ASK - Better Than New

Tim Snyder
Boston Scientific Corporation
480 Pleasant Street
Watertown, MA 02172

INTRODUCTION

For those Data Processing professionals who have worked with mainframes and minicomputers and with the HP3000 in particular, the opportunity to work in a department that uses the ASK software is a mixed blessing. ASK's MANMAN system is certainly the most widely installed application software system in use on the HP3000 worldwide, ever. It is becoming one of the most widely used MRP II systems on any hardware (HP3000 and VAX combined).

The reasons for the success of the MANMAN System are not only due to its implementation of the guidelines of MRP II functionality, but also because of its wise use of the particular advantages of the HP3000 itself. The success factors include:

- -It is online and the functions are integrated. IMAGE (see Note 1) allows for multiple users concurrently reading and updating multiple databases with a minimal amount of system overhead.

- -It is reliable. The HP3000 hardware is very high quality, the operating system (MPE) is mature and reliable, and the DBMS (IMAGE) is extremely reliable.

- -It gives predictable and rapid online performance. The operating system and the DBMS combine to give good response time to online users and to degrade in a uniform and predictable manner as it approaches overload.

- -It is easy to use. The online HELP functions and the consistent syntax make the packages intructional for new users and fairly intuitive for experienced users.

- -It works the same in batch and online. MPE and IMAGE handle batch and online activity through an identical interface and (prior to 6.0) the ASK software functions similarly in batch and in session mode.

- -It is extensible without FORTRAN programming. QUIZ provides a powerful and yet easy to use 4GL interface to the data upon which the ASK systems are built while posing no threat to the integrity of the transactions or the data in the databases.

-It is well-documented. The manuals provide user documentation for every command and technical information about the data files accessed/updated by each command.

There are however some weaknesses in the ASK implementation on the HP3000. This paper will address some of those problem areas and most of the topics relate to performance, reliability, or consistency. Some of these weaknesses are totally within the realm of normal technical support responsibilities of the customer; some require minor programming (usually MPE commands and utilities or QUIZ), and others would need basic design or programming changes to address properly. Since much of the success of the MANMAN software is integrally linked to its intelligent use of IMAGE, many of the weaknesses that it contains are related to improper use or inherent limits of IMAGE. For each issue requiring only conventional technical support or light programming, this paper will provide the specifics of how to accomplish those tasks as well as the quantitative performance impact. (see Note 2 for hardware/software configuration)

TECHNICAL SUPPORT ACTIVITIES

Repack Your Datasets

In many cases the database schemas supplied by ASK have the wrong path identified as the primary path in the detail datasets of the IMAGE databases. One can only identify the correct path if one knows the usage pattern of that dataset; and that pattern may vary widely between customers. However, in no case is the correct primary path the one whose path length is always one. Those are often the paths chosen by ASK as primary (SONUM in SOEFIL, REFNUM in ARFIL). The path lengths and other useful information about a database can be viewed by running either DBLOADNG (from the Contributed Software Library from INTEREX) or HOWMESSY (from Robelle). These programs will produce a table (see Figure 1) that will show not only the path length (maximum, average, and standard deviation) but also the inefficieny of the pointers. What this means is the percentage of time that a program must access a different physical block to read the next entry on a chain. Reading a different physical block can take 10 to 100 times as long as reading an entry in the same block. Thus, the performance impact of inefficient pointers on both batch and online performance can be enormous.

To determine which path should be primary, one must find out which path is used most frequently. This can be done using Turbo Profiler from HP (part number 36914A, $3500) if you have converted to Turbo-IMAGE, or by reviewing the programs that access the file. Once the primary path is determined, it may seem useless. Even if you change your primary path (via DBUNLOAD/DBLOAD, ADAGER, DBGENERAL, or DBCHANGE, or other means), the path efficiency will not change significantly over time. This is because the records are added in random order and are placed wherever there is free space.

The only solution is to occassionally repack your detail datasets along the primary path. This can be done via a chained DBUNLOAD/DBLOAD, but for most databases the time involved can be prohibitive. A selective repack for individual datasets is the preferred approach and a number of database utilities (including ADAGER Model I) can do this very rapidly. ADAGER Model I does not even require that the path on which the set is to be repacked be the primary path. You can choose any path.

Repacking does require exclusive access to the database. Be sure to DBSTORE or STORE that database first. This activity should be scheduled for some non-prime time to prevent users from attempting the run applications during the repack. Cryptic error messages advising a call to the "system manager" may result. The pointer efficiency may be reviewed after the repack using DBLOADNG or HOWMESSY again. See Figure 1a for the jobstreams to run DBLOADNG and Figure 1b to run Adager.

