A MANAGEMENT INFORMATION SYSTEM

FOR DATA ENTRY

A project submitted in partial satisfaction of the requirements for the degree of Master of Science in

Computer Science

by

Dennis Erle Frady

May 1982
The Project of Dennis Erle Frady is approved:

Philip Gilbert

Nirmal K. Mishra

Russell J. Abbott, Committee Chairperson

California State University, Northridge
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF FIGURES</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>SECTION</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2 ANALYSIS</td>
<td>7</td>
</tr>
<tr>
<td>3 CONCEPTUAL ORGANIZATION OF THE MANAGEMENT INFORMATION SYSTEM</td>
<td>28</td>
</tr>
<tr>
<td>4 PROJECT EVALUATION</td>
<td>38</td>
</tr>
<tr>
<td>APPENDIX</td>
<td></td>
</tr>
<tr>
<td>A DETAIL DESIGN</td>
<td>44</td>
</tr>
<tr>
<td>B PROGRAM LISTINGS</td>
<td>57</td>
</tr>
<tr>
<td>C SAMPLE REPORTS</td>
<td>100</td>
</tr>
<tr>
<td>D USER MANUAL</td>
<td>111</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master File Record Format Table</td>
<td>30</td>
</tr>
<tr>
<td>Edit and Update Process</td>
<td>34</td>
</tr>
<tr>
<td>Period and Year-to-Date Process</td>
<td>35</td>
</tr>
<tr>
<td>Operator Analysis Process</td>
<td>36</td>
</tr>
<tr>
<td>Initialization Process</td>
<td>37</td>
</tr>
<tr>
<td>Update Transaction Table</td>
<td>114</td>
</tr>
</tbody>
</table>
ABSTRACT

A MANAGEMENT INFORMATION SYSTEM

FOR

DATA ENTRY

by

Dennis Erle Frady

Master of Science in Computer Science

The project consisted of developing a system which provides management information about data entry in a data processing organization. All the necessary steps to develop a management information system were performed which consisted of the requirements analysis, designing the system and programs, coding, and implementation.

The important part of the project is the methodology used to perform the requirements analysis. The methodology used was Business Systems Planning (BSP) developed and supported by IBM. BSP can be used to identify the information systems requirements for a large corporation or to identify the information requirements for one
component of an organization. This project used BSP for one component of a data processing organization, which was the Data Conversion Section of the Data Service Bureau in the City of Los Angeles.

BSP is successful because it is a top down approach to requirements analysis. That is, the information system requirements are identified with the participation of top management. Therefore, the management information systems which are the result of BSP requirements are consistent with management needs and objectives and not the traditional management information systems which are the by-product of systems which support operational needs.

After the requirements analysis was completed, the management information system was designed using top down design methodology. The programs were written in PL/1 using structured programming, and the system was implemented.

At the time of this project report, the management information system has been running for three years and successfully solved the problems and is satisfying the objectives of the management.
Section 1

INTRODUCTION

Data Service Bureau

The Data Service Bureau (DSB) is the data processing department for the City of Los Angeles. The DSB services forty operating departments, operates twenty-four hours per day for the Los Angeles Police Department and the Los Angeles Fire Department. The DSB provides the analysis, design, programming, documentation, data entry, and operation of all data processing applications for these forty operating departments costing fifteen million dollars per year. And, due to the need for productivity increases by the operating departments, the Data Service Bureau is expecting to grow to provide many of the productivity increases.

Data Conversion Section

The Data Conversion Section performs the data entry function in the Data Service Bureau. Data entry was performed in a centralized location and office for all but three of the four hundred fifty data processing applications at the time this project was initiated. The other three data processing applications had been decentralized to the operating department which keyed the
data using terminals online to the general purpose computers which ultimately process the data.

The Data Conversion Section used two minicomputers with thirty-two keystations and thirty keypunch machines on two shifts.

The Data Conversion Section was staffed with one hundred fourteen operators, ten group supervisors, one shift supervisor for each of two shifts, and one data conversion supervisor. The Data Conversion Section was also spending one hundred five thousand dollars per fiscal year to contract some of the data entry to an outside agency and fifty thousand dollars in overtime per fiscal year.

The Data Service Bureau management could not justify the above expenditures because the only measure available was total keystrokes, which was an inadequate measure to budget or evaluate performance. The inadequacies are specified in detail in Section 2, Analysis.

Therefore, a project was started to develop new measures and to implement a reporting mechanism to provide management information.

The Project

The project consisted of doing the analysis, design, programming, and implementation to provide a management information system which would support the objectives of the Data Service Bureau. The following paragraphs (1)
summarize the methodology used to perform the analysis; (2) state the objectives which are the major result of the analysis and which are requirements for the management information system; and (3) summarize the results of the project.

I. Methodology

The methodology used is Business Systems Planning (BSP) which was developed and is supported by IBM. BSP is being used by many organizations to define their management information requirements and is applicable to any size organization from a small business to a corporation the size of General Motors.

The major difference between BSP and the traditional means of providing management information is that BSP uses a top down methodology which affords management the opportunity of specifying the information requirements whereas the traditional approach provides management information as a by-product of operational information needs. That is, the information requirements were specified by management for management use and, therefore, the probability that the information system will meet management requirements is greater.

BSP is a formal thirteen step approach to conducting an analysis. However, because only one person conducted the analysis, because only one
line management series was involved, and because only one operational process was of interest, the BSP was simplified. The following steps were used to perform the analysis:

A. Define the Processes

   This step defines the processes which one performs in the subject organization to produce the product.

B. Define the Data Classes

   This step defines the data necessary to support the desired management information.

C. Analyze Business/Systems Relationships

   The purpose of this step is to define how current systems are supporting the business.

D. Determine the Executive Perspective

   The purpose of this step is to validate the processes, the data, and the systems defined in the above steps and to determine the objectives, the information requirements, and the current problems with the information currently provided.

E. Assessing Business Problems

   This step analyzes the information problems identified from the Executive Perspective.

F. Define the Information Architecture

   This step defines the information systems and the accompanying data required to support the
objectives and solve the information problems identified from the Executive Perspective.

II. Objectives

Management objectives provide the motivation for the information required to manage an organization. The first three objectives below are common to any organization. They are the specific objectives to support the general management functions to plan, to schedule, and to control. The last two objectives are unique to the Data Conversion Section and were developed to improve the data entry function in the City of Los Angeles.

The objectives of the Data Service Bureau and the City of Los Angeles are:

1. To budget the necessary equipment and people to meet workload requirements.
2. To schedule the workload for optimal performance.
3. To evaluate the performance of the Data Conversion Section, including evaluation of each operator and each supervisor as well as the entire Data Conversion Section.
4. To eliminate data entry onto cards, because cards require heavy manual processing which is error prone.
5. To distribute data entry to the department which is responsible for the creation of the data,
because the department which creates the data can enter the data with greater accuracy and efficiency.

III. Results

The management information system has produced excellent results by supporting the management objectives and meeting managements information requirements.

The net result is that the City of Los Angeles is saving over $300,000 per year and at the same time the data is being keyed and processed in a more timely manner. And the prospect is good that the entire data entry function may some day be decentralized and, in fact, combined with other decentralized functions such as word processing.
Section 2

ANALYSIS

The analysis was performed using a subset of the Business Systems Planning (BSP) methodology developed and supported by IBM. A subset was able to be used because the analysis was concerned with only the budget and performance (or operational) aspects of the Data Conversion Section. The analysis was not concerned with such business functions such as disbursement of funds, materials, or facilities. Furthermore, the analysis was performed by one individual interacting with one line management team as opposed to the usual BSP involving six to eight individuals working with the entire management team. Therefore, some of the more formal steps in BSP were not necessary.

The analysis was performed using the six most central steps in BSP. The results of these six steps are contained in the following paragraphs.

A. The first step in the analysis is to define the processes. A process is a group of logically related acts which are performed by the organization.

Data entry is performed in two basic steps, writing and verifying, in that order. The data is first written onto a storage medium, such as cards,
tape, or disk. Then the same data is verified by
keying the data a second time. During verification,
if the keying differs from what was written, the
machine will lock and the verifier must either change
what was written or allow what was written to remain
unchanged. If a change is made, that is called error
correction. The specific processes for data entry
are below.
1. A supervisor divides a set of like source
documents into what is called batches. The source
documents are the documents on which the data
resides from which the data will be keyed. The
source documents in each batch must be the same
because the operators use formats (or programs)
to aid the speed of the keying.
2. An operator obtains a batch and opens the
batch on a computer, or writes the start time on
a log for cards.
3. The operator keys the data. If this is
verification, the operator corrects the errors.
4. When the operator completes the keying, the
operator closes the batch on the computer, or logs
the stop time for cards, and returns the batch to
the supervisor.
B. The second step defines the data classes which
are required for management information. This step
does not produce specific fields in specific records, but rather the general types of data required.

Three performance factors were identified, which are:

1. The speed which an operator keys.
2. The number of hours actually keying.
3. The error correction rate.

One data class was identified as being required for budget formulation which is the expected number of hours actually keying for each operator over a specific time period (such as day, or week).

C. The third step is to analyze the business/system relationships to determine how current systems are supporting the business.

The current system reported keystroke hourly rate for each operator and total keystrokes per accounting period. The City of Los Angeles operates on thirteen four week accounting periods per fiscal year.

No workload calculations were performed which could be used for budget purposes. A manual calculation was made from the total keystrokes per accounting period.

No workload calculations were performed for each application which could be used for scheduling or
used to determine which applications were the most beneficial to distribute to the user department.

This analysis discovered that the data was being collected which could be effectively used to provide the management information. Each time an operator writes a batch or verifies a batch, the minicomputer (or the operator in the case of cards) logged a data record containing the following fields.

1. The type of machine used, eg. terminal or the computer or keypunch machine.
2. The application number which is unique to each application being keyed.
3. Each batch is assigned a batch number which is recorded in the data record. Source documents are divided into groups called batches. Batches provide the ability to assign work to various operators in time increments which are neither too long nor too short.
4. The type of keying being performed, write or verify.
5. The keystation (terminal or keypunch machine) number upon which the work was performed.
6. The identification number of the operator who performed the work. The identification number also identifies in which group and shift the operator was assigned.
7. Number of records written or verified.
8. Number of minutes taken to perform the work.
9. Number of keystrokes used to perform the work. The work produced on cards contained blanks for this field.
10. Number of records corrected by the verifier.
11. The shift, date, and time the work was started (called opened).
12. The shift, date, and time the work was completed (called closed). One record was created for each time the operator completed work. Normally, two records per batch are produced, one record to write a batch and one record to verify a batch.

D. The fourth step is to determine the executive perspective by interviewing the affected managers in order to (1) verify the processes and data classes; (2) establish the management objectives; and (3) compile the information related problems with which the managers are concerned.

It must be noted that the interview process and the establishment of the processes and data classes is an iterative process. Therefore, this step of the BSP may be performed more than once or some of the managers may be interviewed more than once for the purposes of clarification and agreement. However,
the processes and data classes were ultimately verified as previously reported.

The objectives were easily established because they were generally known at the time the project was initiated. This is not always true in a BSP, but in this BSP the objectives were known early due to the previously stated limited nature of this BSP.

Therefore, the most significant result of this step was the documentation of the management information problems stated in the following paragraphs.

1. The lack of resource information by application made budgeting difficult when a new application was added. If resource information was available, reasonable budget requirements could be calculated by comparing the new application to an existing application. Without this information, supervisor estimates were given for budget formulation which in turn led to long debates between the Data Service Bureau and the City Administrative Officer, who is responsible for formulating the initial City budget.

2. The lack of resource information by application made budgeting difficult when an existing application was distributed to the user department. If resource application were available, the exact resources used to perform the data
entry work would be removed from the budget without any discussion. Without this information, supervisor estimates were given which led to similar debates as described above.

3. Cards were assumed to be 80 keystrokes. However, most of the card applications used less than 80 columns. Therefore, when an application was converted to the minicomputer, the decrease in keystrokes appeared to warrant a decrease in budgeted resources. This led to confusion over the budget requirements which also help to obscure and complicate the previously stated problems for new or distributed applications.

4. The number of cards punched were estimated. The estimate by the supervisors was usually high in order not to be unfair to the operator. However, this led to even higher keystrokes for card applications further complicating the above stated problems.

5. The lack of resource information by application prevented management from scheduling resources to accommodate peak workloads. Management could schedule vacations, overtime, or outside data entry resources to accommodate peaks and valleys if resources by application for each accounting period were available.
6. The lack of resource information prevented an accurate assessment of the size of a backlog, management was guessing when to use overtime or outside resources which could lead to needless expenditures.

7. Keystrokes per hour were used to evaluate keypunch operators. However, applications are not all the same difficulty. A good performance ranged from 6,000 keystrokes per hour to 12,000 keystrokes per hour depending upon which application was keyed. Also, keystrokes per hour for cards were inflated due to the estimates made for the number of cards keypunched and due to the assumption of 80 keystrokes per card. This led to many grievances by operators who believed they had been given the more difficult work or from operators who believed the work on the minicomputer was leading to smaller keystroke rates. Therefore, yearly evaluations and promotions led to many grievances and management had no quantitative defense. The results were low morale and low performance.

8. No information existed which could be used to evaluate a supervisor, a shift supervisor, or the entire operation. Therefore, low performance
9. The minicomputer being used was old and was experiencing considerable downtime. The age of the minicomputer was causing the vendor problems maintaining technically trained personnel. Also, the minicomputer lacked sufficient capacity to process the desired volume of converted card applications. Therefore, a project was initiated to find a new minicomputer with sufficient capacity to process the entire Data Conversion workload and would provide acceptable availability.

Although replacing the minicomputer was not part of this project, the replacement project is mentioned here because of its critical nature to the objective of converting card applications to the minicomputer.

10. The lack of resource usage information for each application prevented an accurate cost/benefit assessment to be made for converting the card applications or distributing data entry to user departments.

If the volume of keying is not sufficiently large, the cost savings would not be sufficient to offset the program development investment. Generally, the greater the volume, the greater
the cost savings. Therefore, resource usage information was required for management to make decisions on which applications to convert or distribute.

E. The next step was to analyze the business problems. Considerable analysis of the problems was performed in the last step in order to state them clearly. This step was used to produce a top level definition of the management information system. That is, this step was used to make certain basic design decisions on how, what, and when the management information was to be produced. These basic design decisions are stated in the following paragraphs.

1. As the analysis was proceeding, an attempt was made to compile information for each application with the aid of a word processing system from the existing available data. Although the information was ultimately compiled, two problems were identified with this method. First, the compilation of the management information took over five hundred hours of employee time. This amount of time could not be made available as an ongoing task. Second, the amount of manual compilation resulted in errors. No probability analysis was made of the deviation due to the errors, but the heavy manual compilation of
statistics, which had come from computer readable data, was ludicrous.

Therefore, the basic decision was to computerize the production of the management information.

2. For the purpose of budgeting, the number of operators required to meet the workload is necessary. Because the City operates on thirteen accounting periods of four weeks each, the calculation should be performed each accounting period.

The start (open) and end (close) times for each batch were available and also the elapsed time for each batch to write and for each batch to verify could be calculated. By adding the elapsed time for each batch, total keying time could be calculated. If the total hours were divided by the number of hours an operator was expected to be keying, personnel resources would be the result. However, the number of hours an operator was expected to key was not available. In order to produce this number, a detailed study of the operators time was performed.

The minicomputer automatically logged the time a batch was started, the time a batch was completed, and the elapsed time. However, other time such as training, obtaining batches to be
done, and returning finished batches is productive time which must be allowed. Non-available time such as sick leave, vacations, or breaks must also be allowed. The study used two years of detailed attendance records to calculate non-available time and an outside auditor observed and calculated the productive time not recorded by the mini-computer. The results of the study, in percentages, are documented below.

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiving and returning work</td>
<td>6.27</td>
</tr>
<tr>
<td>Personal time</td>
<td>8.15</td>
</tr>
<tr>
<td>Breaks</td>
<td>6.45</td>
</tr>
<tr>
<td>Vacation</td>
<td>4.15</td>
</tr>
<tr>
<td>Holidays</td>
<td>4.42</td>
</tr>
<tr>
<td>Sick Leave</td>
<td>4.18</td>
</tr>
<tr>
<td>Other Leave</td>
<td>4.37</td>
</tr>
<tr>
<td>Training</td>
<td>3.56</td>
</tr>
<tr>
<td>Miscellaneous non-available time</td>
<td>0.47</td>
</tr>
<tr>
<td>Administrative time</td>
<td>2.82</td>
</tr>
</tbody>
</table>

The total percentage of time an operator is not keying is 45.5. On the average an operator should be keying 54.5 percent of the time or 21.8 hours per week. This is an average which can be used over a long period of time.
The keypunch machines do not automatically record the data. The data is manually recorded. In order to simplify this manually recorded data, operators assigned to keypunch record all hours for which they are at work. Therefore, they must average 78.66 percent or 31.5 hours over a long period of time.

For the minicomputer, personnel resources are calculated from total hours and from the standard hours keying per week developed in the previous paragraphs. As was stated, the average for hours keyed per week is 21.8. Because each accounting period in the City of Los Angeles is four weeks, one operator keys an average of 87.2 (21.8 times 4) hours per accounting period. Therefore, personnel resources are calculated by dividing the total hours by 87.2 for each accounting period. For a year-to-date report, the total hours must be divided by the product of 87.2 and the number of periods being reported.

For card applications, one operator keys an average of 126 (31.5 times 4) hours per accounting period. For a year-to-date report, the total hours must be divided by the product of 126 and the number of periods being reported.
Therefore, personnel resources could be calculated by application, by user department, or by total City using the results of the above analysis.

The number of hours an operator keys is not only important for budget purposes, it is also important for performance evaluations of operators, of supervisors, and of the total Data Conversion section. Therefore, the standard time that an operator should be keying while at work was set at 72.25 percent or 28.9 hours for each 40 hours at work.

3. The number of hours that an operator is expected to be keying is not the only factor in evaluating performance. The other two factors are speed and accuracy. Speed is discussed in this section and accuracy is discussed below.

Keystrokes had been used to evaluate speed. However, due to the problems identified earlier in the analysis, another approach was desirable.

The solution developed was to compare actual records per hour to an established standard records per hour. The standard records per hour is the average records per hour. Using data which had been collected over eighteen months of keying, an average records per hour for write and for
verify for every application was calculated by a program which was used only once.

Averages for applications with less than 50 hours of keying time per year were not calculated because applications with so few hours would not be statistically valid.

Efficiency is now defined as the actual records per hour divided by the average (standard) records per hour. In order to make the efficiency easy to use, it is multiplied by 100. Therefore, an efficiency of 100 is average, below 100 is below average, and above 100 is above average.

Operators work on many different batches from many different applications and also may write or verify. A weighted efficiency for both write and verify is calculated, using hours as the weight. The overall efficiency is required and can be calculated as the weighted average of the write and verify efficiencies.

4. Accuracy is difficult to fully computerize due to factors such as the difficulty of the application or the unavailability of data.

Therefore, the decision was made to monitor accuracy as follows:

The number of corrections made by the verifier is contained in the data record. An acceptable
error rate for the write operator is one error in 500 keystrokes or one error record in ten. However, the difficulty of the application or the legibility of the source document are two factors which must be considered by supervisors when evaluating error rates. The number of errors and the number of records verified is the information required to monitor errors written. No other information is required due to the mitigating factors, such as the ones mentioned above, which the supervisor must evaluate.

Accuracy for a verifier obviously does not appear in the data record. However, all data entered by the Data Conversion Section is edited by programs on the central computers. Edit reports produced from these programs give the number of records processed and the number of records in error. The Data Control Section at the Data Service Bureau has specific procedures for each application when excessive errors are reported. These records which were in error must be rekeyed by the Data Conversion Section. Any source documents which must be rekeyed represent a delay to the departments who are responsible or use the resulting information. The feedback provided by Data Control, the rekeying effort, and complaints
by departments provide the information necessary to rate the accuracy of the verifier.

F. The final step of the analysis is to define the system and how the system will satisfy the objectives and solve the problems identified in the executive perspective step. The following reports were generally defined. See Appendix C for examples of the reports.

1. A year-to-date report to be produced each accounting period was defined which would contain, by application:
   a. Application number
   b. Application name
   c. Records written
   d. Records verified
   e. Records corrected
   f. Hours written
   g. Hours verified
   h. Total hours written and verified
   i. Personnel resources
   j. Average record length
   k. Average write rate (records per hour)
   l. Average verify rate (records per hour)
   m. Cards or minicomputer
   n. Efficiency calculation for write, verify and application
Totals for each department are required for records written, records verified, records corrected, hours written, hours verified, total hours and resources used. The same totals for card application and minicomputer applications were required. Finally, totals for all of Data Conversion were required.

This report would allow management to budget by use of the personnel resources. Applications discontinued could be subtracted from the total personnel resources and new applications could be compared to existing applications and added to the personnel resources.

Therefore, using this report would solve problems 1, 2, 3 and 4 from Part D and satisfy objective number 1.

Also, the efficiency calculations would aid the supervisor in assessing the degree of fairness to the operators when evaluating the operators' speed. This would act as a control on the solution to problem number 7.

Management could also use this report to identify which applications were the most beneficial to convert from cards and to distribute to a user department. Therefore, this report would
satisfy objectives 4 and 5 and solve problem number 10.

2. A period report was defined to contain the same information as the year-to-date report, except using only one period of data.

The period report would be useful to identify peak workloads and backlogs which would aid management in the solution of problems 5 and 6.

Also, the efficiency calculations would act as a further control on the solution to problem number 7 by alerting supervisors of sudden changes in efficiency for a particular application.

Changes in the size of the batches can have beneficial or detrimental effect on performance. Too few source documents in a batch causes the operators to spend too much time obtaining and returning batches to the supervisor.

Too many source documents frustrate the operator and force concentration over too long a time period. Therefore, supervisors could adjust batch sizes, observe the effects, and finally arrive at optimum batch sizes which would increase productivity and aid the scheduling of the workload, thus, aiding the solution of problems 5 and 6 and satisfying objective number 2.
3. A report showing the detail of operator activity was defined. For each operator the record for each batch was required in the chronological order in which the work was performed. Also, the record length, averages (standards), and efficiency for each batch would be displayed. Totals for records written, records verified, hours written, hours verified, and efficiencies for write, verify, and write and verify were calculated and reported.

This report would partially satisfy objective number 3 and would solve problem number 7.

In addition to the detail operator report, a summary operator report would be produced showing the totals for each operator including the efficiencies and would calculate totals and efficiencies for each supervisor, shift, and the entire Data Conversion Section.

The summary operator report would satisfy the remaining portion of objective number 3 and solve problem number 8.

The results of this step in the analysis serve as a requirement document for an automated system which would satisfy the objectives and the informational related problems.
The only problem not solved by the automated system would be problem number 9 which was the need to replace the minicomputer. Without the replacement, significant progress could not be made towards satisfying objective number 4.

Although more detail on the conversion to a new minicomputer is given in Section 4 "Project Evaluation", for the sake of completeness it is noted here that the minicomputer was successfully replaced.
Summary

The objective of the management information system is to produce the three report types defined as a result of the analysis: (1) the period report; (2) the year-to-date report; and (3) the detail operator analysis report and the summary operator analysis report.

The master year-to-date file contains the year-to-date information for each application. The master file is maintained by an edit and update program. An initialization program is used to initialize the file for each new fiscal year.

The management information system consists of the file, the edit and update process, the period and year-to-date process, the operator analysis process, and the initialization process which are described in the following paragraphs and the respective data flows.

File Description

In order to understand the system, it is necessary to understand the master file. The file has three types of records, (1) one file header record; (2) one or more
department or agency header records; and (3) application
records.

The first record in the file is called the file
header. This record contains the accounting period and is
updated each period by the period program.

One department or agency header record is maintained
at the beginning of each department or agency section of
the file. The record contains the name of the department
or agency.

Following the department or agency header record are
the application records associated with the previous de­
partment or agency record. The table on the next page
shows the fields in the application records.

The records in the file are maintained in an ascending
sorted sequence on the application number.

