Lecture 1: Intro to Database

Database and SQL: Course Notes | Brendan Shea, Ph.D. (Brendan.Shea@rctc.edu)

1 Contents

2 What are Databases? Why Do We Use Them?

In this lesson, we'll be answering the following questions:

1. What is the difference between **data** and **information** (or **knowledge)?**
2. What are databases, and how can they help us turn data into information?
3. What advantages do databases have over **flat files,** such as paper records or spreadsheets?

2.1 Class Overview

In this class, we'll study the basics of **databases:** what they are, how they are used, and what you can do with them. In particular, you'll be learning about the following:

1. General database design and implementation issues,

2. How information can be entered and retrieved from databases
3. How to use **Structured Query Language (SQL),** the most common way of *interacting* with relational databases.
4. How to query a database using either SQL or other interfaces.
5. How to analyze and use the results of these queries.

Figure 1A database wizard (Brendan Shea x Dall-E)

A note: Databases should be fun! Databases often strike people (even some programmers) as "boring." I hope this class will convince you this isn't true! Database design can often be the most creative part of a programming project, and the **declarative** language used to access databases (**Structured Query Language, or SQL**) offers a significantly different way of thinking about things than the **procedural** languages (like Python, C, or Java) that you may be used to. Designing and implementing a good database often requires answering hard (and interesting!) questions about how different parts of the world "fit together" and how we can use computer systems to help us make sense of it.

2.2 Data, Information, and Knowledge

In some general sense, a database is simply a collection of data, whether this is a pile of notecards, a shopping list typed into a word processor, or a spreadsheet tracking your recent purchases. However, a database in the context of computer science more often refers to a particular data storage method. **Databases** store **end-user data** (the "content" stored and retrieved by database users) which is organized using **metadata** (data which describes the format/meaning of end-user data). This is a very different from other ways of storing data! So, for example:

1. An MS-word document (such as this one) might contain a string of characters like "$15,235". However, a well-designed database could use its metadata to determine that this is meant to be the amount of money in a particular person's bank account.
2. A spreadsheet might contain numbers like 35, 54, 56, etc. A well-designed database would "know" that these represent the hours per week worked by different employees (and it would know how these numbers are *related* to employees' total compensation).

A **database management system (DBMS)** is a program (or collection of programs) that allows people to *interact* with databases.

Why Databases Matter. Databases help humans convert **data** (raw, unstructured facts about the world, many of which are useless to us!) into **information** (where we understand the *meaning* of specific data for our interests). They then help turn information into **knowledge**--the "collected" and "organized" information we have about a particular subject. Knowledge allows us to do things such as *explain* and *predict*. This, in turn, helps us make better decisions about our lives. For example, suppose we are running a bookstore:

- *Data* consists of the ISBNs of various books, the price for each book, customer credit card numbers, and the like.
- *Information* results from processing data in ways that make it meaningful to the user. For example, we might have information on "the total sales over the last month" or "number of books customer X purchased by author Y."

- *Knowledge* is a comprehensive collection of information relevant to our purposes. For example, as booksellers, we might use our information to answer questions like "How do customers' reading tastes change from winter to summer?"

Well-structured databases can help us receive better medical care, make education work better, help us find products to purchase and advance scientific research. In the proper context, databases can make querying (asking questions) data much more efficient and accurate. The flip side is that NOT understanding how databases work can lead people to make bad, unproductive decisions. They might waste time searching through unorganized data (and potentially missing correct answers), make unnecessary and harmful mistakes, or even compromise the security of their data.

2.3 The Problem With "Flat Files" and the Advantages of Databases

To fully understand the "why" of databases, it will be helpful to consider the main alternative: the so-called **flat-file** or **"file system"** approach. This approach collects **records** (end-user data particular object or entity) into **files.** For example, in spreadsheets, each row represents one entity, with the columns representing attributes of that entity. Unlike databases, there is no "metadata" of the sort that would allow to determine relationships *between* different files and records. The choices of which records and which attributes to store in a particular file is determined entirely the file's intended *use* (by contrast, databases lend themselves well to *multiple* uses). Examples of this approach would be.

1. Writing down (or typing) regularly-formatted data on paper and storing papers together (using things like file folders and fine cabinets). A small business, for example, might have a "file" for employees, a "file" for customers, a "file" for tax information, etc.
2. Doing the same thing electronically by storing information in comma-separated value or Excel spreadsheet files.

