

redbricks | school

SHIKSHA • SAADHANA • SWARA

SANTEJ, AHMEDABAD

M. S. Access – A Relational Database Management System



Class 8 & 9

Contents

Introduction to Data Type and Organisation	3
Different Data Types	3
Numeric Data	3
Integers	3
Real Numbers.....	3
Currency.....	4
Percentage	4
Alphanumeric (Text) Data	4
Date and Time Data	5
Boolean (Logical) Data	5
Selecting Data Types	6
Data Organisation	6
What is a Record?	7
Field and Field Name.....	8
What is a Key Field / Primary Key?	8
Database Viewed as a Table	9
Types of Database.....	10
Flat-file	10
Relational Databases.....	11
Multiple Tables	12
Linking Tables - Relationships	12

Introduction to Data Type and Organisation

Different Data Types

Before we enter data into a computer system, we usually need to tell the computer what **type of data** it is. This is because the computer stores and processes different types of data in different ways...

Numeric Data

Numeric data simply means **numbers**. However, just to complicate things for you, numbers come in a variety of different **types**...

Integers

An integer is a **whole number** - it has **no decimal or fractional parts**. Integers can be either **positive** or **negative**.

Examples

- 12
- 45
- 1274
- 1000000
- -3
- -5735

123

Real Numbers

Any number that you could place on a number line is a real number. Real numbers include **whole numbers** (integers) and **numbers with decimal/fractional parts**. Real numbers can be **positive** or **negative**.

Examples

- 1
- 1.4534
- 946.5
- -0.0003
- 3.142

3.1

*Some computer software used strange names for real data.
You might see this data type referred to as '**single**', '**double**' or '**float**'.*

Currency

Currency refers to **real** numbers that are **formatted** in a specific way. Usually currency is shown with a **currency symbol** and (usually) **two decimal places**.

Examples

- £12.45
- -£0.01
- €999.00
- \$5500

£99

Percentage

Percentage refers to **fractional real** numbers that are formatted in a specific way - **out of 100**, with a **percent symbol**.

So, the real value **0.5** would be shown as **50%**, the value **0.01** would be shown as **1%** and the number **1.25** would be shown as **125%**

Examples

- 100%
- 25%
- 1200%
- -5%

8%

*Inside the computer, the 50% is stored as a **real** number: 0.5, But when it is displayed, it is shown **formatted** as a percentage*

Alphanumeric (Text) Data

Alphanumeric (often simply called 'text') data refers to data made up of **letters** (alphabet) and **numbers** (numeric). Usually **symbols** (\$%^+@, etc.) and spaces are allowed.

Examples

- DOG
- "A little mouse"
- ABC123
- enquiries@bbc.co.uk

abc

Text data is often input to a computer with **speech marks** (". . .") around it:

"MONKEY"

These tell the computer that this is text **data** and not some special command.

Date and Time Data

Date (and time) data is usually **formatted** in a specific way. The format depends upon the **setup** of the computer, the software in use and the user's **preferences**.

Date Examples

- 25/10/2007
- 12 Mar 2008
- 10-06-08

Time Examples

- 11am
- 15:00
- 3:00pm
- 17:05:45

23/09/78
8:49am

With inputting dates particular care has to be taken if the data contains **American** style dates and the computer is setup to expect **international** style dates (or vice-versa)...

The date 06/09/08 refers to 6th September 2008 in the international system, but would be 9th June 2008 in America!

Check your computer's settings.

Boolean (Logical) Data

Boolean data is sometimes called 'logical' data (or in some software, 'yes/no' data). Boolean data can only have two values: **TRUE** or **FALSE**

Examples

TRUE	ON	YES
FALSE	OFF	NO

TRUE
FALSE

Note that TRUE and FALSE can also be shown as **YES / NO**, **ON / OFF**, or even graphically as **tick boxes** (ticked / unticked)

Selecting Data Types

When we are presented with data to be input into a computer system, we must analyse it and select **appropriate data types** for each value...

e.g. For the following data, we might use the data types shown:

Data Name	Data Type	Example Data
Name	Text	"Bob Gripper"
Height	Real	1.85
Date of Birth	Date	19-May-80
Phone No.	Alphanumeric	012 44565
Pay Rate	Currency	£35.75
Tax Rate	Percentage	15%

Data Organisation

An organised set of data is usually referred to as a database.

