acs academic computing services **Newsletter**

Volume 24, Number 11 | University of Minnesota, Twin Cities | November, 1990

In this Issue

Learning about LOOKUP.
See page 255.

Interactive Access Update.
See page 268.

Database Computing

Peter J. Oberg
PJO@UMNACVW

R. W. Engles has provided us with a widely accepted definition of the *database*: "A database is a collection of stored operational data used by the application systems of some particular enterprise." In other words, it's a program that stores your data and makes it easy to get it back. A database is a collection of data stored on diskettes, disk drives, magnetic tape, or other media. Database programs retrieve, update, add, and delete the data. Many people can access this data and share it in different ways.

The data in a database is a set of related information pertinent to a research project or business operation. In a research project, for example, the data might include:

Recorded observations of the subject.

Computed data from initial data collected.

Transforming data or merging various data from several observations.

A database offers advantages over simple ASCII data files:

A database provides *data independence*—i.e., the organization of the data and access to it is built within the programs that run the database.

A database can reduce inconsistencies in data and provides integrity. A program that manages the database can, for example, be set to reject an employee who claims to have worked for more hours than exist in a week or fictitious employees who are not in personnel records.

A database can eliminate redundancy, reducing the quantity of data that actually needs to reside on disk. An application within a database can use one occurrence of an employee information record

continued on page 248

Help Page

ACS HELP-Lines

Central Systems (UX,VX,VZ,CA):

Software (including Graphics, Databases, and Statistics Packages), Hardware, Networking

626-5592	8 am to 5 pm, weekdays

Artificial Intelligence:

625-8332	3 to 4 pm, Monday, Wednesday, Friday

Liberal Arts, Text Analysis:

625-8332	3 to 4 pm, Monday, Wednesday, Friday

LUMINA (communications questions):

626-2272	8 am to 5 pm, weekdays

Lauderdale Tape Library:

626-1838	9 am to 3 pm, weekdays

Microcomputer and Workstation Networks Center

Micro HelpLine

626-4276	9 am to 4 pm, weekdays

Consulting

Walk-In Consulting

ACS, East Bank:

1 Nicholson Hall	10 am to 4 pm, Monday through Friday

Microcomputer and Workstation Networks Center:

125 Shepherd Lab	9 am to 4 pm, Monday through Friday

Electronic Mail Consulting

Consulting is now available via the mail facility on all ACS systems (the UX, VX, VZ, and CA). Send mail to user name CONSULT for questions after hours and for low-priority questions that are not critical to your immediate computing work. Replies will be sent to your account through the mail facility on your system.

Instructional Computing Consultant

Department instructors may call 626-0200 for assistance in choosing ACS systems (ENCORE/UNIX, VAX/VMS, CYBER/NOS), software, and for answers to any other inquiries on using computers for instructional computing.

Computing Information Center

1 Nicholson Hall, 625-7397, MAD@UMNACVX, MAD@VX.ACS.UMN.EDU
8 am to noon and 1:00 to 4:30 pm, Monday through Friday

Computing account and grant applications available for ENCORE, VAX, and CYBER computers.

Short course enrollment. Short course schedules and class descriptions available.

Assistance in ordering vendor documentation. Vendor documentation is not always available in the University bookstores and may be ordered directly from the company.

Complete documentation collection. Reference copies of vendor and all other documentation for ACS software.

Free ACS documentation. General information and central system information available.

Computing Newsletters. Subscribe to the *ACS Newsletter*. Newsletters from other computing centers are also available for reference.

acs

Academic Computing Services

Director, ACS: Michael Skow
Editors: Steven Brehe, Paula Goblirsch

The *ACS Newsletter* is published monthly by Academic Computing Services of the University of Minnesota, Twin Cities.

Direct comments, suggestions, articles, and announcements to the editors, ACS, Technical Publications, 2520 Broadway Drive, Lauderdale, MN 55113, or call (612) 626-1828, or 626-1093. SKB@UMNACVX/ SKB@VX.ACS.UMN.EDU, PMG@UMNACVX/ PMG@VX.ACS.UMN.EDU.

For a free subscription call (612) 625-7397, or send your name and address to the Computing Information Center, 1 Nicholson Hall, University of Minnesota, 216 Pillsbury Drive SE, Minneapolis, MN 55455. MAD@UMNACVX/MAD@VX.ACS.UMN.EDU. On-campus address changes *must* include your department name and address.

The University of Minnesota is committed to the policy that all persons shall have equal access to its programs, facilities, and employment without regard to race, religion, color, sex, national origin, handicap, age, veteran status, or sexual orientation.

The *ACS Newsletter* is produced on an Apple Macintosh IIcx running Microsoft Word and Aldus PageMaker software. Camera-ready copy is produced on an Apple LaserWriter Plus and then printed at the University Printing Dept.

Contents

<u>**SOFTWARE**</u>
Databases

LOOKUP

Math and Engineering

Statistics

<u>**SERVICES**</u>
Documentation

<u>**NETWORKING**</u>
Access

<u>**SYSTEMS**</u>
DEC Systems Support

VAX Cluster

continued from page 245

and refer to it from any application of the database, eliminating duplication.

A database allows users to share data, again eliminating the need to duplicate information.

A database provides security, allowing only authorized users to use the information.

Database Architecture

Database programs define an *architecture* within the database—they organize the stored information so that the users of the database can access the information quickly and efficiently. There are three general kinds of architecture.

1. The Hierarchical Organization

In the data records collected in a research study, an organizational structure is usually quite evident. In a controlled clinical experiment, for example, many records will be collected on each individual in the experiment. It makes sense to keep all of the records collected on an individual together as we would in a file folder. The collection of records on one individual is called a *case* in database terminology. The records within this case can be naturally organized into categories or record types (e.g., demographic information, follow-up outcomes, laboratory tests, etc.).

When there is more than one record of a particular type within a case (e.g., when more than one follow-up record has been collected), the researcher will choose a variable to distinguish one follow-up record from another.

So we've arrived at a logical organization of data records into two levels: first into cases and then into record types within cases. This type of organizational structure is called a *hierarchy* and we have organized our data for this study into a two-level hierarchical model. The hierarchical model is a very natural one and has been a very popular model for database management systems.

The hierarchical model is not limited to two levels; we could, for example, add another level to the hierarchy in the clinical trial example. Suppose that on some follow-up measurements we order a laboratory test when certain indicators are present. Thus, a lab record will be collected for some of the follow-up records. This lab record will be stored in a laboratory record type within the appropriate case. A particular laboratory record is owned by the follow-up visit record that generated it. We indicate this ownership by placing the laboratory records at a lower level in the hierarchy, below the follow-up records that own it.

Figure 1 illustrates the hierarchical model for the clinical trial we've just discussed.

We can see that a hierarchy creates a nested or tree-like structure among the records. As a tree is composed of a root and trunk section that divides into branches and those continue to divide until they finally terminate in the leaves, so the records of a study can often be organized hierarchically as shown in the above figure.

