EPA-450/3-74-045

JULY 1974

COMPREHENSIVE DATA HANDLING SYSTEM, AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS-II) PROGRAM DOCUMENTATION AND USER'S GUIDE

U.S. ENVIRONMENTAL PROTECTION AGENCY
Office of Air and Waste Management
Office of Air Quality Planning and Standards
Research Triangle Park, North Carolina 27711

EPA-450/3-74-045

COMPREHENSIVE DATA HANDLING SYSTEM, AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS-II) PROGRAM DOCUMENTATION AND USER'S GUIDE

by

The IBM Corporation
Federal Systems Division
18100 Frederick Pike
Gaithersburg, Maryland 20760

Contract No. 68-02-0638

EPA Project Officer: Lloyd Hedgepeth

Prepared for

ENVIRONMENTAL PROTECTION AGENCY
Office of Air and Waste Management
Office of Air Quality Planning and Standards
Research Triangle Park, N. C. 27711

July 1974

This report is issued by the Environmental Protection Agency to report technical data of interest to a limited number of readers. Copies are available free of charge to Federal employees, current contractors and grantees, and nonprofit organizations as supplies permit from the Air Pollution Technical Information Center, Environmental Protection Agency, Research Triangle Park, North Carolina 27711; or, for a fee, from the National Technical Information Service, 5285 Port Royal Road, Springfield, Virginia 22151.

This report was furnished to the Environmental Protection Agency by the IBM Corporation, Gaithersburg, Maryland 20760, in fulfillment of Contract No. 68-02-0638. The contents of this report are reproduced herein as received from the IBM Corporation. The opinions, findings, and conclusions expressed are those of the author and not necessarily those of the Environmental Protection Agency. Mention of company or product names is not to be considered as an endorsement by the Environmental Protection Agency.

Publication No. EPA-450/3-74-045

TABLE OF CONTENTS

FIGURES

PREFACE

This version of the Air Quality Data Handling Subsystem (AQDHS) of the Comprehensive Data Handling System (CDHS) is a major revision of and completely replaces the version of AQDHS described in APTD-1086. Conversion programs have been provided to assist the user in his transfer from the old AQDHS to the new AQDHS. Unless specifically mentioned, any occurrence of the term AQDHS refers to this version of AQDHS and not the previous version.

All of the programs in this subsystem have been written in ANS COBOL with the exception of one program. The Data Analysis program has been written in ANS FORTRAN for the Level G IBM FORTRAN-IV compiler.

1.0 INTRODUCTION

This document is aimed at two different audiences. Its primary target is the person who will be using the Air Quality Data Handling Subsystem (AQDHS) of the Comprehensive Data Handling System (CDHS). By referring to the general program write-ups, input card formats, and cataloged procedures, the AQDHS user should be able to use the system without reference to detailed program documentation. However, sufficient detailed documentation is provided for the programmer responsible for the maintenance of AQDHS. This documentation takes the form of flow diagrams, organization, data formats and subroutines for each program in AQDHS, as well as several appendices. With these goals in mind, the document is organized in the following fashion:

- Section 2.0 - Contains an overview of AQDHS, a brief functional description of each program comprising the system, and a detailed discussion of the system master file and the transactions required to build the master file.

- Sections 3.0 through 7.0 - Contain detailed functional descriptions of each program in AQDHS, along with complete instructions on the use of all program functions. Following these are the descriptions of program logic, organization, data formats and subroutines for each program. The sections are organized as follows:

  Section 3.0 - Conversion programs
  Section 4.0 - Table Maintenance programs
  Section 5.0 - File Maintenance and Transaction Editor programs
  Section 6.0 - Data Retrieval programs
  Section 7.0 - Output programs

In general, each program in AQDHS is organized in the top-down manner in which higher level programming modules execute one or more lower level modules to perform specific functional tasks. These lower level modules may in turn execute still lower level modules to perform other specific tasks. Each module has one entry and one exit only. Thus, each program basically consists of a hierarchical structure of programming modules.

In addition, each program in AQDHS is programmed using structured programming techniques. These techniques include using only three basic types of programming construction blocks (IF/ELSE, DO and PROCESS) and preclude the use of explicit branches. These techniques, along with the top-down organization, make for programs which are extremely readable with straightforward logic and no branching to confuse the program flow. This also enhances the maintainability of the programs which comprise the system.

Because of the advanced manner in which AQDHS was developed, it became obvious that the standard method of program documentation, instruction by instruction and field by field, would not provide the level of information about the programs necessary for valid understanding. Therefore, the AQDHS programs have been documented by first describing their hierarchical top-down structure and then giving detailed descriptions of each main module within the structure. Thus, the serious AQDHS user can find his way directly into virtually any subroutine of a given program where he will then find the programming details in the structured code.

In addition, descriptions of all important data areas and all crucial high level modules within each program have been provided as well as hierarchical flow charts of each program.

2.0 AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS) OVERVIEW

When dealing with atmospheric pollution, it is necessary to amass, catalog, sort, evaluate, and perform calculations upon large volumes of data. The Air Quality Data Handling Subsystem of the Comprehensive Data Handling System provides a systematic method for collecting this data in a data base that will provide a central source for the information needed to help control air pollution. If the system is to be helpful, it must maintain the data base, keeping the information current, and provide a means for access to the information, presenting it in a usable form.

The Air Quality Data Handling Subsystem provides the ability to create and maintain, and to retrieve and print data from the data base. The creation and maintenance is accomplished with the File Maintenance program. This program allows the user to keep his data base information current and useful. Access to the data base information is provided by the Retrieval program set. These programs provide the means of extracting desired information from the data base. The output print programs are then used to convert the extracted information to a form readable by the user. These three functions form the basic system.

In addition to the basic system, several preprocessor and postprocessor programs are provided which perform functions necessary to make this system compatible with existing systems. All of the system programs are described in detail in the following sections.

2.1 ORGANIZATION

AQDHS is organized around two main programs, File Maintenance and Retrieval. There are fourteen other programs in the system that perform service functions. The interface for the programs that feed the File Maintenance program is the AQDHS transaction card. The AQDHS master file serves as the interface for the other programs.

The AQDHS components are:

- Old AQDHS File to New AQDHS Input - Converts existing AQDHS files to new AQDHS transactions for building a new AQDHS master file.

- Old AQDHS (SAROAD) Input to New AQDHS Input - Converts old AQDHS transactions (SAROAD format) to new format AQDHS transactions.

- Parameter Code Table Maintenance - Maintains a table containing the valid combinations of parameter, method and unit codes along with their descriptions plus the minimum detectable value of the parameter.

- Site Code Table Maintenance - Maintains a table containing the valid combinations of state, area, site, agency and project codes along with a description of the site.

- Parameter Standards Table Maintenance - Maintains a table containing state and federal standards for parameters.

- Transaction Editor - Edits file maintenance transactions and converts them to an internal form usable by File Maintenance.

- File Maintenance - Creates and maintains the AQDHS master file.

- <u>Retrieval Language Processor</u> - Generates a COBOL program to retrieve records from the master file based on user specifications.

- <u>Retrieval</u> - Generated by the Retrieval Language Processor.

- <u>Detail List</u> - Provides a detailed formatted listing of the contents of the master file.

- <u>Sliding Average</u> - Formats each record in the master file and computes a sliding average of the readings contained therein.

- <u>Answer File Flagging</u> - Appends an end-of-file (EOF) sentinel record to the answer or master file for use by Data Analysis.

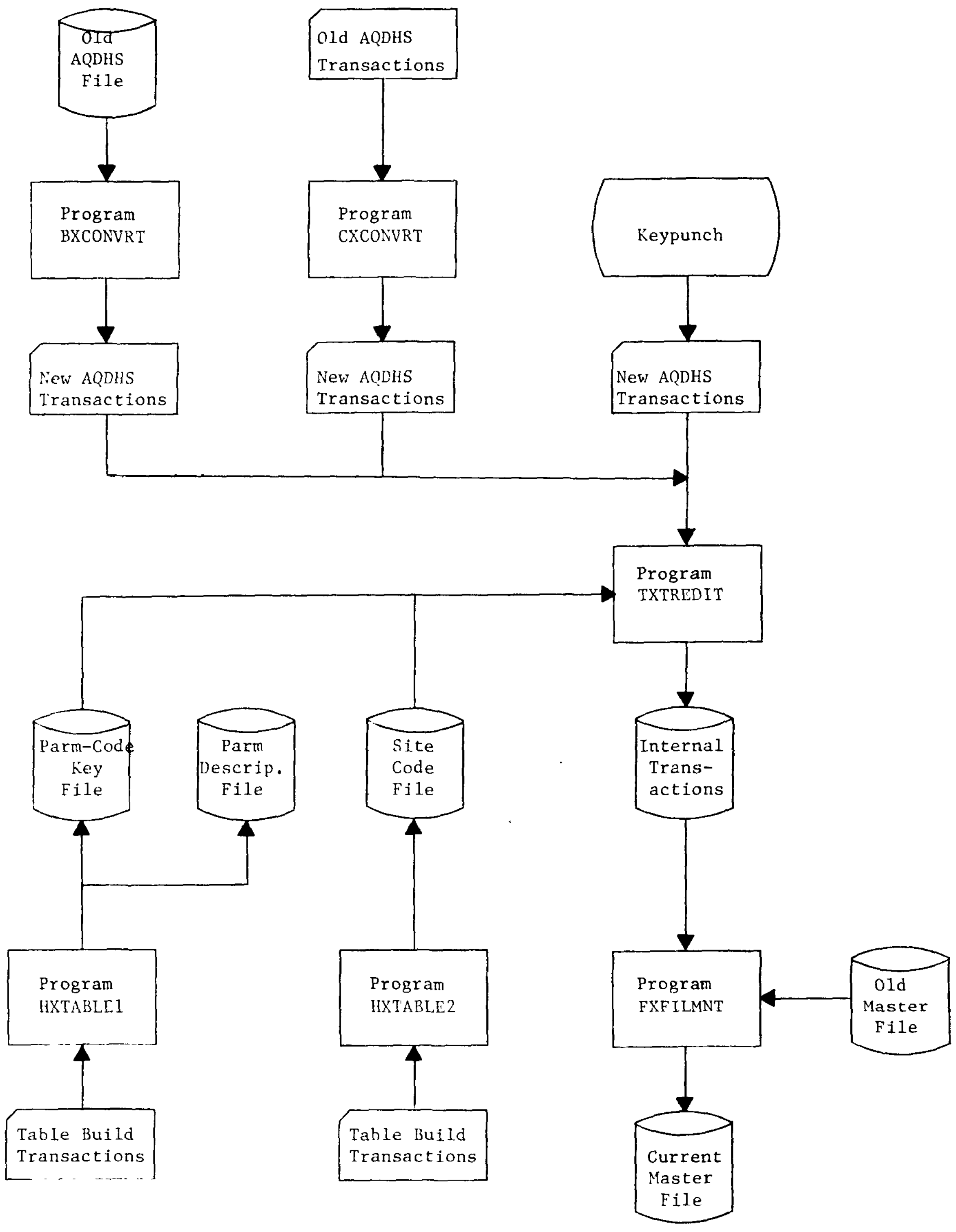
- <u>Parm-Code-Key File Flagging</u> - Appends an end-of-file (EOF) sentinel record to the key portion of the parameter, method, unit codes table file for use by Data Analysis.