How often should sets be repacked? The answer is simple. Often enough but not too often. The schedule for repacking depends on how volatile the data is, how long it takes to do the repack, and how much benefit results. The database utilities DBLOADNG or HOWMESSY can be used to monitor the pointer efficiency. Each user can then decide on how often to do each repack. Some sets (INVFIL in MANDB) may need to be repacked daily while others may only need to be repacked every quarter (INVFIL, SHDFIL).

```
 :JOB DBL6,MANAGER.MANMAN,DATABASE
 PRIORITY = DS; INPRI = 8; TIME = UNLIMITED SECONDS
 JOB NUMBER = #J804
 SUN, JAN 10, 1988,  2:08 PM
 HP3000 / MPE V  G.02.04 (BASE G.02.04).
 :TELLOP BEGIN DBL6 - DATABASE ANALYSIS
 :TELLOP              ANALYZE MANDB.DATABASE.MANMAN,INVFIL
 :RUN DBLOADNG.PUB.UTILITY

 Data Base Loading Analysis -- DBLOADNG  v2.3

 Data Base name:
MANDB.DATABASE.MANMAN
 Password:
ASK
 Mode:
5
 Data set:
INVFIL
 Data set:

 Data Base name:


 END  OF  PROGRAM  :TELLOP  END  DBL6  :EOJ CPU  SEC. = 200.
 ELAPSED MIN. = 9.  SUN, JAN 10, 1988,  2:17 PM
```

Figure 1a. Jobstream to Run DBLOADNG

```
!JOB PACKINV,MANAGER/.MANMAN,DATABASE
!TELLOP BEGIN PACKINV - REPACKS INVFIL IN MANDB
!TELLOP MUST HAVE EXCLUSIVE ACCESS TO MANDB
!ADAGER
MANDB
DETPACK
INVFIL
4
1

5



!TELLOP END PACKINV
!EOJ
```

Figure 1b. Jobstream to Repack INVFIL in MANDB

For example an INVFIL with 8656 records took 13 minutes to repack using option 4 (SUPERCHAINED+) in ADAGER Model I and it reduced the inefficiency from 83% to 4%. The impact on LI,160 was to reduce the number of physical reads for a chain length of 20 from 16 to 2. In another example, SHDFIL with 144303 records took 75 minutes to repack using option 4. The inefficiency went from 66% to 4% on the key HPROD which had an average chain length of 63. Each time RE,830 (Shipping History by Product) is run the average number of physical reads dropped from 42 to 3. A month later INVFIL's inefficiency had risen to 22%.

Override Your Default Database Buffer Specifications

Whenever an IMAGE database is created, one of the parameters indicates how many data buffers should be allocated when the database is opened. The default values for this parameter are the same ones that were appropriate a decade ago when an HP3000 with 512k of memory was considered a large minicomputer. Now that memory is cheaper and therefore more plentiful, one need not be so stingy with data buffers. More buffers can improve database performance. With Turbo-IMAGE, the number can be set as large as desired and the DBMS will only allocate as many as will fit into a data segment. Using IMAGE however, the maximum is limited by the fact that the buffer space and the lock descriptor space share the same extra data segment. Allocating too much space to data buffers could prevent anyone from locking the database.

More significantly, the number of buffers should be set to a single constant number instead of being dependent on the number of users. With a variable number of buffers, IMAGE must interrupt processing everytime the number of users rises or falls by two to expand or shrink the size of the extra data segment. This is simply wasted overhead if your databases are opened hundreds of times each day.

To set the number of buffers in an existing database, sign on as the database creator in the group and account where the database resides (:HELLO MGR.MANMAN,DATABASE). While running DBUTIL (:RUN DBUTIL.PUB.SYS) enter the command

```
>>SET MANDB BUFFSPECS 24(1/120)
```

Separate Your Active Files

Many of the ASK programs and user-written QUIZ reports access more than one file at the same time. One strategy for improving performance is to make sure the files accessed by a single program are on separate disk drives. This can have a significant impact on long transactions (RE,UT commands and long QUIZ reports). With a limited number of disk drives, exactly how to arrange the datasets can become a complex juggling act. ASK does provide some suggestions for 4 and 5 disk drive systems. If you would like to do your own research, the command >LISTF filename;MAP in LISTDIR5.PUB.SYS will show you where each extent of each file is located. Another way to check is to look at the SYSLIST from your full system backups but this only shows the drive of the first extent. For databases and especially if you have recently done a reload, all of the extents will usually be on the same drive. For IMAGE databases the datasets are numbered in the same order as they appear in UT,979 or on the FORM SETS command in QUERY. The two guidelines are: place the masters for each detail dataset on different drives from the detail and from the other masters for that detail and place datasets the are frequently used together on different drives.

