Database Querying with Direct Manipulation

by

Mark Boyle

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at

Dalhousie University

Halifax, Nova Scotia

September, 1997

0-612-24732-5

DALHOUSIE UNIVERSITY

FACULTY OF GRADUATE STUDIES

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "Database Querying with Direct Manipulation"

by Mark Boyle

in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dated: September 5, 1997

External Examiner [redacted]

Research Supervisor

Research Supervisor

Examining Committee

DALHOUSIE UNIVERSITY

DATE: September 1997

AUTHOR: Mark Boyle

TITLE: Database Querying with Direct Manipulation

DEPARTMENT OR SCHOOL: Interdisciplinary Studies

DEGREE: Ph.D. CONVOCATION: October YEAR: 1997

[redacted]

Signature of Author

Dedication

This dissertation is dedicated to my wife Mary who, through strength of character, showed me the way.

Table of Contents

List of Tables and Figures

Abstract

The goal of the study was to investigate the application of direct manipulation interface techniques in the area of database querying. Past research had explored this in a small scale manner. The current study examined whether this earlier research would scale up to a real-world situation. Two specific variables were examined; structure of the search factor list and the implementation of a Boolean 'Or' query facility. The search factor list varied between a scrolling list design and a windowing design. The Boolean 'Or' facility varied between direct and indirect designs. Data were collected from 72 graduate students on performance and preference measures. Performance was measured through time to complete the task and correct responses. Preference was measured by a short series of bipolar adjectives. Results were equivocal. When performance was measured by time to complete the task no significant differences were found for either variable. However, when performance was measured by the number of correct responses, then a significant effect was found for the implementation of the Boolean 'Or' facility. Subjects performed significantly better when using the direct design versus the indirect design. Although no significant main effects were found for preference across the two variables a significant interaction was supported.

Acknowledgments

As with any large project, this one was not completed alone. I would like to thank all the committee members: Andrew Peacock, Mike Shepherd, Carolyn Watters, and Jack Duffy for their help and support throughout the endeavour. I would like to especially thank Jack Duffy for without his help and support this project would not have been finished. His contribution cannot be overestimated. Thanks must also be directed towards my wife Mary for her constant support and sacrifice.

Chapter 1 - Introduction

Introduction and Rationale

This research evolved from the author's work on an information system for a local company. The basic requirement was to build a system in which economic development information could be stored and then retrieved by end users. Economic development information in the database consisted of summary population parameters (e.g., persons per household), and demographics (e.g., unemployment rates) A complete description of the database and the associated fields is given in Table 4. One prominent function of the system would be as a site location aid for interested parties (e.g., government agencies, site location consultants, businesses, etc.). The users of the system would be experts in the field of economic development but would, for the most part, be naive to the technology of information storage and database querying. The specifications for the system called for a large repository of economic information with multiple entry points for the end user. These entry points, or interfaces, would vary based on the goal of the user and the nature of the information required. In some cases the user would simply specify an entity (e.g., municipality) and the desired information (e.g., summary population statistics), whereas in other cases the user might take a more traditional querying approach. That is, he or she would specify values for certain attributes and receive a resultant set that satisfied the search. In the parlance of the project, these attributes were referred to as search factors. It was this searching or querying type of database interaction that was the focus of the research. Specifically, the author was interested in examining the process of database querying and database query languages and how this process might be improved.

"A query language is a special-purpose language for constructing queries to retrieve information from a database of information stored in a computer" (Reisner, 1981, p. 14). Query languages come in many forms, ranging from those based on an English like syntax (textual-based) to those rooted in graphics. Research on query languages and database querying has been undertaken since the early 1970s. In the nascent stages researchers were focused on languages based on a terse English-like approach. These languages required that the user be proficient in database design as well as database querying. During the mid seventies, as computing technology became more affordable, researchers began to focus on easing the knowledge requirements for database interaction. They began to explore alternative ways for users to extract information from computer databases. At this time they also realized that users were less and less likely to be computer experts and more likely to be experts in some other domain, usually the domain of the information contained in the database. Early attempts revolved around relaxing the syntactical requirements and making the languages more English-like. A number of researchers explored 'natural language' interfaces (c.f., Harris, 1977; Kelly & Chapanis, 1977), those in which the interaction resembled a conversation between two persons. Others concentrated on highly structured approaches; one of the best known of these is SEQUEL, the precursor to SQL (Chamberlin, Gray, & Traiger, 1975). Still others believed this textual approach to be a dead-end and looked for alternative interfaces. Query-By-Example (QBE) was one such early attempt (Zloof, 1975). Here, the authors removed the textual basis from the query language and replaced it with a table based paradigm. Some promising results were reported from these endeavours. Evolving from this table based approach, and a desire to further eliminate the textual requirement, was the graphical approach to the querying interface. In this area, the investigators attempted to reduce or eliminate problems with the process by tying the querying process to a graphical representation of either the query, the database, or both (c.f., Michard, 1982). In the late eighties and early nineties research continued on the graphical approach to database querying. At this point the interface to computer operating systems had changed drastically from the time of the early investigations of querying and some researchers sought to incorporate these new operating system features into the database area. In one case research was carried out on the use of Direct Manipulation techniques as a means of increasing the efficacy of database querying (Ahlberg, Williamson, & Shneiderman, 1992). Again some promising results were reported. A summary of these studies is presented in the next section.

This direct manipulation approach is the focus of the present study and it follows from work performed at the University of Maryland Human Computer Interaction Laboratory. Briefly, direct manipulation involves the use of screen based graphical objects as a means of communication between the user and the system. As mentioned earlier some existing research at the University of Maryland had pointed to promising results when using this approach for database querying. However, the research had looked only at very small scale systems, those in which the number of searchable attributes was fixed and less than five or six. Additionally, these existing systems restricted the search logic to Boolean *'Ands'*.

The current research attempted to investigate how this basic approach to database querying could be extended to more real-world problems. Specifically, two factors were studied in the current investigation. Whereas in the previous studies users were restricted to searching the database on a finite and small number of factors, the test system allowed for a user defined list of search factors that could vary in length and content. The investigation examined how to extend the interface in this way. Two approaches were studied. In one, the user could add to a scrolling icon list the search factor that he or she wanted to use in the search. This was referred to as the scrolling implementation. In the second, the factors were chunked into sets of four with additional controls for navigation between sets. This was referred to as the windowing implementation. A comparison between the scrolling interface and the windowing implementations constituted the first independent variable, **Search Factor List Type** or simply **List Type**. Figure 1 shows the list setup used in the two implementations. On the top is the scrolling implementation. In this condition users added search factors in a linear fashion. As there was only enough room to display four search factor icons, a scroll bar control was displayed when the user added more than four factors. The user could then use this scroll bar control to move forward and backward in the list. Below the scrolling illustration is the Windowing implementation. Here, the user could add up to four search factor icons to the list. When more than four factors were required then the user could add factors to different groups or sets. The user could then navigate between sets by using the control to the left of the search factor list.

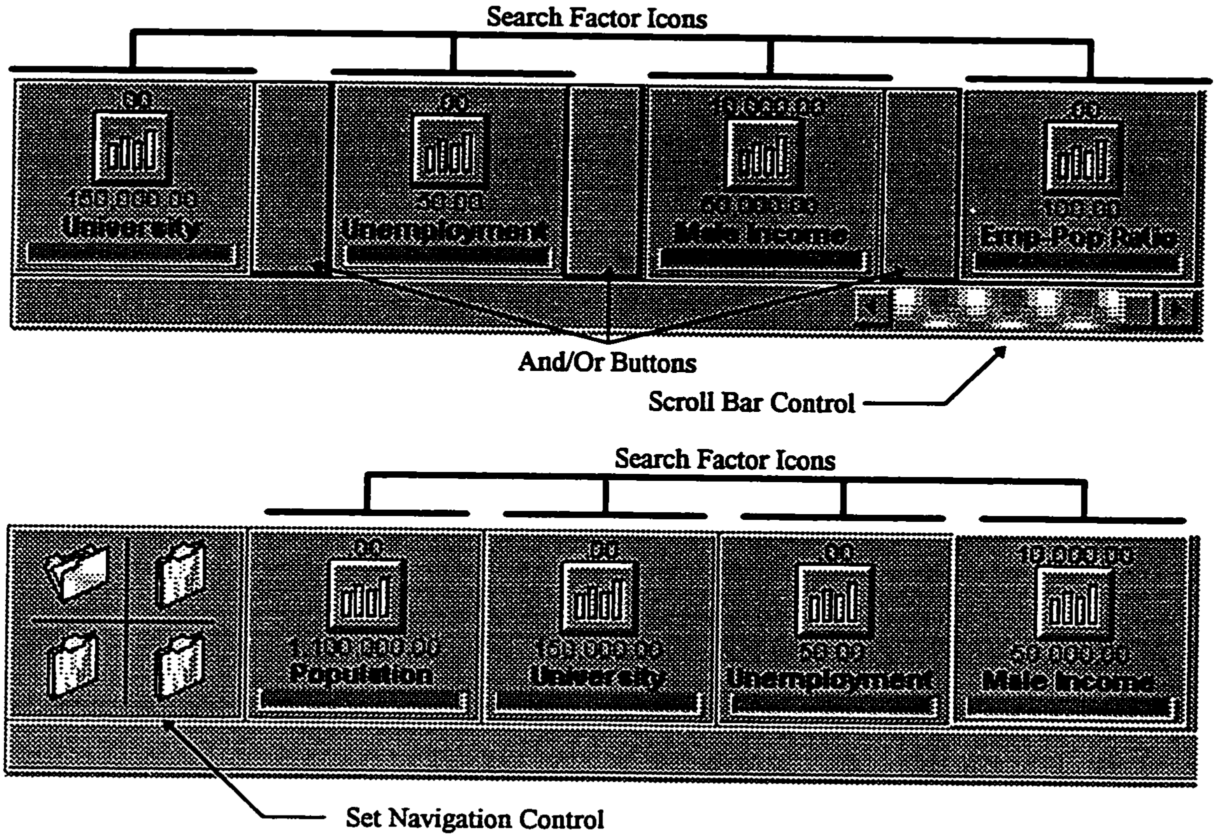

Figure 1: Example of the two implementations of the Search Factor List Type variable. The Scrolling implementation is shown first followed by the Windowing implementation.

The second extension involved the addition of the Boolean *'Or'* operation to the interface. Many systems have not allowed for this type of search, relying instead on multiple searches to emulate this capability. It was felt that any system that is to function in the real world would need to have a more direct implementation of this capability. Therefore, the second independent variable for the study was the Boolean *'Or'* implementation with levels being direct and indirect. This variable was referred to as **Boolean Operations**. In the direct implementation users would form queries with Boolean *'Or'* connectives in response to a question. The manner in which the user performed this operation differed depending on which List Type variable was in effect. In the scrolling implementation, the user single clicked on the gray vertical buttons between the search factor icons to cycle between an 'And' operator and an 'Or' operator. These are labeled And/Or buttons in Figure 1. In the windowing implementation the user added search factors to different groups or sets to form a Boolean 'Or'. That is, the groups of factors were connected with an 'Or' operator. The indirect implementation closely emulated earlier work in that the user would basically carry out multiple intermediate searches to respond to questions. This condition had the added functionality of providing the user with a place to store the results of these intermediate operations. Figure 2 shows the dialog box used to store the responses. Users could utilize this dialog to collect results from multiple searches.

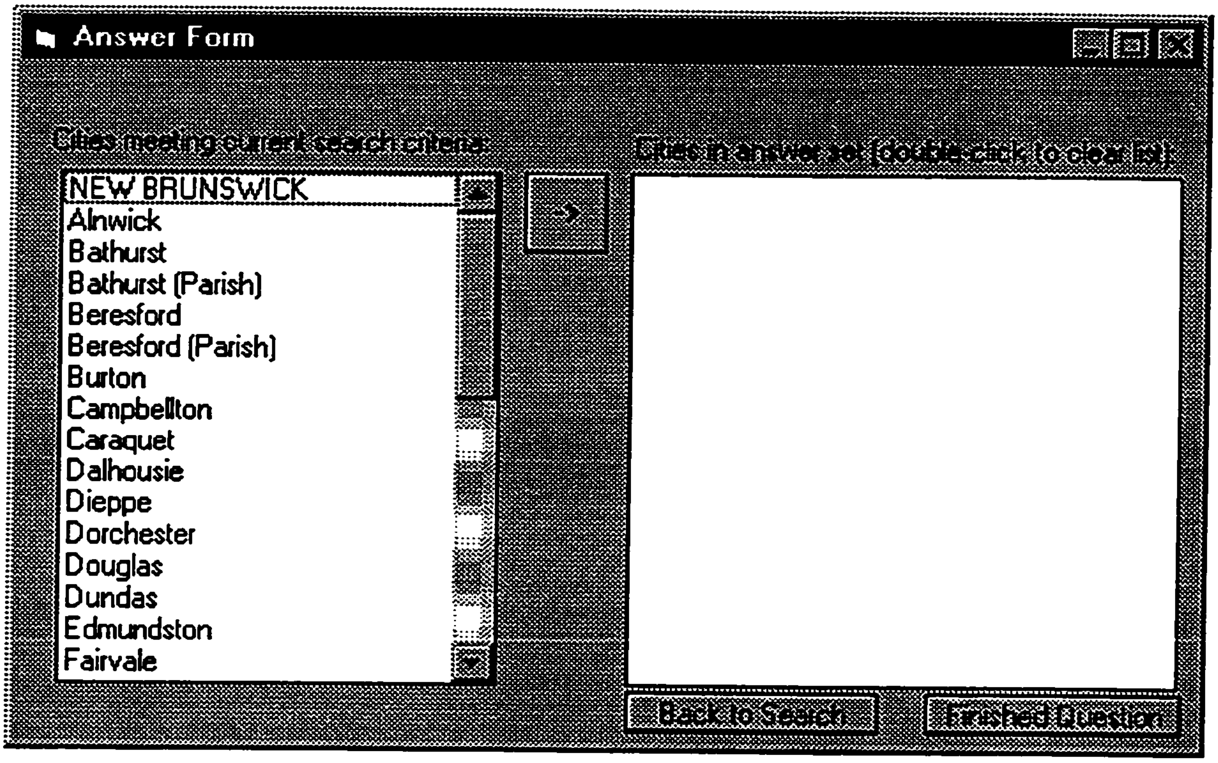

Figure 2: Dialog used to collect responses in the experimental system.

The combination of these two variables resulted in an experiment with four distinct conditions requiring an experimental querying system with four distinct interfaces. In the first condition subjects employed the scrolling type of list and the direct implementation of the Boolean operation (Condition SD). In the second condition subjects again employed the scrolling list but were required to perform multiple searches to respond to questions requiring an 'Or' operator (Condition SI). Condition three utilized the other type of list, windowing, and the direct implementation of the Boolean operation (Condition WD). That is, users added factors to different groups in response to a question requiring an 'Or'. In the final condition users worked with the windowing implementation of the list and were required to perform multiple searches to answer questions with 'Or' responses (Condition WI). Each subject participated in only one condition. Figure 3 shows the experimental setup.

	Search Factor List Type	
Boolean Operations	*Scrolling*	*Windowing*
Direct	SD	WD
Indirect	SI	WI

Figure 3: The experimental design for the study.

Chapter 2 - Review of Previous Research

The following sections on database query language research cover the work performed on query languages for the last 30 years. Research literature on this topic has usually taken one of three tacks. In the first, the author or authors present information on a new query language system they have developed or improved upon. In some cases these articles have included an experimental evaluation of the new system. In the second, a system or multiple systems are subjected to a human factors evaluation in an attempt to empirically reveal the relative strengths or weaknesses of a system. In the third, the author presents an overview of the existing literature on query languages and, perhaps, suggests some areas for improvement. The literature review follows a chronological path through existing research and where possible presents articles dealing with systems for querying databases, continues with human factors research, and concludes with articles devoted to summarizing the research area.

Where possible, examples are provided of queries written in the various systems. For this exercise the queries are based on a simple two table database containing information on cities in Canada. The graphical representation of this sample database is shown in Figure 4. The simple database contains tables of basic information and demographic information. These tables are linked with a common ID code key field. Users could retrieve information on city demographics from the database.

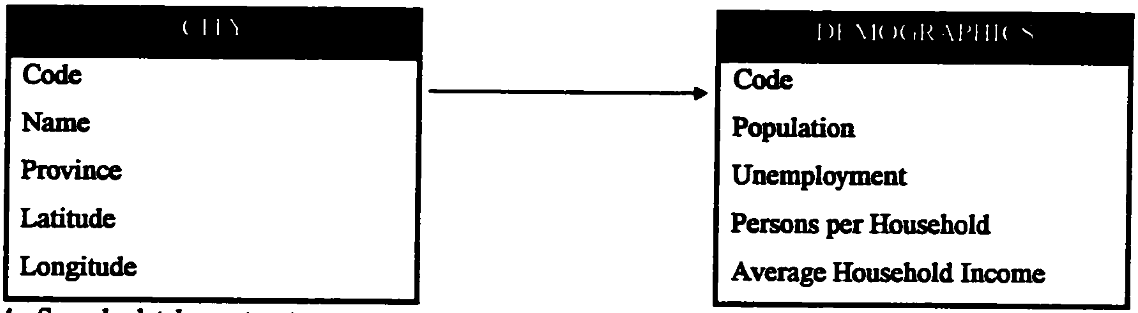

Figure 4: Sample database structure

Database Query Language Research: Pre 1985

A study of the relevant literature reveals a dearth of research into query languages until the mid nineteen seventies. There are some references to articles appearing before this time; however, they are very few (c.f., Childs, 1968 cited in Boyce, et al., 1975; Senko, et al., 1973). One watershed event in the study of query languages occurred in 1975. That year, at the National Computer Conference in the United States, a number of researchers presented work on query language design and use. It appeared that two events precipitated this change in direction. First, the introduction of the minicomputer in the early seventies put the power of computing into the hands of many small and medium sized companies and these companies were looking for a way to manage their data (McLeod & Meldman, 1975). They specifically wanted to place access to the data into the hands of those who could best utilize it, namely the decision makers who were not necessarily computer experts. The second event was the introduction of the relational model of databases by Codd (1970). This relational approach made available a very powerful database environment to the decision makers. Powerful in the sense that it had the following advantages over existing systems: (1) a simpler data model resulting in systems that were easier to use and maintain, (2) enhanced sharing among users as well as enhanced database security, and (3) easier expression of retrieval requests (McLeod & Meldman, 1975). The problem was, of course, how to provide a conduit between the naive database user, the decision maker, and this powerful database engine.

In the first of a series of papers presented at the 1975 National Computer Conference McLeod and Meldman (1975) describe their idealized system, made possible by the advent of minicomputers. They state that their system "... provides a 'naive' user interface, to allow nonprogrammers to deal routinely with a database without the aid of a programmer..."(p. 397). The authors listed the four different types of query languages available at the time in the following manner. At the most basic level, access to the data is provided at the 'element-by-element' level where access is restricted to a single record or tuple. Next, they described the relational languages, relational algebra and calculus, that allowed operations on sets of records. The final two approaches described a more user-oriented view, first, mapping type languages and second, natural language interfaces. They concluded their discussion of query languages by asserting that database systems should provide a querying mechanism based on all of the approaches and that it is extremely important that mechanisms be provided at the mapping and natural language level if both casual and naive users are to benefit from the system. The authors then described their implementation of the above concepts in a system named RISS (Relational Inquiry and Storage System). This database management system was based on the relational model of data storage (Codd, 1970). The system provided access to data at two levels, the programming level and naïve user level. The programming interface allowed access at the record or tuple level. The naïve user interface was based on a mapping approach.

In their paper on relational database technology, Mylopoulos, Schuster, and Tsichritzis (1975) classified the type of user into three groups: the application programmer, the technical person, and the casual user, each of whom had their own individual concerns and needs. For example, the base or programmer level requires significant flexibility and power. Users at this level are experts in the operation of the database. Technical personnel, such as lawyers and engineers, require an interface that affords them the power to retrieve requisite information without relying on help from the programmers. Casual users on the other hand require a simple interface, one that is based on a natural language dialog. To achieve ease of use the authors developed a natural language interface based on semantic nets. Their goals were (1) to remove the requirement that the user know how the information was stored in the database and (2) to allow the user to converse with the database in an 'English-like' manner. At the time of the report a rudimentary system had been built and testing was beginning. It was, however, unclear if the authors had any success with the project.

A system that has survived the test of time was next described at the conference: the INGRES relational database system. In their paper, Held, Stonebraker, and Wong (1975) described the query language QUEL used to access data in the system. QUEL was a calculus based language modeled after an earlier language called Alpha. It was considered an improvement over the earlier language, Alpha, since it incorporated some important changes, the most noteworthy being the inclusion of keywords as statement delimiters. An example of a query in QUEL provided by the authors is shown in Figure 5.

```
RANGE OF C IS CITY
RETRIEVE INTO W(C. NAME)
        WHERE PROVINCE = 'Nova Scotia'
```

Figure 5: Example of a QUEL query.

Chamberlin, Gray, and Traiger (1975) described the SEQUEL (Structured English Query Language) system for database manipulation and management. They stated that SEQUEL is 'a data sublanguage based on English keywords and intended for interactive problem-solving by users who are not computer specialists.'(p. 425). They also provided some examples of queries written in the SEQUEL language. A SEQUEL query to retrieve all the names of the cities in the province of Nova Scotia is shown in Figure 6.

```
SELECT Name
FROM City
WHERE Province = 'Nova Scotia';
```

Figure 6: Example of a SEQUEL query.

The above query would return a list of names for cities whose province is Nova Scotia. This was definitely a step in the direction of easing the burden on the user with respect to finding or retrieving data from the system. The authors go on to explain and demonstrate how mutli-table queries can be submitted.

In a follow-up article, Chamberlin et al. (1976) described the next generation of the SEQUEL system called SEQUEL 2. From a series of human factors tests carried out on the original implementation a number of changes were made to the query facility of the system. SEQUEL 2 provided a full set of query facilities for the naive user; however, it continued the requirement of learning of a complex sublanguage for database query. In another paper concerning the SEQUEL system Astrahan and Chamberlin (1975) described the language as being based on neither Codd's relational algebra nor relational calculus, but as "a nonprocedural language which does not make use of quantifiers or other mathematical concepts; rather, SEQUEL uses a block structured format of English key words ..."(p. 580). Figure 7 shows a more complex query, one involving nesting, written in SEQUEL 2. This query would retrieve the names of the all the cities in the database with a population greater than 50,000. The nested subquery is required since data from two separate tables are needed to satisfy the query (See Figure 4).

```
SELECT Name
FROM City
WHERE Population =
        SELECT Population
        FROM Demographics
        WHERE Population >= 50,000
```

Figure 7: Example of a SEQUEL 2 query with nesting.