- <u>Data Analysis</u> - Performs various statistical analyses on the readings in the master file.

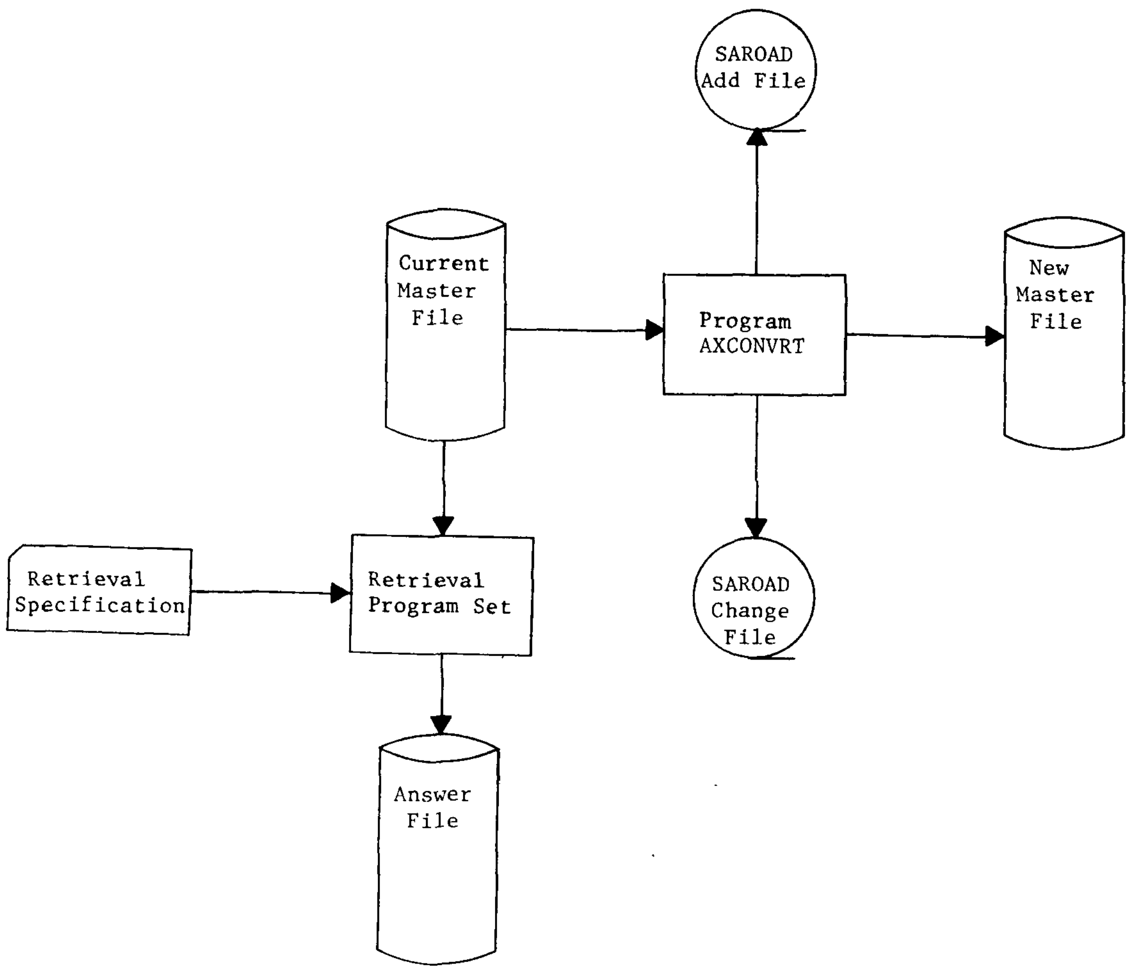
- <u>Statistical List</u> - Formats the results of the analyses performed by Data Analysis.

- <u>AQDHS File to SAROAD Input</u> - Extracts new and changed readings in the AQDHS file and generates magnetic tapes for submission to SAROAD.

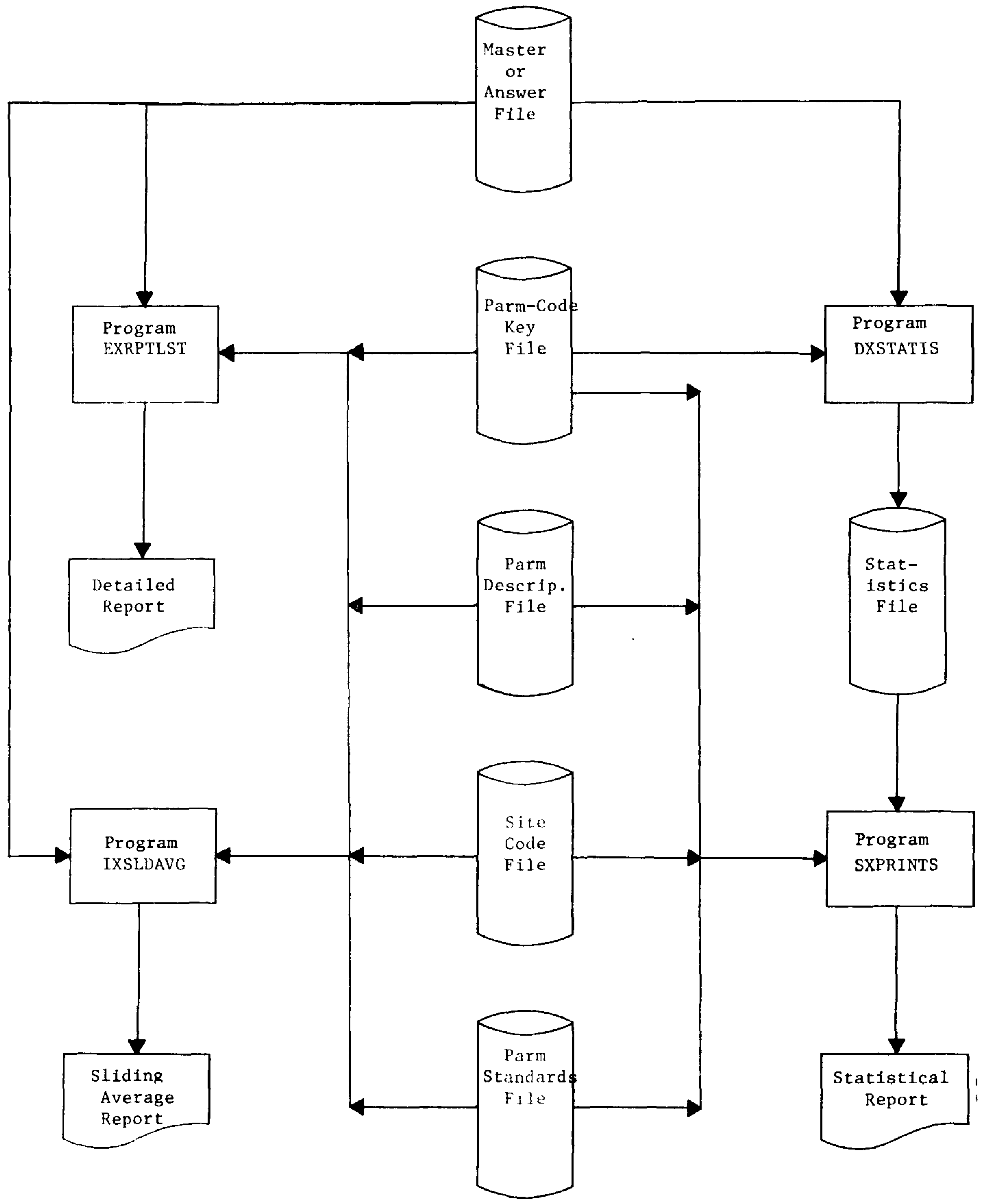
Figure 2.1.1 illustrates a possible data flow through AQDHS.