Databases vs. File Systems. Many people (probably too many people!) use file systems to organize their information. However, databases have significant advantages over file systems, at least in many contexts:

1. **Databases present users with a single "logical file."** While the data in databases might be spread over many locations (and different drives, computer files, etc.), the DBMS that users interact with shows them the results of data requests a single logical file (which appears as a "table."). Many of the advantages below stem from this.
2. **Databases are "self-describing."** Databases allow us to easily encode metadata stored separately from the data itself. So, for example, a music database will "know" that "Taylor Swift" refers to an artist and that artists are *related* in specific ways to songs and albums. In file systems, by contrast, there is no way to formally "encode" this sort of information.
3. **Databases minimize data redundancy and ensure consistency. They enforce transactions.** In a well-structured database, we should store a given piece of data (such as a person's current bank balance) exactly ONCE. Avoiding **data redundancy** (unnecessary duplication of data) helps prevent **data anomalies** (which occur when data isn't internally consistent). For example, in a bank database, we would like to store a customer's balance in a single place, which makes updating this balance much easier. Many Databases also support **transactions**, which ensure our data remains "consistent" even when it is being simultaneously accessed and changed by many users.
4. **Databases allow for (much) quicker answers to ad hoc queries.** A **query** is a "request" for data (usually expressed in a formal language such as SQL), while an **ad hoc query** is one that we ask, "on the fly" (e.g., we'd like an answer quickly and don't want to refer it to a librarian or other data expert). In a flat file, it can take a LONG time to find the data we are looking for, and even longer to "put it together" with other needed data. Databases make it much easier to ask questions (we can use SQL!) and much faster to get our answers.
5. **Data independence.** A database allows us to separate (1) how users (such as programmers, business analysts, researchers, medical staff, or even visitors to a website) *use* the data from (2) the way the

data is stored. This is important because it means we can *change* how the data is stored without causing problems for users. Flat files do not accommodate this, and even small changes to spreadsheets can cause many problems for programmers and other users.

6. **Data can be shared between users with different views.** Many people need to access the same data in an organization such as a hospital. However, they require very different things from data, depending on their role (e.g., physicians' use of data differs from that of I.T. staff, which in turn differs from that of patients). Databases allow not only for shared access to data but for the design of multiple views (what the user "sees" and "interacts with" the data) for different users.
7. **Data security and backup/recovery.** Databases make it much easier to control *which* users can access/change *which* data. We can also *log* changes as they happen, making it possible to recover from crashes or other problems.

The main *drawback* of databases concerns the cost (in terms of time, expertise, and money) to get them set up and running. So, they might not make sense for all projects, particularly small ones. However (as I hope I'll convince you), it isn't all *that* difficult to set up a simple database and take advantage of these benefits.

2.4 WHY GOOD DESIGN MATTERS

To take advantage of the power of databases, you need to have a good **database design.** Database design will be a major subject of this course, and it is somewhere between a "science" (requiring knowledge about databases and the subject matter) and an "art" (requiring creativity and the ability to see "connections" between things that others may not have noticed).

To see how database design matters, let's consider an example from the fictional "Hogwarts school" attended by aspiring witches and wizards in the fictional *Harry Potter* universe. Dumbledore, an excellent wizard but subpar database designer, decides to encode data on students, instructors, and the courses they are taking or teaching as follows:

Name	Course 1	Course 2	Course 3	Notes
Ginny Weasley	Potions (3 cr)	History of Magic (4cr)	None	
Harry James Potter	Transmutation (4)	Potions (3 cr)	Defense Against Dark Arts	
Severus P Snape	Potions (three credits)			Teacher
Minerva McGonagall	Transfiguration (3)			Teacher
Hermione Granger	Hist of Magic (4cr)	Potions (3 cr)	Transfiguration (3)	T.A. for Potions

In separate files, he keeps track of other student and faculty info (such as contact info for students' guardians or faculty salaries). There are several things wrong with this table, including the following:

- **The data can't easily be searched or sorted in meaningful ways.** The fact that "name" contains both first and last name (and sometimes middle name) makes it challenging to sort by fields such as people's last names. It's also tough to search for things like "which classes are worth three credits?" or "how many classes is Professor Snape teaching right now?"
- **The database isn't easy to expand.** There are only three columns (or **fields)** for courses! What happens if someone wants to take four classes that semester? Adding a fourth column for "class 4" is more painful than it might seem since it will require that everyone who *interacts* with the database adjust to this.
- **There are data redundancies and (because of this) an increased chance for inconsistencies and errors.** The fact that class names and credit information is *repeated* can cause problems. In

particular, users (such as Dumbledore) may (unintentionally) enter the "same" data in different ways (for example, "History of Magic" vs. "Hist of Magic"), which can lead to lousy query results down the road. There's also an increased chance of error if some copies aren't updated correctly. For example, the Potions class is three credits every place but one.