In earlier work, Boyce, Chamberlin, King, and Hammer (1975) presented their alternative for database querying as a database sublanguage called SQUARE (Specifying Queries as Relational Expressions). It was intended for use by nonprogramming professionals who might want to interact with large databases. The authors attempted to design a query language that would overcome the problems associated with the relational algebra and relational calculus approaches. Specifically, they wanted to remove the procedurality requirement of an algebra-based approach in a manner similar to a calculus-based approach; however, they also wanted to reduce the mathematical sophistication requirement of a calculus approach. Boyce and his colleagues were concerned with two problems, first, algebra-based languages operated much like other computer programming languages and required some familiarity with this type of problem solving system, and second, calculus-based languages required users to form variables and employ mathematical quantifiers to retrieve data. In their words, "SQUARE enables users to express queries in terms of the natural primitive functions used by people to find information in tables, such as looking up the value in a column and finding the associated value in another column."(p. 622). The difference between SQUARE and SEQUEL was the use of English like keywords in SEQUEL versus a more terse approach in SQUARE. Figure 8 shows a query that would retrieve the names of the cities in Nova Scotia. In English the query reads 'from the table City retrieve name where the province is Nova Scotia.'

```
        CITY        ('Nova Scotia')
   NAME       PROVINCE
```

Figure 8: SQUARE query from Boyce, Chamberlin, King, and Hammer (1975).

Reisner, Boyce, and Chamberlin (1975) presented the results of a usability study of both SEQUEL and SQUARE. In that study, utilizing undergraduate and graduate students (N = 64), they sought to discover whether data sublanguages, or query languages, could be used by the non-programming professional, what differences existed between the two languages in terms of usability, and what sorts of errors were users likely to commit when using the languages to query databases. In the experiment two groups of subjects, programmers and non-programmers, were taught either SQUARE or SEQUEL in a classroom setting, thus resulting in a 2 x 2 factorial design. The subjects participated in 12 to 14 hours of instruction and were then tested on various features of the languages.

Testing involved simple queries as well as complex ones. Analysis revealed that programmers did significantly better than non-programmers, regardless of language. Results also showed some increase in scores for the SEQUEL group as compared to the SQUARE group for the non-programmers. It appeared that making the language more English like and less mathematical resulted in an improvement in query performance. This result was particularly true for non-programmers.

In a related project at the time, Zloof (1975, 1977) introduced his Query-By-Example (QBE) system. Zloof's approach was to remove the linearity of existing text-based query systems and to replace it with a 'two-dimensional' system reflecting the underlying nature of the relational database. His system simply required the user to fill in a skeleton table with examples and constant values to submit a query. In the 1977 paper he describes how the system could be used for expressing everything from simple single table queries to complex queries containing Boolean expressions. Figure 9 shows an example of a query to print the names of cities in Nova Scotia.

CITY	CODE	NAME	PROVINCE	LATITUDE	LONGITUDE
		P. Windsor	Nova Scotia		

Figure 9: QBE query from Zloof (1975).

Thomas and Gould (1975) undertook a psychological or human factors evaluation of Zloof's QBE system. They described query systems as belonging to one of three groups: those based on a natural language dialog between the user and the system, those based on a English-like formal command language (e.g., SEQUEL), and those that do not appear English-like, but that rely on some other mechanism to aid querying, such as QBE. The study examined query performance of non-programmers on a number of factors: time to learn the system, accuracy, confidence, and time in producing queries from English language prompts. The subjects were 39 high school and college students ranging in age from 16 to 24. They reported that the time required to learn the language was very short (approximately 2.5 hours) and was probably the shortest training time of any existing system. Subjects' error rates increased as the complexity of the query increased, as would be expected. One interesting finding was that subjects had little problem differentiating between the Boolean *'And'* and Boolean *'Or'* in formulating queries. In fact, the authors stated that this is one of the 'positive characteristics' of the system.

In a paper presented to the conference on Very Large Databases, Thomas (1977) addressed some of the psychological issues involved in database usage. He began by expounding on the virtues of a human factors approach to system design rather than the intuitive approach. Thomas examined some of the early studies in database management (c.f. SEQUEL/SQUARE and QBE studies) and provided an overview of the pertinent psychological issues. These issues included: user input, information output, and characteristics of the interaction between the user and the system. Thomas' review of the existing database studies came to the following conclusions. First, it was possible to teach a database query language to naive users. Second, from a human factors perspective, it was advantageous to test a query language in a paper and pencil environment before the full implementation was built. In fact, doing so may elucidate some future problems. Third, one major problem area was the specification of table and attribute names. Users would often incorrectly spell these components of the query, thus, leading to mistakes. Fourth, there was a correlation between the probability of an incorrect query and a number of user characteristics, namely, user experience, time to submit query, and confidence. Fifth, user's sometimes take a phrase by phrase translation approach in converting English language questions to the appropriate query. This can be especially troublesome in the area of conjunctive and disjunctive operators.

Michard (1982), citing some of the aforementioned problems with existing query languages, set out to develop a facility based on a graphical rather than textual approach. The author gave a number of reasons why a new query language was needed. This list included the increased use of microcomputers by people in business , especially "naive" users. The fact was that these naive users had little time to devote to the learning of a complex query language and that they also lacked an opportunity to practice their querying skills. Also, Michard pointed to what he considered the poor results obtained from the existing query tools, namely QBE and SEQUEL. He cited research showing long learning times and mediocre performance. In an attempt to combat these problems Michard used Venn diagrams as the basis for his interface, citing the widespread use of such diagrams to teach Boolean logic within the school system. Along the way he produced a list of design principles to which query languages must adhere. The list included the following:

(1) a query language must be non-procedural. That is, the language must not require the user to specify how the information will be found, but only that he or she specify the criteria for retrieval.

(2) the order of the criteria for selection must be irrelevant.

(3) feedback is an important component of the system. The user should know immediately, the effect of his or her query.

(4) some assistance must be provided to the user in the formulation of Boolean queries. Michard states that a "well known" problem with query languages is the difficulty users have in forming correct Boolean logic for their queries.

(5) some sort of menuing system must be employed in order to reduce the clerical type of errors associated with a text based interface.

Michard developed his Graphical Query Language (GQL) system under the above guidelines or principles. A representation of the system is shown in Figure 10.

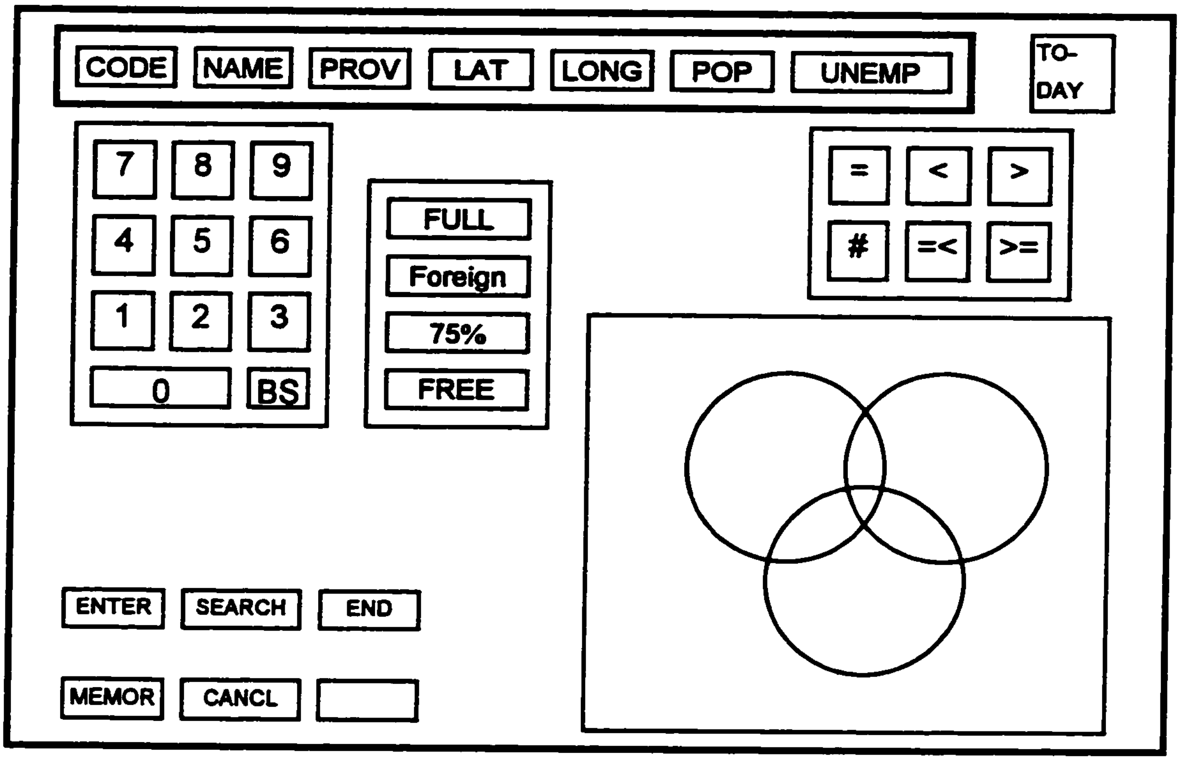

Figure 10. User view of Michard's GQL system (p. 283).

An operator using the GQL system would be presented with the above screen, save for the three interlocking circles in the lower right-hand corner. The user would click on one of the field name buttons and set the criteria (i.e., UNEMPL <= %10). The system would then draw a circle representing the result space for the given criterion. The user could then add another search factor and set its value. Again the system would draw a circle to represent the result space. The user could add up to three factors at a time. To specify the Boolean operators, the user would click on different sections of the interlocking circles. Figure 11 shows a simple "AND" operation involving two factors.

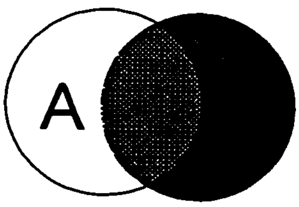

Figure 11: Example of Anded factors in GQL.

The user would click on the lightly shaded region to specify A and B. To combat the problem of an extremely complicated graphic output the system was restricted to displaying a maximum of three circles. In order to use more than three factors in a search a user would have to take an iterative approach. That is, the user would first create a query with three of the search factors. At this point he or she would click on the memorize button and the three factor query would be replaced with a single circle representation. The user would then be free to use this single circle representation in further Boolean operations. Michard tested his interface with 12 college level subjects and, although his statistical sensitivity was very low, found promising results. He tested his GQL interface against a similar one that varied only in the use of Venn diagrams. In the comparison interface users specified factors and criteria the same way but had to build queries in a standard, text-based way, one that involved the explicit use of parentheses and Boolean operators. The results favoured the GQL interface with subjects displaying far fewer errors. In an analysis of the errors the author noted that the *'And'* operator is the easiest one to employ and the majority of the errors involved the misplacement of parentheses in Boolean expressions.

Ehrenreich (1981) presented a number of design recommendations for query languages based on a review of the existing human factors literature. He reviewed both natural language approaches and formal query language approaches. On the formal side he reviewed literature that dealt with a number of pertinent issues. These included: ease of use of the query language, problems emanating from spelling and grammatical errors, errors of a semantic nature, and term specificity. He provided a number of recommendations for the development and design of a query language, the most pertinent being the introduction of layering. Layering refers to the use of a multi-tiered interface that shields the naive user from the complex power of the interface but allows the experienced user to employ that power when necessary or desired.

Small and Weldon (1983) undertook a comparison of formal retrieval languages and natural language. In their study 20 volunteer subjects, recruited through a local paper, queried a database using both approaches in a counterbalanced design. The study started with 34 subjects but 14 dropped out after the instructional phase. The formal language used was a subset of SQL, one that allowed for simple mapping retrieval, composite retrieval, and logical conjunction and disjunction. The formal language SQL was the outgrowth of the original formal language SEQUEL (Date, 1986). The natural language mock up for the study possessed the same functionality. Results showed no advantage for the natural language implementation over SQL. In an analysis of speed and accuracy few significant differences were found.

In a study of the impact of procedurality on query writing performance, Welty and Stemple (1981) found support for procedural languages. They trained participants, 150 undergraduates with little or no programming experience, on two identical languages, save the procedurality component, and then applied performance tests. The languages used were SQL (non-procedural) and TABLET (procedural). An example of a TABLET query is shown in Figure 12. As mentioned, results favoured the procedural language. The difference was, however, restricted to what the authors called 'hard queries', those queries that involved the use of Boolean set operations, join operations, and combinations. On simple queries (e.g., simple mapping on a single attribute), no significant differences were found.

```
FORM NSCITIES FROM NAME, PROVINCE
KEEP ROWS WHERE PROVINCE = 'NOVA SCOTIA'
PRINT NAME
```

Figure 12: Example of a TABLET query.

In a 1981 review of existing studies of query languages Reisner presented a good analysis of the field up to that point and made a number of suggestions for the improvement of query languages (Reisner, 1981). She covered the early studies on language design (Chamberlin, Gray, & Traiger, 1975; Zloof, 1975; Zloof, 1977). By way of suggested improvements Reisner (1981) approached the problem in the following manner. She stated that in order to improve the query facility it was necessary to take a three step approach:

(1) problem identification;
(2) cause identification;
(3) improvement suggestion.

In a systematic attempt to classify problems, the author found, as expected, that some functions of query languages were more difficult to use than others. In particular, she found that few users had problems with simple mapping functions and selection, but that many had problems with computed variables and the "group by" clause of SQL. In a classification of minor errors she found many syntactical errors, such as misspellings and incorrect punctuation. According to Reisner, in an attempt to identify the causes of problems, it is sometimes necessary to first identify the problem, then generate possible hypotheses for the error, and finally test the hypotheses. She gave an example of the misuse of the Group By clause in SQL. She suggested that the error emanates from the procedurality of the language and hypothesized that a change from a translation strategy (one encouraged by the structure of SQL) to a table operation strategy would alleviate the problem.

Fogg (1984) produced a graphical query system based on earlier work using E-R (Entity-Relationship) diagrams. The basic idea was to give the user orientation information about the structure and layout of the database through the presentation of an E-R diagram. Fogg's system was an improvement over earlier text based E-R systems in that it employed a bit-mapped display and a pointing device. Users could select tuples for viewing by clicking on them with the correct mouse button. Fogg did not provide any empirical evidence for the efficacy of his system, but did say that an informal viewing of the system received enthusiastic response, especially in comparison to an earlier E-R system that used a character based display.

In an human factors experiment of error correction using the SQL approach, Welty (1985) found support for the inclusion of such a correction facility. Past research into the use of SQL as a query language had revealed a number of standard errors made by users. Basically, these errors included misspellings of function names, misplaced parentheses in function calls, additional "where" keywords, misplaced quotation marks for field names, and misuse of function calls in the "where" clause. The author designed a system that would allow for and correct errors of the aforementioned types. He ran an experiment utilizing two randomly formed groups to test the error correcting system against a standard SQL system. There were 40 undergraduate subjects in the standard group and 39 undergraduate subjects in the error correcting group. None of the subjects had any previous programming experience. It should be noted that this was a paper and pencil system and that no computer implementation was made. Welty found support for the error correcting system. Specifically, subjects displayed a 26% improvement on a retention test while using the new system over the standard system. Further analysis revealed that subjects in the error correcting condition did not make more non-correctable errors, thus confirming the hypothesis that subjects would not extrapolate the error correcting mechanism beyond its actual scope.

The preceding discussion gives an overview of the early research into query languages. It appears that the main concern was to put querying power into the hands of naive users - naive in the sense that they would not be computer experts, but rather experts in some field of inquiry. A number of the authors of articles in this time period clearly state this as an objective. In a similar vein, many of the researchers realized the need for a multi-tiered querying interface. A layered approach was recognized as needed to fulfill the needs of all users, some naive and casual, some proficient in the content area, and some experts in the use of computers and databases. Many different approaches were taken in an attempt to realize these goals. The most prominent tack was the investigation of some structured or formal language as a querying mechanism. Early attempts included the languages SQUARE and SEQUEL. Here, researchers attempted to remove the mathematical knowledge requirement from the querying process. It appears from the research that some of the goals were met, namely, it became possible, through the use of one of these languages, to teach database querying to naive users. The overarching goal of all these early studies was to increase the ease of use of the querying system. As some authors worked with structured languages others went off on a completely different tangent. These researchers believed that in order to achieve the aforementioned goals, the linearity of an English-like language would have to be removed from the process. The first of these researchers, Zloof, designed and implemented his Query-by-Example (QBE) system. He reported some very promising results, the most salient being the short learning time required for the system. Others in the area looked at the use of a graphical approach to solving the problem. Again, some systems were built and tested and positive results reported. One other prevailing theme in the research at the time was the emphasis on a usability or human factors approach to query language design. It should be noted, however, that many studies utilizing this human factors approach had low sample sizes and correspondingly low statistical sensitivity.

Database Query Language Research: 1986 - Present

Research continued on database query languages during the 1986 - present period. During this time computers and their operating systems changed. The availability of desktop microcomputers continued to grow and researchers were afforded additional power at the desktop level in their search for optimized query language systems.

Katzeff (1986) investigated the reaction of users to new query situations. In her study, 20 users were first introduced, through a printed booklet and performance test, to a simple query facility called CAR (Category and Restriction). After reaching proficiency, the subjects returned within two days to complete an experimental test that involved two types of questions. One type echoed what the students had learned in the introduction and the other type involved complex queries to which the students had not been exposed. Students were free to tackle the second type of problem in any way they saw fit. The analysis showed that two approaches were taken. In the first, the student would submit consecutive queries and mentally combine the results to determine the correct response set. In the second approach students extended the query language to produce complex queries, those involving Boolean connectives and negation, that would return the required set. Katzeff's analysis showed there was marked and significant propensity towards the second approach. In her words, "The most striking of our findings is that when dealing with the more complicated questions, the largest proportion of the Ss seemed to prefer the use of one complex query instead of two simple queries"(p.14).

In a follow-up article Katzeff (1988) examined the role conceptual models play in query writing. Specifically, she wanted to know if giving the user different levels of information concerning the database and the query facility would lead to differing performance levels in a query writing task. In the study, 20 undergraduates with little or no experience with computer terminals, were provided with introductory booklets containing one of the following: no database model, a table model, a shallow set model, or a deep set model. The set models differed in that one included information on the intersection and union of negative sets. Additionally, a second independent variable, linguistic structure, was investigated. Results showed a significant main effect for the model type on the number of correct queries as well as a significant main effect for the linguistic structure factor. No significant interaction was found. Contrast analyses showed a significant difference between the set model levels and the no model-table model conditions. Subjects produced significantly more correct queries when presented with the manual containing information on the set model on which queries are based. In the discussion Katzeff noted some of the problems subjects had in formulating queries. One error subjects made was to substitute the union operator when the intersection operator was required and vice versa. She attributed some of this error to the fact that subjects often see the query writing task as a language translation operation, from the English language question to the formal language query, rather than a concept translation operation.

In two studies of procedural versus non-procedural languages Hansen and Hansen (1987,1988) sought to ascertain the human factors qualities of query facilities. They studied languages based on relational algebra, tuple relational calculus, and domain relational calculus. They found support for their hypothesis that the procedurality of a query language affects query writing performance using 83 advanced level information management students as subjects. Specially, they found significantly higher performance for subjects writing queries in a relational algebra system than in the others. This result was seen in complex queries only, no difference was observed for simple queries. In addition, the authors found increased performance in domain relation calculus over tuple relational calculus.

Kim, Korth, and Silberschatz (1988) developed a graphical query language, PICASSO, as an attempt to alleviate some of the existing problems with query languages. They listed the problems as: the difficulty for new and naive users to use and understand existing languages; the extensive learning period required by users; the attention to detail, (field names for example), that users must maintain in order to construct queries and; the unfamiliarity of naive users with the mathematical concepts, such as relational algebra and set theory. To combat these problems the authors presented a graphical query tool designed around a hypergraph implementation. Their system was built on earlier work by Maier and Ullman (1983) on the universal relation model. Under this model the user does not have to concern himself or herself with the logical structure of the database since the database is presented as one relation or table to the user. The user poses queries using a pointing device such as a mouse, clicking on the desired attribute (e.g., Unemployment), and typing in values if needed. Unfortunately, the authors do not provide any empirical evidence concerning the usability of the system by the intended user population. The implementation of the PICASSO system is shown in Figure 13.

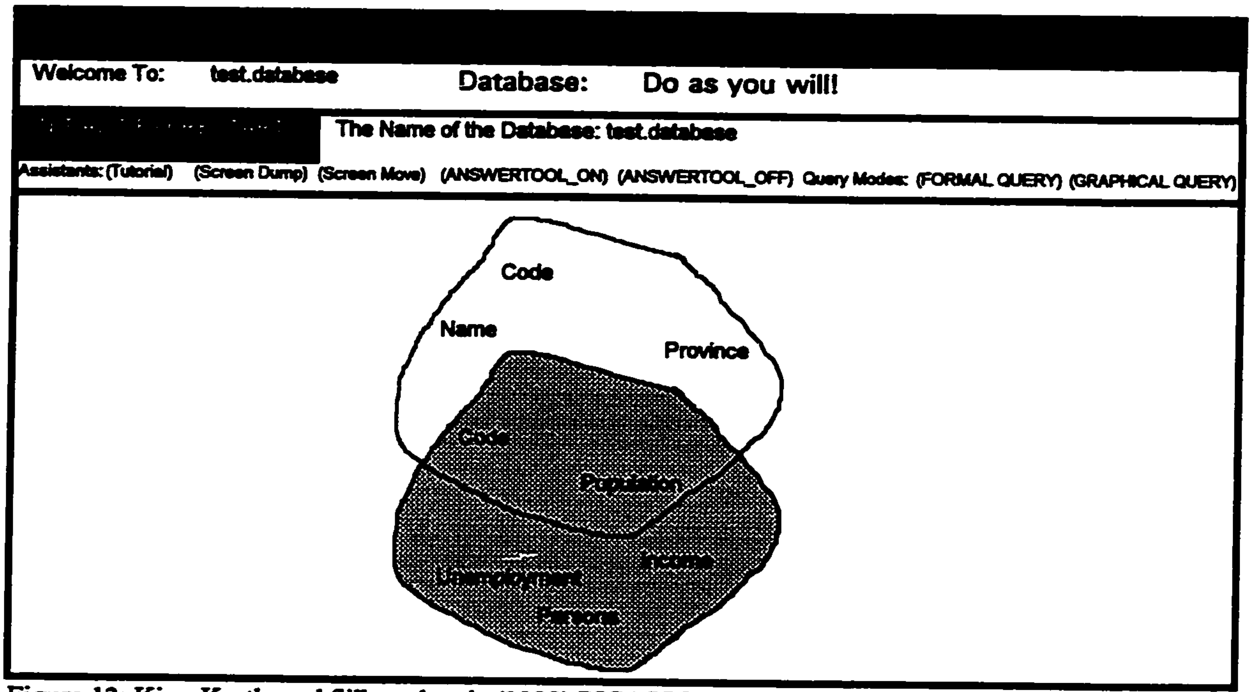

Figure 13: Kim, Korth, and Silberschatz's (1988) PICASSO system (p. 174).