AQDHS System Data Flow
Figure 2.1.1



AQDHS System Data Flow
Figure 2.1.1 (continued)



AQDHS System Data Flow
Figure 2.1.1 (continued)

2.2 COMMUNICATION AND DATA FORMATS

2.2-1 MASTER FILE

The AQDHS master file record is designed to contain all data related to a particular parameter collected at a specific site. Each record represents a certain logical period of time--the length of the period being determined by the interval at which samples are taken. For any sampling interval less than 24 hours, the record represents one day's worth of data. Daily and weekly sampling intervals are contained in records representing one month's worth of data. Monthly and quarterly sampling intervals yield records representing a full year's worth of data. All records have the same format and are variable in length, with the length being determined by the number of readings actually stored in the record, not the maximum possible. For instance, a record for hourly intervals could hold a maximum of 24 readings. If readings 1, 3 and 6 were supplied when the record was created, the physical record would be 6 readings long, with reading 2, 4 and 5 filled with 9s to indicate a null value.

The AQDHS master file records will also store composite data. The composite readings are grouped in records representing a full year's worth of data with the exception of weekly composite data. Weekly composite data is stored one reading per record.

The format of the master file record is illustrated in Figure 2.2-1.1.

AQDHS Master Record

Position	Format	Symbol	Description
1	9	Numeric	Action Code
2	9	Numeric	Form Code
3 - 4	XX	Alphameric	State Code
5 - 8	9999	Numeric	Area Code
9 - 11	999	Numeric	Site
12	A	Alphabetic	Agency
13 - 14	99	Numeric	Project
15	X	Alphameric	Time Code
16 - 17	99	Numeric	Year
18 - 19	99	Numeric	Month
20 - 21	99	Numeric	Day
22 - 23	99	Numeric	Start Hour
24 - 28	99999	Numeric	Parameter Code
29 - 30	99	Numeric	Method Code
31 - 32	99	Numeric	Unit Code
33	9	Numeric	Decimal Code
34 - 35	99	Numeric	Number of Readi
36	A	Alphabetic	Status Flag
37 - 40	9999	Numeric	Data Field

Note: Positions 36 - 40 may be repeated to include more than one reading in the Master Record. The number of repetitions is determined by the value in the Number of Readings field.

AQDHS Master Record Format

Figure 2.2-1.1

2.2-1.1 DEFINITION OF AQDHS MASTER FILE RECORD FIELDS

Action Code: This code indicates the last action that was performed on this record. The action codes are two (2) for Add and three (3) for Change.

Form Code: The three SAROAD transmittal-form codes currently accepted by AQDHS are: form #1, less than 24-hour sampling interval; form #2, 24-hour or greater sampling interval; and form #3, multiple station form. The corresponding AQDHS forms use the same form codes. Form #2 is also used for composite data.

State Code: State names are arranged in alphbetic sequence and assigned numbers from 01 to 52. To maintain consistency, both the District of Columbia and the territory of Puerto Rico are considered as states.

Area Code: Within each state the names of all incorporated areas with a population of more than 2500 and all counties are arranged alphabetically and assigned a four-digit number. County codes are used only for stations located outside incorporated areas.

Site: Specific sampling sites are designated by a three-digit number that permits a maximum of 999 sites in each city or county area within a state. Users are advised to contact the National Aerometric Data Bank to obtain their site codes. The valid SAROAD site codes from NADB will be used by the edit program when the transaction cards are checked.

Agency: The type of agency responsible for the sampling is designated by a single, alphabetic code. The current agency codes are shown in Appendix A, Table 1.

Project: The project codes classify projects according to the purpose of the project under which the data is generated. The principal categories are shown in Appendix A, Table 2.

Time Code: A single-digit alphameric code. The time codes used by AQDHS to indicate sampling intervals are shown in Appendix A, Table 3.

Year: The AQDHS date validation check accepts years from 1960 to the current year.

Month: The AQDHS date validation check accepts months from 01 to 12.

Day: The AQDHS date validation check accepts days from 01 to the maximum for each month. The maximum for February is 29.

Start Hour: A two-digit numeric code that indicates the hour of the first reading. The range of hours is from 00 to 23.

Parameter Code: A five-digit numeric code which permits a branching sub-categorization of pollutants. A list of currently assigned codes may be found in EPA Publication No. APTD-0633; SAROAD Parameter Coding Manual.

Method Code: A two-digit numeric code designating the methods of collection and analysis. A list of valid codes may be obtained from EPA.

Unit Code: A two-digit numeric code used to designate the unit of measurement. A list of valid unit codes may be obtained from EPA. A partial list may be found in Appendix A, Table 4.

Decimal Code: A single-digit numeric code from 0 to 4 which indicates the number of digits in the data field that are to fall to the right of the decimal point.

Number of Readings: A two-digit numeric field that indicates the number of Data Fields and Status Flags that are in the record. This number is generated by the File Maintenance program and is used by the output programs.

Status Flag: This is a single-digit alphabetic code that indicates that the associated Data Field has been sent to SAROAD (value is 'S'), or is to be sent as an Add Record (value is 'A'), or Change Record (value is 'C'). The flag is set by the File Maintenance program and by the ADQHS File to SAROAD Input conversion program.

Data Field: The data fields contain the data as a four-digit number right-justified with leading-left zeros.

2.2-2 MASTER FILE MAINTENANCE TRANSACTIONS

The master file is constructed from the information contained on the AQDHS input transaction cards. These are three types of cards: Format 1, Format 2 and Format 3. The Format 1 transactions are used to enter readings taken at less than daily intervals. The Format 2 and Format 3 transactions are used to enter readings taken at daily or greater than daily intervals. Format 2 allows readings for multiple parameters to be entered while Format 3 allows readings from multiple stations to be entered. Format 2 transactions are also used to enter composite data.

Figure 2.2-2.1 illustrates the formats of the various transactions.

Format 1 AQDHS Transaction

Columns	Format	Symbol	Description
1	9	Numeric	Form Code
2-3	XX	Alphameric	State Code
4-7	9999	Numeric	Area Code
8-10	999	Numeric	Site Code
11	A	Alphabetic	Agency Code
12-13	99	Numeric	Project Code
14	9	Numeric	Time Code
15-16	99	Numeric	Year
17-18	99	Numeric	Month
19-20	99	Numeric	Day
21-22	99	Numeric	Start Hour
23-27	99999	Numeric	Parameter Code
28-29	99	Numeric	Method Code
30-31	99	Numeric	Units Code
32	9	Numeric	Decimal Position
33-36	9999	Numeric	Reading
37-40	9999	Numeric	Reading
41-44	9999	Numeric	Reading
45-48	9999	Numeric	Reading
49-52	9999	Numeric	Reading
53-56	9999	Numeric	Reading
57-60	9999	Numeric	Reading
61-64	9999	Numeric	Reading
65-78			Unused
79	A	Alphabetic	Status Flag
80	9	Numeric	Action Code

AQDHS File Maintenance Transactions

Figure 2.2-2.1

Format 2 AQDHS Transactions

Columns	Format	Symbol	Description
1	9	Numeric	Form Code
2-3	XX	Alphameric	State Code
4-7	9999	Numeric	Area Code
8-10	999	Numeric	Site Code
11	A	Alphabetic	Agency Code
12-13	99	Numeric	Project Code
14	X	Alphameric	Time Code
15-16	99	Numeric	Year
17-18	99	Numeric	Month
19-20	99	Numeric	Day
21-22	99	Numeric	Start Hour
23-27	99999	Numeric	Parameter Code
28-29	99	Numeric	Method Code
30-31	99	Numeric	Units Code
32	9	Numeric	Decimal Position
33-36	9999	Numeric	Reading
37-50			Repeat Columns 23-36
51-64			Repeat Columns 23-36
65-78			Repeat Columns 23-36
79	A	Alphabetic	Status Flag
80	9	Numeric	Action Code

AQDHS File Maintenance Transactions

Figure 2.2-2.1 (continued)

Format 2 AQDHS Composite Transaction

Columns	Format	Symbol	Description
17-18	99	Numeric	Period
19-20	99	Numeric	Number of Samples
21	9	Numeric	Composite Type
22	X	Alphameric	Time Code*

All other fields have the same format and meaning as those in the standard Format 2 transaction.

* Use SAROAD time code (APTD-0663 - Code Table 3) instead of AQDHS time code.

AQDHS File Maintenance Transactions
Figure 2.2-2.1 (continued)

Format 3 AQDHS Transaction

Columns	Format	Symbol	Description
1	9	Numeric	Form Code
2-3	XX	Alphameric	State Code
4	A	Alphabetic	Agency Code
5-6	99	Numeric	Project Code
7	X	Alphameric	Time Code
8-12	99999	Numeric	Parameter Code
13-14	99	Numeric	Method Code
15-16	99	Numeric	Units Code
17	9	Numeric	Decimal Position
18-19	99	Numeric	Year
20-21	99	Numeric	Month
22-23	99	Numeric	Day
24-25	99	Numeric	Start Hour
26-29	9999	Numeric	Area Code
30-32	999	Numeric	Site Code
33-36	9999	Numeric	Reading
37-49			Repeat Columns 24-36
50-62			Repeat Columns 24-36
63-75			Repeat Columns 24-36
76-78			Unused
79	A	Alphabetic	Status Flag
80	9	Numeric	Action Code

AQDHS File Maintenance Transactions

Figure 2.2-2.1 (continued)

2.2-2.1 DEFINITION OF AQDHS FILE MAINTENANCE TRANSACTION FIELDS

Action Code: Indicates the action to be performed by this transaction. Valid codes are:

1 - Delete

2 - Add

3 - Change

Agency Code: Identifies the agency responsible for these readings. Valid codes are found in Appendix A, Table 1.