Once you have determined which datasets to move to which disk drives, you can choose any number of techniques to accomplish your goal. You can run DBUTIL.PUB.SYS and use the command:

```
>>MOVE MANDB01 TO 3
```

ADAGER Model I will let you move datasets or you can choose device-specific dataset assignment as part of capacity changing or repacking. Be sure to always specify the appropriate disc on subsequent ADAGER operations, otherwise ADAGER might move the dataset to another disk.

When you have moved your datasets, you will also need to check your free space on each disk drive to see if you have overloaded one with too many large datasets.

Restrict Long Transactions To Non-Prime Time

Many of the ASK commands (RE,UT especially) can run for hours on a large database even when there are no other users. For this reason, it is best to limit the users who can run these commands during prime shift by removing them from the list authorized in the password database for those passwords. The same holds true for using QUIZ during prime shift to serially read datasets. Unless a QUIZ report is doing keyed access via the CHOOSE command, the overhead of serially reading moderate to large datasets online during the day using QUIZ can make a noticeable difference in the response time of the machine. Although it can help to run these commands and QUIZ in a lower dispatch queue (either in batch or by forcing them into the DS queue), most of that kind of activity can be scheduled to run on non-prime shift. The impact of executing these commands during prime shift will be felt first and most severely by those doing large data entry transactions (eg Kit a Work Order, Enter a Sales Order, Invoice a Sales Order). If one person is executing a long transaction (over 10 minutes), the response time for the long data entry transactions will double (5 seconds to 10 seconds). If a second person is also doing a long transaction at the same time the response time for data entry will now be triple (15 seconds). It wouldn't take many online reports (QUIZ or RE) or utilities to severly degrade the productivity of the data entry transactions.

Erase Before Using

A number of ASK programs erase entire datasets, and deleting tens of thousands of records from an IMAGE dataset can be very time consuming. One method to improve the performance of those programs is to use a utility to erase the set first. Adager Model I has the SETERASE function and there are other utilities as well that can also erase a dataset much faster than the ASK programs can. The SETERASE function can be completed in less than a minute while it may take hours to delete all the records from MRPPLAN or MPSPLAN while running RE,900 or RE,930.

LIGHT PROGRAMMING

QUIZ Can Be Faster Than FORTRAN

Despite the fact that QUIZ is a Fourth Generation Language (4GL) and a relative newcomer to the world of computer languages compared to FORTRAN, there are many ASK reports that can be written in QUIZ to run an order of magnitude faster than their FORTRAN counterparts. QUIZ is often used in MIS departments that have the ASK software to produce variations on the standard reports. Users often require different selection criteria, different sort sequences, more or less subtotals, more or fewer columns, etc. At the same time QUIZ can also allow the MIS department to get more reporting done in the same amount of time. In this case both the users and the MIS deparment win; better reports in less time.

The two techniques used to accomplish this double-win are the use of QUIZP, the privileged-mode version of QUIZ, combined with the approach of only processing those records that are absolutely necessary.

Every ASK user that has QUIZ also has QUIZP. I have seen QUIZP used as the default version of QUIZ in three ASK user sites every day for years with no problems whatsoever. QUIZP accepts exactly the same syntax as QUIZ and produces exactly the same results only faster. It is faster than QUIZ by an order of magnitude in the simplest situation of serially reading a single file with no LINKS. Whenever LINKS are used, the relative advantage of QUIZP is not as pronounced. This limitation is not as severe as it sounds as shown in the following example.