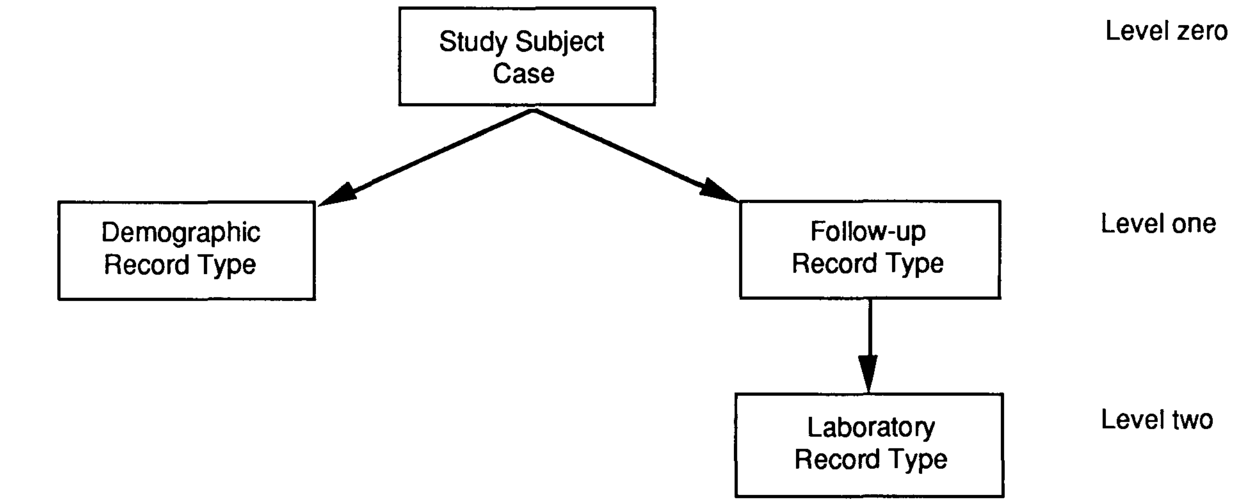
2. The Network Organization

The hierarchical structure is a one-to-one or one-to-many relationship. In our illustration, we see that one case owns many records within the case, and the follow-up records within a case own, at most, one laboratory record. But the hierarchical model is not suitable for handling a structure that requires many-to-many relationships.

Consider an extension to the clinical trial study we've used as an example. Suppose that a group of doctors are involved with this study. Each patient may see many doctors. Each doctor will certainly see many patients.

For each doctor who is involved in the study, we have available a collection of information that we also want to organize and have available in our database. For example, we have a record giving general information about the doctor—one record for each hospital with which he is affiliated and one record for each time he has seen a subject in the study. Thus, the data records for the doctor cases in our study can be organized as a two-level hierarchy.

Figure 1: Hierarchical Model



The records belonging to a particular doctor are collected into a case and organized into three record types within the case.

We now have two sets of cases that contain the data records of interest in this study—the subject cases and the doctor cases. Within each of these case types, the data records are organized in simple hierarchies. The hierarchy does not, however, enable us to represent the relationships between these two types of cases, where each follow-up subject record in a doctor case "owns" a particular subject.

The relationships we have described between the study subject cases and the doctor cases is easily handled by a *network* model superimposed onto the separate case structures. A network is best illustrated by extending the subject case diagram above, to include the doctor cases and to depict network relationships among these cases. (See **Figure 2** on the following page.)

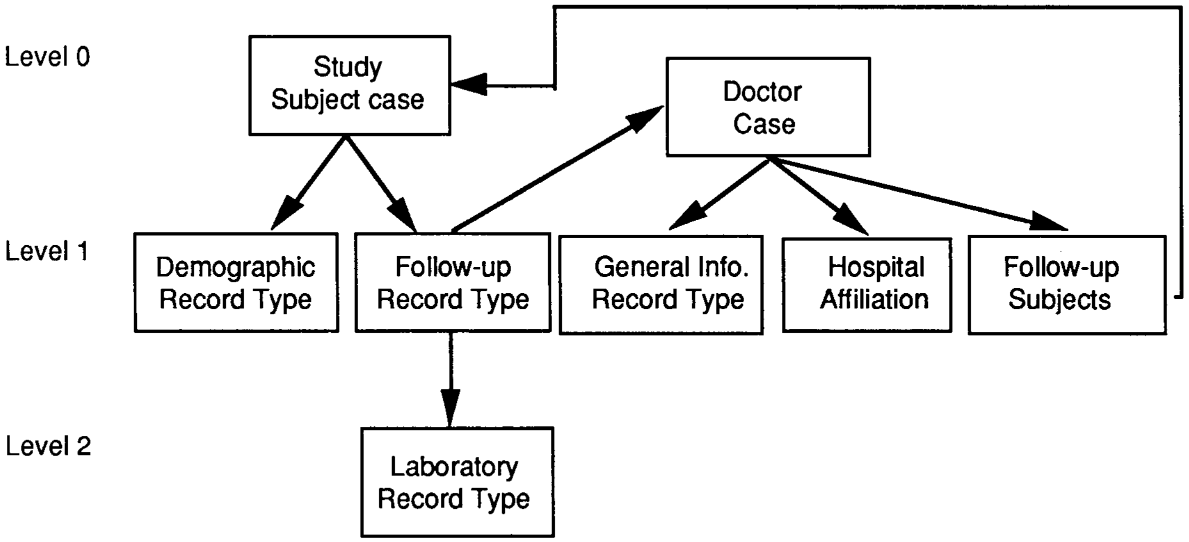
We see that the example in the figure forms a network, since the subject's follow-up records contain the doctor's name. Likewise, each doctor's follow-up record type contains each patient's ID that this doctor has seen. The subject records identify many doctors, and the doctors' records identify many subjects: a many-to-many relationship.

3. The Relational Organization

The network is a fairly simple, pre-defined relationship. Sometimes relationships are much more complicated and depend primarily on the point of view of the researcher at the moment. In this case, we need a more flexible organization.

continued on page 250

Figure 2: Network Relations Among Doctor Cases



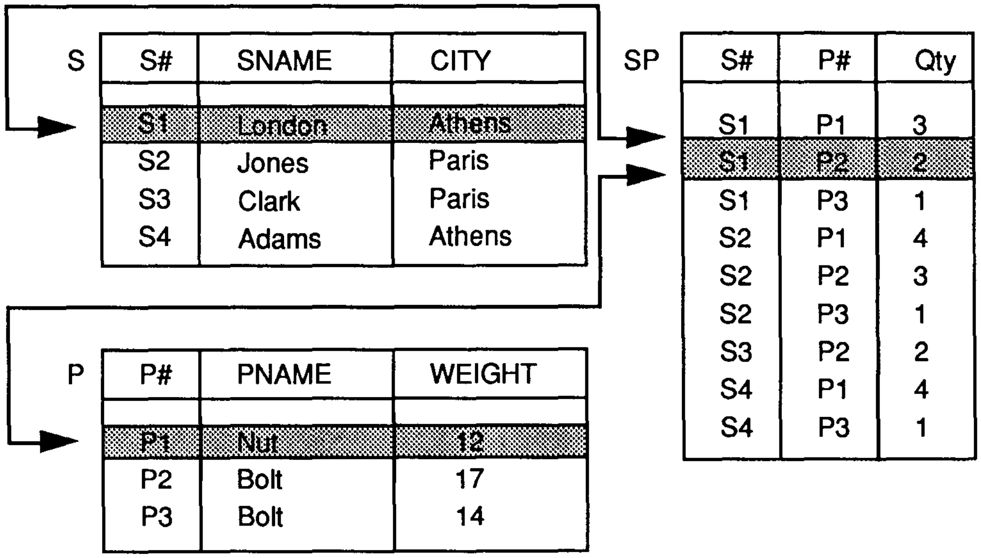
continued from page 249

A *relational* database is the most flexible of databases. It simply considers each of the record types as a separate file, each similar to a spread sheet. These record types are then tied together by common variables among the records. You can retrieve data with complex relationships, often quite simply, by use of an adhoc query language like SQL, which does not require programming ability. If you have programming skills, you can retrieve data of almost any relationship using more formal procedural languages, which are similar to higher-level programming languages like Fortran or Pascal.

The relational model manages data by defining a strict relational algebra that is applied to the ASCII data sets (called *relations*) making up a study. The data for each record type in a study is stored in individual data sets (relations), and the relational algebra enables you to sort, merge, join and project records from one record type to another record type. Each record type contains identifiers you need to link records of one record type with records of another record type as you like. Using the relational algebra properly, you can accomplish almost any type of retrieval from your database.

In **Figure 3**, the columns provide natural links between the tables. For example, the first columns of the Supplier (S) and Parts (P) tables have links into the Supplier-Parts (SP) table that enable you to learn the quantity that exists to fill an order. The first row of the SP table contains the S1 and P1 values with a quantity of 3. Through the S# link of table S we find that Supplier S1 has the supplier name (SNAME) "London" and is located in city "Athens." Through the P# link of table P, value P1 has a part name (PNAME) "nut" and has weight of 12.