A better solution might be to break down the data into (at least) three separate tables: **Person, Course,** and **Enrolled.** I've shown parts of these tables below.

SELECT * FROM Person

Id	First	Last
1	Ginny	Weasley
2	Harry	Potter
3	Severus	Snape
4	Minerva	McGonagall
5	Hermione	Granger

SELECT * FROM Course

Id	Title	Credits
1	Potions	3
2	History of Magic	4
3	Transmutation	4
4	Defense Against the Dark Arts	3

SELECT * FROM Enrolled

person_id	course_id	Role
1	1	3
1	2	3
3	1	1
5	1	2

Here, we might define "role" to be 1 = "teacher", 2 = "TA", and 3 = "student."

2.5 REVIEW QUESTIONS AND ACTIVITIES

1. How would you define the following terms in your own words? Can you give an example of each?
   a. **Data, information,** and **knowledge**
   b. **Database** and **database system**
   c. **Metadata** and **end-user data**
2. Name THREE institutions (such as governments, companies, or medical providers) that likely have a database containing your information. What problems could occur if your data was not maintained or was stored incorrectly?
3. Give an example of a time you have used a "flat file" or "file system" to store data. Might a database work better for this? Why or why not?
4. The above Hogwarts database is *unrealistically* simple. What other data might we want to include in a school database? What database "tables" might we use?

3 TYPES OF DATA, DATABASES, AND DATABASE MANAGEMENT SYSTEMS

In this section, we'll be answering the following questions:

4. What are the differences between **unstructured, structured,** and **semistructured data?**
5. What are the parts of a **database system?** What sorts of services do they provide?
6. What sorts of careers involve working with databases?

3.1 FACTORS IN CHOOSING A DBMS

Figure 2 Disorganized data is like searching for a needle in a haystack (Brendan Shea x Dall-E).

In the previous section, we said that a Database Management System (DBMS) is simply a collection of programs that allows users to interact with the database. More specifically, it will enable users to carry out the core "CRUD" operations of **C**reating new records, Retrieving information to answer queries, Updating existing records by adding or altering them, and **D**eleting records that are no longer needed.

In choosing a particular DBMS, some of the most important distinctions are as follows:

- **How many users are there?** A **personal** database will be used by only one user, while a **workgroup** database will be used by a small-ish group of less than 50 or so people. Finally, an **enterprise** database can support *many* users, even when these users are running **concurrent** processes.
- **Where is the database "located"?** Some **centralized** databases have all their data in a single physical location, while **distributed** databases have data separated across networks. Both centralized and distributed databases require dedicated hardware (and staff to maintain this hardware). By contrast, the increasingly popular **cloud** databases live on remote machines in the "cloud" (provided by companies such as Amazon, Google, IBM, or Microsoft).
- **How is the data structured?** Data is **structured** if and only if each piece of data follows a pre-defined data model. So, for example, if we want to store data on customers, and each customer has an email address, a first name, a last name, and a mailing address, this is structured data. This will, in general, require processing the **unstructured data** that we are initially presented with (for example, we use the "unstructured data" provided by a conversation with the customer to produce a structured database record). Finally, **semistructured** data has *some* shared internal structure (and has been processed to some extent) but does not strictly follow any single data model.
    - Traditional **relational databases** are the best choice for dealing with highly structured data, which is accessed using SQL or similar languages. These databases have dominated the market since the 1970s. By contrast, non-relational **NoSQL** and **XML** databases might deal better with semistructured data (such as websites, which have an internal "structure.").
    - IMPORTANT: All data "begins" life as unstructured data and *becomes* structured by various processes.
- **How much data is there?** In recent years, there's been a lot of talk about **"Big Data,"** which refers to the fact that we can simply collect and store *much, much* more data than we used to. Some database systems scale better than others to massive datasets. For example, computers can automatically collect and store digital photos, social media posts, geographic locations, etc.
- **Is the software proprietary or open source?** Some DBMSs are owned by private corporations, which means users will need to pay for many features (but may, in return, receive additional support). Others are **open source,** which means the application code is freely available both to users and to programmers who might want to *improve them.*