The sorted file begins with the file header record
followed by one group of records for each department or
agency beginning with a department or agency header
record.
<table>
<thead>
<tr>
<th>FIELD</th>
<th>NAME</th>
<th>CHARACTERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>APPLICATION NUMBER</td>
<td>6</td>
</tr>
<tr>
<td>2.</td>
<td>FORMAT</td>
<td>4</td>
</tr>
<tr>
<td>3.</td>
<td>TITLE</td>
<td>20</td>
</tr>
<tr>
<td>4.</td>
<td>AVERAGE RECORD LENGTH</td>
<td>3</td>
</tr>
<tr>
<td>5.</td>
<td>AVERAGE WRITE</td>
<td>3</td>
</tr>
<tr>
<td>6.</td>
<td>AVERAGE VERIFY</td>
<td>3</td>
</tr>
<tr>
<td>7.</td>
<td>ESTIMATE WRITE</td>
<td>3</td>
</tr>
<tr>
<td>8.</td>
<td>ESTIMATE VERIFY</td>
<td>3</td>
</tr>
<tr>
<td>9.</td>
<td>EQUIPMENT</td>
<td>1</td>
</tr>
<tr>
<td>10.</td>
<td>CODED</td>
<td>1</td>
</tr>
<tr>
<td>11.</td>
<td>RESERVED</td>
<td>3</td>
</tr>
<tr>
<td>12.</td>
<td>RECORDS WRITTEN</td>
<td>10</td>
</tr>
<tr>
<td>13.</td>
<td>RECORDS VERIFIED</td>
<td>10</td>
</tr>
<tr>
<td>14.</td>
<td>CORRECTIONS</td>
<td>10</td>
</tr>
<tr>
<td>15.</td>
<td>HOURS WRITTEN</td>
<td>10</td>
</tr>
<tr>
<td>16.</td>
<td>HOURS VERIFIED</td>
<td>10</td>
</tr>
</tbody>
</table>
Edit and Update Process

The edit and update process consists of an edit program and an update program. The process edits the update records to be applied to the master file. An edit report is produced showing each update record and any errors found by the edit program. The update records which contain no edit errors are stored in a temporary file. The temporary file is sorted and used as input to the update program. The update program outputs a new master file with the updates applied and a report showing each update and any errors found during the update. Records found in error by the update program do not change the master file. Errors found during edit or update may be corrected and processed in a subsequent run of the edit and update.

Three types of updates are possible. A record may be added, deleted or modified. The only fields which can be updated are application number, format, title, average write, average verify, equipment, and coded. The remaining fields are calculated by the programs and cannot be updated using the edit and update process.

Period and Year-to-Date Process

The period and year-to-date process sorts the data records into ascending application number order then uses the master file and the sorted data records as input. The outputs are a new master file, the period report, and
the year-to-date report. The period report summarizes the workload for the reporting period and the year-to-date report summarizes the workload for all periods to date.

Operator Analysis Process

The operator analysis process sorts the data records into ascending order by operator number and within each operator number, sorts the data records in ascending order by date and time. The sorted data records are used as input by the operator analysis report program which also uses the master file to build a table of application numbers and the associated averages. A detail operator report and a summary operator report are produced. The detail operator report shows each data record for each operator in chronological sequence and the totals for each operator. The summary operator report shows the totals for each operator, group of operators who report to a supervisor, shift of operators, and totals for the entire Data Conversion Section.

Initialization Process

The initialization process is run at the beginning of the fiscal year. The initialization program inputs the master file and outputs a new master file with zero for all totals and deletes any application record which had
no activity during the fiscal year just ended. A report is produced which shows all deleted application records.
Edit and Update

EDIT REPORT

EDIT

EDITED UPDATE RECORDS

SORT

SORTED UPDATE RECORDS

UPDATE

MASTER FILE

UPDATE REPORT

NEW MASTER FILE
Period and Year-to-Date

PERIOD DATA RECORDS

SORT

SORTED DATA RECORDS

MASTER FILE

PERIOD PROGRAM

NEW MASTER FILE

YEAR TO DATE

YEAR-TO-DATE REPORT

PERIOD REPORT
Operator Analysis

DATA RECORDS

SORT

SORTED DATA RECORDS

MASTER FILE

OPERATOR ANALYSIS PROGRAM

DETAIL REPORT

SUMMARY REPORT
Initialization

MASTER FILE

INIT. PROGRAM

INITIALIZED MASTER FILE
Section 4

PROJECT EVALUATION

The project must be evaluated in terms of the objectives achieved and the problems solved. The management information system has been in use and under evaluation for three years. Therefore, the results have withstood the test of time and are not merely short term results which could be attributed to factors not related to the management information system. The paragraphs below detail the results of the project for each objective followed by a summary of the project's success.

1. The first objective was to provide the management information necessary to formulate yearly budgets. The problems associated with the use of keystrokes were removed by using hours keyed to calculate personnel resource requirements. Because the hours keyed are controlled through the use of efficiency and accuracy, the management of the Data Service Bureau as well as the City Administrative Officer have accepted the budget calculations without debate each of the last three years. Therefore, the objective has been successfully satisfied.

2. The second management objective was to be able to schedule the workload for optimal performance. The
management information system calculates workload information and keying efficiency information by application for each accounting period and also calculates keying efficiency for each operator, group of operators, shift of operators, and the entire Data Conversion Section. The above information allows supervision and management to project workload peaks and valleys, calculate backlogs, and detect performance problems.

By projecting workload peaks and valleys, supervisors and managers can schedule vacations or other leaves during periods of low workload and follow hiring practices which maximize the available people during periods of peak workload.

By accurately calculating backlogs, managers are able to make more intelligent choices in the use of overtime funds or the use of outside agency contingency funds. Also, backlogs can be delayed. Some applications are more efficiently keyed in higher volumes.

Not all operators work well on all applications. By carefully analyzing operator performance, applications can be scheduled in order to optimize the use of the operators by effectively dealing with the operator's likes and dislikes.
The ability to schedule workload is a major factor in optimizing performance and the results were very significant. In the first fiscal year that the management information system was used, only $110 was used for overtime, whereas in the previous year $50,000 of overtime was incurred. Also, 90% of the outside contingency fund was unused and returned to the City's general fund at the end of the fiscal year. In the previous fiscal year, over $100,000, which was three times the original amount of the contingency fund, had been necessary. The most important result, but the least able to measure, was that the data was more timely keyed and, therefore, more timely processed for the various City agencies.

3. The third objective was to be able to evaluate the performance of the Data Conversion Section. The management information system calculates and reports efficiency by operator, group of operators, shift, and totals for the entire Data Conversion Section. Hours and errors are also reported which act as a control on the efficiency calculation. Without hours and errors as controls, it would be possible to make efficiency look extremely good at the expense of errors or by doing very little work very fast. An operator can be evaluated on efficiency assuming that the operator's hours and errors are within the
established standards. Using data over an eighteen month period previous to the implementation of the management information system, standards were established as to the minimum hours and minimum efficiency allowed for an operator. Industry wide standards were used for errors.

Efficiency and average hours were calculated and reported for each group of operators, shift and section.

The results are extremely good. Promotions were based on more objective criteria, such as average hours, errors, efficiency and less on the previous method of supervisory rating. Although supervisory rating is still used, it is less than one third the total score on the promotional examination conducted by the Data Service Bureau. Morale, especially among the good performers, was increased significantly. In the few cases where disciplinary measures were required, very little effort was expended by managers in the grievance process. Both the employee representative organizations and the civil service analysts easily accepted the criteria and measurements used in disciplinary cases.

The most important result was that efficiency increased by 10% while minimum standards for hours and accuracy were maintained.
4. The fourth objective was to eliminate the keying of data onto cards.

As was stated in the analysis, a prerequisite project was necessary which consisted of replacing the minicomputer in order to solve the reliability and capacity problems. This project was successfully accomplished at the same time the management information system was developed. In order to replace the minicomputer, the Data Conversion Supervisors, who are not professional programmers, wrote over 300 programs in two similar data entry languages. In view of the trend to have non-professional programmers perform more of the functions of programmers, this was a significant accomplishment.

Once the minicomputer was replaced, the conversion effort progressed well. After three years, the keypunch machines had been reduced from 30 to 2 and the number of operators keying onto cards were reduced from 50 to 2. The applications remaining on cards are either not cost effective to convert or the application will be replaced thus eliminating the cards as a result of the replacement.

Therefore, this effort is clearly a success.

5. The fifth objective was to gain the efficiencies possible by performing the data entry in the department where the data is created.
The results of this effort have been outstanding. In three years the Data Conversion Section was reduced from 114 operators to 50 without an equal staffing increase in the departments creating the data. In most cases, no staffing increase was necessary in the departments.

In summary, the management information system has produced excellent results by supporting the management objectives. The primary reason that the management information system has been successful is the methodology used in the analysis. The methodology is designed to provide for a top down definition of requirements by management for management use. Therefore, the information ultimately produced by the management information system satisfies the needs of management rather than the traditional management reports which are a by-product of operational information needs.

The net result is that the City is saving over $300,000 per year and at the same time the data is being keyed and processed in a more timely manner. And the prospect is good that the entire data entry function may some day be decentralized and, in fact, combined with other decentralized functions such as word processing.
APPENDIX A

Detail Design
The programs in the Management Information System were all designed using a top down methodology. The detail design is presented in the following sections.

**Edit Program**

The edit program uses an update record as input and produces an edit report and an edited record which is stored for later use by the update program. The formats of the update record is given in the table in Appendix D. The format of the edit report is in the same order as the update record followed by error statements, if any. No edited record is stored if any errors are found. A field is not changed, if blank.

1. Open files and obtain data from the computer.
2. While more update records exist, process update records.
3. Set up return code and return.

1.1 Open files, UPDATE, SYSPRINT, and SYSIN for output records, report, and update records.
1.2 Set end of file flag for SYSIN.
1.3 Obtain and save month, day, and year, separated by dashes.
2.1 Read an input record.
2.2 Zero number of errors and blank edited record.
2.3 If end of file has not been encountered, perform edits.
2.3.1 Edit each field.
2.3.2 Write each record into report.

2.3.3 If no errors, create and write output record; else, write error messages.

2.3.1.1 Edit transaction code; legal transactions are "DELETE", "MODIFY", and "ADD".

2.3.1.2 Edit application number. Must be two digits followed by four blanks or six digits.

2.3.1.3 Edit record length. Must be two digits.

2.3.1.4 Edit coding flag. Must by "Y" or "N".

2.3.1.5 Edit average records written. Must be three digits.

2.3.1.6 If average records written is not blank and no errors are found then calculate hours per thousand records written to the nearest tenth of hour. Calculation is 10000/average records written.

2.3.1.7 Edit average records verified. Must be three digits.

2.3.1.8 If average records verified is not blank and no errors are found, then calculate hours per thousand records verified to the nearest tenth of hour. Calculation is 10000/average records verified.
3.1 Return codes are: 0 if no edit errors.  
4 if some edit errors, but some good records.  
8 if no input or all records are in error.

3.2 Return

_Update Program_

The update program uses the edited records from the edit program as input, inputs the master file, performs the requested transaction, and outputs a new master file. Only two update errors can occur: (1) a record which exists in the master file cannot be added; and (2) a non-existent record in the master file cannot be modified or deleted.

1. Open files, set end of file flags, obtain date from the computer.
2. Process while more updates on more master file.
3. Set return codes and return.

1.1 Open MASTER and UPDATE files for input, and SYSPRINT for output.
1.2 Set up end of file flags for master and update files.
1.3 Obtain and save month, day, and year separated by dashes.
2.1 If the master file read flag is read, then read a record from the master file.
2.2 If the update file read flag is read, then read a record from the update file.

2.3 Blank the output record.

2.4 Set the write flag to write.

2.5 If both the master file and the update file are at the end of file, then end the processing.

2.6 Process three cases based upon a comparison of the system subsystem members from the master file and the update file.

2.7 If write flag is set to write, then write an output record to the new master file.

2.6.1 Process equal case.

2.6.2 Process less than case.

2.6.3 Process greater than case.

2.6.1.1 Set both read flags to read.

2.6.1.2 Output master file record

2.6.1.3 Output update record.

2.6.1.4 If the transaction is an add, then output error message.

2.6.1.5 If the transaction is a delete, then set the write flag off.

2.6.1.6 If the transaction is a modify, then copy the master file to the output record and overlay the output fields with the update fields. Blank field means no update on this field.
2.6.2.1 Copy the master file record to the output record.
2.6.2.2 Set the master file flag to read.
2.6.2.3 Set the update file flag to off.
2.6.3.1 Set the master file flag to off.
2.6.3.2 Set the update file to read.
2.6.3.3 Put the update record in the report.
2.6.3.4 If the transaction is add, copy the update record to the output record; else output error message.

3.1 If no errors occurred, set the return code to zero.
3.2 If some errors occurred, set the return code to four.
3.3 If all updates were in error or the master file was non-existent, then set the return code to eight.
3.4 Return.

Period Report Program

The period report program uses the data collected in data conversion and the master file as input and outputs a period report which represents the workload for the period. The program also adds the workload for the period to the year-to-date totals in the master file to create a new master file.

1. Open files SYSPRINT for the period report, PERDATA to input the data records, MASTER to input the master file, NEWMSTR to output the new master file.

2. Set up the end of file processing for the master files and the period data.
3. Read and process header record.
4. Process all the period data.
5. Sort the department index.
6. Print the department index.
7. Print the keypunch totals.
8. Print the key to disk totals.
9. Print the grand totals.
10. Set return code.
11. Return

3.1 Read header record.
3.2 If the record is a valid header record then increment the period, save the fiscal year, and output the header record to the new master file; else print the error message.

4.1 Read a data record.
4.2 If an end-of-file has been encountered or a new system subsystem number is read then process the new account.
4.3 If a data record contains the same system subsystem as the previous record, then process the record.
4.4 If an end of file was encountered or a new department encountered then process as a new department.
4.2.1 If the record is not the first record then process the totals for the previous system subsystem number.
4.2.2 Search the master file for the system subsystem number just read.
4.2.3 If the search fails, create a new entry for the new master file.

4.2.4 Zero system subsystem totals.

4.2.1.1 Add the system subsystem number totals to the department totals in the department array.

4.2.1.2 Calculate hours written, hours verified, total hours, personnel requirements, write efficiency, verify efficiency, and overall efficiency for system subsystem number.

4.2.1.3 Output system subsystem number period totals and constant data from master file.

4.2.1.4 Calculate record length.

4.2.1.5 Add period totals to the year-to-date totals from the master file.

4.2.1.6 Output new year to date totals for system subsystem number.

4.2.2.1 Copy records from the master file to the new master file until the system subsystem is found or a system subsystem number higher than the desired system subsystem is found. Save department information if a department header record is encountered.

4.3.1 Add keystrokes to total keystrokes.

4.3.2 If write then add record count to total records written, minutes to total minutes for write, and
add record count and minutes to totals for cards verified or key to disk verified.

4.4.1 Add department totals to grand totals.
4.4.2 Output department totals.
4.4.3 Reinitialize department totals.

Year-to-Date Report

The year-to-date report program inputs the master file and outputs the year-to-date report.

1. Open files, SYSPRINT for the report, MASTER for the master file.
2. Read header record.
3. If the record is a valid header record, then save the period number, fiscal year, and calculate the year-to-date and hours and the year-to-date key-to-disk hours, else output invalid header record error message.
4. While there are more master file records, output each record to the report.
5. Sort the department totals array.
6. Print the department totals.
7. Calculate total hours for cards.
8. Calculate and save personnel requirements for cards.
10. Calculate total hours for key-to-disk.
11. Calculate and save personnel requirements for key-to-disk.
12. Output total for key-to-disk.
13. Output year-to-date totals.
14. If year-to-date totals are zero, then set return code to eight.
15. Return.

4.1 Read a record from the master file.
4.2 If a system subsystem record was read, then process the record; else
4.3 Process end-of-file or department header record.
4.2.1 Add system subsystem data to department totals.
4.2.2 If the system subsystem is a card system then add data to card totals; else
4.2.3 Add data to key-to-disk totals.
4.2.4 Calculate write, verify, and overall efficiencies for the system subsystem record.
4.2.5 Output system subsystem year-to-date data.
4.3.1 Add department totals to year-to-date totals.
4.3.2 Output department year-to-date totals.
4.3.3 If department header record was read then initialize new department totals to zero and save department name.

Operator Performance Program

The operator performance program inputs the data records and outputs the detail operator report and the
summary operator report. The master file is used in order to calculate efficiencies.

1. Open files, detail for the detail operator report, summary for the summary operator report, SYSIN for the variable title, SYSPRINT for the error report, and TABLEIN for the master file, input for the data records.

2. Set up end-of-file flag for TABLEIN and input files.

3. Read header record from master file.

4. If valid header record save variable title data; else output error message.

5. Build table from master file.

6. If master file exceeds table size then set return code to sixteen, output error message, end processing.

7. While there are data records, process each record.

8. Output group, shift, and overall totals and calculations.

9. If there were no records for input, then set return code to eight.

10. Return.

7.1 Read a record from input.

7.2 If the operator number differs from the operator number in the previous record then process totals for the previous operator.

7.3 Process the record.
7.2.1 Calculate overall, write and verify efficiency for the previous operator.
7.2.2 Add the operator totals to the group totals.
7.2.3 Calculate operator total hours for write and verify.
7.2.4 Output totals to detail report.
7.2.5 Output totals to summary report.
7.2.6 Zero operator totals.
7.3.1 Find entry in the table with same system subsystem number as the record using binary search.
7.3.2 If a write entry, add to write totals, calculate efficiency and add to weighted totals for write.
7.3.3 If a verify entry, add to verify totals, calculate efficiency and add to weighted totals for verify.
7.3.4 Output record and efficiencies to the detail report.
8.1 For each group, calculate group efficiency, calculate average group hours, output group totals, group efficiency, and average hours, and add totals for shift.
8.2 For each shift, calculate shift efficiency, calculate average shift hours, output shift totals, shift efficiency, average hours, and add totals to overall.
8.3 Output overall totals, overall average hours, and overall efficiency.
Master File Initialization Program

The master file initialization program initializes the master file at the beginning of a new fiscal year by zeroing all year-to-date totals. If no work was performed for a system subsystem number in the fiscal year just completed then the record in the master file is deleted. A report showing the deleted records is produced.

1. Open files, SYSPRINT for the error messages, SYSIN to input new fiscal year, MASTER to input the master file, and the NEWMSTR file for output.

2. Set up end-of-file processing for the master file.

3. Read header record from the master file.

4. If a valid header record then zero period, set fiscal year title, and output the new header record; else

5. Output error message, set return code to eight, end processing.

6. While there are records in the master file, process each record.

7. Return.

6.1 Read record from master file.

6.2 If the record is a department header record or if any work was performed for the system subsystem number, then zero totals and write record, else do not write the record in the master file and write the deleted record into the report.
APPENDIX B

Program Listings
EDITPGM: PROCEDURE OPTIONS (MAIN);

/-------------------------------------------------------------------/
/ THE EDIT PROGRAM EDITS THE UPDATE RECORDS TO BE APPLIED TO THE MASTER FILE. AN EDIT REPORT IS PRODUCED SHOWING EACH UPDATE RECORD AND ANY ERRORS FOUND. THE UPDATE RECORDS WHICH CONTAIN NO ERRORS ARE WRITTEN TO A TEMPORARY FILE WHICH WILL BE USED AS INPUT TO THE UPDATE PROGRAM. 

DCL 1 INPUT_RECORD,
  2 TRANSACTION CHAR(6),
  2 SYSTEM_SUBSYS,
  3 DEPT_NUMBER CHAR (2),
  3 SYSTEM_NUMBER CHAR (2),
  3 SUBSYS_NUMBER CHAR (2),
  2 FORMAT CHAR (4),
  2 TITLE CHAR (20),
  2 RECORD_LENGTH CHAR(3),
  2 AVERAGE_WRITE CHAR(3),
  2 AVERAGE_VERIFY CHAR(3),
  2 ESTIMATE_WRITE CHAR (3),
  2 ESTIMATE_VERIFY CHAR (3),
  2 MEDIA CHAR (1),
  2 CODED CHAR(1),
  2 DIFFICULTY CHAR (1),
  2 SCHEDULE CHAR (2),
  2 RESERVED CHAR (24); 

DCL 1 OUTPUT_RECORD,
  2 SYSTEM_SUBSYS,
  3 DEPT_NUMBER CHAR (2),
  3 SYSTEM_NUMBER CHAR (2),
  3 SUBSYS_NUMBER CHAR (2),
  2 FORMAT CHAR (4),
  2 TITLE CHAR (20),
  2 RECORD_LENGTH CHAR (3),
  2 AVERAGE,
  3 WRITE CHAR (3),
  3 VERIFY CHAR (3),
  2 ESTIMATE,
  3 WRITE CHAR (3),
  3 VERIFY CHAR (3),
  2 FLAGS,
  3 EQUIPMENT CHAR (1),
  3 CODED CHAR (1),
  3 DIFFICULTY CHAR (1),
  3 SCHEDULE CHAR (2),
  2 RECORDS,
  3 WRITTEN CHAR (10),
  3 VERIFIED CHAR (10),
  3 CORRECTED CHAR (10),
  2 HOURS,
  3 WRITTEN CHAR (10),
  3 VERIFIED CHAR (10),
  2 TRANSACTION CHAR (6); 

DCL END_OF_FILE FIXED BINARY INITIAL (0),
 RETURN_CODE FIXED BINARY (31.0) INIT (0),
 DELETE CHAR (6) INIT ('DELETE'),
 MODIFY CHAR (6) INIT ('MODIFY'),
 ADD CHAR (6) INIT ('ADD '),
 BLANK CHAR (1) INIT (' '),
 CURRENT_DATE CHAR (8),
 DASHES CHAR (132) INIT ((132)'-'),
 SEPARATOR CHAR (1),
 LINES FIXED BINARY INIT (100),
 MAXLINES FIXED BINARY INIT (40),
 ERROR_COUNT FIXED BINARY INIT (0),
 ERROR_FLAG FIXED BINARY,
ERRORS(20) FIXED BINARY,
NUMBER_OF_ERRORS FIXED BINARY INIT (0),
ERROR_MESSAGES(20) CHAR (50) INITIAL
('TRANSACTION CODE',
'SYSTEM - SUBSYSTEM NUMBER',
'RECORD LENGTH',
'CODED',
'STANDARD AVERAGE RECORDS WRITTEN',
'STANDARD AVERAGE RECORDS VERIFIED',
)

PAGE FIXED BINARY INIT (0),
TEMP CHAR (10),
COUNT FIXED BINARY INIT (0);
OPEN FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
FILE (UPDATING ) RECORD OUTPUT,
FILE (SYSIN ) INPUT RECORD;
ON ENDFILE (SYSIN ) END_OF_FILE = 1;
ON CONVERSION BEGIN:
  IF ONCHAR ^= BLANK THEN ERROR_FLAG = 1;
  ONCHAR = '0';
END;
CURRENT_DATE = DATE;
CURRENT_DATE = SUBSTR(CURRENT_DATE,3,2) || ' '; ||
SUBSTR(CURRENT_DATE,5,2) || ' '; ||
SUBSTR(CURRENT_DATE,1,2) ;

ADO)
DO WHILE (END_OF_FILE = 0);
  READ FILE (SYSIN ) INTO (INPUT_RECORD);
  NUMBER_OF_ERRORS = 0;
  OUTPUT_RECORD = ' '; 
  IF END_OF_FILE = 0 THEN
    IF INPUT_RECORD.TRANSACTION ^= DELETE &
      INPUT_RECORD.TRANSACTION ^= MODIFY &
      INPUT_RECORD.TRANSACTION ^= ADD THEN
      TRANSACTION_ERROR:
      DO;
        NUMBER_OF_ERRORS = NUMBER_OF_ERRORS + 1;
        ERRORS(NUMBER_OF_ERRORS) = 1;
        END TRANSACTION_ERROR;
        ERROR_FLAG = 0;
        TEMP = INPUT_RECORD.SYSTEM_SUBSYS.DEPT_NUMBER ||
        INPUT_RECORD.SYSTEM_SUBSYS.SYSTEM_NUMBER ||
        INPUT_RECORD.SYSTEM_SUBSYS.SUBSYS_NUMBER ;
        IF SUBSTR(TEMP,1,1) = BLANK |
        SUBSTR(TEMP,2,1) = BLANK |
        (INPUT_RECORD.SYSTEM_SUBSYS.SYSTEM_NUMBER =
        BLANK &
        INPUT_RECORD.SYSTEM_SUBSYS.SUBSYS_NUMBER ^=
        BLANK) |
        (INPUT_RECORD.SYSTEM_SUBSYS.SUBSYS_NUMBER =
        BLANK &
        INPUT_RECORD.SYSTEM_SUBSYS.SYSTEM_NUMBER ^=
        BLANK) | THEN
        ERROR_FLAG = 1;
        I = TEMP;
        IF ERROR_FLAG ^= 0 THEN
          SYSTEM_NUMBER_ERROR:
          DO;
            NUMBER_OF_ERRORS = NUMBER_OF_ERRORS + 1;
            ERRORS(NUMBER_OF_ERRORS) = 2;
            END SYSTEM_NUMBER_ERROR;
            ERROR_FLAG = 0;
            IF INPUT_RECORD.RECORD_LENGTH ^= BLANK THEN
CONVERT_RECORD_LENGTH:
  DO;
    TEMP = INPUT_RECORD.RECORD_LENGTH;
    I = TEMP;
    END CONVERT_RECORD_LENGTH;
RECORD_LENGTH_ERROR:
  DO;
    NUMBER_OF_ERRORS = NUMBER_OF_ERRORS + 1;
    ERRORS(NUMBER_OF_ERRORS) = 3;
    END RECORD_LENGTH_ERROR;
    IF INPUT_RECORD.CODEDoodles = BLANK &
       INPUT_RECORD.CODEDoodles = 'Y' &
       INPUT_RECORD.CODEDoodles = 'N' THEN
    CODE_ERROR:
      DO;
        NUMBER_OF_ERRORS = NUMBER_OF_ERRORS + 1;
        ERRORS(NUMBER_OF_ERRORS) = 4;
        END CODE_ERROR;
        ERROR_FLAG = 0;
        IF INPUT_RECORD.AVERAGE_WRITE = BLANK THEN
          I = INPUT_RECORD.AVERAGE_WRITE;
          END AVERAGE_WRITE_ERROR;
          IF ERROR_FLAG = 0 &
            INPUT_RECORD.AVERAGE_WRITE = 0 THEN
            AVERAGE_WRITE_ERROR:
              DO;
                I = 10000 / I;
                PUT STRING (INPUT_RECORD.EDIT_WRITE)
                   EDIT (I)
                   (F(3)) ;
                END AVERAGE_WRITE_ERROR;
                ERROR_FLAG = 0;
                IF INPUT_RECORD.AVERAGE_VERIFY = BLANK THEN
                  I = INPUT_RECORD.AVERAGE_VERIFY;
                  END AVERAGE_VERIFY_ERROR;
                  IF ERROR_FLAG = 0 &
                    INPUT_RECORD.AVERAGE_VERIFY = 0 THEN
                    AVERAGE_VERIFY_ERROR:
                      DO;
                        I = 10000 / I;
                        PUT STRING (INPUT_RECORD.EDIT_VERIFY)
                           EDIT (I)
                           (F(3)) ;
                        END AVERAGE_VERIFY_ERROR;
                        LINES = LINES + 3 + NUMBER_OF_ERRORS;
                        IF LINES > MAXLINES THEN
                          CALL NEW_PAGE;
                          SEPARATOR = DASHES;
                        IF INPUT_RECORD.SUBSYS_NUMBER = BLANK THEN
                          SEPARATOR = BLANK;
                        IF FILE (SYSPRINT)
                          EDIT (INPUT_RECORD.SYSTEM_SUBSYS.DEPART_NUMBER,
                             INPUT_RECORD.SYSTEM_SUBSYS.SYSTEM_NUMBER,
                             SEPARATOR,
                             INPUT_RECORD.SYSTEM_SUBSYS.SUBSYS_NUMBER,
                             INPUT_RECORD_FORMAT,
INPUT_RECORD.TITLE,
INPUT_RECORD.RECORD_LENGTH,
INPUT_RECORD.AVERAGE_WRITE,
INPUT_RECORD.AVERAGE_VERIFY,
INPUT_RECORD.ESTIMATE_WRITE,
INPUT_RECORD.ESTIMATE_VERIFY,
INPUT_RECORD.MEDIA,
INPUT_RECORD.CORED,
INPUT_RECORD.TRANSACTION)
(SKIP(2),
COL(1), A, A, A, A,
COL(10), A,
COL(16), A,
COL(39), A,
COL(45), A, X(1), A,
COL(55), A, X(1), A,
COL(65), A, A,
COL(123), A);