Figure 3: The Relational Model



S

S#	SNAME	CITY
S1	London	Athens
S2	Jones	Paris
S3	Clark	Paris
S4	Adams	Athens

SP

S#	P#	Qty
S1	P1	3
S1	P2	2
S1	P3	1
S2	P1	4
S2	P2	3
S2	P3	1
S3	P2	2
S4	P1	4
S4	P3	1

P

P#	PNAME	WEIGHT
P1	Nut	12
P2	Bolt	17
P3	Bolt	14

When records are related through links, you can examine, print, and update data from two or more tables at one time.

A Short Course on Databases

It's difficult to cover all the aspects of databases in one article. On November 19, ACS will offer a free short course, entitled "An Introduction to INGRES DBMS," in which we'll discuss databases, and explain how an individual or a department should select, and use a database package. Please call the Computing Information Center at 625-7397 to register.

INGRES Updated to Version 6

Peter J. Oberg
PJO@VW.ACS.UMN.EDU

We've installed Version 6 of the INGRES database management system on our VAX VX system. This version is a complete revision of INGRES that takes advantage of the Client/Server architecture that has been developed in database technology.

The **Client** consists of the programs within INGRES (Query-By-Forms, Report-By-Forms, or Terminal Monitor) that allow users to build commands to access the database files that reside separately from the above programs.

The **Server** receives the commands from the user and communicates to backend programs to start the retrievals, updates, additions, or deletions and send the results to the user's terminal or to an output file.

The data management improvements in Version 6 include on-line backup, defining collating sequences for international alphabets, expansion to 300 columns (from 127) per database table, optimizer improvements for faster SQL subselects, and flexibility to mix SQL and QUEL code in the same applications. The addition of the NULL character has been added to the data types.

SQL is now the default language in INGRES. Unlike Version 5, where SQL commands were first translated into QUEL, the reverse is now true under Version 6.

Converting to Release 6

If you are now using QUEL and want to convert to SQL, it is best to convert Version 5 QUEL to Version 6 QUEL first, and then convert QUEL to SQL as time and need permits. The reasons are as follows:

1. Converting Version 5 QUEL to Version 5 SQL may require a database design change since Version 5 SQL does not allow data in one table to be updated in a second table. This is commonly done in QUEL.

2. Loading tablefields in Version 5 QUEL uses a retrieve loop, while loading table fields in Version 5 SQL requires the use of cursors. Version 6 SQL allows loading tablefields with a new select loop.

3. Pattern matching for Version 6 SQL is: = becomes **like**, * becomes %, and ? becomes -. SQL text strings now require single quotes instead of double quotes.

To access Version 6 of INGRES, type the following command interactively or add it to your LOGIN.COM file:

```
$ INGRESDEF6
```

A sample database is set up for you to create, retrieve, and update data. Type the following commands to set up the demonstration:

```
$ INGDEMO
```
{Places six files on your default account.}

```
$ CREATEDB dbname
```
{Chooses a unique database name.}

```
$ INGRES dbname
```

To create your first table, the script called **first** will need to be executed within the INGRES Monitor. When INGRES says `continue` and you see the INGRES prompt character (*), you are ready to begin.

```
continue

*\i first
```

```
    continue

    *
```

The **\i** stands for **include** and it tells INGRES to read the contents of the directory file called "first" into your workspace. To see the commands that are written in the workspace, type the command **\print** or **\p:**.

To execute the commands in this file, type the execute command: ***\go**. To leave the INGRES session, type the following command:

```
    *\quit
```

You can also access INGRES through the Menu facility, a forms-driven interface. Within the Menu utility, data manipulation, table manipulation, reporting, form design, and application development are available. To use the INGRES/Menu facility, type the command:

```
    $ RTINGRES dbname
```

Documentation

The INGRES reference manuals have been revised from Version 5. To look over the current manuals, consult our reference copies in the Computing Information Center in 1 Nicholson Hall. You may not need an entire set.

The Database staff have been contacting users, discussing the conversion regarding their applications, and working through the various steps. If you have not been contacted, you will hear from us in the coming weeks. If you are eager to get started, please contact me via my E-mail address above or at 626-0200.

There will be a short course introducing INGRES offered from November 19 to 28. To register, call the Computing

VAX Rdb Now on the VAX Cluster

Peter J. Oberg
PJO@VW.ACS.UMN.EDU

VAX Rdb/VMS is a relational database management system for the VMS systems. Rdb has the following features:

1. Represents data as a set of tables.
2. Makes databases easy to understand and easy to change.
3. Uses data values to set up relationships, providing maximum flexibility.
4. The Structured Query Language (SQL) allows operations to select, project, and join as relational operators.
5. Works with programming languages on the VX or VZ, using data manipulation statements included directly in source programs.
6. In programming, Rdb provides searching, sorting, and statistical functions.

The main access for using Rdb is to invoke the interactive SQL module. Within SQL, you develop and test your database design. While using SQL, there is on-line help information about SQL syntax.

To use interactive SQL, define the below symbol to execute the SQL. For permanent setup, define the following symbol in your LOGIN.COM file.

```
$ SQL :== $SQL$
```

To run the utility type the above symbol and press the return key:

```
$ SQL
SQL>    {This is the prompt within the SQL utility.}
```

To exit SQL, press **CTRL/Z** or type **EXIT** and press **RETURN.**

At the SQL prompt, you can type **HELP** to display a list of topics on which information is available:

```
SQL> HELP
```

There is further training available through two programs.

```
$ @RDM$DEMO:RDBDEMO
```

and

```
$ @SQL$SAMPLE:SQL$PERSONNEL
```

RDM$DEMO is a tutorial that explains SQL. **SQL$SAMPLE** is a sample database that will create a database to test the various SQL commands.

For More Information

For documentation, see:

VAX SQL Users Guide
VAX SQL Reference Manual
Guide to Database Design and Definition

These are available for reference in the Computing Information Center, 1 Nicholson Hall. If you have further questions about using database packages, you can call the HELP-Line at 626-5592.

Learning about LOOKUP:

Finding Faculty, Staff, and Students in the University Electronic Directory

Steven Brehe
SKB@UMNACVX

Jim Miner
JFM@UMNACVX

Users of the ACS UX and VX systems can use the LOOKUP utility to search an on-line University directory that now contains information on most faculty and staff, as well as most Twin Cities, Duluth, Crookston, and Morris day-school students.

With LOOKUP you can search a database of University faculty, staff, and students. The database contains nearly all the information in the printed University Student/Staff Directory, but the database will usually be more up-to-date, because a new version will be available once or twice each quarter. (In the future, we hope to include entries for University departments as well.)

You can use LOOKUP two ways: from the operating system or within LOOKUP.

If you work from the operating system, LOOKUP treats the commands you type as a query, searches the database for the requested information, and displays the results. For example, if you're working on VX, the command

```
    $ lookup robert wendall
```

displays the entry for someone named Robert Wendall, who may be a student or a faculty or staff member. (All names used in this article are *fictional.*) For more information on the kinds of queries you can make, see the Queries section in this article. (As most UNIX users will have realized, when you're working in the UNIX operating system, you would use only lowercase characters in the command above: **lookup robert wendall.**)

If you simply type the word **lookup**, you will enter the LOOKUP utility. LOOKUP will print a prompt and wait for commands. Working within LOOKUP is discussed later in this article.

Privacy

LOOKUP entries come from the data that students, faculty, and staff supply to the University about themselves. This information goes into databases from which the printed University student/staff directory is created.

Those who have requested that their directory information be suppressed will also be excluded from the LOOKUP database. Staff can also request that portions of the information be suppressed, and this request will also be observed in the LOOKUP database.

