Why Use Databases?

Today a task like that would be done using a database, a type of record-keeping application which offers many advantages:

- makes it easy to enter, retrieve and update records
- makes reports about part or whole of records easy
- removes redundancy from records
- maintains consistency and integrity across records
- security controls access to part or all of the records
- permits concurrent data sharing by many users

Databases are today an indispensable tool for efficiently keeping track of complex information. They let you rapidly search your records for one specific piece of information, or produce lists of things sorted in different ways. They help catalogue, sort, count and label your stock. They help place and receive orders, do audits, and find and keep clients.

How Do They Work?

Data is stored in the form of tables

Each column represents a field. A field can store a specific type of data, i.e. number or character string or dollar value

Each row represents a record. We can add, remove and alter records. Some times, records are also known as tuples.

The field (or combination of fields) that is unique for each record is also known as primary key.

Query

Records can be searched based on their field values. For example, the following lists all records where

Publisher = "Prentice Hall", and

Year > 1996

Records can be modified based on a search criteria. For example, the following adds $10 to all the books published in 1997.

Why Use Databases?

- Manual record keeping can be a time-consuming and tedious job!
- Imagine you're a manager of a bookshop before the invention of computers. You have to keep paper records for
  - every book (the title, author, price, publisher, etc)
  - every publisher (name, address, phone, discount policy, etc)
  - every order (date placed, books, copies, etc)
- And say you keep them in separate folders.

Now you also need to keep them up to date. Meaning lots of visits to your filing cabinet!
Reports

The result of a search can also be used to generate reports.

Reports are customised printouts listing or summarising part or all of the database for a specific purpose.

Report generation is made easy by the use of database tools.

Forms for Databases

Forms can be used to display, enter and modify records. They can hide database details and provide better user interface and avoid invalid inputs.

Views of the Database

Different tables can be joined together based on some common field values. Such tables could be on the same computer or any where on a network.

User can search the new joined database just like any other database.

Some Applications of Databases

**Personal Use** - keeping track of your CDs, phone numbers

**Charities** - donations, benefactors and recipients addresses

**Small and Large Businesses** - stock control, payroll, tax law compliance, employees, product or service information, etc.

**Libraries** - catalogs, orders, lendings

**Scientific Organisations** - maintenance of scientific data, control of equipment, chemicals or other supplies

**Educational Institutions** - allocation and scheduling of resource materials, records of student marks

**Law Enforcement** - criminal records, forensic data, fraud detection

**Governments** - census data, births and deaths, economic data, citizenship records, taxation records, fraud detection...

Abuses of Databases

- When an attempt to introduce the “Australia Card” - an identification document giving a unique number to each citizen - was made in 1987, protesters in every state accused the government of being “Big Brother”.

- People didn’t want any government to establish a central computer keeping all the facts about individuals in one place. This would give it too much power over people - power that was bound to be abused.

- The Australia Card idea was abandoned as undemocratic.

- But the government didn’t need to put all the data into one database to have such a powerful tool of control. All it needed was the single identifying number to appear on many, much smaller and more acceptable databases held by the tax office, welfare organisations and law enforcement agencies. It could link these together at any time to make a complete dossier on one person or a list of persons.

- This is probably why you always need to quote your Tax File Number - even for things that have nothing to do with taxation.
Let’s now discuss some of these Database topics in more detail …

Data vs. Information

- **Data**
  - raw material: facts and figures
  - Cell entries in database tables

- **Information**
  - organised data
    - records, tables, databases, reports
  - information is used to inform

Information Technologists (and others) argue about the meaning of the terms
- *data*, *information* and *knowledge*

Google these terms to see how much is written about them

When to use a Database

- Databases are ideal for maintaining data related to objects (e.g., student records, inventories, etc.)
- This data can be queried, sorted, etc. and the relationship between the objects and their attributes is maintained.
- While they can be used to perform numeric calculations, they are not usually optimised for this task.

