Transaction Processor For The HP3000

Godfrey lee
Project Manager
Quasar Systems Ltd.

I am sure that each of us has had the need to manipulate files, or perform bulk updates of an application database, and found that the existing methods are either incomplete (i.e. FCOPY) or too troublesome (i.e. COBOL) to use. Most application systems involve several standard batch functions which require custom programming. Yet the task involved is so standard one should be able to specify it in a simple, logical and straightforward manner.

These functions can include:

- Daily, weekly, or monthly rollovers,
- Reformatting a file,
- Producing a summary file,
- Selectively copying based on some condition,
- Copying elements from one file to another,
- Reformatting a database.

File manipulation tasks are a common requirement in developing as well as running most application systems. While excellent productivity tools now exist for large segments of application development and maintenance, batch processing programs still have to be prepared in the same tiresome manner.

The paper introduces the concept of a powerful batch-oriented data manipulation tool called a TRANSACTION PROCESSOR, which will keep pace with and interface with current state-of-the-art productivity tools.

The TRANSACTION PROCESSOR will be called QTP and will complete Quasar Systems' family of application generator products, which currently include QUIZ for reports and QUICK for screen-based input. With this family, users will be able to generate entire applications in a consistent easy-to-use style.

This paper will discuss:

1. QTP in relation to an application dictionary
2. Design objectives
3. QTP in operation (some examples)
4. QTP in the production environment
5. Design considerations.

TRANSACTION PROCESSOR AND THE APPLICATION DICTIONARY

The transaction processor will be able to operate as an independent product in association with Quasar Systems' application dictionary. In essence, QTP will have two components:

1. QSCHEMA

The schema processor compiles a description of data files and element characteristics including data validation and display specifications. The compiled schema functions as an application dictionary, providing central administrative control and freeing users of QTP from a great deal of repetitive programming.

2. QTP

Under the control of specification statements which can be used by both programmers and non-programmers, QTP will carry out two major functions:

- standard batch applications
- file manipulation.

DESIGN OBJECTIVES

The design objectives of QTP are:

- to support the standard maintenance functions of add, change and delete against all data permanently on file
- to support standard editing of input including, type checking, value range checking, and pattern matching
- to support the copying of elements from one file to another
- to allow the reformatting of files and databases
- to be able to produce summary files
- to support these summary options: sum, count, average, maximum, minimum, percentage and ratio
- to be able to specify the sorting and selection of input files
- to support any combination of IMAGE, KSAM and MPE files
- to reference the structure, composition and elements of files in a central independent schema
- to use concise specification statements in simple free-form syntax.

The Transaction Processor in Operation

To show the scope of the QTP in operation, here are four short examples of situations which occur frequently and which normally require specially written programs.

1. New product number

The manager of inventory control wants to assign a new product series "M" to all series "S" product numbers greater than 6000. With QTP, this task could be accomplished by entering the following statements:

```
>ACCESS PRODUCTS
>SELECT IF PRODUCT-CODE = "S" AND PRODUCT-NUM > 6000
>FILE PRODUCTS UPDATE
>  ITEM PRODUCT-CODE FINAL "M"
>GO
```

The ACCESS statement specifies which file(s) are to be read-in this case, the file PRODUCTS. The SELECT statement then restricts the selection of records from the product file to those records to be changed. The FILE and ITEM statements specify the changes to be made to selected records. The GO statement causes the QTP request to be executed.

2. Organizational change

The San Francisco branch has been reorganized and is now part of California branch. All reference to San Francisco is to be deleted and all records for San Francisco employees are to be updated to reflect their new status as records of California branch employees.

```
>ACCESS BRANCHES LINK TO EMPLOYEES LINK TO BILLINGS
>SELECT IF BRANCH-NO OF BRANCHES = "SF"
>FILE EMPLOYEES UPDATE
>  ITEM BRANCH-NO FINAL "CA"
>FILE BILLINGS UPDATE
>  ITEM BRANCH-NO FINAL "CA"
>GO
```

The ACCESS statement in this example illustrates multi-file access. Keyed linkages between files can typically be performed automatically, using information in QSCHEMA. The ITEM statements in this example set BRANCH-NO to "CA" in the selected EMPLOYEES and BILLINGS records.