Students who want to suppress their entry or update any information in their entry, can fill out a Notice of Address Change form at Student Relations, 150 Williamson Hall, or at St. Paul Admissions and Records, 130 Coffey Hall. Staff who want to change or suppress information can fill out a Staff Demographic Information Form. Ask staff in your department for details.

Not a Mailing List Generator

LOOKUP is not intended as a way to generate mailing lists. Therefore, it will refuse any requests resulting in more than a small number of matches. (At this time the maximum is twenty.) There are no exceptions to this limitation.

To request access to mailing lists of students, contact John Kellogg, Data Reporting Services, 260 Williamson Hall (625-3387).

To request access to mailing lists of staff, contact the AIS Information Center, 625-3832.

continued on page 256

continued from page 255

LOOKUP Queries

The LOOKUP directory database contains tens of thousands of entries. Each entry is comprised of multiple *fields*, each containing information about the entry. Each field has a name that describes what the field contains; for example, the field named **UPDATE** contains the date that the information in question was last processed by ACS for inclusion in LOOKUP. (For more information on fields, see the description of the **HELP FIELDS** command on page 258.)

By default, queries are assumed to refer to the name field of the entry. Therefore, typing **lookup robert wendall** causes LOOKUP to search for entries whose name field contains **ROBERT** and **WENDALL** in any order—it could include, for example, not only Robert Wendall but also Theresa Wendall Robert and Wendall Robert Zappa. To eliminate these other names, you'd need to use the **FIRSTNAME** and **LASTNAME** fields, described later in this article under the heading **LOOKUP Commands**.

You can specify fields other than the name field to reduce the number of entries located and displayed for any single query. For example,

```
$ lookup wendall address=elm
```

would display entries containing the name **Wendall** in the name field and containing the word **elm** in the address field. **ADDRESS** is only one of several fields you can use. See the description of the **HELP FIELDS** command later in this article for a complete list.

Matching in LOOKUP is done on a word-by-word basis. That is, both the query and the entry are broken up into words, and the individual words are compared. It doesn't matter if characters are capitals or lowercase, but otherwise words must match exactly, with no characters left over; JOHN does not match JOHNSON, for example. You can override this limitation by using wildcard characters as explained elsewhere in this article.

LOOKUP will only display those entries that match all of the specifications in the query. For example,

```
$ lookup robert wendall
```

will display all entries with both **Robert** and **Wendall** in the name field, including, for example, anyone named Robert Wendall or Wendall Robert, or someone named Robert Wendall Smith or Wendall Robert Jones.

Modifying Output

LOOKUP displays certain fields by default, but it is possible to ask for different fields in a query. You can do this by using the **RETURN** keyword, and listing the fields you're interested in. For example,

```
$ lookup robert wendall return department
```

would print only the department affiliation of Robert Wendall, Wendall Robert, and the others.

You can also ask for all fields in the entry, by using the keywords **RETURN ALL.** This will show you every field that you are allowed to see in the person's entry.

Electronic Mail Addresses

Using the **RETURN** keyword, you can also learn a person's electronic mail address. The command

```
$ lookup robert wendall return email
```

would print only the electronic mail address that Robert Wendall has, if he has one.

This feature makes LOOKUP particularly useful to users of electronic mail, and makes it doubly important for E-mail users to include their E-mail addresses in the University directory database.

Using Wildcard Characters

A wildcard character in LOOKUP stands for one or more characters in a name or other field.

? matches any single character,
* matches zero or more characters, and
[] encloses two or more characters and matches a single character from this set.

For example,

```
$ lookup wend*
```

looks up all names beginning with the letters **wend**. Any number of letters may follow **wend.**

When you are using LOOKUP from the UNIX operating system (that is, when you are not inside UX LOOKUP), wild cards have to be preceded by the backslash (\):

```
lookup wend\*
```

This command looks up all names beginning with the letters **wend**. Any number of letters may follow **wend**.

On UX you can also enclose each word containing wildcards within single quotes (apostrophes) instead of using the backslash.

Within LOOKUP on UX, wildcards need no backslash or single quotes.

Working within LOOKUP

If you simply type the word **lookup**, you enter the LOOKUP utility. This is called working in *interactive mode*. In this mode, LOOKUP prompts for commands, executes them, and displays the results.

Interactive mode provides access to more database features than the mere queries you can use from the operating system, as described elsewhere in this article. Within LOOKUP, commands can be abbreviated if you provide enough characters to distinguish them from other commands. On both VX and UX, you can use capital or lowercase characters within LOOKUP.

See the following page for commands that work within LOOKUP.

Examples

Here are some examples of LOOKUP queries. We assume that the queries are being done from the VX operating system, rather than by using the interactive mode within LOOKUP.

To find the LOOKUP entry for Robert Wendall:

```
$ lookup robert wendall
```

In response to this most basic form of the LOOKUP command, LOOKUP will display entries on all persons—faculty, students, or staff—with the names ROBERT and WENDALL in their own name.

The displayed entries will be basic: Just five lines for students (name, address, phone, unit, and class) and just four lines for faculty and staff (name, department, departmental address, and office phone). If the persons have electronic mail addresses, these will be included in the basic display.

To find the LOOKUP entry for R. Wendall, if you don't know the rest of the first name, type:

```
$ lookup r* wendall
```

To find Robert Wendall, or possibly Wendell, type:

```
$ lookup robert wend?ll
```

or

```
$ lookup robert wend[ae]ll
```

To find Robert W., where you don't know the rest of the last name, you might try the following command:

```
$ lookup robert w*
```

But this last query might produce only an error message if it matches more than the allowed maximum of 20 entries in any one display. You need to narrow your selection using fields in your query.

For example, suppose you know that Professor Robert W. works in the History department. You could try:

```
$ lookup robert w* department=history
```

continued on page 258

continued from page 257

LOOKUP Commands

Once inside LOOKUP, you can use these commands:

HELP

HELP provides explanations of LOOKUP commands. When you simply type **HELP**, LOOKUP lists the available help topics. When you type **HELP** and one of these topics, LOOKUP will print help for that topic. Obtain a list of commands and a one-line description of each command by typing **HELP COMMANDS.**

HELP FIELDS

HELP FIELDS lists and describes the fields currently in use in LOOKUP. All the current fields are listed below, but more may be added in the future.

As indicated below, some fields apply only to students, others only to faculty and staff. Students who are also staff members will have all of these fields in their entries unless they have requested some to be suppressed. See **Table 1** for a summary.

Name
The person's full name, to a maximum of thirty characters.

Firstname
The person's first name. (A search by first name alone won't work; you must include **lastname.**)

Middlename
The person's middle name. (A search by middle name alone won't work; you must include **lastname.**)

Lastname
The person's last name.

Email
The person's electronic mail address. Applies to anyone who has included an e-mail address in their directory listing.

Phone
The person's telephone number. In the case of faculty and staff, this is the office number only.

Address
For students, the home address. For faculty and staff, their campus mail address (i.e., their departmental address).

Type
Person or department. (You'll be pleased to know that LOOKUP classifies all students, faculty, and staff as persons.) Departments have not yet been included in LOOKUP.

Department
The department employing the faculty or staff member. This does not apply to students.

Title
For faculty, the academic title (**ASST PROF, ASSOC PROF, PROF,** etc.) For staff, the civil service title or other rank (e.g., **RESEARCH ASSISTANT**).

Home_phone
For faculty and staff. Students' home phone numbers are displayed under **Phone.**

LOOKUP Commands, continued

Home_address	For faculty and staff. Students' home addresses are displayed under **Address**.
Unit	University unit in which the student is registered (CLA, the graduate school, etc.). This field doesn't apply to faculty and staff.
Class	Senior, Junior, Sophomore, Freshman, Doctoral, etc. This field doesn't apply to faculty and staff.
Last reg	Last period of registration.
Campus	Principal campus of employment or course of study.
Student	Indicates if the entry is for a student. This field won't appear in faculty and staff entries, unless the student is also a staff member.
Staff	Indicates if the entry is for a faculty or staff member. This field won't appear in student entries, unless the faculty or staff member is also a student.
Update	The last time the information in a particular entry was updated.