Databases

- One important feature of a database is that the stored information is organised and structured in a particular way
- Usually databases try to allow for fast retrieval of information (via queries) even when they contain vast amounts of data
- It is important to pay careful attention to the design so as to structure the storage of data appropriately

Database Management System

- Database Management System (DBMS) is a computer program that stores related information in a structured way so as to maintain relationships between information and allow access to and modification of data
- Most common model is Relational DBMS

Structured Data

- Structure of information in DBMS is maintained by a relation(s)
- A relation associates the data belonging to a single object or entity
- For example:
  - Students: Name, address, ID number, courses enrolled, etc.
  - Stock: Part #, description, quantity, unit price
- We shall see later that relations are stored in tables
Example Databases

- Catalogues (e.g., library, museums)
- Student records
- Inventories (e.g., parts, stock)
- Address books
- Events (e.g., swim meeting)

Example Databases

- Inventory

<table>
<thead>
<tr>
<th>Part No.</th>
<th>Description</th>
<th>Quantity</th>
<th>Unit Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>P001</td>
<td>Widget</td>
<td>10</td>
<td>$1.50</td>
</tr>
<tr>
<td>P002</td>
<td>Gadget</td>
<td>12</td>
<td>$10.20</td>
</tr>
<tr>
<td>P201</td>
<td>Object</td>
<td>25</td>
<td>$0.12</td>
</tr>
</tbody>
</table>

Example Databases

- Student database

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Faculty</th>
<th>Degree</th>
<th>Credit Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234567</td>
<td>Jane Doe</td>
<td>Science</td>
<td>MSc</td>
<td>50</td>
</tr>
<tr>
<td>7654321</td>
<td>John Smith</td>
<td>Arts</td>
<td>BA</td>
<td>48</td>
</tr>
</tbody>
</table>

Example Databases

Differences Between Spreadsheets and Databases

- In a spreadsheet, **changes** to elements may affect other elements however the elements are not related in any other way.
- A database groups the **attributes of an object together** and treats them as “belonging” to that object.
- In a database, elements have a “type” (e.g., text, number, date, yes/no, etc.)

When Should You Use a Database Rather than a Spreadsheet?

- Database stores information in a structured way as relations: attribute values of a single object
- Different attributes have a data type
- Spreadsheet doesn’t maintain information in this way (only formula dependencies) and has only a few data types
- Use DBMS when it is important to maintain attributes of objects as single entities, and **relationships** between objects
- **Spreadsheet** is normally useful for **calculations and visualisation** of information

Problems that Databases Try to Deal With

- **Redundancy** and **inconsistency** of data
  - Different copies of information may exist (redundancy) and some attributes may differ (inconsistency)
- **Access**
  - May need to **allow/restrict access** to different portions of data
  - Multiple users accessing/modifying data at the same time
- **Integrity**
  - Constraints on data that can be stored (e.g., only Gold status members entitled to a discount)
MS Access Database

- MS Access is a Relational DBMS
- It allows you to:
  - Maintain tables
  - Query the database
  - Produce forms for data entry and viewing information
  - Produce reports for viewing information

Tables

- Collection of objects or entities
  - In database terminology, collection of “records”
  - Each record has a number of attributes or “fields” describing entity
- An MS Access database may have a number of tables

Tables

- Collection of objects or entities
  - In database terminology, collection of “records”
  - Each record has a number of attributes or “fields” describing entity
- An MS Access database may have a number of tables

Tables

- Collection of objects or entities
  - In database terminology, collection of “records”
  - Each record has a number of attributes or “fields” describing entity
- An MS Access database may have a number of tables

Records

- Information about a single object
- Example:

  | P002 | Gadget | 12  | $10.20 |

- A row of database table
- Each attribute is known as a field of the record or table

Fields

- Attribute of an object or entity
- Examples: part number, quantity, name, student number, location
- A column of database table
- Every field has a data type (e.g., numeric, text, hyperlink, etc.)
- Information that does not conform to the data type cannot be entered into the database
  - One way for DBMS to maintain integrity of data

Field Data Types

- Autonumber - MS Access adds value
- Numeric
- Yes/No - Boolean: true/false; on/off values
- Text
- Memo
- Date/time
- Currency
- OLE object - pictures, etc.
- Hyperlink - URL

Why Enforce Data Types?