Area Code: Identifies the area in which the sampling site is located.

Composite Period: Identifies the period during which the composite sample was taken. Valid codes are:

01 - 04	Quarterly and Seasonal Composite
01 - 12	Monthly Composite
01 - 52	Weekly Composite
00	Annual Composite

Refer to APTD - 0663 for a full discussion of composite data.

Composite Number of Samples: Indicates the number of individual samples that were composited.

Composite Time Code: Indicates the interval at which the individual composited samples were taken. This time code should be taken from Code Table 3 in APTD-0663 rather than from Appendix A, Table 3 in this document.

Composite Type: Indicates the interval at which the samples are composited. Valid codes are:

1 - Quarterly Composite

2 - Seasonal Composite

3 - Monthly Composite

4 - Weekly Composite

5 - Annual Composite

Day: The day of the month on which the sample was taken.

Decimal Position: A single-digit number from 0 to 4 which indicates the number of digits in the reading that are to fall to the right of the decimal point.

Form Code: The identification number of the form being used; 1, 2 or 3.

Method Code: Identifies both the collection method and the analysis method of the parameter being measured.

Month: The month of the year during which the sample was taken.

Parameter Code: Identifies the parameter being measured. Refer to APTD-0633 for a full list of currently accepted parameter codes.

Project Code: Identifies the project in association with which the sample was taken. Valid codes are found in Appendix A, Table 2.

Reading: The value of the sample taken.

Site Code: Identifies the site at which the sample was taken.

Start Hour: On the Format 2 and Format 3 transactions, the hour at which the sample was taken. On the Format 1 transaction, the hour at which the first reading was taken.

State: Indicates the state (or other geographic division) in which the sampling site is located.

Status Flag: Indicates the status of the readings on the transaction.

Valid codes are:

S	previously sent to SAROAD
blank	to be sent to SAROAD

Time Code: Indicates the interval at which the samples were taken. Valid codes are found in Appendix A, Table 3.

Units Code: Indicates the units in which the parameter was measured. A partial list may be found in Appendix A, Table 4.

Year: The year in which the sample was taken.

2.3 ROUTINES

Within each program in AQDHS, certain similarities of construction and format will immediately become apparent. The top level paragraph (the one with which execution begins) in each COBOL program is named ROOT-SEGMENT. This paragraph controls, on a gross level, the sequence of execution of all other paragraph in the program. ROOT-SEGMENT performs at least three other paragraphs in this order: PGM-INIT, MAIN-LOOP, WRAP-UP. Within ROOT-SEGMENT there may be other paragraphs performed or other READS, WRITES or switch settings to tailor ROOT-SEGMENT to the particular program, however, the three basic paragraphs remain.

- [illegible]
  This paragraph performs whatever program initialization functions are necessary. These generally consist of opening all files, clearing work areas and initializing switches.

- MAIN-LOOP
  This paragraph controls the main-line processing of the program. Usually, this consists of reading an input record and, based on that input, determining the action to be taken.

- WRAP-UP
  This paragraph performs whatever program termination functions are necessary. These usually consist of writing program execution statistics messages and closing all files.

Because the paragraphs ROOT-SEGMENT, PGM-INIT, MAIN-LOOP and WRAP-UP are found in each COBOL program in AQDHS and their functions are similar, the detailed program documenation usually begins with a description of MAIN-LOOP.

Figure 2.3.1 shows the general AQDHS program organization.

During the development of AQDHS, it became apparent that many programs had certain identical functions to perform. For example, each program needed a print routine to control the printing of any messages it may have. Whenever a function performed by more than one program could be identified and could be accomplished by identical code, a paragraph or group of paragraphs was developed to fulfill the functions. These functional modules could then be "plugged into" any program needing this function. The use of "global code" aided greatly in the reduction of coding time and effort during the development of AQDHS. Some of the global modules are identified below.

- <u>PRINT-ROUTINE</u>
  This routine controls the printing of any messages from the program to the user. It keeps track of the number of lines printed on the page and handles the printing of titles at page overflow.

- <u>BUILD-TABLES</u>
  This routine reads in the valid site code table file and the key portion of the parameter, method, unit codes table file and builds them as arrays in core storage.

- <u>BUILD-PARM-DESCRIPTION</u>
  This routine reads in the description portion of the parameter, method, unit codes table file and builds it as an array in core storage.

- <u>BUILD-STANDARDS-TABLE</u>
  This routine reads in the parameter standards table file and builds it as an array in core storage.

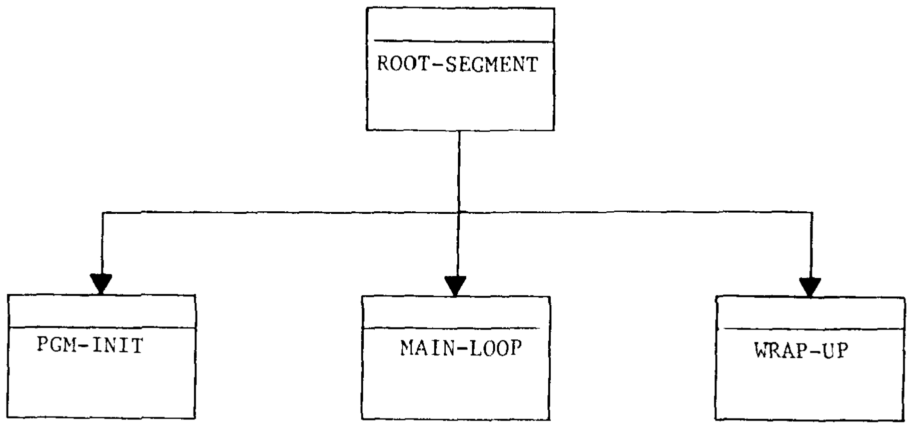
- <u>SEARCH-TABLES</u>
  This routine searches the valid site code table array and the parameter, method, unit codes table array and returns the subscripts of the correct entries.

- <u>LOCATE-STANDARD</u>

  This routine locates the appropriate state and federal standards for the parameter in question. Refer to section 4.3 for a discussion of the selection algorithm.

- <u>ERROR-ROUTINE</u>

  This routine handles the printing of error messages.



AQDHS - Organization
Figure 2.3.1

3.0 [illegible] .OGRAMS

AQDHS provides three conversio[illegible] [illegible]ng data into and extracting data from the AQDHS master file. Two provide conversi[illegible] of data for input to AQDHS. They are the Old AQDHS File to New AQDHS Input and the Old AQDHS (SAROAD) Input to New AQDHS Input conversion programs. The AQDHS File to SAROAD Input conversion program extracts data from the AQDHS master file for submission to SAROAD.

3.1 OLD AQDHS FILE TO NEW AQDHS INPUT

The Old AQDHS File to New AQDHS Input conversion program extracts the information contained in an old format AQDHS master file and converts it to the new AQDHS input format. These transactions may then be run through the Transaction Editor and File Maintenance programs to build a new format AQDHS master file.

All data in the old AQDHS master file should be sent to SAROAD before the conversion to the new AQDHS master file as this program sets the status flag to 'S', indicating that the data has been sent to SAROAD.

In order for this conversion program to work correctly, the format of the old AQDHS master file must correspond to the file description in APTD-1086. Any other file format will cause unpredictable results.

The name of the load module is BXCONVRT.

3.1.1 ORGANIZATION

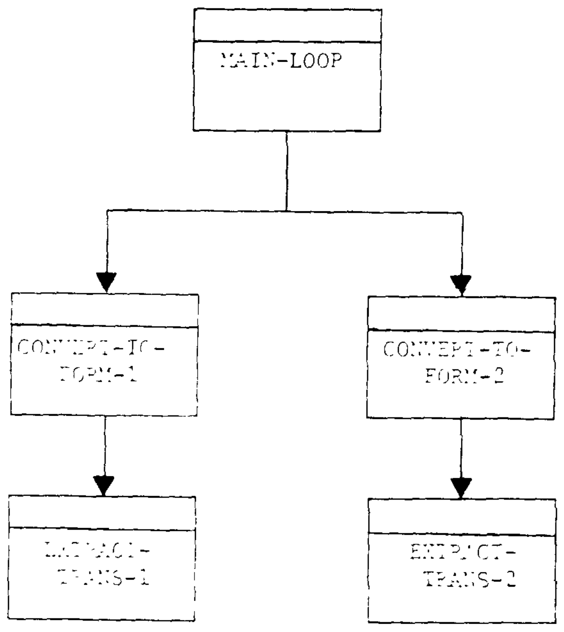
The Old AQDHS File to New AQDHS Input conversion program is organized in a top down, modular structure. There is only one entrance and one exit from the program, both contained in the highest level module. Each lower level module invoked has the same characteristics: one entrance and one exit.

- MAIN-LOOP
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the old AQDHS file. First the time codes are checked for form-1, form-2, or an invalid form number. If it is a form-1 (less than daily sampling interval), CONVERT-TO-FORM-1 is performed. If it is a form-2 then CONVERT-TO-FORM-2 is performed, otherwise an error message is printed. Finally READ-ROUTINE is performed which [illegible] next old AQDHS record.