Jih, Bradbard, Synder, and Thompson (1989) investigated the use of direct data models on query writing performance. Specifically, they were interested in whether the use of relational diagrams or entity-relationship diagrams would result in differing query writing performance. To investigate, the researchers developed two similar workbooks that differed only in the diagram used to show the conceptual model of the database to the user. They then had users, 56 students enrolled in an information management course, practice query writing tasks and complete a four query test. The variables of interest were conceptual model and query complexity. The authors measured syntax errors, semantic errors, and completion time. Results were equivocal. They found a significant main effect for data model on syntax errors, but this did not occur for semantic errors. Additionally, they found no significant interaction effects. They concluded that either model conveyed the same amount of information for query writing. The authors expressed concern for their results and pointed to experimental problems that may have caused them, such as subject population (students), type of task, and duration of task.

Davis (1989) undertook an investigation of database querying using a formal language, SQL, versus menus. He believed that the use of menus to access database information was an underutilized approach to querying and examined a menuing system MAPICS (Manufacturing and Production Information Control System) from IBM. He departed from many earlier studies in that he had subjects complete actual querying tasks on a computer as he believed that results obtained from paper and pencil tests were not truly indicative of user performance. The study involved two experiments, varying across delivery platform, with N = 116 in experiment 1 and N = 81 in experiment 2. Results favoured the formal language interface, with subjects performing significantly better with the SQL interface and reporting significantly higher preference scores for that interface. Furthermore, a hierarchical regression analysis revealed no significant effect of subjects' background on performance across interface types. In the second experiment, Davis examined the use of directories to aid in database querying with menu systems. He found support for the use of directory information with a significant result between performance by subjects without directory information and those with such information. In an informal protocol analysis Davis found subjects reporting problems with both interfaces. With the menu system subjects reported unclear menu labels as a major source of frustration. The end result of the label problem was the leading of users down blind alleys, in other words, to a screen at the end of a hierarchy that did not contain the required information. Also, subjects complained of crowded screens making interpretation difficult and recognition of embedded menus difficult. On the SQL side, problems with correct syntax, the choice of appropriate data tables, and multi-table queries were cited as the major problems.

In an analysis of the effects of training on detection of query errors Dayton, Gettys, and Unrein (1989) found some support for the addition of theoretical training. In their experiment, 60 undergraduate psychology students received either straight procedural training or training specified as theoretical. The theoretical training consisted of the procedural training as well as instruction in the Boolean model of database querying. The authors were interested in whether this enhanced training would lead to higher performance in the detection of errors in database queries. Additionally, they factored in the type of error, either a slip (simple syntax error) or a mistake (error based on semantic problem) and query complexity, defined as scrambled versus orderly queries. Results showed a lower sensitivity for scrambled errors in the procedural group, whereas, the theoretically trained group showed no decrement in performance across query complexity.

Greene, Delvin, Cannata, and Gomez (1990) undertook a study of database querying that examined a number of factors involved in the process. They developed their own query interface, the Truth-Table Exemplar-Based Interface (TEBI), and compared query performance against the standard SQL type interface. The interfaces differed mainly on the syntax of the query language. The TEBI interface allowed users to submit queries without regard for Boolean operators or parentheses. The system would respond with exemplar records satisfying all combinations of logic and the user would indicate their preferred logic by selecting the appropriate exemplar record. They were also interested in a number of other independent variables including individual differences and query logic. The dependent measures in the study included number of correct queries, time performance, confidence, and error analysis. Subjects in the study, 80 females recruited through a local paper, were given a series of cognitive tests on the first day and then randomly assigned to one of four conditions. The second day consisted of a training period for each of the four groups. The authors attempted to make the training equivalent across groups. The final day included the performance tests. Analysis of the data revealed that the type of interface significantly affected performance, with subjects in the TEBI conditions outperforming those in the SQL conditions. A further analysis of these data revealed that subjects were able to correctly understand the queries, that is, there were no significant differences across conditions, and that the performance differences were the result of the query language itself.

These authors were also interested in the effect of query complexity on performance. To ascertain the effect they classified the queries into four types, those requiring a simple *'And'*, those requiring a simple *'Or'*, those requiring negation, and those requiring *'And'* and *'Or'*. Analysis revealed a number of significant main effects and interactions. First, query complexity had a significant effect with the more complex queries, those involving *'And'* and *'Or'*, being more difficult. The remaining order of difficulty was: negation, simple *'Or'*, and simple *'And'* queries. Additionally, as stated before, subjects in the SQL condition performed worse than those in the TEBI condition. The authors found a significant interaction between complexity and query system with the effect most pronounced for negation queries. In the realm of individual differences the authors found reasoning skills to be the most pertinent to query performance, although this influence was somewhat reduced in the TEBI condition. That is, subjects with low scores on the reasoning skills tests performed well when using the TEBI interface whereas, the same claim could not be made for the users of the SQL interface. An analysis of the errors made by subjects revealed that the most common was one in which flawed logic was the culprit. Table 1 contains a list of the errors found in the study.

Table 1: Error results from the Greene et al. (1990, p. 320) study.

Parentheses	Percent of incorrect queries with at least one error in parentheses	23%
	Percent of parentheses errors in queries that required parentheses for disambiguation	50%
Logical operators	Percent of incorrect queries with at least one operator error	75%
	Percent substitutions of ANDs for Ors, given the opportunity in incorrect queries	43%
	Percent substitutions of ORs for ANDs, given the opportunity in incorrect queries	10%
Parentheses or logical operators	Percent of incorrect queries with either at least one parenthesis error or one operator error	84%
Syntax	Percent of incorrect queries with at least one syntax error	68%
Selection Criteria	Percent of incorrect queries with at least one error in selection criteria	28%

These included the substitution of the intersection operator for the union operator, and vice versa. The substitution of *And* for *Or* was the most prevalent. The authors attribute this finding to the fact that in natural conversation *And* often represents union rather than intersection. Other logical errors included the improper use of parentheses in the query formation. The authors conclude that it may be necessary to remove or reduce the requirements for understanding formal logic in order to allow all users to effectively query a database.

Linde and Bergström (1991) studied the transfer effect of instruction on query writing performance. In the study, 20 undergraduate subjects received instruction either classified as deep structure training or shallow structure training. Briefly, deep structure training involved teaching users to first examine a native language query in terms of "what is asked for" and "what is known" before transferring the query to a formal language. Shallow or surface structure training consisted of teaching the subjects the rules of syntax for the language. In addition, the authors investigated the type of query interface, textual or graphical. The end result was an experiment with a 2 x 2 factorial design (training by interface type). Results showed no significant main effects, but a significant interaction between interface type and training on dependent measures of number of errors and frequency of assistance. Linde and Bergström (1991) concluded that the interaction occurred because the users were forced to analyze the native language query in terms of deep structure when using the textual based query system but not when using the graphical system. They also pointed out that a positive effect occurred for the graphical interface and deep structure training but that the effect failed to reach significance.

A study of logical operators in database querying was performed by Essens, McCann, and Hartevelt (1991) in an attempt to better understand this problematic part of the process. Their study examined query interpretation by people not trained in logic. They also manipulated the complexity of the queries interpreted, varying the number of operands and the degree of negation. Essens and his colleagues conceptualized database querying as a three step process. First, the user creates a mental representation of the database subset desired. This step is contingent on the user's model of the database. Second, the mental representation is used to construct a query that will result in the desired information.

Third, the resulting subset is examined for congruence with what the user expected. If the result is congruent the process stops, otherwise, it continues in an iterative fashion until the third step is satisfied. The authors were most interested in the specification of the subset of data from the database. Here, users construct queries using formal Boolean logic and it is here that the authors believe the system breaks down. They stated that the problems with the use of Boolean operators are well known and that much anecdotal evidence exists. Their study consisted of 12 university and high school age subjects completing three sets of questions. The experiment required the users to respond to a query and result as being correct or incorrect. The categories of queries were single operator, double operator, mixed operator, and mixed operator with parentheses. Additionally, the queries could contain 0, 1, or 2 negations. Dependent measures were latency time measures and errors. The results showed no significant differences between single and double queries and none between queries using *'Ands'* and those using *'Ors'*. One significant finding was the difference between the mixed category and the mixed with parentheses category with the parentheses speeding up performance. Also, significant effects were found for the number of negations. There was no significant interaction between the two variables on the time data. Analysis of the error data revealed a significant difference between the mixed category and the mixed with parentheses category. As well, a significant difference was found between the double and mixed categories. In summary, the data showed that subjects had more difficulty with more complex queries (i.e., those containing mixed operators and negations), but, that the inclusion of parentheses had a strong positive effect on time and error rate.

Burgess (1991) sought to ascertain whether different graphical representations of a database would facilitate querying for naive users. In this study naive users, who were considered experts in the database content, were subjected to a number of varying graphical interfaces and their performance monitored (Subjects were 11 clinicians from the Affective Disorders Unit of the University of Texas Health Services Centre). The graphical interfaces varied on the amount of detail they presented on the underlying structure of the database. Some presented a simple hierarchical view of the data, whereas others presented a network view. Detail varied from presenting simple graphical links to showing links containing relational information (in much the same manner as an E-R diagram). Results showed a general improvement with the graphical interface over the existing text based one. Further analysis showed a significant difference on preference for the simple graphical interface over the E-R style one.

In a similar study, Davis (1990) investigated the use of graphical diagrams of the database in querying. He assigned subjects to one of four groups. The first group received a list of the table contents in the database, the second group received a data structure diagram, and the third and fourth groups received an E-R diagram of the database. In total, 46 undergraduate students enrolled in a Management Information Systems course were studied. The final two groups differed in that the first received a series of E-R diagrams (one per table per page) and the second received a global E-R diagram (one diagram covering the entire database). Results showed a query performance improvement for all of the diagrams over the simple contents list, the improvement was most pronounced for multi-table queries. Contrast analyses revealed no significant differences among the diagram conditions. The authors concluded that any of the diagram types improved performance.

A paper from the Information Retrieval (IR) literature looked at the use of graphics to aid in the querying of databases (Olsen, et al., 1993). This system, VIBE, although intended for the searching of databases of text documents, allowed for the searching of databases of numerical information. The system called for users to define Points of Interest (POI) through either standard queries, user profiles, or existing documents as a first step in the querying process. The system, using these POIs, would then display record surrogates using some similarity index. The user was free to adjust the position of the POIs to better understand the structure of the database.

Direct Manipulation

Recent research has focused on the use of direct manipulation (DM) interfaces for database querying. Basically, direct manipulation interfaces conform to a number of principles. According to Williamson and Shneiderman (1992) these include:

> "- continuous visual representation of objects and actions of interest
> - physical actions or labeled button presses instead of complex query syntax
> - rapid, incremental, reversible operations whose results are immediately visible
> - layered or spiral approaches to learning that permit usage with minimal knowledge." (p. 338)

The principles of DM have been used to design and build a number of differing applications including wordprocessors, spreadsheets, graphics packages, and even operating systems (Ahlberg, Williamson, & Shneiderman, 1992). Furthermore, the use of DM for these applications has been a success (Shneiderman, 1987).

In the area of information retrieval from structured databases these principles have a number of implications. First, there is a requirement that the user be made immediately aware of his or her query results. It is not enough to have the system respond to a textual query with a list of the records matching the query criteria. The results must be more immediate and more obvious to the user. Second, complicated text driven queries do not meet the standards laid down by the principles. The interface must represent attributes in such a way that the user gains some knowledge of the attribute and is able to vary the value searched, in an easy and intuitive manner. Third, the interface must be such that naive users, those without prior knowledge of database querying, can quickly and effectively search or browse a database. Finally, the interface must be malleable enough to allow for mistakes without causing dead-end searches and quickly return to the system state before the mistake was made.

One of the earliest papers to examine the use of DM interfaces in information retrieval was that of Ogden (1986). The author investigated DM interfaces along with natural language and formal (i.e., SQL) interfaces. The results favoured the DM interface as a solution for a number of existing query language problems, namely, training and error rate. Research undertaken at the University of Maryland has also concentrated on the use of direct manipulation principles in the design of retrieval interfaces (Ahlberg, Williamson, & Shneiderman, 1992; Williamson & Shneiderman, 1992).

In the first of these studies, Ahlberg, Williamson, & Shneiderman (1992) sought to find an easier way for users to query databases. They based their attempt around the use of a direct manipulation approach. The authors stated that an interface to a database should meet a number of criteria. First, the query should be represented in some graphical manner and this graphical representation should contain some information vis-à-vis the range of values appropriate for the query. Second, the user should receive a graphical picture of the database and the results of any query should be represented in this picture. Third, the user must receive immediate feedback of a change to a query parameter. Fourth, the interface should be such that a new or naive user can complete a query without much training but scale up to the needs of expert users. To meet the above criteria the authors built a system for querying the periodic table. In the experiment they compared the DM interface to one that mimicked the DM interface in output but required textual input and one that was entirely text based. Dependent measures included performance measured in time, correct answers, and preference or satisfaction. Results, from 18 undergraduate chemistry students, favoured the DM interface on the performance measure with a significantly higher performance for the DM interface over the hybrid or the text-based. For preference, a significant difference was found between the DM interface and the text-based one; however, no significant difference was found between the DM interface and the hybrid. Statistical results were not reported for the error rate across interface type.

In a similar study, Williamson and Shneiderman (1992) investigated the use of direct manipulation for querying of a real estate database. In this study they compared the 'dynamic query' interface, one based on DM principles, to a natural language interface and to a traditional paper data source. The study employed 18 subjects in a within-subjects design. Dependent measures included a performance score measured in time to complete a query and a preference score. As in the earlier study, results clearly and significantly supported the DM interface on the performance measure. Furthermore, significant results were found for the preference measure with subjects showing a preference for the DM interface.

In both of these studies the use of Dynamic Queries, those based on a DM interface, was investigated. In the first case, the database consisted of the periodic table that the user could query using values for six attributes. In the second case, the database consisted of real estate listings and, again, the user could query the database using a fixed number of attributes. Both studies compared the use of DM with standard query mechanisms, namely SQL and QBE interfaces. The DM approach employed the use of graphical widgets, sliders and on-screen buttons, for inputting the user's query. In both cases the results were similar. The DM interface showed a performance increase, as measured by time to complete a task, over the standard query mechanisms. The authors, however, pointed out a number of deficiencies in the interface and suggested some areas for future research. These include; making use of the limited screen area when a large number of attributes are available for searching and how to include a full set of Boolean operators into the interface.

Research continued on the use of dynamic queries through the mid 1990s, although the direction changed somewhat. The papers in this time period focused on the back-end issues in such a system. Specifically, the researchers were interested in examining data structures that would allow for the rapid search of large record sets (Tanin, Beiger, & Shneiderman, 1996). This was of particular importance as one of the hallmarks of the dynamic query system is the rapid update of feedback to the user. During the period the system was also tested with larger scale databases, most noteworthy a database of satellite data (Pointek, 1995) and with networked database systems (Doan, Plaisant, & Shneiderman, 1996).

In summary, research on database query languages continued during the 1985 - present period. Much of the research followed the path set by the early investigators with some diverging avenues. Many of the studies investigated the use of graphical diagrams of the database structure to aid querying. This was not new, it had occurred in the first period, but it became a more prevalent research topic. One possible explanation for the increase revolves around the changes made in computers in general. Specifically, computers during the period 1986 - present became more powerful and allowed for these graphical type of displays. Another hot topic in those ten years was the use of training as an aid to querying. Many researchers had combined training with graphical depictions of the database for some positive results. There was a continued move to get away from text-based systems (e.g., SQL) in favour of a less linear approach. For example, the use of menus as part of a querying system was examined. Boolean logic, and the problems emanating from its use, were also examined with the conclusion that users have problems with this area of querying but will employ it if given the requisite tools. Finally, a new approach at database querying was studied, that based on direct manipulation interfaces.

Experimental Hypotheses for the Study

The current research sought to extend the findings of studies on dynamic queries in two distinct ways. First, a user-defined list of factors that was controllable for searching was examined. This list could vary both in content and number. The study concentrated on two methods to extend this capability. Second, the inclusion of a direct Boolean *'Or'* facility was examined. Here, the option was to compare an indirect implementation to a more direct implementation. Following the existing literature, dependent measures were performance, as measured by time to complete a query as well as correctness, and satisfaction.

Two independent variables were studied in the experiment. The first, factor list type, had two levels. The scrolling implementation allowed the user to add search factors to a scrolling list whereas the windowing implementation allowed users to add search factors in groups. The second independent variable, Boolean operations, dealt with the mechanics of forming queries containing *'Or'* connectives. The indirect implementation closely emulated earlier work in that the user would basically carry out multiple intermediate searches to respond to questions with the added functionality of providing the user with a place to store the results of these intermediate operations. In the direct implementation users would form queries with Boolean 'Or' connectives in response to a question.

The rationale for the implementation of the first independent variable came from a number of sources, not the least of which was the suggestions for further research by the investigators of dynamic querying. They stated that one necessary addition to the direct manipulation approach for database querying would be the capability for users to search the database on a large number of factors. This is commonly referred to as the *'real estate'* problem in discussions of interface design. Specifically, there is a finite amount of space to display input and output tools to the user and, the problem is, how best to utilize this finite space. In the case of querying, a facility for the user to construct a set of search factors must be provided. In this experiment two designs were tested. In the first the user could add to a scrolling list the factor they wished to consider in their search, the scrolling design, and in the other the user could add to a preset group the factor they wished to consider, the windowing design. These two designs evolved from the research into other areas of interface design, those in which the information to be presented exceeds the available display space. In many of these cases one of two designs is used, either the information is presented in a scrolling window (e.g., a word processing program) or the information is presented in small chunks (e.g., the tabbed selection dialogue boxes popular in many current programs). The windowing design was also based on the idea that a design that allowed the user to insert factors into preset groups might result in different performance than a design that allowed the user to insert factors into a scrolling list in much the same way that chunking allows for increased short term memory performance. For example, people may exhibit better recall performance on lists of digits when they employ a chunking strategy rather than a serial approach (Miller, 1956). A chunking strategy involves the person grouping the digits together into chunks and committing these chunks to memory. For example, people are better at remembering long distance telephone numbers, ten digits, than they are at remembering a ten digit serial number. Apparently, people chunk the telephone number into area code, exchange, and number and remember the three chunks rather than the ten serial digits. The third concept driving the design of the scrolling versus windowing dichotomy was the expression in earlier research that users had problems forming queries with the correct syntax and, in particular, with the placement of parentheses in the search logic. It was hoped that the windowing style of list type would reduce the incidence of this problem.

The second design variation, direct versus indirect Boolean 'Or' facilities, was empirically derived. Some of the earlier research had pointed to problems with this area of database querying. In fact, some research had concluded that users were incapable of correctly forming proper queries from a logic point of view (Greene, et al., 1990). The research on direct manipulation had not addressed the problem as a design goal but had stated the inclusion of a Boolean 'Or' facility as a goal of further research. They had favoured, instead, a solution that required the user to complete multiple queries and manually combine the results to achieve this capability. Other research had provided conflicting evidence; it had found that users, when presented with the capability to form complex queries containing unions, would rather solve a problem in this way (Katzeff, 1986). The second variable in this study attempted to solve this conundrum. In one case, the indirect approach, the user was provided with a tool to combine multiple search results thus emulating earlier work done on dynamic queries. The main difference between this study and the earlier work on direct manipulation, the inclusion of the tool for combining lists, was built into the querying application. In the other design, users could form queries by directly inserting an 'Or' into the list in the scrolling design or by adding factors across groups in the windowing design.

Given the above the following hypotheses were made:

Hypothesis 1.1: There will be a significant difference in *performance* between the scrolling extension and the windowing extension of the search factor list variable.
Hypothesis 1.2: There will be a significant difference in *satisfaction* between the scrolling extension and the windowing extension of the search factor list variable.

Hypothesis 2.1: There will be a significant difference in *performance* between the direct and indirect Boolean 'Or' operations.
Hypothesis 2.1: There will be a significant difference in *satisfaction* between the direct and indirect Boolean 'Or' operations.

Exploratory Data Collection

Also of interest in the current study was the impact of mental workload and interface design for database querying. According to Eggemeier (1988, p. 41) mental workload "refers to the degree of processing capacity that is expended during task performance" This concept is of particular interest in the area of interface design since the processing capacity that is devoted to the use of the interface is not available for the primary task. For instance, if a query interface is such that the user has to devote a high degree of mental capacity to using the interface then this capacity is not available for the task at hand, retrieving records from the database. A display of the hypothetical effect of mental workload on task performance is shown in Figure 14. The curve denotes, at the low end, a situation where some workload must be present for task performance to occur. The middle section represents optimal performance and the upper end represents the situation where mental workload is so great that task performance suffers. The curve follows those well established in the areas of information processing and arousal (Cox, 1978) and activation level and performance (Blum & Naylor, 1968).