- Prevent data entry errors
- Reduce amount of storage space required to store database
Creating Tables

- Design view
  - Name fields and assign data type
- Wizard
  - Select from some standard field names
- Enter data
  - Enter data directly into tables
  - Can modify in Design View later
Key Fields

- One or more fields of a database table
- **Uniquely** distinguish each record in a database table
  - No two rows of the database can have the same value for the key field(s)
- When key consists of more than one field it is referred to as a **compound key**
- Examples: student number, part number, ISBN, Dewey decimal number
- If key field is not a common attribute of an object, may need to “create” one (e.g., part number)

Operations on Databases

- **Queries**
  - Select records from table(s) satisfying certain criteria
  - MS Access allows a number of different types of queries
- **Sorting and Filtering** table data
- Construct forms for data entry and display
- Develop reports to present data

Queries

- Once you have a large amount of data stored in a database, you need a quick and easy way to access that information
- Queries allow you to access data that conforms to certain criteria
- Queries also allow you to update and modify data in tables
- Let's initially consider queries on single database tables …

Creating Queries

- MS Access provides a number of ways of designing queries
  - Query wizard
  - Design View
  - SQL - Structured Query Language

Types of Queries

- There are 4 different types of queries in MS Access
  - **Select** query
    - Most basic form of query, returning records that satisfy specified criteria
  - **Action** query
    - Update query
    - Append query
    - Delete query
    - Make-table query
  - **Total** query - performs calculation
  - **Crosstab** query
**Query Wizard**

- As with most wizards, the Query Wizard guides you through the design of a basic query.
- Can set:
  - Tables used
  - Fields
- Can refine design in Design View.

**Design View**

- A more flexible way of designing queries is in Design View.
- The following steps will guide you in designing a basic Select Query:
  - Select tables containing fields that you would like to view or use in specifying criteria.
  - Select fields in the order you wish to view them.
  - Set any sort criteria.
  - Select/De-select fields that will be displayed by the query.
  - Set criteria.

**Datasheet View**

- For tables, Datasheet view can be used to add, edit and delete records in the tables.
- Changes made to the table in Datasheet view cannot be reversed.
- For queries, Datasheet view can be used to view the results of performing a query.
- Datasheet displays the dynaset.
- Again, changes made in Datasheet view will be reflected in the original table.

**Running a Query**

- Once designed a query can be performed by:
  - Clicking on the Run icon (!) in the menu bar or from the Query menu.
  - Selecting Datasheet view.
- Results of query will be displayed in Datasheet just as you would view the contents of a database table.
- Results of query are temporarily stored in a dynaset.

**Select Query**

- The simplest Select Query simply lists certain chosen fields from among those in the table.
- No criteria are specified so all records are displayed but only the selected fields are shown for each record.
- The steps involved:
  - Select tables.
  - Select fields to display.
  - Impose any sorting order.

**Select Query**

- The figure illustrates the design of a Select Query.
- The tables involved in the query are shown.
- The sort order is specified.
- Criteria rows are used to define the conditions for displaying data.
### Sorting Query Results

- Results of a query can be **sorted** on certain fields by specifying a sort order
  - Ascending or Descending
- Sort order is applied on fields from **left-to-right**
  - I.e., Results are sorted on left-most field with a sort order specified first, then next field to the right with a sort order specified and so on
- How can we sort in a different order?

### Specifying Selection Criteria

- Can reduce the number of records displayed by specifying criteria
- Each record whose fields match the specified criteria will be displayed (but note that only the selected fields are shown)
- Several criteria rows can be specified
  - In any given criteria row, a record must match **every** field with a criterion
  - If there are multiple criteria rows, the record must match **at least one** row to be displayed

### Numeric Criteria

- For numeric fields:
  - Entering a single number in the criteria row of a field means that only records containing that value in the field will match
  - Can prefix number with logical criteria
    - <, <=, >, >=, <>

### Text Criteria

- Can specify text in double quotes:
  - "search text"
  - Double quotes only necessary if search text contains space
- Again, only records that have the text in that field will match
- Text is not case sensitive
- Can use wildcards for matching more sophisticated patterns
Wildcards

- Wildcards can be used to specify patterns in a text query criterion
  - * - matches 0 or more characters
  - ? - matches any single character
  - [ ] - match any single character specified inside the brackets
  - [! ] - match any single character except those specified inside the brackets
  - - match a range of characters (e.g., a - z)
  - # - matches any single numeric character
- To search for literal wildcard characters place them in brackets (e.g., \[?\] matches the character '?')