- CONVERT-TO-FORM-1
  This routine builds a new AQDHS input record. First the data fields are moved. Then EXTRACT-TRANS-1 is performed which edits and moves the valid readings to the new record. If there are any readings in the new record then WRITE-ROUTINE is performed which writes a new AQDHS record.

- CONVERT-TO-FORM-2
  This routine builds a new AQDHS input record. First the data fields are moved. Then EXTRACT-TRANS-2 is performed which edits a specific data field and moves it to the new record; afterwards, WRITE-ROUTINE is performed.

Figure 3.1.1 shows the organization of BXCONVRT.



BXCONTRL - Organization
Figure 5.1.1

3.1.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

3.1.3 ROUTINES

The following are major subroutines of the Old AQDHS File to New AQDHS Input conversion program.

- PRINT-SAROAD-RECORD
  This routine reformats the old AQDHS file record and prints it out.

- READ-ROUTINE
  This routine reads an AQDHS record and performs PRINT-SAROAD-RECORD.

- WRITE-ROUTINE
  This routine writes a new AQDHS record.

3.2 OLD AQDHS (SAROAD) INPUT TO NEW AQDHS INPUT

This program accepts old AQDHS (SAROAD) format transactions and action cards and outputs new AQDHS format transactions. These transactions may then be used to create or update the AQDHS master file.

An action card indicates the action to be performed by all old AQDHS transactions following it until the next action card is encountered. Action cards are identified by a '$' in column 1 and an action code in column 2. There are three valid action codes: '1', '2', and '3'. '1' signals 'delete', '2' indicates 'add' and '3' means 'change'. A '$4' card is a status flag card and indicates that <u>all</u> old AQDHS transactions following it have been sent to SAROAD and the program begins generating a status flag of 'S' on the new AQDHS output transactions. This status flag code does not override the action code currently in effect (1, 2 or 3) nor can it be reset for the rest of the execution of the program.

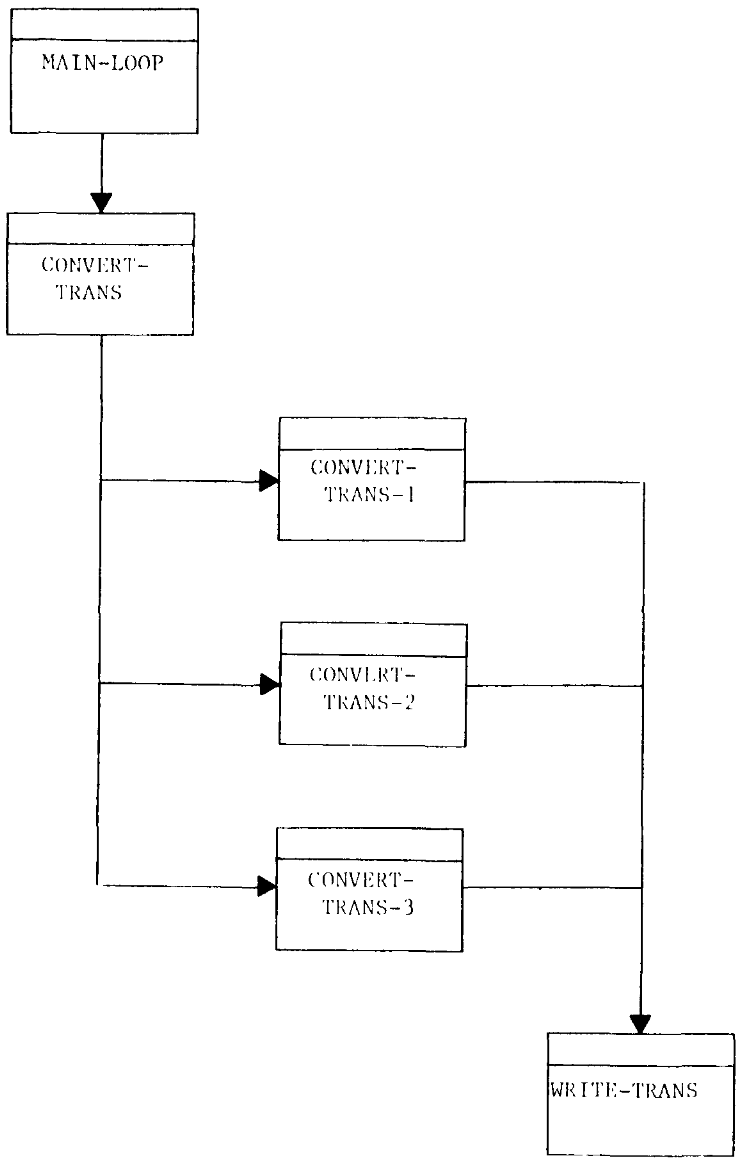
The name of the load module is CXCONVRT.

3.2.1 ORGANIZATION

The Old AQDHS (SAROAD) Input to New AQDHS Input conversion program is organized in a top down, modular structure. There is only one entrance and one exit from the program, both contained in the highest level module. Each lower level module invoked has the same characteristics: one entrance and one exit.

- MAIN-LOOP
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the input file. First the input record is tested for an action card and, if it is, the action code is saved. If it is a valid transaction, then CONVERT-TRANS is performed which branches to CONVERT-TRANS-1, CONVERT-TRANS-2, or CONVERT-TRANS-3 depending on the transaction code. Finally, READ-ROUTINE is performed which reads the next input record.

Figure 3.2.1 shows the organization of CXCONVRT.



CXCONVRT - Organization
Figure 3.2.1

3.2.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- SAROAD-STATUS-IS-SENT-SW
  May contain TRUE or FALSE. If true, the status of TRANS-STATUS-FLAG will be sent ('S').

- ACTION-CODE-SAVE
  Carries the value of the current action card in effect (other than 'sent').

3.2.3 <u>ROUTINES</u>

The following are major subroutines of the Old AQDHS (SAROAD) Input to New AQDHS Input conversion program.

- <u>CONVERT-TRANS</u>
  This routine edits the fields on the input record for valid data. If invalid, an appropriate error message is printed out.

- <u>READ-ROUTINE</u>
  This routine reads an input record until end-of-file. Then TRUE is moved to END-OF-FILE-SW.

- <u>SPLIT-TRANS-1</u>
  This routine splits the input transaction between two work areas and re-calculates the start hour in the second work area. If there is data in the work data fields, then WRITE-TRANS is performed.

- <u>WRITE-TRANS</u>
  This routine writes the output transactions and clears the transaction work area.

3.3 AQDHS FILE TO SAROAD INPUT

The AQDHS File to SAROAD Input conversion program provides the user with the capability of periodically extracting new or changed data from the AQDHS master file and sending it to SAROAD for inclusion in the National Aerometric Data Bank. It is still the user's responsibility, however, to notify the EPA Regional Office and the NADB when previously submitted data is deleted from the AQDHS master file.

All data which has been previously sent to SAROAD is not re-sent. All data which has been previously sent to SAROAD but has been changed is sent as a change transaction. All new data is sent as an add transaction. As each reading is sent to SAROAD, its associated status flag is changed from 'A' or 'C' to 'S', indicating that it has been sent. The master file output from this program contains the same information as the input master file except that all status flags are set to 'S'. This master file thereby becomes to current AQDHS master file.

There are two ouput files containing transactions to be sent to the NADB. One file will begin with a '$2' action card followed by all add transactions, if any. The other file will contain a '$3' action card and all change transactions, if any. The '$2' and '$3' cards indicate the action to be performed by the transactions following them, add or change.

The name of the load module is AXCONVRT.

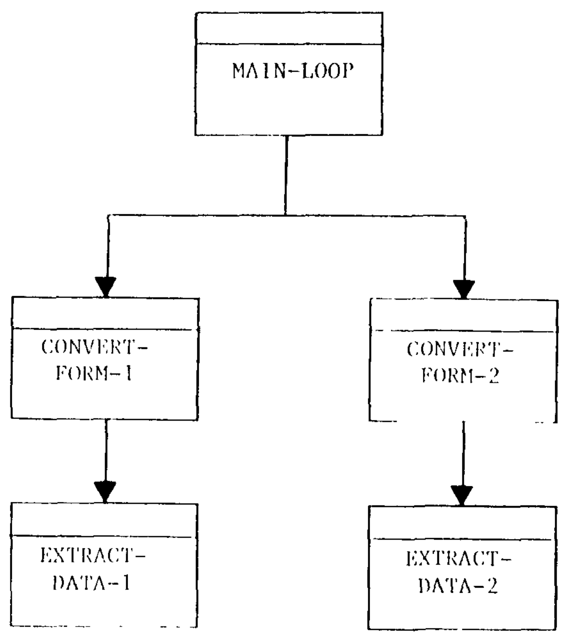
3.3.1 ORGANIZATION

The AQDHS File to SAROAD Input conversion program is organized in a top down, modular structure. There is only one entrance and one exit from the program, both contained in the highest level module. Each lower level module invoked has the same characteristics: one entrance and one exit.

- MAIN-LOOP
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the AQDHS master file. It determines the form code and invokes the appropriate conversion routine. If the record has any add or change fields, a record is written to the SAROAD add file or the SAROAD change file. Afterwards, a new master file record is written and an old master file record is read.

- CONVERT-FORM-1 and CONVERT-FORM-2
  These routines build the SAROAD transaction records. First the ident key is moved from the master record. Then CONVERT-TIME-CODE is performed to convert the AQDHS time code to SAROAD time code. Next EXTRACT-DATA is performed to build the SAROAD add or change fields of the SAROAD transaction record. If the add or change fields are built, a record is then written.