WRITE_UPDATE: IF NUMBER_OF_ERRORS = 0 THEN
DO;
PUT FILE (SYSPRINT)
EDIT ('NO ERRORS FOUND')
(COL(10), A);
COUNT = COUNT + 1;
OUTPUT_RECORD = INPUT_RECORD, BY NAME;
OUTPUT_RECORD.AVERAGE_WRITE =
INPUT_RECORD.AVERAGE_WRITE;
OUTPUT_RECORD.AVERAGE_VERIFY =
INPUT_RECORD.AVERAGE_VERIFY;
OUTPUT_RECORD.ESTIMATE_WRITE =
INPUT_RECORD.ESTIMATE_WRITE;
OUTPUT_RECORD.ESTIMATE_VERIFY =
INPUT_RECORD.ESTIMATE_VERIFY;
OUTPUT_RECORD.FLAGS.EQUIPMENT =
INPUT_RECORD.MEDIA;
OUTPUT_RECORD.FLAGS.CORED = INPUT_RECORD.CORED;
WRITE FILE (UPDATE ) FROM (OUTPUT_RECORD);
END WRITE_UPDATE;
ELSE
PRINT_ERRORS: DO;
ERROR_COUNT = ERROR_COUNT + 1;
PUT FILE (SYSPRINT)
EDIT ('*** ERRORS FOUND ***')
(COL(1), A);
PRINT_EACH_ERROR:
DO I = 1 TO NUMBER_OF_ERRORS;
PUT FILE (SYSPRINT)
EDIT (ERROR_MESSAGES(ERRORS(I)))
(COL(10), A);
END PRINT_EACH_ERROR;
END PRINT_ERRORS;
END EDITS;
END INPUTS;
END EDITS;
END EDITS;
END INPUTS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
END EDITS;
LINES = NUMBER_OF_ERRORS + 3;
PUT FILE (SYSPRINT) PAGE
  EDIT ('CITY OF LOS ANGELES',
      'DATA SERVICE BUREAU',
      'DATE ', CURRENT_DATE,
      'DATA CONVERSION EDIT REPORT',
      'PAGE ', PAGE,
      DASHES,
      'SYSTEM',
      'EQUIP')
  (LINE(6), COL(70), A,
   COL(70), A,
   COL(10), A, A,
   COL(66), A,
   COL(110), A, F(4),
   SKIP(2), COL(1), A,
   COL(1), A,
   COL(65), A);

PUT FILE (SYSPRINT)
  EDIT ('SUB-SYS',
        'RECORD',
        'AVERAGE ESTIMATING',
        'CODED',
        'RECORDS',
        'RECORDS',
        'HOURS',
        'HOURS')
  (COL(1), A,
   COL(37), A,
   COL(45), A,
   COL(65), A,
   COL(74), A,
   COL(86), A,
   COL(99), A,
   COL(111), A);

PUT FILE (SYSPRINT)
  EDIT ('NUMBER',
        'FORMAT',
        'TITLE',
        'LENGTH',
        'WRT VER WRT VER',
        'DIFF',
        'WRITTEN',
        'VERIFIED',
        'VERIFIED',
        'TRANSACTION',
        DASHES)
  (COL(1), A,
   COL(9), A,
   COL(21), A,
   COL(37), A,
   COL(45), A,
   COL(65), A,
   COL(74), A,
   COL(85), A,
   COL(98), A,
   COL(109), A,
   COL(120), A,
   COL(1), A);

END NEW_PAGE;
END EDITPGM;
TBLUPDT: PROCEDURE OPTIONS (MAIN); 

THE UPDATE PROGRAM READS THE EDITED UPDATE RECORDS AND APPLIES THE UPDATE RECORDS TO THE MASTER FILE. A NEW MASTER FILE AND AN UPDATE REPORT IS PRODUCED. THE UPDATE REPORT CONTAINS THE ORIGINAL MASTER FILE RECORD, THE UPDATE RECORD, AND ANY ERRORS FOUND. FIELDS WHICH CONTAIN BLANKS ARE NOT CHANGED.

DCL 1 MASTER_FILE, 
  2 SYSTEM_SUBSYS, 
  3 DEPT_NUMBER CHAR (2), 
  3 SYSTEM_NUMBER CHAR (2), 
  3 SUBSYS_NUMBER CHAR (2), 
  2 FORMAT CHAR (4), 
  2 TITLE CHAR (20), 
  2 RECORD_LENGTH CHAR (3), 
  2 AVERAGE, 
  3 WRITE CHAR (3), 
  3 VERIFY CHAR (3), 
  2 ESTIMATE, 
  3 WRITE CHAR (3), 
  3 VERIFY CHAR (3), 
  2 FLAGS, 
  3 EQUIPMENT CHAR (1), 
  3 CODED CHAR (1), 
  3 DIFFICULTY CHAR (1), 
  3 SCHEDULE CHAR (2), 
  2 RECORDS, 
  3 WRITTEN CHAR (10), 
  3 VERIFIED CHAR (10), 
  3 CORRECTED CHAR (10), 
  2 HOURS, 
  3 WRITTEN CHAR (10), 
  3 VERIFIED CHAR (10); 

DCL 1 OUTPUT LIKE MASTER_FILE; 

DCL 1 UPDATE_FILE, 
  2 SYSTEM_SUBSYS, 
  3 DEPT_NUMBER CHAR (2), 
  3 SYSTEM_NUMBER CHAR (2), 
  3 SUBSYS_NUMBER CHAR (2), 
  2 FORMAT CHAR (4), 
  2 TITLE CHAR (20), 
  2 RECORD_LENGTH CHAR (3), 
  2 AVERAGE, 
  3 WRITE CHAR (3), 
  3 VERIFY CHAR (3), 
  2 ESTIMATE, 
  3 WRITE CHAR (3), 
  3 VERIFY CHAR (3), 
  2 FLAGS, 
  3 EQUIPMENT CHAR (1), 
  3 CODED CHAR (1), 
  3 DIFFICULTY CHAR (1), 
  3 SCHEDULE CHAR (2), 
  2 RECORDS, 
  3 WRITTEN CHAR (10), 
  3 VERIFIED CHAR (10), 
  3 CORRECTED CHAR (10), 
  2 HOURS, 
  3 WRITTEN CHAR (10), 
  3 VERIFIED CHAR (10), 
  2 TRANSACTION CHAR (6); 

DCL WRITE_FLAG FIXED BINARY, 
READ_FLAG1 FIXED BINARY INIT (1), 
READ_FLAG2 FIXED BINARY INIT (1),
LINES FIXED BINARY INIT (100),
MAXLINES FIXED BINARY INIT (40),
COUNT FIXED BINARY INIT (0),
ERROR_COUNT FIXED BINARY INIT (0),
RETURN_CODE FIXED BINARY (31,0) INIT (0),
ADD CHAR (6) INIT ('ADD'),
DELETE CHAR (6) INIT ('DELETE'),
MODIFY CHAR (6) INIT ('MODIFY'),
BLANK CHAR (1) INIT (' '),
CURRENT_DATE CHAR (8),
PAGE FIXED BINARY INIT (0),
TEMP1 CHAR (10),
TEMP2 CHAR (10),
EOF_SYSTEM_NUMBER CHAR (6) INIT ('""""""'),
*/ THE ABOVE INITIAL IZATION FIELD CONTAINS
HEXADECIMAL 'FF' CHARACTERS
*/
SEPARATOR CHAR (1),
DASHES CHAR (132) INIT ((132)'-');
OPEN FILE (MASTER ) RECORD INPUT,
FILE (NEUMSTR) RECORD OUTPUT,
FILE (UPDATE ) RECORD INPUT,
FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
FILE (SYSIN) INPUT RECORD ;
ON ENDFILE (MASTER ) BEGIN;
READ_FLAG1 = 2;
MASTER_FILE.SYSTEM_SUBSYS =
EOF_SYSTEM_NUMBER ;
END;
ON ENDFILE (UPDATE ) BEGIN;
READ_FLAG2 = 2;
UPDATE_FILE.SYSTEM_SUBSYS =
EOF_SYSTEM_NUMBER ;
END;
CURRENT_DATE = DATE;
CURRENT_DATE = SUBSTR(CURRENT_DATE,3,2) || '-';
SUBSTR(CURRENT_DATE,5,2) || '-';
SUBSTR(CURRENT_DATE,1,2) ;
UPDATE_PROCESSING:
DO WHILE (READ_FLAG1 < 2 | READ_FLAG2 < 2);
IF READ_FLAG1 = 1 THEN
READ FILE (MASTER) INTO (MASTER_FILE);
IF READ_FLAG2 = 1 THEN
READ FILE (UPDATE) INTO (UPDATE_FILE);
OUTPUT = '' ;
WRITE_FLAG = 1;
IF READ_FLAG1 = 2 & READ_FLAG2 = 2 THEN
WRITE_FLAG = 0;
TEMP1 = MASTER_FILE.SYSTEM_SUBSYS.DEPT_NUMBER ||
MASTER_FILE.SYSTEM_SUBSYS.SYSTEM_NUMBER ||
MASTER_FILE.SYSTEM_SUBSYS.SUBSYS_NUMBER ||
MASTER_FILE.FORMAT ;
TEMP2 = UPDATE_FILE.SYSTEM_SUBSYS.DEPT_NUMBER ||
UPDATE_FILE.SYSTEM_SUBSYS.SYSTEM_NUMBER ||
UPDATE_FILE.SYSTEM_SUBSYS.SUBSYS_NUMBER ||
UPDATE_FILE.FORMAT ;
IF WRITE_FLAG = 1 &
TEMP1 = TEMP2 THEN
PROCESS_EQUAL_CASE:
DO;
READ_FLAG1 = 1;
READ_FLAG2 = 1;
IF LINES > MAXLINES THEN
CALL NEW_PAGE;
SEPARATOR = DASHES ;
IF MASTER_FILE.SUBSYS_NUMBER = BLANK THEN
SEPARATOR = BLANK ;
PUT FILE (SYSPRINT)
EDIT (MASTER_FILE.DEPT_NUMBER,
MASTER FILE.SYSTEM_NUMBER,
SEPARATOR,
MASTER FILE.SUBSYS_NUMBER,
MASTER FILE.FORMAT,
MASTER FILE.TITLE,
MASTER FILE.RECORD_LENGTH,
MASTER FILE.AVERAGE.WRITE,
MASTER FILE.AVERAGE.VERIFY,
MASTER FILE.ESTIMATE.WRITE,
MASTER FILE.ESTIMATE.VERIFY,
MASTER FILE.FLAGS,
MASTER FILE.RECORDS.WRITTEN,
MASTER FILE.RECORDS.VERIFIED,
MASTER FILE.HOURS.WRITTEN,
MASTER FILE.HOURS.VERIFIED )
(SKIP(2),
COL(1), A, A, A, A,
COL(10), A, A,
COL(16), A, A,
COL(39), A, A,
COL(45), A, X(1), A,
COL(55), A, X(1), A,
COL(65), A, A, A, A,
COL(72), A, A,
COL(84), A, A,
COL(96), A, A,
COL(108), A ) ;
SEPARATOR = DASHES ;
IF UPDATE_FILE.SUBSYS_NUMBER = BLANK THEN
SEPARATOR = BLANK ;
PUT FILE (SYSPRINT)
EDIT (UPDATE_FILE.DEPT_NUMBER,
UPDATE_FILE.SYSTEM_NUMBER,
SEPARATOR,
UPDATE_FILE.SUBSYS_NUMBER,
UPDATE_FILE.FORMAT,
UPDATE_FILE.TITLE,
UPDATE_FILE.RECORD_LENGTH,
UPDATE_FILE.AVERAGE.WRITE,
UPDATE_FILE.AVERAGE.VERIFY,
UPDATE_FILE.ESTIMATE.WRITE,
UPDATE_FILE.ESTIMATE.VERIFY,
UPDATE_FILE.FLAGS,
UPDATE_FILE.RECORDS.WRITTEN,
UPDATE_FILE.RECORDS.VERIFIED,
UPDATE_FILE.HOURS.WRITTEN,
UPDATE_FILE.HOURS.VERIFIED,
UPDATE_FILE.TRANSACTION )
(COL(1), A, A, A, A,
COL(10), A, A,
COL(16), A, A,
COL(39), A, A,
COL(45), A, X(1), A,
COL(55), A, X(1), A,
COL(65), A, A, A, A,
COL(72), A, A,
COL(84), A, A,
COL(96), A, A,
COL(108), A, A,
COL(123), A ) ;
LINES = LINES + 3;
IF UPDATE_FILE.TRANSACTION = ADD THEN
DO:
ADD_ERROR:
PUT FILE (SYSPRINT)
EDIT ('*** ERROR *** ATTEMPT TO ',
'ADD AN EXISTING TABLE ENTRY.')
(COL(10), A, A);
LINES = LINES + 1;
MODIFICATIONS:

ERROR_COUNT = ERROR_COUNT + 1;
OUTPUT = MASTER_FILE;
END ADD_ERROR;
IF UPDATE_FILE.TRANSACTION = DELETE THEN
  WRITE_FLAG = 0;
IF UPDATE_FILE.TRANSACTION = MODIFY THEN
  DO;
    OUTPUT = MASTER_FILE;
    IF UPDATE_FILE.FORMAT = BLANK THEN
      OUTPUT.FORMAT = UPDATE_FILE.FORMAT;
    IF UPDATE_FILE.TITLE = BLANK THEN
      OUTPUT.TITLE = UPDATE_FILE.TITLE;
    IF UPDATE_FILE.RECORD_LENGTH = BLANK THEN
      UPDATE_FILE.RECORD_LENGTH = OUTPUT.RECORD_LENGTH;
    IF UPDATE_FILE.AVERAGE.WRITE = BLANK THEN
      UPDATE_FILE.AVERAGE.WRITE = OUTPUT.AVERAGE.WRITE;
    IF UPDATE_FILE.AVERAGE.VERIFY = BLANK THEN
      UPDATE_FILE.AVERAGE.VERIFY = OUTPUT.AVERAGE.VERIFY;
    IF UPDATE_FILE.ESTIMATE.WRITE = BLANK THEN
      UPDATE_FILE.ESTIMATE.WRITE = OUTPUT.ESTIMATE.WRITE;
    IF UPDATE_FILE.ESTIMATE.VERIFY = BLANK THEN
      UPDATE_FILE.ESTIMATE.VERIFY = OUTPUT.ESTIMATE.VERIFY;
    IF UPDATE_FILE.FLAGS.EQUIPMENT = BLANK THEN
      UPDATE_FILE.FLAGS.EQUIPMENT = OUTPUT.FLAGS.EQUIPMENT;
    IF UPDATE_FILE.FLAGS.CODED = BLANK THEN
      UPDATE_FILE.FLAGS.CODED = OUTPUT.FLAGS.CODED;
    IF UPDATE_FILE.FLAGS.DIFFICULTY = BLANK THEN
      UPDATE_FILE.FLAGS.DIFFICULTY = OUTPUT.FLAGS.DIFFICULTY;
    IF UPDATE_FILE.RECORDS.WRITTEN = BLANK THEN
      UPDATE_FILE.RECORDS.WRITTEN = OUTPUT.RECORDS.WRITTEN;
    IF UPDATE_FILE.RECORDS.VERIFIED = BLANK THEN
      UPDATE_FILE.RECORDS.VERIFIED = OUTPUT.RECORDS.VERIFIED;
    IF UPDATE_FILE.HOURS.WRITTEN = BLANK THEN
      UPDATE_FILE.HOURS.WRITTEN = OUTPUT.HOURS.WRITTEN;
    IF UPDATE_FILE.HOURS.VERIFIED = BLANK THEN
      UPDATE_FILE.HOURS.VERIFIED = OUTPUT.HOURS.VERIFIED;
    COUNT = COUNT + 1;
  END MODIFICATIONS;
END PROCESS_EQUAL_CASE;

IF TEMP1 = TEMP2 THEN
  END

MASTER_FILE_ONLY:
  DO;
    OUTPUT = MASTER_FILE;
    IF READ_FLAG1 = 0 THEN READ_FLAG1 = 1;
    IF READ_FLAG2 = 1 THEN READ_FLAG2 = 0;
    COUNT = COUNT + 1;
  END MASTER_FILE_ONLY;

ADD_AN_ENTRY:
  DO;
    IF READ_FLAG1 = 1 THEN READ_FLAG1 = 0;
    IF READ_FLAG2 = 0 THEN READ_FLAG2 = 1;
    IF LINES > MAXLINES THEN
      CALL NEW_PAGE;
      SEPARATOR = DASHES;
    END IF;
END IF;

IF UPDATE_FILE.SUBSYS_NUMBER = BLANK THEN
  SEPAREATOR = BLANK;
  PUT FILE (SYSPRNT)
    EDIT (UPDATE_FILE.DEPT_NUMBER,
        UPDATE_FILE.SYTEM_NUMBER,
        SEPARATOR,
        UPDATE_FILE.SUBSYS_NUMBER,
        UPDATE_FILE.FORMAT,
        UPDATE_FILE.TITLE,
        UPDATE_FILE.RECORD_LENGTH,
        UPDATE_FILE.AVERAGE.WRITE,
        UPDATE_FILE.AVERAGE.VERIFY,
        UPDATE_FILE.ESTIMATE.WRITE,
        UPDATE_FILE.ESTIMATE.VERIFY,
        UPDATE_FILE.FLAGS,
        UPDATE_FILE.RECORDS.WRITTEN,
        UPDATE_FILE.RECORDS.VERIFIED,
        UPDATE_FILE.HOURS.WRITTEN,
        UPDATE_FILE.HOURS.VERIFIED,
        UPDATE_FILE.TRANSACTION)
  (SKIP(2),
    COL(1), A, A, A, A,
    COL(10), A,
    COL(16), A,
    COL(39), A,
    COL(45), A, X(1), A,
    COL(55), A, X(1), A,
    COL(65), A, A, A, A,
    COL(72), A,
    COL(84), A,
    COL(96), A,
    COL(108), A,
    COL(123), A );
LINES = LINES + 2;
IF UPDATE_FILE.TRANSACTION = ADD THEN
  COPY_UPDATE_FILE: DO;
    OUTPUT = UPDATE_FILE, BY NAME ;
    COUNT = COUNT + 1;
    END COPY_UPDATE_FILE;
  ELSE
    MODIFY_OR_DELETE_ERROR:
      DO;
        WRITE_FLAG = 0;
        PUT FILE (SYSPRNT)
          EDIT ("*** ERROR *** ATTEMPT TO 'UPDATE_FILE.TRANSACTION, 'A NON-EXISTENT ENTRY.'")
          (COL(10), A, A, A );
        LINES = LINES + 1;
        ERROR_COUNT = ERROR_COUNT + 1 ;
        END MODIFY_OR_DELETE_ERROR;
      END ADD_AN_ENTRY;
      IF WRITE_FLAG = 0 THEN
        WRITE FILE (NEWSTR ) FROM (OUTPUT) ;
      END UPDATE_PROCESSING;
      IF ERROR_COUNT = 0 THEN
        RETURN_CODE = 4;
      IF COUNT = 0 THEN
        RETURN_CODE = 8;
      CALL IONESARCCRETURN_CODE);
NEW_PAGE: PROCEDURE ;

*/
/*
THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON EACH NEW PAGE FOR THE UPDATE REPORT.*/
/*
PAGE = PAGE + 1;
LINES = 0;
PUT FILE (SYSPRINT) PAGE
EDIT ('CITY OF LOS ANGELES',
'DATA SERVICE BUREAU',
'DATE ', CURRENT_DATE,
'DATA CONVERSION UPDATE REPORT',
'PAGE ', PAGE,
DASHES,
'SYSTEM',
'EQUIP')
(LINE(6), COL(70), A,
COL(70), A,
COL(10), A, A,
COL(65), A,
COL(110), A, F(4),
SKIP(2), COL(1), A,
COL(1), A,
COL(65), A);

PUT FILE (SYSPRINT)
EDIT ('SUB-SYS',
'RECORD',
'AVERAGE ESTIMATING',
'CODED',
'RECORDS',
'RECORDS',
'HOURS',
'HOURS')
(COL(1), A,
COL(37), A,
COL(45), A,
COL(65), A,
COL(74), A,
COL(66), A,
COL(99), A,
COL(111), A);

PUT FILE (SYSPRINT)
EDIT ('NUMBER',
'FORMAT',
'TITLE',
'LENGTH',
'WRT VER',
'WRT VER',
'DIFF.',
'WRITTEN',
'WRITTEN',
'WRITTEN',
'WRITTEN',
'WRITTEN',
'WRITTEN',
'TRANSACTION',
DASHES)
(COL(1), A,
COL(9), A,
COL(21), A,
COL(37), A,
COL(45), A,
COL(65), A,
COL(74), A,
COL(85), A,
COL(98), A,
COL(109), A,
COL(120), A,
COL(1), A);

END NEW_PAGE;
END TBLUPDT;
PERIOD: PROCEDURE OPTIONS (MAIN);

THE PERIOD PROGRAM READS THE SORTED DATA RECORDS AND
PRODUCES A PERIOD REPORT, AND A YEAR-TO-DATE MASTER FILE.
THE PERIOD REPORT CONTAINS THE PERIOD DATA ENTRY WORKLOAD
FOR EACH SYSTEM AND FORMAT. TOTALS FOR EACH CITY AGENCY
ARE PRODUCED AT THE END OF EACH AGENCY'S PERIOD ACTIVITY.
IN ADDITION, A SUMMARY REPORT IS PRODUCED WHICH SHOWS TOTALS
FOR EACH CITY AGENCY, TOTALS FOR CARDS AND MINICOMPUTER
WORKLOAD, AND TOTALS FOR ALL DATA CONVERSION. THE
YEAR-TO-DATE MASTER FILE CONTAINS YEAR-TO-DATE TOTALS FOR
EACH SYSTEM AND FORMAT.