In OMAR the Backlog Reports (RE,710) can take hours to run if there are tens of thousands of orders in SOEFIL. This can be true even if the backlog is only one order. The reason it takes so long is that the program reads SOEFIL first and for each order examines each record in SODFIL to see if it has been fully shipped. Using QUIZP (see Figure 2), QUF7103 first processes all the records in SODFIL to find the ones on backorder. The first step creates a subfile which is the basis for the subsequent steps which do the linking to other files and which finally produce the report. Other variations of RE,710 could be generated from the same original subfile without adding more than a few minutes to the elapsed time. Figure 3 shows the actual run times when the number of records in SOEFIL were 32,338 and in SODFIL were 77,694. (see Note 2 for hardware configuration)

```
ACCESS SODFIL
DEFINE BO NUMERIC*7 = SODQO - ( SODQS + SODSNI )
SELECT IF BO NE 0
REPORT SUMMARY SONUM SODREQ SODSCH BO SODPRI PRONUM
SET SUBFILE NAME Q7103 TEMP
SET REPORT LIMIT 8000
GO

ACCESS *Q7103 LINK PRONUM TO PRONUM OF PROMAS OPTIONAL
REPORT SUMMARY SONUM SODREQ SODSCH BO SODPRI PRONUM PRODES
SET SUBFILE NAME Q7103A
GO

ACCESS *Q7103A LINK PRONUM TO ITNO OF IM OPTIONAL
REPORT SUMMARY SONUM SODREQ SODSCH BO SODPRI PRONUM PRODES &
               QOH
SET SUBFILE NAME Q7103B
GO

ACCESS *Q7103B LINK SONUM TO SONUM OF SOEFIL OPTIONAL
SORT ON PRONUM ON SODREQ
DEFINE DSCHDATE DATE MMDDYY=DATE  (SODSCH  + 26235 )
DEFINE DSODTOT  NUMERIC*9  =BO*SODPRI
DEFINE DSODPRI NUMERIC*9   =SODPRI
DEFINE HOLD STRING*1= "*" IF SOEHLD NE 0 ELSE &
  " "

REPORT  TAB 1 HOLD TAB 2 PRONUM TAB 17 SHPNO TAB 28 PRODES &
  TAB 51 SOES1 TAB 81 DSCHDATE TAB 90 SONUM TAB 99 BO &
  TAB 108 DSODPRI TAB 120 DSODTOT
FOOTING  AT PRONUM  SKIP 2  &
    TAB  43 " SUBTOTAL   " PRONUM  &
    TAB 67 "QTY-ON-HAND" QOH  &
    TAB  99 BO  SUBTOTAL      &
    TAB  120 DSODTOT  SUBTOTAL  SKIP 2

FINAL   FOOTING    &
     TAB   93  "FINAL TOTAL  "   &
     TAB   120  DSODTOT   SUBTOTAL

PAGE HEADING TAB 45 "COMPANY NAME" SKIP 1 TAB 2 SYSDATE &
 TAB 15 SYSTIME TAB 40 "DAILY INVENTORY TO BACKLOG" &
 TAB 105 "PAGE" SYSPAGE SKIP 1 TAB 1 "REPORT QUF7103" &
 TAB 42 " SORTED BY PRODUCT NUMBER" SKIP 1
SET REPORT DEVICE PRINTER
SET REPORT NAME QUF7103
SET REPORT LIMIT 8000
GO
```

FIGURE 2. QUIZ Report to imitate RE,710,5 only faster

```
:JOB TEST7103,SNYDER.MANMTI,SNYDER
PRIORITY = DS; INPRI = 8; TIME = UNLIMITED SECONDS
JOB NUMBER = #J802
SUN, JAN 10, 1988, 12:50 PM
HP3000 / MPE V  G.02.04 (BASE G.02.04).
:QUIZP 01,01
Q U I Z   (5.01.E1)F     LEVEL 50   06014
Copyright 1985 COGNOS INCORPORATED
> USE QUF7103.QUIZMTI
> GO
*W* Attempting to open primary file in privileged mode.
*W* Primary file opened with privileged mode.

Records selected:  5226
Records written:   5226

Records selected:  5226
Records written:   5226

Records selected:  5226
Records written:   5226

Records selected:  5226
Records sorted:    5226
Lines printed:     6639
Pages printed:     139

> SHOW ACTIVITY

Elapsed time (minutes):  11
CPU time (seconds):      342
Files opened:            7
Records read:            108803
Records selected:        20904
Records sorted:          5226
Records written:         15678
Records reported:        20904
Lines printed:           6639
Pages printed:           139
Reports produced:        4

> EXIT

END OF PROGRAM
:EOJ
CPU SEC. = 356.ELAPSED MIN. = 12.SUN, JAN 10, 1988,1:01 PM
```

FIGURE 3.   STDLIST from Running QUIZ Report

Use the CASE Contruct In QUIZ

For QUIZ (5.01) reports, one suggestion is to use the CASE construct instead of IF...ELSE in DEFINEd temporary variables. Since the options of the CASE statement (equal to one value or between two values) are much simpler than the Boolean nightmares that can be constructed with IF...ELSE, the results can be evaluated using a simpler and faster algorithm. The following is a CASE (!) study of exactly how much faster.

The report (QUF102) in Figure 4. refers to a usefile (PTLEVEL4) in Figure 5 that defines a four-level hierarchy for grouping products by product type in Ask's Order Processing System - OMAR. The original usefile contained eight DEFINE statements which contained a total of 117 IF clauses. The new usefile has the same DEFINE statements with 117 WHEN clauses.

The report sorts by product number within product type within each of the four levels of the hierarchy to create a listing of all products with HEADINGs, FOOTINGs and COUNTs at each of the four levels. The file contains 5856 records of which 3154 are selected, sorted and reported. The IF...ELSE syntax required 129 CPU seconds over 35 minutes whereas the CASE syntax required 69 CPU seconds over 10 minutes. (See Note 2 for Configuration)

The results of this test were sufficiently significant (the faster syntax required only 53% of the CPU time) that the guidelines for QUIZ reporting now specify the use of the CASE syntax whenever possible in this MIS department.