Figure 3.3.1 shows the organization of AXCONVRT.



AXCONVRT - Organization
Figure 3.3.1

3.3.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- TIME-CODE-LOCATED-SW
  May contain TRUE or FALSE. If true, a match from a master file record is equal to a time code in the time-code-table.

- TYPE-IS-ADD-SW
  May contain TRUE or FALSE. If true, a record is written to the add file.

- WRITE-TRANS-SW
  May contain TRUE or FALSE. If true, a Form-1 record is written.

3.3.3 <u>ROUTINES</u>

The following are major subroutines of the AQDHS File to SAROAD Input conversion program.

- <u>EXTRACT-DATA-1</u>
  This routine extracts readings and converts all the add or change flags in the master record to 'S', indicating that it has been sent to SAROAD.

- <u>EXTRACT-DATA-2</u>
  This routine extracts readings and converts all the add or change flags in the master record to 'S'. If the status is not 'S', the INCREMENT-BY-TIME-CODE routine is performed. Then a SAROAD add or change record is written.

- <u>INCREMENT-BY-TIME-CODE</u>
  This routine converts the time code from the master record to hour, day, or month for the SAROAD transaction record.

- <u>READ-ROUTINE</u>
  This routine writes a new master file record and then reads the next old master file record.

- <u>WRITE-ROUTINE</u>
  This routine writes a SAROAD add or change record and performs INCREMENT-BY-TIME-CODE.

4.0 TABLE MAINTENANCE

Three programs are provided by AQDHS to maintain tables necessary for editing and report formatting. They are the Parameter Code Table Maintenance, the Site Code Table Maintenance and the Parameter Standards Table Maintenance programs. As with File Maintenance, they are used to create and update their respective tables.

4.1 PARAMETER CODE TABLE MAINTENANCE

The Parameter Code Table Maintenance program creates and maintains a table containing the valid combinations of parameter code, analysis and collection methods code, units code and the minimum detectable value of the parameter.

The table is stored externally in two segments. The key portion of the table, containing the parameter codes, methods codes, units codes and minimum detectable values, is stored in one segment while the associated description portion is stored in the other segment. The description portion contains prose descriptions of the values contained in the key portion of the table.

The Parameter Code Table Maintenance program operates in one of three modes: ADD, CHANGE and DELETE. The transactions entered into the program determine the mode of operation.

Two transactions must be entered for each entry in the table. The table file is searched in an attempt to match the key of the transactions with the key of an entry in the table. If no match can be found, the program operates in ADD mode and adds the entry described by the transactions to the table. If a matching key is found, the description portions of the transactions are examined. If they are blank, the program operates in DELETE mode and deletes the entry from the table. Otherwise, the program operates in CHANGE mode and replaces the description portion of the table entry with the description portions of the transactions.

Refer to Figure 4.1.4 for a description of the transactions.

The transactions must be sorted on the following fields before being read:

Key	bytes 2-15	ascending
Transaction-id	byte 1	ascending

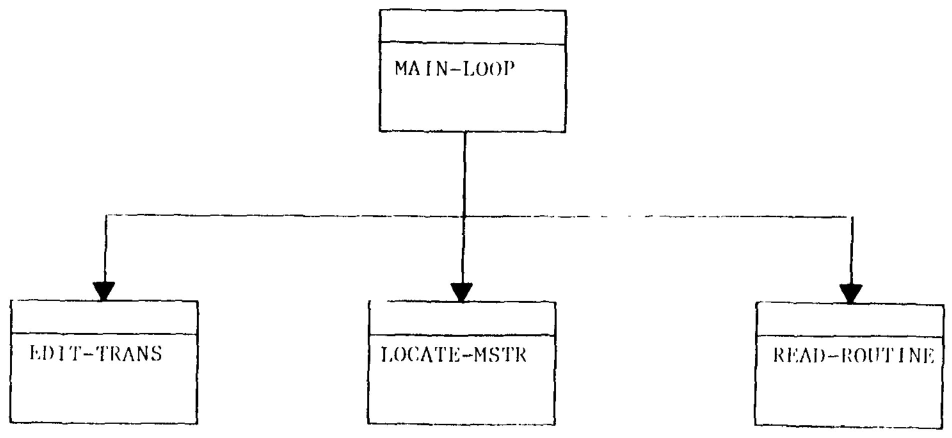
The name of the load module is HXTABLE1.

4.1.1 ORGANIZATION

The Parameter Code Table Maintenance program is organized in a top down, modular structure. There is only one entrance and one exit from the program, both contained in the highest level module. Each lower level module invoked has the same characteristics: one entrance and one exit.

- MAIN-LOOP
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the transaction file. First, EDIT-TRANS is performed, which validates the fields in the transactions. If the cards are valid, LOCATE-MSTR is performed until the proper position is located. The required update is then performed. Finally, READ-ROUTINE is performed which reads two cards into the transaction work area.

Figure 4.1.1 shows the organization of HXTABLE1.



HXTABLE1 - Organization
Figure 4.1.1

4.1.2 <u>COMMUNICATION AND DATA FORMATS</u>

The following defines the usage of certain working storage names:

- <u>END-OF-FILE-SW</u>
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- <u>ERROR-FOUND-SW</u>
  May contain TRUE or FALSE. If true, an error has been detected during the card read routine processing.

- <u>LAST-TRANS-WAS-DELETE-SW</u>
  May contain TRUE or FALSE. If true, the last transaction deleted a parameter table entry.

- <u>READ-SUPPRESSED-SW</u>
  May contain TRUE or FALSE. If true, suppresses the reading of the transaction file.

- <u>RECORD-LOCATED-SW</u>
  May contain TRUE or FALSE. If true, the transactions have been matched against the parameter table.

- <u>TRANS-REJECTED-SW</u>
  May contain TRUE or FALSE. If true, there is an error in one or more of the input data fields being edited.

- <u>KEY-SAVE</u>
  Contains the current old parameter table key until end-of-file, then it is replaced with high values.

4.1.3 ROUTINES

The following are major subroutines of the Parameter Code Table Maintenance program.

- COPY-MSTR
  This routine performs WRITE-MSTR which writes out the new parameter table records unless the last transaction delete switch is on. Next, it moves the old parameter table records to work areas. Last, READ-MSTR is performed which reads the old parameter table files.

- EDIT-TRANS
  This routine edits the fields in the input card for valid data. If invalid, an appropriate error message is printed out.

- LOCATE-MSTR
  This routine cycles through the parameter table files, performing COPY-MSTR and WRITE-MSTR until an equal-to or greater-than condition occurs between the old parameter table and the input data.

- READ-CARD-ROUTINE
  This routine reads two cards and verifies that they have the same key and are in the proper sequence. If there is an error, an appropriate error message is printed out.

- READ-ROUTINE
  This routine performs READ-CARD-ROUTINE until END-OF-FILE-SW is true or ERROR-FOUND-SW is false.

Parameter Code Table Transactions

Card 1

Columns	Format	Symbol	Description
1	9	Numeric	Card Type
2 - 6	99999	Numeric	Parameter Code
7 - 8	99	Numeric	Method Code
9 - 10	99	Numeric	Units Code
11 - 14	9999	Numeric	Minimum Detectable
15	9	Numeric	Decimal Point
16 - 45	X - X	Alphameric	Parameter Description
46 - 70	X - X	Alphameric	Collection Method
71 - 80			Unused

Card 2

Columns	Format	Symbol	Description
1 - 15			Same as Card 1
16 - 40	X - X	Alphameric	Analysis Method
41 - 70	X - X	Alphameric	Units Description
71 - 80			Unused

Parameter Code Table Transactions

Figure 4.1.4

4.1.4 DESCRIPTION OF PARAMETER CODE TABLE TRANSACTION FIELDS

Card Type: Identifies the transaction as either card 1 or card 2 of a two card set. Valid values are 1 and 2.

Parameter Code: Identifies the parameter being described. Refer to APTD-0633 for a full list of currently accepted parameter codes.

Method Code: Identifies both the collection method and the analysis method of the parameter being described.

Units Code: Indicates the units in which the minimum detectable value is expressed. A partial list may be found in Appendix A, Table 4.

Minimum Detectable: Specifies the minimum value detectable using the specified collection and analysis methods.

Decimal Position: A single-digit number from 0 to 4 which indicates the number of digits in the minimum detectable value that are to fall to the right of the decimal point.

Parameter Description: A prose description of the parameter.

Collection Method: A prose description of the collection method.

Analysis Method: A prose description of the analysis method.

Units Description: A prose description of the units.

4.2 SITE CODE TABLE MAINTENANCE

The Site Code Table Maintenance program creates and updates a table containing valid site codes for this installation of AQDHS plus a prose description of the site.

The key of each transaction contains the full coded identification of the site: state code, area code, site code, agency code and project code. If a matching key is found in the table file and the site description in the transaction is blank, the table entry is deleted from the file. If the transaction description is not blank, it replaces the description in the table. If no matching key is found, a new table entry is created. This process is identical to that described in section 4.1.

Refer to Figure 4.2.4 for a description of transaction.

The transactions must be sorted on the following field before being used:

Key	bytes 1-12	ascending

The name of the load module is HXTABLE2.