AND vs. OR

- In a given criteria row all fields must match criteria (AND)
- If multiple criteria are specified, record can match any of them (OR)
- Can also use logical operators to specify criteria in a single field
  - E.g., >=110 And <=130
    - Value between 110 and 130
  - E.g., “A” Or “B”
    - Strings starting with an ‘A’ or ‘B’

BETWEEN, IN, LIKE

- Between - specifying ranges
  - Between 110 And 130
    - Same as >=110 And <= 130
- In - lists of values
  - ln[two, four]  
    - Same as “two” Or “four”
- Like - searching patterns
  - Like “%”
    - Strings that end in letter ‘z’

Dates and Times

- Can specify dates and times by surrounding them with # signs
  - E.g., #26/4/05#, #April 26, 2005#, #26-Apr-2005#, #1:15PM#, #13:15PM#
- MS Access also provides the following functions
  - Day, Month, Year, Weekday, Hour, DatePart, Date
- Criterion to select fields where no value has been entered
  - Use key word Null in Criteria row
    - Take care not to quote the string “Null” otherwise you will be looking for the text string composed of those four letters

Parameter Query

- The queries that we have designed so far are static. They do not request input from the user.
- Can do so by using the criterion:
  - Like [Prompt text]

Action Queries

- Action queries allow you to modify and update values in database table
  - Update query
  - Append query
  - Delete query
  - Make-table query
- The effects of action queries cannot be undone
Update Query

- Update Query can modify the values in database records
- Specify an expression in the Update To: row that determines the new value of a field
- **NOTE:** Once modified, a field cannot be changed back to its previous values

Append Query

- Append Query will add records from one table to the end of another table
- Must specify table to which records will be added
- Need to specify which fields of source table will be added to target table and also to which fields they will be added (in case the field names differ)

Delete Query

- Delete Query removes records from a table
- Can use to remove outdated data or data that is no longer required
- **NOTE:** Cannot undo the deletion of records

Make-table Query

- When you perform a Select Query, Total Query, etc., results are temporarily stored in a dynaset
- They are not stored in a table in the MS Access database
- A Make-table Query will store the results of a query in a new table in the database

Total Query

- Total Query performs calculations on groups of records
- Click on the sigma icon \( \Sigma \) in Design View to create a Total Query
- Total: row appears
- Can use this to group records according to the same values using Group By
- Can apply functions to perform calculations
  - Sum, Avg, Min, Max, Count, StDev, Var, First, Last, Expression, Where
Crosstab Query

- A Crosstab Query allows you to summarise data in row and column format
- Must specify one field whose values will be the columns and can have several fields whose values will form rows
- Can apply a function to the values of a third field
  - Sum, Avg, Min, Max, Count, StDev, Var, First, Last, Expression, Where

SQL

- Structure Query Language
- Standardised language for querying and manipulating databases
- Used in virtually all commercial DBMSs
- Can view SQL for a query that you have designed by selecting SQL View from View menu
- SQL’s query structure has the following syntax
  
  ```
  SELECT List of fields
  FROM Table(s)
  WHERE Constraints on the rows
  ```

Example: SQL

```
SELECT Enrolments.StuID, Student.FirstName, Student.LastName, Courses.CourseName, Enrolments.Grade
FROM Student
INNER JOIN (Courses
INNER JOIN Enrolments
ON Courses.CourseID = Enrolments.CourseID) ON Student.StuID = Enrolments.StuID
WHERE ((Enrolments.Grade)="FL") AND ((Enrolments.CourseID)="MATH1090");
```

Multiple Tables

- What is wrong with the following table?

