Informatics 1: Data & Analysis
Session 2015/2016, Semester 2

Assignment Feedback

This is a report on the written coursework assignment for *Informatics 1: Data & Analysis.* That assignment included exam questions from 2013 and 2015, and this report is based on solutions submitted by students. Please note the following:

- This is not a set of "model" answers. It does contain solutions, which can be used to check your own answers; but there are also notes on different possible answers, key points, possible errors, and comments on the ways people approached each question in the exam itself and as coursework.
- Not all the questions have a single "right" answer. There can be multiple correct ways to write a database query, explain a concept, or construct an example. This report includes some variants on answers, but still cannot cover every possible correct alternative.
- Practising past exam questions is one way to learn more about a subject, but it is quite limited and not enough on its own. Even when an exams routinely follow a fixed structure, the questions change and successful performance does essentially depend on a good understanding of the material in the course.

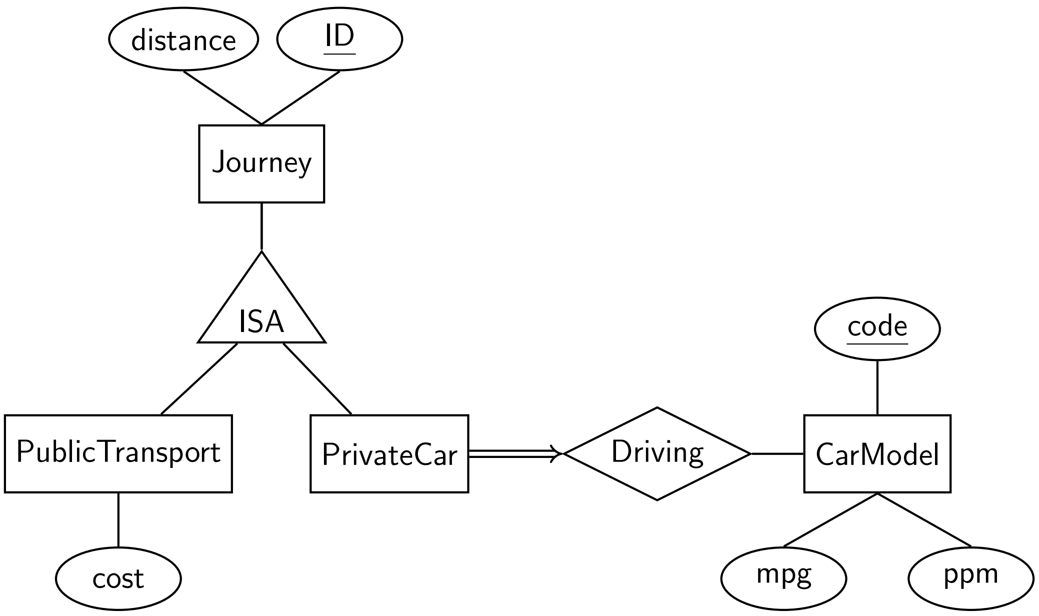
The assignment consisted of three questions, each with several subquestions. The rest of this report gives the full text of each question followed by notes on solutions and feedback on the answers given by students.

Where you find errors in these notes, please send them to me at Ian.Stark@ed.ac.uk

Ian Stark
2016-03-27

Question 1 [*This question is worth a total of 35 marks.*]

The entity-relationship diagram below shows a fragment of a proposed model for a database managing travel claims by university staff. Each journey is labelled with a unique ID, and journeys by car or by public transport have different information recorded.



You also have the following information about the model.

- Attributes distance, cost, mpg (miles per gallon fuel consumption) and ppm (pence per mile travel allowance) are represented as integers.
- All other attributes are alphanumeric values up to 8 characters long.
- Every instance of the CarModel entity must have mpg and ppm values provided.

(a) Draw up an SQL data declaration of appropriate tables to implement this entity-relationship model. (You need not include **on delete** declarations). [*20 marks*]

QUESTION CONTINUES ON NEXT PAGE

Another part of the same database is to contain information about parking provision for cars and bikes at different campuses around the University. This includes the following three SQL tables.

```
create table Parking (
  ID          varchar(10),
  type        varchar(10) not null,
  number      integer not null,
  location    varchar(16) not null,
  primary key (ID),
  foreign key (type) references ParkingType(code),
  foreign key (location) references Campus(name)
)
```

Typical Parking record: {'AHT_FRONT','SHEF_WIDE','3','BioCampus'}

This records that parking area AHT_FRONT on the BioCampus has 3 parking spots of type SHEF_WIDE.

```
create table ParkingType (
  code        varchar(10),
  name        varchar(80) not null,
  capacity    integer not null,
  mode        varchar(6) not null,
  primary key (code)
)
```

Here the transport mode is either bike or car, and a typical ParkingType record would be {'SHEF_WIDE','Sheffield Rack, double width',12,'bike'}

This records that a SHEF_WIDE parking spot is a "double-width Sheffield Rack" with a capacity of 12 bikes.

```
create table Campus (
  name    varchar(16),
  primary key (name)
)
```

The Campus table contains rows like 'BioCampus', 'Central Area' and 'Sandwell Site'.

