age > 15
```

gets the names of
students older than 15

SELECT picks fields, FROM names the table, WHERE sets the condition

Reading a SELECT query

SELECT

which **fields** (columns) to show

FROM

which **table** to take them from

WHERE

which **records** (rows) to include – the condition

ORDER BY

how to sort the result, ascending or descending

reading one

SELECT FirstName, Class FROM Students WHERE Paid = TRUE shows the first name and class of **every student who has paid the fees**

the order

work out FROM, then WHERE, then SELECT – rows first, columns second

SELECT chooses **columns**; WHERE chooses **rows**. Every query question is answered by keeping those two jobs apart.

Building a query in reading order

the order to build it

FROM names the table, **WHERE** keeps only the matching records, **SELECT** then chooses the fields

AND

both conditions must be true

OR

at least one must be true

a worked query

a Book table with Title, Author, Price, InStock

the statement

```
SELECT Title, Price FROM Book WHERE Price < 10 AND InStock = TRUE;
```

a single-table database

all the fields in one table – what the IGCSE paper always uses

Build **FROM**, then **WHERE**, then **SELECT**, even though you write them in the other order. Rows first, columns second.

Exam tips

- A **record** is a row (all the data about one thing); a **field** is a column (one item that every record has).
- A **primary key** must be *unique* for every record, so it picks out exactly one record (StudentID, not FormClass).
- Learn the SQL parts: SELECT (which fields), FROM (which table), WHERE (the condition), ORDER BY (sort, ASC or DESC).
- Read a query in the order **FROM → WHERE → SELECT → ORDER BY** to work out its output.
- COUNT counts the matching records; SUM adds up a number field.



Topic 9 – key takeaways

1 Structure

record = row (item); field = column (attribute)

2 Key

primary key uniquely identifies each record

3 Validation

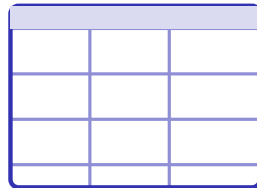
range, type, presence, format at entry

4 SQL

SELECT columns FROM table
WHERE rows

Check yourself

- 1 Is a record a row or a column?
- 2 Why is a name a poor primary key?
- 3 Which SQL clause filters rows?
- 4 Best data type for a phone number?



Answers 1. a row 2. names repeat 3. WHERE 4. TEXT