<table>
<thead>
<tr>
<th>Order ID</th>
<th>Description</th>
<th>Quantity</th>
<th>Unit Price</th>
<th>Family Name</th>
<th>Given Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>C001</td>
<td>Widget</td>
<td>5</td>
<td>$1.50</td>
<td>Doe</td>
<td>Jane</td>
<td>412-Anzac Pde, KINGSFORD 2032</td>
</tr>
<tr>
<td>C002</td>
<td>Widget</td>
<td>12</td>
<td>$1.50</td>
<td>Smith</td>
<td>John</td>
<td>3 Main St, YOUNG 2594</td>
</tr>
<tr>
<td>C003</td>
<td>Gadget</td>
<td>7</td>
<td>$2.05</td>
<td>Doe</td>
<td>Jane</td>
<td>412-Anzac Pde, KINGSFORD 2032</td>
</tr>
<tr>
<td>C004</td>
<td>Widget</td>
<td>14</td>
<td>$1.50</td>
<td>Doe</td>
<td>Jane</td>
<td>412-Anzac Pde, KINGSFORD 2032</td>
</tr>
<tr>
<td>C005</td>
<td>Gadget</td>
<td>4</td>
<td>$2.05</td>
<td>Smith</td>
<td>John</td>
<td>3 Main St, YOUNG 2594</td>
</tr>
<tr>
<td>C006</td>
<td>Gadget</td>
<td>23</td>
<td>$2.05</td>
<td>Smith</td>
<td>John</td>
<td>3 Main St, YOUNG 2594</td>
</tr>
<tr>
<td>C007</td>
<td>Widget</td>
<td>3</td>
<td>$1.50</td>
<td>Doe</td>
<td>Jane</td>
<td>412-Anzac Pde, KINGSFORD 2032</td>
</tr>
</tbody>
</table>
### Multiple Tables

**Parts**

<table>
<thead>
<tr>
<th>Part ID</th>
<th>Description</th>
<th>Unit Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>P01</td>
<td>Widget</td>
<td>$1.50</td>
</tr>
<tr>
<td>P02</td>
<td>Gadget</td>
<td>$2.05</td>
</tr>
</tbody>
</table>

**Customers**

<table>
<thead>
<tr>
<th>Customer ID</th>
<th>Family Name</th>
<th>Given Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>C01A</td>
<td>Doe</td>
<td>Jane</td>
<td>412 Anzac Pde, KINGSFORD 2032</td>
</tr>
<tr>
<td>C02A</td>
<td>Smith</td>
<td>John</td>
<td>3 Main St, YOUNG 2594</td>
</tr>
</tbody>
</table>

**Orders**

<table>
<thead>
<tr>
<th>Order ID</th>
<th>Customer ID</th>
<th>Part ID</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>O001</td>
<td>C01A</td>
<td>P01</td>
<td>5</td>
</tr>
<tr>
<td>O002</td>
<td>C02A</td>
<td>P01</td>
<td>12</td>
</tr>
<tr>
<td>O003</td>
<td>C01A</td>
<td>P02</td>
<td>7</td>
</tr>
<tr>
<td>O004</td>
<td>C01A</td>
<td>P01</td>
<td>14</td>
</tr>
<tr>
<td>O005</td>
<td>C02A</td>
<td>P02</td>
<td>4</td>
</tr>
<tr>
<td>O006</td>
<td>C02A</td>
<td>P02</td>
<td>23</td>
</tr>
<tr>
<td>O007</td>
<td>C01A</td>
<td>P01</td>
<td>3</td>
</tr>
</tbody>
</table>

**Normalisation**

- The problems on the previous slide can be reduced through the process of normalisation.
- Normalisation is a complex process and we will not go into the theory here. The essence of the process can be described through several simple rules.
Normalisation

- Every field should be **atomic** and represent **unique** type of data
- Every table must have a **primary key** consisting of one or more fields. The primary key must be unique for each record. (Use “natural” keys where possible.)
- For every value of the primary key, field values must be relevant and completely describe objects.
- Changes to data in a field other than the primary key should not affect data in other fields.
- Data generally shown in de-normalised (flattened) form in reports

Primary Keys and Foreign Keys

- Notice that **Customer ID** field occurs in both Customers table and Orders table.
- In Customers table it **uniquely identifies** each record
  - **Customer ID** is the **primary key** of the Customers table
- In the Orders table **Customer ID** field does **not** uniquely identify each record but serves to “link” to the Customers table
  - **Customer ID** is a **foreign key** in the Orders table

Relationships

- Each customer can have many orders but an order can only have one customer
  - This is an example of a **one-to-many** relationship
- One customer can have many orders
- There are situations in which you might have **one-to-one** relationships
  - For every record in the first table there is exactly one related record in the second and vice versa
  - This can help to separate data that is not as important or that needs to be restricted
- If relationship is **many-to-many** you will need to introduce another database table

