January 1993

Database Design For Corpus Storage: The ET10-63 Data Model

Tony McEnery & Béatrice Daille

I. General Presentation

Within the ET10-63 project, a French-English bilingual corpus of about 2 million words is available. This corpus is stored in a relational database. This paper seeks to show the advantages of storing a corpus in a database, and to show the formal design methodology which underlies the storage model of the ET10-63 corpus.

Traditional file systems are the most common storage medium for corpora at present. Within these, corpora are stored as simple linear text files, with only primitive attempts made at systematic organization. For example, the LOB corpus (Johnasson, Leech and Goodluck, 1978) is stored as a series of files, each file reflecting a particular textual genre. Hence the only apparent organization within the file structure of LOB is a rough genre based division. This imposes serious limits on the functionality of the system. The corpus when stored should meet a goal of functional adequacy. To enumerate the aspects of functional adequacy one should consider that:

- all implicit or explicit data in the corpus, e.g. from the tagger, lemmatizer or other well-defined application, is stored in a way which makes recovery possible,
- the data is the same for all programmers and users,
- the data is structured in a way which allows statistics to be performed on it with ease.

Clearly this only states the functional adequacy for the user poulation envisaged by the ET10-63 project: computer scientists and linguists. It is still important to note, however, that the intended use of the corpus should be a deciding factor in the design of the storage model for the corpus. Also, the data model as presented is relatively extensible, and thus may be looked upon as the embryo from which other applications, with different user requirements, may be built.

In the storage of the corpus the concept of formal adequacy is important. Formal adequacy is composed of two sub-goals which state that the corpus must be stored:

- in the most efficient way possible,
- in a way which ensures that all of the information can be recovered, i.e. none is lost.

The first step towards achieving these goals is to design the database conceptually. The data derived from the corpus, and related encoding programmes, is conceptually modelled using an Entity-Relationship Model (ERM) normalized and then transformed to produce the Relational Data Model (Howe 1989; Codd 1970, 1974; Date 1986a, 1986b). All the concepts that can be modelled in a normalized ERM can be represented in a relational data model, free of redundancies.

II. An Entity-Relationship Model of the Corpus

An ERM collects entities and their relationships. These entities and relationships are then drawn together, and the result is a set of specifications for data storage. These specifications are then collected in an entity relationship schema which is represented graphically (see section C).

A. Entities

Clearly-distinguishable objects which arise from the corpus are called *entities*. In the first instance, five entities for the storage of the monolingual corpus have been developed. These entities are:

- Corpus
- Tokens
- Lemma
- Tags
- Files

We have considered letters as being subservient to tokens which are words.

Entities have properties which are called attributes. The values of an attribute of an entity are called the *domain* of that attribute. To describe a particular entity, it is often not necessary to list values for all its attributes; instead it can suffice merely to provide values for some attributes only which completely characterize the entity (for example, the tag for the Tag entity). Such attributes are called *key attributes*. Collections of key attributes are called the *key* for the entity set. A key is required to be free of redundancy.

To clarify these concepts it is useful to examine the various attributes of the entity sets developed as part of ET10-63 and their keys. The domains of these attributes are fully presented in the Technical section (see IV).

1. Corpus

The data present in the corpus to be represented:

- the *tokens*: words, punctuation, numerals,....

- where they appear (file name and number of sentence in the file),
- their typology (capitalized, italics, bold, ...),
- what allows us to isolate them (left-right context)

The data obtained with the tagger and the lemmatizer are for each word encountered in the corpus:

- the TAG,
- the POS,
- the lemma

No primary key appears in this entity; all the attributes are not free of redundancies. A primary key of the entity Corpus shall be the index of the token encountered in the corpus.

So, the attributes of the entity Corpus are summarized in the following schema where an entity is represented by a rectangle and the attributes by circles; the bold circle is the key:

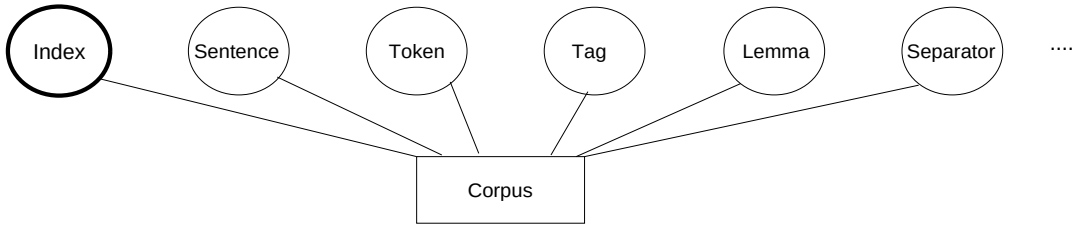

Figure 1: Corpus

2. Tokens and Lemma

The entity Token and the entity Lemma have only one attribute: respectively, the attribute token and Lemma, which are primary keys (Figure 2).

