

THE PRINCIPLES OF TEXT-BASED DATABASES: QUALITATIVE DATA ANALYSIS

Introduction

This document is designed to complement the tutorial on *Orbis* that forms part of the Nota Bene suite of research and writing applications. It is **not** intended to substitute for a careful review of the tutorial, which should take the average user no more than one hour to complete. The *Orbis* tutorial is found within the application itself. To open *Orbis*, Nota Bene offers -- as is common with this program -- a variety of options that include both the mouse and the keyboard.

For mouse users, click the icon in the Nota Bene window above the ruler line representing a pencil and sheet of paper superimposed on a globe. Keyboard warriors need only press F4.

Once *Orbis* is open, click on its toolbar menu for Help, Contents, Tutorial, then follow the tutorial step by step. It will introduce the user to the workings of *Orbis* and walk the user into creating a database or two. This brief document only seeks to present the principles of databases and the mechanisms by which the user can maximize the services that *Orbis* provides by going beyond the basics into an understanding of the principles of databases and the conceptual constructs about research that *Orbis* invites users to consider.

Introduction to Databases

The goal of all databases is the same: to establish a pool of information into which to enter data and from which to draw such data at any time subsequently. The effort is always in entering the data -- consistent with the nature of all computing activity -- *someone* has to enter data for there to be gratification and yields subsequently.

Databases normally involve entering values (*data*) associated with particular types of information (*fields*), which, taken as a whole, take on a particular structure (*form*). Thus, in a commercial business, one would see database applications built to suit that business' needs to have a database template or form into which one would enter the price for a particular type of product, in a particular store, along with the stock number, the manufacturer, its telephone number, the name of the contact person, and so on. The *form*, or data-entering template, would consist of *values* entered for each *field* and might look something like this:

| | |
|----------------|-----------------|
| Product | Pizzazz shirt |
| Price | \$25.00 |
| Vendor | Ralph Mauren |
| Store # | 24 |
| Provider | Harry Knowitall |
| Provider phone | 212.555.1234 |

The data, the information, might come from different databases. There might be a database of providers with their information. There might be another one of vendors. A form would bring in data from these different databases which contain one or more logical relationships that, in their aggregate, make sense. Thus, we talk about *relational* databases. In addition, the form has a special and fixed structure, that is, a particular set of fields, each of which has a value assigned or entered. Thus we talk about *fixed-field* forms or *fixed-field* structures. Most databases follow the organizational principles just described.

Hypertext and Free-form Structures

The Nota Bene suite includes *Ibidem*, a bibliography management application that follows the organizational principles of a fixed-field database format. Nota Bene's suite includes another application,

Orbis, designed as a form-free mechanism for entering information and retrieving the data in a very swift manner. In other words, *Orbis* is free of any templates because it is form-less. It follows similar principles as the information one would retrieve from the World Wide Web, even though it pre-dates the Web by many years. *Orbis* is based on hyper-text principles -- a user can jump from one word or term and land on a broad range of information dealing with related topics. Think of the possibilities: the concept of “inheritance” can lead to information on peasant landholding practices, or to family size, or the economic well-being of a society, or the biographical details of a family’s empire, and so on.

Because *Orbis* is a database application based on a form-free structure, the user does not need to be concerned with what fields are appropriate, with whether the fields strike a balance that is both comprehensive and flexible. Furthermore, building a fixed-form database structure usually presents the designer with trade-offs between templates’ comprehensiveness and functionalities. And more often than not, designing each template involves a great deal of experimentation and starting anew as the designer “plays” with different forms until the appropriate balance is achieved. In *Orbis*’ form-free structure, there is no template to design: virtually all text entered into any document created in or converted into Nota Bene automatically becomes a potential candidate for incorporation into an *Orbis* database.

The Structure and Logic of *Orbis* Databases

Even if *Orbis* is not based on structured templates for its data retrieval and relational power, it nonetheless has a structure and logic, which, once understood, permits the researcher to make full use of its potential. To explain the structure of *Orbis*, I will employ metaphors and analogies from our behaviors in the normal physical world in which we operate when we write.

File management: the drawer

Orbis works on the basis of a set of files which are responsible for indexing, maintaining, and presenting the data. You can have as many *Orbis* databases as you wish. Think of an *Orbis* database in the context of a specific desk or file drawer to which you have given a name. In other words, in your mind, write a name on a small card and paste that card on the drawer’s faceplate. This is the equivalent to an *Orbis* database name. To give this metaphor a certain amount of real-world meaning, give this drawer the name of “Research Notes.” This drawer -- this database -- will hold all sorts of information. The information consists of words. And the words are located in regular Nota Bene files.