3. Culling obsolete data

A company wants to streamline their customer file and delete anyone on their mailing list who hasn't corresponded for over a year.

```
>ACCESS MAIL-LIST
>SELECT IF 365 < (DAYS (SYSDATE) - DAYS (RESPONSE-DATE))
>FILE MAIL-LIST DELETE
>GO
```

The FILE statement in this example deletes all records of MAIL-LIST that have satisfied the condition in the SELECT statement.

4. Reformatting a file

QTP will be ideally suited to problems involving the reformatting of files. Assume for instance, that the old customer file shown in Figure 1 is obsolete. The "PYR-SALES" (previous years sales) item is to be dropped; "YTD-SALES" (year to date totals) is to be expanded for larger dollar volumes; item "CUSTOMER-ID" is to be expanded; an item "SALESMAN-CODE" is to be added; and all items are to be re-ordered.

```
OLD CUSTOMER MASTER                NEW CUSTOMER MASTER

CUSTOMER-NAME        X(20)         CUSTOMER-ID          X(10)
CUSTOMER-ID          X(6)          CUSTOMER-NAME        X(20)
CUSTOMER-ADDRESS     X(60)         CUSTOMER-ADDRESS     X(60)
PYR-SALES            9(6)          SALESMAN-CODE        X(6)
YTD-SALES            9(6)          YTD-SALES            9(10) COMP
```

Figure 1

The steps needed to format the new customer file are:

(a) Unload the master file.

```
>ACCESS CUSTOMER
>SUBFILE TMP OUTPUT CUSTOMER
>GO
```

(b) Change the schema, purge and recreate the customer file (details not shown).

(c) Reload the new master file.

```
>ACCESS *TMP
>FILE CUSTOMER ADD
>GO
```

(d) Purge the temporary file.

```
:PURGE TMP
```

The SUBFILE statement creates an ad-hoc file containing specified information. Subfiles automatically contain their own schema and are therefore self-describing. In this example SUBFILE creates a temporary file TMP containing a copy of the customer master file.

QTP automtically performs the following manipulations for commonly named items in the two files:

- changes item type
- changes item size
- changes item order.

QTP IN THE PRODUCTION ENVIRONMENT

The following two examples look in detail at how QTP might handle two common month-end production situations.

1. Adding 1% interest to all invoices over 30 days due

To expand on a typical accounts receivable situation, assume a company has reached the due date for monthly accounts receivable. The account manager wants to add 1% interest to all outstanding accounts and update the master file. QTP performs this task in fewer than 20 specification lines.

```
>ACCESS ACCOUNT-MASTER LINK TO ACCOUNT-DETAIL
>SORT ON ACCOUNT-NO, INVOICE-NO
>TEMPORARY INVOICE-DATE RESET AT INVOICE-NO &
>  INITIAL DATE OF ACCOUNT-DETAIL &
>                IF TYPE OF ACCOUNT-DETAIL="INVOICE"
>TEMPORARY INVOICE-BALANCE RESET AT INVOICE-NO INITIAL 0
>SUM AMOUNT OF ACCOUNT-DETAIL INTO INVOICE-BALANCE &
>      IF TYPE OF ACCOUNT-DETAIL = "INVOICE" OR &
>         TYPE OF ACCOUNT-DETAIL = "INTEREST"
>SUM AMOUNT OF ACCOUNT-DETAIL INTO INVOICE-BALANCE NEGATIVE &
>      IF TYPE OF ACCOUNT-DETAIL = "PAYMENT"
>FILE ACCOUNT-DETAIL ALIAS INTEREST ADD AT INVOICE-NO &
>      IF DAYS(SYSDATE)  > DAYS(INVOICE-DATE) + 30
>  ITEM AMOUNT FINAL INVOICE-BALANCE * 0.01
>  ITEM TYPE FINAL "INTEREST"
>FILE ACCOUNT-MASTER UPDATE AT ACCOUNT-NO
>SUM AMOUNT OF INTEREST INTO BALANCE OF ACCOUNT-MASTER
>GO
```

The account details are accessed and sorted on account number and invoice number.