By using the **RETURN** keyword, you can modify the output—and the amount of information—you receive. **RETURN ALL** displays all the fields available in the entries you've requested.

If you omit **RETURN**, LOOKUP displays five fields for students: Name, Phone, Address, Unit, and Class. (A sixth field, Email, will appear if the student has included an Email address.) For faculty and staff, LOOKUP displays three fields: Name, Phone, and Address (or four, including Email).

If you specify one field only (for example, **RETURN CLASS, RETURN TITLE, RETURN DEPARTMENT, RETURN EMAIL**), LOOKUP displays only the information in that particular field for the specified person(s).

QUERY (or **LOOKUP**)
QUERY performs searches and works exactly like LOOKUP when used from the operating system. Within LOOKUP, there is also a **LOOKUP** command that is synonymous with the **QUERY** command.

QUIT
Exits LOOKUP. You can also type **EXIT** or **BYE**.

Suppose there were several faculty named Zorner, but you're looking for the one in the English department. In this case, you'll have to use the **DEPARTMENT** field in your LOOKUP command:

```
$ lookup zorner department=english
```

Or suppose the ZORNER you want is a graduate student living on Maple Street. You might try:

```
$ lookup zorner address=maple unit=grad
```

continued on page 260

continued from page 259

Suppose all that you want is the full name and class of a student named S. Brady. Use this command:

```
$ lookup s* brady return name class
```

Separate words in a query are allowed to match the same word in the entry; **lookup s* smith** is functionally equivalent to **lookup smith,** because the **s*** matches **smith**.

See the examples following this article to learn more about LOOKUP commands and output.

Questions? Comments?

Please send questions or comments to INFOADM@VX.ACS.UMN.EDU if you have access to electronic mail. Otherwise call our HELP-Line at 626-5592.

On-line Documentation

Much of the same information presented here is available on-line on VX as **HELP LOOKUP** and on UX as **man lookup**.

Table 1: Fields in LOOKUP

The LOOKUP database contain the fields listed in the following table, but not all fields appear in every entry. University **staff** (faculty and other employees) have some fields that students do not have, and **students** have fields that staff do not have. Any individual may choose to suppress one or more fields in his or her entry.

In this table, an X or other indication in the **Staff** or **Students** column indicates that a particular field appears in the database entry for persons in that category.

Field name	Staff	Students	Notes	Description
name	X	X	D,I	Full name (last first middle)
email	X		D,I	Electronic mail addresses
phone	work	home	D	Work or home phone number
address	work	home	D	Work or home address
department	X		D	Department(s) in which employed
title	X			Academic or civil service title
home_phone	X			Home telephone number
home_address	X			Home address
update	X	X		When data base was last updated
campus	X	X		Principal campus
lastname	X	X	I	Last name only
firstname	X	X		First name only
middlename	X	X		Middle name only
unit		X	D	Academic unit
class		X	D	Student's class (e.g., **senior**)
last reg		X		Last time student registered
student		X		Always **yes** for students
staff	X			Always **yes** for staff
type	X	X		Always **person**

Notes

D This field will be listed by *default* for every query that does not contain the **return** keyword. Within LOOKUP, see **help query** for more information.

I This field is *indexed*. Every LOOKUP query must contain at least one indexed field. Within LOOKUP, see **help restrictions** for more information.

More Examples from LOOKUP

Here are several examples from the LOOKUP on-line faculty, staff, and student directory. (All example entries are fictional.) We'll assume that you're working within LOOKUP on VX, but all LOOKUP commands shown here will work exactly the same on UX.

First you begin the LOOKUP utility.

```
$ lookup

lookup - UofM Electronic Phone Book.  CSO Version 3.10

For instructions on how to use lookup, enter 'help'.
Database ready.

lookup>
```

Now you're ready to look up a name in the on-line directory by typing in a command. In response, LOOKUP responds with an entry in its default format:

```
lookup> LOOKUP ROBERT WENDALL
--------------------
         name: WENDALL ROBERT B
        phone: 612-626-4567
      address: 321 MONTY H (MINNEAPOLIS)
   department: HISTORY
--------------------
```

This is clearly a faculty or staff entry: It has just three lines in its default form, including Wendall's University phone number and address.

Now you try to use the **RETURN** keyword and the **UNIT** field:

```
lookup> LOOKUP ROBERT WENDALL RETURN UNIT
--------------------
         unit: Not present in entry.
--------------------
```

Faculty and staff entries do not contain the **UNIT** field. Now you use the **DEPARTMENT** field with the **RETURN** keyword:

```
lookup> LOOKUP ROBERT WENDALL RETURN DEPARTMENT
--------------------
   department: HISTORY
--------------------
```

continued on page 262

continued from page 261

If you know that there's a Professor Wendall in the History department, you can use this information to locate his entry:

```
lookup> LOOKUP WENDALL DEPARTMENT=HISTORY
--------------------
          name: WENDALL ROBERT B
         phone: 612-626-4567
       address: 321 MONTY H (MINNEAPOLIS)
    department: HISTORY
--------------------
```

Here you use the **RETURN ALL** keywords to see the complete entry for Robert Wendall:

```
lookup> LOOKUP WENDALL ROBERT RETURN ALL
--------------------
          name: WENDALL ROBERT B
         phone: 612-626-4567
       address: 321 MONTY H (MINNEAPOLIS)
          type: PERSON
    department: HISTORY
         title: PROF
    home_phone: 612-234-5678
  home_address: 123 ARTHUR LANE N; BLOOMINGTON, MN 55123
        update: 1990-08-27
        campus: MINNEAPOLIS
      lastname: WENDALL
     firstname: ROBERT
    middlename: B
         staff: YES
--------------------
```

In the following example, you look up everyone with the name **WENDALL** and the initial **R**. This produces three entries:

```
lookup> LOOKUP R* WENDALL
--------------------
          name: WENDALL ROBERT B
         phone: 612-626-4567
       address: 321 MONTY H (MINNEAPOLIS)
    department: HISTORY
--------------------
          name: CONDON WENDALL R
         phone: 612-626-6789; 612-626-7891
       address: 209 TULLY H (MINNEAPOLIS)
    department: ECONOMICS
```

```
--------------------
          name: RUSSELL WENDALL C
         phone: 612-536-1234
       address: 444 44th AVE SE, MPLS, MN 55414
          unit: MED SCH-TC
         class: MED FELL
--------------------
```

The first two entries are faculty or staff; the third is a student. Note that the name and initial can occur in any order in the **LOOKUP** command.

In the next example, you want to look up every faculty member who has the letters **WENDAL** in his name. (Using the **TITLE=PROF** field works for assistant and associate professors as well as full professors.)

```
lookup> LOOKUP WENDAL* TITLE=PROF
--------------------
          name: WENDALL ROBERT B
         phone: 612-626-4567
       address: 321 MONTY H (MINNEAPOLIS)
    department: HISTORY
--------------------
          name: CONDON WENDAL R
         phone: 612-626-6789; 612-626-7891
       address: 209 TULLY H (MINNEAPOLIS)
    department: ECONOMICS
--------------------
          name: NICKELS WENDALYN A
         phone: 612-626-8901
       address: TRAVERSE H (ST PAUL)
    department: RHETORIC
--------------------
```

Assume in our final example that you know a student's electronic mail address, but not his campus mail address or phone number. With LOOKUP you can use his electronic mail address to get the other information:

```
lookup> LOOKUP EMAIL=SKB
--------------------
          name: BRAUNSCHWEIGER SHARON K
         email: BITNET: SKB@UMNACVX
         phone: 612-123-4567
       address: 1066 124TH AVE SE, MPLS, MN 55414
          unit: GRAD SCHOOL
         class: DOCTORAL
--------------------
```

And finally you leave LOOKUP:

```
lookup> EXIT
```

Software

ANSYS Version 4.4A on VX

Michael J. Frisch
MJFRISCH@UMNACVX
MJFRISCH@VX.ACS.UMN.EDU

On October 15, we updated the ANSYS Finite Element package from Version 4.4 to Version 4.4A on the VAX VX computer. Many improvements have been made and new features added. The only significant change we have found so far is that the PLT33I44 program has been replaced by a similar program called DISPLAY44. In the example given in our local documentation, the DISPLAY44 commands are all the same as the PLT33I44 commands from before, except that TYPE has been changed to /SHOW.