3.2 Most Commonly Used DBMS

As of 2022, the most widely used databases are the following. Unless explicitly noted otherwise (e.g., NoSQL) all of these are *relational database.*

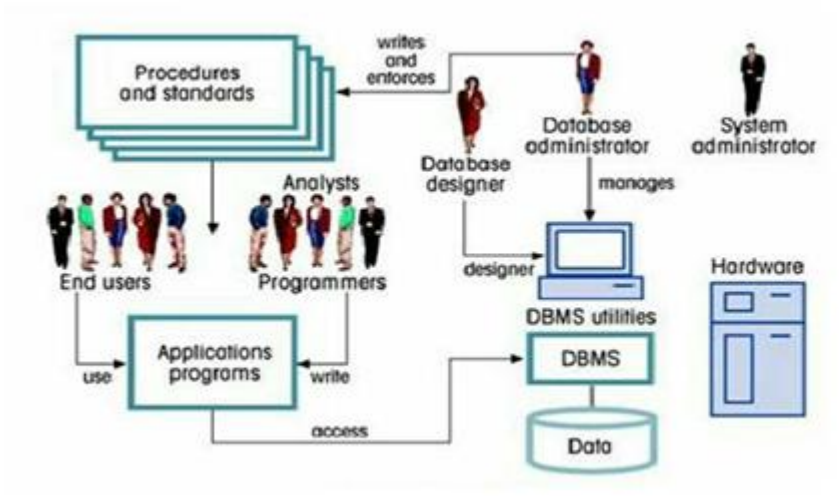
1. **Oracle Database (www.oracle.com/database/), Microsoft SQL Server (www.microsoft.com/en-us/sql-server/sql-server-2019) and IBM DB2 (www.ibm.com/products/db2-database).** These are proprietary, well-supported relational DBMSs that have been around for years. They are designed to handle huge datasets with large numbers of concurrent users distributed across computer networks. Many large organizations (such as Fortune 500 companies, government entities, universities, etc.) will adopt one of the systems. Luckily, they all use variants of SQL, which means that many of the skills needed to use are transferable to the others.
2. **MySQL (www.mysql.com/) , PostgreSQL (www.postgresql.org/) , and MariaDB (mariadb.org/).** These are the most common open-source alternatives to the databases described above and have much of the same functionality. MySQL has traditionally been dominant, but there has been a move toward Postgres and MariaDB after Oracle acquired MySQL. Some people have worried that the "free" version of MySQL would eventually become outdated (if Oracle failed to maintain it). These also use SQL.
3. **SQLite (www.sqlite.org/)** is a database engine widely deployed in the "internals" of various computer programs, mobile phones, Internet of Things devices, and elsewhere. There are *many, many* copies of SQLite in existence (in fact, multiple copies likely exist on your computer, phone, smart devices, etc.). It is helpful for many applications that do NOT require large numbers of concurrent writers. This includes many websites, scientific research projects, desktop/mobile apps, etc. It uses SQL!
4. **Microsoft Access** is a personal/workgroup database included with Office that provides a user-friendly interface and is often the database of choice for individuals and small businesses. Access supports SQL queries alongside its own visual interface. However, it can't support massive datasets or large numbers of concurrent users.
5. **MongoDB (www.mongodb.com)** is the most widely used non-relational **NoSQL** database. It is based on the **Javascript Object Notation (JSON)** format for storing/exchanging data regarding objects and lists (i.e., the data structures used internally by programming languages like Javascript, Java, or Python). In applications that deal primarily with semi-structured data (such as many web applications), this sort of database can perform very well but struggles on tasks that relational databases do well on (such as transaction support or responding to certain ad hoc queries). Other NoSQL databases include Amazon's **DynamoDB** and Google's **Bigtable**, developed originally to meet the respective corporations' unique data needs.

This is an incomplete list! However, the databases listed here dominate the market, and most others will follow the same general "rules" as these do. For many small or medium-size projects, something simple like Microsoft Access or SQLite can often be perfectly adequate, with the open-source software Postgres or MySQL providing additional functionality. In corporate environments, the choice of the database will usually be made long before you arrive, with Oracle and Microsoft being the current standards.