Files: the documents

As with any word-processing application, words exist (virtually) in documents we call files. Normally, when we type a document, for example, a research paper, or a letter, or a memo, or some jotting, the words have meaning only within the confines of the document. Once we save a file, the words in those documents bear no relationship to any other words or concepts. Furthermore, over the course of time -- often not a very long time -- the *location* of those words, that is, those files becomes a challenge in itself. As documents proliferate in the computer’s hard disk, the management and the human memory for managing those files becomes increasingly . . . well, less manageable. Let’s come back to the metaphor of the file drawer labeled “Research Notes” to illustrate this point. In this metaphor, imagine tossing a document -- with words in it, of course -- into the drawer; then toss another one, and another one, and so on. Pretty soon, the drawer “Research Notes” will be holding a lot of words in a lot of documents. The advantage of this much planning is that you can be pretty certain that all your notes are contained in this drawer. But what do those notes contain? Which is the document with the terms and issues you are interested in finding *now*, when you need them? Indeed, which *are* the several documents containing the terms and issues which you are interested in re-visiting? Who can remember the files’ names?

The scenario depicted above illustrates the dilemma with word-processing: a document’s “life” is difficult to extend beyond the period of time in which the document is written, that is, the time during which the data -- the words -- are entered.

Orbis and the Irrelevance of File Structure

Because Orbis does not require the user to know the names of files where terms and relations may be located, it acts as a virtually invisible application to the user. To act in the most useful and automated fashion, Orbis needs to be informed of only two things: 1) the folders in which the files are located, and 2) the file extension (the last 3 letters following the period after the file name) of the files. Other folders and file extensions can be added after initial setup, if necessary. By giving the same extension to files representing certain types of documents, users benefit from knowing that Orbis keeps track consistently of both new and changed files. Orbis calls this method of identifying files as a “mask,” that is, files with a given extension are masked for inclusion in the database and those without such an extension, are left out. A user may want to give the extension “not” or “res” to files with research notes; the extension is entirely up to the user’s discretion, but it should be easy to remember and should be reserved only for the appropriate type of document so that no unintended documents appear in the database.

How does Orbis work? Let’s resume the file drawer analogy to understand Orbis’ processes. Orbis takes images of all the words in the documents, virtually prying them free of the constraints associated with individual files. Thus, words are indexed into one database that grows continually as documents are created and saved. In this fashion, the user’s efforts in writing documents take on a dynamic and continuous usefulness. Once liberated from any given file or document, Orbis is used to inquire into the relationships that may exist among terms. For example, a user may have taken notes on family and household which contained information on liturgical material as it related to procreation and which authorities implemented through institutional mechanisms. Information of this sort, containing similar or different terms -- strewn about any number of files -- is brought together, on the basis of user’s needs and demands. This is the electronic equivalent of going into the file drawer and pulling out all the requested words that reflect the requested relationship, then bringing up the paragraphs or the entire contents of the relational files in which the terms appear in order that the user review and employ in further writing. The user could request of Orbis for the texts that respond, for example, to the following request: (social and control) and (number and children). Orbis would present to the user all the text, along with information regarding the files’ names and their locations. On a simpler level, a user may wish to search for only a single term. In either case, the user is then free to select the amount of found text to incorporate into a new document, for example, into a research paper, a thesis, an article, or any other type of document.

In addition to searching for terms users transcribe, Orbis provides hypertext capability by allowing users to jump off any term in the found set of documents and lead them into other documents with related information. All found documents can be saved into a single compiled file containing the set of found and desired “hits.” This has the advantage of providing users with convenient clusters of data for subsequent analysis and use.

Strengthening conceptual power

Orbis does not limit users to only the exact words that appear in their documents. Orbis adds power to the users does this in two ways. First, it encourages users to tag their texts with their own terms so that they can expand further particular concepts or ideas. These are called labels and the chosen terms must be preceded either by the special characters @ or #. Second, Orbis contains a synonyms feature, which acts as one’s own thesaurus containing the user’s analogous words or terms that, in the user’s mind, should be linked. For example, “authoritarianism” is a concept that encapsulates other, equally meaningful terms. These terms might include: “torture,” “military,” “dictator,” or other terms that the user encounters concretely or which the user conceives of in the process of taking notes. With this, the user can expand dramatically the analytical power simply by searching for “+military” (that is, a plus sign is written immediately before the desired search term). Passages of documents prepared by the user would then be

presented, including those in which the term “military” does not even appear, but which are, in the user’s conceptual framework, associated with the military. By using Labels and Synonyms, *Orbis* expands significantly its search and retrieval capacity. The user is then free to view and select information apt for the purpose.

Summary and Conclusion

Orbis is a free-form database application that takes words as the unit of analysis. It serves two purposes: 1) to depict relationships among terms strewn about the computer’s hard disk, and 2) to present the finds resulting from the user’s queries. The “finds” or “hits” can then be used in new documents or in compiling sets of found information about particular issues. By employing @ or # Labels and the Synonyms personal thesaurus mechanism, users can significantly expand their search and retrieval efforts in the process of conceptualizing specific areas of their research and their research notes.