The local documentation for ANSYS 4.4A has been updated, although very little was changed. The starting point for local documentation is still

```
$ MOREHELP APPLICATIONS ANSYS44
```

which tells how to access further information on the package. Our vendor, Swanson Analysis, has published a manual *Revision 4.4A Changes to ANSYS Program* ($20 plus shipping) which goes together with the original two-volume *ANSYS User's Manual* (about $80 plus shipping). Copies are on reserve in the ACS Computing Information Center, 1 Nicholson Hall, 625-7397. To order your own copies, contact:

Swanson Analysis Systems, Inc.
P.O. Box 65, Johnson Road
Houston, PA 15342-0065
Phone: (412) 746-3304
FAX: (412) 746-9494

To access Version 4.4A, continue to use the command

```
$ ANSYS44
```

followed by the usual ANSYS statements such as /INTER to begin interactive mode.

The ANSYS package is one of the most well-known Finite Element Method structural analysis programs. It has a full set of features, including many graphics facilities. We have an academic version to be used for research and instruction only. It can solve small to medium-sized problems and has a wavefront limitation of 200. The ANSYS manuals attempt to explain how wavefront size relates to the ordering of nodes and elements.

ANSYS supports a number of DEC and Tektronix graphics terminals (and their emulators). We have also set up our version to generate paper plots on our PostScript and Xerox 8700 printers and on our Versatec plotter.

ANSYS

SAS 6.06 Installed on VMS

Bruce A. Center
BAC@UMNACVX

We have installed Version 6.06 of SAS on VX and VZ. This new version of SAS contains numerous improvements and differences from the current version, including the addition of **SAS/ASSIST**, which is a menu-driven front-end for SAS. These improvements and differences will be the discussion of a future article.

Access SAS 6.06 by typing:

```
    $ SAS6  {This sets up the SAS 6.06 command file.}
    $ SAS filename.SAS
```

As before, your logfile will be on *filename*.LOG and your listing file on *filename*.LIS. Subsequent SAS commands will continue to invoke SAS 6.06.

If you want to return to the current version of SAS, SAS 5.18, type

```
    $ SAS5
```

This will set up the SAS 5.18 command file. Subsequent calls to SAS will run the current SAS 5.18.

Along with SAS 6.06, there is an entirely new set of manuals. These include:

SAS Language
SAS Procedures Guide
SAS Language and Procedures
SAS/Stat Volume 1 User's Guide
SAS/Stat Volume 2 User's Guide
SAS Companion for the VMS Environment

and many many more.

They are available for perusal at the Computing Information Center in Nicholson 1.

This version is new. It may not be bug-free. Please contact me with any questions or problems.

S
A
S
VMS

Software

SPSS Version 4 on VX

Bruce A. Center
BAC@UMNACVX

We have installed SPSS Version 4 on VX and VZ. To access it, type

```
$ SPSS/OUTPUT=outfile command_file
```

Note that it is no longer called SPSSX. For the sake of confusion, SPSS Inc. has dropped the X, and we can access Version 4 simply as SPSS.

SPSS contains several new procedures including:

EXAMINE which performs Tukey's Exploratory Data Analysis, complete with histograms, stem-and-leaf plots, and boxplots.

LOGISTIC REGRESSION, which regresses a dichotomous dependent variables on a set of independent variables.

MATRIX, which is a comprehensive matrix manipulation language, the functional equivalent of SAS/IML.

SPSS 4 also contains a menu-driven front end called the SPSS Manager, which you can access by typing:

```
$ SPSS/MANAGER
```

at the VMS $ prompt. **SPSS /MANAGER** is not necessarily an improvement for experienced SPSS users. Or for inexperienced ones either.

SPSS 4 is documented in the:

SPSS Reference Guide
SPSS Base System User's Guide
SPSS Advanced Statistics User's Guide
SPSS for VAX/VMS Operations Guide

These manuals are sold as a set, and, except for the *VMS Operations Guide*, they are identical to the documentation for SPSS 4 on the Macintosh and for SPSS 4 on any other computer.

We will have much more to say about SPSS 4.0 in a future article. Please contact me with any questions or problems or comments you have about the new version.

New Brief: A Comparison of the VMS EDT and UNIX vi Editors

Michael J. Frisch
MJFRISCH@UMNACVX
MJFRISCH@VX.ACS.UMN.EDU

A few years ago, I was faced with the task of learning the UNIX **vi** editor. I had already struggled mightily to learn the VMS **EDT** editor and it took me a long time to get used to **EDT**. A major problem I had in learning **vi** was trying to understand the correspondences in action between **vi** and **EDT** commands. I finally learned **vi** by reading and re-reading the documentation and just trying out the features. But I always felt that if I had had a comparison guide, I might have learned **vi** easier.

I have prepared a document that is intended to help other **EDT** users who want to know about **vi**. I suspect this Brief might also help a **vi** user learn some things about **EDT**, but such users may be accustomed to doing things in **vi** that have no direct equivalent in **EDT**. In any case, both present users of **vi** and **EDT** may find some features in the Brief they didn't know about their respective editors.

In the Brief, I cover only the most important commands and ignore the rest. I group the command descriptions by function and I put the functions in the order a user would be likely to need to learn. Also, I gloss over some technicalities which I feel are unimportant or difficult to explain.

This Brief is now available in our Computing Information Center, 1 Nicholson Hall.

Holiday Hours for November and December

Donna Strovers

In observance of Thanksgiving, ACS systems will run in unattended mode from midnight Wednesday, November 21, until midnight Thursday, November 22.

In observance of Christmas, our systems will run in unattended mode from noon Monday, December 24, until midnight Tuesday, December 25.

In observance of the New Year, our systems will run in unattended mode from 5 pm Monday, December 31, until midnight Tuesday, January 1.

It is unlikely that any tape requests or printing will be processed during these hours. Normal operations on all systems will resume at midnight Thursday, November 22, on Tuesday, December 25, and on Tuesday, January 1.

Interactive Access Update

Paul Tranby
PDT@UX.ACS.UMN.EDU
PDT@UMNACVX

There have been a number of recent developments in the area of interactive access to the ACS central systems. For users of our terminal labs and dial-in service, there have been a number of new terminal server features added that give you greater terminal control over your interactive session. Second, the University Telecommunications Services Department has installed a dial-in terminal server to be used for access to all systems connected to the University Internet, including the ACS systems.

New Terminal Server Features

The hardwired terminals in the ACS campus labs and staff offices and the ACS dial-in rotaries are connected to ethernet terminal servers that allow users to create multiple concurrent connections to one or more systems. There have been a number of new user escape sequences added to the terminal server command interface. To aid in the explanation of these sequences, a review of the terminal server escape mechanism is in order.

The terminal server, as I just mentioned, allows the user to create multiple connections to one or more systems and to switch back and forth between these connections at will. To do this, a mechanism is needed to temporarily change your connection so that the user is communicating with the terminal server instead of the system that the user is actually working on. The mechanism used by the terminal server is called a **user escape sequence**. The user escape sequence is a two character sequence that causes the terminal server to wake up and do something special. The key to this sequence is the first character, which is called the **escape character**. This character gets the terminal server's attention and the following character tells it what to do.

The default escape character is the ASCII character <CTRL>-^ (hex 1E). Depending on terminal and keyboard, this character is usually produced by entering **<CTRL>-^** (the **^/6** key), **<CTRL><SHIFT>-^**, **<CTRL>-~** (the ~/' key) or **<CTRL><SHIFT>-~**. If you have trouble determining how to send the hex 1E character from your keyboard, check your terminal users manual or your microcomputer and software reference manuals if you are using a personal computer.

