Need anything? Tell your Class Reps

You can:
• Friend us on Facebook
• Talk to us face-2-face
• Ask for us in the Group Chat
• Email ug1-reps@inf.ed.ac.uk

JOIN NOW!! Link in the group chat!!
Informatics Class of 2021 facebook group
Homework from Tuesday

Jeannette Wing

“Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science”

See also https://is.gd/channel9wing
Course website and blog https://blog.inf.ed.ac.uk/da18

Piazza discussion group Inf1-DA

Mailing list inf1-da-students@inf.ed.ac.uk
Posting is restricted to list members; send from your University email address.

Facebook group inf1da2018

Twitter @inf1da18
This is not a textbook course, and there is no single compulsory book.

For certain parts of the course, however, I shall indicate one or more books which cover the current material — usually in much more depth and generality than required for this introductory course.

You can consult these books in the library, or borrow them, and you may find one or other helpful to you. Although the content is often similar, styles and tastes can differ significantly.

Occasionally I shall distribute PDF and photocopies of an individual textbook chapter when it is especially relevant to the course.
Tutorials and Exercise Sheets

Tutorials start in **Week 3** of semester and continue each week until the end of semester, except for Flexible Learning Week which falls between lecturing weeks 5 and 6.

Your *course tutor* leads your tutorial group; this is not the same as your *personal tutor*. Group membership, times and study rooms will be completed during Week 2: most allocations are done, but not all.

If you wish to move to a different tutorial group then you will be able to do so using the Informatics student portal:

https://www.student.inf.ed.ac.uk

Studying for Inf-DA requires you to attend the weekly tutorials. If you are ill or otherwise unable to attend one week then email your course tutor, and if possible attend another tutorial group in the same week.
Structured Data

- e.g. University database of students, staff, courses, rooms, etc.
- Which students take Inf1-DA? How do we timetable exams?

Semistructured Data

- e.g. Tourist factbook about countries, regions, cities, ...
- e.g. BioModels repository of computational models for biochemical reactions and metabolic pathways.

Unstructured Data

- e.g. British National Corpus of spoken and written English.
Some application domains involve handling quantities of data that can be very strictly organised. For example:

- The University of Edinburgh records some standardized pieces of information about each of several thousand students.
- A supermarket chain will maintain information on tens of thousands of product lines, and the stock in each shop where they are sold.
- A web browser may keep details of usernames, passwords, and preferences for the websites a user visits.

What’s central to this structure is that we are working with the same information about many different individuals. Even when there are different kinds of individual (product lines, shops, staff, ...) there are far more items of each kind than there are different kinds.
Structured Data

Some application domains involve handling quantities of data that can be very strictly organised. For example:

- The University of Edinburgh records some standardized pieces of information about each of several thousand students.
- A supermarket chain will maintain information on tens of thousands of product lines, and the stock in each shop where they are sold.
- A web browser may keep details of usernames, passwords, and preferences for the websites a user visits.

As well as individuals or entities, it’s usually important to also work with the relationship between individuals: which students take which course, or which shop stocks which product.
Some application domains involve handling quantities of data that can be very strictly organised. For example:

- The University of Edinburgh records some standardized pieces of information about each of several thousand students.
- A supermarket chain will maintain information on tens of thousands of product lines, and the stock in each shop where they are sold.
- A web browser may keep details of usernames, passwords, and preferences for the websites a user visits.

It turns out — perhaps unexpectedly — to be very effective to concentrate more on the relations between things than on the things themselves.

“The fundamental interconnectedness of all things”
Douglas Adams, Dirk Gently’s Holistic Detective Agency
Data Representation

This first course section starts by presenting two common data representation models.

- The entity-relationship (ER) model
- The relational model

Note slightly different naming: -relationship vs. relational

Data Manipulation

This is followed by some methods for manipulating data in the relational model and using it to extract information.

- Relational algebra
- The tuple-relational calculus
- The query language SQL
Database Design

1. **Requirements analysis**
   Understand what data is to be stored in the database and what operations on it are likely to be needed.

2. **Conceptual design**
   Develop a high-level description of data to be stored, and any *constraints* that might apply to the data.
   This is the level where we might use an **ER data model**.

3. **Logical design**
   Implement the conceptual design by mapping it to a specific data representation. The outcome is a *logical schema*.
   For example, implementation can be performed by translating the ER data model into a relational data model.
The ER Data Model

- **What is it?**
  The ER model is a way to organise description of *entities* (individual things in the real world) and the *relationships* between them.

- **Why is it useful?**
  It readily maps into different *logical data models*, such as the relational model.

- **How is it used?**
  As a graphical notation for visualizing the structure of data, to clarify and communicate that structure.

---

**P. P. Chen**

Entities and Entity Sets

An *entity* is any kind of thing that we want to model in our database. For example, a university database might be designed to capture notions of course or student.

Within this, each individual item is an *entity instance*; and the collection of all such items is an *entity set*.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Student</th>
<th>Course</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity instance</td>
<td>A. Lovelace</td>
<td>Inf1-DA 2017/18</td>
</tr>
<tr>
<td>Entity set</td>
<td>Edinburgh students</td>
<td>Edinburgh courses</td>
</tr>
</tbody>
</table>
Entity-relationship modelling provides a graphical language for describing entities and the relationships between them in an *ER diagram*.