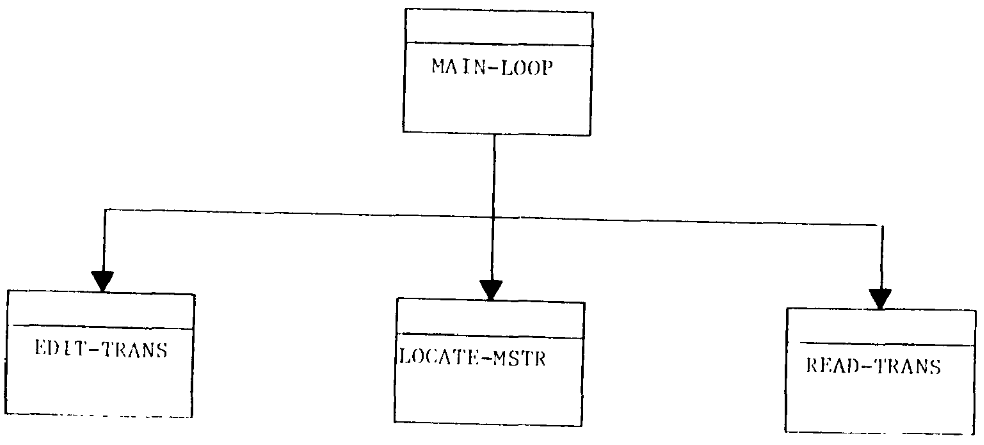
4.2.1 ORGANIZATION

The Site Code Table Maintenance program is organized in a top down, modular structure. There is only one entrance and one exit from the program, both contained in the highest level module. Each lower level module invoked has the same characteristics: one entrance and one exit.

- MAIN-LOOP

  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the transaction file. First EDIT-TRANS is performed which validates the fields on card input. If the fields on the card are valid, LOCATE-MSTR is performed until the master file is positioned properly. The required update is then performed. Finally, READ-TRANS is performed which reads the next card.

Figure 4.2.1 shows the organization of HXTABLE2.



HXTABLE2 - Organization
Figure 4.2.1

4.2.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- LAST-TRANS-WAS-DELETE-SW
  May contain TRUE or FALSE. If true, the last transaction deleted a site code table entry.

- RECORD-LOCATED-SW
  May contain TRUE or FALSE. If true, the transaction has been matched against the site code table.

- TRANS-REJECTED-SW
  May contain TRUE or FALSE. If true, there is an error on one or more of the input data fields being edited.

- KEY-SAVE
  Contains the current site code table key until end-of-file, then it is replaced with high values.

4.2.3 ROUTINES

The following are major subroutines of the Site Code Table Maintenance program.

- COPY-MSTR
  This routine performs WRITE-MSTR which writes out the site code table record unless the last transaction delete switch is on. Next, it moves the old site code table record to a work area.

- EDIT-TRANS
  This routine edits the fields on card input for valid data. If invalid, an appropriate error message is printed out.

- LOCATE-MSTR
  This routine cycles through the site code table file, performing COPY-MSTR and WRITE-MSTR until an equal-to or greater-than condition occurs between the old site code table and the input data.

- READ-MSTR
  This routine reads a site code table record and moves the old site code table key to key-save until end-of-file, then high values are moved to key-save.

- READ-TRANS
  This routine reads a transaction record and at end-of-file turns the END-OF-FILE-SW on.

Site Code Table Transaction

Columns	Format	Symbol	Description
1 - 2	XX	Alphameric	State Code
3 - 6	9999	Numeric	Area Code
7 - 9	999	Numeric	Site Code
10	A	Alphabetic	Agency Code
11 - 12	99	Numeric	Project Code
13 - 72	X - X	Alphameric	Site Description
73 - 80			Unused

Site Code Table Transaction

Figure 4.2.4

4.2.4 DESCRIPTION OF SITE CODE TABLE TRANSACTION FIELDS

State Code: Indicates the state (or other geographic division) in which the sampling site is located.

Area Code: Identifies the area within the state in which the sampling site is located.

Site Code: Identifies the sampling site.

Agency Code: Identifies the agency responsible for this sampling site. Valid codes are found in Appendix A, Table 1.

Project Code: Identifies the project using this sampling site. Valid codes are found in Appendix A, Table 2.

Site Description: A prose description of the sampling site.

4.3 PARAMETER STANDARDS TABLE MAINTENANCE

The Parameter Standards Table Maintenance program is used to create and maintain a table containing both state and federal parameter standards. Both of these standards may then appear on any reports relating to those parameters.

A maximum of one state standard and one federal standard will be selected for each report. Each standard is selected in the following fashion. The table is searched and the location of the first state or the first federal standard for the parameter in question is saved. An attempt is then made to match the unit codes in the master file with those in the table. If more than one match is made on the unit codes, the first one encountered is used. If no match on unit codes is made, the first standard for the parameter is used. Therefore, it is important to enter standards in order of decreasing priority. It may be desirable to create several standards tables with different priorities assigned to a given standard.

The key of each transaction contains the following information: parameter code, federal/state flag, and standard number. If a matching key is found in the table file and the description on the transaction is not blank, the proper description (primary or secondary) in the table entry is replaced. If the description is blank and the transaction indicates a primary standard, the entire table entry is deleted. Otherwise, only the secondary standard is deleted. If no matching key is found, a new table entry is created.

Refer to Figure 4.3.4 for a description of the transaction.

The transactions must be sorted before being input on the following fields:

parameter code	byte 1-5	ascending
federal/state flag	byte 6	ascending

standard number	bytes 7-8	ascending
primary/secondary flag	byte 9	ascending

The name of the load module is HXTABLE3.

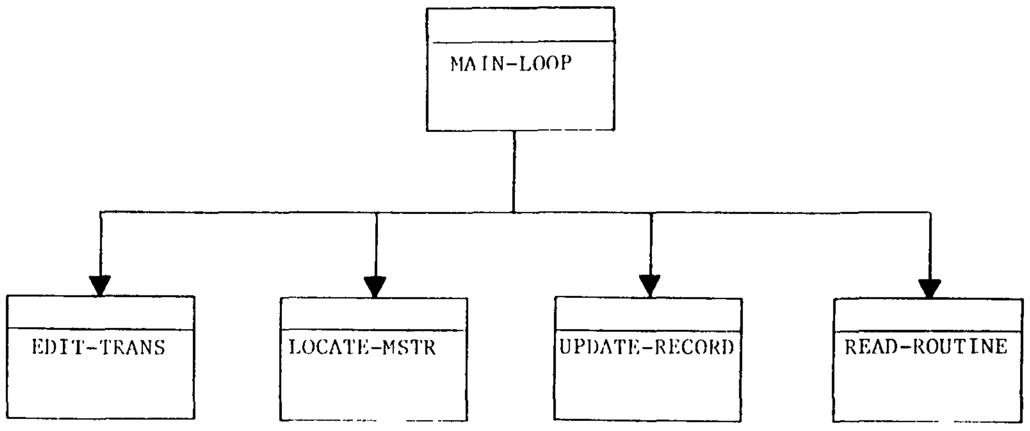
4.3.1 ORGANIZATION

The Parameter Standards Table Maintenance program is organized into two routines, the root segment and the main loop.

- ROOT-SEGMENT
  Opens the data sets required by the program module, initializes work areas, switches, and files. Next, the data file is processed by MAIN-LOOP until end-of-file. Table entries are written or deleted according to edited input. Finally, summary totals are written, files closed, and then the root segment terminates the run.

- MAIN-LOOP
  Editing is performed on card input by EDIT-TRANS. If no errors are found, LOCATE-MSTR is performed until record found switch is true and it is not a delete entry. Then the master record is updated or a new record added. Perform the READ-ROUTINE.

Figure 4.3.1 shows the organization of HXTABLE3.



HXTABLE3 - Organization
Figure 4.3.1

4.3.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- LAST-TRANS-WAS-DELETED-SW
  May contain TRUE or FALSE. If true, the last transaction deleted a standards table record.

- RECORD-FOUND-SW
  May contain TRUE or FALSE. If true, either a new record is added or an old record is changed.

- TRANS-REJECTED-SW
  May contain TRUE or FALSE. If true, an error or errors have been found upon editing card input.

- KEY-SAVE
  Contains the old standards table key until end-of-file, then it is changed to high values.

4.3.3 ROUTINES

The following subroutines are part of the MAIN-LOOP.

- EDIT-TRANS
  This paragraph edits all fields for valid data. If any fields are invalid, they are flagged and ERROR-ROUTINE is performed which lists an appropriate error message.

- LOCATE-MSTR
  This paragraph positions the standards file so that an existing record may be updated or deleted or a new record may be created.

- READ-ROUTINE
  This paragraph reads a card and lists it.

- UPDATE-RECORD
  This paragraph checks to see if the input record is new or old, lists an added or replaced message, and updates the work area.

Parameter Standards Table Transaction

Columns	Format	Symbol	Description
1 - 5	99999	Numeric	Parameter Code
6	A	Alphabetic	Standard Source (F/S)
7 - 8	99	Numeric	Standard Number
9	9	Numeric	Standard Type
10 - 11	99	Numeric	Units Code
12 - 61	X - X	Alphameric	Standard
62 - 80			Unused

Parameter Standards Table Transaction
Figure 4.3.4

4.3.4 DESCRIPTION OF PARAMETER STANDARDS TABLE TRANSACTION FIELDS

Parameter Code: Identifies the parameter for which this standard applies. Refer to APTD-0633 for a full list of currently accepted parameter codes.

Standard Source: Identifies the source of this standard. Valid codes are F for federal and S for state.

Standard Number: A two-digit number indicating the frequency of use of this standard. 01 should represent the most frequently used standard for this parameter (i.e. the default to be used when no match can be made on units code).

