Databases - Introduction
What is data?

- A **datum** is a single fact about some *universe of discourse*.
  
  *Jack Johnson has student number 20723081*

- *Data* is the plural of datum, hence refers to a collection of facts.
  
  By itself data has little or no meaning.

- *Information* is the *interpretation* of data — attributing *meaning* to the data.

  As one writer has said:

  *Information is what you want; data is what you get.*
Databases

A *database* is an organized collection of data, usually representing a model of the activity of some business or other organization.

- Students, Courses, Units and Grades
- Customers, Products, Orders and Deliveries
- Doctors, Patients, Prescriptions, Medications
- Students, Books, Periodicals, Loans

The fundamental role of *database technology* is to allow users, typically organizational users, to extract information from their data.
Database Management Systems

A database management system (DBMS) is any system that allows users to manage their data, although nowadays the term is used almost exclusively for *software* rather than manual systems.

The amount of data that can be collected automatically is growing exponentially, and so there is a strong demand for database professionals, particularly database administrators.

Many different types of DBMS, but *relational database systems* are by far the most important in practice.
Why use a DBMS

What are the benefits of using specialized DBMS tools rather than simply having data stored as a collection of files on a file system?

- Data Independence
- Efficiency
- Data Integrity
- Data Administration
- Concurrency Control
- Application Development
Data Independence

Data independence provides analogous benefits to the *encapsulation* found in object-oriented programming languages:

- Applications use a *logical model* of the underlying data, rather than directly manipulating the physical files storing the data.
- Implementation of physical storage can be altered or improved without affecting client code.
- Physical storage can be remote, or distributed, or both with no alteration in client code.
A DBMS can implement a number of storage strategies and optimizations to make the most common operations as fast as possible.

In particular, the DBMS can maintain various *indexes* to the data to make querying the database quick; the user can control which indexes are present, but need not know how they are implemented.

Database storage and indexing strategies are extremely sophisticated applications of data structures techniques.
Data Integrity

A critical role of a DBMS is to ensure that the entire collection of data is maintained in a *consistent* state.

At its simplest, this means that data should be stored in a single place — if several parts of an organization use personal data about an employee, and each stores that data separately, then it is difficult to enforce integrity.

More significantly, a change in one data item often has a “ripple effect” of consequences for data in other areas: a DBMS can ensure that all of these consequences occur and disallow an operation that would leave the database corrupt.

This is often described as the DBMS enforcing *integrity constraints*. 
Data Administration

A DBMS allows the organization a fine degree of control over who is permitted various levels of access to the database.

In most operating systems (Unix, Windows etc) users can either read an entire file or none of it, but a DBMS can present different views of the same data to different groups of users.

For example, a lecturer may be able to look up a student’s academic record, but not their personal or financial details, while only certain staff will be able to alter their academic record.
Concurrency Control

In a large organization, there will often be several people accessing the same data item at the same time.

While this is not a problem if all users are simply viewing the data, it becomes a major problem if some of the users need to update the data.

For example, an airline reservation system may have several travel agents viewing availability at the same time, but the DBMS must prevent two agents from booking the same seat at the same time.
Application Development

Analysing data may require more sophisticated and application-dependent programs than a general-purpose DBMS can provide.

This can be accomplished by having a general purpose programming language such as Java and C accessing the data through the DBMS and then performing additional processing with the results.

This combination permits the developer to focus on the “business logic” of the application.

The power and success of this form of application development can be seen by the fact that essentially every large dynamic website is a database-backed application.
A large subject

Each of the topics listed above has enough theory, practice and technology associated with it to form an entire unit that could legitimately be called *Databases*.

Ramakrishnan & Gehrke identify two major approaches:

- **Systems Emphasis**
  
  *Building* database systems — the nuts and bolts of storage, indexing, query optimization, transaction management.

- **Applications Emphasis**
  
  *Using* database systems — data modelling, data query languages, database-backed applications.
This unit

We will take the *applications emphasis* covering

- The relational data model
- Relational algebra and calculus
- SQL, mySQL and Oracle
- Database-backed applications with Java (or Python)