Creating Relationships

- Once the tables have been created you can create relationships between them
- Note that fields in different tables **do not have to have the same field name** in order to be related. They must however have values in common
- To create relationships, select Relationships from menu or Tools Relationships...
- In dialog box that appears select tables; drag a field name in one table to the field you want to link it to in the other table
- These relationships are created for the table as a whole. Can also create them for queries only

Referential Integrity

- Tables in database must have **consistent** values
- Example 1: Creating an order for a non-existent part (obvious)
- Example 2: Deleting a part when an order for that part still exists (not always obvious)
- This is known as referential integrity
- MS Access will check **referential integrity** if you check the appropriate item when creating a relationship between tables

Queries on Multiple Tables

- Queries on **multiple tables** are similar to those for single tables
- In Query Design View first add all the required tables
  - Those containing fields which will be displayed or have criteria specified
  - May also need to add any tables that “link” these tables together
- Select fields that need to be displayed or have criteria specified as you would with a single table
  - Where a field occurs in two tables (usually because it links them together) you can select this field from either table
Joins

- In query using multiple tables, so far records from table are only included where the fields that link them have the same value (e.g., record in Orders table with same Customer ID as record in Customers table)
  
  - This is known as an *equi-join*

- There are other ways of "joining" tables

- For example, you may want to see all Customers that have not placed an order

- Can use outer join to display all records in one table with matching records in another if they exist

- Can do this by clicking on link between tables in Query Design View or using Relationships dialog

Security

- A weak form of **security** is provided by Data Types, Validation rules, Locking fields, etc. This helps to ensure that incorrect data is not entered.

- Stronger forms of security are available

  - Can set database password

  - Can also set user and group level access

Validation Rules

- Some fields require that data is of a particular form that cannot be checked entirely by conforming to the field data type

- For example, Customer ID in Customers table has been restricted to letter 'C' followed by two digits followed by a character

- **Validation Rule** can be set in Table Design View for this field to restrict the data that can be entered into this field. Can also be set in Forms using field properties.

- **For example:** Like “C##?”
  
  - (see Wildcard characters for more)

- **Validation Text** is text that will appear if incorrect data is entered into field

Locking Fields

- Fields in Forms can be **locked** to stop the field from being altered

- Set **Locked property** of field in Form to Yes

- This is a very weak form of security as it can be easily circumvented

Filters and Sorting

- In similar fashion to MS Excel, can filter and sort **records** when in Datasheet View (Table and Queries)

- Two methods for filtering
  
  - Filter by Selection
  
  - Filter by Form

- Can sort fields by Sort Ascending and Sort Descending

Summary – DBMS

- When we have large amounts of data to store and manipulate can use DBMS

- A DBMS stores data in a structured way

- One of the most common database models is the Relational DBMS

- Spreadsheet does not maintain relationship between an entity’s attributes in the way a database does

- DBMS tries to deal with redundancy, consistency, integrity of and access to data
Summary – Database structures

- Database consists of tables storing information about objects and entities
- Each table consists of a number of records (rows)
- Each record has a number of fields storing the values of attributes for an object (columns)
- Each field has a data type to which the value must conform
- A key field(s) uniquely identifies each record

Summary – Queries

- Queries can be used to access, modify and update data stored in database tables
- Records satisfying query criteria are displayed in dynaset - a dynamic subset of a table
- Select query displays those records satisfying criteria
- Crosstab query displays results of query in row/column format
- Parameter query obtains input from user in order to perform query
- Action queries modify and update records: Update, Append, Delete and Make-table queries
- Total query performs calculation

Summary – MS Access

- Multiple views provided for flexible management of data
- Design view can be used to create queries while Datasheet view displays results
- Criteria can be specified to select records of interest, including Boolean and wildcard operators
- SQL - standardised language for querying and manipulating databases
- Importing and exporting data
- MS Access allows for various levels of security in a database
- Filters and sorting can also be applied to database in similar way to MS Excel

Summary – Relationships

- Multiple tables reduce data redundancy in database
- The process of normalisation can help structure database into multiple tables
- Once multiple tables are constructed need to link them using relationships
- Queries using multiple tables require that tables be linked
- Relationships can be one-to-many and one-to-one
- Referential integrity can be checked by MS Access