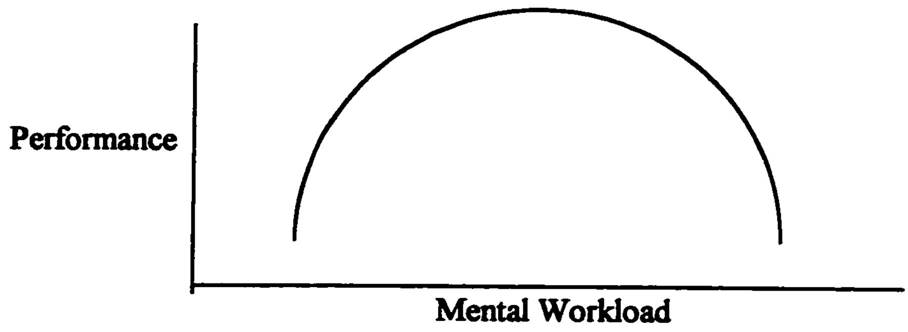

Figure 14: Mental workload and task performance - hypothetical curve.

The assessment of mental or cognitive workload has been used in a number of situations, primarily in those in which the outcomes are dangerous and/or catastrophic (e.g., nuclear power plant operation, jet fighter flight, etc.). It has also found its way into the literature on user-computer interaction. A number of researchers in the area of human computer interaction have investigated the concept of mental workload and its application to interface design and computer operation. Some studies have focused on the acquisition of information from the system. For example, the interpretation of graphical information (Casali & Gaylin, 1988) and studies regarding the reading of text from the computer screen (Irler & Barbieri, 1990). Some researchers have looked at the acquisition of general operational skills in computing systems and the impact of cognitive load (Harvey & Rousseau, 1994; Payne, 1994) while others have focused on specific functional skills such as completion of simultaneous tasks (Bannon, Cypher, Greenspan, & Monty, 1983; Holden & O'Neal, 1992), menu traversal and memory load (Wright & Likorish, 1994) and the reduction of cognitive load through the use of non-speech audio (Brewster, Wright, & Edwards, 1994). Other researchers have focused on multimedia database design (Laurel, Oren, & Don, 1990; Yamada, Hong, & Sugita, 1995).

Some studies have investigated the measurement of mental workload and computer system operation. The primary focus in these studies has been to identify an index of workload. To this end, researchers have studied heart rate variability (Itoh, Hayashi, Tsukui, & Saito, 1989; Murata, Mikaye, & Kumashiro, 1989; Murata, 1991; Wastell, 1991) as well as pupil reflex and movement as indicators of cognitive load (Saito &

Taptagaporn, 1991; Saito, Taptagaporn, Hirose, Saito, 1991). Still others have sought composite indexes of mental workload employing physiological and psychological measures (Weiland-Eckelmann & Schwarz, 1993) and interface evaluation from a users point of view (Houwing, Weithoff, & Arnold, 1993).

In the present study no firm hypotheses were made in relation to mental workload; however, some exploratory data were collected. The study utilized a psychological measure of mental workload, a subset of the Subjective Workload Assessment Technique, to collect responses on the subjective impact of using the system (See the Method section for a complete description of this instrument).

Chapter 3 - Method

Participants

Selection of Participants

Subjects for the study were recruited from the MBA class at Dalhousie University. The recruitment procedure involved a class announcement of the study and a sign-up opportunity for one of the experimental sessions. Subjects were selected from classes in Training and Development and Human Resource Management. In all, data were collected from 72 subjects randomly assigned to one of the experimental conditions. The rationale for including only MBA students versus other disciplines was predicated on the idea of content experts. As mentioned earlier, one of the goals of the early research was to provide query mechanisms that could be used by persons who were not necessarily computer experts but that were fully proficient in the content area of the database. It was felt that MBA students most closely approached this ideal in the available subject population. They would not necessarily be computer experts but they would be familiar with the content of the test database, which in this case was demographic and economic information tied to geographic locations. Although an abundant collection of undergraduate students were available it was felt that these students might be intimidated by both the situation and content of the experiment.

Demographic Profile of Participants

Since subjects were recruited from the ranks of the MBA program the majority reported MBA as their current degree program. In fact, 76.4% of the subjects reported their current degree program as MBA. Other disciplines reported included Master of Public Administration, 8.3%; Master of Library and Information Studies, 4.2%; and Other; 8.3%. Two subjects, representing 2.8% of the sample did not report a current degree program.

A summary is provided in Table 2 of the demographic profile of the sample. A full set of experimental data were collected from 72 subjects. In a few cases, subjects experienced technical difficulties with either the network or the test software and could not complete the study. These data were purged from the database before the analysis. The sample heavily favoured male participants with 48 males versus 24 females completing the study. Group numbers were fairly consistent across the experimental conditions with 19 subjects in the most populous condition versus 17 in the least populous (See Table 2 for a complete summary). Males and females were represented across all of the experiment's conditions. A chi square test of independence between the subject's gender and condition resulted in a non significant finding - $\chi^2(3, n = 72) = 3.91$, n.s. The sample had a mean age of 24.31 years with a standard deviation of 8.32. The average program year, defined as the study year for the student's current degree program, was 1.31 with a standard deviation of .68 indicating a slight tendency for first year Masters students. Subjects were also asked to report on work experience, computer experience, and database experience. The first two, work and computer experience, contained no explanatory text and no problems were experienced. The final question, on database experience, appeared somewhat problematic during the pilot stages (some pilot subjects asked what was meant by database experience) and had the explanatory phrase, "Experience with computer databases including library CD-ROMs", added to guide the user. During the actual study no problems were encountered with this question. The means and standard deviations are given for these questions for the entire sample as well as the gender subgroups and condition subgroups in Table 2. An analysis revealed no significant differences across conditions on any of these factors. It was therefore felt that the analysis could proceed on the experimental hypotheses.

Table 2: Summary of the demographic profile of the sample used in the current study.

	Total Sample n = 72	Females n = 24	Males n = 48	Cond SD n = 19	Cond SI n = 18	Cond WD n = 18	Cond WI n = 17
Age	24.31	24.83	24.04	22.26	25.5	24.22	25.41
	8.38	10.78	7.00	10.12	7.55	7.28	8.49
Program Year	1.31	1.45	1.23	1.05	1.55	1.33	1.29
	0.68	0.93	0.52	0.62	0.85	0.59	0.59
Work Experience	4.17	5.63	3.44	2.89	4.17	4.22	5.53
	4.46	5.23	3.88	2.23	4.38	4.14	6.34
Computer Experience	5.79	5.29	6.04	4.74	6.11	5.89	6.53
	3.67	4.28	3.34	2.74	4.38	3.32	4.12
Database Experience	2.32	2.58	2.19	1.42	2.67	3.00	2.35
	2.7	2.94	2.59	1.43	3.82	2.89	2.02

Notes: Top number in each cell is the mean and bottom number is the standard deviation

Apparatus

Experimental Computer Program

The main apparatus used in the study was a computer system written to present the various querying interfaces to the experimental subjects. All interaction between the subjects and the study took place via the program, save for a brief introduction by the experimenter. A full set of screen shots from the experiment is available in Appendix A. The flow structure of the experimental program is show in Figure 15. The main test system was built around the principles of direct manipulation as provided by Shneiderman (1987). These principles include: "continuous visual representation of objects and actions of interest, physical actions or labeled button presses instead of complex query syntax, and rapid, incremental, reversible operations whose results are immediately visible" (p. 201). In the test system subjects' responses were represented by a series of icons, each icon representing one factor as well as the upper and lower limits for that particular factor. Additionally, the user was presented with a map of Canada as the control for selecting provinces in the search. Users controlled the environment through a number of physical actions. They included provinces in their search by clicking on the map. They added search factors in response to the test questions by selecting the appropriate factor from a menu system. Factor position could be changed by dragging the factor to a new position, either within the linear list or between groups. They also could remove factors from a search by double-clicking that factor's icon. Users set the limits for a particular factor by dragging a limit control icon to a specific location within the slider interface. They were free to move the limit control up and down incrementally and reset the limits to their original values. Users were not confronted with any typing during the test phase and were therefore free from making grammatical errors (e.g., misspelling factor names).

Screen	Information
1	Introduction
2	Demographic Questionnaire
3	Map Interface
4	Factor Operations
5	Boolean Operations
6	Slider Control
7	Summary
8	Splash
9	Practice 1 - 3
10	Question 1 - 8
11	Satisfaction
12	Cognitive Effort

Figure 15: List of the experimental screens presented to study subjects.

All subjects experienced the same series of screens regardless of condition. There were some changes in the content on some screens based on the experimental condition. The introduction and demographic questionnaire screen were identical for all subjects, as was the screen instructing subjects on the use of the map interface to add and delete provinces to the search. Screens covering the addition and deletion of search factors and Boolean operations differed based on the condition. Appendix A shows the layout and content for the various conditions. The slider and summary screens were identical in all conditions. In the test phase of the experiment, practice screens and question screens presented identical questions. In the practice phase, instructions differed across the conditions. The satisfaction and cognitive effort screens did not differ.

The subjects in the study operated on a database of demographic information for Canadian municipalities. The database contained information on the following 17 factors (Table 3). These data were extracted from the Statistics Canada 1991 Census. The database contained records for 900 Canadian municipalities.

Table 3: Database fields from the experimental system

Average Income - Female	Participation Rate
Average Income - Male	Persons per Household
Bilingual Speakers	Population
Employment Ratio	Population Density
Government Transfers	Population Growth
Graduates - Trade School	Unemployment Rate
Graduates - University	Workforce - Female
Household Income	Workforce - Male
Income Composition	

Dependent Measures

Performance: Following the approach of earlier studies the subjects' performance for the experiment was collected by a timing and a number correct measure (c.f., Alberg, Williamson & Shneiderman, 1992). Subjects' performance was timed from the outset of each question until the subject indicated that he or she had finished. Timing started after the subject had read the question and was ready to begin finding a solution. For the analysis, timing data were aggregated across the eight question trials. The other performance measure, number correct, was computed during the analysis phase. The responses to all questions for every subject were scored as either correct or incorrect and aggregated across the eight experimental trials.

Satisfaction: Satisfaction data were collected using a subset of the Questionnaire for User Interface Satisfaction (QUIS) (Chin, Diehl, & Norman, 1988). The QUIS contains four primary factors, overall reactions to the system, screen, terminology and system information, and learning. For this experiment the questions dealing with overall reaction to the system were utilized as this section of the test best captured the information about the current system. The psychometric properties of the QUIS are reported in an article concerning its development (Chin, Diehl, & Norman, 1988). In this article the authors report a Cronbach's Alpha for the test of .94. For the present study a Cronbach's Alpha of .87 was found. Briefly, the Cronbach's Alpha coefficient is an index of the scale's reliability and reliability is a necessary precondition for scale validity. Additionally, the authors acknowledge that abbreviated version of the test may be used in the future as presentation moves from a paper and pencil medium to an electronic medium. The format of the test is the presentation to the respondent of a series of nine point scales anchored with bipolar adjectives. The items used in the experiment are shown in Table 4. In addition, the actual presentation of the items is shown in Appendix A.

Table 4: Satisfaction measure used in the experiment

Overall reactions to the system										
Terrible	1	2	3	4	5	6	7	8	9	Wonderful
Frustrating	1	2	3	4	5	6	7	8	9	Satisfying
Dull	1	2	3	4	5	6	7	8	9	Stimulating
Difficult	1	2	3	4	5	6	7	8	9	Easy
Inadequate Power	1	2	3	4	5	6	7	8	9	Adequate Power
Rigid	1	2	3	4	5	6	7	8	9	Flexible

Mental Workload: Mental workload or cognitive workload was assessed using a subset of the Subjective Workload Assessment Technique (SWAT) developed by researchers for the US Air Force (Reid & Nygren, 1988). SWAT was developed to subjectively measure mental workload in a number of situations, primarily military in nature. The scale views mental workload as varying along three dimensions, time pressure, mental effort, and stress load. For the purposes of this study the mental effort subscale was employed. The rationale for this decision was that subjects would not find the situation overly stressful (as compared to flying a combat mission in a jet aircraft) and that since subjects were being allowed to complete each trial time pressure would be at a minimum. The scale used in the experiment is shown in Figure 16.

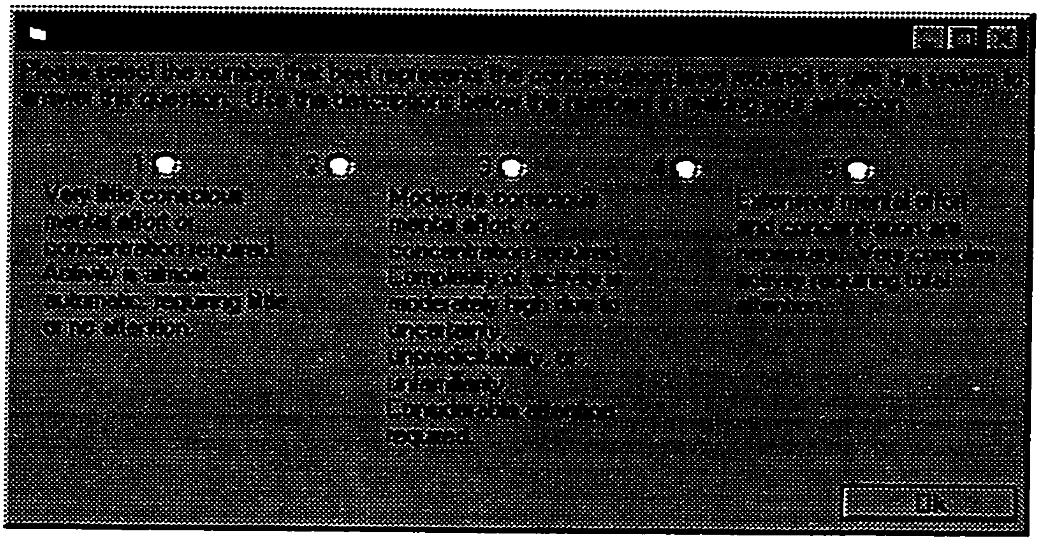

Figure 16: Mental workload scale used in the study.

Query Questions

The questions to which the experimental subjects responded to are shown in Table 5. The session contained three practice questions and eight test questions. The full text of the experimental stimuli for the questions is shown in Appendix C. It does not vary from what is shown below for the test phase of the experiment but does diverge for the practice phase. In the practice phase, as well as presenting the question, subjects were given detailed solution steps to complete the query. All subjects in all conditions experienced the same stimuli. However, the detailed solution steps provided in the practice phase did vary based on the condition. The practice questions covered the three main query types used in the experiment, simple queries containing one search factor and single or multiple parameters, complex queries requiring multiple factors joined by a Boolean *'And'*, and complex queries requiring multiple search factors connected with a Boolean *'Or'*. Compound queries requiring both *'And'* and *'Or'* were not presented during the practice phase.

Table 5: List of the questions used in the experimental session

Practice 1	In this instance you are interested in compiling a list of municipalities in Newfoundland that have an unemployment rate lower than 20%.
Practice 2	Find the list of cities in Quebec or New Brunswick that have a population between 11,000 and 33,000 and a household income greater than $50,000.
Practice 3	Find the list of cities in Quebec or New Brunswick that have a Population between 11,000 and 33,000 or a Household Income greater than $50,000.
Question 1	Find the cities in Quebec the have a population greater than 110,000.
Question 2	Find the cities in British Columbia that have a population growth rate between 1.0 and 10.
Question 3	Find the cities in Nova Scotia that have a Household Income between 40,000 and 50,000 and an Unemployment Rate greater than 10.0.
Question 4	Find the cities in Alberta or Saskatchewan that have either Government Transfers less than 15.0 or a Population Density greater than 1000.0.
Question 5	Find the cities in Ontario that have either an Unemployment Rate between 5.0 and 10.0 and a Population greater than 220,000 or a Participation Rate greater than 80.0.
Question 6	Find the cities in New Brunswick that have either an Average Male Income less than 30,000 and a Male Workforce greater than 4,000 or a Average Female Income greater than 10,000 and a Female Workforce less than 1,500 or an Employment Ratio greater than 80.
Question 7	Find the cities in New Brunswick, Nova Scotia, or Prince Edward Island that have either Bilingual Speakers greater than 5,000 or a Household Income between 30,000 and 35,000 or an Unemployment Rate greater than 20.0.
Question 8	Find the cities in Saskatchewan that have a Trade School Graduate population greater than 900 and a University Graduate population greater than 15,000 and a Population Density greater than 1,000 and a Household Income greater than 44,000.

The eight test questions were dichotimized into simple and difficult queries. Questions 1 - 3 and Question 8 were considered simple with the remaining questions, Question 4 - 7, considered difficult. This dichotomy was made on the basis of the Boolean *'Or'* requirement. Those questions not requiring a solution containing an *'Or'* were deemed simple and those requiring such a solution were deemed difficult.

Experimental Setting

Procedure

Subjects were assigned or selected a session time at the sign-up procedure. Subjects then arrived at a general computer lab for the experiment. The study was conducted in the School of Business computer lab during off hours. The timing allowed for uninterrupted completion of and concentration on the study by the participants. The general procedure was as follows. Subjects arrived at the designated session time (in total, eight sessions were run), up to 12 subjects could be tested in a single one hour session. After arrival the subject was given an Informed Consent to read and sign as required by the university's ethics committee (See Appendix B). At the completion of this stage the subjects, as a group, were given a very brief introduction to the experiment. The experimenter first introduced himself and the lab assistants and then gave a terse overview of the experiment. The most salient point of the introduction was that all interaction throughout the experiment would take place between the subject and the computer, including experiment introduction, demographic questioning, practice, and testing. They were told, however, to ask the experimenter if they were having trouble. Subjects were then asked if they had any questions; if not or after answering their concerns, they were instructed to select one of the experimental workstations and begin.

The experimental session consisted of a number of discrete sections, all presented via the computer. Subjects were first presented with an introductory screen explaining the basic idea of the experiment (See Appendix A). When satisfied that they understood the preliminary instructions, the subjects clicked the Start Button and moved on to the second section, a short demographic questionnaire. The demographic questionnaire was placed at this point to allow the subjects to familiarize themselves with the basic operation of the computer system, primarily the operation of the mouse. After responding to the questions the subjects moved on to the instructional section of the session. Here, subjects interacted with five screens outlining the basic functionality of the experimental system. Appendix A contains a complete set of screen shots of this section. Subjects first learned how to use the map interface to add and delete provinces to the search, the requisite first step in solving any query problem. Next, the subjects learned how to set up the factors to be used in the query problem. In the first screen on factor setup, instruction concerned simply adding and deleting factors to the query and second the instruction concerned how to solve a query requiring a Boolean *'Or'*. Setting the parameters for the search factors came next, followed by a summary screen outlining what had been presented and what was next in the session.

At the completion of the instructional section, subjects were taken to the first of three practice questions. The interaction for the practice section was identical to the operation of the test section, save for two parts, the collection of cognitive load information and the timing of responses. Subjects were informed that their interaction with the system would be timed during the test phase of the session but not during the practice phase. Subjects selected a practice question from the Questions menu and proceeded to follow the instructions presented on the screen. At the completion of the three practice questions the test section began. Again subjects traversed this section by first selecting a question, in order, from the questions menu, setting up the query correctly, and transferring the results to the answer list. When satisfied they had successfully responded to the question they clicked the Finished Question button and were asked to report, on a five point scale, the level of concentration required to complete the question. In total, subjects responded to eight questions. At the completion of this section subjects responded to a short preference questionnaire and selected an overall concentration level for the entire experiment. Subjects were thanked for their participation and dismissed at this point.

Debriefing of the subjects occurred after all testing had taken place. The experimenter visited the classes from which the subjects had volunteered and made a small presentation on the logic and goals of the study. The experimenter also answered any questions or concerns from the class.

Data Analysis

Data from the experiment were subjected to an Analysis of Variance (ANOVA) for the computation of the primary results. The two-way ANOVA allows for the simultaneous investigation of two or more independent variables (Ferguson, 1981). In this experiment the two variables were list type and Boolean operations. The analysis revealed information on the effects of each of these variables as well as the interaction between them. The ANOVA uses the F statistic based on the work of R.A. Fisher, first reported in 1923.

The reliability of the responses to the preference questionnaire was assessed using Cronbach's Alpha. This statistic yields an index of the internal consistency of a scale based on the attenuation of the reliability coefficient due to sample size reduction. Alpha values should fall above 0.6 for a scale (Carmines & Zeller, 1979).

Chapter 4 - Results

To reiterate, the hypotheses for the study were as follows:

Hypothesis 1.1: There will be a significant difference in performance between the scrolling extension and the windowing extension of the factor display variable.
Hypothesis 1.2: There will be a significant difference in satisfaction between the scrolling extension and the windowing extension of the factor display variable.

Hypothesis 2.1: There will be a significant difference in performance between the direct and indirect Boolean 'Or' operations.
Hypothesis 2.1: There will be a significant difference in satisfaction between the direct and indirect Boolean 'Or' operations.

These two hypotheses were tested using a 2 x 2 factorial design. As performance was measured in a number of different ways, total time to complete the task, total number correct, and a composite index, the results are presented according to these measures, as well as the satisfaction measure. The composite index was created by adding the standard scores of the time and number correct measures. A table of means and standard deviations for the dependent measures is presented below (Table 6) as well as a table of intercorrelations among the primary dependent measures (Table 7). Summary tables for each dependent measure are then reported with associated graphs.

Table 6: Means and standard deviations for the dependent variables. The first number in the cell represents the mean and the second the standard deviation.

Condition	Total Time	Total Correct	Composite	Satisfaction
SD	885.79 230.97	5.63 1.64	0.41 1.15	6.61 1.26
SI	963.61 274.93	4.50 1.47	0.06 1.68	5.51 1.67
WD	814.48 216.37	5.61 1.33	0.08 0.90	5.66 1.63
WI	890.88 168.45	3.94 1.52	-0.60 1.28	6.15 1.44
Total	888.62 228.14	4.94 1.63	0.00 1.31	5.99 1.54

Notes: SD = Scrolling factor list and direct implementation of Boolean 'Or'
SI = Scrolling factor list and indirect implementation of Boolean 'Or'
WD = Windowing factor list and direct implementation of Boolean 'Or'
WI = Windowing factor list and indirect implementation of Boolean 'Or'

Table 7: Intercorrelation matrix for dependent measures from study.

	Total Time	Number Correct	Satisfaction	Cognitive Level
Total Time				
Number Correct	-.15			
Satisfaction	-.01	**.25**		
Cognitive Level	.19	**-.39**	**-.35**	

Note: Bold face type denotes p < .05.