DCL 1 RECORD,
  2 FORMAT CHAR (1),
  2 NAME_FLAG CHAR (1),
  2 ACCOUNT_FIELD CHAR (10),
  2 JOB CHAR (10),
  2 BATCH CHAR (4),
  2 STATUS CHAR (1),
  2 KEYPAD CHAR (4),
  2 RECORD_FORMAT CHAR (4),
  2 OPERATOR CHAR (3),
  2 COUNT CHAR (4),
  2 MINUTES CHAR (4),
  2 KEYSTROKES CHAR (8),
  2 CORRECTIONS CHAR (4),
  2 OPEN_TIME CHAR (11),
  2 CLOSE_TIME CHAR (11),
  2 FILLER CHAR (20);

DCL 1 OUTPUT,
  2 SYSTEM_SUBSYS,
    3 DEPT_NUMBER CHAR (2),
    3 SYSTEM_NUMBER CHAR (2),
    3 SUBSYS_NUMBER CHAR (2),
  2 FORMAT CHAR (4),
  2 TITLE CHAR (20),
  2 RECORD_LENGTH CHAR (3),
  2 AVERAGE,
    3 WRITE CHAR (3),
    3 VERIFY CHAR (3),
  2 ESTIMATE,
    3 WRITE CHAR (3),
    3 VERIFY CHAR (3),
  2 FLAGS,
    3 EQUIPMENT CHAR (1),
    3 CODED CHAR (1),
    3 DIFFICULTY CHAR (1),
    3 SCHEDULE CHAR (2),
  2 RECORDS,
    3 WRITTEN CHAR (10),
    3 VERIFIED CHAR (10),
    3 CORRECTED CHAR (10),
  2 HOURS,
    3 WRITTEN CHAR (10),
    3 VERIFIED CHAR (10);
EFFICIENCY (3) FLOAT DECIMAL,
MAXLINES FIXED BINARY INIT (40),
LINES FIXED BINARY INIT (0),
PAGE FIXED BINARY INIT (0),
CURRENT_DEPT FIXED BINARY INIT (0),
DEPT_NAME CHAR (20) INIT (''),
OLD_DEPT_NAME CHAR (20) INIT (''),
DASHES CHAR (132) INIT ('132'),
HEADER CHAR (2) INIT ('00'),
FISCAL_YEAR CHAR (10),
PERIOD FIXED BINARY,
RETURN_CODE FIXED BINARY (31,0) INIT (0),
ACCOUNT CHAR (10) DEF ACCOUNT FIELD POSITION (1),
CARD_HOURS FLOAT DECIMAL INIT (126.656),
KTD_HOURS FLOAT DECIMAL INIT (87.2),
CARD (5) FLOAT DECIMAL (10) INIT (0),
KTD (5) FLOAT DECIMAL (10) INIT (0),
GRAND_TOTAL (5) FLOAT DECIMAL (10) INIT (0),
OLD_ACCOUNT CHAR (10) INIT (''),
SYSTEM CHAR(4) DEF OLD_ACCOUNT POSITION (1),
SUB_SYSTEM CHAR (2) DEF OLD_ACCOUNT POSITION (5),
LINE_TOTALS(5) FLOAT DECIMAL (10) INIT (0),
TOTAL_KEYSTROKES FIXED BINARY (31,0) INIT (0),
1 DEPT(100),
2 PAGE FIXED BINARY,
2 NAME CHAR (20),
3 WRITTEN FIXED BINARY (31,0),
3 VERIFIED FIXED BINARY (31,0),
3 CORRECTED FIXED BINARY (31,0),
2 HOURS,
3 WRITTEN FLOAT DECIMAL (10),
3 VERIFIED FLOAT DECIMAL (10),
2 PERSONNEL FLOAT DECIMAL (10);
OPEN FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
FILE (PERDATA ) INPUT RECORD,
FILE (MASTER ) INPUT RECORD,
FILE (NEWMSTR ) OUTPUT RECORD ;
ON ENDFILE (PERDATA ) BEGIN;
   END_OF_FILE = 1 ;
   ACCOUNT = EOF_SYSTEM_NUMBER ;
END;
ON ENDFILE (MASTER ) INPUT.SYSTEM_SUBSYS =
   EOF_SYSTEM_NUMBER ;
ON CONVERSION ONCHAR = '0' ;
READ FILE (MASTER ) INTO (INPUT ) ;
IF INPUT.DEPT_NUMBER = HEADER THEN;
   PERIOD = INPUT.SCHEDULE + 1 ;
   PUT STRING (INPUT.SCHEDULE) EDIT (PERIOD) (F(2)) ;
   FISCAL_YEAR = INPUT.TITLE ;
   OUTPUT = INPUT ;
   WRITE FILE (NEWMSTR) FROM (OUTPUT) ;
   OLD_ACCOUNT = HEADER ;
   END ;
   ELSE
   DO;
   PUT FILE (SYSPRINT)
      EDIT ('HEADER RECORD IS MISSING, ','EXECUTION TERMINATED.')
      (A, A) ;
   END;
   END;
DO_IT: DO;
   WHILE (END_OF_FILE = 0);
   READ FILE (PERDATA ) INTO (RECORD);
   IF END_OF_FILE = 1 |
      OLD_ACCOUNT = ACCOUNT THEN
   NEW_ACCOUNT: DO;
IF CURRENT_DEPT ~/= 0 THEN
DO;
I = CURRENT_DEPT;
DEPT(I).RECORDS.WRITTEN =
DEPT(I).RECORDS.WRITTEN + LINE_TOTALS(1);
DEPT(I).RECORDS.VERIFIED =
DEPT(I).RECORDS.VERIFIED + LINE_TOTALS(2);
DEPT(I).RECORDS.CORRECTED =
DEPT(I).RECORDS.CORRECTED + LINE_TOTALS(3);
DEPT(I).HOURS.WRITTEN = DEPT(I).HOURS.WRITTEN +
LINE_TOTALS(4);
DEPT(I).HOURS.VERIFIED =
DEPT(I).HOURS.VERIFIED + LINE_TOTALS(5);
TEMP1 = (LINE_TOTALS(4) + LINE_TOTALS(5)) / 60.0;
LINE_TOTALS(4) = LINE_TOTALS(4) / 60.0;
LINE_TOTALS(5) = LINE_TOTALS(5) / 60.0;
IF OUTPUT.EQUIPMENT = 'C' THEN
TEMP2 = TEMP1 / CARD_HOURS;
ELSE
TEMP2 = TEMP1 / KTD_HOURS;
DEPT(I).PERSONNEL = DEPT(I).PERSONNEL + TEMP2;
EFFICIENCY = 0;
IF LINE_TOTALS(4) ~/= 0 &
OUTPUT.AVERAGE.WRITE ~/= 0 THEN
EFFICIENCY(1) = LINE_TOTALS(1) /
LINE_TOTALS(4) / OUTPUT.AVERAGE.WRITE;
IF LINE_TOTALS(5) ~/= 0 &
OUTPUT.AVERAGE.VERIFY ~/= 0 THEN
EFFICIENCY(2) = LINE_TOTALS(2) /
LINE_TOTALS(5) / OUTPUT.AVERAGE.VERIFY;
EFFICIENCY(3) = EFFICIENCY(1);
IF EFFICIENCY(1) = 0 THEN
EFFICIENCY(3) = EFFICIENCY(2);
IF EFFICIENCY(1) ~/= 0 & EFFICIENCY(2) ~/= 0 THEN
EFFICIENCY(3) = (EFFICIENCY(1) * LINE_TOTALS(4) +
EFFICIENCY(2) *
LINE_TOTALS(5)) / (LINE_TOTALS(4) + LINE_TOTALS(5));
EFFICIENCY(1) = EFFICIENCY(1) * 100;
EFFICIENCY(2) = EFFICIENCY(2) * 100;
EFFICIENCY(3) = EFFICIENCY(3) * 100;
LINES = LINES + 1;
IF LINES > MAXLINES THEN
CALL NEW_PAGE;
IF OLD_ACCOUNT ~/= HEADER THEN
PUT FILE (SYSPRINT)
EDIT (SYSTEM, '-'), SUB_SYSTEM,
OUTPUT_FORMAT,
OUTPUT_TITLE,
LINE_TOTALS,
TEMP1,
TEMP2,
OUTPUT.RECORD_LENGTH,
OUTPUT.AVERAGE_WRITE,
OUTPUT.AVERAGE_VERIFY,
OUTPUT_ESTIMATE_WRITE,
OUTPUT_ESTIMATE_VERIFY,
EFFICIENCY)
(COL(1), A,
COL(10), A,
COL(17), A,
COL(38), F(8), COL(48), F(8),
COL(57), F(6),
COL(65), F(7,1), COL(74), F(7,1),

OUTPUT_LINE:
IF \(\text{LINE\_TOTALS}(1) \neq 0\)  
\(\text{LINE\_TOTALS}(2) \neq 0\) THEN  
\(I = \frac{\text{TOTAL\_KEYSTROKES}}{\text{LINE\_TOTALS}(1)} + \frac{\text{LINE\_TOTALS}(2)}{\text{LINE\_TOTALS}(2)}\);  
PUT STRING (\text{OUTPUT\_RECORD\_LENGTH})  
EDIT ([I])  
\(\text{(F}(3))\);  
I = \text{OUTPUT\_RECORD\_WRITTEN} + \text{LINE\_TOTALS}(1);  
PUT STRING (\text{OUTPUT\_RECORD\_WRITTEN})  
EDIT ([I])  
\(\text{(F}(3))\);  
\(\text{TEMP}1 = \text{OUTPUT\_HOURS\_WRITTEN} + \text{LINE\_TOTALS}(4);\)  
PUT STRING (\text{OUTPUT\_HOURS\_WRITTEN})  
EDIT ([\text{TEMP}1])  
\(\text{(F}(10,1))\);  
I = \text{OUTPUT\_RECORD\_VERIFIED} + \text{LINE\_TOTALS}(2);  
PUT STRING (\text{OUTPUT\_RECORD\_VERIFIED})  
EDIT ([I])  
\(\text{(F}(10))\);  
\(\text{TEMP}1 = \text{OUTPUT\_HOURS\_VERIFIED} + \text{LINE\_TOTALS}(5);\)  
PUT STRING (\text{OUTPUT\_HOURS\_VERIFIED})  
EDIT ([\text{TEMP}1])  
\(\text{(F}(10,1))\);  
I = \text{OUTPUT\_RECORD\_CORRECTED} + \text{LINE\_TOTALS}(3);  
PUT STRING (\text{OUTPUT\_RECORD\_CORRECTED})  
EDIT ([I])  
\(\text{(F}(10))\);  
\(\text{CALCULATE\_ESTIMATES: DO ;}\)  
IF \(\text{OUTPUT\_FORMAT} = \text{BLANK}\) THEN  
\(\text{CALCULATE\_WRITE\_ESTIMATE: DO ;}\)  
\(\text{TEMP}1 = \text{OUTPUT\_AVERAGE\_WRITE};\)  
\(\text{TEMP}1 = 10000 / \text{TEMP}1;\)  
PUT STRING (\text{OUTPUT\_ESTIMATE\_WRITE})  
EDIT ([\text{TEMP}1])  
\(\text{(F}(3))\);  
END \text{CALCULATE\_WRITE\_ESTIMATE};  
IF \(\text{OUTPUT\_AVERAGE\_VERIFY} = 0\) THEN  
\(\text{CALCULATE\_VERIFY\_ESTIMATE: DO ;}\)  
\(\text{TEMP}1 = \text{OUTPUT\_AVERAGE\_VERIFY};\)  
\(\text{TEMP}1 = 10000 / \text{TEMP}1;\)  
PUT STRING (\text{OUTPUT\_ESTIMATE\_VERIFY})  
EDIT ([\text{TEMP}1])  
\(\text{(F}(3))\);  
END \text{CALCULATE\_VERIFY\_ESTIMATE};  
END \text{CALCULATE\_ESTIMATES};  
\text{WRITE FILE (NEW_MSTR) FROM (OUTPUT)};  
\text{END OUTPUT\_LINE};  
\text{SEARCH\_FOR\_NEW\_ACCOUNT: DO WHILE (ACCOUNT > (\text{INPUT\_DEPT\_NUMBER} || \text{INPUT\_SYSTEM\_NUMBER} || \text{INPUT\_SUBSYS\_NUMBER} || \text{INPUT\_FORMAT})) ;}\)  
IF \(\text{OLD\_ACCOUNT} = \text{HEADER} \& \text{OLD\_ACCOUNT} < (\text{INPUT\_DEPT\_NUMBER} || \text{INPUT\_SYSTEM\_NUMBER} || \text{INPUT\_SUBSYS\_NUMBER} || \text{INPUT\_FORMAT})\) THEN  
DO;
IF INPUT.SYSTEM_NUMBER = BLANK &
INPUT.SUBSYS_NUMBER = BLANK THEN
DEPT_NAME = INPUT.TITLE;
OUTPUT = INPUT;
WRITE FILE (NEWMSTR ) FROM (OUTPUT); END;
READ FILE (MASTER ) INTO (INPUT);
IF ACCOUNT > (INPUT.DEPT_NUMBER ||
INPUT.SYSTEM_NUMBER ||
INPUT.SUBSYS_NUMBER ||
INPUT.FORMAT) THEN
DO;
IF INPUT.SYSTEM_NUMBER = BLANK &
INPUT.SUBSYS_NUMBER = BLANK THEN
DEPT_NAME = INPUT.TITLE;
OUTPUT = INPUT;
WRITE FILE (NEWMSTR ) FROM (OUTPUT);
END/account = ACCOUNT;
END;
END SEARCH FOR NEW ACCOUNT;
IF ACCOUNT = (INPUT.DEPT_NUMBER ||
INPUT.SYSTEM_NUMBER ||
INPUT.SUBSYS_NUMBER ||
INPUT.FORMAT) THEN
OUTPUT = INPUT;
IF ACCOUNT < (INPUT.DEPT_NUMBER ||
INPUT.SYSTEM_NUMBER ||
INPUT.SUBSYS_NUMBER ||
INPUT.FORMAT) THEN
CREATE_NEW_ACCOUNT:
DO;
OUTPUT = ''; 
OUTPUT.DEPT_NUMBER = SUBSTR(ACCOUNT,1,2) ;
OUTPUT.SYSTEM_NUMBER = SUBSTR(ACCOUNT,3,2) ;
OUTPUT.SUBSYS_NUMBER = SUBSTR(ACCOUNT,5,2) ;
OUTPUT.FORMAT = SUBSTR(ACCOUNT,7,4) ;
END CREATE_NEW_ACCOUNT;
OLD_ACCOUNT = ACCOUNT;
LINE_TOTALS = 0;
TOTAL_KEYSTROKES = 'A' THEN
OUTPUT_EQUIPMENT = 'T' ;
IF RECORD FORMAT = '1' &
OUTPUT_EQUIPMENT = BLANK THEN
OUTPUT_EQUIPMENT = 'C' ;
END NEW ACCOUNT;
ADD_LINE: IF END_OF_FILE = 0 & OLD_ACCOUNT = ACCOUNT THEN
DO;
TOTAL_KEYSTROKES = TOTAL_KEYSTROKES + KEYSTROKES ;
IF STATUS = 'A' THEN
ADD_WRITE_TOTALS: DO;
LINE_TOTALS(1) = LINE_TOTALS(1) + COUNT;
LINE_TOTALS(4) = LINE_TOTALS(4) + MINUTES;
IF OUTPUT_EQUIPMENT = 'C' THEN
ADD_CARD_WRITE: DO;
CARD(1) = CARD(1) + COUNT ;
CARD(4) = CARD(4) + MINUTES ;
END ADD_CARD_WRITE;
ELSE
ADD_KTD_WRITE: DO ;
KTD(1) = KTD(1) + COUNT ;
KTD(4) = KTD(4) + MINUTES ;
END ADD_KTD_WRITE;
END ADD_WRITE_TOTALS;
ELSE
ADD_VERIFY_TOTALS: DO;
LINE_TOTALS(2) = LINE_TOTALS(2) + COUNT;
LINE_TOTALS(3) = LINE_TOTALS(3) + CORRECTIONS;
LINE_TOTALS(5) = LINE_TOTALS(5) + MINUTES;
IF OUTPUT_EQUIPMENT = 'C' THEN
  ADD_CARD_VERIFY:
    DO;
      CARD(2) = CARD(2) + COUNT;
      CARD(5) = CARD(5) + MINUTES;
      CARD(3) = CARD(3) + CORRECTIONS;
    END ADD_CARD_VERIFY;
  ELSE
    END ADD_VERIFY_TOTALS;
  END ADD_LINE;
IF END_OF_FILE = 1 OR OLD_DEPT_NAME = DEPT_NAME THEN
  NEW_DEPT:
    DO;
      IF CURRENT_DEPT = 1 THEN
        PUT_TOTALS;
      END IF;
      I = CURRENT_DEPT;
      GRAND_TOTAL(1) = GRAND_TOTAL(1) + DEPT(I).RECORDS.WRITTEN;
      GRAND_TOTAL(2) = GRAND_TOTAL(2) + DEPT(I).RECORDS.VERIFIED;
      GRAND_TOTAL(3) = GRAND_TOTAL(3) + DEPT(I).RECORDS.CORRECTED;
      GRAND_TOTAL(4) = GRAND_TOTAL(4) + DEPT(I).HOURS.WRITTEN;
      GRAND_TOTAL(5) = GRAND_TOTAL(5) + DEPT(I).HOURS.VERIFIED;
      DEPT(I).HOURS.WRITTEN = DEPT(I).HOURS.WRITTEN / 60.0;
      DEPT(I).HOURS.VERIFIED = DEPT(I).HOURS.VERIFIED / 60.0;
      TEMPI = DEPT(I).HOURS.WRITTEN + DEPT(I).HOURS.VERIFIED;
      PUT FILE (SYSPRINT) EDIT (DASHES, 'DEPARTMENT PERIOD TOTALS',
          DEPT(I).RECORDS.WRITTEN,
          DEPT(I).RECORDS.VERIFIED,
          DEPT(I).RECORDS.CORRECTED,
          DEPT(I).HOURS.WRITTEN,
          DEPT(I).HOURS.VERIFIED,
          TEMPI,
          DEPT(I).PERSONNEL)
          (COL(1), A, COL(1), A,
           COL(38), F(8), COL(48), F(8),
           COL(57), F(6),
           COL(65), F(7,1), COL(74), F(7,1),
           COL(82), F(8,1), COL(91), F(5,1)) ;
    END PUT_TOTALS;
IF END_OF_FILE = 0 THEN
  INITIALIZE_NEW_DEPT:
    DO;
      CURRENT_DEPT = CURRENT_DEPT + 1;
      I = CURRENT_DEPT;
      DEPT(I).NAME = DEPT_NAME;
      OLD_DEPT_NAME = DEPT_NAME;
      CALL NEW_PAGE;
      DEPT(I).PAGE = PAGE;
      DEPT(I).RECORDS.WRITTEN = 0;
      DEPT(I).RECORDS.VERIFIED = 0;
      DEPT(I).RECORDS.CORRECTED = 0;
      DEPT(I).HOURS.WRITTEN = 0;
      DEPT(I).HOURS.VERIFIED = 0;
      DEPT(I).PERSONNEL = 0;
END INITIALIZE_NEW_DEPT;
END NEW_DEPT;
END DO_IT;
CALL SUMMARY_PAGE;
CALL SORT;
PRINT_INDEX:
DO I = 1 TO CURRENT_DEPT BY 1;
  LINES = LINES + 1;
  IF LINES >= MAXLINES THEN
    CALL SUMMARY_PAGE;
  END IF;
  TEMP1 = DEPT(I).HOURS.WRITTEN + DEPT(I).HOURS.VERIFIED;
  PUT FILE (SYSPRINT)
    EDIT(DEPT(I).NAME,
      DEPT(I).PAGE,
      DEPT(I).RECORDS.WRITTEN,
      DEPT(I).RECORDS.VERIFIED,
      DEPT(I).RECORDS.CORRECTED,
      DEPT(I).HOURS.WRITTEN,
      DEPT(I).HOURS.VERIFIED,
      TEMP1,
      DEPT(I).PERSONNEL)
      (COL(7), A,
      COL(30), F(4),
      COL(37), F(10),
      COL(50), F(10),
      COL(63), F(10),
      COL(77), F(10.1),
      COL(90), F(10.1),
      COL(103), F(10.1),
      COL(116), F(10.1));
END PRINT_INDEX;
TEMP1 = (CARD(4) + CARD(5)) / 60.0;
CARD(4) = CARD(4) / 60.0;
CARD(5) = CARD(5) / 60.0;
TEMP2 = TEMP1 / CARD_HOURS;
TEMP3 = TEMP2;
PUT FILE (SYSPRINT)
  EDIT (DASHES,
    'CARD TOTALS FOR PERIOD', PERIOD,
    CARD,
    TEMP1,
    TEMP2)
    (COL(1), A,
    SKIP(2), COL(1), A, F(3),
    COL(37), F(10),
    COL(50), F(10),
    COL(63), F(10),
    COL(77), F(10.1),
    COL(90), F(10.1),
    COL(103), F(10.1),
    COL(116), F(10.1));
TEMP1 = (KTD(4) + KTD(5)) / 60.0;
KTD(4) = KTD(4) / 60.0;
KTD(5) = KTD(5) / 60.0;
TEMP2 = TEMP1 / KTD_HOURS;
TEMP3 = TEMP3 + TEMP2;
PUT FILE (SYSPRINT)
  EDIT ('KEY-TO-DISK TOTALS FOR PERIOD', PERIOD,
    KTD,
    TEMP1,
    TEMP2)
    (COL(1), A, F(3),
    COL(37), F(10),
    COL(50), F(10),
    COL(63), F(10),
    COL(77), F(10.1),
    COL(90), F(10.1),
    COL(103), F(10.1),
    COL(116), F(10.1));
TEMP1 = GRAND_TOTAL(4) + GRAND_TOTAL(5) / 60.0;
GRAND_TOTAL(4) = GRAND_TOTAL(4) / 60.0;
GRAND_TOTAL(5) = GRAND_TOTAL(5) / 60.0;

PUT FILE (SYSPRINT)
EDIT (DASHES,
'TOTALS FOR PERIOD', PERIOD,
GRAND_TOTAL,
TEMP1,
TEMP3)
(COL(1), A,
SKIP(2), COL(1), A, F(3),
COL(37), F(10),
COL(50), F(10),
COL(63), F(10),
COL(77), F(10.1),
COL(90), F(10.1),
COL(103), F(10.1),
COL(116), F(10.1)) ;

IF GRAND_TOTAL(1) = 0 & GRAND_TOTAL(4) = 0 THEN
RETURN_CODE = 8 ;
CALL IRESARC(RETURN_CODE) ;

NEW_PAGE: PROCEDURE;

/-------------------------------------------------------------------/
/ THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON /
/ EACH NEW PAGE OF THE PERIOD REPORT. /
/-------------------------------------------------------------------/

LINES = 0;
PAGE = PAGE + 1;
PUT FILE (SYSPRINT) PAGE
EDIT ('CITY OF LOS ANGELES',
'DATA SERVICE BUREAU',
'PERIOD', PERIOD,
DEPT_NAME,
'DATA CONVERSION PERIOD REPORT',
FISCAL_YEAR,
'PAGE', PAGE )
(LINE(3), COL(56), A,
COL(56), A,
COL(6), A, F(3), X(5), A,
COL(51), A,
COL(92), A,
COL(110), A, F(3)) ;

PUT FILE (SYSPRINT)
EDIT (DASHES,
'SYSTEM',
'SYSTEM',
'COR-', 'PERS', 'AVG',
'EFFICIENCY')
(SKIP(2), COL(1), A,
COL(1), A, COL(58), A,
COL(92), A, COL(97), A,
COL(122), A) ;

PUT FILE (SYSPRINT)
EDIT ('SUB-SYS', 'RECORDS', 'RECORDS',
'REC-',
'HOURS', 'HOURS', 'TOTAL', 'SON-',
'REC',
'AVERAGE ESTIMATING',
'RATIOS')
(COL(1), A, COL(39), A, COL(49), A,
COL(58), A,
COL(66), A, COL(75), A, COL(85), A,
COL(92), A, COL(97), A, COL(102), A,
COL(124), A) ;
PUT FILE (SYSPRINT)
EDIT ('NUMBER', 'FORMAT', 'JOB NAME',
 'WRITTEN', 'VERIFIED',
 'TIONS',
 'WRITTEN', 'VERIFIED',
 'HOURS', 'NEL', 'LEN',
 'WRT VER WRT VER',
 'WRT VER ALL',
 DASHES, '')
(COL(1), A, COL(9), A, COL(23), A,
 COL(39), A, COL(48), A,
 COL(58), A, COL(65), A, COL(73), A,
 COL(85), A, COL(92), A, COL(97), A,
 COL(102), A, COL(122), A,
 COL(11), A, COL(1), A);
END NEW_PAGE;

SUMMARY_PAGE:
PROCEDURE;
/
/ THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON /
/ EACH NEW SUMMARY PAGE. /
/
LINES = 0;
PAGE = PAGE + 1;
PUT FILE (SYSPRINT) PAGE
EDIT ('CITY OF LOS ANGELES',
 'DATA SERVICE BUREAU',
 'PERIOD', PERIOD,
 'INDEX TO DEPARTMENTS',
 'DATA CONVERSION PERIOD REPORT',
 FISCAL_YEAR,
 'PAGE', PAGE)
(LINE(3), COL(56), A,
 COL(56), A,
 COL(6), A, F(3), X(5), A,
 COL(51), A,
 COL(92), A,
 COL(110), A, F(3));