3.3 What are the Parts of Database Systems? What do They Do?

A **database system** is "An organization of components that defines and regulates the collection, storage, management, and use of data in a database environment" (C-M). It includes a wide variety of people and machines. Here is a simple picture[1] of a database system:

[1] Prabhjot Prabhjot and N. Sharma, "Overview of the Database Management System," 2017, https://doi.org/10.26483/IJARCS.V8I4.3778.



What does the DBMS do? In this picture, the DBMS serves as a "translator" between users of data (customers, analysts, programmers, etc.) and the data. Some of its most essential functions are as follows:

1. **It provides data abstraction and data independence.** End-users and programmers do NOT want to worry about how their data is stored. They don't care about file names, which machine it happens to "live" on, or any of this. They *really* don't want to change *their* processes just because there's been an internal change to how data is stored/arranged. The DBMS provides this by allowing users a "logical" view of data that is abstracted from these details and is independent of the underlying physical processes.
    a. DBMSs take care of all the messy business about where to save files, what to call them, what format (B-tree vs. hash table or whatever) will provide the best search results, etc.
    b. DBMSs basically promise users that IF you ask them a question using the right language (SQL), they'll answer it, and you don't need to worry about *how* they do it.
2. **It ensures data integrity and security.** DBMSs make sure that different users all have access to the data *they* need without giving them access to other data that may be sensitive. They ensure data integrity by making sure that the changes made by different users lead to an accurate and consistent internal state of the database.
    a. For example, suppose that we begin with "$1,000" in a customer's account and two *different* processes add "$100" to it simultaneously. At the end of the day, we want the account to have "$1,200" (instead of $1,100). We also don't want *unauthorized* users to alter this balance.
3. **It provides different "views."** A DBMS can serve different views of data to different people. So, for example, it "knows" that people in the U.S. like to have dates formulated as "mm/dd/yyyy," while those in the U.K. would prefer them as "dd/mm/yyyy." More importantly, it can display precisely the data that an individual user wants, without showing data that is *irrelevant* to them.

3.4 WHO WORKS WITH DATABASES?

While many people (nearly everyone) rely on databases in various aspects of their lives, a smaller number of people make their living on designing, maintaining, and using these systems:

- **Database administrators (DBA)** install and maintain the hardware and software required for DBMSs. They also help develop, test, and optimize the database to meet users' needs. These jobs generally require a B.S. or the equivalent.

- **Data engineers and database developers** often *design* databases to fit the data. In contrast to DBAs, who focus on ensuring the DBMS is functioning correctly, they tend to spend more time thinking about how to format and represent data. Like DBAs, these jobs generally require a B.S.
- **Data analysts and data scientists** spend their time writing programs and queries to "mine" various databases for data, using these to answer questions of interest. Many of these jobs require *at least* a B.S. (with many requiring an M.S. or even Ph.D.).
- **Software and web developers, cybersecurity professionals, and others** don't necessarily need to be experts on every aspect of databases. However, a basic knowledge of how databases (and SQL) work can help make everyday tasks easier and faster. These jobs have a wide variety of entrance requirements, with most requiring at least an associate's degree in Computer Science or a closely related field.

These jobs are, in general, well-paying and highly-regarded jobs. However, even if you are NOT planning on working in these careers, a basic knowledge of databases can help you understand why/how modern business and science "work" and how this might affect you.

3.5 Review Questions and Activities

5. Have you used any database software before? If so, what was it?
6. How do you use databases now? How might you use them in your future intended career?
7. Suppose you have a friend who owns a computer repair shop and is considering using a DBMS (she has been using spreadsheets until now). Describe what a DBMS is and what advantages or disadvantages this might have for her.
8. Look up ONE of the DBMSs mentioned above. Answer the following questions:
    a. When was it first released? Who developed it?
    b. What "type" of a database is it? What are some of its main features?
    c. Is there a "free" version of the database software available? Does it work for P.C.s, Macs, and/or Linux?

4 Optional Reading: Information, Knowledge & Intelligence (by Alistair MacFarlane)

[Brendan's Note: These readings are "optional" because I won't test you on them! However, I think they are interesting and can help connect some of the (theoretical) ideas we're learning about to "real-world" problems, which can sometimes be an issue in classes such as this 😊.]