(b) The Campus table is referenced from Parking, but only has one column. Why might it be helpful to have a "relation" like this with just a single field? [*2 marks*]

(c) Give SQL queries, using the tables above, that carry out the following.

(i) List the names of every different type of bicycle parking in the database.

(ii) List, without duplication, all university campuses where there is parking for cars.

(iii) Compute how many space for bike parking there are on the BioCampus.

[*13 marks*]

Notes on Question 1

(a) The following tables capture the ER model given.

```
create table Journey (
  ID         varchar(8),
  distance  integer,
  primary key (ID)
)

create table PublicTransport (
  ID    varchar(8),
  cost  integer,
  primary key (ID),
  foreign key (ID) references Journey(ID)
)

create table PrivateCar (
  ID        varchar(8),
  driving   varchar(8) not null,
  primary key (ID),
  foreign key (ID) references Journey(ID),
  foreign key (driving) references CarModel(code)
)

create table CarModel (
  code varchar(8),
  mpg  integer not null,
  ppm  integer not null,
  primary key (code)
)
```

The **not null** declarations here are required by the constraints given in the question; it's also acceptable to put in further **not null** declarations.

Using a standalone Driving table would not be satisfactory, as this could not capture the key and participation constraints from the ER diagram. Instead, here the driving field within the PrivateCar table links directly to a record in the CarModel table. Note that this field is declared as **not null** to enforce the total participation constraint from the ER diagram.

The PublicTransport and PrivateCar tables should not duplicate the distance field as that is already available from the parent Journey table.

(b) Using a single-column table means that values in the location field of the Parking must be taken from this fixed list. This maintains consistency across the database — for example, by avoiding multiple alternate spellings of the same location.

Some students also suggested that a single-column table could later be expanded with more information about the different campuses. I think that's also a reasonable observation.

(c) (i)
```
select name from ParkingType where mode = 'bike'
```

(ii)
```
select distinct P.location
from Parking P, ParkingType T
where P.type = T.code and T.mode = 'car'
```

(iii)
```
select sum (P.number * T.capacity)
from Parking P, ParkingType T
where P.type = T.code and T.mode = 'bike' and P.location = 'BioCampus'
```

Note the multiplication in the sum for (c)(iii). This takes account both of the fact that bike parking units may have space for several bikes, and any one parking location might host several such units — as with the SHEF_WIDE rack mentioned in the question.

There are many possible minor variations in syntax. The **distinct** qualifier is required for (c)(ii); it's optional on (c)(i).

Many students listed correct tables in parts (c)(ii) and (c)(iii), but then failed to include the P.type = T.code test to make sure that rows from the two tables matched up.

Some students included the Campus table. In fact that's not necessary, as the Parking table already contains the name of the campus.

Question 2 [*This question is worth a total of 35 marks.*]

The following small XML document is a marked-up version of a speech from one of Shakespeare's plays.

```
<speech speaker="First Witch">
    <line>
        <w>When</w>
        <w>shall</w>
        <w>we</w>
        <w>three</w>
        <w>meet</w>
        <w>again</w>
    </line>
    <line>
        <w>In</w>
        <w>thunder</w>
        <punct>,</punct>
        <w>lightning</w>
        <punct>,</punct>
        <w>or</w>
        <w>in</w>
        <w>rain</w>
        <punct>?</punct>
        </line>
</speech>
```

(a) Draw this XML document as a tree, following the XPath data model. [*9 marks*]

(b) Write an XML DTD for a Speech document type to validate such speeches. Assume that every speech must have an identified speaker. [*12 marks*]

(c) Suppose a large XML document contains many such speeches, nested at various levels inside Plays, Acts, Scenes and so forth. Write XPath expressions to identify:

  (i) All lines spoken by Macbeth

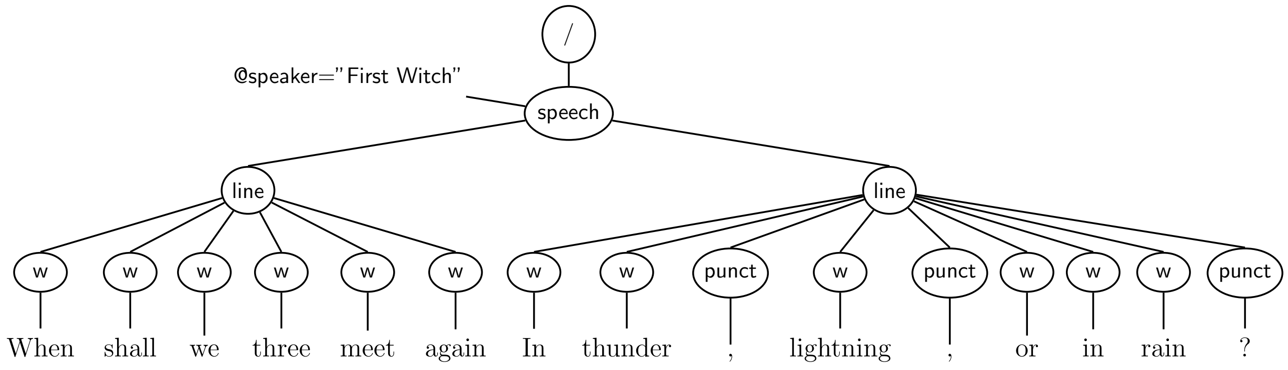
  (ii) All speakers using the word "blood" in a speech. [*8 marks*]

(d) The lines above come from the works of a single author. Standard corpora for linguistic research like the *British National Corpus* or the *Penn Treebank* bring together work from many sources. Building them requires balancing and sampling in order to ensure that they are representative.

Explain the meaning of *balancing*, *sampling* and *representative* here. [*6 marks*]

Notes on Question 2

(a) The document has the following tree structure in the XPath data model.



Note here: the root node "/"; the attribute node "@speaker=..."; and both word and punctuation nodes with text at the leaves.

Several students wrongly used backslash "\" for the root node.

(b) Here is an appropriate DTD.

```
<!ELEMENT speech (line+)>
<!ELEMENT line ((w|punct)+)>
<!ELEMENT w #PCDATA>
<!ELEMENT punct #PCDATA>
<!ATTLIST speech speaker CDATA #REQUIRED>
```

The speech and line elements could possibly use "$*$" repetition instead of "+", although that then allows empty speeches and lines. The question demands that the speaker attribute on speech be #REQUIRED.

Several students declared the line element as "(w+,punct$*$)" which is incorrect: the comma "," forces ordering, so this only works where punctuation is all at the end of the line.

Notice that the DTD does not include <!**DOCTYPE** Speech [...]> lines. Those are only required when inlining a DTD within an XML document, and isn't strictly part of the DTD itself.

(c) (i) All lines spoken by Macbeth:

```
//speech[@speaker='Macbeth']/line
```

Note that this will return the lines as XML nodes, with the actual lines as subtrees of individual words and punctuation marks.

(ii) All speakers using the word blood somewhere in a speech:

```
//speech[line/w/text()='blood']/@speaker
```

Various abbreviations or variations on this are possible too, for example:

```
//speech[.//w='blood']/@speaker
//w[text()='blood]/../../@speaker
```

Notice the self-reference '.' in the first excursion to avoid returning to the root node (this was specifically mentioned in lectures).

Quite a few students gave versions that didn't correctly walk around the tree — either too many or too few uses of "..", or using a predicate like "[/w='blood']" which goes back to the root of the tree.

(d) **Balancing** means choosing a range of different types of sources for the corpus: books, newspapers, blogs, letters, etc.

Sampling refers to selecting texts at random from the chosen sources.

Representative A corpus is *representative* if it contains a similar mix of text to the language variant for which it is being developed.

Question 3 [*This question is worth a total of 30 marks.*]

The standard "information retrieval (IR) task" is, given a *query* and a collection of *documents*, to find those documents that are relevant to the query.

(a) One measure of performance of an information retrieval algorithm is its *precision* P, usually computed using the following formula.

$$P = \frac{TP}{TP + FP}$$

State the matching formula for the other common performance measure, *recall* R.

[*2 marks*]

(b) Explain the following abbreviations used in these formulae — for each one, state what it stands for, and describe in a single sentence what it means. [*8 marks*]

$$TP, FP, TN, FN$$

(c) You are evaluating information retrieval systems to use with a collection of 2000 documents from an archive of 20th-century material. There are two candidate systems: *Happy* and *Sleepy*. Each is tested on the query "Enigma machine", for which there are 150 relevant documents in the archive.

Happy returns 600 documents, with 100 of those being relevant; *Sleepy* returns only 25, but all of them are relevant.

For each system, draw up a table of retrieval against relevance from this data, and use it to calculate precision and recall on this test. Show your working. [*8 marks*]

(d) The F_α performance measure combines precision and recall using the formula

$$F_\alpha = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} .$$

Compute $F_{0.8}$ for *Happy* and *Sleepy*. Which performs more strongly?