Since this particular usefile is referenced by forty reports, the benefits of this improvement will be immediately incorporated into a large portion of the regular reports for this company. Perhaps this will delay the need for that hardware upgrade by a few weeks.

```
ACCESS PROMAS LINK PROTYP TO PTCODE OF PTMAS
DEFINE DPROTYP NUMERIC*10 = PTCODE
USE PTLEVEL4.QUIZMTI.MANMTI
SORT ON LEVEL1 ON LEVEL2 ON LEVEL3 ON LEVEL4 ON PROTYP &
     ON PRONUM
PAGE HEADING TAB 1 "DATE: " SYSDATE &
             TAB 20 "TIME: " SYSTIME &
             TAB 120 "PAGE: " SYSPAGE &
             SKIP &
             TAB 1 "PROGRAM: QUF102" &
             TAB 50 "PRODUCT LISTING BY PRODUCT TYPE" &
             SKIP &
        TAB 50 "========================================" &
             SKIP 2 &
             TAB 1 "DESCRIPTION" &
             TAB 35 "PART NUMBER" &
             TAB 60 "PROD TYPE" &
             TAB 72 "UOM" &
             SKIP 2
REPORT       TAB 1 PRODES &
             TAB 35 PRONUM &
             TAB 60 PROTYP &
             TAB 72 PROUOM &
             SKIP
HEADING AT LEVEL1 TAB 1 HEAD1
HEADING AT LEVEL2 TAB 3 HEAD2
HEADING AT LEVEL3 TAB 5 HEAD3
HEADING AT LEVEL4 TAB 7 HEAD4
HEADING AT PROTYP TAB 9 PTDESC
FOOTING AT PROTYP &
    TAB 9 PTDESC "COUNT=" &
    TAB 40 COUNT SKIP 2
FOOTING AT LEVEL4 &
    TAB 7 HEAD4 "COUNT=" &
    TAB 40 COUNT SKIP 3
FOOTING AT LEVEL3 &
    TAB 5 HEAD3 "COUNT=" &
    TAB 40 COUNT SKIP 4
FOOTING AT LEVEL2 &
    TAB 3 HEAD2 "COUNT=" &
    TAB 40 COUNT SKIP 5
FOOTING AT LEVEL1 &
    TAB 1 HEAD1 "COUNT=" &
    TAB 40 COUNT SKIP 6
FINAL FOOTING SKIP 5 TAB 1 "TOTAL COUNT=" TAB 20 COUNT
SET REPORT LIMIT 10000 DEVICE PRINTER PRIORITY 5 COPIES 1
SET REPORT NAME QUF102
GO
```

FIGURE 4.   QUIZ Report that Refers to Usefile

```
DEFINE LEVEL1 STRING*1=  CASE OF DPROTYP &
       WHEN 1000 TO 24999 THEN "1" &
       WHEN 25000 TO 26999 THEN "2" &
       WHEN 27000 TO 29998 THEN "3" &
           DEFAULT "4"
DEFINE HEAD1 STRING*20=  CASE OF LEVEL1 &
       WHEN "1" THEN "TOTAL TRADE" &
       WHEN "2" THEN "TOTAL OEM" &
       WHEN "3" THEN "TOTAL INTERCOMPANY" &
           DEFAULT "MISC"
DEFINE LEVEL2 STRING*1=  CASE OF DPROTYP &
       WHEN 1 TO 11999 THEN "1" &
       WHEN 12000 TO 21999 THEN "2" &
       WHEN 22000 TO 24999 THEN "3" &
           DEFAULT "7"
DEFINE HEAD2 STRING*15=  CASE OF LEVEL2 &
       WHEN "1" THEN "THERAPEUTIC" &
       WHEN "2" THEN "DIAGNOSTIC" &
       WHEN "3" THEN "SURGERY" &
           DEFAULT "MISC"
DEFINE LEVEL3 STRING*2=  CASE OF DPROTYP &
       WHEN 1000 TO 2999 THEN " 1" &
       WHEN 3000 TO 4999 THEN " 2" &
       WHEN 5000 TO 11999 THEN " 3" &
       .  .  .  .
           DEFAULT "12"
DEFINE HEAD3 STRING*20=  CASE OF LEVEL3 &
       WHEN " 1" THEN "VASCULAR" &
       WHEN " 2" THEN "DRAINAGE" &
       WHEN " 3" THEN "URORADIOLOGY" &
       .  .  .  .

           DEFAULT "MISC"
DEFINE LEVEL4 STRING*2=  CASE OF DPROTYP &
       WHEN 1000 TO 1099 THEN " 1" &
       WHEN 1100 TO 1199 THEN " 2" &
       WHEN 1200 TO 1299 THEN " 3" &
       WHEN 1300 TO 1399 THEN " 4" &
       .  .  .  .
           DEFAULT "40"
DEFINE HEAD4 STRING*20=  CASE OF LEVEL4 &
       WHEN " 1" THEN "STEERABLES" &
       WHEN " 2" THEN "OCCLUSION BALLOONS" &
       WHEN " 3" THEN "DILATATION CATH/PEMT" &
       WHEN " 4" THEN "BLUE MAX" &
       .  .   .  .
           DEFAULT "MISC"
```