Databases can be found at the heart of almost every computer system:

- Databases of users

- Databases of files

- Databases of webpages

- Databases of blog entries

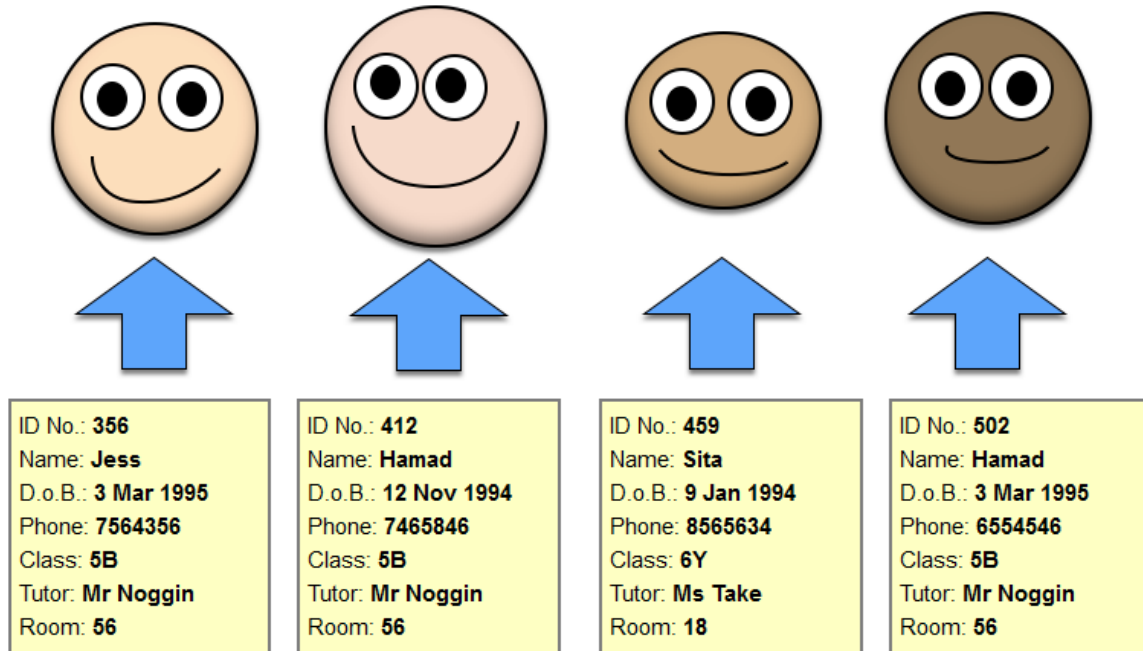
- Databases of photos

- Databases of products

Databases are everywhere!

Databases can be a little difficult to understand, so I will try to illustrate the concept with a few diagrams. We will use some student data as an example.

Here are our students.



You will see that each student has some **data** associated with them (name, d.o.b., etc.) We want to **store** this data in an **organised** way so that we can easily access it in the future. We want to create a **student database**.

So, how should we organise this data?

What is a Record?

The **set of data** associated with a **single object or person** is known as a **record**.

In the example of our students, the data associated with each student is a record.

Here is Jess's record...

Each student has their own record just like Jess's but with different data.

The **data** in each record is **different**, but each **record** has the **same structure**. (each one has a name, d.o.b., phone, etc.)

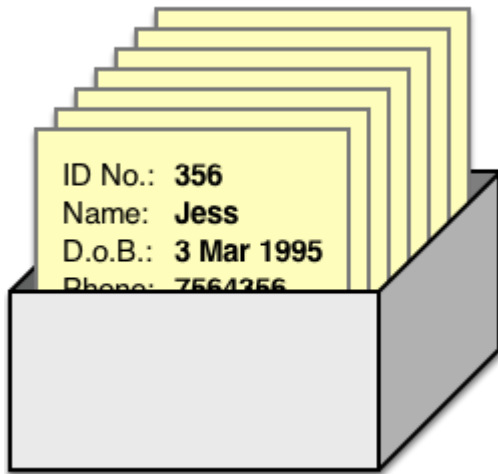
We say that each record contains the same **fields**.

A database is a collection of records.

You can imagine a single record being a card with one the details of one person/object written on it.

A database would be a boxful of these record cards...

<p>ID No.: 356 Name: Jess D.o.B.: 3 Mar 1995 Phone: 7564356 Class: 5B Tutor: Mr Noggin Room: 56</p>
--



This is exactly how a lot of old, manual databases used to look. If you went to a public library 30 years ago, and you wanted to find a specific book, you would have to look through boxes of index cards until you found the details of your book.

Field and Field Name

You will see that each of our student's records contain the same items. These items are known as **fields**.

Each field has a **field name** (e.g. 'Date of Birth')

Each field will contain **different data** in each of the records (e.g. in Jess's record, the Phone field contains 7564356, but in Sita's record the Phone field contains 8565634 - same field, different data values)

It can be a bit confusing - what's the difference between the field, the field name, and the data in the field?!