The author of this piece (Alistair MacFarlane) was the head of Information Engineering at Cambridge in the late 1970s when modern data models were taking off. Here, he provides a helpful discussion of some key concepts, though his terminology is somewhat different than that I've used above).

Alistair MacFarlane considers the differences between these crucial concepts, and the implications for how we think about computers.

Anybody reading this will have interacted with a computer. These exchanges can hardly be called conversations. One no more converses with current computers than a soldier on parade converses with a drill sergeant. However, when working on the Internet using your personal computer as an intermediary, things are rather better. Searching for a book, inspecting its contents online, then ordering it, can be a much more satisfying form of interaction. You are never, however, under any illusion that you are dealing with a fellow human: you don't ask the Internet, "What's the weather like down there?… Have you read it yourself?" Computer-driven machines can now carry out a huge range of very highly skilled tasks, from navigating and landing aircraft, to manufacturing and assembling a wide range of products. How feasible is it that, in a few decades, one might have great difficulty in knowing whether or not one was talking to a computer? And if unable to finally evade all your attempts to engage it in stimulating conversation ("I'm sorry, but I'm too busy to chat with you right now, what was it you wanted?"), it could answer virtually any general knowledge question, provide detailed guidance over the whole range of literature and science, do seriously advanced mathematics, and play a mean game of poker, should you call it *intelligent*? After all, it knows the capital, population and main exports of every country in the world, and you don't. Furthermore, are there serious implications for society if computers linked to machines and communication systems could run all the railroads, fly all the aircraft, manage all the traffic, make all the cars and other products, act as vast reservoirs of factual knowledge, and perform almost any other activity requiring great skill? These are not merely interesting philosophical questions. Should machines reach the requisite levels of knowledge and skill, their integration into society could pose very severe problems.

To address these questions, we need to define carefully three basic concepts – information, knowledge, and intelligence – and explore the relationships between them. A good way to begin to distinguish between them, is to note how they reflect our relationship to present, past, and future. *Information describes*: it tells us how the world is now. *Knowledge prescribes*: it tells us what to do on the basis of accumulated past experience. *Intelligence decides*: it guides, predicts and advises, telling us what may be done in circumstances not previously encountered, and what the outcome is likely to be.

4.1 INFORMATION

Information is a meaningful, shareable *pattern*. We have evolved as a species, and learned as individuals, to recognise and ascribe meaning to patterns. A good example is the text you are reading now. This linguistic pattern is realised physically, grasped mentally, and can be shared socially. Medical technology routinely correlates images of brain activity with mental processes, illustrating how our mental constructs are grounded in physical patterns in our brains. Over a vast span of time, evolution has given us a set of interlinked physical processes by means of which we can transform these information-bearing brain patterns into other physically-grounded patterns, such as sound waves and signs so that we can communicate, and so share our information.

Since the patterns that carry information have a physical realisation, we can *measure* amounts of information. All patterns, such as the pictures on a computer or phone screen, can be built up from fundamental building blocks; each elementary block supplying one *bit* of information. (The bit is the basic unit of measurement of information.) There is a well-established quantitative theory of physical information that underpins information technology. However, this theory completely ignores the semantic [meaning-related] and social aspects of information. Science and technology only deal with the physical part. For a satisfactory philosophical treatment, information must be treated as a *triadic* concept: we must deal with its mental, social and physical aspects. Despite this, some misguided attempts have been made to construct philosophical theories based solely on a physical characterisation of information.

4.2 KNOWLEDGE

According to Plato, knowledge is *justified true belief*, and this was also the view of many philosophers until recently. But in the light of modern evolutionary and biological theory, we now know this is only part of the story. We are born with innate knowledge, whereas we are not born with innate beliefs. A newborn baby knows how to breathe and suckle, but it does not have religious, or any other, beliefs.

Knowledge is a store of information proven useful for a capacity to act. Some knowledge is innate, but most is gained by interaction with the world. For the simplest kind of agents, like insects or robots, all their knowledge is built into their structure. Large molecular machines, like the ribosomes in our cells, 'know' how to make proteins because their complex shapes have evolved to possess this potential. A thermocouple 'knows' what temperature corresponds to the closure of a switch because its designer made it in such a way that the switch will close when a certain temperature is reached.