FIGURE 5.   USEFILE with CASE statements (abbreviated)

Use QUIZ To Switch Your Transaction Logs Before They Fill Up

One of the most unpleasant surprises is to have your MANMAN/MFG system shut itself down during the day without warning. One easy way to do this is to fill up the Transaction Log file. When this occurs, all users must log off (or be logged off) from the system; and the user who shut it down must start it up again. If you are lucky, no transactions have been lost and the only long-term effect is a few dozen (or hundred) mildly aggravated users.

It would be nice if ASK provided a utility which would decide based on some COMIN variable setting whether or not to switch logfiles when doing the nightly UT,990/UT,999. One certainly does not want to create a new logfile every day. If the second shift operator is dependable, s/he can be assigned the task of daily monitoring the need to switch logfiles.

There is a way to automate the task without a single line of FORTRAN code. QUIZ and a series of jobstreams can be used to check the EOF on the TR file; and if it exceeds a certain value, it can choose between streaming a job to switch or one not to switch the logfiles. These jobs can be part of your nightly schedule of unattended activities but should be done before any other jobs or sessions are permitted after the backups.

Figure 6a. shows the dictionary that defines the format of the LISTF,2 for QUIZ. The jobstream in Figure 6b. will determine if the most recent Transaction Log File has exceeded 20,000 records (capacity set to 30,000). If it has, then the subsequent job STOPMANW will switch to a new Log File; whereas STOPMAND will do the shut down/start up but will not switch logfiles.

```
SCHEMA "     LISTF FORMAT                  "

FILE LISTFILE          TYPE MPE                 &
  OPEN LISTFILE

ELEMENT FNAME             X(008) HEADING "FILE NAME  "
ELEMENT EOF     9(008)          HEADING "END-OF-FILE"
ELEMENT FILLER23          X(023)
ELEMENT FILLER41          X(041)

RECORD LISTFILE
  ITEM FNAME                       CHARACTER
  ITEM FILLER23                    CHARACTER
  ITEM EOF                         DECIMAL
  ITEM FILLER41                    CHARACTER

BUILD SCHEMA
```

FIGURE 6a.   QUIZ Dictionary Definition of LISTF,2 file

```
!JOB STOPMAN,BATCHM/BATCHM.MANMAN;inpri=13
!TELLOP BEGIN STOPMAN-GENERATE JOB TO SHUT/RESTART MANMAN
!PURGE LISTFILE
!LISTF TR@.DATABASE.MANMAN,2;LISTFILE
!SAVE LISTFILE
!EDITOR
T LISTFILE
CQ 1/1,"",ALL
DQ 1/5
SET FIXED
SET RIGHT=80
SET LENGTH=80
K
EXIT
!FILE QUIZLIST=$NULL
!FILE QSCHEMAC=LISTFQ
!QUIZ
ACCESS LISTFILE
SORT ON FNAME D
SET SUBFILE NAME LTEMP
REPORT SUMMARY ALL
GO
ACCESS *LTEMP
SET REPORT LIMIT 1
SET REPORT DEVICE DISC NAME STOPMANT
SET NOHEAD NOWAIT NOBLANKS
DEFINE MAXEOF STRING*17 = "STOPMANW.JOB" IF EOF > 20000 &
                     ELSE "STOPMAND.JOB"
INITIAL HEADING &
  TAB 1 "!JOB STOPMANT,BATCHM/BATCHM.MANMAN;INPRI=13" SKIP &
  TAB 1 "!TELLOP BEGIN STOPMANT"
REPORT TAB 1 "!STREAM" TAB 13 MAXEOF
FINAL FOOTING  TAB 1 "!RUN PAUSE" SKIP TAB 1 "30" &
          SKIP TAB 1 "!TELLOP END STOPMANT" &
          SKIP TAB 1 "!SET STDLIST=DELETE" &
          SKIP TAB 1 "!EOJ"
GO
EXIT
!STREAM STOPMANT
!TELLOP END STOPMAN
!SET STDLIST=DELETE
!EOJ
```

Figure 6b.   Jobstream to Switch Logfile when Appropriate

Why Isn't The COMIN File In Your Dictionary?