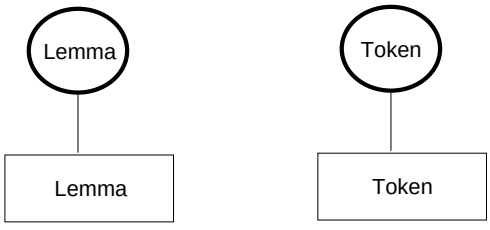

Figure 2: Token and Lemma

3. Tags and Files

Each tag is associated with a part of speech; the tag attribute is the key. Each file is associated with a Unix identifier; the file name has been choosen to be the key (Figure 3).

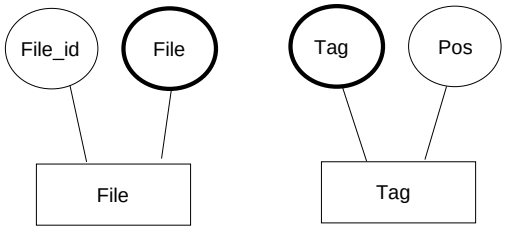

Figure 3: File and Tag

B. Relationships

Entity sets do not exist in isolation. They are usually associated with each other in various ways (Figure 4). For example, the entity Corpus is associated with the entity Tag. Such an association is called a *relationship*. Relationships link entities. If a key has been defined for the entities participating in a relationship then it is obviously sufficient simply to list the key values for each entity representing a relationship, since these identify each entity uniquely. The relationships identified for the monolingual corpus with the different entities developed are:

1. R (Corpus, Token)

The corpus is composed of tokens which are words, punctuation marks, numerals, etc. Each token can appear several times in the corpus, but it must appear at least once in the corpus to be identified as a token. Hence the R (Corpus, Token) is a many to one relationship: each entity over the Corpus entity is related to at most one entity over the Token entity, but each entity over the Token entity is associated with at least one entity over the Corpus entity.

2. R (Corpus, File)

The corpus is divided into files, so each entity of the corpus appears in only one file and each file potentially contains several entities (but at least one). Hence the R (Corpus, File) is a many to one relationship.

3. R (Corpus, Tag)

Each token encountered in the corpus has received a tag, but this tag could be different, depending upon the syntactic context of the token. For example, the word *compte* in

the sentence:

l 'installation compte habituellement deux canaux

is a verb and receives the French tag: VERB3. However, in the sentence:

INMARSAT les exploitera pour son propre compte

it is a noun and receives the French tag: SUBSMS. Similarly each tag may be associated with more than one token. Hence the R (Corpus, Tag) is a many to one relationship.

4. R (Corpus, Lemma)

Each token encountered in the corpus has received a lemma. This lemma is either the canonical form or the token itself depending upon whether the token is a word or not. For a token, several lemmas are possible as the preceeding example shows: in the first sentence, the lemma of *compte* is COMPTER and in the second the lemma is COMPTE. Similarly each lemma may be associated with more than one token. Hence the R (Corpus, Lemma) is a many to one relationship.

C. Schema

The entity-relationship diagram consists of the following 'building blocks':

- an entity is represented by a *rectangle* which carries the name of the table, and by *circles* for the attributes (the bold circle is the key). Each circle is connected to the corresponding rectangle via an undirected edge. Domains are not shown.
- a relationship is represented by a *rhombus* that carries the name of the table, and this is connected to the participating entity declarations by undirected edges.

III. Relational Data Model

With this model built, the ERM designed for the corpus may be translated into a Relational Data Model: the entity set and the relationship set become tables, while the entities and the relationships become a record (or a row) of the table.