PUT FILE (SYSPRINT)
EDIT (DASHES,
 'RECORDS', 'RECORDS', 'RECORDS',
 'HOURS', 'HOURS',
 'TOTAL',
 'DEPARTMENT',
 'PAGE',
 'WRITTEN', 'VERIFIED', 'CORRECTED',
 'WRITTEN', 'VERIFIED',
 'HOURS',
 'PERSONNEL',
 DASHES, '')
(COL(1), A,
 COL(39), A, COL(52), A, COL(65), A,
 COL(80), A, COL(93), A,
 COL(106), A,
 COL(12), A,
 COL(30), A,
 COL(39), A, COL(51), A, COL(64), A,
 COL(79), A, COL(91), A,
 COL(106), A,
 COL(117), A);
END SUMMARY_PAGE;
SORT: PROCEDURE;

/*-------------------------------------------*/
/* THIS PROCEDURE Sorts the department totals into alphabetical order for output on the summary page. */
/*-------------------------------------------*/

DCL 1 TEMP LIKE DEPT,
    SIZE FIXED BINARY,
    SWITCHES FIXED BINARY INIT (1);

SIZE = CURRENT_DEPT;

SORT_PASS: DO WHILE (SWITCHES /= 0);
    SWITCHES = 0;

COMPARE_LOOP: DO I = 2 TO SIZE BY 1;
    IF DEPT(I-1).NAME > DEPT(I).NAME THEN
        INTERCHANGE: DO;
            TEMP = DEPT(I-1);
            DEPT(I-1) = DEPT(I);
            DEPT(I) = TEMP;
            SWITCHES = SWITCHES + 1;
        END INTERCHANGE;
        END COMPARE_LOOP;
    SIZE = SIZE - 1;
END SORT_PASS;

END SORT;
END PERIOD;
YTD: PROCEDURE OPTIONS (MAIN);

/* THE YEAR-TO-DATE PROGRAM INPUTS THE MASTER FILE AND */
/* PRODUCES A YEAR-TO-DATE REPORT DISPLAYING THE SAME */
/* INFORMATION AS THE PERIOD REPORT. */

DCL 1 INPUT,
  2 SYSTEM_SUBSYS,
    3 DEPT_NUMBER CHAR (2),
    3 SYSTEM_NUMBER CHAR (2),
    3 SUBSYS_NUMBER CHAR (2),
  2 FORMAT CHAR (4),
  2 TITLE CHAR (20),
  2 RECORD_LENGTH CHAR (3),
  2 AVERAGE,
    3 WRITE CHAR (3),
    3 VERIFY CHAR (3),
  2 ESTIMATE,
    3 WRITE CHAR (3),
    3 VERIFY CHAR (3),
  2 FLAGS,
    3 EQUIPMENT CHAR (1),
    3 CODED CHAR (1),
    3 DIFFICULTY CHAR (1),
    3 SCHEDULE CHAR (2),
  2 RECORDS,
    3 WRITTEN CHAR (10),
    3 VERIFIED CHAR (10),
    3 CORRECTED CHAR (10),
  2 HOURS,
    3 WRITTEN CHAR (10),
    3 VERIFIED CHAR (10);

DCL END_OF_FILE FIXED BINARY INIT (0),
  BLANK CHAR (1) INIT (' '),
  I FIXED BINARY (31,0),
  TEMP1 FLOAT DECIMAL (10),
  TEMP2 FLOAT DECIMAL (10),
  TEMP3 FLOAT DECIMAL (10),
  CARD(5) FLOAT DECIMAL (10) INIT (0),
  KTD (5) FLOAT DECIMAL (10) INIT (0),
  EFFICIENCY (3) FLOAT DECIMAL,
  MAX_LINES FIXED BINARY INIT (10),
  LINES FIXED BINARY INIT (0),
  PAGE FIXED BINARY INIT (0),
  CURRENT_DEPT FIXED BINARY INIT (0),
  DASHES CHAR (132) INIT ('-' ),
  PERIOD FIXED BINARY,
  RETURN_CODE FIXED BINARY (31,0) INIT (0),
  HEADER_CHAR (2) INIT ('00'),
  FISCAL_YEAR CHAR (10),
  CARD_HOURS FLOAT DECIMAL INIT (126.656),
  KTD_HOURS FLOAT DECIMAL INIT (87.2),
  GRAND_TOTAL (5) FLOAT DECIMAL (10) INIT (0),
  1 DEPT (100),
  2 PAGE FIXED BINARY,
  2 NAME CHAR (20),
  2 RECORDS,
    3 WRITTEN FIXED BINARY (31,0),
    3 VERIFIED FIXED BINARY (31,0),
    3 CORRECTED FIXED BINARY (31,0),
  2 HOURS,
    3 WRITTEN FLOAT DECIMAL (10),
    3 VERIFIED FLOAT DECIMAL (10),
  2 PERSONNEL FLOAT DECIMAL (10); 
OPEN FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
FILE (MASTER ) INPUT RECORD;
ON ENDFILE (MASTER ) END_OF_FILE = 1;
ON CONVERSION ONCHAR = '0';
READ FILE (MASTER ) INTO (INPUT ) ;
IF INPUT.DEPT_NUMBER = HEADER THEN DO;
   PERIOD = INPUT.SCHEDULE ;
   CARD_HOURS = CARD_HOURS * PERIOD ;
   KTD_HOURS = KTD_HOURS * PERIOD ;
   FISCAL_YEAR = INPUT.TITLE ;
   END ;
ELSE
   DO;
   PUT FILE (SYSPRINT )
      EDIT ('HEADER RECORD IS MISSING.
             EXECUTION TERMINATED.' )
         (A, A) ;
   END;
DO IT: DO WHILE (END_OF_FILE = 0 );
READ FILE (MASTER ) INTO (INPUT ) ;
IF END_OF_FILE = 0 & INPUT.SYSTEM_NUMBER ~= BLANK THEN OUTPUT_LINE: DO;
   I = CURRENT_DEPT ;
   DEPT(I).RECORDS.WRITTEN = DEPT(I).RECORDS.WRITTEN + INPUT.RECORDS.WRITTEN ;
   DEPT(I).RECORDS.VERIFIED = DEPT(I).RECORDS.VERIFIED + INPUT.RECORDS.VERIFIED ;
   DEPT(I).RECORDS.CORRECTED = DEPT(I).RECORDS.CORRECTED + INPUT.RECORDS.CORRECTED ;
   DEPT(I).HOURS.WRITTEN = DEPT(I).HOURS.WRITTEN + INPUT.HOURS.WRITTEN ;
   DEPT(I).HOURS.VERIFIED = DEPT(I).HOURS.VERIFIED + INPUT.HOURS.VERIFIED ;
   IF INPUT.EQUIPMENT = 'C' THEN CARD_TOTALS: DO;
      CARD(1) = CARD(1) + INPUT.RECORDS.WRITTEN ;
      CARD(2) = CARD(2) + INPUT.RECORDS.VERIFIED ;
      CARD(3) = CARD(3) + INPUT.RECORDS.CORRECTED ;
      CARD(4) = CARD(4) + INPUT.HOURS.WRITTEN ;
      CARD(5) = CARD(5) + INPUT.HOURS.VERIFIED ;
   ELSE
      END CARD_TOTALS ;
   ELSE
      END KTD_TOTALS ;
   EFFICIENCY = 0 ;
   TEMPI = INPUT.HOURS.WRITTEN ;
   IF TEMPI ~= 0 &
      INPUT.AVERAGE.WRITE ~= 0 THEN
      EFFICIENCY(1) = INPUT.RECORDS.WRITTEN /
                     TEMPI / INPUT.AVERAGE.WRITE ;
   TEMPI = INPUT.HOURS.VERIFIED ;
   IF TEMPI ~= 0 &
      INPUT.AVERAGE.VERIFY ~= 0 THEN
      EFFICIENCY(2) = INPUT.RECORDS.VERIFIED /
                     TEMPI / INPUT.AVERAGE.VERIFY ;
   TEMPI = TEMPI + INPUT.HOURS.WRITTEN ;
   IF INPUT.EQUIPMENT = 'C' THEN
      TEMP2 = TEMPI / CARD_HOURS ;
   ELSE
\[ \text{TEMP2} = \text{TEMP1} / \text{KTD.HOURS} \];
\[ \text{DEPT(I).PERSONNEL} = \text{DEPT(I).PERSONNEL} + \text{TEMP2} \];
\[ \text{EFFICIENCY(3)} = \text{EFFICIENCY(1)} \];
\[ \text{IF EFFICIENCY(1)} = 0 \text{ THEN} \]
\[ \text{EFFICIENCY(3)} = \text{EFFICIENCY(2)} \];
\[ \text{IF EFFICIENCY(1)} = 0 \text{ & EFFICIENCY(2)} = 0 \text{ THEN} \]
\[ \text{EFFICIENCY(3)} = (\text{EFFICIENCY(1)} \times \text{INPUT.HOURS.WRITTEN} + \text{EFFICIENCY(2)} \times \text{INPUT.HOURS.VERIFIED}) / \text{TEMP1} \];
\[ \text{EFFICIENCY(1)} = \text{EFFICIENCY(1)} \times 100 \];
\[ \text{EFFICIENCY(2)} = \text{EFFICIENCY(2)} \times 100 \];
\[ \text{EFFICIENCY(3)} = \text{EFFICIENCY(3)} \times 100 \];
\[ \text{LINES} = \text{LINES} + 1 \];
\[ \text{IF LINES} > \text{MAXLINES} \text{ THEN} \]
\[ \text{CALL NEW_PAGE;} \]
\[ \text{PUT FILE (SYSPRINT)} \]
\[ \text{EDIT (INPUT.DEPT_NUMBER,} \]
\[ \text{INPUT.SYSTEM_NUMBER,} \]
\[ \text{INPUT.SUBSYS_NUMBER,} \]
\[ \text{INPUT.FORMAT,} \]
\[ \text{INPUT.TITLE,} \]
\[ \text{INPUT.RECORDS.WRITTEN,} \]
\[ \text{INPUT.RECORDS.VERIFIED,} \]
\[ \text{INPUT.RECORDS.CORRECTED,} \]
\[ \text{INPUT.HOURS.WRITTEN,} \]
\[ \text{INPUT.HOURS.VERIFIED,} \]
\[ \text{TEMP1,} \]
\[ \text{TEMP2,} \]
\[ \text{INPUT.RECORD_LENGTH,} \]
\[ \text{INPUT.AVERAGE.WRITE,} \]
\[ \text{INPUT.AVERAGE.VERIFY,} \]
\[ \text{INPUT.ESTIMATE.WRITE,} \]
\[ \text{INPUT.ESTIMATE.VERIFY,} \]
\[ \text{EFFICIENCY \}} \]
\[ \text{(COL(1), 4 A,} \]
\[ \text{COL(10), A,} \]
\[ \text{COL(17), A,} \]
\[ \text{COL(38), F(8), COL(48), F(8),} \]
\[ \text{COL(57), F(6),} \]
\[ \text{COL(65), F(7,1), COL(74), F(7,1),} \]
\[ \text{COL(82), F(8,1), COL(91), F(5,1),} \]
\[ \text{COL(97), A,} \]
\[ \text{COL(102), A,} \]
\[ \text{COL(107), A,} \]
\[ \text{COL(112), A,} \]
\[ \text{COL(117), A,} \]
\[ \text{COL(122), F(3), X(1), F(3), X(1), F(3)) ;} \]
\[ \text{END OUTPUT LINE;} \]
\[ \text{IF END_OF_FILE = 1 \} \]
\[ \text{INPUT.SYSTEM_NUMBER = BLANK THEN} \]
\[ \text{NEW_DEPT: DO; IF CURRENT_DEPT = 0 THEN} \]
\[ \text{PUT_TOTALS: DO; \} \]
\[ \text{GRAND_TOTAL(1) = GRAND_TOTAL(1) +} \]
\[ \text{DEPT(I).RECORDS.WRITTEN;} \]
\[ \text{GRAND_TOTAL(2) = GRAND_TOTAL(2) +} \]
\[ \text{DEPT(I).RECORDS.VERIFIED;} \]
\[ \text{GRAND_TOTAL(3) = GRAND_TOTAL(3) +} \]
\[ \text{DEPT(I).RECORDS.CORRECTED;} \]
\[ \text{GRAND_TOTAL(4) = GRAND_TOTAL(4) +} \]
\[ \text{DEPT(I).HOURS.WRITTEN;} \]
\[ \text{GRAND_TOTAL(5) = GRAND_TOTAL(5) +} \]
\[ \text{DEPT(I).HOURS.VERIFIED;} \]
\[ \text{TEMP1 = DEPT(I).HOURS.WRITTEN +} \]
\[ \text{DEPT(I).HOURS.VERIFIED;} \]
\[ \text{PUT FILE (SYSPRINT)} \]
EDIT (DASHES,
'DEPARTMENT YEAR-TO-DATE TOTALS',
DEPT(I).RECORDS.WRITTEN,
DEPT(I).RECORDS.VERIFIED,
DEPT(I).RECORDS.CORRECTED,
DEPT(I).HOURS.WRITTEN,
DEPT(I).HOURS.VERIFIED,
TEMP1,
DEPT(I).PERSONNEL)
(COL(1), A, COL(1), A,
COL(38), F(8), COL(48), F(8),
COL(57), F(6),
COL(65), F(7,1), COL(74), F(7,1),
COL(82), F(8,1), COL(91), F(5,1))
END PUT_TOTAIS;
IF END_OF_FILE = 0 THEN
INITIALIZE_NEW_DEPT:
DO;
CURRENT_DEPT = CURRENT_DEPT + 1;
I = CURRENT_DEPT;
DEPT(I).NAME = INPUT.TITLE;
CALL NEW_PAGE;
DEPT(I).PAGE = PAGE;
DEPT(I).RECORDS.WRITTEN = 0;
DEPT(I).RECORDS.VERIFIED = 0;
DEPT(I).RECORDS.CORRECTED = 0;
DEPT(I).HOURS.WRITTEN = 0;
DEPT(I).HOURS.VERIFIED = 0;
DEPT(I).PERSONNEL = 0;
END INITIALIZE_NEW_DEPT;
END NEW_DEPT;
END DO_IT;
CALL SUMMARY_PAGE;
CALL SORT;
PRINT_INDEX:
DO I = 1 TO CURRENT_DEPT BY 1;
LINES = LINES + 1;
IF LINES >= MAXLINES THEN
CALL SUMMARY_PAGE;
TEMP1 = DEPT(I).HOURS.WRITTEN + DEPT(I).HOURS.VERIFIED;
PUT FILE (SYSPRINT)
EDIT(DEPT(I).NAME,
DEPT(I).PAGE,
DEPT(I).RECORDS.WRITTEN,
DEPT(I).RECORDS.VERIFIED,
DEPT(I).RECORDS.CORRECTED,
DEPT(I).HOURS.WRITTEN,
DEPT(I).HOURS.VERIFIED,
TEMP1,
DEPT(I).PERSONNEL)
(COL(7), A,
COL(30), F(4),
COL(37), F(10),
COL(50), F(10),
COL(63), F(10),
COL(77), F(10,1),
COL(90), F(10,1),
COL(103), F(10,1), COL(116), F(10,1))
END PRINT_INDEX;
TEMP1 = CARD(9) + CARD(5);
TEMP2 = TEMP1 / CARD_HOURS;
TEMP3 = TEMP2;
PUT FILE (SYSPRINT)
EDIT (DASHES,
'CARD YEAR-TO-DATE TOTALS FOR PERIOD', PERIOD,
CARD,
TEMP1,
TEMP2)
(COL(1), A,
   SKIP(2), COL(1), A, F(3),
   COL(37), F(10),
   COL(50), F(10),
   COL(63), F(10),
   COL(77), F(10,1),
   COL(90), F(10,1),
   COL(103), F(10,1),
   COL(116), F(10,1))
TEMP1 = KTD(4) + KTD(5);
TEMP2 = TEMP1 / KTD_HOURS;
TEMP3 = TEMP3 + TEMP2;
PUT FILE (SYSPRINT)
   EDIT ('KEY-TO-DISK YEAR-TO-DATE TOTALS FOR PERIOD',
         PERIOD,
         KTD,
         TEMP1,
         TEMP2)
   (COL(1), A, F(3),
   COL(37), F(10),
   COL(50), F(10),
   COL(63), F(10),
   COL(77), F(10,1),
   COL(90), F(10,1),
   COL(103), F(10,1),
   COL(116), F(10,1))
TEMP1 = GRAND_TOTAL(4) + GRAND_TOTAL(5);
PUT FILE (SYSPRINT)
   EDIT ('YEAR-TO-DATE TOTALS FOR PERIOD ', PERIOD,
         GRAND_TOTAL,
         TEMP1,
         TEMP3)
   (COL(1), A,
   SKIP(2), COL(1), A, F(3),
   COL(37), F(10),
   COL(50), F(10),
   COL(63), F(10),
   COL(77), F(10,1),
   COL(90), F(10,1),
   COL(103), F(10,1),
   COL(116), F(10,1))
IF GRAND_TOTAL(1) = 0 & GRAND_TOTAL(4) = 0 THEN
RETURN_CODE = 8;
CALL INESARC(RETURN_CODE);

NEW_PAGE: PROCEDURE;

********************************************************************
* THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON *
* EACH NEW PAGE OF THE YEAR-TO-DATE REPORT. *
********************************************************************

LINES = 0;
PAGE = PAGE + 1;
PUT FILE (SYSPRINT) PAGE
   EDIT ('CITY OF LOS ANGELES',
         'DATA SERVICE BUREAU',
         'PERIOD', PERIOD,
         DEPT(CURRENT DEPT).NAME,
         'DATA CONVERSION YEAR-TO-DATE REPORT',
         FISCAL_YEAR,
         'PAGE', PAGE)
   (LINE(3), COL(56), A,
   COL(56), A,
   COL(6), A, F(3), X(5), A,
COL(48), A,
COL(92), A,
COL(110), A, F(3));

PUT FILE (SYSPRINT)
EDIT (DASHES,
'SYSTEM',
'COR-', 'PERS', 'AVG',
'EFFICIENCY')
(SKIP(2), COL(1), A,
COL(1), A, COL(58), A,
COL(92), A, COL(97), A,
COL(122), A);

PUT FILE (SYSPRINT)
EDIT ('SUB-SYS', 'RECORDS', 'RECORDS',
'REC-',
'HOURS', 'HOURS', 'TOTAL', 'SON-',
'REC',
'AVERAGE ESTIMATING',
'RATIOS')
(COL(1), A, COL(39), A, COL(49), A,
COL(58), A,
COL(66), A, COL(75), A, COL(85), A,
COL(92), A, COL(97), A, COL(102), A,
COL(126), A);

PUT FILE (SYSPRINT)
EDIT ('NUMBER', 'FORMAT', 'JOB NAME',
'WRITTEN', 'VERIFIED',
'TIONS',
'WRITTEN', 'VERIFIED',
'HOURS', 'NEL', 'LEN',
'WRT VER WRT VER',
'WRT VER ALL',
DASHES, ' ')
(COL(1), A, COL(9), A, COL(16), A,
COL(39), A, COL(48), A,
COL(58), A, COL(65), A, COL(73), A,
COL(85), A, COL(92), A, COL(97), A,
COL(102), A, COL(122), A,
COL(1), A, COL(1), A);

END NEW_PAGE;

SUMMARY_PAGE:
PROCEDURE;

/*---------------------------------------------------------------*/
/* THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON */
/* EACH NEW SUMMARY PAGE OF THE YEAR-TO-DATE REPORT. */
/*---------------------------------------------------------------*/

LINES = 0;
PAGE = PAGE + 1;
PUT FILE (SYSPRINT) PAGE
EDIT ('CITY OF LOS ANGELES',
'DATA SERVICE BUREAU',
'PERIOD', PERIOD,
'INDEX TO DEPARTMENTS',
'DATA CONVERSION YEAR-TO-DATE REPORT',
FISCAL_YEAR,
'PAGE', PAGE)
(LINE(3), COL(56), A,
COL(56), A,
COL(6), A, F(3), X(5), A,
COL(48), A,
COL(92), A,
COL(110), A, F(3));

PUT FILE (SYSPRINT)
EDIT (DASHES, 'RECORDS', 'RECORDS', 'RECORDS', 'HOURS', 'HOURS', 'TOTAL', 'DEPARTMENT', 'PAGE', 'WRITTEN', 'VERIFIED', 'CORRECTED', 'HOURS', 'PERSONNEL', DASHES, ' ')
(COL(1), A, COL(39), A, COL(52), A, COL(65), A, COL(80), A, COL(93), A, COL(106), A, COL(12), A, COL(30), A, COL(39), A, COL(51), A, COL(64), A, COL(79), A, COL(91), A, COL(106), A, COL(117), A);
END SUMMARY_PAGE;

SORT: PROCEDURE;
/X---------------------------------------------*/
/X        THIS PROCEDURE SORTS THE DEPARTMENT TOTALS INTO ALPHABETICAL */
/X        ORDER BY DEPARTMENT NAME FOR OUTPUT ON THE SUMMARY REPORT. */
/X---------------------------------------------*/