The store of knowledge with which we are born does not enable us to cope with all the problems the world poses. As we interact with the world and with society, we have to generate new knowledge by learning. This gives rise to candidates for knowledge that we call *beliefs*. Our actions are driven by knowledge *and* beliefs. In the jargon of philosophy of mind, beliefs are *intentional* mental states, meaning that they are *about* the world. Belief tells us how the world might be. Knowledge tells us the way the world is. Learnt knowledge is distilled out of beliefs. Explicit knowledge is distinguished from belief in terms of its coherence, persistence, reliability, and effectiveness: knowledge is achieved through an accumulated and refined experience which ensures that our actions conform to what we experience. So only sustained experience based on repeated interactions with the world can justify the upgrading of a belief to knowledge. But beneath the veneer of learning and experience, we are complex, irrational, and emotional creatures constantly negotiating our own compromises between beliefs and knowledge.

The ability to form beliefs and test them in action is essential to our survival in an ever-changing world, and the development of this ability was a crucial step in our evolution. The flexibility and robustness of intentional action has an immense evolutionary advantage over purely innate, fixed-representation-based activity such as is manifested as instinct. It allows the manipulation and examination of possible options before a final determination of action.

Only explicit, language-based knowledge can be easily transmitted via books and machines. But much of our knowledge is *tacit*; it is not expressible in language. This was memorably expressed by Michael Polanyi when he said, "We know more than we can tell." This is true not only for innate knowledge, but also for those internalised skills which come from observing and copying others. Furthermore, much of our knowledge is difficult to access at will, and may require group interaction (brainstorming) to surface. Accessing and validating human knowledge will remain a difficult process.

Belief and knowledge are mental phenomena, which are notoriously resistant to simplistic physical explanations. Attempts to explain contents of consciousness such as belief in purely physical terms have had very limited success. The difficulty stems from their immaterial nature. They can be *correlated* with physical events, but correlation is not an explanation. Description, however, is a much more modest goal than explanation. A top-down *description* of how our experience is grounded in our physical, mental and social interactions is therefore a philosophical construction or way of looking at the world, not a scientific theory.

4.3 INTELLIGENCE

Unlike belief and knowledge, intelligence is not information: it is a *process*, or an innate capacity to use information in order to respond to ever-changing requirements. It is a capacity to acquire, adapt, modify, extend and use information in order to solve problems. Therefore, intelligence is the ability to cope with unpredictable circumstances. But intelligence is not merely analytical: to survive and flourish in society, we must also have social and emotional intelligence. (However, I do not here assume an equating of intelligence with *consciousness*.)

Intelligence involves a capacity to adapt and learn that has been accumulated throughout our species' development, yet intelligence is something we have by virtue of our *individual* genetic inheritance. By assiduous practice we can become more skilled. By diligent study and careful observation, we can become more knowledgeable. But there is no way we can personally become significantly more intelligent. You can no more double your natural intelligence than you can double your natural height.

4.4 A.I. AND YOU-AND-I

'Artificial Intelligence is a misleading term for the technology of endowing machines with agency (an ability to act autonomously in the world). In our present state of technical and scientific knowledge, a much better term would be 'Artificial Knowledge', although this doesn't have quite the same ring. Recent advances in the technology for the search and retrieval of semantically-tagged factual information – as in the systems used by Google's search engines – have led to astonishing improvements in the speed and relevance of Internet searches. (The semantic tagging is provided by human users: the more frequently users use specific search words, the more weight these are given by the search procedures.) By virtue of these techniques, information technology can make accessible a virtually unbounded store of knowledge.

This capability for the acquisition of knowledge applies not only to facts, but to any process that can be precisely and explicitly described in the languages used to program computers. Thus in principle, and increasingly in practice, machines can be made skilful in mechanical tasks to any level attainable by humans, and higher, and can be made knowledgeable to depths far beyond any individual human capacity. Creating machine *intelligence*, however, poses challenges of an altogether different order. The reason for this is that intelligence is basically a capacity to use what has been learned to deal with the novel and the unexpected. Intelligence deals with what we *don't* know, so it is not easily describable by rules and procedures; it does not simply draw on a reservoir of facts. So this is a process for which we are presently unable to formulate a precise description that we could program into a computer. We must note however the crucial fact that much of the *knowledge* on which human agency depends *is* objective, formal and explicit, and so is of a form implementable via machine agency.