QUIZ can be used in most of the ASK systems to access all the data that the system uses. There are a few files in some of the systems that are not included in the Dictionary supplied by ASK and therefore, cannot be accessed with QUIZ. The dictionary can be modified to add new files and new elements. One important source of data is the COMIN file in MANMAN/MFG and in OMAR. The number of days/months usage and the fiscal period are two pieces of information which are necessary for producing useful QUIZ reports and which reside in the COMIN files.

Prior to version 5.01, the method that QUIZ used to open MPE files (ie COMIN) made it difficult to allow QUIZ to access these files while the ASK system was updating them. QUIZ would lock out the ASK programs which was a disaster for the users. The newer version of QUIZ, however, does not lock out ASK users nor is it locked out by ASK users.

Figure 7a. shows the entries that need to be added to the QUIZ Dictionary. After using the editor to modify the source (QSCHEMA.QUIZ.MANMAN), the Cognos utility QDD.CURRENT.COGNOS must be run to recompile the Dictionary. The recompilation may take a while and will make the compiled dictionary (QSCHEMAC.QUIZ.MANMAN) inaccessible for some time. Plan this task for a time when no one is using QUIZ. Figure 7b. shows a simple QUIZ report to access the number of months usage from the COMIN file and to calculate the average monthly usage.

```
FILE MCOMIN TYPE MPE OPEN COMIN/LOCK.DATABASE

ELEMENT COMINEL  9(005)
ELEMENT COMINELX X(002)

RECORD MCOMIN
  ITEM COMINEL     INTEGER SIGNED SIZE 2
  REDEFINED BY
    ITEM COMINELX  CHARACTER SIZE 2
  END
```

FIGURE 7a. QUIZ Source Dictionary Code for COMIN File

```
ACCESS IM LINK TO RECORD 9 OF MCOMIN
CHOOSE ITNO PARM
DEFINE AVGUSG NUMERIC*8 = PTDUNM / COMINEL
REPORT ITNO AVGUSG
GO
```

FIGURE 7b. QUIZ Report Using the COMIN file

What's New Pussycat?

One type of question to which users frequently require an answer is "Who are the new customers?" or "What parts/products are new this year?" The ASK systems track most of the activity files by date (sales orders, purchase order, work orders) but none of the master files indicate when the record was added. Perhaps someday ASK will add the logic to its AD,nnn programs to record the two-byte ASK formatted date in IM,PROMAS,CUSFIL,BILMAS,etc.; but one need not wait. Again QUIZ can provide a work-around.

On a specified schedule (weekly, monthly, yearly) a simple QUIZ report can be run to copy the unique keys from the file (eg PRONUM from PROMAS) into a permanent subfile (see Figure 8a.). The subfile can them be renamed with an appropriate suffix (eg :RENAME PROMAS,PR880101 for January 1, 1988). The subfile can then be used to generate a report at a later date of all those products on PROMAS which were not in PROMAS on 1/1/88 (see Figure 8b.). This approach assumes that products are never deleted, but a slight variation could address that situation as well.

```
ACCESS PROMAS
REPORT SUMMARY PRONUM
SET SUBFILE NAME PROMAS KEEP
SET REPORT LIMIT 9000
GO
```

FIGURE 8a. QUIZ Report to Create Snapshot Subfile of PROMAS

```
ACCESS *PROMAS
REPORT SUMMARY ALL
SET SUBFILE NAME PROTEMP
SET REPORT LIMIT 20000
GO
ACCESS PROMAS
DEFINE PROCOUNT NUMERIC*2 = 1
DEFINE COUNTDATE DATE     = SYSDATE
SET REPORT LIMIT 20000
REPORT SUMMARY PRONUM PROCOUNT COUNTDATE
SET SUBFILE NAME PROTEMP APPEND
GO
ACCESS *PROTEMP
SORT ON PRONUM
REPORT SUMMARY PRONUM PROCOUNT SUBTOTAL COUNTDATE
SET SUBFILE AT PRONUM NAME PROTEMP2
GO
ACCESS *PROTEMP2 LINK TO PROMAS
DEFINE SINCE YYMMDD = PARM
SELECT IF PROCOUNT = 1
PAGE HEADING TAB 1 "DATE: " SYSDATE &
             TAB 20 "TIME: " SYSTIME &
             TAB 120 "PAGE: " SYSPAGE &
             SKIP &
             TAB 1 "PROGRAM: QUN002 " &
             TAB 55 "NEW PRODUCTS ON OMAR              " &
             SKIP &
             TAB 55 "SINCE                 " &
             TAB 78 SINCE &
             SKIP 2 &
             TAB 1 "PRODUCT" SKIP &
             TAB 1"NUMBER" &
             TAB 18 "DESCRIPTION" SKIP 2
REPORT TAB 1 PRONUM &
       TAB 18 PRODES
SET REPORT DEVICE PRINTER NAME QUN002 PRIORITY 5
GO
```

FIGURE 8b. QUIZ Report Showing New Products in PROMAS

FLAWS WE HAVE LEARNED TO LIVE WITH

Why Not IMAGE?