Performance - Time

Analysis of the time data revealed no significant differences for either the interaction component or the main effects of list type or Boolean operations. Time was measured as the total number of seconds required to complete all eight query questions, excluding the practice questions.

Table 8: Summary table of ANOVA for total time.

Source	SS	df	MS	F
List Type	106858.66	1	106858.66	2.08
Boolean Operations	93224.43	1	93224.43	1.81
Interaction	8.92	1	8.92	.00
Error	3495095.75	68	51398.47	
Total	36995500.74	71	52049.31	

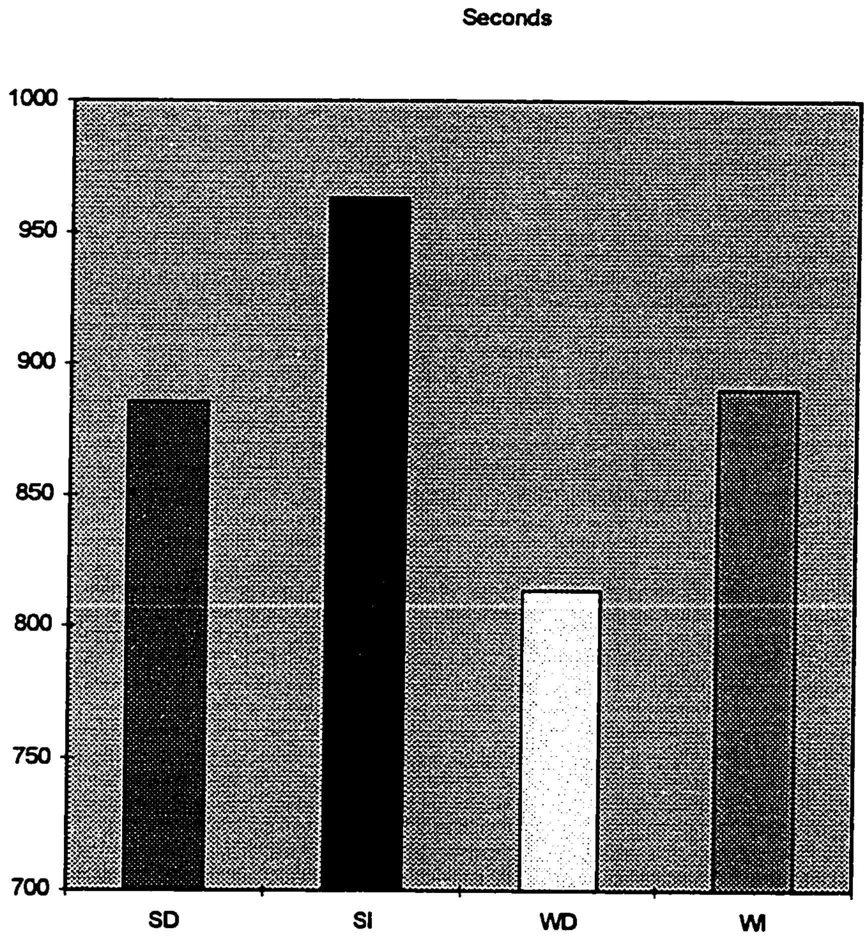

Figure 17: Bar graph of cell means for total time.

Performance - Number Correct

Analysis of the data for number correct revealed a strong main effect for Boolean operations but no main effect for list type or the interaction. Responses were judged as either correct or incorrect and the number correct summed for each subject to yield a score for the total correct variable.

Table 9: Summary table of ANOVA for total correct.

Source	SS	df	MS	F
List Type	1.51	1	1.51	.67
Boolean Operations	35.26	1	35.26	15.76**
Interaction	1.30	1	1.30	.58
Error	152.14	68	2.24	
Total	189.78	71	2.67	

** $p < .001$

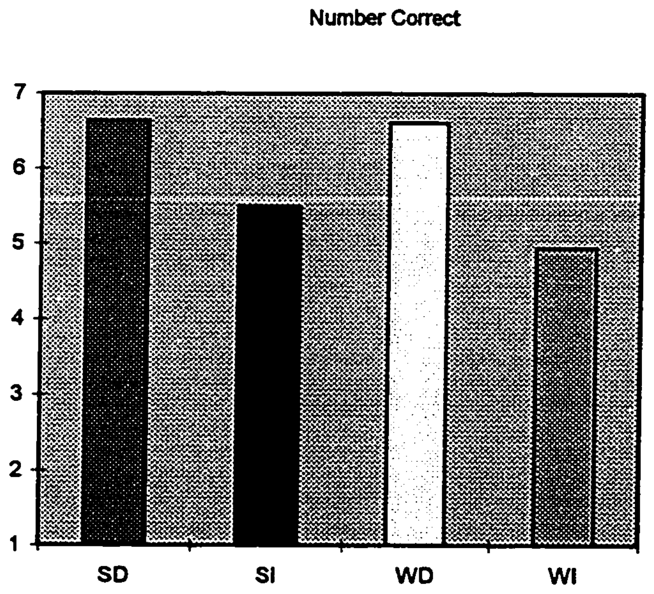

Figure 18: Bar graph of cell means for total number correct.

Performance - Composite Index

A composite index was formed by combining the subjects' normalized scores for the two performance measures in an additive manner. Results showed no significant effects for either main effects or interactions.

Table 10: Summary table of ANOVA for composite.

Source	SS	df	MS	F
List Type	4.84	1	4.84	2.94
Boolean Operations	4.37	1	4.37	2.66
Interaction	.51	1	.51	.31
Error	111.69	68	1.64	
Total	121.23	71	1.71	

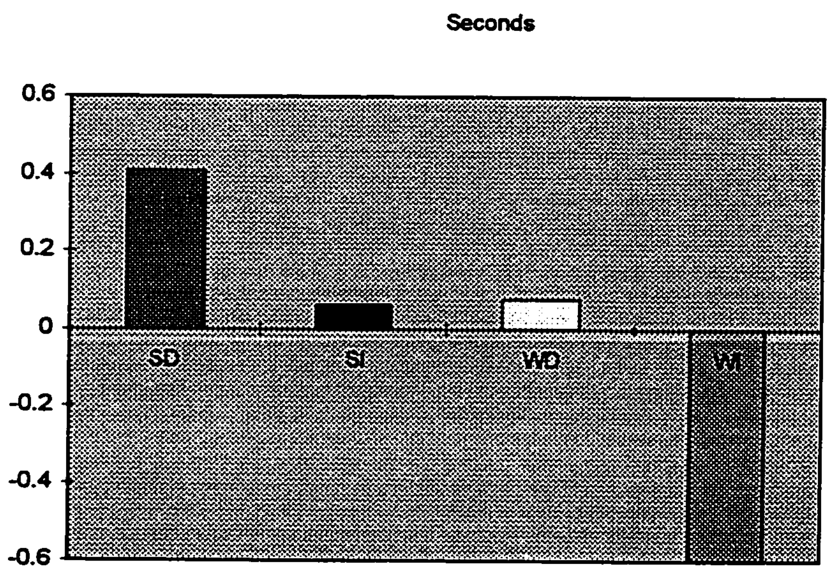

Figure 19: Bar graph of cell means for composite performance index.

Post Hoc Analysis on Performance Measures

Due to the lack of convincing results of the analyses on the performance measures, a post hoc analysis was performed on the data using question type as a dividing line. The results from the questions that did not require a Boolean 'Or' were considered simple and those requiring this solution were deemed complex. A summary of the analysis of the data for these two groups of questions is shown in table 10. The full ANOVA summary tables are provided in Appendix E

Table 11: Summary results (F values) for post hoc analysis.

		Question Type	
		Simple	Complex
Time	List type	.33	2.24
	Boolean Operations	.25	2.01
	Interaction	.25	.07
Correct	List type	.02	.88
	Boolean Operations	**4.23**	**10.37**
	Interaction	.14	1.37
Cognitive Level	List type	.00	.66
	Boolean Operations	1.21	2.92
	Interaction	.04	1.54

Note: Bold face type denotes significance at $p<.05$.

Appendix F contains an analysis of the data for the individual questions to which the subjects responded.

Satisfaction

Satisfaction data were collected at the end of the experiment through a series of bipolar questions based on a subset of the QUIS. The analysis revealed a significant interaction for list type by Boolean operations. The main effects were not significant.

Table 12: Summary table of ANOVA for satisfaction.

Source	SS	df	MS	F
List Type	1.71	1	1.71	.75
Boolean Operations	.46	1	.46	.20
Interaction	11.39	1	11.39	5.02*
Error	154.24	68	2.27	
Total	168.19	71	2.37	

- p < .05

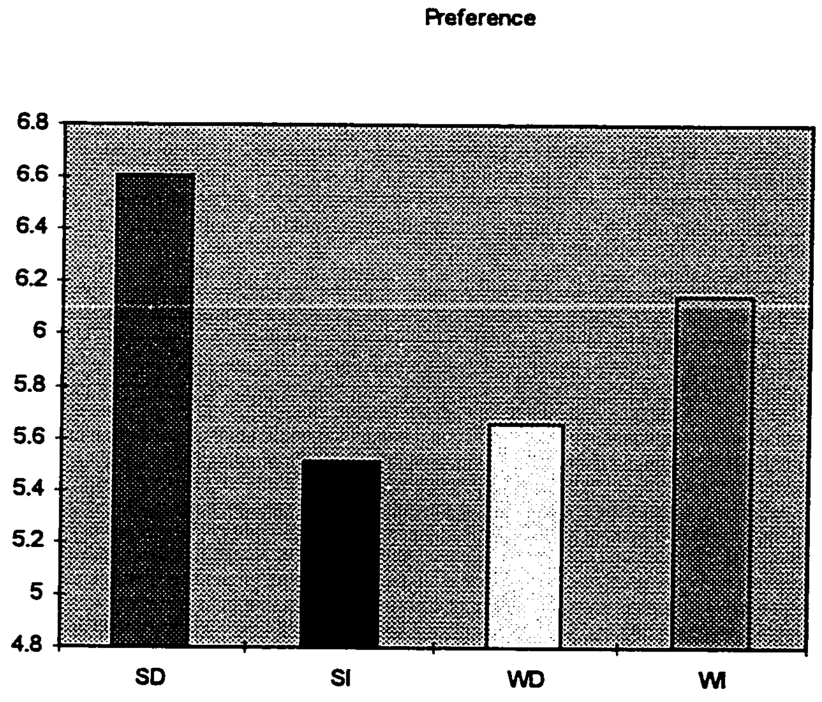

Figure 20: Bar graph of cell means for the satisfaction measure.

Exploratory Analysis

The results of the exploratory data collection on mental workload are shown below. A factorial ANOVA was performed using the SWAT data as the dependent variable with the experiment's independent variables. The analysis revealed no significance for either the main effects or the interaction.

Table 13: Summary table of ANOVA for mental workload.

Source	SS	df	MS	F
List Type	1.27	1	1.27	1.82
Boolean Operations	.43	1	.43	.62
Interaction	.03	1	.03	.05
Error	47.52	68	.40	
Total	49.28	71	.69	

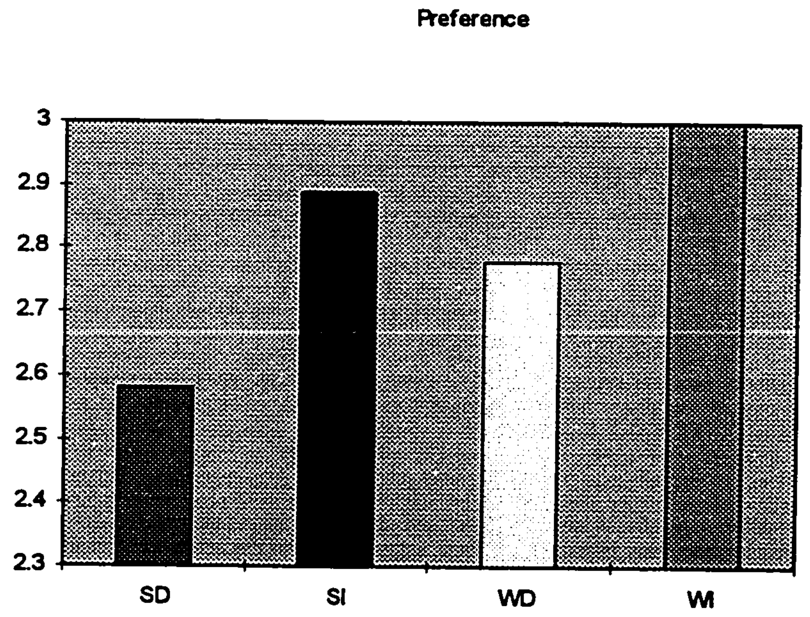

Figure 21: Bar graph of cell means for the mental workload measure.

Chapter 5 - Discussion

The results of the study were less than convincing with respect to the predictions. The results are discussed based on the following template. First, a wrap up of the support for the hypotheses is given. Next, an examination of the results from the point of view of the dependent measures is provided. This is followed by a general overview of the experiment and study and concludes with a discussion of the limitations of the study as well as future considerations.

Support for Hypotheses

This study sought to extend knowledge in the area of database querying. In particular, it attempted to further investigate the area of direct manipulation interface design for database querying. The impetus for the study came from earlier work on the question of database querying and particularly from the work carried out at the University of Maryland on dynamic querying (Alhberg, Williamson, & Shneiderman, 1992; Williamson & Shneiderman, 1992). This latter work introduced the idea of coupling the direct manipulation approach to interface design with the problems of database querying.

Answers to two general questions were explored. First, would users be more satisfied, and would they perform better in, an environment for querying databases that allowed them to construct a linear list of search factors or would they be more satisfied and perform better in an environment that provided a convenient method of organizing the search factors into subsets or groups. Second, would users be more satisfied and perform better in an environment that allowed them to construct queries requiring complex Boolean logic directly or would an environment that emulated this feature in an indirect, and less complex manner, result in greater satisfaction and performance.

Given the above the following hypotheses were made:

Hypothesis 1.1: There will be a significant difference in performance between the scrolling extension and the windowing extension of the factor display variable.

Support was not found for this hypothesis. Whether performance was measured in time to complete the task or the number of correct responses to the test questions, proficiency, subjects showed no difference across the factor display variable.

Hypothesis 1.2: There will be a significant difference in satisfaction between the scrolling extension and the windowing extension of the factor display variable.

Support was again not found for this hypothesis directly. The analysis showed no significant difference between these levels for the satisfaction measure. It did reveal, however, a significant interaction with the other main independent variable

The second design variation, direct versus indirect Boolean 'Or' facilities, was empirically derived.

Hypothesis 2.1: There will be a significant difference in performance between the direct and indirect Boolean 'Or' operations.

Support for this hypothesis is equivocal. If the measure of performance is time to complete the task then support cannot be found. However, if performance is measured by correct answers to experimental questions, proficiency, then support is found. Subjects performed significantly better when using an interface that employed a direct implementation of this capability. The question is, which measure is the better. If response time is paramount then obviously the time measure is more important, but if proficiency, as measured by correct answers, is the goal then the second performance measure is the correct yardstick.

Hypothesis 2.1: There will be a significant difference in satisfaction between the direct and indirect Boolean 'Or' operations.

The analyses did not reveal direct support for this hypothesis in isolation; there was, however, support for the interaction between this variable and the list type variable. Whether subjects preferred the scrolling or windowing extension of the factor display variable was dependent on whether they were engaged in the direct or the indirect Boolean 'Or' implementation. Likewise, whether subjects preferred the direct or indirect Boolean operation was dependent on whether they were using a scrolling extension or a windowing extension for search factor display. Within the scrolling implementation subjects preferred the direct approach to forming Boolean queries, whereas, within the windowing implementation the subject preferred the indirect approach.

So, support for the hypotheses can be described, at best, as equivocal. The type of list the users employed for constructing searches resulted in no difference in performance. Whether performance was measured in time or proficiency remained irrelevant to this conclusion. The implementation of the Boolean operations did not have such a clear cut outcome. When performance was measured in elapsed time, no difference was found for the effect of this variable. However, when proficiency was the measure of performance then a significant difference did exist. This difference favoured the direct implementation. In a like manner, no significant differences existed for the two variables with respect to satisfaction in isolation. There was, however, a significant interaction between these two variables and satisfaction.

Performance

As was mentioned earlier, performance was measured by two indices - time and proficiency. In addition, a composite index was computed based on an equal weighting of the performance on both of these measures.

With respect to the time measure no statistically significant differences were found in the performance of the subjects. It appeared that either style of list type or Boolean operations had little effect on the subjects' performance. That is, the time to complete a task was independent of the interface type for both variables. An examination of the graph for the time performance does reveal that there was a difference in performance across these variables, however it was not statistically significant. Subjects performed the tasks in less time in the direct condition of the Boolean operations variable and they performed more quickly in the windowing level of the list type variable. There is no indication of an interaction between these two variables. One possible explanation for results such as these is the large amount of within subjects variability. In other words, there was a great deal of difference between the performance of the subjects irrespective of the condition in which they participated. The study attempted to control for this phenomenon by selecting subjects from a highly homogeneous pool, namely masters level business students. Though the primary reason for choosing from this pool was to maximize the generalizability of the study an ancillary benefit was a sample of subjects expected to be alike in many respects. In fact, an analysis of the demographic data revealed such an occurrence with no differences found between the subjects across conditions on a number of variables. The subjects were alike on issues of age, work experience measured in years, computer experience measured in years, and database experience also measured in years. So an attempt was made to control for the inter-subject differences. However, the subjects still differed on time performance. One possible explanation could be the differing learning styles of the subjects. An informal observation of the subjects during the instructional phase of the experiment revealed large discrepancies. Some subjects read through the instructions and completed the instructional tasks without incident and moved seamlessly to the test part of the session. Others showed more trepidation in approaching the task. They took longer to move through the instructional phase and asked the experimenter more questions. They seemed less likely to experiment with the task. One possible solution for this problem might be to employ a more demonstrational instruction phase. The subject would simply observe the system solving problems and learn the operation from this interaction. It is also possible that subjects differed on their proficiency with windows style interfaces and this could have contributed to the variance of the data. A possible solution would be to have the subjects perform an independent task before the experiment to ascertain their level of functioning with the general operation of the computer and use this measure as a covariate in the analysis of the performance data. Time, as a dependent measure, is highly susceptible to differences in general ability with the computer interface and individual differences among subjects.

An alternative to elapsed time for measuring performance is proficiency or number correct. In the study subjects had to respond to eight questions by generating a list of entities that satisfied the criteria provided in the question. The subjects' scores could range from zero to eight. An analysis of the data revealed that all subjects managed to answer at least one question correctly. A summary of the responses is provided in Appendix F. The analysis of the data revealed no significant difference between the list type variable for the number of correct responses. Subjects performed the same whether they formed queries in groups of factors or added them to a list. There was also no significant interaction. On the other independent variable, Boolean operations, a significant difference was found. That is, subjects performed significantly better when using the direct method of inserting Boolean *'Ors'* to satisfy the given criteria than when using an iterative approach of performing multiple intermediate searches. It should be noted that this is in direct conflict with what the direct manipulation literature suggests (Alhberg, Williamson, & Shneiderman, 1992; Williamson & Shneiderman, 1992). Other research, however, has pointed to the fact that subjects, when given the choice, will opt for the complex solution rather than multiple simple solutions (Katzeff, 1986). One possible explanation for this difference in performance across Boolean operations is that users have a better understanding of Boolean logic than thought. The explosion of the internet, with the associated search engines, and the increase in reference CD-ROM products has exposed many more students to the type of logic required to retrieve information from electronic sources. Any conjecture on the logic abilities of users based on the results of this study must be tempered, however, by the recognition that the subjects were highly educated having completed at least an undergraduate degree before taking part in the study. On the other variable, list type, the results were not so promising. It is possible that the task difficulty was not at such a level as to reveal this effect. Perhaps if the subjects had had to form queries with more factors than those required in the study some difference might have been observed.

Since equivocal results were found across the two performance measures a composite index was formed based on the subjects responses. This index, an equally weighted composite based on normalized scores for both measures, was subjected to the same analysis. Results followed that of the time performance measure. No significant differences were found for either independent variable or for the interaction at the $p<.05$ level. However, the probability levels for both variables were in the .10 range. An examination of Figure 15, the graph of the results, shows differing performance levels for both variables but none that reach significance.

Post Hoc Performance

Due to the nature of the analysis on the performance measures it was decided to undertake a post hoc analysis of the data with respect to question difficulty. An examination of the test questions revealed a discrepancy with respect to question difficulty. Some questions, namely 1, 2, 3 and 8, did not require a Boolean 'Or' to solve correctly while the remaining four questions did. The analyses for the performance measures were rerun with two separate groups of questions, simple and complex. The results for this analysis are shown in Table 10. The analysis revealed no substantive change as compared to that of the full set of questions. For the time measure, no significant results were found for either independent variable for the simple questions or for the complex questions. The interaction also remained non significant across question type. Much the same was found for the number correct performance measure. The significant difference on the Boolean operations variable was supported across both question types. As well, no significant difference was found for the list type variable. One promising result from the analysis across question type was the direction of the F values as the question type moved from simple to complex. In all but one case, the time by satisfaction interaction, the F value increased as the question type became more complex. Although the coefficients did not move enough to reach significance this may be an indication that the tasks were too simple for the subject population.

Satisfaction

Data collected on satisfaction were subjected to the same analyses across the two independent variables. Results here showed no significant differences for the main effects of either variable. Subjects rated their satisfaction the same for both implementations of the search factor list and both implementations of the Boolean operations. However, the results did show a significant interaction between these two variables on the issue of satisfaction. When subjects used the scrolling implementation of the list type they preferred the direct implementation for Boolean operations and when they used the windowing type of list they preferred the indirect approach to Boolean operations. This is an interesting result and runs counterintuitive to what one might expect. In the windowing implementation the direct Boolean operation is automatic, the user just has to place the search factors in different groups, no other interaction is required. In the scrolling implementation, the user is required to explicitly change the value of the Boolean operator to achieve this outcome. Therefore, one would expect less direct effort required in the first case. The results, however, as mentioned do not support this contingency. In fact, the opposite is true. A number of possible explanations can be proffered for this result. First, the subject may have found the additional navigation between groups in the windowing interface to be onerous or bothersome, thus preferring the indirect approach to Boolean operations since in this instance it would be possible to solve four or less factor queries without resorting to using the additional navigation tools. On the flip side, the users of the scrolling interface may have preferred to have a complete representation of the query placed in front of them for inspection. Some earlier research has pointed to users preferring this approach.