Two temporary items, INVOICE-DATE and INVOICE-BALANCE are created to hold the date and accumulated outstanding balance of each invoice.

The two SUM statements accumulate the outstanding balance.

The first FILE statement together with the following two ITEM statements create a new detail record for the interest if the invoice is past due.

The last FILE statement and following SUM statement update the account balance to reflect the new interest change.

2. Standard Batched Update

The standard batch update is probably the most universal QTP application. At the end of each day, a company wants to total all money received and prepare for the next day's transactions. With QTP, this assignment could be performed in ten specification lines.

```
>ACCESS BATCH-HEADER LINK TO TRANS
>SELECT IF TOTAL-ENTERED = TOTAL-CALCULATED
>SORT ON ACCOUNT-NO
>FILE ACCOUNT-DETAIL ADD
>  ITEM TYPE INITIAL "PAYMENT"
>FILE ACCOUNT-MASTER UPDATE AT ACCOUNT-NO
>SUM AMOUNT OF TRANS INTO BALANCE OF ACCOUNT-MASTER
>FILE BATCH-HEADER DELETE
>FILE TRANS DELETE
>GO
```

The ACCESS, SELECT and SORT statements retrieve transactions from balanced batches and sort them by account number.

The FILE statement for ACCOUNT-DETAIL creates payment records from the transactions.

The FILE statement together with the following SUM statement update ACCOUNT-DETAIL to reflect the new payments.

The final two FILE statements delete all processed batches and transactions.

TRANSACTION PROCESSOR STATEMENTS

The major specification statements used in QTP will be as follows:

ACCESS: specifies the files to be read, the order in which they should be read and linkage between files.

BUILD: takes all requests defined up to the BUILD statement and saves these requests into a named MPE file for future use.

CHOOSE: specifies an explicit set of data by key for retrieval.

DEFINE: used to define a frequently used expression.

EDIT: specifies that input files be edited according to editing defined in QSCHEMA.

FILE: defines an output action to be performed on a file. These actions are:

ADD add if record does not exist
UPDATE add if record does not exist, else replace
REPLACE replace if record exists
DELETE delete if record exists.

ITEM: indicates specific items to be assigned initial or final values, or to accumulate totals.

RESET: resets status control options to original status.

SELECT: restricts selection of records for processing to those which satisfy a condition.

SORT: specifies the order in which records are sorted.

SUBFILE: creates a sequential file (MPE).

Does not require the file to exist in the QSCHEMA. Will produce its own schema information in the header of the file, and be accessible to both QTP and QUIZ.

TEMPORARY: creates a temporary-item which does not exist in the database.

DESIGN CONSIDERATIONS

To arrive at a smoothly functioning product, certain design considerations were uppermost in the thoughts of the development team.

- The desirability of a specification based language to insulate users from procedural constructs.
- The need to support complicated production runs as well as ad-hoc file maintenance functions.
- The need for efficient run time performance. Since QTP will run repetitively against bulk volumes of data, its design will differ significantly from a data entry system which requires a high degree of user interaction.
- The need for effective interaction with QUICK to allow class data changes in conjunction with data entry.
- The automatic insulation of users from migrating secondary and other file positioning problems inherent in IMAGE and KSAM file updates.

SUMMARY

Application systems on the HP3000 are typically composed of numerous data entry screens, numerous reports and a relatively small number of batch processes which run on a regular basis at day-end, month-end and year-end. Significant progress has been made towards eliminating the need to custom program data entry and reporting functions. Very little attention has been paid to the development of productivity tools to perform standard batch processing functions. The transaction processor is designed to perform most standard batch operations as well as a wide range of file manipulation functions. QTP, together with its companion products QUIZ, QUICK, and QSCHEMA, will form a complete application generator for the HP3000. Complete systems can be built using these components with major savings in programmer resources applied to development and maintenance, and with real gains in data integrity, system consistency and flexibility.