Yet while this model suffices for monolingual corpora, the ET10-63 project requires, in addition, to store statistical data retrieved by the comparison of the two monolingual corpora, and also to store the alignment necessary to render the monolingual corpora as parallel aligned corpora. To this end further tables are needed:

The monolingual tables:

- the corpus has been divided in files:
  - the **File** table
- the initial table lists the tokens of the corpus:
  - the **Corpus** table

- the tables built from data provided by the tagger and lemmatizer:
  - the **Lemma** table,
  - the **TAG** table
- the tables that will be built with the data provided by the monolingual statistical program:
  - the **MWU** tables
  - the **Pattern** table

The **Corpus** table is free of redundancies thanks to the following tables:

- the **File** table:
  A File is associated to a numeric (integer) identifier and to a Unix identifier. This is the first association that takes place in the **Corpus** table.
- the **Token** table:
  A token can appear several times in the corpus and so be recorded as such in the Corpus table. The **Token** table contains only one entry per token, irrespective of the number of times it has been encountered. To this unique token is assigned a numeral identifier of integer type. This identifier replaces the string of the token in the **Corpus** table.
- the **Tag** table:
  A Tag is associated with a numeric (integer) identifier. This is the identifier posted into the **Corpus** table.
- the **Lemma** table:
  A lemma is associated to a numeric (integer) identifier. This is the identifier which is used in the **Corpus**.

The bilingual tables:

- the tables obtained by the sentence level alignment program:
  - the **Synchronised-sent** table
- the tables obtained by the bilingual statistical program:
  - the **Synchronized-MWU** table

IV. Technical Description

The data are recorded in a table and index-linked to the other tables via a unique identifier. This identifier is called *reference* and noted *id* in the following section. The table which defines the associations between data and references is called *directory*.

A. Monolingual Tables

The **Corpus** table contains the data. It lists the tokens in the order they appear in the original text.

Each token will be described by:

- its reference in the **Token** directory,
- a reference in the **File** directory,
- a token number, indexed,
- an integer to encode the capitalization of the word,
- a reference in the **Tag** directory,
- a reference in the **Lemma** directory,
- the separator which appears before the token (blank, hyphen, etc.),
- the type setting: italic, bold, etc.

This list is not exhaustive.

Corpus:

token_id	integer, non-null
file #	integer, indexed, non-null
token #	integer, indexed, non-null
capitalization	integer, non-null
tag #	integer, indexed, non-null
lemma #	integer, indexed, non-null
separator	one character, non-null
typeset	one character, non-null
...	

The **File** table allows the identification of the file inside the Unix System. This directory is kept up to date when the **Corpus** table is created.

File: unique

File id	integer
File name	variable-length character string,indexed, non-null

The **Token** directory contains all the lexical units of the corpus. This directory is kept up to date when the **Corpus** table is created.

Token: unique

id	integer, indexed, non-null
text	variable-length character string, indexed, non-null

The **Lemma** directory contains all the lemmas used in the corpus. This directory is kept up to date when the **Corpus** table is created.

Lemma: unique

id	integer, indexed, non-null
text	variable-length character string,indexed, non-null

The **Tag** directory contains all the tags encountered in the corpus, and the corresponding part of speech. This directory is kept up to date when the **Corpus** table is created.

Tag: unique

id	integer, indexed, non-null
text	variable-length character string,indexed, non-null
pos	one character, non-null

The **MWU** directories are built by an external program. A MWU table is built for base-MWUs of length 2 and 3 (Daille, 1992).

Let's examine the table of base-MWUs of length 2:

Each MWU will be described by:

- its reference in the **Pattern** directory,
- a MWU number, indexed,
- two references in the **Lemma** directory,
- the frequency of the MWU,
- the frequencies of each component of the MWU,
- mutual information associated with the couple,
- average of the distance exiting between the two MWU elements,
- variance associated with the distance average,
- average of the number of *main items* occuring between the two MWU elements.

This list is not exhaustive: in particular, it could be interesting to know the morphological inflexions and the orthographical variants of the MWUs which appear in the corpus; this information could be derived using the references in the **Token** directory.

MWU-length2: unique

id	integer, indexed, non-null
pattern #	integer, indexed, non-null
lemma_id 1	integer
lemma_id 2	integer
frequency_MWU	integer
frequency_id 1	integer
frequency_id 2	integer
MI	float
distance	float
Variancy	float
main_distance	float

The **Pattern** directory enumerates all the various linguistic patterns of base-MWUs of length 2 encountered in the corpus.

Pattern: unique

id	integer, indexed, non-null
text	variable-length character string, indexed, non-null

B. Bilingual Tables

The **Synchronized-sent** table allows the alignment of French and English sentences. A French sentence will always be linked to a unique English one. If a French sentence is translated into two English sentences (alignment 1-2), only the index of the first one will be indicated.

Synchronized-sent:

Sent-fr id	integer
Sent-en id	integer
Alignment-type	integer

The **Synchronized-MWU** table allows to align French and English MWU.