Exploratory - Cognitive Load

An investigation of the mental workload or cognitive load experienced by users was also explored in the experiment. Results of responses to the SWAT subscale were analyzed in a like manner to the other dependent variables. That is, the responses were compared across the independent variables of list type and Boolean operations. The analysis showed no significant differences across theses variables for either main effects or the interaction. When the analysis was rerun with the dichotomy of simple versus complex questions in effect these outcomes did not change, although the Boolean operations reached marginal significance. This is an interesting finding in that all combinations of the variables received very similar scores. One might be tempted to conclude that the measure was not sensitive enough for the task as a plausible explanation. However, a look at the intercorrelation table shows a strong negative correlation between cognitive load and number correct. It appears that the scale was sensitive enough to capture the association between an increase in mental work and a decrease in performance across the entire experiment. Another possible explanation for the lack of significant differences could be that users of all combinations were comparing the interface to other querying systems or to other application software. There is no evidence for this, but, anecdotal conversations with the subjects did point to a feeling that the software was easy to use and a pleasure to use. Only future research, incorporating this type of comparison can answer this question conclusively. It is also possible that users were engaging in a central tendency response set in answering the mental workload question (See Appendix G for a breakdown of the cognitive level responses). It is also possible that the tasks were not of sufficient difficulty to tax the cognitive resources of the subjects. In other words, they found the experiment, regardless of condition, too easy. Referring to the proposed graph (Figure 11), the subjects were operating in the left hand side of the curve. That is, the tasks were not difficult enough to produce a level of mental workload to impact performance. Again, there is no hard evidence for this, but it is something that might be addressed in future studies. Perhaps placing the subjects under severe time constraints could further elucidate the presence or absence of an effect.

General Observations

This study was carried out to investigate the effects specific interface design variations have on database querying. The study was modeled on earlier investigations of this problem. It followed existing studies in some respects and diverged from these studies in others. With respect to subject population the present study utilized graduate students, primarily those enrolled in an MBA program. As stated, the reason for this decision was to closely emulate the type of user who might engage such a system in the real world. That is, users were expected to understand the data contained in the database but would not be expected to be database experts. This is not to say that only persons with graduate degrees will profit from research into database querying but rather that in this type of environment the user is likely to have some acquired knowledge of the content. Some earlier studies have taken this approach whereas others have simply used the available subject pool, making no claim as to ability of the subject pool. In fact, this second approach, that of using the college sophomore, has been the most popular. Clearly, of the studies reviewed the vast majority have used university undergraduates. In some cases, these subjects have been chosen from the ranks of the information management or management information systems courses while in others it appears that psychology undergraduates have been the most popular. In only one reviewed study, that performed by Burgess (1991), did the subject pool approximate the intended population. Burgess used working clinicians to test a system designed for this population. In other studies the sample has ranged from high school students (Essens, McCann, & Hartevelt, 1991) to female homemakers and part time workers recruited through the local newspaper (Green, Devlin, Cannata, & Gomez, 1990).

In terms of sample size this study falls approximately in the middle when compared with existing studies. Some early studies used extremely small samples, in the order of 12 subjects while others have employed large samples in the 150 subject range. Any results of these small sample studies must be tempered by the fact that extremely low statistical sensitivity was present. The predictions made in this study were not all supported, it seems unlikely, however, that this was a product of low sensitivity. The sample size of 72 used in this study appears sufficient for most factorial designs involving two factors with two levels each.

One area in which this study did differ from others was in the presentation medium for the test system. In a number of earlier studies the medium of choice was a paper and pencil mockup of the experimental system. This paper and pencil approach, although a necessary first step in the design of any system, should not supplant the use of a real, functioning system in the testing phase. Some researchers even went so far as to make the point that too many studies on database querying employed this surrogate approach (c.f., Davis, 1989). It was based on this, and the characteristics of the experimental system, that the study delivered the test material and collected responses through the electronic medium.

On the issue of instructional time this study also departed from many of the reported works. Some studies carried out on the querying of databases used long term classroom instruction as a requisite part of the investigation. In fact, upwards of 12 hours were devoted to this phase in some studies. In the present study, although not empirically derived, instructional time was very short. The majority of the subjects completed the entire task in under an hour. This included orientation, demographics, instruction, testing, and satisfaction questionnaire. Informal observation of the process revealed that most subjects reviewed and mastered the instructions in approximately 30 minutes. This represents an extremely short instructional time given that the subjects had not been introduced to the system previous to the testing session. Unfortunately precise timing of this phase was not captured and the association with the subject's performance could not be determined. Additionally, this short period of time for instructions for the group translated into fairly good performance for the group. The average number of correct responses across all conditions for the sample was 5.82 correct answers to eight query questions (approximately 73%).

Implications for Software Design

With results as such it is difficult to derive any clear cut design suggestions from the study. It does appear, however, that users can form queries based on complex Boolean logic in a direct manner. Therefore, the inclusion of this type of facility into querying systems for databases is warranted. One caveat to this suggestion is based on the user population. That is, the users of the software should closely exhibit the same characteristics as this test population. It should be noted that this is not the first study to conclude that users are capable of this type of responding; others have found similar behavior (Katzeff, 1986).

Conclusions and Limitations of the Study

As with any study, hindsight reveals a number of limitations. It appears that one major limitation in this investigation was the amount of variability between subjects. This was particularly acute in the measurement of elapsed time to respond to the query questions. Although an attempt was made to select subjects from a homogeneous pool there still appeared to be large differences in their timed interaction with the system. This variability is a problem whenever subjects' performance is measured and is exacerbated when that measure involves time. A larger sample might obviate some of this variability, but there is usually a practical limit to the amount of time that can be spent obtaining and testing subjects. There are restrictions on the total available pool as well as the inclination of the individuals within the pool to take part in such an endeavour. Other solutions are available to combat this variability. One might be to extend the training time or type of instruction to even the playing field for all subjects. It might be possible to provide instruction in a demonstrational interface rather than a narrative approach used in this study. Users might better learn the intricacies of computer interfaces through a 'show me' approach. Microsoft's wizards are a good example of this. It might also be possible to measure the subjects' interaction speed with some standard interface operations and use this independent measure as a covariate in the analysis of the experimental data.

It might also be the case that the tasks given the subjects were too easy. An approach that forces the user to devote more cognitive resources might reveal differences not apparent at the present task difficulty level. One proposition might be to make the questions more complex by either increasing the number of search factors the subject must operate on or by increasing the logic difficulty. Users could be expected to respond to questions requiring nine or ten search factors, thus further taxing their cognitive abilities. In a like manner questions could require more complicated Boolean logic in their solutions. Again, the effect could be users devoting more resources to answering the questions and interacting with the system. Another approach could be to tax the user's cognitive functioning by requiring the operation of a simultaneous task, a sort of forcing function. This approach is often used in cognitive psychology studies to reveal small, but real, effects. As an example, the user could simultaneously monitor and respond to an auxiliary signal, such as a tape recording of normal office background noise.

In summary, a number of possibilities exist for future research. First, the subject pool could be varied to ascertain whether similar results are found. Populations of interest could range from undergraduate business students to seasoned professionals from the world of work. Second, it would be interesting to vary both the instruction type and task difficulty as a search for confirmatory data. Third, the design could be changed to a within subjects design in an attempt to reduce the between subjects variability. Also, this design would control for the effects of extraneous comparisons in the cognitive load assessment. That is, each subject would be assessing the mental workload required , after the first condition, to the other conditions within the experiment and not to some external standard. A search for a possible covariate could be another approach. People vary in their interaction with novel computer applications and this variability could serve as a useful covariate. Possibilities include the users' learning style, motor control, or risk taking behavior.

Appendix A: Screen Shots from experimental system

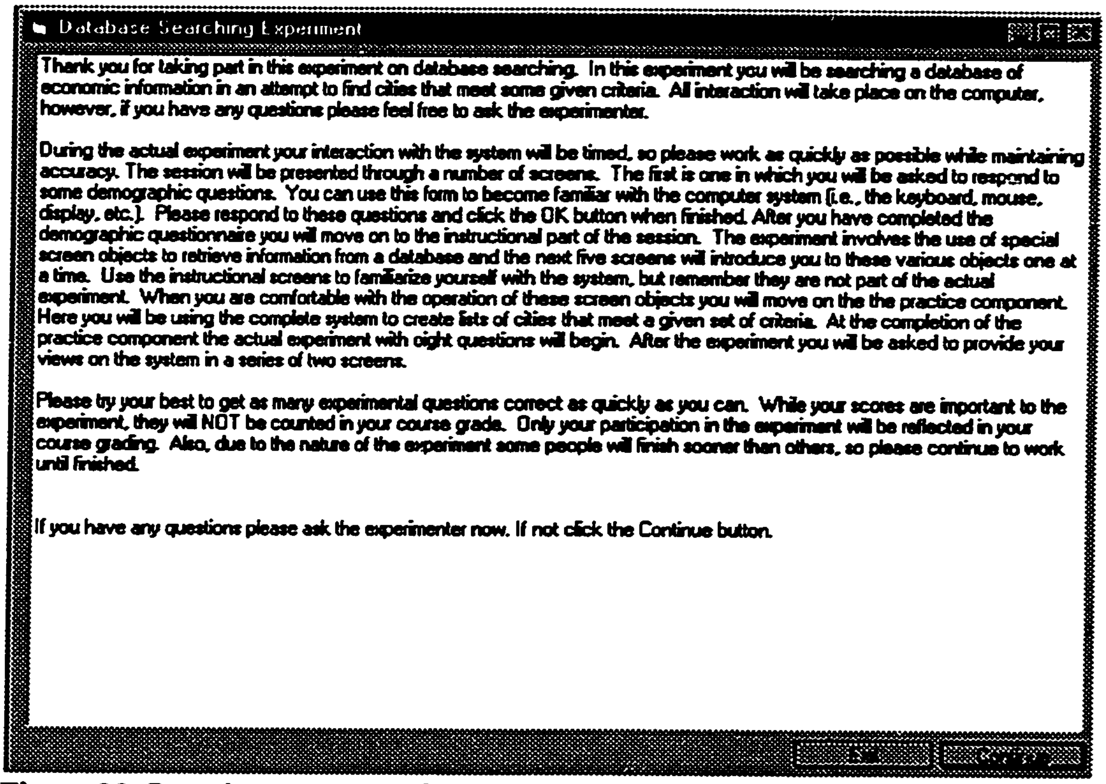

Figure 22: Introductory screen from the experimental system.

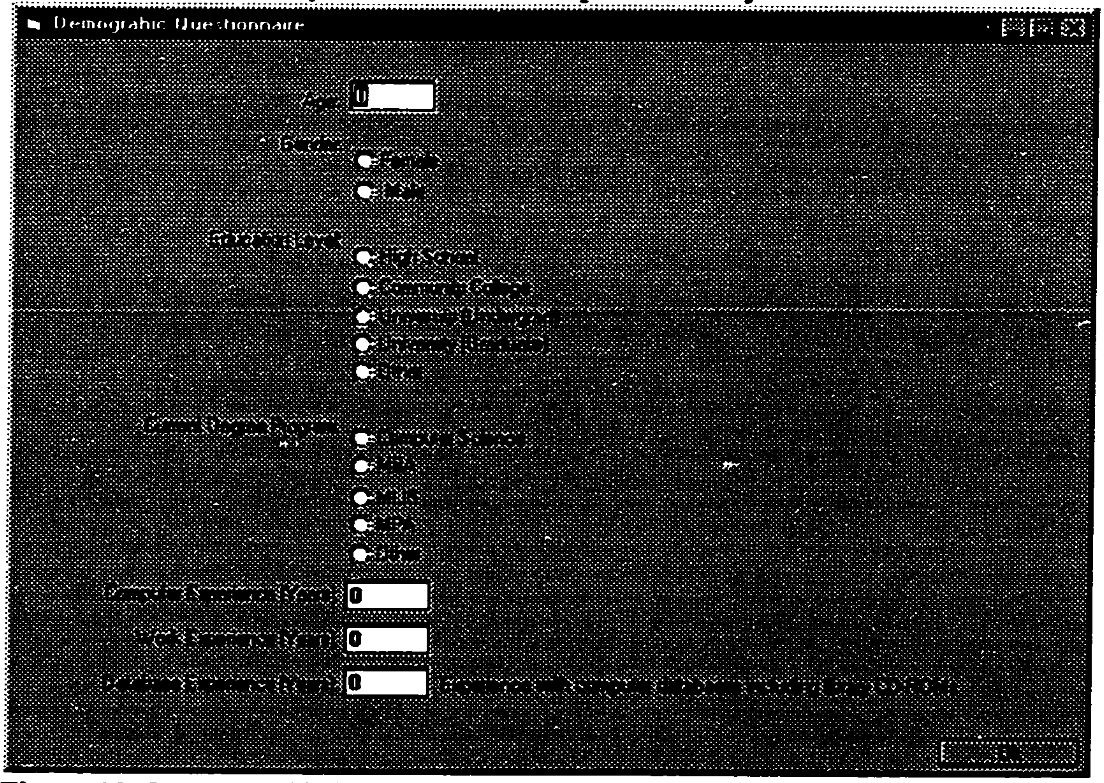

Figure 23: Demographic input screen from the experimental system.

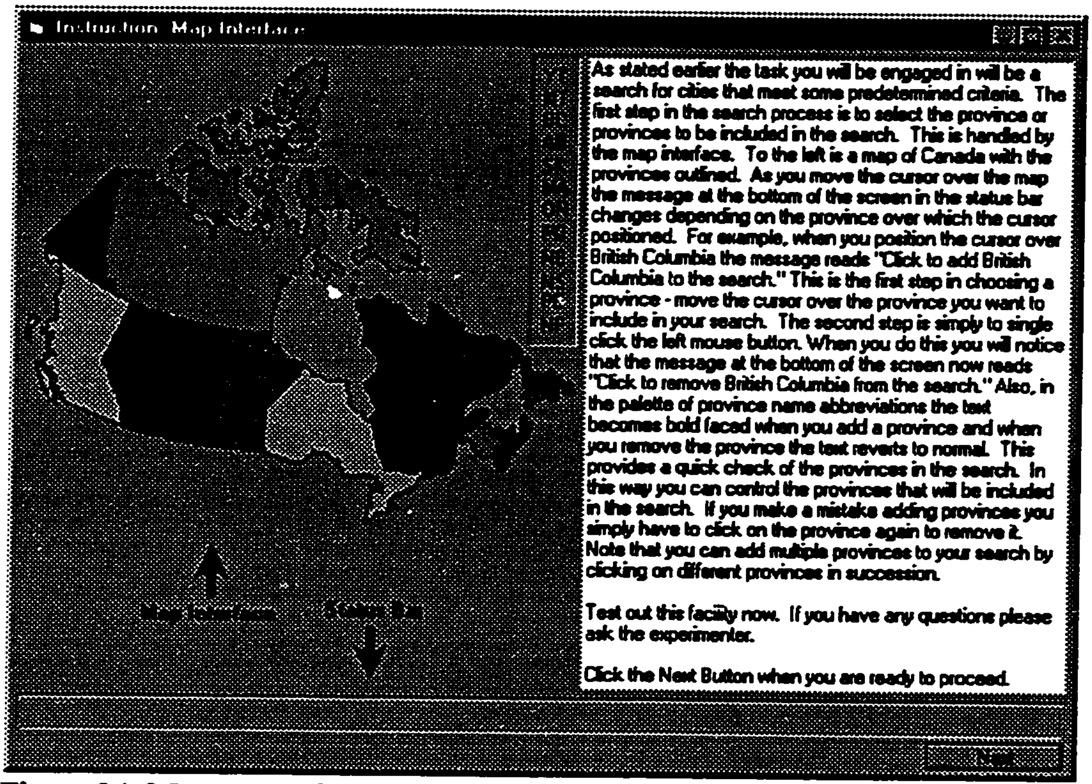

Figure 24: Map screen from experimental system.

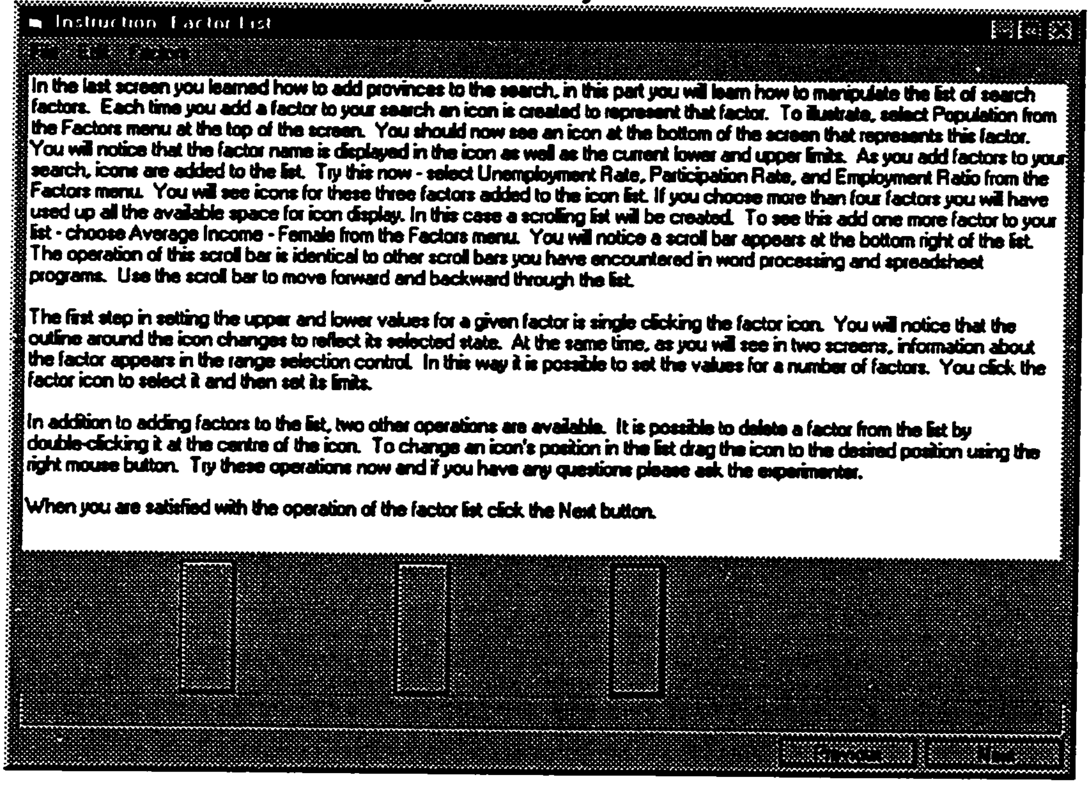

Figure 25: Introduction to factor operations screen from the experimental system. Screen from the scrolling and direct conditions displayed.

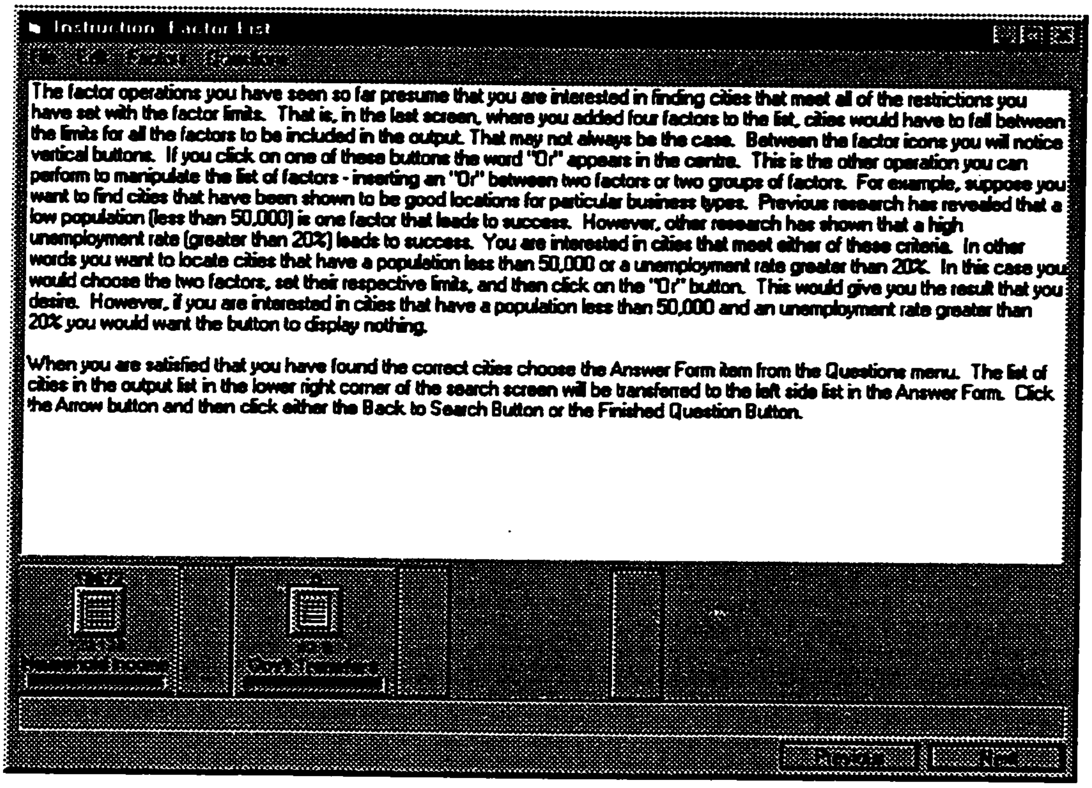

Figure 26: Introduction to Boolean operations screen from the experimental system. Screen from the scrolling and direct condition displayed.