DCL 1 TEMP LIKE DEPT,
SIZE FIXED BINARY,
SWITCHES FIXED BINARY INIT (1);
SIZE = CURRENT_DEPT;
SORT_PASS: DO WHILE (SWITCHES ^= 0);
SWITCHES = 0;
COMPARE_LOOP: DO I = 2 TO SIZE BY 1;
IF DEPT(I-1).NAME > DEPT(I).NAME THEN 
INTERCHANGE: DO;
TEMP = DEPT(I-1);
DEPT(I-1) = DEPT(I);
DEPT(I) = TEMP;
SWITCHES = SWITCHES + 1;
END INTERCHANGE;
END COMPARE_LOOP;
SIZE = SIZE - 1;
END SORT_PASS;
END SORT;
END YTD;
OPERATR: PROCEDURE OPTIONS (MAIN);
OLD OPERATOR CHAR (3) INIT (', '),
OPER_TOTALS(6) FLOAT DECIMAL INIT (0)
DCL 1 TABLE (0-MAXENTRIES),
2 ACCOUNT_NUMBER CHAR (10),
2 TITLE CHAR (20),
2 RECORD_LENGTH CHAR (3),
2 AVERAGE,
3 WRITE CHAR (3),
3 VERIFY CHAR (3),
2 ESTIMATE,
3 WRITE CHAR (3),
3 VERIFY CHAR (3),
2 FLAGS,
3 EQUIPMENT CHAR (1),
3 CODED CHAR (1),
3 DIFFICULTY CHAR (1),
3 SCHEDULE CHAR (2),
2 RECORDS,
3 WRITTEN CHAR (10),
3 VERIFIED CHAR (10),
3 CORRECTED CHAR (10),
2 HOURS,
3 WRITTEN CHAR (10),
3 VERIFIED CHAR (10);
DCL 1 TEMP LIKE TABLE;
DCL SHIFT_NAME (3) CHAR (7) INIT ('DAY', 'NIGHT', 'MORNING'),
SHIFT_MAP (0:9) FIXED BINARY
INIT (1, 1, 1, 1, 1, 1, 1, 1, 0),
1 GROUPS (0:9),
2 RECORDS_WRITTEN FLOAT DECIMAL (10) INIT (0),
2 ERRORS_WRITTEN FLOAT DECIMAL (10) INIT (0),
2 HOURS_WRITTEN FLOAT DECIMAL (10) INIT (0),
2 RECORDS_VERIFIED FLOAT DECIMAL (10) INIT (0),
2 ERRORS_VERIFIED FLOAT DECIMAL (10) INIT (0),
2 HOURS_VERIFIED FLOAT DECIMAL (10) INIT (0),
2 CORRECTIONS FLOAT DECIMAL (10) INIT (0, 0, 0),
2 EFFICIENCY_TIME (2) FLOAT DECIMAL (10) INIT (0, 0),
2 COUNT FIXED BINARY INIT (0),
1 SHIFTS (3) LIKE GROUPS,
1 TOTALS LIKE GROUPS;
OPEN FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
FILE (DETAIL ) OUTPUT STREAM PRINT LINESIZE(132),
FILE (SUMMARY) OUTPUT STREAM PRINT LINESIZE(132),
FILE (SYSSIN ) INPUT STREAM,
FILE (TABLEIN ) INPUT RECORD,
FILE (INPUT ) INPUT RECORD;
ON ENDFILE (INPUT ) END_OF_FILE = 1;
ON ENDFILE (TABLEIN ) END_OF_FILE = 1;
ON ENDFILE (SYSSIN ) END_OF_FILE = 1;
ON CONVERSION ONCHAR = '0';
TABLE(0) = ' ';
READ FILE (TABLEIN ) INTO (TEMP );
IF TEMP.ACCOUNT_NUMBER = HEADER THEN
  DO;
    PERIOD = 'PERIOD ' || TEMP.SCHEDULE;
    FISCAL_YEAR = TEMP.TITLE;
    END;
  ELSE
    DO;
      PUT FILE (SYSPRINT)
      EDIT ('HEADER RECORD IS MISSING. ',
            'EXECUTION TERMINATED.')
      (A, A);
      END_OF_FILE = 1;
      RETURN_CODE = 8;
      END;
BUILD_TABLE:  DO WHILE (END_OF_FILE = 0);
READ FILE (TABLEIN) INTO (TEMP);
IF END_OF_FILE = 0 &
    SUBSTR(TEMP, ACCOUNT_NUMBER, 3, 2) ~= BLANK THEN
NEW_ENTRY: DO;
    ENTRY = ENTRY + 1;
    IF ENTRY <= MAXENTRIES THEN
        TABLE_ENTRY = TEMP;
    END NEW_ENTRY;
END BUILD_TABLE;
END_OF_FILE = 0;
IF ENTRY <= MAXENTRIES THEN
    MAXENTRIES = ENTRY;
ELSE
    ERROR: DO;
    PUT FILE (SYSPRINT)
        EDIT ('STANDARDS TABLE HAS EXCEEDED MAXIMUM ',
        'SIZE. EXECUTION TERMINATED. ')
        (COL(1), A, A);
    END_OF_FILE = 1;
    RETURN_CODE = 16;
    ERROR;
END ERROR;
GET FILE (SYSIN) DATA (PERIOD, FISCAL_YEAR);
DO_IT: DO WHILE (END_OF_FILE = 0);
READ FILE (INPUT) INTO (RECORD);
IF END_OF_FILE = 1 |
    OLD_OPERATOR ~= OPERATOR THEN
NEW_OPERATOR: DO;
    IF OLD_OPERATOR ~= BLANK THEN
        PUT_TOTALS: DO;
        IF WRITE_TIME ~= 0 THEN
            EFFICIENCY(1) = EFFICIENCY(1) / WRITE_TIME;
        IF VERIFY_TIME ~= 0 THEN
            EFFICIENCY(2) = EFFICIENCY(2) / VERIFY_TIME;
        IF WRITE_TIME ~= 0 |
            VERIFY_TIME ~= 0 THEN
            EFFICIENCY(3) = (EFFICIENCY(1) * WRITE_TIME +
                EFFICIENCY(2) * VERIFY_TIME) /
            (WRITE_TIME + VERIFY_TIME);
        IF INDEX < DIGITS,
            SUBSTR(OLD_OPERATOR, 1, 1)) ~= 0
            THEN
            CALL SAVE_GROUP_INFORMATION;
            OPER_TOTALS(3) = OPER_TOTALS(3) / 60.0;
            OPER_TOTALS(4) = OPER_TOTALS(4) / 60.0;
            PUT_FILE (DETAIL)
                EDIT (DASHES,
                    'OPERATOR PERIOD TOTALS - WRITE',
                    OPER_TOTALS(1),
                    OPER_TOTALS(3),
                    OPER_TOTALS(5),
                    EFFICIENCY(1),
                    'OPERATOR PERIOD TOTALS - VERIFY',
                    OPER_TOTALS(2),
                    OPER_TOTALS(4),
                    OPER_TOTALS(6),
                    EFFICIENCY(2),
                    'OVERALL EFFICIENCY RATING - ',
                    EFFICIENCY(3))
                (COL(1), A,
                SKIP(2), COL(1), A,
                COL(47), F(8),
                COL(56), F(8,1),
                COL(65), F(8),
                COL(123), F(3),
                COL(1), A,
                COL(47), F(8),
                COL(56), F(8,1),
                COL(65), F(8),
                COL(123), F(3),
SKIP(2), COL(1), A, F(3));
TEMP1 = OPER_TOTALS(3) + OPER_TOTALS(4);
IF SUMMARY_LINES >= MAX_LINES THEN
CALL NEW_SUMMARY_PAGE;
PUT FILE (SUMMARY) EDIT (OLD_OPERATOR, 
OPER_TOTALS(1), 
OPER_TOTALS(5), 
OPER_TOTALS(3), 
OPER_TOTALS(2), 
OPER_TOTALS(6), 
OPER_TOTALS(4), 
TE_{PL}, 
EFFICIENCY) 
(SKIP(2), COL(4), A, 
COL(10), F(8), 
COL(19), F(8), 
COL(28), F(8,1), 
COL(38), F(8), 
COL(49), F(8), 
COL(59), F(8,1), 
COL(68), F(8,1), 
COL(79), F(3), 
COL(86), F(3), 
COL(93), F(3));
SUMMARY_LINES = SUMMARY_LINES + 2;
END PUT_TOTALS;
OPER_TOTALS = 0; OLD_OPERATOR = OPERATOR;
IF END_OF_FILE = 0 THEN
CALL NEW_PAGE;
EFFICIENCY = 0;
WRITE_TIME = 0;
VERIFY_TIME = 0;
END NEW_OPERATOR;
IF END_OF_FILE = 0 THEN
DO;
CALL SEARCH;
TEMP1 = 0;
IF STATUS = 'A' THEN
ADD_WRITE_TOTALS: DO;
OPER_TOTALS(1) = OPER_TOTALS(1) + RECORD.COUNT;
OPER_TOTALS(3) = OPER_TOTALS(3) + MINUTES;
OPER_TOTALS(5) = OPER_TOTALS(5) + RECORD.CORRECTIONS;
STATUS_TYPE = 'WRITE';
IF MINUTES = 0 & TABLEENTRY).AVERAGE.WRITE = 0 THEN
TEMP1 = 60.0 * 100.0 * RECORD.COUNT / MINUTES / TABLEENTRY).AVERAGE.WRITE;
IF SUBSTR(ACCOUNT_FIELD,1,7) = '190001A' | 
SUBSTR(ACCOUNT_FIELD,1,7) = '190001B' THEN
TEMP1 = TEMP1 * 4;
EFFICIENCY(1) = EFFICIENCY(1) + TEMP1 * MINUTES;
IF TEMP1 = 0 & RECORD.COUNT = 0 THEN
WRITE_TIME = WRITE_TIME + MINUTES;
END ADD_WRITE_TOTALS;
ELSE
ADD_VERIFY_TOTALS: DO;
OPER_TOTALS(2) = OPER_TOTALS(2) + RECORD.COUNT;
OPER_TOTALS(4) = OPER_TOTALS(4) + MINUTES;
OPER_TOTALS(6) = OPER_TOTALS(6) + RECORD.CORRECTIONS;
STATUS_TYPE = 'VERIFY';
IF MINUTES = 0 & TABLEENTRY).AVERAGE.VERIFY = 0 THEN
TEMP1 = 60.0 * 100.0 * RECORD.COUNT / MINUTES /
TABLE(ENTRY).AVERAGE.VERIFY:
IF SUBSTR(ACCOUNT_FIELD,1,7) = '190001A' 
  SUBSTR(ACCOUNT_FIELD,1,7) = '190001B' THEN
  TEMP1 = TEMP1 * 6;
  EFFICIENCY(2) = EFFICIENCY(2) + TEMP1 * MINUTES;
  IF TEMP1 == 0 & RECORD.COUNT == 0 THEN
    VERIFY_TIME = VERIFY_TIME + MINUTES;
    END ADD_VERIFY_TOTALS;
  IF LINES >= MAXLINES THEN
    CALL NEW_PAGE;
    PUT FILE(DETAIL )
    EDIT (SYSTEM, '-', SUB_SYSTEM, 
      FORMAT, 
      TABLE(ENTRY).TITLE, 
      BATCH, 'STATUS', 
      RECORD.COUNT, 
      OPEN_MONTH, '-', 
      OPEN_DAY, '-', 
      OPEN_YEAR, 
      CLOSE_TIME, 
      MINUTES, 
      RECORD.CORRECTIONS, 
      TABLE(ENTRY).RECORD.LENGTH, 
      TABLE(ENTRY).AVERAGE.WRITE, 
      TABLE(ENTRY).AVERAGE.VERIFY, 
      TEMP1))
      (COL(1), 3 A, 
       COL(10), A, 
       COL(16), A, 
       COL(40), A, COL(46), A, 
       COL(55), F(4), 
       COL(63), 5 A, 
       COL(73), A, 
       COL(79), A, 
       COL(85), F(4), 
       COL(92), F(4), 
       COL(101), A, 
       COL(108), A, 
       COL(115), A, 
       COL(125), F(3)) ;
    LINES = LINES + 1;
    END PUT_LINE;
  END DO_IT;
  CALL OUTPUT Totals ;
  IF PAGE = 0 THEN
    RETURN_CODE = 8 ;
    CALL IHESARC(RETURN_CODE) ;
SEARCH:
PROCEDURE ;

/* THIS PROCEDURE PERFORMS A BINARY SEARCH IN ORDER TO FIND THE */
/* CURRENT APPLICATION ENTRY IN ORDER THAT THE AVERAGES CAN BE */
/* USED FOR THE EFFICIENCY CALCULATION. */

LOWER = 1 ;
UPPER = MAXENTRIES ;
FIND_ENTRY:
  DO WHILE (LOWER <= UPPER &
    ACCOUNT -= TABLE(ENTRY).ACCOUNT_NUMBER) ;
    IF ACCOUNT > TABLE(ENTRY).ACCOUNT_NUMBER THEN
      LOWER = ENTRY + 1 ;
    ELSE
      UPPER = ENTRY - 1 ;
ENTRY = (LOWER + UPPER) / 2;
END FIND_ENTRY;
IF LOWER > UPPER THEN
ENTRY = 0;
END SEARCH;

SAVE_GROUP_INFORMATION:
PROCEDURE;

/* THIS PROCEDURE SAVES AND CALCULATES GROUP, SHIFT, AND SECTION */
/* TOTALS FOR LATER OUTPUT ON THE SUMMARY REPORT */

I = SUBSTR(OLD_OPERATOR, 1, 1);
GROUPS(I).RECORDS_WRITTEN = GROUPS(I).RECORDS_WRITTEN +
OPER_TOTALS(I);
GROUPS(I).ERRORS_WRITTEN = GROUPS(I).ERRORS_WRITTEN +
OPER_TOTALS(5);
GROUPS(I).HOURS_WRITTEN = GROUPS(I).HOURS_WRITTEN +
OPER_TOTALS(3);
GROUPS(I).RECORDS_VERIFIED = GROUPS(I).RECORDS_VERIFIED +
OPER_TOTALS(2);
GROUPS(I).CORRECTIONS = GROUPS(I).CORRECTIONS +
OPER_TOTALS(6);
GROUPS(I).HOURS_VERIFIED = GROUPS(I).HOURS_VERIFIED +
OPER_TOTALS(4);
GROUPS(I).EFFICIENCY(1) = GROUPS(I).EFFICIENCY(1) +
EFFICIENCY(1) * WRITE_TIME;
GROUPS(I).EFFICIENCY(2) = GROUPS(I).EFFICIENCY(2) +
EFFICIENCY(2) * VERIFY_TIME;
GROUPS(I).EFFICIENCY_TIME(1) = GROUPS(I).EFFICIENCY_TIME(1) +
WRITE_TIME;
GROUPS(I).EFFICIENCY_TIME(2) = GROUPS(I).EFFICIENCY_TIME(2) +
VERIFY_TIME;
GROUPS(I).COUNT = GROUPS(I).COUNT + 1;
END SAVE_GROUP_INFORMATION;

OUTPUT_TOTALS:
PROCEDURE;
CALL NEW_SUMMARY_PAGE;
PUT FILE(OUTPUT_GROUPS);
EDIT (DASHES, ' G R O U P S ', DASHES)
(SKIP(4), COL(1), A(41), A, A(42));

OUTPUT_GROUPS:
DO I = 0 TO 9;
IF GROUPS(I).EFFICIENCY_TIME(1) = 0 THEN
GROUPS(I).EFFICIENCY(1) = GROUPS(I).EFFICIENCY(1) /
GROUPS(I).EFFICIENCY_TIME(1);
IF GROUPS(I).EFFICIENCY_TIME(2) = 0 THEN
GROUPS(I).EFFICIENCY(2) = GROUPS(I).EFFICIENCY(2) /
GROUPS(I).EFFICIENCY_TIME(2);
IF GROUPS(I).EFFICIENCY_TIME(1) = 0 THEN
GROUPS(I).EFFICIENCY(3) = (GROUPS(I).EFFICIENCY(1) *
GROUPS(I).EFFICIENCY_TIME(1) +
GROUPS(I).EFFICIENCY(2) *
GROUPS(I).EFFICIENCY_TIME(2))
(GROUPS(I).EFFICIENCY_TIME(1) +
GROUPS(I).EFFICIENCY_TIME(2));
TEMP1 = 0;
IF GROUPS(I).COUNT = 0 THEN
TEMP1 = (GROUPS(I).HOURS_WRITTEN +

GROUPS(1).HOURS_VERIFIED = (GROUPS(1).COUNT * 60.0);
GROUPS(1).HOURS_WRITTEN = GROUPS(1).HOURS_WRITTEN / 60.0;
GROUPS(1).HOURS_VERIFIED = GROUPS(1).HOURS_VERIFIED / 60.0;
J = SHIFT_MAP(I);
IF J > 0 THEN
  PUT FILE (SUMMARY)
  EDIT (I, GROUPS(I).RECORDS_WRITTEN, GROUPS(I).ERRORS_WRITTEN, GROUPS(I).HOURS_WRITTEN, GROUPS(I).RECORDS_VERIFIED, GROUPS(I).CORRECTIONS, GROUPS(I).HOURS_VERIFIED, TEMPL, GROUPS(I).EFFECTIVENESS(I), GROUPS(I).EFFECTIVENESS(2), GROUPS(I).EFFECTIVENESS(3)) (SKIP(2), COL(4), F(3), COL(10), F(8), COL(19), F(8), COL(28), F(8,1), COL(38), F(8), COL(49), F(8), COL(59), F(8,1), COL(68), F(8,1), COL(79), F(3), COL(86), F(3), COL(93), F(3));
END SAVE_SHIFT_INFORMATION;
END OUTPUT_GROUPS;
PUT FILE (SUMMARY)
EDIT (DASHES, 'SHIFTS', DASHES)
OUTPUT_SHIFTS:
  DO I = 1 TO 3
    IF SHIFTS(I).EFFICIENCY_TIME(1) = 0 THEN
      SHIFTS(I).EFFICIENCY(1) = SHIFTS(I).EFFICIENCY(1) / SHIFTS(I).EFFICIENCY_TIME(1);
    IF SHIFTS(I).EFFICIENCY_TIME(2) = 0 THEN
      SHIFTS(I).EFFICIENCY(2) = SHIFTS(I).EFFICIENCY(2) / SHIFTS(I).EFFICIENCY_TIME(2);
    IF SHIFTS(I).EFFICIENCY_TIME(1) = 0 | SHIFTS(I).EFFICIENCY_TIME(2) = 0 THEN
    TEMP1 = 0
    IF SHIFTS(I).COUNT = 0 THEN
      TEMP1 = (SHIFTS(I).HOURS_WRITTEN + SHIFTS(I).HOURS_VERIFIED) / SHIFTS(I).COUNT
    PUT FILE (SUMMARY)
      (SKIP(2), COL(4), F(3), COL(10), F(8), COL(19), F(8), COL(28), F(8,1), COL(38), F(8), COL(49), F(8), COL(59), F(8,1), COL(68), F(8,1), COL(79), F(3), COL(86), F(3), COL(93), F(3));
    TOTALS.RECORDS_WRITTEN = TOTALS.RECORDS_WRITTEN + SHIFTS(I).RECORDS_WRITTEN;
    TOTALS.ERRORS_WRITTEN = TOTALS.ERRORS_WRITTEN + SHIFTS(I).ERRORS_WRITTEN;
    TOTALS.HOURS_WRITTEN = TOTALS.HOURS_WRITTEN + SHIFTS(I).HOURS_WRITTEN;
    TOTALS.RECORDS_VERIFIED = TOTALS.RECORDS_VERIFIED + SHIFTS(I).RECORDS_VERIFIED;
    TOTALS.CORRECTIONS = TOTALS.CORRECTIONS + SHIFTS(I).CORRECTIONS;
    TOTALS.HOURS_VERIFIED = TOTALS.HOURS_VERIFIED + SHIFTS(I).HOURS_VERIFIED;
    TOTALS.EFFICIENCY(1) = TOTALS.EFFICIENCY(1) + SHIFTS(I).EFFICIENCY(1) * SHIFTS(I).EFFICIENCY_TIME(1);
    TOTALS.EFFICIENCY(2) = TOTALS.EFFICIENCY(2) + SHIFTS(I).EFFICIENCY(2) * SHIFTS(I).EFFICIENCY_TIME(2);
    TOTALS.EFFICIENCY_TIME(1) = TOTALS.EFFICIENCY_TIME(1) + SHIFTS(I).EFFICIENCY_TIME(1);
TOTALS.EFFICIENCY_TIME(2) = TOTALS.EFFICIENCY_TIME(2) +
SHIFTS(1).EFFICIENCY_TIME(2);
TOTALS.COUNT = TOTALS.COUNT + SHIFTS(1).COUNT;
END OUTPUT_SHIFTS;
IF TOTALS.EFFICIENCY_TIME(1) /= 0 THEN
TOTALS.EFFICIENCY(1) = TOTALS.EFFICIENCY(1) / 
TOTALS.EFFICIENCY_TIME(1);
IF TOTALS.EFFICIENCY_TIME(2) /= 0 THEN
TOTALS.EFFICIENCY(2) = TOTALS.EFFICIENCY(2) / 
TOTALS.EFFICIENCY_TIME(2);
IF TOTALS.EFFICIENCY_TIME(1) /= 0 |
TOTALS.EFFICIENCY_TIME(2) /= 0 THEN
TOTALS.EFFICIENCY(3) = (TOTALS.EFFICIENCY(1) * 
TOTALS.EFFICIENCY_TIME(1) + 
TOTALS.EFFICIENCY(2) * 
TOTALS.EFFICIENCY_TIME(2)) / 
(TOTALS.EFFICIENCY_TIME(1) + 
TOTALS.EFFICIENCY_TIME(2));
TEMPI = 0;
IF TOTALS.COUNT /= 0 THEN
TEMPI = (TOTALS.HOURS_WRITTEN + TOTALS.HOURS_VERIFIED) / 
TOTALS.COUNT;
PUT FILE (SUMMARY )
EDIT (DASHES, 'O V E R A L L ', DASHES,
TOTALS.RECORDS_WRITTEN,
TOTALS.ERRORS_WRITTEN,
TOTALS.HOURS_WRITTEN,
TOTALS.RECORDS_VERIFIED,
TOTALS.CORRECTIONS,
TOTALS.HOURS_VERIFIED,
TEMPI,
TOTALS.EFFICIENCY)
(SKIP(4), COL(1), A(40), A, A(41),
SKIP(2), COL(10), F(8),
COL(19), F(8),
COL(28), F(8.1),
COL(38), F(8),
COL(49), F(8),
COL(59), F(8.1),
COL(68), F(8.1),
COL(79), F(3),
COL(86), F(3),
COL(93), F(3));
END OUTPUT_TOTALS;

NEW_PAGE: PROCEDURE;

/*-------------------------------------------------------------*/
/* THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON */
/* EACH NEW PAGE FOR THE DETAIL OPERATOR ANALYSIS REPORT. */
/*-------------------------------------------------------------*/
LINES = 0;
PAGE = PAGE + 1;
PUT FILE (DETAIL ) PAGE
EDIT ('CITY OF LOS ANGELES',
'DATA SERVICE BUREAU',
PERIOD,
'OPERATOR ', OPERATOR,
'DETAIL OPERATOR ANALYSIS REPORT',
FISCAL_YEAR,
'PAGE', PAGE)
(LINE(4),
COL(57), A,
COL(57), A,
COL(6), A,
NEW_SUMMARY_PAGE:
PROCEDURE;

/*-----------------------------------------------*/
/* THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON */
/* EACH NEW SUMMARY PAGE OF THE OPERATOR ANALYSIS REPORT. */
/*-----------------------------------------------*/

PUT FILE (DETAIL)
EDIT (',', DASHES,
'SYSTEM')
(COL(1), A, COL(1), A,
COL(1), A) ;

PUT FILE (DETAIL)
EDIT ('SUB-SYS',
'OPEN',
'CLOSE',
'ELAP',
'CORREC-',
'RECORD',
'AVERAGE',
'EFFICIENCY')
(COL(1), A,
COL(73), A,
COL(79), A,
COL(85), A,
COL(91), A,
COL(99), A,
COL(109), A,
COL(121), A) ;

PUT FILE (DETAIL)
EDIT ('NUMBER',
'FORMAT',
'JOB NAME',
'BATCH',
'Status',
'RECORDS',
'OPEN DATE',
'TIME',
'TIME',
'TIME',
'TIONS',
'LENGTH',
'WRITE VERIFY',
'RATIO',
'DASHES')
(COL(1), A,
COL(9), A,
COL(23), A,
COL(39), A,
COL(45), A,
COL(54), A,
COL(62), A,
COL(73), A,
COL(79), A,
COL(85), A,
COL(92), A,
COL(99), A,
COL(107), A,
COL(124), A,
COL(1), A) ;

END NEW_PAGE;
SUMMARY_PAGE = SUMMARY_PAGE + 1;
PUT FILE (SUMMARY) PAGE
EDIT ('CITY OF LOS ANGELES',
   'DATA SERVICE BUREAU',
   PERIOD,
   'SUMMARY OPERATOR ANALYSIS REPORT',
   FISCAL_YEAR,
   'PAGE', SUMMARY_PAGE)
   (LINE(4),
    COL(57), A,
    COL(57), A,
    COL(6), A,
    COL(51), A,
    COL(93), A,
    COL(110), A, F(4));
PUT FILE (SUMMARY)
EDIT (DASHES,
   'OPERATOR',
   'RECORDS',
   'ERRORS',
   'HOURS',
   'RECORDS',
   'ERRORS',
   'HOURS',
   'TOTAL',
   'EFFICIENCY')
   (SKIP(2), COL(1), A,
    COL(1), A,
    COL(11), A,
    COL(21), A,
    COL(30), A,
    COL(39), A,
    COL(50), A,
    COL(60), A,
    COL(71), A,
    COL(78), A);
PUT FILE (SUMMARY)
EDIT ('NUMBER',
   'WRITTEN',
   'WRITTEN',
   'WRITTEN',
   'VERIFIED',
   'CORRECTED',
   'VERIFIED',
   'HOURS',
   'WRITE VERIFY TOTAL',
   DASHES)
   (COL(2), A,
    COL(11), A,
    COL(20), A,
    COL(29), A,
    COL(38), A,
    COL(48), A,
    COL(59), A,
    COL(71), A,
    COL(78), A,
    COL(1), A);
SUMMARY_LINES = 0;
END NEW_SUMMARY_PAGE;
END OPERATR;
NEWYEAR: PROCEDURE OPTIONS (MAIN);

/-------------------------------------------------------------------/
/ THE INITIALIZATION PROGRAM IS RUN ONLY AT THE BEGINNING OF THE FISCAL YEAR. THE INITIALIZATION PROGRAM INPUTS THE MASTER FILE AND OUTPUTS A NEW MASTER FILE WITH ZERO FOR ALL YEAR-TO-DATE TOTALS AND DELETES ANY SYSTEM WHICH HAD NO ACTIVITY DURING THE FISCAL YEAR JUST ENDED. A REPORT IS PRODUCED WHICH SHOWS ALL DELETED RECORDS. 