The *balanced* F-score uses *alpha* $= 0.5$. Does the choice of $\alpha = 0.8$ favour systems that are good at precision, or recall? [*4 marks*]

QUESTION CONTINUES ON NEXT PAGE

A simple document retrieval algorithm might look at whether certain words appear in a document. A more sophisticated algorithm might weigh those words by their significance, using a a measure such as *term frequency - inverse document frequency* (tf-idf).

One of the items in the 2000-document collection described earlier is identified as *Report Q*. You have the following information about word occurrences in this document, and across the collection.

- The word "Turing" appears 590 times in the collection, across 85 different documents; it appears 8 times in Report Q.
- The word "computer" appears 22 times in Report Q, and a total of 1700 times in the whole collection — it is in 400 of the 2000 documents.
- Of all the documents, 1600 contain no mention at all of either "Turing" or "computer".

(e) Use these figures to compute tf-idf for both "Turing" and "computer" in Report Q. (Note: you don't need all of the numbers above for this calculation, just some of them.) [*6 marks*]

(f) Which word is most characteristic of Report Q, "Turing" or "computer"? Explain your answer. [*2 marks*]

Notes on Question 3 Many students gave complete and correct answers to this question. The most common error was to omit parts (e) and (f) entirely.

(a) Recall is computed using the following formula.

$$R = \frac{TP}{TP + FN}$$

(b) **TP** True Positives: the number of documents returned by the system that are indeed relevant to the query.

FP False Positives: the number of documents returned by the system that are in fact not relevant to the query.

TN True Negatives: the number of documents not returned by the system that are not relevant to the query.

FN False Negatives: the number of documents that are relevant to the query but are not returned by the system.

Very many students got this correct, although a few mixed up the false positives and false negatives; care is still required.

(c) The following tables give all the necessary figures for the calculation.

Happy	Relevant	Not relevant	Total
Retrieved	100	500	600
Not retrieved	50	1350	1400
Total	150	1850	2000

Sleepy	Relevant	Not relevant	Total
Retrieved	25	0	25
Not retrieved	125	1850	1975
Total	150	1850	2000

The *marginal* values in these tables — the row and column totals — are helpful to first calculate and then check the values in all the cells.

From these we can evaluate the performance of each system on this single query.

$$\textit{Happy}\text{ precision } P = \frac{100}{600} = 1/6 = 0.17 \qquad \textit{Sleepy}\text{ precision } P = \frac{25}{25} = 1$$

$$\textit{Happy}\text{ recall } R = \frac{100}{150} = 2/3 = 0.67 \qquad \textit{Sleepy}\text{ recall } R = \frac{25}{150} = 1/6 = 0.17$$

Either fractions or decimals are fine here; precision of two decimal places is quite enough.

(d) The *Sleepy* system has the stronger $F_{0.8}$ score.

$$F_{0.8}(\textit{Happy}) = \frac{1}{0.8\frac{1}{0.17} + 0.2\frac{1}{0.67}} = 0.20$$

$$F_{0.8}(\textit{Sleepy}) = \frac{1}{0.8\frac{1}{1} + 0.2\frac{1}{0.17}} = 0.50$$

Choosing $\alpha = 0.8$ favours precision.

The F-score is always between 0 and 1. A few students made mistakes that gave values much higher than 1, which is a clear sign that something has gone wrong in the calculation.

Some students also missed out parts of the question, such as reporting which system had the stronger $F_{0.8}$ score, or what aspect $\alpha = 0.8$ favours. Once you have done the calculation these aren't especially hard, so it seems likely to me this was a slip in not reading and checking the question.

(e) The formula for tf-idf is as follows.

$$\text{tf-idf}(w) = f_i(w) \log\left(\frac{N}{N_w}\right)$$

where $f_i(w)$ is the absolute frequency of word w in document i, N is the total number of documents, and N_w is the number of documents in which word w appears.

Applying this to Report Q, we obtain:

$$\begin{aligned}\text{tf-idf(“Turing”)} &= 8\log\left(\frac{2000}{85}\right) &&= 8 \times 1.37 &&= 11.0\\ \text{tf-idf(“computer”)} &= 22\log\left(\frac{2000}{400}\right) &&= 22 \times 0.70 &&= 15.4\end{aligned}$$

The lecture course considered exactly one form of tf-idf, that given above. There are variations in the literature, which would be acceptable and shouldn't actually change the results: for example, in this case there isn't enough data given to use relative term frequency; and using logarithms to a different base gives the same ranking.

In the original exam, several students gave their answers to a quite unreasonable number of decimal places. Three significant figures is certainly enough. After all, the only test we are going to apply here is whether one is larger than the other.

(f) Of the two words, “computer” has the higher tf-idf score, which suggests that it is more characteristic of Report Q.

Notice that this is true even though “Turing” occurs in far fewer documents from the collection. In the end the high number of occurrences of “computer” in Report Q outweighs the rarity of “Turing”.