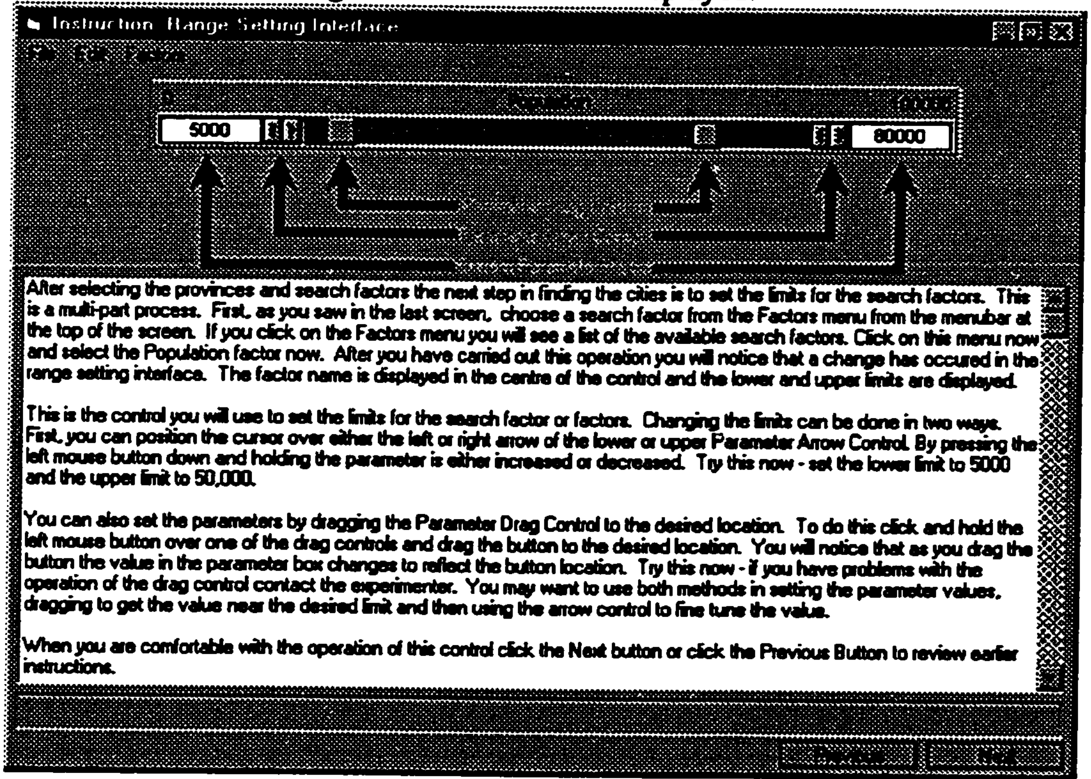

Figure 27: Slider control screen from the experimental system.

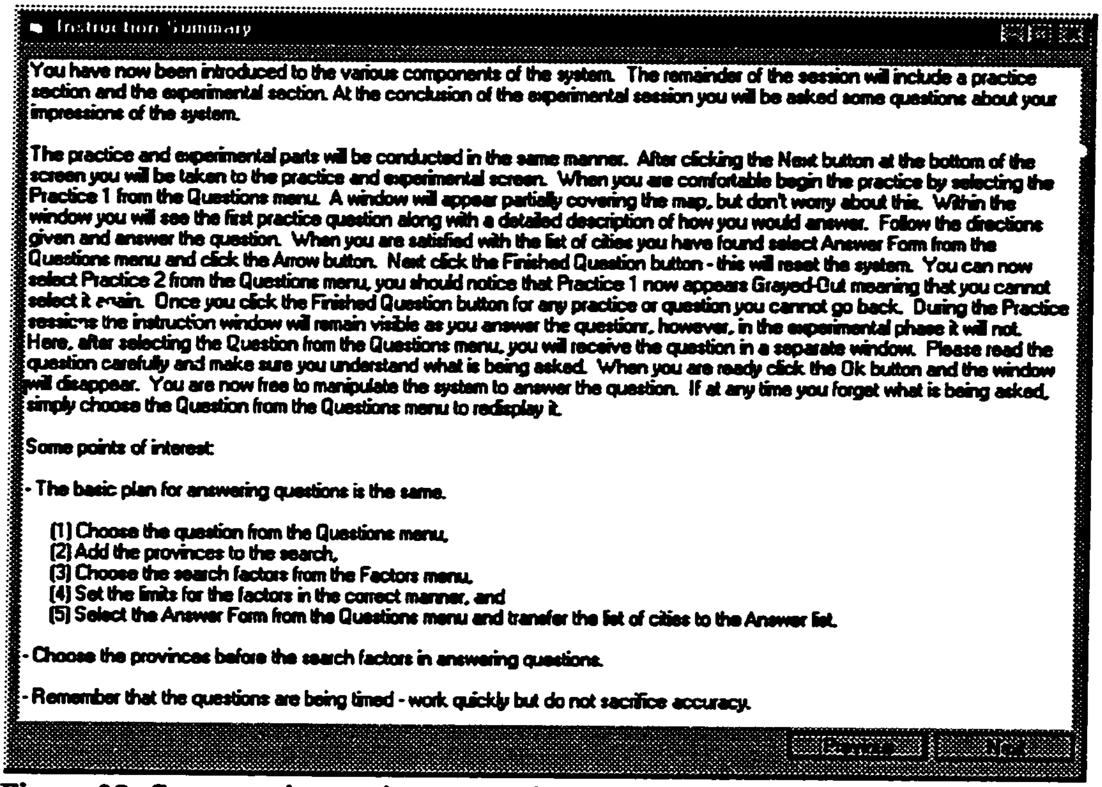

Figure 28: Summary instruction screen from the experimental system.

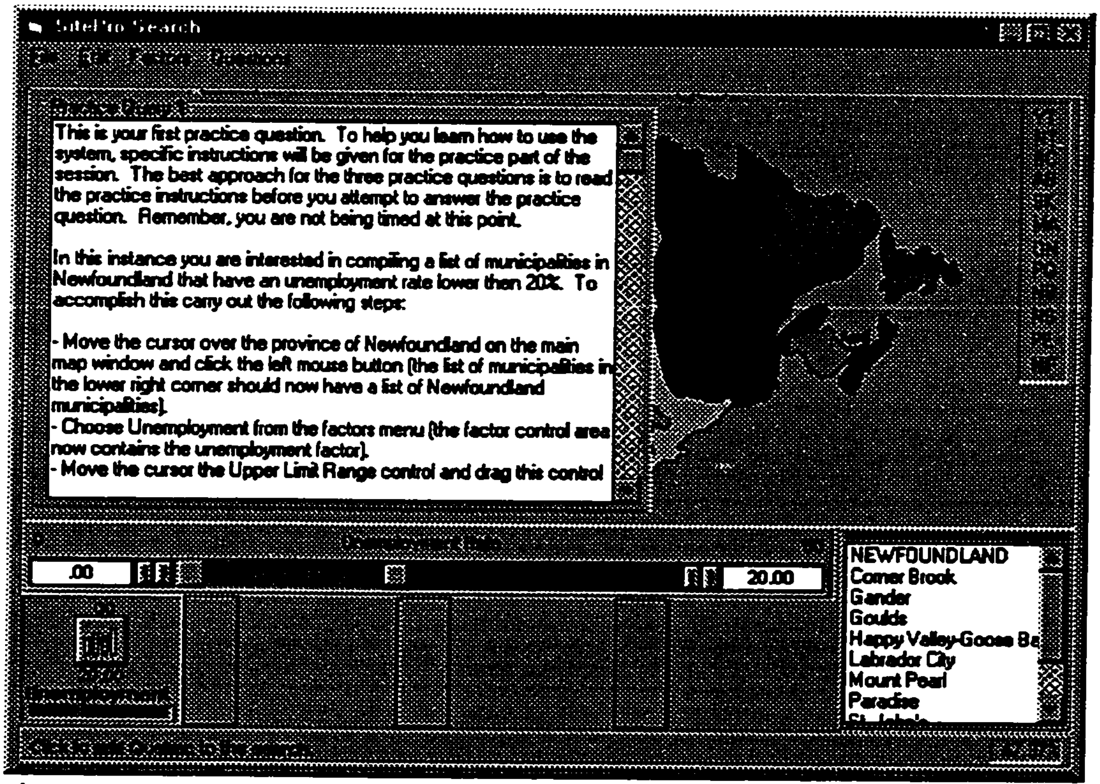

Figure 29: Screen from practice portion of experimental system. Shown is first practice screen from scrolling and direct condition with question solved correctly.

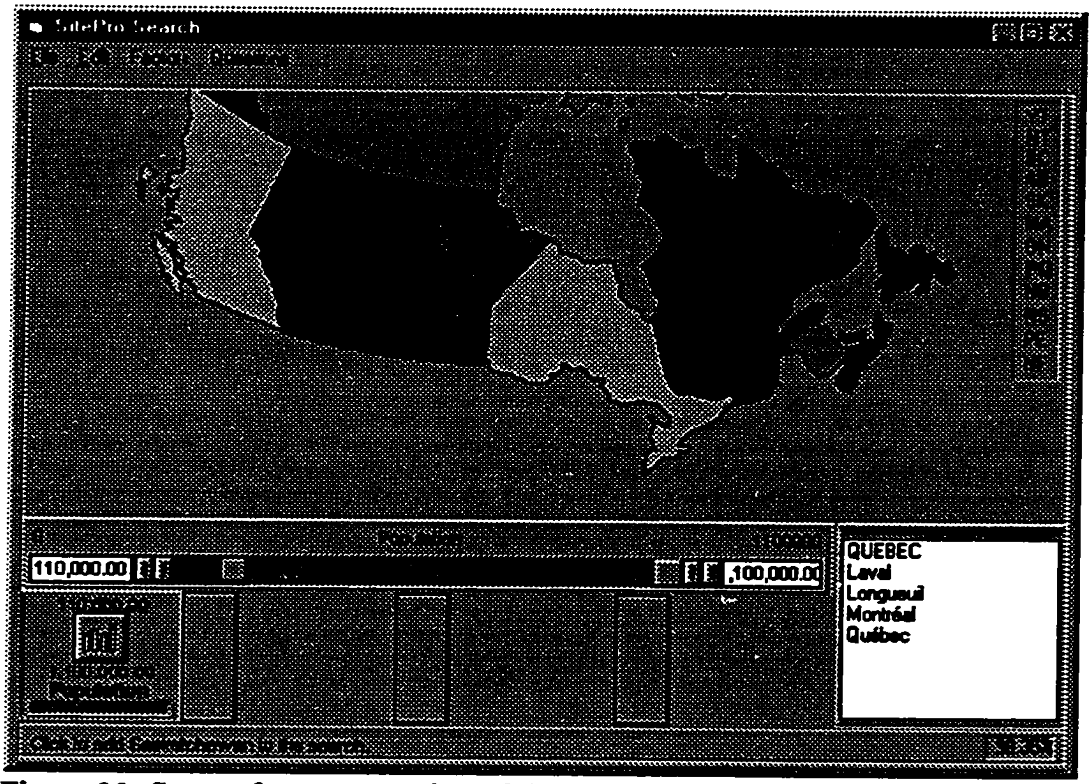

Figure 30: Screen from test portion of the experimental system. Shown is the correct response to question 1 of the test.

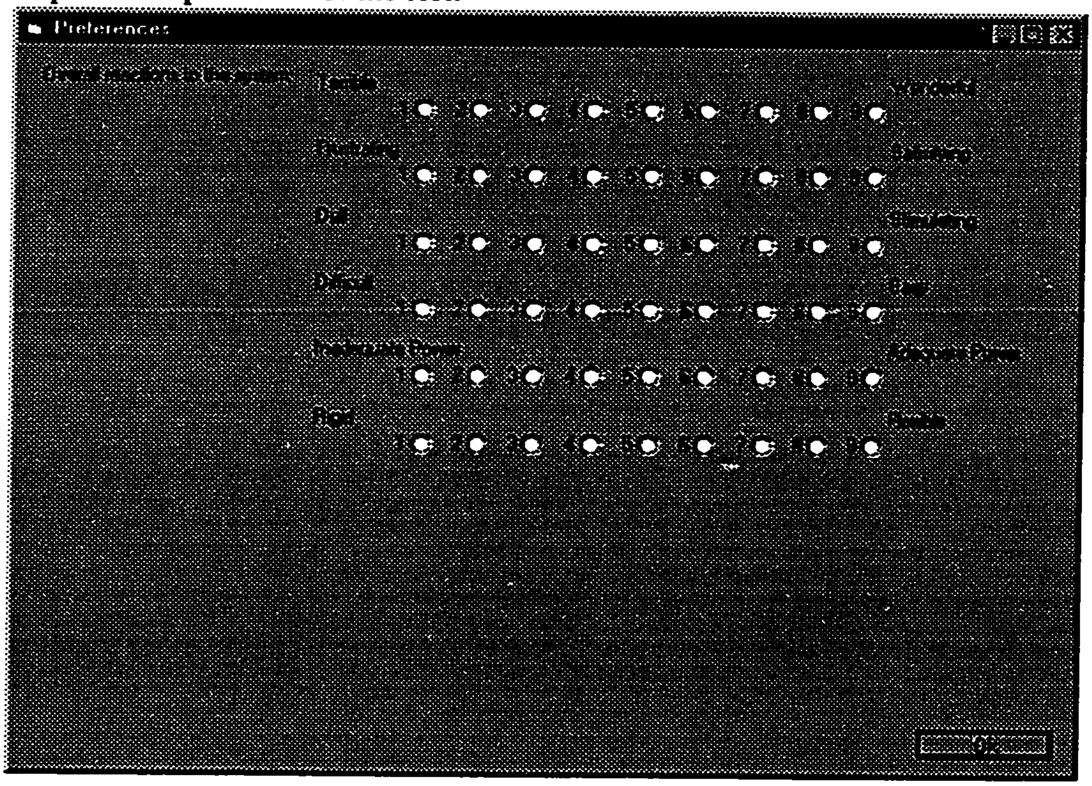

Figure 31: Preferences screen from the experimental system.

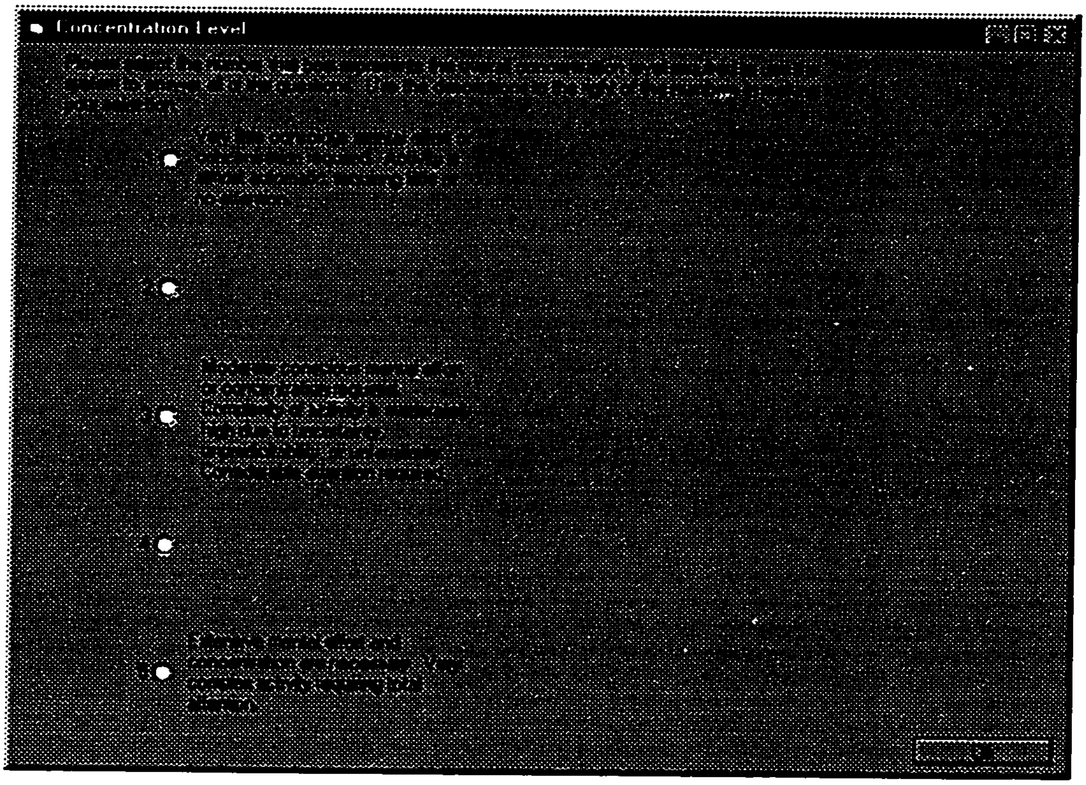

Figure 32: Concentration level screen from the experimental system.

Appendix B: Informed Consent

Consent Form

Please read and sign this form to indicate your agreement to participate in this study

I agree to take part in this experiment on database querying.

I acknowledge that my participation is voluntary.

I realize that I may discontinue my participation at any point within the experiment.

I expect that the experimenter will take every precaution to ensure the anonymity of the data.

I expect to be debriefed (as a group) at the end of the data collection and analysis phase of the experiment.

Name: ____ *(please print)*

Student Number: ______________________________________

Signature: ____

Appendix C: Full Text of Test Queries

Practice 1 This is your first practice question. To help you learn how to use the system, specific instructions will be given for the practice part of the session. The best approach for the three practice questions is to read the practice instructions before you attempt to answer the practice question. Remember, you are not being timed at this point.

In this instance you are interested in compiling a list of municipalities in Newfoundland that have an unemployment rate lower then 20%. To accomplish this carry out the following steps:

- Move the cursor over the province of Newfoundland on the main map window and click the left mouse button (the list of municipalities in the lower right corner should now have a list of Newfoundland municipalities).
- Choose Unemployment from the factors menu (the factor control area now contains the unemployment factor).
- Move the cursor the Upper Limit Range control and drag this control to the left until it is set at 20.
- The municipal list in the lower right corner now contains the requested information.
- Choose Answer Form from the Questions menu.
- Click the right arrow button between the two lists on the Answer Form.
- Click the Finished Question Button on the Answer Form.

After completing these steps you will have solved the first question. The basic steps to solving any question are the same: choose the province or provinces by clicking on the map; selecting the search factors from the factors menu; setting the limits for the factor or factors; and moving the found cities to the answer list on the answer form.

Go ahead and try the first question now. If you have questions, ask the experimenter. When finished with one question move on to the next by choosing it from the Questions menu.

Practice 2 In this case you are interested in finding cities in two provinces that meet multiple criteria

Find the list of cities in Quebec or New Brunswick that have a population between 11,000 and 33,000 and a household income greater than $50,000.

- The first step is to add the two provinces to the search - do this by single clicking Quebec and New Brunswick. You can check this operation by scrolling through the city list in the lower right corner of the screen or by checking the palette to the right of the map and ensuring that the PQ and NB are in bold face..

- Next, select the two search factors from the Factors Menu.

- Now set the limits for the two factors. Single click the Household Income icon and drag the lower limit control until you get the value 50,000. Next, single click the Population icon and drag the limit control to the desired values.

- When satisfied with the settings choose Answer Form from the Questions menu and click the Arrow button.

- Click the Finished Question Button.

Practice 3-1 In the last practice question you compiled a list of municipalities that met both the first criterion (Population between 11,000 and 33,000) and the second (Household Income greater than $50,000). In this practice question you are going to find a list of cities that meet either one criterion or another criterion.

Find the list of cities in Quebec or New Brunswick that have a Population between 11,000 and 33,000 or a Household Income greater than $50,000.

- Start this question in the same manner as the last one, click Quebec and New Brunswick to add these provinces to the search. Remember, you can verify which provinces are in the search by checking the palette beside the map for bold-faced abbreviations.

- Next select the search factors from the Factors menu. In this case you want Population and Household Income.

- Single click Household Income and set the lower limit to the desired value. In this case it is 50,000. Next click the Population icon and set the limits to the desired values. In this case 11,000 for the lower limit and 33,000 for the upper limit.

- The list now contains the cities that meet both criteria, but remember the goal is to find the list of cities that meet either one criterion or the other.

- Click the vertical bar between the two factor icons and the word "Or" should appear. You have now found the list of cities that have a Population between 11,000 and 33,000 or a Household Income greater than 50,000.

- Select Answer Form from the Questions menu and click the Arrow Button. You are now finished so click the Finished Question button.

Practice 3-2 In the last practice question you compiled a list of municipalities that met both the first criterion (Population between 11,000 and 33,000) and the second (Household Income greater than $50,000). In this practice question you are going to find a list of cities that meet either one criterion or another criterion.

Find the list of cities in Quebec or New Brunswick that have a Population between 11,000 and 33,000 or a Household Income greater than $50,000.

- Begin this question in the same way as the last, add the two provinces to the search and then add the two search factors - Population and Household Income.

- The next step is to set the limits for the factors. Remember, however, that you are going to have to do this using a two step approach. Start by single clicking on the Household Income icon and set its lower limit to 50,000.

- Now choose Answer Form from the Questions menu and click the Arrow button to transfer the list of cities and then click the Back to Search Button.

- Reset the limits for Household Income to the extremes.

- Single click on the Population icon and set its limits to the desired values. When satisfied select Answer Form from the Questions menu and click the Arrow button.

- After you transfer the city list for this second factor you are finished. Now click the Finished Question button to begin the first question.

Practice 3-3 In the last practice question you compiled a list of municipalities that met both the first criterion (Population between 11,000 and 33,000) and the second (Household Income greater than $50,000). In this practice question you are going to find a list of cities that meet either one criterion or another criterion.

Find the list of cities in Quebec or New Brunswick that have a Population between 11,000 and 33,000 or a Household Income greater than $50,000.

- Start this question in the same manner as the last - click Quebec and New Brunswick to add them to the search.

- The next step is to select the search factors from the Factors menu. Remember, however, that in this question you are interested in compiling the list of cities that meet either one criterion or another. To accomplish this you are going to add the search factors to different groups.

- Select Population from the Factors menu. You have now added Population to the first search group.

- Click the closed folder icon immediately to the right of the open folder icon below. You should now see an empty group. Select Household Income from the Factors menu.

- Set the limits for the Household Income factor - greater than 50,000.

- Click the folder icon for the first factor group. You should see the Population factor. Set the limits for the Population factor.

- Select Answer Form from the Questions menu and click the Arrow button.

Practice 3-4 In the last practice question you compiled a list of municipalities that met both the first criterion (pop between 11,000 and 33,000) and the second (Household income greater than $50,000). In this practice question you are going to find a list of cities that meet either one criterion or another criterion.

Find the list of cities in Quebec or New Brunswick that have a population between 11,000 and 33,000 or a household income greater than $50,000.

This is much like the last practice question except in one very important aspect, here you are interested in cities that either fall between the limits set for one factor or that fall between the limits set for the other factor.