Synchronized-MWU:

MWU-fr id	integer
MWU-en id	integer

V. Conclusion

The work outlined here is the formal basis upon which the ET10-63 corpus is being encoded. The data model as it stands meets one of our basic requirements: formal adequacy. The functional adequacy of the database will only become apparent with use. However, as the design has been largely user led there are good grounds for assuming that this goal will also be attained: within the user population envisaged by the ET10-63

project at least.

The role of the database as the storage medium for large corpora seems destined to grow. The British National Corpus and the Spoken English Corpus are two corpora that will be exploiting this form of storage. We hope that in this paper we have outlined the type of development that needs to go into the creation of an adequate storage model for modern text corpora.

References

Codd, E.F. (1970). A Relational Model of Large Shared Data Banks. *Communications of the ACM* 13:6, 377-87.

Codd E.F. (1974). Recent Investigations in Relational Database Systems. *Information Processing* 74, 1017-21.

Daille, B. (1992). *Typology, Composition and Modification of MWUs found in the French Telecom Corpus.* Technical Report.

Date, C.J. (1986a). *An Introduction to Database Systems* Vol 1 (4th edition). London: Addison Wesley.

Date, C.J. (1986b). *Relational Databases: Selected Writings.* London: Addison Wesley.

Howe, D.R. (1989). *Data Analysis for Database Design* (2nd edition). London: Edward Arnold.

Johansson, S., Leech, G.N. and Goodluck, H. (1978). *Manual of Information to Accompany the Lancaster-Olso/Bergen Corpus of British English, for Use with Digital Computers.* Department of English, University of Oslo.

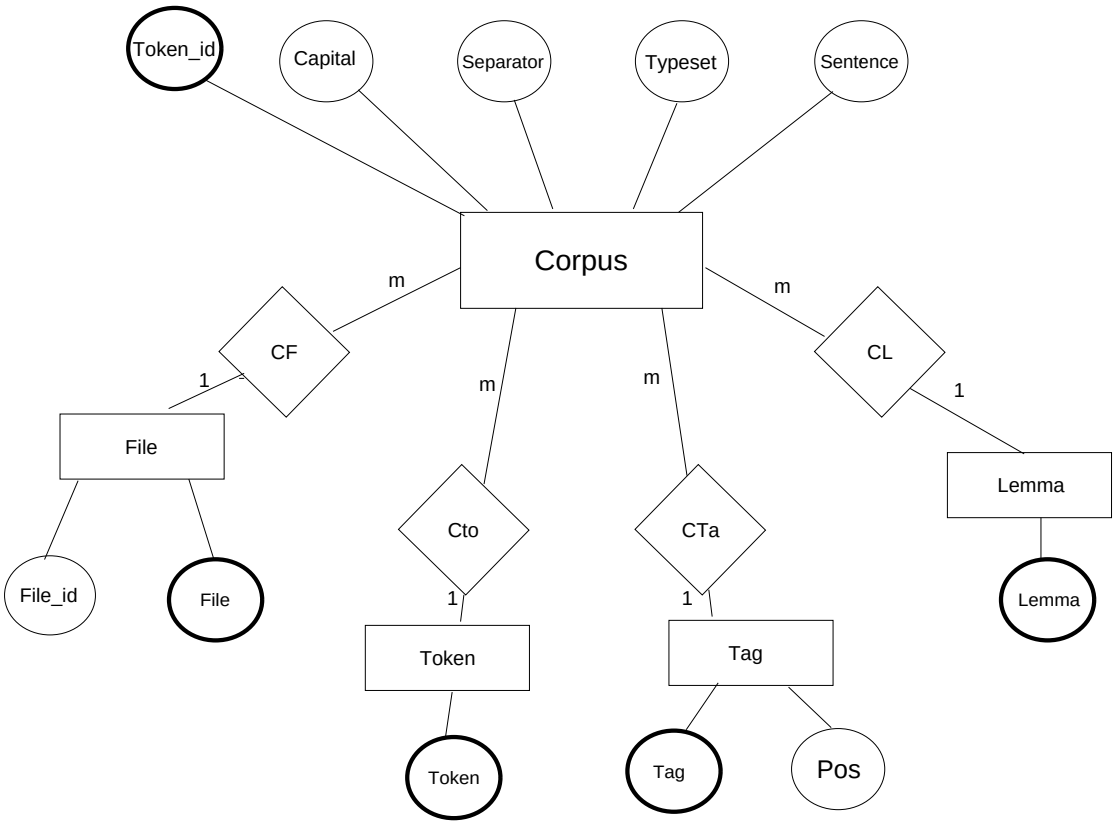

Figure 4: Schema