The differences, both qualitative and quantitative, between human and machine agency can be summarised in terms of three gaps corresponding to different levels of agency: a *skills* gap, a *knowledge* gap, and a *personhood* gap. We cannot hope to match machines in terms of the range and accuracy of their perceptions, the speed and power of their calculations, or the delicacy and precision of their manipulations. As processing power, the range and sensitivity of sensors, and the creation of new forms of actuators continue to develop, machines will become ever more capable over the whole range of manipulative skills. An ever-widening *skills* gap will open up between human and machine. Nor can we hope to match machines in handling intractable masses of data, or in applying processing power to complex formal systems such as mathematics. Computers are better at storing and retrieving knowledge, and at manipulating formal, symbol-based systems like mathematics. There will be an ever-increasing *knowledge* gap between human and machine.

However, there are immensely complex information-processing systems that have evolved in the human brain that cannot be replicated in any machine by any process of formal design or experiment, certainly not for decades to come, perhaps not for centuries. The complexity of our brains is vast. It has arisen from a compression of aeons' worth of information accumulated over the evolution of our species into the genetic material that defines us, and there are no short cuts to its replication. So there will remain a *personhood* gap between human and machine that will continue to make human levels of intelligence, emotional insight, and ability to handle uncertainty, unavailable to machines. Within any currently conceivable future horizon of prediction, human and machine agency will remain complementary. We will have to learn how to live with them, but they cannot replace us.

4.5 Living With Machines

We can now respond to the three questions posed in the introduction. First, could we, in the fairly near future, tell whether we were talking to a computer? Like all philosophical questions, the answer depends crucially on the understanding of the terms involved. This question is best answered in two parts: specific and generic. Specifically, for basic question-and-answer computer systems, there would be no financial incentive for making them any more sophisticated than necessary in terms of general knowledge and pseudo-conversation. Thus, although there will be a progressive removal of their more irritating limitations, it will remain clear that no human was directly involved. For more complex systems designed to give better access to general knowledge questions, or to solve mathematical and scientific problems, the answer is even counter-intuitive: the more they know, and the more rational they appear, the surer we could be that they're not human! We would experience a dawning conviction that we're talking to some sort of automated encyclopedia, perhaps equipped with a low form of cunning. It could be programmed in such a way that, when faced with any question it could not handle, it would always give an evasive answer ("I'm sorry but I'm not allowed to answer personal questions…") So it will be increasingly difficult to be absolutely certain. You might, after all, be talking to someone who had just won a Nobel Prize.

So in principle, a supercomputer could, I believe, be programmed so that engaging it in general conversation, however wide-ranging, would not be enough to convince you that you were not conversing with a human. Skill in answering, and even more, in evading, questions, linked to a comprehensive general knowledge, and great ability to solve scientific, logical and mathematical problems, could be combined with social skills and a spurious empathy. A much more stringent approach would need to be applied to discover the truth, which would involve searching tests for creative intelligence, deep empathy and social adaptation. And devising adequate tests will become more and more difficult as processing power increases. If dedicated experts could no longer decide one way or another, the implications would be momentous. So, to answer the second question: Should we call such a computer intelligent? Not yet, but we could certainly call one very *knowledgeable*. However, achieving human levels of intelligence, empathy and emotional capacity probably lies far into the future.

The answer to the third question is that there are immensely important implications for society as computer skills increasingly surpass human skills, yet the huge economic gains that would accrue from an increasing use of information technology harnessed to machines will prove irresistible. Our ultimate challenge will not be how to further develop such technology, but to learn how to live with this in a new form of society. So the basic problems facing that society will be political rather than technical. Many of the core problems of philosophy will need to be re-examined.

There are grounds for both optimism and pessimism when contemplating this future. Those of a pessimistic nature may well conclude that the strains imposed on society by the continuing impact of information technology on employment, commerce and education may prove intolerable. Those of a more sanguine outlook might conclude that, with all basic material needs satisfied, and with the vast resources of an ever-increasing fund of technical and scientific knowledge readily available, we will be entering a new Golden Age free from want, disease and privation. Sceptics might point out that, when speculating on what might lie beyond a presently insurmountable mountain range, all we can do is survey thoroughly the terrain leading up to it, and issue some warnings.

The philosophical, scientific, technical, economic and social consequences promised by the ever-increasing use of information technology harnessed to machines are enormously significant. Scientists and technologists may find the challenges exhilarating, but the consequent political, social and economic difficulties for the rest of the population are daunting. Those who first see clearly what is in store for us will face an awesome prospect.