Start the solution in the same way as before, single clicking Quebec and New Brunswick to add them to the search.

- Next select the search factors from the Factors menu - in this case Population and Household Income.

- At this stage things change a little. Since you are faced with determining the cities that meet the limits for one or another criterion you will have to take a two pronged approach. Start by single clicking the Household Income icon and dragging the lower limit control to 50,000. You now have the list of cities that meet this requirement in the City list. You now want to transfer this list to the Answer Form, but remember you are not finished with this question.

- Select Answer Form from the Questions menu and click the Arrow button and then click the Back to Search button.

- Now reset the limits for the Household Income factor to the extremes.

- Single click the Population icon and set the limits to the desired values. When satisfied with the settings choose Answer Form from the Questions Menu and click the Arrow button.

- You may notice that there are no duplicates in the answer list, don't worry the program is set up to function like this.

Question 1 Find the cities in Quebec that have a population greater than 110,000.

Question 2 Find the cities in British Columbia that have a population growth rate between 1.0 and 10.0.

Question 3 Find the cities in Nova Scotia that have a Household Income between 40,000 and 50,000 and an Unemployment Rate greater than 10.0.

Question 4 Find the cities in Alberta or Saskatchewan that have either Government Transfers less than 15.0 or a Population Density greater than 1000.0.

Question 5 Find the cities in Ontario that have either an Unemployment Rate between 5.0 and 10.0 and a Population greater than 220,000 or a Participation Rate greater than 80.0.

Question 6 Find the cities in New Brunswick that have either an Average Male Income less than 30,000 and a Male Workforce greater than 4,000 or a Average Female Income greater than 10,000 and a Female Workforce less than 1,500 or an Employment Ratio greater than 80.

Question 7 Find the cities in New Brunswick, Nova Scotia, or Prince Edward Island that have either Bilingual Speakers greater than 5,000 or a Household Income between 30,000 and 35,000 or an Unemployment Rate greater than 20.0.

Question 8 Find the cities in Saskatchewan that have a Trade School Graduate population greater than 900 and a University Graduate population greater than 15,000 and a Population Density greater than 1,000 and a Household Income greater than 44,000.

Appendix D: Calculation of Cronbach's Alpha for experimental data

Correlation matrix for preference data.

	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6
Question 1						
Question 2	.55					
Question 3	.68	.52				
Question 4	.60	.54	.57			
Question 5	.47	.35	.55	.49		
Question 6	.55	.36	.50	.51	.59	

Cronbach's Alpha: Equation

$$\alpha = \frac{[N\bar{p}]}{[1+\bar{p}(N-1)]}$$

$\bar{p} =$ mean interitem correlation

$N =$ no. of items in scale

Cronbach's Alpha: Computation

$$\alpha = \frac{[6x0.52]}{[1+0.52(6-1)]}$$

$$\alpha = 0.87$$

Appendix E: Summary Tables for Simple versus Complex Questions

Table 14: ANOVA for time performance on simple questions.

Source	SS	df	MS	F
List Type	2183.55	1	2183.55	.25
Boolean Operations	2931.15	1	2931.15	.33
Interaction	2165.72	1	2165.72	.25
Error	596743.44	68	8775.64	
Total	604015.88	71		

Table 15: ANOVA for time performance on complex questions.

Source	SS	df	MS	F
List Type	66873.07	1	66873.07	2.01
Boolean Operations	74393.86	1	74393.86	2.24
Interaction	2452.69	1	2452.69	.07
Error	2260353.03	68	33240.49	
Total	2404333.98	71		

Table 16: ANOVA for number correct on simple questions.

Source	SS	df	MS	F
List Type	.01	1	.01	.02
Boolean Operations	2.31	1	2.31	4.23
Interaction	.08	1	.08	.14
Error	37.12	68	.55	
Total	39.50	71		

Table 17: ANOVA for number correct on complex questions.

Source	SS	df	MS	F
List Type	1.58	1	1.58	.88
Boolean Operations	18.66	1	18.66	10.37
Interaction	2.46	1	2.46	1.37
Error	122.38	68	1.80	
Total	144.61	71		

Table 18: ANOVA for cognitive load on simple questions.

Source	SS	df	MS	F
List Type	.12.80	1	12.80	1.21
Boolean Operations	.04	1	.04	.00
Interaction	.38	1	.38	.04
Error	720.16	68	10.59	
Total	733.50	71		

Table 19: ANOVA for cognitive load on complex questions.

Source	SS	df	MS	F
List Type	34.13	1	34.13	2.92
Boolean Operations	7.73	1	7.73	.66
Interaction	17.94	1	17.94	1.54
Error	794.57	68	11.68	
Total	856.44	71		

Appendix F: Summary Analysis for Test Items

Table 20: F values for the dependent measures for each question.

	Performance - Time	Performance - Correct	Cognitive Level
Question 1			
List Type	.088	.091	.441
Boolean Operations	.174	.225	.609
Interaction	**4.717**	.154	.639
Question 2			
List Type	.009	2.396	.662
Boolean Operations	2.381	2.396	.662
Interaction	.684	2.396	1.373
Question 3			
List Type	2.723	.945	2.086
Boolean Operations	.517	3.569	.737
Interaction	.402	.147	.141
Question 4			
List Type	.001	.074	**5.858**
Boolean Operations	.588	.037	.250
Interaction	.059	1.949	1.033
Question 5			
List Type	.049	1.080	**3.853**
Boolean Operations	**3.940**	**9.140**	.189
Interaction	.377	1.830	.325
Question 6			
List Type	**4.651**	.005	1.567
Boolean Operations	.380	**3.991**	.066
Interaction	.005	.609	2.476
Question 7			
List Type	1.873	1.768	.368
Boolean Operations	2.474	**20.242**	1.836
Interaction	1.497	.134	1.142
Question 8			
List Type	.073	.513	.356
Boolean Operations	.650	1.803	.072
Interaction	1.703	.324	.180

Appendix G: Responses to Cognitive Load Questions

Table 21: Subjects' responses to cognitive workload question.

Response Level	Descriptor	Number
1	Very little conscious mental effort...	4
2		21
3	Moderate mental effort...	32
4		15
5	Extensive mental effort...	0

Appendix H: Raw data from the experiment

Subject	List Type	Boolean Ops	Time	Correct	Composite	Preference	Cognitive Level
31	1	1	893.63	7	1.28	7.33	3
35	1	1	862.83	7	1.14	6.83	3
41	1	1	952.25	6	0.92	8	3
42	1	1	565.79	6	-0.77	5.33	3
50	1	1	789.17	7	0.82	6.17	2
55	1	1	700.51	5	-0.79	4.5	4
61	1	1	1078.90	6	1.48	8.5	2
65	1	1	905.36	7	1.33	6.67	3
70	1	1	910.19	6	0.74	7.5	2
72	1	1	1170.20	4	0.66	5.67	3
75	1	1	679.65	4	-1.49	6.17	3
77	1	1	992.53	1	-1.96	7.33	3
78	1	1	756.12	7	0.68	3.83	2
81	1	1	504.84	5	-1.65	6.83	1
83	1	1	673.17	7	0.31	7.83	2
90	1	1	1486.18	3	1.43	5	3
105	1	1	1147.39	6	1.78	7.67	1
113	1	1	875.89	7	1.20	7.17	3
114	1	1	885.49	6	0.63	7.33	3
9	1	2	1087.68	7	2.13	7	4
25	1	2	676.05	5	-0.90	6.17	3
28	1	2	875.88	4	-0.63	5.33	4
44	1	2	1085.36	3	-0.33	5.5	3
45	1	2	842.34	4	-0.78	8	2
47	1	2	712.04	6	-0.13	1.5	1
51	1	2	1411.59	6	2.94	6.17	3
54	1	2	709.09	3	-1.98	7	3
57	1	2	871.18	6	0.57	6	3
64	1	2	1302.99	4	1.24	4.5	4
71	1	2	1260.97	6	2.28	6.67	2
93	1	2	1377.66	5	2.18	4.17	3
100	1	2	826.56	5	-0.24	7.33	2
101	1	2	945.12	5	0.28	6.17	3
102	1	2	819.53	5	-0.27	6.33	1
106	1	2	1254.15	2	-0.20	4.67	4
110	1	2	411.12	2	-3.89	2.83	4
115	1	2	875.62	3	-1.25	3.83	3
10	2	1	965.88	6	0.98	6	3
11	2	1	729.35	7	0.56	5	2
24	2	1	832.34	7	1.01	7.67	3
38	2	1	618.73	6	-0.54	6.17	2
39	2	1	881.17	6	0.61	6.33	3
40	2	1	781.19	7	0.79	7	2

49	2	1	602.15	7	0.00	6.83	2
53	2	1	1483.03	3	1.42	3.83	4
60	2	1	740.73	5	-0.61	5.67	3
62	2	1	585.11	6	-0.68	6.83	2
63	2	1	896.39	7	1.29	7.17	2
74	2	1	719.30	5	-0.71	6.17	2
76	2	1	612.48	5	-1.18	5.33	2
85	2	1	724.96	5	-0.68	4.83	3
88	2	1	624.07	5	-1.13	4.67	4
92	2	1	873.69	7	1.19	6.67	4
104	2	1	963.95	3	-0.86	5.17	3
111	2	1	1026.05	4	0.02	0.5	4
13	2	2	1112.95	5	1.02	4.83	4
27	2	2	586.83	2	-3.12	7.5	3
32	2	2	747.85	7	0.64	5.33	3
34	2	2	907.37	4	-0.50	7.5	3
36	2	2	1037.44	2	-1.15	6.33	4
43	2	2	1148.39	4	0.56	7	2
48	2	2	608.04	3	-2.42	4.33	4
56	2	2	928.91	5	0.21	8.83	2
59	2	2	869.02	1	-2.50	4.5	4
66	2	2	633.83	4	-1.69	5.83	3
67	2	2	908.37	5	0.12	4.67	3
79	2	2	964.01	3	-0.86	7.83	2
86	2	2	973.53	4	-0.21	6.5	4
87	2	2	971.96	5	0.40	5.17	2
103	2	2	958.06	3	-0.88	7.5	3
107	2	2	1020.94	6	1.23	6.83	2
112	2	2	767.48	4	-1.11	4	3

References

Ahlberg, C., Williamson, C., & Shneiderman, B. (1992). Dynamic queries for information exploration: An implementation and evaluation. *Proceedings of CHI '92*, 619 - 626.

Astrahan, M.M. & Chamberlin, D.D. (1975). Implementation of a structured english query language. *Communications of the ACM, 18*, 580 - 588.

Bannon, L., Cypher, A., Greenspan, S, & Monty, M.L. (1983). Evaluation and analysis of users' activity organization. *Proceedings of the ACM CHI'83 Conference on Human Factors in Computing Systems*, 54-57.

Blum, M.L. & Naylor, J.C. (1968). Industrial psychology: Its theoretical and social foundations. Harper & Row: New York.

Boyce, R.F., Chamberlin, D.D., King, W.F., & Hammer, M.M. (1975). Specifying queries as relational expressions: The SQUARE data sublanguage. *Communications of the ACM, 18*, 621 - 628

Brewster, S.A., Wright, P.C., & Edwards, A.D. (1994). The design and evaluation of an auditory-enhanced scrollbar. *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, 211.

Brosey, M. & Shneiderman, B. (1978). Two experimental comparisons of relational and hierarchical database models. *International Journal of Man-Machine Studies, 10*, 625 - 637.

Burgess, C.G. (1991). A graphical, database-querying interface for casual, naïve computer users. *International Journal of Man-Machine Studies, 34*, 23 - 47.

Carmines, E.G. & Zeller, R.A. (1979). Reliability and validity assessment. London: Sage.

Casali, J.G. & Gaylin, K.B. (1988). Selected graph design variables in four interpretation tasks: A microcomputer-based pilot study. *Behaviour and Information Technology, 7,* 31-49.

Chamberlin, D.D., Astrahan, M.M., Eswaran, K.P., Griffiths, P.P., Lorie, R.A., Mehl, J.W., Reisner, P., & Wade, B.W. (1976). SEQUEL 2: A unified approach to data definition, manipulation, and control. *IBM Journal of Research and Development, Nov,* 560 - 575.

Chamberlin, D.D., Gray, J.N., & Traiger, I.L. (1975). Views, authorization, and locking in a relational data base system. *National Computer Conference,* 425 - 431.

Childs, D.L. (1968). Description of a set theoretic data structure. *Proceedings of AFIPS Fall Joint Computer Conference,* 557 - 564

Codd, E.F. (1970). A relational model of data for large shared data banks. *Communications of the ACM, 15,* 377 - 387.

Cox, T. (1978). Stress. University Park Press: Baltimore.

Date, C.J. (1986). An introduction to database systems. Addison-Wesley: Reading, Massachusetts.

Davis, J.S. (1989). Usability of SQL and menus for database query. *International Journal of Man-Machine Studies, 30,* 447 - 455.

Davis, J.S. (1990). Experimental investigation of the utility of data structure and E-R diagrams in database query. *International Journal of Man-Machine Studies, 32,* 449 - 459.

Dayton, T., Gettys, C.F., & Unrein, J.T. (1989). Theoretical training and problem detection in a computerized database retrieval task. *International Journal of Man-Machine Studies, 30,* 619 - 637.

Doan, K., Plaisant, C., & Shneiderman, B. (1996). Query previews in networked information systems. Proceedings of the Third Forum on Research and Technology Advances in Digital Libraries, ADL '96, IEEE CS Press.

Eggmeier, F.T. (1988). Properties of workload assessment techniques. In P.A. Hancock and N. Meshkati (Eds.), *Human Mental Workload*, Elsevier Science Publishers B.V.: North Holland.

Ehrenreich, S.L. (1981). Query languages: Design recommendations derived from the human factors literature. *Human Factors, 23,* 709 - 725.

Essens, P.J., McCann, C.A., & Hartevelt, M.A. (1991). An experimental study of the interpretation of logical operators in database querying. *Acta Psychologica, 78,* 201 - 225.

Ferguson, G.A. (1981). Statistical analysis in psychology and education. New York: McGraw-Hill.

Fogg, D. (1984). Lessons from a "Living in a Database" graphical query interface. *SIGMOD '84: Proceedings of the Annual Meeting,* 100-106

Greene, S.L., Delvin, S.J., Cannata, P.E., & Gomez, L.M. (1990). No Ifs, ANDs, or Ors: A study of database querying. *International Journal of Man-Machine Studies, 32,* 303 - 326.

Hansen, G.W. & Hansen, J.V. (1987). Procedural and non-procedural query languages revisited: A comparison of relational algebra and relational calculus. *International Journal of Man-Machine Studies, 26,* 683 - 694.

Hansen, G.W. & Hansen, J.V. (1988). Human performance in relational algebra, tuple calculus, and domain calculus. *International Journal of Man-Machine Studies, 29,* 530 - 516.

Harris, L.R. (1977). User oriented data base query with the ROBOT natural language query system. *International Journal of Man-Machine Studies, 9,* 697 - 713.

Harvey, L. & Rousseau, R. (1994). Development of text-editing skill: From semantic and syntactic mappings to procedures. *Human-Computer Interaction, 10,* 345-400.

Held, G.D., StoneBraker, M.R., & Wong, E. (1975). INGRES - A relational data base system. *National Computer Conference,* 409 - 416.

Holden, K.L. & O'Neal, M.R. (1992). The utility of various windowing capabilities for single-task and multi-task environments. *Proceedings of ACM CHI'92 Conference on Human Factors in Computing Systems,* 52.

Houwing, E.M., Wiethoff, M., & Arnold, A.G. (1993). Interface evaluation from users' point of view: Three complementary measures. *Proceedings of ACM INTERCHI'93 Conference on Human Factors in Computing Systems,* 197-198.

Irler, W.J. & Barbieri, G. (1990). Non-intrusive hypertext anchors and individual colour markings. *Proceedings of the ECHT'90 European Conference on Hypertext,* 261-273.

Itoh, Y., Hayashi, Y., Tsukui, I., & Saito, S. (1989). Heart rate variability and subjective mental workload in flight task validity of mental workload measurement using H.R.V. method. *Proceedings of the Third International Conference on Human-Computer Interaction, 1,* 209-216.

Jih, W.J., Bradbard, D.A., Snyder, C.A., & Thompson, N.G. (1989). The effects of relational and entity-relationship data models on query performance of end users. *International Journal of Man-Machine Studies, 31,* 257 - 267.

Katzeff, C. (1986). Dealing with a database query language in a new situation. *International Journal of Man-Machine Studies, 25,* 1 - 17.

Katzeff, C. (1988). The effect of different conceptual models upon reasoning in a database query writing task. *International Journal of Man-Machine Studies, 29*, 37 - 62.

Kelly, M.J. & Chapanis, A. (1977). Limited vocabulary natural language dialogue. *International Journal of Man-Machine Studies, 9*, 479 - 501.

Kim, H., Korth, H.F., & Silberschatz, A. (1988). PICASSO: A graphical query language. *Software Practice and Experience, 18*, 169 - 203.

Laurel, B., Oren, T., & Don, A. (1990). Issues in multimedia interface design: Media integration and interface agents. *Proceedings of the ACM CHI'90 Conference on Human Factors in Computing Systems*, 133-139.

Linde, L. & Bergström, M. (1991). Analysing the deep structure of queries: Transfer effect on learning a query language. *Acta Psychologica, 78*, 227 - 241.

Maier, D. & Ullman, J.D. (1983). Maximal objects and the semantics of universal relational databases. *ACM Transactions on Database Systems, 8*, 1 - 14.

McLeod, D. & Meldman, M. (1975). RISS - A generalized minicomputer relational data base management system. *National Computer Conference*, 397 - 402.

Michard, A. (1982). Graphical presentation of Boolean expressions in a database query language: Design notes and an ergonomic evaluation. *Behaviour and Information Technology, 1*, 279 - 288,

Miller, G.A. (1956). The magical number seven, plus or minus two: Some limits on our capacity to process information. *Psychological Review, 63*, 81 - 97.

Murata, A. (1991). Evaluation of mental workload in location task by HRV measures - Relation between work level and mental workload. *Proceedings of the Fourth International Conference on Human-Computing Interaction, 1*, 190-194.

Murata, A., Mikaye, S., & Kumashiro, M. (1989). Heart rate variability in remote manipulation system. *Proceedings of the Third International Conference on Human-Computer Interaction, 1,* 217-224.

Mylopoulos, J., Schuster, S., & Tsichritzis, D. (1975). A multi-level relational system. *National Computer Conference,* 403 - 408.

Olsen, K. A., Korfhage, R. R., Sochats, K. M., Spring, M. B., & Williams, J. G. (1993). Visualization of a Document Collection: The VIBE System. *Information Processing & Management, 29,* 69-81.

Ogden, W.C. (1986). Implications of a cognitive model of database query: Comparison of a natural language, formal language and direct manipulation interface. *SIGCHI Bulletin, 18,* 51 - 54.

Payne, S.J. (1994). Acquisition of display-based skill. *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems, 2,* 299-300.

Pointek, J. (1995). Data structures for dynamic query browsing of EOS data directories. Proceedings of NASA Science Information Systems Interoperability Conference.

Reid, G.B. & Nygren, T.E. (1988). The subjective workload assessment technique: A scaling procedure for measuring mental workload. In P.A. Hancock and N. Meshkati (Eds.), *Human Mental Workload,* Elsevier Science Publishers B.V.: North Holland.

Reisner, P (1981). Human factors studies of database query languages: A survey and assessment. *Computing Surveys, 13,* 13 - 31.

Reisner, P, Boyce, R.F, & Chamberlin, D.D. (1975). Human factors evaluation of two data base query languages - Square and Sequel. *National Computer Conference,* 447 - 453.

Rothnie, J.R. (1975). Evaluating inter-entry retrieval expressions in a relational data base management system. *National Computer Conference,* 417 - 423.

Saito, S & Taptagaporn, S. (1991). Pupillary reflexes and accommodation as physiological indices of visual fatigue due to VDT operation. *Proceedings of the Fourth International Conference on Human-Computer Interaction, 1,* 233-237.

Saito, S., Taptagaporn, S., Hirose, N., & Saito, S. (1991). Evaluation of the relationship between pupil movements and visual tasks. *Proceedings of the Fourth International Conference on Human-Computer Interaction, 1,* 238-242.

Senko, M.E., Altman, E.B., Astrahan, M.M., & Fehder, P.L. (1973). Data structures and acessing in database systems. *IBM Systems Journal, 12,* 30 - 93.

Shneiderman, B. (1987). Designing the user interface: Strategies for effective human-computer interaction. Reading, MA: Addison-Wesley Publishing.

Small, D.W. & Weldon, L.J. (1983). An experimental comparison of natural and structured query languages. *Human Factors,* 25, 253 - 263.

Tanin, E., Beiger, R., & Shneiderman, B. (1996). Incremental data structures and algorithms for dynamic query interfaces. *SIGMOD Record, 25,* 21 - 25

Thomas, J.C. & Gould, J.D. (1975). A psychological study of query by example. *National Computer Conference,* 439 - 445.

Thomas, J.C. (1977). Psychological issues in data base management. *Proceedings on Very Large Databases,* 169 - 185.

Wastell, D. (1991). Physiological measurement of cognitive load during interaction with process control displays. *Proceedings of the Fourth International Conference on Human-Computer Interaction, 1,* 228-232.

Welty, C. (1985). Correcting user errors in SQL. *International Journal of Man-Machine Studies, 22,* 463 - 477.

Welty, C. & Stemple, D.W. (1981). Human factors comparison of a procedural and nonprocedural query language. *ACM Transactions on Database Systems, 6,* 626 - 649.

Williamson, C. & Shneiderman, B. (1992). The dynamic homefinder: Evaluating dynamic queries in a real-estate information exploration system. *Proceedings of SIGIR '92 Conference,* 338 - 346.

Yamada, S., Hong, J., & Sugita, S. (1995). Development and evaluation of hypermedia for museum education: Validation of metrics. *ACM Transactions on Computer-Human Interaction, 2,* 284-307.

Zloof, M.M. (1975). Query by Example. *National Computer Conference,* 431 - 438.

Zloof, M.M. (1977). Query-by-Example: A data base language. *IBM Systems Journal, 4,* 324 - 343.

IMAGE EVALUATION TEST TARGET (QA-3)

150mm

6"

APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989