DCL TABLE_ENTRY,
  DEPT_NUMBER CHAR (2),
  SYSTEM_NUMBER CHAR (2),
  SUBSYS_NUMBER CHAR (2),
  FORMAT CHAR (4),
  TITLE CHAR (20),
  RECORD_LENGTH CHAR (3),
  AVERAGE CHAR (3),
  VERIFY CHAR (3),
  ESTIMATE CHAR (3),
  WRITE CHAR (3),
  VERIFY CHAR (3),
  FLAGS CHAR (2),
  EQUIPMENT CHAR (1),
  CODED CHAR (1),
  DIFFICULTY CHAR (1),
  SCHEDULE CHAR (2),
  RECORDS CHAR (3),
  WRITTEN CHAR (10),
  VERIFIED CHAR (10),
  CORRECTED CHAR (10),
  HOURS CHAR (10),
  WRITTEN CHAR (10),
  VERIFIED CHAR (10),

DCL END_OF_FILE FIXED BINARY (10),
  HEADER CHAR (2) INIT ('00'),
  FISCAL_YEAR CHAR (10),
  RETURN_CODE FIXED BINARY (31,0) INIT (0),
  TEMPI FLOAT DECIMAL (10),
  TEMP2 FLOAT DECIMAL (10),
  LINES FIXED BINARY INIT (10),
  PAGE FIXED BINARY INIT (0),
  MAX_LINES FIXED BINARY INIT (40),
  CURRENT_DATE CHAR (8),
  SEPARATOR CHAR (1) INIT ('-'),
  DASHES CHAR (132) INIT ((132)'-'),
  BLANK CHAR (1) INIT (' ');

OPEN FILE (SYSPRINT) OUTPUT STREAM PRINT LINESIZE(132),
  FILE (SYSIN ) INPUT STREAM,
  FILE (MASTER ) INPUT RECORD,
  FILE (NEWMSTR ) OUTPUT RECORD;
ON ENDFILE (MASTER ) END_OF_FILE = 1;
ON CONVERSION ONCHAR = '0';
GET FILE (SYSIN ) DATA (FISCAL_YEAR);
READ FILE (MASTER ) INTO (TABLE_ENTRY);
IF DEPT_NUMBER = HEADER THEN DO;
  SCHEDULE = BLANK;
  TITLE = FISCAL_YEAR;
  WRITE FILE (NEWMSTR ) FROM (TABLE_ENTRY);
END;
ELSE DO;
  PUT FILE (SYSPRINT)
    EDIT ('HEADER RECORD IS MISSING. '),
-------------------------------------------------------------------
'EXECUTION TERMINATED."
(A, A);
END;
RETURN_CODE = 8;
END;
CURRENT_DATE = DATE;
CURRENT_DATE = SUBSTR(CURRENT_DATE,3,2) || ':-' ||
SUBSTR(CURRENT_DATE,5,2) || ':-' ||
SUBSTR(CURRENT_DATE,1,2);  
INITIALIZE_NEW_YEAR:
DO WHILE (END_OF_FILE = 0);
READ FILE (MASTER ) INTO (TABLE_ENTRY);
TEMP1 = HOURS.WRITTEN;
TEMP2 = HOURS.VERIFIED;
IF END_OF_FILE = 0 &
(SYSTEM_NUMBER = BLANK |
TEMP1 > 0.0 |
TEMP2 > 0.0) THEN
INITIALIZE_ENTRY:
DO;
RECORDS.WRITTEN,
RECORDS.VERIFIED,
RECORDS.CORRECTED,
HOURS.WRITTEN,
HOURS.VERIFIED = BLANK ;
WRITE FILE (NEWSTR ) FROM (TABLE_ENTRY);
END INITIALIZE_ENTRY;
ELSE
DELETE_ENTRY: DO ;
IF LINES > MAX_LINES THEN
CALL NEW_PAGE ;
PUT FILE"(SYSPRINT)
EDIT (TABLE_ENTRY.DEPT_NUMBER,
TABLE_ENTRY.SYSTEM_NUMBER,
SEPARATOR,
TABLE_ENTRY.SUBSYS_NUMBER,
TABLE_ENTRY.FORMAT,
TABLE_ENTRY.TITLE,
TABLE_ENTRY.RECORD_LENGTH,
TABLE_ENTRY.AVERAGE_WRITE,
TABLE_ENTRY.AVERAGE_VERIFY,
TABLE_ENTRY_ESTIMATE_WRITE,
TABLE_ENTRY_ESTIMATE_VERIFY,
TABLE_ENTRY_FLAGS,
TABLE_ENTRY.RECORDS.WRITTEN,
TABLE_ENTRY.RECORDS.VERIFIED,
TABLE_ENTRY.HOURS.WRITTEN,
TABLE_ENTRY.HOURS.VERIFIED )
(SKIP(2),
COL(1), A, A, A, A,
COL(10), A,
COL(16), A,
COL(39), A,
COL(45), A, X(1), A,
COL(55), A, X(1), A,
COL(65), A, A, A, A,
COL(72), A,
COL(84), A,
COL(96), A,
COL(108), A ) ;
LINES = LINES + 2;
END DELETE_ENTRY;
END INITIALIZE_NEW_YEAR ;
CALL IHESARC(RETURN_CODE);

NEW_PAGE: PROCEDURE ;
THIS PROCEDURE OUTPUTS THE HEADER AND COLUMN HEADINGS ON EACH NEW PAGE FOR THE PURGE REPORT.

PAGE = PAGE + 1;
LINES = 0;
PUT FILE (SYSPRINT) PAGE
EDIT ('CITY OF LOS ANGELES',
'DATA SERVICE BUREAU',
'DATE ', CURRENT DATE,'DATA CONVERSION INITIALIZATION REPORT',
'PAGE ', PAGE,
DASHES,
'SYSTEM',
'EQUIP')
(LINE(6), COL(70), A,
COL(70), A,
COL(10), A, A,
COL(61), A,
COL(110), A, F(4),
SKIP(2), COL(1), A,
COL(1), A,
COL(65), A) ;

PUT FILE (SYSPRINT)
EDIT ('SUB-SYS',
'RECORD',
'AVERAGE ESTIMATING',
'CODED',
'RECORDS',
'RECORDS',
'HOURS',
'HOURS')
(COL(1), A,
COL(37), A,
COL(45), A,
COL(65), A,
COL(74), A,
COL(86), A,
COL(99), A,
COL(111), A ) ;

PUT FILE (SYSPRINT)
EDIT ('NUMBER',
'FORMAT',
'T I T L E',
'LENGTH',
'WRT VER WRT VER',
'DIFF.',
'WRITTEN',
'VERIFIED',
'WRITTEN',
'VERIFIED',
DASHES )
(COL(1), A,
COL(9), A,
COL(21), A,
COL(37), A,
COL(45), A,
COL(65), A,
COL(74), A,
COL(85), A,
COL(98), A,
COL(109), A,
COL(1), A) ;

END NEW_PAGE;
END NEWYEAR ;
APPENDIX C

Sample Reports
<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>SUB-SYS</th>
<th>RECORD</th>
<th>AVERAGE LENGTH</th>
<th>ESTIMATING CODED</th>
<th>EQUIP RECORDS</th>
<th>RECORDS VERIFIED</th>
<th>HOURS WRITTEN</th>
<th>HOURS VERIFIED</th>
<th>TRANSACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>5740-10</td>
<td>C</td>
<td>858999</td>
<td>11</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MODIFY</td>
</tr>
</tbody>
</table>
CITY OF LOS ANGELES
DATA SERVICE BUREAU
DATA CONVERSION UPDATE REPORT

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>SUB-SYS NUMBER</th>
<th>FORMAT</th>
<th>TITLE</th>
<th>RECORD LENGTH WRT</th>
<th>AVERAGE ESTIMATING</th>
<th>EQUIP CODED</th>
<th>RECORDS WRITTEN</th>
<th>RECORDS VERIFIED</th>
<th>HOURS WRITTEN</th>
<th>HOURS VERIFIED</th>
<th>TRANSACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>5740-10</td>
<td>C</td>
<td>BI-WKLY TRANS GS/FS</td>
<td>7 780 118 13 9 T</td>
<td>153528</td>
<td>152167</td>
<td>184.7</td>
<td>131.3</td>
<td>MODIFY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERIOD</td>
<td>PERSONNEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CITY OF LOS ANGELES
DATA SERVICE BUREAU
DATA CONVERSION PERIOD REPORT 1981-1982
PAGE 7

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>SUB-SYS</th>
<th>FORMAT</th>
<th>JOB NAME</th>
<th>RECORDS</th>
<th>RECORDS</th>
<th>HOURS</th>
<th>HOURS</th>
<th>HOURS</th>
<th>HOURS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>WRITTEN</td>
<td>VERIFIED</td>
<td>WRITTEN</td>
<td>VERIFIED</td>
<td>WRITTEN</td>
<td>VERIFIED</td>
</tr>
<tr>
<td>1541-02</td>
<td>A</td>
<td>TELEPHONE</td>
<td>MEMO</td>
<td>792</td>
<td>175</td>
<td>13.0</td>
<td>10.4</td>
<td>23.4</td>
<td>22.5</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>HEADER</td>
<td>8</td>
<td>5</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>299</td>
<td>299</td>
<td>48.6</td>
<td>2.7</td>
<td>7.4</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>1550</td>
<td>1500</td>
<td>11.0</td>
<td>8.3</td>
<td>19.4</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>1317</td>
<td>1318</td>
<td>13.2</td>
<td>9.4</td>
<td>22.6</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>115</td>
<td>115</td>
<td>1.0</td>
<td>0.6</td>
<td>1.6</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>G</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>27</td>
<td>30</td>
<td>0.1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>MEDICAL</td>
<td>BILLS</td>
<td>126</td>
<td>126</td>
<td>1.0</td>
<td>1.1</td>
<td>2.1</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>I</td>
<td>MASTER</td>
<td>FILE</td>
<td>205</td>
<td>205</td>
<td>3.3</td>
<td>2.3</td>
<td>5.6</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>J</td>
<td>SICK OR</td>
<td>INJURY</td>
<td>REPT</td>
<td>2905</td>
<td>1902</td>
<td>43.8</td>
<td>31.8</td>
<td>75.6</td>
</tr>
</tbody>
</table>

DEPARTMENT PERIOD TOTALS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TOTALS</td>
<td>10151</td>
<td>9853</td>
<td>2394</td>
<td>149.2</td>
<td>106.9</td>
<td>256.1</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEPARTMENT</td>
<td>PAGE</td>
<td>RECORDS</td>
<td>RECORDS</td>
<td>RECORDS</td>
<td>HOURS</td>
<td>HOURS</td>
<td>HOURS</td>
<td>TOTAL</td>
<td>PERSONNEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACCOUNTING</td>
<td>17</td>
<td>13166</td>
<td>14511</td>
<td>1188</td>
<td>56.2</td>
<td>62.4</td>
<td>98.8</td>
<td>1.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUILDING AND SAFETY</td>
<td>3</td>
<td>8994</td>
<td>8990</td>
<td>1328</td>
<td>71.8</td>
<td>52.9</td>
<td>124.7</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAS</td>
<td>4</td>
<td>161</td>
<td>161</td>
<td>0</td>
<td>0.8</td>
<td>0.6</td>
<td>1.4</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CITY ATTORNEY</td>
<td>2</td>
<td>24707</td>
<td>24294</td>
<td>3267</td>
<td>121.1</td>
<td>40.2</td>
<td>221.3</td>
<td>2.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CITY CLERK</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMMUNITY DEVELOPMENT</td>
<td>1</td>
<td>38977</td>
<td>38181</td>
<td>2592</td>
<td>143.3</td>
<td>125.9</td>
<td>271.2</td>
<td>3.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>8</td>
<td>242597</td>
<td>33972</td>
<td>11295</td>
<td>781.2</td>
<td>691.9</td>
<td>1473.2</td>
<td>16.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA SERVICE BUREAU</td>
<td>18</td>
<td>9556</td>
<td>9536</td>
<td>21</td>
<td>32.2</td>
<td>25.5</td>
<td>57.7</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENGINEERING</td>
<td>10</td>
<td>3768</td>
<td>3999</td>
<td>36</td>
<td>26.3</td>
<td>13.7</td>
<td>40.0</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIRE DEPARTMENT</td>
<td>11</td>
<td>15449</td>
<td>15193</td>
<td>770</td>
<td>42.7</td>
<td>34.9</td>
<td>77.6</td>
<td>0.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLEET SERVICES</td>
<td>24</td>
<td>36756</td>
<td>36388</td>
<td>831</td>
<td>50.4</td>
<td>38.5</td>
<td>88.9</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIBRARY</td>
<td>13</td>
<td>987</td>
<td>985</td>
<td>11</td>
<td>1.3</td>
<td>1.4</td>
<td>2.7</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MULTI DEPTS</td>
<td>29</td>
<td>2651</td>
<td>2653</td>
<td>276</td>
<td>15.9</td>
<td>13.5</td>
<td>29.4</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P &amp; T</td>
<td>16</td>
<td>5772</td>
<td>5678</td>
<td>412</td>
<td>22.1</td>
<td>17.1</td>
<td>39.2</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PENSION</td>
<td>12</td>
<td>15941</td>
<td>16642</td>
<td>256</td>
<td>24.1</td>
<td>15.8</td>
<td>40.0</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERSONNEL</td>
<td>1</td>
<td>18151</td>
<td>18253</td>
<td>2394</td>
<td>149.2</td>
<td>106.9</td>
<td>256.1</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLANNING</td>
<td>14</td>
<td>4366</td>
<td>4468</td>
<td>55</td>
<td>13.6</td>
<td>8.0</td>
<td>21.7</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLICE</td>
<td>15</td>
<td>97683</td>
<td>92825</td>
<td>3453</td>
<td>256.7</td>
<td>200.1</td>
<td>456.8</td>
<td>4.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PUBLIC BUILDINGS</td>
<td>19</td>
<td>16229</td>
<td>19539</td>
<td>648</td>
<td>51.1</td>
<td>40.2</td>
<td>91.3</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PURCHASING</td>
<td>26</td>
<td>1892</td>
<td>1887</td>
<td>159</td>
<td>15.9</td>
<td>11.3</td>
<td>27.2</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RECREATION AND PARKS</td>
<td>25</td>
<td>6177</td>
<td>5877</td>
<td>422</td>
<td>33.6</td>
<td>25.9</td>
<td>59.5</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RETIREMENT</td>
<td>6</td>
<td>2769</td>
<td>2529</td>
<td>369</td>
<td>21.3</td>
<td>17.0</td>
<td>38.4</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIGHT OF WAY/LAND</td>
<td>26</td>
<td>1350</td>
<td>1162</td>
<td>124</td>
<td>4.8</td>
<td>2.9</td>
<td>7.7</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SANITATION</td>
<td>21</td>
<td>32498</td>
<td>31765</td>
<td>2176</td>
<td>118.0</td>
<td>90.1</td>
<td>208.1</td>
<td>2.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STREET LIGHTING</td>
<td>22</td>
<td>5445</td>
<td>5385</td>
<td>290</td>
<td>21.2</td>
<td>17.3</td>
<td>38.5</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STREET MAINTENANCE</td>
<td>23</td>
<td>6226</td>
<td>6060</td>
<td>106</td>
<td>21.5</td>
<td>15.1</td>
<td>36.7</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRANSPORTATION</td>
<td>27</td>
<td>14798</td>
<td>13218</td>
<td>1580</td>
<td>76.5</td>
<td>59.6</td>
<td>136.1</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TREASURER</td>
<td>28</td>
<td>279</td>
<td>90</td>
<td>0</td>
<td>3.0</td>
<td>0.7</td>
<td>3.7</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CARD TOTALS FOR PERIOD 9: 12780 9181 826 40.8 123.4 1.0
KEY-TO-DISK TOTALS FOR PERIOD 9: 598259 694357 33593 2868.1 1735.1 3565.2 43.6

TOTALS FOR PERIOD 9: 611029 705458 33593 2150.6 1775.9 3926.5 44.6
<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>PERIOD</th>
<th>SUB-SYS</th>
<th>RECORDS</th>
<th>RECORDS</th>
<th>COR- HOURS</th>
<th>HOURS</th>
<th>TOTAL HOURS</th>
<th>JOB NAME</th>
<th>EFFICIENCY RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>8590-61</td>
<td>9</td>
<td>BUILDING PERMITS</td>
<td>8297</td>
<td>9377</td>
<td>0</td>
<td>27.8</td>
<td>27.5</td>
<td>55.3</td>
<td>All</td>
</tr>
<tr>
<td>8590-61 A</td>
<td>Building Permits</td>
<td>26189</td>
<td>26276</td>
<td>7757</td>
<td>358.1</td>
<td>262.7</td>
<td>620.8</td>
<td>0</td>
<td>55</td>
</tr>
<tr>
<td>8590-61 B</td>
<td>BLDG. PERMITS</td>
<td>1842</td>
<td>1791</td>
<td>412</td>
<td>24.9</td>
<td>14.1</td>
<td>38.0</td>
<td>0.1</td>
<td>52</td>
</tr>
<tr>
<td>8590-61 C</td>
<td>COMPLETION CARDS</td>
<td>16426</td>
<td>16195</td>
<td>1006</td>
<td>63.9</td>
<td>54.0</td>
<td>117.9</td>
<td>0.2</td>
<td>25</td>
</tr>
<tr>
<td>8590-61 D</td>
<td>BLDG/SAFETY BOILERS</td>
<td>729</td>
<td>524</td>
<td>0</td>
<td>3.3</td>
<td>2.7</td>
<td>6.0</td>
<td>0</td>
<td>80</td>
</tr>
<tr>
<td>8590-61 E</td>
<td>BLDG/SAFETY BOILERS</td>
<td>4771</td>
<td>4633</td>
<td>846</td>
<td>42.6</td>
<td>28.0</td>
<td>70.6</td>
<td>0.1</td>
<td>47</td>
</tr>
<tr>
<td>8590-61 F</td>
<td>BLDG/SAFETY BOILERS</td>
<td>4668</td>
<td>4675</td>
<td>102</td>
<td>10.6</td>
<td>8.0</td>
<td>18.6</td>
<td>0.0</td>
<td>12</td>
</tr>
<tr>
<td>8590-61 G</td>
<td>BLDG/SAFETY BOILERS</td>
<td>561</td>
<td>561</td>
<td>5</td>
<td>1.7</td>
<td>0.8</td>
<td>2.5</td>
<td>0.0</td>
<td>12</td>
</tr>
<tr>
<td>8590-62</td>
<td>BLDG/SAFETY ELEVATOR</td>
<td>7887</td>
<td>6723</td>
<td>0</td>
<td>37.1</td>
<td>29.9</td>
<td>67.0</td>
<td>0.1</td>
<td>80</td>
</tr>
<tr>
<td>8590-62 A</td>
<td>BLDG/SAFETY ELEVATOR</td>
<td>3529</td>
<td>3444</td>
<td>85</td>
<td>30.3</td>
<td>22.5</td>
<td>52.8</td>
<td>0.1</td>
<td>47</td>
</tr>
<tr>
<td>8590-62 B</td>
<td>BLDG/SAFETY ELEVATOR</td>
<td>3469</td>
<td>3467</td>
<td>99</td>
<td>6.9</td>
<td>5.8</td>
<td>12.7</td>
<td>0.0</td>
<td>13</td>
</tr>
<tr>
<td>8590-62 C</td>
<td>BLDG/SAFETY ELEVATOR</td>
<td>340</td>
<td>334</td>
<td>3</td>
<td>0.0</td>
<td>0.7</td>
<td>1.3</td>
<td>0.0</td>
<td>12</td>
</tr>
<tr>
<td>8590-70</td>
<td>BLDG/SAFETY ELEVATOR</td>
<td>27</td>
<td>27</td>
<td>0</td>
<td>0.5</td>
<td>0.6</td>
<td>1.1</td>
<td>0.0</td>
<td>0</td>
</tr>
</tbody>
</table>

DEPARTMENT YEAR-TO-DATE TOTALS 78346 78647 11108 608.5 459.3 1067.8 1.4
### CITY OF LOS ANGELES
DATA SERVICE BUREAU

#### PERIOD 9 INDEX TO DEPARTMENTS

<table>
<thead>
<tr>
<th>DEPARTMENT</th>
<th>PAGE</th>
<th>RECORDS</th>
<th>VERIFIED</th>
<th>CORRECTED</th>
<th>HOURS</th>
<th>VERIFIED</th>
<th>TOTAL</th>
<th>PERSONNEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCOUNTING</td>
<td>22</td>
<td>129800</td>
<td>126826</td>
<td>9855</td>
<td>620.6</td>
<td>468.0</td>
<td>1088.2</td>
<td>1.7</td>
</tr>
<tr>
<td>ANIMAL REGULATION</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.9</td>
<td>0.0</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>BUILDING AND SAFETY</td>
<td>4</td>
<td>78346</td>
<td>78647</td>
<td>11018</td>
<td>688.5</td>
<td>459.3</td>
<td>1147.8</td>
<td>1.4</td>
</tr>
<tr>
<td>CAO</td>
<td>5</td>
<td>23673</td>
<td>23533</td>
<td>1606</td>
<td>85.9</td>
<td>79.0</td>
<td>164.9</td>
<td>0.2</td>
</tr>
<tr>
<td>CHARITABLE ORG</td>
<td>30</td>
<td>52058</td>
<td>4653</td>
<td>0</td>
<td>21.4</td>
<td>12.1</td>
<td>34.2</td>
<td>0.5</td>
</tr>
<tr>
<td>CITY ATTORNEY</td>
<td>30</td>
<td>119691</td>
<td>201111</td>
<td>20372</td>
<td>978.6</td>
<td>810.1</td>
<td>1788.7</td>
<td>2.3</td>
</tr>
<tr>
<td>CITY CLERK</td>
<td>6</td>
<td>3393</td>
<td>3468</td>
<td>0</td>
<td>17.3</td>
<td>12.0</td>
<td>29.3</td>
<td>0.6</td>
</tr>
<tr>
<td>COMMUNITY DEVE/DEPT</td>
<td>1</td>
<td>55745</td>
<td>53495</td>
<td>3277</td>
<td>366.2</td>
<td>128.5</td>
<td>494.7</td>
<td>1.1</td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>9</td>
<td>2324911</td>
<td>2913166</td>
<td>184135</td>
<td>782.1</td>
<td>679.5</td>
<td>14619.8</td>
<td>18.6</td>
</tr>
<tr>
<td>DATA SERVICE BUREAU</td>
<td>11</td>
<td>386487</td>
<td>85976</td>
<td>5461</td>
<td>300.2</td>
<td>234.1</td>
<td>534.3</td>
<td>0.7</td>
</tr>
<tr>
<td>ENGINEERING</td>
<td>21</td>
<td>41134</td>
<td>40874</td>
<td>463</td>
<td>388.9</td>
<td>124.9</td>
<td>513.8</td>
<td>0.8</td>
</tr>
<tr>
<td>FIRE DEPARTMENT</td>
<td>12</td>
<td>116125</td>
<td>111287</td>
<td>7199</td>
<td>409.8</td>
<td>327.9</td>
<td>737.7</td>
<td>0.9</td>
</tr>
<tr>
<td>FLEET SERVICES</td>
<td>28</td>
<td>146734</td>
<td>144116</td>
<td>3959</td>
<td>305.7</td>
<td>232.2</td>
<td>537.9</td>
<td>0.7</td>
</tr>
<tr>
<td>LIBRARY</td>
<td>15</td>
<td>12902</td>
<td>12913</td>
<td>128</td>
<td>27.7</td>
<td>24.1</td>
<td>51.8</td>
<td>0.1</td>
</tr>
<tr>
<td>MAYOR</td>
<td>13</td>
<td>18592</td>
<td>1895</td>
<td>627</td>
<td>29.6</td>
<td>22.9</td>
<td>52.5</td>
<td>0.1</td>
</tr>
<tr>
<td>MULTI DEPTS</td>
<td>34</td>
<td>25912</td>
<td>25749</td>
<td>2260</td>
<td>149.2</td>
<td>113.6</td>
<td>262.8</td>
<td>0.5</td>
</tr>
<tr>
<td>P U &amp; T</td>
<td>19</td>
<td>38081</td>
<td>38314</td>
<td>2294</td>
<td>165.2</td>
<td>124.2</td>
<td>289.4</td>
<td>0.4</td>
</tr>
<tr>
<td>PENSION</td>
<td>14</td>
<td>22770</td>
<td>22779</td>
<td>1344</td>
<td>124.1</td>
<td>184.8</td>
<td>308.9</td>
<td>0.5</td>
</tr>
<tr>
<td>PERSONNEL</td>
<td>8</td>
<td>112097</td>
<td>114599</td>
<td>2096</td>
<td>1348.0</td>
<td>1630.5</td>
<td>2370.5</td>
<td>3.0</td>
</tr>
<tr>
<td>PLANNING</td>
<td>16</td>
<td>68529</td>
<td>68610</td>
<td>421</td>
<td>140.8</td>
<td>78.7</td>
<td>219.5</td>
<td>0.3</td>
</tr>
<tr>
<td>POLICE</td>
<td>17</td>
<td>617672</td>
<td>61842</td>
<td>29509</td>
<td>2066.9</td>
<td>1675.1</td>
<td>3740.6</td>
<td>4.6</td>
</tr>
<tr>
<td>PUBLIC BUILDINGS</td>
<td>22</td>
<td>136436</td>
<td>141876</td>
<td>5197</td>
<td>554.4</td>
<td>323.2</td>
<td>877.6</td>
<td>1.0</td>
</tr>
<tr>
<td>PURCHASING</td>
<td>31</td>
<td>601486</td>
<td>596977</td>
<td>25284</td>
<td>1607.2</td>
<td>1450.6</td>
<td>3058.8</td>
<td>3.9</td>
</tr>
<tr>
<td>RECREATION AND PARKS</td>
<td>29</td>
<td>78904</td>
<td>78907</td>
<td>6209</td>
<td>414.5</td>
<td>315.1</td>
<td>729.6</td>
<td>0.9</td>
</tr>
<tr>
<td>RETIREMENT</td>
<td>7</td>
<td>16466</td>
<td>15746</td>
<td>2009</td>
<td>112.2</td>
<td>99.7</td>
<td>211.9</td>
<td>0.3</td>
</tr>
<tr>
<td>RIGHT OF WAY/LAND</td>
<td>23</td>
<td>7662</td>
<td>6388</td>
<td>555</td>
<td>26.0</td>
<td>17.0</td>
<td>43.0</td>
<td>0.1</td>
</tr>
<tr>
<td>SANITATION</td>
<td>29</td>
<td>188336</td>
<td>187954</td>
<td>1172</td>
<td>708.6</td>
<td>649.9</td>
<td>1358.3</td>
<td>1.8</td>
</tr>
<tr>
<td>STANDARDS</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>STREET LIGHTING</td>
<td>26</td>
<td>39022</td>
<td>38559</td>
<td>2233</td>
<td>134.9</td>
<td>117.7</td>
<td>252.6</td>
<td>0.4</td>
</tr>
<tr>
<td>STREET MAINTENANCE</td>
<td>27</td>
<td>60459</td>
<td>57411</td>
<td>1129</td>
<td>168.5</td>
<td>115.7</td>
<td>284.2</td>
<td>0.4</td>
</tr>
<tr>
<td>TRANSPORTATION</td>
<td>22</td>
<td>123468</td>
<td>123468</td>
<td>1651</td>
<td>157.9</td>
<td>121.3</td>
<td>279.2</td>
<td>0.4</td>
</tr>
<tr>
<td>TREASURER</td>
<td>33</td>
<td>9760</td>
<td>8386</td>
<td>96</td>
<td>69.1</td>
<td>41.7</td>
<td>110.8</td>
<td>0.1</td>
</tr>
</tbody>
</table>