Many of the newer ASK systems have all the files including the COMIN file defined as part of the database. Why haven't the older systems been brought up to date to include all the data in the IMAGE database? This dichotomy between IMAGE and non-IMAGE files complicates backup, recovery, support, programming, QUIZ access.

As a corollary problem, dealing with data in the non-IMAGE COMIN files can be very difficult especially if the format is a double word integer split into two COMIN variables. These parameter files should be easier to work with.

Why Not Transaction Logging?

Since much of the data for the ASK systems (MFG,OMAR,AP) exists outside the IMAGE database, the possibility of using IMAGE logging (Roll-Forward, ILR, or Roll-Back) is very remote. The options for ILR and Roll-Back recovery are fairly new and there is a performance price to be paid for any of these techniques; but ASK could allow users to make their own choices about the trade-offs between reliability and performance. Even with many of the files outside IMAGE, the programs could at least include DBBEGIN/DBEND blocks to bracket logical transactions. If a user is not using logging, the DBBEGIN/DBEND are ignored by IMAGE and take a miniscule amount of CPU time compared to the whole transaction.

Why Integer Keys?

There are a number of keys for IMAGE datasets in the ASK system which represent ticking time bombs. Most of the keys in the databases are character data elements but several are integers (PTCODE in OMAR, CHKNO in PAYDB). These keys can cause severe performance problems if the values of the keys begin to generate duplicate values for a remainder when divided by the capacity of the dataset. These keys could be converted to right-justified zero-filled character strings to preserve the sort sequence while avoiding the hashing problem that integer keys can often cause.

Data Element Numbers, Names, Item Lists, and *

The ASK databases violate one of the most basic guidelines for designing normalized databases. The same data element is given different names in each dataset. This practice makes the structure of the database seem far more complex than it is and makes programming and QUIZ reporting far more difficult. Even more troublesome is the practice of using the same element to mean two different things. In ARFIL the element SONUM can mean the original Sales Order Number or it might mean the referenced invoice number for a credit memo. If it is a duck, call it a duck; and if it is not a duck, don't call it a duck.

When referencing an item list in a database call (DBGET,etc), the ASK programs use item numbers. This practice may seem efficient but it is very rigid and is not much more efficient than using the special lists (@ for all elements, or * for the same list as previously used). Item number lists could also be used without hard coding the item numbers into the programs. The IMAGE Intrinsic DBINFO will convert item names to numbers and remove the need to predefine all elements in a database (DUM001,etc.).

Command Logging Could Be More Useful

In version 5.0 ASK provided the option of logging the commands being used to a disk file. This was a wonderful idea but it was incompletely implemented. The records did not properly record the CPU time consumed by sub-processes (RE,UT and some TR commands) and the file was not defined in the POWERHOUSE dictionary. Since the information is incomplete and since access is limited to a single report, this potentially valuable tool is almost useless.

Database Locking: Sometimes too Much, Sometimes too Little

All the systems do all database locking unconditionally. Why couldn't the choice be left to the users via a COMIN variable as to what kind of locking will be done? Some of the utilities lock after the open and unlock before the close of the database when the lock could be issued around each update or could be restricted to a single file instead of the whole database. In some cases less global locking would be more compatible with other users of the database.

Many reports read a dataset, create a work file with record numbers from the dataset, sort the workfile, reread the workfile and try to reaccess the records in the dataset by record number. This occassionally results in IMAGE errors when the record has been deleted while the report was in progress. If this two-step reporting is going to be used, why doesn't the program lock the set until it is complete?

CONCLUSION

The ASK Software is superior in functionality and performance on the HP3000, but there is always room for improvement. Each ASK user can add to the success of his/her environment by optimizing the performance of his/her own activities. ASK will continue to improve their software and may include some of these ideas in the future.

NOTE 1. Throughout this paper IMAGE refers to either IMAGE or Turbo-IMAGE interchangeably unless otherwise noted.

NOTE 2. The hardware configuration for most of the statistics was an HP3000/68 with 5 Megabytes of memory, 5 7933XP disc drives, two IMBs and three high speed GICs with Memory Caching turned off running UB delta 4 MIT (G.02.04) of MPE and Version 5 of ASK Software.