Imagine that you were manually filling in a record card for Jess. The card would have various labels and boxes to write in...

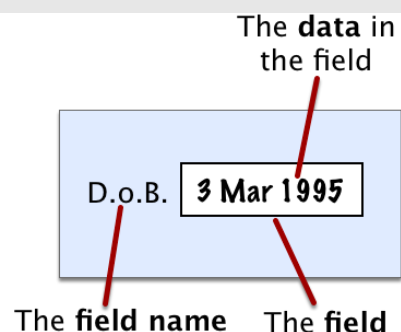
- The **field** is the **box** that you would write in
- The **field name** is the **label** next to the box
- The **data** is what you would **write** in the box

Each of our student records contains seven fields:

- *ID Number*
- *Name*
- *Date of Birth*
- *Phone Number*
- *Class*
- *Tutor*
- *Room*

What is a Key Field / Primary Key?

It is very important that every **record** in a database can be **individually identified**. We need to be sure that



when we access a record, we are accessing the correct one.

Take a look at our students - what item of data identifies them from all of the other students?

- Name? *No - we have two Hamads*
- Date of Birth? *No - Jess and Hamad share the same birthday*
- Phone? *No - two or more students may live at the same address*
- Class / Tutor / Room? *No - each class has many students*

Because all of these fields might contain the same data for more than one record, we cannot use them to identify each record.

Therefore, we have given each student an ID number. We can guarantee that this number will be **unique** for every student.

The ID number is the ideal field to use to **uniquely identify** each individual **record**. We call this field the **Key Field**, or **Primary Key**.

ID No.: 356
Name: **Jess**
D.o.B.: **3 Mar 1995**
Phone: **7564356**
Class: **5B**
Tutor: **Mr Noggin**
Room: **56**

*It is usual for a **code** or **ID number** to be used as the **key field**.*

*In the example, we could have used the student **name** as the key field, but this would be a **bad idea**.*

*Why? Because two or more students might have the **same name**, so the name would not identify each student **uniquely**.*

Database Viewed as a Table

It is quite common to view the contents of a database as a **table** instead of one record at a time. A tabular view is **compact** and allows you to see many records in one go.

Our student database would look like this...

The tabular view of a database is exactly the view that you see when working with your database software (e.g. Microsoft Access).

<i>ID No.</i>	<i>Name</i>	<i>D.o.B.</i>	<i>Phone</i>	<i>Class</i>	<i>Tutor</i>	<i>Room</i>
356	Jess	3 Mar 1995	7564356	5B	Mr Noggin	56
412	Hamad	12 Nov 1994	7465846	5B	Mr Noggin	56
459	Sita	9 Jan 1994	8565634	6Y	Ms Take	18
502	Hamad	3 Mar 1995	6554546	5B	Mr Noggin	56

Each **row** of the table corresponds to a database **record**...

<i>ID No.</i>	<i>Name</i>	<i>D.o.B.</i>	<i>Phone</i>	<i>Class</i>	<i>Tutor</i>	<i>Room</i>
356	Jess	3 Mar 1995	7564356	5B	Mr Noggin	56
412	Hamad	12 Nov 1994	7465846	5B	Mr Noggin	56
459	Sita	9 Jan 1994	8565634	6Y	Ms Take	18
502	Hamad	3 Mar 1995	6554546	5B	Mr Noggin	56

One Record

The **column headings** correspond to the database **field names**...

<i>ID No.</i>	<i>Name</i>	<i>D.o.B.</i>	<i>Phone</i>	<i>Class</i>	<i>Tutor</i>	<i>Room</i>
356	Jess	3 Mar 1995	7564356	5B	Mr Noggin	56
412	Hamad	12 Nov 1994	7465846	5B	Mr Noggin	56
459	Sita	9 Jan 1994	8565634	6Y	Ms Take	18
502	Hamad	3 Mar 1995	6554546	5B	Mr Noggin	56

Field Names

Each **cell** of the table corresponds to a **field**, and contains an item of data...

<i>ID No.</i>	<i>Name</i>	<i>D.o.B.</i>	<i>Phone</i>	<i>Class</i>	<i>Tutor</i>	<i>Room</i>
356	Jess	3 Mar 1995	7564356	5B	Mr Noggin	56
412	Hamad	12 Nov 1994	7465846	5B	Mr Noggin	56
459	Sita	9 Jan 1994	8565634	6Y	Ms Take	18
502	Hamad	3 Mar 1995	6554546	5B	Mr Noggin	56

The D.o.B. Field for Student 412

Types of Database

Flat-file

A '**flat-file**' database is one that only contains a **single table** of data.