#### CARD YEAR-TO-DATE TOTALS FOR PERIOD 9
106787

#### KEY-TO-DISK YEAR-TO-DATE TOTALS FOR PERIOD 9
5166580

#### YEAR-TO-DATE TOTALS FOR PERIOD 9
5273367
<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>SUB-SYS</th>
<th>FORMAT</th>
<th>JOB NAME</th>
<th>BATCH</th>
<th>STATUS</th>
<th>RECORDS</th>
<th>OPEN DATE</th>
<th>OPEN</th>
<th>CLOSE ELAP</th>
<th>CORRECT</th>
<th>RECORD</th>
<th>AVERAGE</th>
<th>EFFICIENCY</th>
<th>RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1900-01</td>
<td>O</td>
<td>TIMESHEETS DAILY</td>
<td>WRITE</td>
<td>0410</td>
<td>Records</td>
<td>60</td>
<td>02-18-82</td>
<td>1511</td>
<td>1517</td>
<td>6</td>
<td>0</td>
<td>19</td>
<td>249</td>
<td>313</td>
</tr>
<tr>
<td>1900-01</td>
<td>O</td>
<td>TIMESHEETS DAILY</td>
<td>WRITE</td>
<td>0408</td>
<td>Records</td>
<td>79</td>
<td>02-18-82</td>
<td>1518</td>
<td>1526</td>
<td>6</td>
<td>0</td>
<td>19</td>
<td>249</td>
<td>313</td>
</tr>
<tr>
<td>1900-01</td>
<td>O</td>
<td>MOU RETRO PAY</td>
<td>WRITE</td>
<td>0076</td>
<td>Records</td>
<td>16</td>
<td>02-19-82</td>
<td>0717</td>
<td>0729</td>
<td>12</td>
<td>0</td>
<td>28</td>
<td>166</td>
<td>197</td>
</tr>
<tr>
<td>1900-01</td>
<td>O</td>
<td>MOU RETRO PAY</td>
<td>VERIFY</td>
<td>0065</td>
<td>Records</td>
<td>68</td>
<td>02-19-82</td>
<td>0730</td>
<td>0739</td>
<td>9</td>
<td>0</td>
<td>28</td>
<td>166</td>
<td>197</td>
</tr>
<tr>
<td>1900-01</td>
<td>O</td>
<td>MOU RETRO PAY</td>
<td>WRITE</td>
<td>0065</td>
<td>Records</td>
<td>72</td>
<td>02-19-82</td>
<td>0742</td>
<td>0804</td>
<td>22</td>
<td>0</td>
<td>28</td>
<td>166</td>
<td>197</td>
</tr>
<tr>
<td>5740-10</td>
<td>C</td>
<td>WEEKLY CREW TMSHTS</td>
<td>VERIFY</td>
<td>0354</td>
<td>Records</td>
<td>279</td>
<td>02-19-82</td>
<td>0806</td>
<td>0819</td>
<td>13</td>
<td>5</td>
<td>7</td>
<td>78</td>
<td>118</td>
</tr>
<tr>
<td>1900-01</td>
<td>N</td>
<td>WEEKLY CREW TMSHTS</td>
<td>WRITE</td>
<td>0427</td>
<td>Records</td>
<td>90</td>
<td>02-19-82</td>
<td>0826</td>
<td>0848</td>
<td>14</td>
<td>0</td>
<td>31</td>
<td>193</td>
<td>264</td>
</tr>
<tr>
<td>1900-01</td>
<td>N</td>
<td>WEEKLY CREW TMSHTS</td>
<td>WRITE</td>
<td>0425</td>
<td>Records</td>
<td>57</td>
<td>02-19-82</td>
<td>0858</td>
<td>0908</td>
<td>18</td>
<td>0</td>
<td>31</td>
<td>193</td>
<td>264</td>
</tr>
<tr>
<td>1900-01</td>
<td>N</td>
<td>WEEKLY CREW TMSHTS</td>
<td>WRITE</td>
<td>0425</td>
<td>Records</td>
<td>38</td>
<td>02-19-82</td>
<td>0916</td>
<td>0916</td>
<td>6</td>
<td>0</td>
<td>31</td>
<td>193</td>
<td>264</td>
</tr>
<tr>
<td>1900-01</td>
<td>N</td>
<td>WEEKLY CREW TMSHTS</td>
<td>VERIFY</td>
<td>0425</td>
<td>Records</td>
<td>31</td>
<td>02-19-82</td>
<td>0934</td>
<td>0942</td>
<td>6</td>
<td>0</td>
<td>31</td>
<td>193</td>
<td>264</td>
</tr>
<tr>
<td>1900-01</td>
<td>N</td>
<td>WEEKLY CREW TMSHTS</td>
<td>WRITE</td>
<td>0425</td>
<td>Records</td>
<td>20</td>
<td>02-19-82</td>
<td>1014</td>
<td>1018</td>
<td>4</td>
<td>0</td>
<td>31</td>
<td>193</td>
<td>264</td>
</tr>
<tr>
<td>1541-02</td>
<td>E</td>
<td>MED PMS LINE 0-5</td>
<td>WRITE</td>
<td>0276</td>
<td>Records</td>
<td>47</td>
<td>02-19-82</td>
<td>1029</td>
<td>1035</td>
<td>15</td>
<td>0</td>
<td>85</td>
<td>85</td>
<td>106</td>
</tr>
<tr>
<td>1541-02</td>
<td>E</td>
<td>MED PMS LINE 0-5</td>
<td>VERIFY</td>
<td>0277</td>
<td>Records</td>
<td>46</td>
<td>02-19-82</td>
<td>1034</td>
<td>1034</td>
<td>18</td>
<td>0</td>
<td>85</td>
<td>85</td>
<td>106</td>
</tr>
<tr>
<td>1541-02</td>
<td>E</td>
<td>MED PMS LINE 0-5</td>
<td>VERIFY</td>
<td>0265</td>
<td>Records</td>
<td>47</td>
<td>02-19-82</td>
<td>1105</td>
<td>1127</td>
<td>22</td>
<td>11</td>
<td>85</td>
<td>85</td>
<td>106</td>
</tr>
<tr>
<td>1541-02</td>
<td>H</td>
<td>MED PMS LINE 7</td>
<td>VERIFY</td>
<td>0261</td>
<td>Records</td>
<td>36</td>
<td>02-19-82</td>
<td>1134</td>
<td>1138</td>
<td>6</td>
<td>1</td>
<td>81</td>
<td>105</td>
<td>119</td>
</tr>
<tr>
<td>1541-02</td>
<td>E</td>
<td>MED PMS LINE 0-5</td>
<td>WRITE</td>
<td>0274</td>
<td>Records</td>
<td>24</td>
<td>02-19-82</td>
<td>1139</td>
<td>1148</td>
<td>9</td>
<td>0</td>
<td>85</td>
<td>85</td>
<td>106</td>
</tr>
<tr>
<td>1541-02</td>
<td>E</td>
<td>MED PMS LINE 0-5</td>
<td>WRITE</td>
<td>0274</td>
<td>Records</td>
<td>24</td>
<td>02-19-82</td>
<td>1151</td>
<td>1207</td>
<td>16</td>
<td>0</td>
<td>85</td>
<td>85</td>
<td>106</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>CAMIS</td>
<td>WRITE</td>
<td>0485</td>
<td>Records</td>
<td>75</td>
<td>02-19-82</td>
<td>1215</td>
<td>1237</td>
<td>22</td>
<td>0</td>
<td>27</td>
<td>176</td>
<td>176</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>WORKSHEETS</td>
<td>WRITE</td>
<td>0285</td>
<td>Records</td>
<td>123</td>
<td>02-19-82</td>
<td>1321</td>
<td>1404</td>
<td>43</td>
<td>0</td>
<td>37</td>
<td>121</td>
<td>171</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>CAMIS</td>
<td>WRITE</td>
<td>0466</td>
<td>Records</td>
<td>46</td>
<td>02-19-82</td>
<td>1457</td>
<td>1448</td>
<td>11</td>
<td>0</td>
<td>27</td>
<td>174</td>
<td>176</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>CAMIS</td>
<td>WRITE</td>
<td>0466</td>
<td>Records</td>
<td>9</td>
<td>02-19-82</td>
<td>1501</td>
<td>1505</td>
<td>2</td>
<td>0</td>
<td>27</td>
<td>174</td>
<td>176</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>CAMIS</td>
<td>WRITE</td>
<td>0487</td>
<td>Records</td>
<td>75</td>
<td>02-19-82</td>
<td>1563</td>
<td>1564</td>
<td>13</td>
<td>0</td>
<td>27</td>
<td>174</td>
<td>176</td>
</tr>
<tr>
<td>0420-01</td>
<td>C</td>
<td>CAMIS</td>
<td>WRITE</td>
<td>0494</td>
<td>Records</td>
<td>45</td>
<td>02-19-82</td>
<td>1517</td>
<td>1525</td>
<td>8</td>
<td>3</td>
<td>27</td>
<td>174</td>
<td>176</td>
</tr>
</tbody>
</table>

OPERATOR PERIOD TOTALS - WRITE: 3335 13.3 0 147
OPERATOR PERIOD TOTALS - VERIFY: 2483 4.4 128
OVERALL EFFICIENCY RATING - 204
<table>
<thead>
<tr>
<th>OPERATOR NUMBER</th>
<th>RECORDS NUMBER</th>
<th>ERRORS WRITTEN</th>
<th>HOURS VERIFIED</th>
<th>RECORDS CORRECTED</th>
<th>HOURS WRITE</th>
<th>TOTAL HOURS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
<th>TOTAL ERRORS</th>
</tr>
</thead>
<tbody>
<tr>
<td>337</td>
<td>3264</td>
<td>0</td>
<td>9.5</td>
<td>1389</td>
<td>66</td>
<td>4.8</td>
<td>14.3</td>
<td>121</td>
<td>153</td>
<td>132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>338</td>
<td>3586</td>
<td>0</td>
<td>15.2</td>
<td>0</td>
<td>0</td>
<td>2.0</td>
<td>15.2</td>
<td>95</td>
<td>95</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>374</td>
<td>4641</td>
<td>0</td>
<td>4.3</td>
<td>2353</td>
<td>143</td>
<td>7.5</td>
<td>11.0</td>
<td>151</td>
<td>173</td>
<td>166</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>377</td>
<td>2678</td>
<td>0</td>
<td>7.0</td>
<td>4768</td>
<td>286</td>
<td>10.3</td>
<td>17.2</td>
<td>128</td>
<td>250</td>
<td>203</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>3306</td>
<td>0</td>
<td>12.9</td>
<td>1989</td>
<td>135</td>
<td>7.5</td>
<td>20.4</td>
<td>154</td>
<td>175</td>
<td>163</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>407</td>
<td>564</td>
<td>0</td>
<td>3.5</td>
<td>4044</td>
<td>265</td>
<td>10.8</td>
<td>14.3</td>
<td>163</td>
<td>300</td>
<td>264</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>411</td>
<td>7683</td>
<td>0</td>
<td>25.7</td>
<td>0</td>
<td>0</td>
<td>25.7</td>
<td>0</td>
<td>102</td>
<td>0</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>413</td>
<td>3568</td>
<td>0</td>
<td>8.3</td>
<td>4156</td>
<td>408</td>
<td>11.6</td>
<td>20.0</td>
<td>164</td>
<td>236</td>
<td>202</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>429</td>
<td>2612</td>
<td>0</td>
<td>8.0</td>
<td>3970</td>
<td>204</td>
<td>7.2</td>
<td>15.2</td>
<td>120</td>
<td>385</td>
<td>248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>456</td>
<td>2741</td>
<td>0</td>
<td>5.0</td>
<td>9479</td>
<td>293</td>
<td>11.0</td>
<td>16.0</td>
<td>196</td>
<td>150</td>
<td>165</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>458</td>
<td>4804</td>
<td>0</td>
<td>9.9</td>
<td>3499</td>
<td>127</td>
<td>8.3</td>
<td>18.3</td>
<td>145</td>
<td>145</td>
<td>145</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>463</td>
<td>3250</td>
<td>0</td>
<td>8.5</td>
<td>2779</td>
<td>92</td>
<td>10.1</td>
<td>18.6</td>
<td>85</td>
<td>87</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Summary Operator Analysis Report 1981-1982

<table>
<thead>
<tr>
<th></th>
<th>OPERATOR RECORDS</th>
<th>ERRORS</th>
<th>HOURS</th>
<th>RECORDS</th>
<th>ERRORS</th>
<th>HOURS</th>
<th>TOTAL EFFICIENCY</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>35171</td>
<td>0</td>
<td>86.4</td>
<td>34153</td>
<td>0</td>
<td>1779</td>
<td>77.0</td>
</tr>
<tr>
<td>1</td>
<td>26331</td>
<td>0</td>
<td>87.9</td>
<td>56389</td>
<td>0</td>
<td>2100</td>
<td>112.1</td>
</tr>
<tr>
<td>2</td>
<td>29866</td>
<td>0</td>
<td>95.6</td>
<td>24068</td>
<td>0</td>
<td>740</td>
<td>62.5</td>
</tr>
<tr>
<td>3</td>
<td>29040</td>
<td>0</td>
<td>104.6</td>
<td>30951</td>
<td>0</td>
<td>1604</td>
<td>80.1</td>
</tr>
<tr>
<td>4</td>
<td>28528</td>
<td>0</td>
<td>81.9</td>
<td>29916</td>
<td>0</td>
<td>1524</td>
<td>66.6</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

### Groups

<table>
<thead>
<tr>
<th></th>
<th>Number Written</th>
<th>Written Corrected</th>
<th>Written Verified</th>
<th>HOURS</th>
<th>WRITE</th>
<th>VERIFY</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>35171</td>
<td>34153</td>
<td>0</td>
<td>1779</td>
<td>77.0</td>
<td>157</td>
<td>151</td>
</tr>
<tr>
<td>1</td>
<td>26331</td>
<td>56389</td>
<td>0</td>
<td>2100</td>
<td>112.1</td>
<td>127</td>
<td>152</td>
</tr>
<tr>
<td>2</td>
<td>29866</td>
<td>24068</td>
<td>0</td>
<td>740</td>
<td>62.5</td>
<td>113</td>
<td>124</td>
</tr>
<tr>
<td>3</td>
<td>29040</td>
<td>30951</td>
<td>0</td>
<td>1604</td>
<td>80.1</td>
<td>132</td>
<td>142</td>
</tr>
<tr>
<td>4</td>
<td>28528</td>
<td>29916</td>
<td>0</td>
<td>1524</td>
<td>66.6</td>
<td>129</td>
<td>203</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Shifts

<table>
<thead>
<tr>
<th></th>
<th>Number Written</th>
<th>Written Corrected</th>
<th>Written Verified</th>
<th>HOURS</th>
<th>WRITE</th>
<th>VERIFY</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>143876</td>
<td>175477</td>
<td>12542</td>
<td>398.3</td>
<td>18.2</td>
<td>126</td>
<td>143</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Overall

<table>
<thead>
<tr>
<th></th>
<th>Number Written</th>
<th>Written Corrected</th>
<th>Written Verified</th>
<th>HOURS</th>
<th>WRITE</th>
<th>VERIFY</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>143876</td>
<td>456.4</td>
<td>175477</td>
<td>12542</td>
<td>398.3</td>
<td>18.2</td>
<td>126</td>
<td>143</td>
</tr>
<tr>
<td>SYSTEM SUB-SYS NUMBER</td>
<td>FORMAT</td>
<td>TITLE</td>
<td>RECORD LENGTH</td>
<td>AVERAGE LENGTH</td>
<td>ESTIMATING CODED</td>
<td>HOURS WRITTEN</td>
<td>HOURS VERIFIED</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------</td>
<td>------------------------</td>
<td>---------------</td>
<td>----------------</td>
<td>------------------</td>
<td>---------------</td>
<td>----------------</td>
</tr>
<tr>
<td>0100-01</td>
<td></td>
<td>ONE TIME TRACS DATE</td>
<td>0 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0710-03</td>
<td></td>
<td></td>
<td>80 0 0</td>
<td></td>
<td>C 1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0750-05 D</td>
<td></td>
<td>CAO WK ASSN REVISION</td>
<td>43 0 0</td>
<td></td>
<td>T 2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0750-05 E</td>
<td></td>
<td>CAO WK UNIT INPUT</td>
<td>8 0 0</td>
<td></td>
<td>T 9</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0981-01</td>
<td></td>
<td>SR CITIZEN EXP RFND</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0991-99</td>
<td></td>
<td></td>
<td>80 0 0</td>
<td></td>
<td>C 2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1335-01</td>
<td></td>
<td>DELETS</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1510-01 B</td>
<td></td>
<td>POOL LIFEGUARD EXAM</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1541-02</td>
<td></td>
<td>WORKERS COMP MED PAY</td>
<td>79 104 104</td>
<td>TN</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 CC</td>
<td></td>
<td>DIRECT DEPOSITS</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 FF</td>
<td></td>
<td>W2 ADDRESS CHANGE</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 GO</td>
<td></td>
<td>W2 SDI-FICA</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 HH</td>
<td></td>
<td>W2 NAME CHANGE</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 II</td>
<td></td>
<td>CANCEL CHK PROCESS</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 RR</td>
<td></td>
<td>BONUS ADJ.</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 SS</td>
<td></td>
<td>UNEMPLOYMENT INS.</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 UU</td>
<td></td>
<td>ADJ CANCEL EMER CHECKS</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 VV</td>
<td></td>
<td>UNEMPLOYMENT INS</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-01 ZZ</td>
<td></td>
<td>POLICE OT YEARLY</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-10 A</td>
<td></td>
<td>UPDATE TABLE SC</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1900-10 B</td>
<td></td>
<td>UPDATE TABLE VC</td>
<td>0 0</td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
APPENDIX D

User Manual
Update Process

I. Purpose

The purpose of the update process is to apply changes to the master file. Three transactions are provided: ADD, MODIFY, and DELETE to add a record, change a record, or delete a record, respectively.

The update process is a sequence of two steps. The first step is an edit step in order to edit the transaction records for any errors. The second step is to update the master file by applying the transaction records to the master file. For convenience, the transaction records can be entered in any order. The process will sort the edited records before attempting to update the master file.

II. Use

A. Restrictions

The update process can be run as often and as many times with as many transaction records as desired. However, only one update processing must be running at any point in time and none of the other processes which use the master file can be run at the same time as the update process. If any other process is run with an update process, the master file will either fail to reflect the results of one of the processes or the master file may become unusable.
B. Inputs and Outputs

There are two inputs and three outputs. The two inputs are the update transaction records and the master file. The three outputs are the updated master file, the edit report, and the update report.

The update transaction record input format appears in the table on the next page.

Transaction and system subsystem are required for each transaction. If any of the other fields are blank, no change will occur to the field for a MODIFY transaction and will be blank or zero for an ADD transaction. The transaction field, the application number, and the format are necessary for all transactions and are the only fields which are required or used for a DELETE transaction.

Error messages are included in the edit report and update report but are included here for easy reference. If no edit errors are found in a transaction, the message "NO ERRORS FOUND" is printed in the edit report after the transaction. If any edit errors are found, the message "***ERRORS FOUND***" is printed followed by one or more field names where an error was detected, such as: "TRANSACTION CODE", "SYSTEM-SUBSYSTEM"
# Update Transaction Record

<table>
<thead>
<tr>
<th>Field No.</th>
<th>Field Name</th>
<th>Start Column</th>
<th>Length</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Transaction</td>
<td>1</td>
<td>6</td>
<td>Must be present; ADD MODIFY or DELETE</td>
</tr>
<tr>
<td>2</td>
<td>Application Number</td>
<td>7</td>
<td>6</td>
<td>Must be present, numeric</td>
</tr>
<tr>
<td>3</td>
<td>Format</td>
<td>13</td>
<td>4</td>
<td>Form</td>
</tr>
<tr>
<td>4</td>
<td>Title</td>
<td>17</td>
<td>20</td>
<td>Application title</td>
</tr>
<tr>
<td>5</td>
<td>Record Length</td>
<td>37</td>
<td>3</td>
<td>Average record length, numeric</td>
</tr>
<tr>
<td>6</td>
<td>Average write</td>
<td>40</td>
<td>3</td>
<td>Average number of records written per hour, numeric</td>
</tr>
<tr>
<td>7</td>
<td>Average verify</td>
<td>43</td>
<td>3</td>
<td>Average number of records verified per hour, numeric</td>
</tr>
<tr>
<td>8</td>
<td>Reserved</td>
<td>46</td>
<td>6</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>9</td>
<td>Media</td>
<td>52</td>
<td>1</td>
<td>Key to disk or Tape (T), Cards (C)</td>
</tr>
<tr>
<td>10</td>
<td>Coded</td>
<td>53</td>
<td>1</td>
<td>Whether or not the data is coded in the department (Y) or (N)</td>
</tr>
<tr>
<td>11</td>
<td>Reserved</td>
<td>54</td>
<td>27</td>
<td>Reserved for future use</td>
</tr>
</tbody>
</table>
Only two errors can occur from the update step. The first is trying to add an already existing master file record. The error message is "***ERROR*** ATTEMPT TO ADD AN EXISTING ENTRY." The second is trying to modify or delete a master file record which does not exist. The error message is "***ERROR*** ATTEMPT TO (DELETE or MODIFY) a NONEXISTENT ENTRY."

All transactions which are not in error, update the master file. In the case of any error, either edit or update, the master file is not affected by that transaction. The transactions in error must be corrected and may be reprocessed, if desired.

Period and Year-to-Date Process

I. Purpose

The purpose of this process is to produce a history file of data records, update the master file, and produce the period report and the year-to-date report.

II. Use

A. Restrictions

This process should not be run at the same time as the update process. An unusable or
inaccurate master file could result. Although not a restriction, best results are obtained when run in time periods of equal durations, such as an organization's accounting period. If the process is not run by such a time period, the reports may need to be interpolated to obtain meaningful workload results.

B. Inputs and Outputs

The inputs are the data record file and the master file. The outputs are an updated master file, a period report, and a year-to-date report.

The updated master file contains year-to-date totals for records written, records verified, records corrected, hours written, hours verified for each application number and format. The other fields in the master file are not modified by this process. These fields are modified by use of the update process.

Operator Analysis Process

I. Purpose

Although this process could be contained in the period and year-to-date process, experience has taught that quick feedback is necessary to make optimum use of the operator analysis reports. Therefore, a separate process is included which may be run more frequently than every accounting period. This allows
supervisors to detect problems as quickly as is desired, which will in turn permit a solution to a problem to be implemented before the organization incurs an unacceptable amount of damage.

II. Use

A. Restrictions

No restrictions are recommended. This process could be run at the same time as any other process without any unfavorable results except the update process. In fact, it can be run as often as is deemed necessary.

B. Inputs and Outputs

The operator analysis process uses the master file and the data record file as inputs and produces the detail operator analysis report and the summary operator analysis report. Because this process is not necessarily run in an accounting period, the period and fiscal year contained in the master file can be overridden with keyword input. In order to replace these, the keywords PERIOD= and FISCAL_YEAR= are used with the desired overrides enclosed in single quote marks. The desired overrides will appear in the reports in place of the period and fiscal year. An example would be: PERIOD='5/25 - 5/31, FISCAL_YEAR='80-81';
This would replace the period and fiscal year contained in the master file with 5/25 - 5/31 and 80-81, respectively. Notice that a comma or a space or both must separate the keywords and a semicolon must be used to signal termination of the overrides. The period data may be from 0 to 15 characters in length and the fiscal year data may be 0 to 9 characters in length. The fields will be padded with blanks. A zero length is provided by two successive quote marks and will result in all blanks.

Initialization Process

I. Purpose

The initialization process is used to purge the master file of unused entries. This process is run only at the beginning of each fiscal year and any application number and format which has not had any keying activity during the previous fiscal year is deleted from the master file.

II. Use

A. Restrictions

This process cannot be run at the same time as any other process or the master file may become unusable or inaccurate.
B. Inputs and Outputs

The primary input and output is the master file and a new master file. The fiscal year must be provided as input as a keyword parameter. An example is:

FISCAL_YEAR='1980-1981';

The data is enclosed within single quotes and terminated by a semicolon. This fiscal year will be used to replace the fiscal year contained in the master file and will appear in the new master file.

For convenience, the initialization process has been concatenated by the period and year-to-date process to be run for the first accounting period of the year. Therefore, it is not necessary to run two separate processes at the end of the first accounting period.