The ER diagram syntax for an entity is a rectangle, labelled with the kind of entity it represents.
Attributes

An entity is described by its characteristic *attributes*.

These are the properties to be captured in the data model.

An ER diagram shows attributes as ovals, labelled with the attribute’s name, connected to the appropriate entity.
Domains

Each attribute has a *domain* of allowed values, similar to the use of types in Haskell or Java. For example, *Age* could have domain “positive integers”, while the domain for *email* might be “strings of up to 254 characters”.
Attributes list the information recorded in a model for each entity instance. Attributes are also used to identify entity instances and distinguish between them. This is the role of a *key*: a chosen set of attributes.
A set of attributes is a *superkey* for an entity if those attributes, taken together, always uniquely identify every entity instance.

A set of attributes is a *key* if it is a minimal set of identifying attributes — removing any one attribute would make it no longer uniquely identifying.

<table>
<thead>
<tr>
<th>Building Number</th>
<th>Street Name</th>
<th>Town</th>
<th>Postcode</th>
<th>Appleton Tower:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superkey</td>
<td>BN</td>
<td>SN</td>
<td>T</td>
<td>11 Crichton Street</td>
</tr>
<tr>
<td></td>
<td>SN</td>
<td>T</td>
<td>PC</td>
<td>Edinburgh EH8 9LE</td>
</tr>
<tr>
<td>Superkey</td>
<td>BN</td>
<td>SN</td>
<td>T</td>
<td>Key</td>
</tr>
<tr>
<td>Superkey</td>
<td>BN</td>
<td>SN</td>
<td>PC</td>
<td>Key</td>
</tr>
</tbody>
</table>

BN SN T PC
Primary Keys

Again: a *key* is a **minimal** set of attributes whose values **uniquely** identify each entity instance in an entity set.

Where there is more than one such attribute set, each is a *candidate key*.

From all the candidates keys we choose a *primary key* to be used as the unique identifier for the entity.

<table>
<thead>
<tr>
<th>Building Number</th>
<th>Street Name</th>
<th>Town</th>
<th>Postcode</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Superkey</strong> BN</td>
<td>SN</td>
<td>T</td>
<td>PC</td>
</tr>
<tr>
<td><strong>Superkey</strong> BN</td>
<td>SN</td>
<td></td>
<td>PC</td>
</tr>
<tr>
<td><strong>Superkey</strong> BN</td>
<td>SN</td>
<td>T</td>
<td>Key</td>
</tr>
<tr>
<td><strong>Superkey</strong> BN</td>
<td>SN</td>
<td></td>
<td>PC</td>
</tr>
<tr>
<td><strong>BN SN</strong></td>
<td></td>
<td></td>
<td>PC</td>
</tr>
</tbody>
</table>

Appleton Tower: 11 Crichton Street Edinburgh EH8 9LE
Primary Keys

Any key is a set of attributes: it may contain one, two, or more attributes.

A key made of more than one attribute is a *composite key*.

The ER diagram syntax underlines each attribute that is part of the chosen primary key.
Summary: Keys

- A superkey is any set of attributes whose values uniquely identify each entity instance in an entity set.

- A key is a minimal set of attributes whose values uniquely identify each entity instance in an entity set.

- Where there is more than one such set, each forms a candidate key.

- Any key with more than one attribute is a composite key.

- One of the candidate keys is selected as the primary key.

- In an ER diagram each attribute in the primary key is underlined.
A *relationship* is an association between entities. For example, the *takes* relationship between students and courses.

Each individual occurrence of the relationship is a *relationship instance*, and the collection of all such is a *relationship set*.

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Takes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationship instance</td>
<td>((s0456782), (INF08013, 2016))</td>
</tr>
<tr>
<td>Relationship set</td>
<td>Edinburgh course registrations</td>
</tr>
</tbody>
</table>
ER diagrams show relationships as diamonds, labelled with the name of the relationship and connected to all the participating entities.

A relationship may also have its own attributes.
Relationships can be between two entities ("binary"), three ("ternary") or more ("n-ary").

**Relationship instance**  
(ASDA, Heinz, Ketchup, Edinburgh, £1.50)
There is no bound on the number of entities participating in a relationship. An entity may be involved in any number of different relationships.
Summary

The entity-relationship (ER) model is a way to organise the description of *entities* (individual things) and the *relationships* between them.

So far we have seen the following elements of ER modelling:

- **Entities**, all with characteristic *attributes*;
- For each kind of entity there are *entity instances* that together make up an *entity set*;
- A *key* as a minimal set of attributes to identify and distinguish entity instances;
- *Relationships* between entities.

Entity-relationship modelling provides a *graphical language* for describing this structure in an **ER diagram**.
Homework

Read This

Sections 2.1–2.4 of this textbook on databases: paper copies available here; I shall circulate PDF by email.

This is the recommended textbook for the third-year course Database Systems. It’s a large book, with thorough and extensive material on a wide range of database topics.

It is not necessary to buy this book for Inf1-DA. Instead:

Do This

Step inside the library, locate the HUB and find this book. Have a look at the other textbooks there, too, and compare style and content.

Finally: Find the video of this lecture online and watch the first minute. If you have any problems then read the blog for instructions, or ask on Piazza/Facebook.