If using this character as an escape character presents a problem for your particular application, you can change it with the **terminal** command. For example, to change the escape character to **<CTRL>-]** type in the following command at the terminal server prompt (`Dialin1>` in this example):

```
Dialin1>terminal escape-character ^]
"^]" is the escape character
Dialin1>
```

The ^] in the **terminal** command was typed by actually holding the **CTRL** key down while pressing the] key.

The terminal server has a number of functions that can be invoked via an escape sequence. The character following the escape character is used to select a particular function. **Table1** lists these functions.

As the table indicates, if the second character of the escape sequence is an upper or lowercase **x**, you will again see the terminal server prompt, at which point you could create another connection, change terminal parameters, etc. Enter a question mark (?) to see a list of the terminal server commands. You can use the **where** or **show sessions** command to see your current open connections. Use the **resume** command to resume a particular connection by name or number or just a return to resume the most recent connection.

If your connection appears hung for any reason, you can send the "are you there" sequence listed above to the host

Table 1: Terminal Server Functions

If the second character is:	the function invoked is:
x or **X**	return to terminal server command mode
c, or **C**, or **<CTRL>-c**	send interrupt process
o, or **O**, or **<CTRL>-o**	send abort output
b, or **B**, or **<CTRL>-b**	send break
t, or **T**, or **<CTRL>-t**	send "are you there" request

you are connected to to see if it is still responding. If it is, you will see a `[Yes]` displayed on your screen.

It is sometimes difficult to quickly interrupt and stop continuous output from coming to your screen because of the large amount of data that gets buffered in the network between you and the central system. This problem is worse when dialing in because of the relatively slow speed of the modem connection compared to the speed of the actual network. The use of the *interrupt process* and *abort output* escape sequences described above are both effective mechanisms for reducing unwanted output. The difference between the two is that the *interrupt process* sequence sends an interrupt on to the host to interrupt the process generating the output *and* flushes the output in the network. The *abort output* sequence just flushes the output from the network.

When dialing in with a modem into the 626-1630 rotary this problem can be even more bothersome as the modems in this rotary do internal buffering. To flush the buffer in the modem you can hit the **BREAK** key. This causes the modem on the University side to throw away any data that it has in its internal buffer while sending the break on to the host to interrupt the process generating the output. Combining the **BREAK** with the abort *output sequence* reduces the output significantly. The 626-1631 rotary, however, does not use buffering modems and therefore does not have as many problems in this area. The *interrupt process* and *abort output* escape sequences by themselves work well with this rotary.

The best solution is to change the way that you display your data on the screen. For example, use text editors to display the data instead of just listing it to your screen. UNIX users can also pipe their output through **more**; see **man more** for details. VMS users can use the /PAGE option on the **TYPE** command, see **HELP TYPE** for more details. NOS users can also use **more**, see **WRITEUP,COMMAND=MORE** on CA (NOS).

continued on page 270

continued from page 269

Many terminals and microcomputer communications packages allow the user to program function keys to send a sequence of characters when pressed. Programming the user escape sequences mentioned above into these keys might be beneficial for users that use them frequently. Consult your terminal or microcomputer reference manual for specific instructions.

Some applications require a high degree of transparency in the data connection before they will function as intended. In other words, these applications require that none of the characters that get sent over the data connection get stripped out for any reason. They also usually require that you use an 8-bit no-parity connection instead of a 7-bit connection with parity. Examples of these types of applications are some file transfer protocols (XMODEM,YMODEM), some editors (EMACS), and some graphics packages.

When using an application like this, make sure that you use the 626-1631 rotary if you're dialing in (8 databits, no parity) and type the following at the terminal server prompt before entering the application:

```
terminal flowcontrol none
terminal download
```

These two commands prepare your connection to pass all data transparently. For example, the terminal server will not intercept flow control characters or the normal escape character. The **terminal download** command is a short cut for the combination of the **terminal no escape-character, terminal telnet-transparent** commands and a few others. It does not change flow control, however, therefore you must set it with the **terminal flowcontrol** command. Note that the **terminal download** command results in your escape character being set to the **BREAK** key so that your normal escape character can be sent as data. The **BREAK** key does not actually send a character

Table 2: New ACS Dial-in Numbers

phone	speed (bps)	databits	parity	stopbits
626-1200	1200	8	none	1
626-2400	2400	8	none	1
626-9600	9600	8	none	1

but instead causes the terminal to create an electronic condition on the data connection that can be detected by the equipment to which it is connected. With the escape character set to the **BREAK** condition, you will not be able to use the **BREAK** key as an interrupt mechanism to your hosts. To set the escape character back to an actual character, follow the instructions earlier in this article.

This setup can be reversed by the following commands:

```
terminal flowcontrol software
terminal no download
```

New University-wide Dial-In Service

As I mentioned at the outset, the University Telecommunications Services Department has installed a terminal server to be used for dial-in access to all systems connected to the University Internet. You can use the server to access systems that are connected to the University Internet and that support the TCP/IP protocol family (Telnet or Rlogin in particular). Therefore, besides calling the familiar ACS numbers (626-1630 and 626-1631), you can now also call any of the numbers in **Table 2** to access the ACS systems.

This new service uses the same equipment as the ACS dial-in service. You can call up and create your connections in the same manner as you do on the ACS rotaries. Note that all of these rotaries are 8 databits; no parity with one stopbit and function the same as the ACS 626-1631 rotary.

As the list above indicates, a 9600 bps dial-in rotary is also available. If you are planning to purchase a 9600-bps modem, you should contact the Telecommunications Services Department at 626-0260 to discuss which modems are the most compatible with those in the central modem pool.

For people who would like to access an outgoing modem from systems on the University Internet, there is also a new outgoing rotary that can be accessed by **Telnet**ing to mko.telecomm.umn.edu (128.101.53.99). After connecting you will be presented with the same MKO (Machine Keyboard Origination) prompt that is displayed on ITE or like devices attached to the University phone system. An authorization code will be required if attempting to dial off-campus. For assistance in using the MKO interface, obtaining an authorization code or any other question related to this new service, call Telecommunications Services at 626-0260.

Systems

VMS and Ultrix System Support: ACS Facility Management

Marisa Riviere
MARISA@UMNACVX

Win past years, ACS has negotiated discount programs with Digital Equipment Corporation (DEC) on behalf of University DEC systems. The discount programs include software distribution for the most commonly used VMS and Ultrix software products in academic environments. University systems participate in this program for a low yearly membership fee. The programs also include discount grants for hardware and documentation and umbrella contracts for hardware maintenance.

ACS complements DEC's programs with general user support, including phone consulting, training classes, online training software, orientation for the system managers, and maintenance of the University-wide DECnet networks.

ACS also establishes third-party umbrella contracts and offers contract services for modifying software, analyzing special projects, and creating general purpose software—databases, text processing, statistics, and graphics for Ultrix and VMS.

ACS's full or partial facility management services provide one more layer for the support of DEC systems. For facility management, ACS offers a wide range of options and services for managing VMS and Ultrix systems. You may select from the services listed below, combining them according to the requirements of your particular site. Service costs are negotiated for each combination of choices and system configuration. Services include:

- system and network configuration planning
- help with software and hardware planning strategies
- full operational assistance by "housing" the system at Lauderdale with uninterruptable power
- assistance with hardware installation
- software system and layered products upgrades—on site or over the University network
- file backup—on site or over University networks
- VMS system tuning
- resource utilization, performance, and capacity evaluations
- VMS system security and users data integrity reviews
- terminal server support

All these different levels of support provide options that, if selected according to the needs and experience of each researcher, instructor, or administrator, could greatly help relieve and complement the work of system maintenance tasks.