All of the data in the database is stored in this one place. The student database example that we looked at in the [previous section](#) was a flat-file database...

*The database work that you have to do for the **practical exam** always uses flat-file databases.*

ID No.	Name	D.o.B.	Phone	Class	Tutor	Room
356	Jess	3 Mar 1995	7564356	5B	Mr Noggin	56
412	Hamad	12 Nov 1994	7465846	5B	Mr Noggin	56
459	Sita	9 Jan 1994	8565634	6Y	Ms Take	18
502	Hamad	3 Mar 1995	6554546	5B	Mr Noggin	56

Relational Databases

A '**relational**' database is one that contains **two or more tables** of data, connected by **links** called **relationships**.

Why would you want to have more than one database table?

Look at the student database example....

ne	Class	Tutor	Room
356	5B	Mr Noggin	56
846	5B	Mr Noggin	56
634	6Y	Ms Take	18
546	5B	Mr Noggin	56

Notice that the table contains several items of data that are **repeated** repeatedly:

- Class (5B)
- Tutor (Mr. noggin)
- Room (56)

In fact, every student in class 5B will have these items of data.

Repeated data in a database is generally considered a **bad** thing:

- It **wastes space** in the database
- It takes **time to input**, typing the same data over and over (and mistakes may be made)
- It is a pain to **update** (if class 5B gets a new tutor, we have to find every 'Mr. Noggin' and change it to the new name)


So how do we avoid repeated data?

*You have to **understand the concept** of relational databases, but you will not be required to use/create them in the **performance task and practical exam!***


Multiple Tables

The solution is to split the data: The repeating data is removed from the main table, and placed in a table of its own...

Student Table

 ID No.	Name	D.o.B.	Phone	Class
356	Jess	3 Mar 1995	7564356	5B
412	Hamad	12 Nov 1994	7465846	5B
459	Sita	9 Jan 1994	8565634	6Y
502	Hamad	3 Mar 1995	6554546	5B

Class Table

 Class	Tutor	Room
5B	Mr Noggin	56
6Y	Ms Take	18

Note: we need to leave the Class field in the main table, as we still need to know which class each student belongs to, but the data relating to each class (Tutor, Room) can be removed.

Therefore, now the main **Student table** just contains data **directly related to students**, whilst the new **Class table** contains data **directly related to classes**.

Note that both tables are independent, and each one has its own **key field / primary key**:

- **Student** table key field is student **ID number**
- **Class** table key field is **class code**

Ok... so we have solved the repeating-data problem, but we seem to have created a new problem: how do we know the name of each student's tutor - it is no longer in the Student table?

Now imagine that class 5B has a new tutor... How much data would you need to update?

*That is correct: only **one** item!*

Remember that, with a flat-file, we had to find every student in class 5B and update the tutor field.

Linking Tables - Relationships

We need to **link** the table together so that we can connect a student to a specific tutor and room.

The **common field** in both tables is the Class field.


We use this field to create a **relationship** (link) between the two tables...

Now imagine that class 5B has a new tutor... How much data would you need to update?


*That is correct: only **one** item!*

Remember that, with a flat-file, we had to find every student in class 5B and update the tutor field.

Student Table

 ID No.	Name	D.o.B.	Phone	Class
356	Jess	3 Mar 1995	7564356	5B
412	Hamad	12 Nov 1994	7465846	5B
459	Sita	9 Jan 1994	8565634	6Y
502	Hamad	3 Mar 1995	6554546	5B

Class Table

 Class	Tutor	Room
5B	Mr Noggin	56
6Y	Ms Take	18

The Class field acts as a **relationship** (link) between the tables


Note that to create the **relationship**; we are using the **key field** (primary key) from one table to link it to another.

When a **key field** from one table appears in a **different** table (e.g. the Class field in the Student table), we call this a **foreign key**.

Database design is a very complex business. It is a career for some people.


For complex databases, it can take a lot of skill to plan what tables and what relationships are required.

Student Table

 ID No.	Name	D.o.B.	Phone	Class
356	Jess	3 Mar 1995	7564356	5B
412	Hamad	12 Nov 1994	7465846	5B
459	Sita	9 Jan 1994	8565634	6Y
502	Hamad	3 Mar 1995	6554546	5B

In the Student table, the Class field is a **Foreign Key**

Class Table

 Class	Tutor	Room
5B	Mr Noggin	56
6Y	Ms Take	18

In the Class table, the Class field is the **Key Field**

- X -