For short descriptions of the current DEC contracts and more information on ACS support services for DEC systems, contact the ACS HELP-Line at 626-0200 and consult the **DEC_Systems** topic within ListDoc on VX.

FTP and Telnet Update on VAX Cluster

Peter J. Oberg
PJO@UMNACVW

On the VAX systems at ACS, the MultiNet software has been installed to replace the Wollongong software for FTP and Telnet utilities. Anyone accessing the ACSS$UTIL:WINTCP_UTILITIES can now use the **FTP** and **TELNET** commands without any major differences in execution.

The FTP utility was modified in order to provide better compatibility with the most commonly used versions on other systems. The changes are:

1. The default option is now **/PROMPT**; that is, you will be asked for an account or user name and a password when you begin a session.
2. The **-n** parameter is recognized and equated to **/NOPROMPT**.
3. The default output file name for the **get** and **put** commands is the same as the input file name.

This change, in particular, provides compatibility with NSCA Telnet, which is commonly used by Macintosh users to access VMS. If you have been using the "old" FTP from ACSS$Util because of incompatibilities with the new software, please try the current version.

The Telnet utility was modified to use ^] the current escape character on the local host. See **Example 1** for an example of using Telnet to go from a VAX system to the CYBER CA.

The above describes the Telnet application on the VX system. The Telnet application may be different on other systems.

Please take this opportunity to test the new updated version. If problems occur, please call the HELP-Line at 626-5592.

Example 1: Using Telnet to Connect from the VAX VX to the CYBER CA

```
$ telnet
VX.ACS.UMN.EDU MultiNet
TELNET-32 2.2(75)
TELNET> CONNECT CA.ACS

Trying... Open Connected to CA.ACS.UMN.EDU, a CDC/830 running NOS.

Your Escape Character is '^]'.

WELCOME TO THE NOS SOFTWARE SYSTEM.
COPYRIGHT CONTROL DATA 1978, 1988.
90/09/25. 16.16.33. T040306
UOFM CYBER-CA        (CE) NOS 2.7.1-716/716.
FAMILY:

bye

TELNET> exit
```

Free Central System Computing Courses Fall 1990

Our courses teach you the operating systems and software on **central system** computers, large systems used by many people at the same time. These courses do not cover microcomputer software unless otherwise indicated in the following descriptions.

Academic Computing Services

To register call 625-7397, 8:00 am to noon and 1:00 to 4:30 pm, Monday through Friday.

Registration is located at ACS's Computing Information Center, 1 Nicholson Hall. Mail registrations are accepted or you can also register by electronic mail–write to classes@umnacvx or classes@vx.acs.umn.edu. Include a day-time phone number. Please call to cancel if you later decide not to attend, so we know how many to expect. Deadline for registering is 4:00 pm on the last working day before the class begins.

Math and Engineering Software
November 12-14, MW, 2:30-4:30 pm
Capabilities and use of mathematical software and program libraries on all ACS systems.

INGRES (VAX Database Package)
November 19-28, MW, 2:30-4:30 pm
Creating databases and retrieving information from the INGRES Database package on ACS' VAX/VMS.

Electronic Mail and Networks
November 13-15, TTh, 2:30-4:30 pm
Descriptions of the BITNET and Internet networks, and using them for electronic mail and transferring files. Emphasis on the VAX.

Health Sciences Computing Services

To register please call 625-5444 during regular University business hours.
Walk-in registrations are also accepted at 5-235 Moos Tower.

Introduction to NOS/VE Part 1
November 15, 2:00-4:00 pm
This course covers connecting to NOS/VE, entering commands, managing files, obtaining output, and on-line help.

Introduction to NOS/VE Part 2
November 22, 2:00-4:00 pm
This course covers batch jobs, printing options, job/command control, special files, and file transfers.

NOS/VE Full Screen Editor
November 14, 2:00-4:00 pm
This course covers creating and editing NOS/VE files, expanded on-line help for each function, and customized functions.

Basics of Minnesota MEDLINE Searching
November 28, 8:30-10:15 am
This class covers basic commands and the use of Medical Subject Headings (MeSH) in on-line medical literature searching.

ACS PHONE NUMBERS

Administrative Office: 626-1600

HELP-Line 626-5592

Access:	
ACS systems (UX,VX,VZ,CA)	
3/12/2400 bps + 7/Even/1	**626-1630**
12/2400 bps + 8/None/1	**626-1631**
LUMINA	**626-2206**
Accounts:	
ENCORE, VAX, CYBER	625-1511
Computer Hours (recorded message)	626-1819
Computing Information Center, 1 Nicholson Hall	625-7397
Contract Services	625-2303
East Bank I/O, 1 Nicholson Hall	625-5082
Engineering Services	625-1595
Equipment Maintenance/Repair	625-1595
E-mail	625-1543
FAX	626-7440
Graphics Software	626-5592
Information, Lauderdale	626-1600
Lauderdale Computer Room	626-0550
LUMINA	626-2206
LUMINA Consultant	626-2272
Math and Engineering Software	625-5830
Newsletter Subscription	625-7397
Permanent File Restoration	626-0595
Public Labs (with ACSnet)	
140 Blegen Hall	624-5278
B40 Central Library	624-3269
207/270 Diehl Hall	624-3128
4-204/4-250 EE/CSci	625-9081
121 Elliott Hall	624-0866
14 Folwell Hall	625-4896
1 Lind Hall	625-0801
308 Mechanical Engineering	625-7352
1 Nicholson Hall	625-5082
130 Physics	625-6820
9 Walter Library	626-1899
MWNC Lab Manager (14 Folwell Hall)	625-7850
Publications Information	626-1093
Short Course Registration	625-7397
Shuttle Bus Service	625-9525
System Status (recorded message)	626-1819
Tape Librarian/Operations Services	626-1838

PUBLIC LABS
TWIN CITIES CAMPUS

	Central System Printing	Interactive	Micro
East Bank			
ApH 117			X
Arch 148			X
CenH		X	
ComH		X	
DiehlH 207/ 270	L	X	X
EddyH Annex 54			X
EE/CSci 4-204/250	I, L	X	X
EltH 121	I	X	X
FolH 14, 14a	L	X	X
FronH		X	
LindH 1	I,L	X	
LindH 26			X
LindH 306B			X
MasCanCtr M39		X	
MechE 308	L	X	
MoosT 8-425			X
Nich H 1	L	X	
Phys 130	L	X	X
PioH		X	
SanH		X	
TerrH		X	
VinH 203			X
WaLib 9	L	X	X
West Bank			
AndH 170	L		X
BlegH 140	I	X	
MdbH		X	
OMWL B2			X
St. Paul			
BaH		X	
CentLib B40	I,L	X	
CentLib B50			X
ClaOff 135	L		X
McNH 305			X
Vet 436	closed fall quarter for remodeling		

I - Impact line printers.

L - Laser printers.

SYSTEM OPERATING HOURS

The ENCORE UX, VAX VX, VAX VZ, and CYBER CA systems run continuously from 6 pm Sunday until 6 am the following Sunday.

On the second and fourth Fridays of each month from 5 am to 7 am the CYBER CA and NV systems are unavailable. Low-rate hours are from 8 pm to 8 am Monday through Friday, and all operating hours on Saturday and Sunday.

ACS Newsletter Subscription Request

Send to **ACS Computing Information Center, 1 Nicholson Hall, 216 Pillsbury Drive SE, Minneapolis, MN 55455**

Add □ Cancel □ Change □

Name ____________________

Campus Mail (You must provide your department's name and departmental address.)

Department ____________________

Room __________ Building ____________________

U. S. Mail

Address ____________________

City ____________________ State _____ Zip _______

Change/Old Address

Department ____________________

Room __________ Building ____________________

Address ____________________

University of Minnesota Affiliation

Department □ Faculty □ Staff □ Student □ Alumni □ Other □

acs

Academic Computing Services

Technical Publications
100 Lauderdale CF
University of Minnesota
2520 Broadway Drive
Lauderdale, Minnesota 55113