Standard Type: Differentiates between primary and secondary standards. Valid codes are 1 for primary standard and 2 for secondary standard.

Units Code: Indicates the units in which the standard is expressed. A partial list may be found in Appendix A, Table 4.

Standard Description: A prose description of the standard.

5.0 FILE MAINTENANCE AND TRANSACTION EDITOR

Two programs are used to build and maintain the AQDHS master file. They are the Transaction Editor program and the File Maintenance program. The AQDHS File to SAROAD Input program also modifies the AQDHS master file. Refer to Section 3.3 for a discussion of these modifications.

5.1 TRANSACTION EDITOR

The Transaction Editor program performs two basic functions. First, it edits all input transactions for correctness and informs the user of any errors. Second, it converts all input transactions passing the edit into an internal format usable by the File Maintenance program. The editing and conversion of a transaction is unaffected by any other transaction, either preceding or following. Therefore, transactions may be entered in any order.

The internal format transactions output from the Transaction Editor program must be sorted into master file sequence before they are input to the File Maintenance program. The order of the sort is as follows:

Key-1	bytes 4-18	ascending
Key-3	bytes 25-34	ascending
Key-2	bytes 19-24	ascending
Action-code, Form-code & Status-flag	bytes 1-3	ascending

The action to be performed by an input transaction is indicated by the action code in column 80. A one (1) indicates that the master file record identified by this transaction is to be deleted. A two (2) indicates that the data on this transaction is to be added to the master file. This may occur through the creation of a new master file record or through addition to an existing master file record. A three (3) in column 80 indicates that the data on this transaction is to replace data already existing in a master file record. The same action code appears in the internal transaction following conversion.

Each diagnostic message pertaining to an input transaction is self-explanatory and contains the column number of the first column of the field in error. Errors detected in the repeating fields of the form 2 and form 3 transactions do not cause the entire transaction to be rejected but only the repeating field containing the error.

Refer to Figure 2.2-2.1 for the formats of the transactions.

The name of the load module is TXTREDIT.

5.1.1 ORGANIZATION

The following are the main modules of the Transaction Editor program.

- o <u>MAIN-LOOP</u>
  Performed by ROOT-SEGMENT, this routine edits input transactions, creating internal (system recognizable) transactions. Depending on the input transaction code, the routine performs a lower level segment to edit the different transaction formats.

- o <u>CONVERT-TRANS</u>
  Performed by MAIN-LOOP, this routine performs lower level segments to do the actual editing and to write the "edited" form out to the internal transaction file.

- o <u>SCAN-FULL-KEY</u>
  Performed by CONVERT-TRANS, this routine scans the input transaction in its entirety for user errors, e.g., invalid time code, non-numeric project code, invalid day, etc.

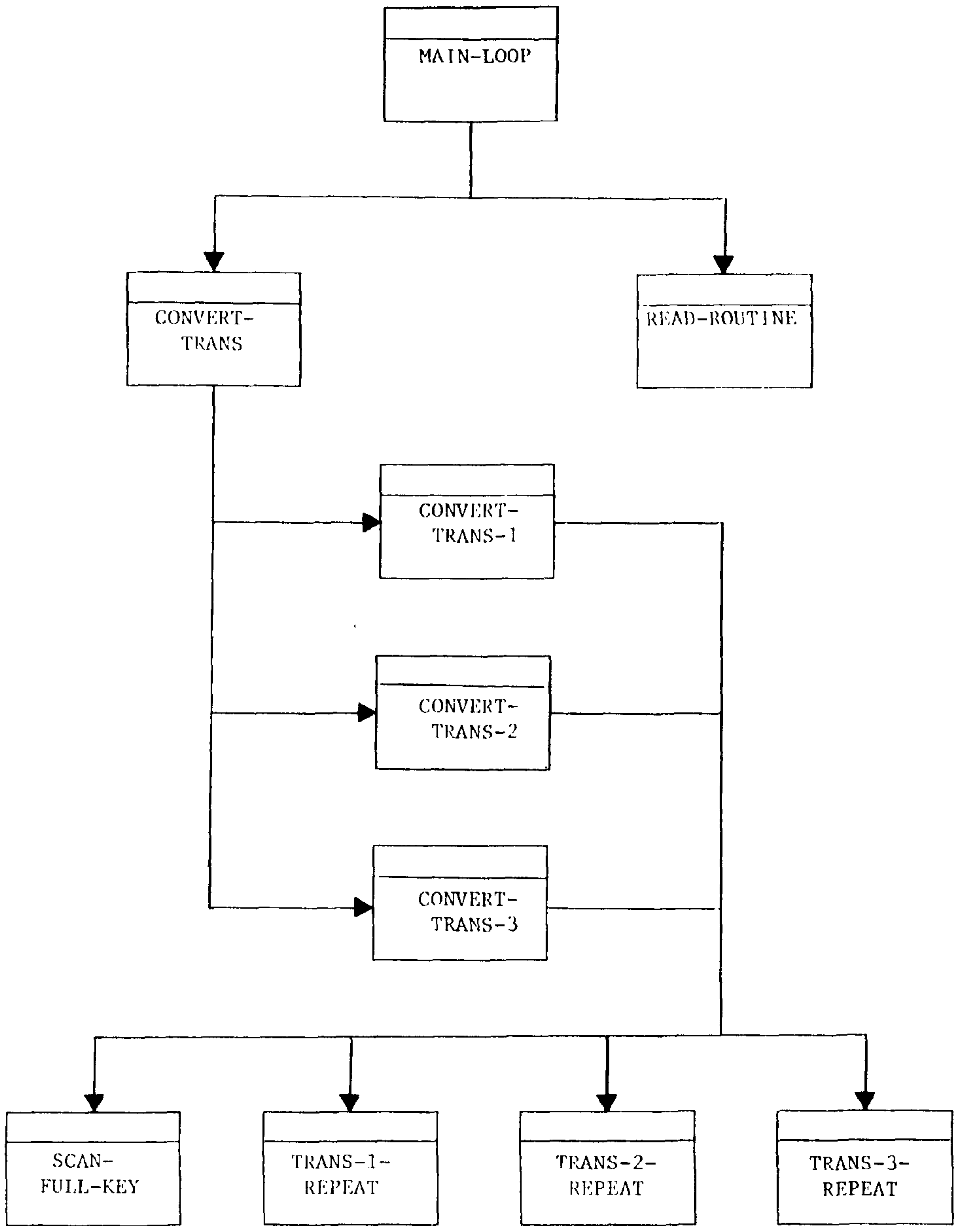
- o <u>TRANS-1-REPEAT</u>
  This routine is performed by CONVERT-TRANS when the input transaction form code is "1". It performs the necessary editing functions on the readings and in the case of invalid entries, prints out appropriate error messages.

- o <u>TRANS-2-REPEAT</u>
  This routine is performed by CONVERT-TRANS when the input transaction code is "2". It performs the

necessary editing functions on the repeating fields and, in the case of invalid entries, prints out appropriate error messages.

- TRANS-3-REPEAT
  This routine is performed by CONVERT-TRANS when the input transaction code is "3". It performs the necessary editing functions on the repeating fields and, in the case of invalid entries, prints out appropriate error messages.

Figure 5.1.1 shows the organization of TXTREDIT.



TXTREDIT - Organization
Figure 5.1.1

5.1.2 COMMUNICATION AND DATA FORMATS

The following COBOL working storage section fields are the major control and data areas internal to the Transaction Editor program.

- END-OF-FILE-SW

  When equal to TRUE, indicates that an end-of-file has occurred on the input transaction file.

- FULL-SCAN-SW

  Used by the routines which scan the form 2 and form 3 transactions. When equal to FALSE, indicates that only the fields in the repeating segments are to be scanned rather than the entire key.

- MAXIMUM-VALUES

  A table containing the maximum values of certain parameter/unit codes combinations. Any reading exceeding a value in this table will be rejected.

- WORK-TRANSACTION

  The area in which the internal format transaction is built.

5.1.3 ROUTINES

The following are internal subroutines of the Transaction Editor program.

- LOCATE-MAX-VAL
  Searches the MAXIMUM-VALUES table attempting to match the parameter and unit codes in the transaction with those in the table.

- SEARCH-FOR-PARM
  Searches the key portion of the parameter, method, unit codes table attempting to match the combination in the transaction with one in the table. If no match is found, the field in error in the transaction is flagged by a diagnostic message.

- SEARCH-FOR-SITE
  Searches the valid site code table attempting to match the site identification in the transaction with an entry in the table. If no match is found, the field in error in the transaction is flagged by a diagnostic message.

5.2 FILE MAINTENANCE

The File Maintenance program is used to create and maintain the AQDHS master file. Internal format transactions from the Transaction Editor program are used to accomplish these functions. These internal transactions must be sorted into file sequence before they are input to the File Maintenance program (see cataloged procedure AQSFUPDT).

Each internal transaction either adds data to, changes data in or deletes data from the AQDHS master file. Any combination of sites, parameter codes, sampling intervals, etc., may be entered in one execution of the File Maintenance program with the following exception: *each composite type (annual, weekly, etc.) must be entered in a separate execution of the program.* Otherwise, the data may be separated in the file rather than being collected into one record (i.e. multiple reports may be generated with one reading each).

If, at the time a master file record is created, all readings are not supplied, all blank fields up to the last supplied reading are filled with nines to indicate a null reading. These fields may then be updated by either an Add or a Change transaction. It is illegal to attempt to Change a non-existing reading or to Add to an existing reading.

If a reading has been unchanged since its submission to SAROAD, its status flag is 'S'. If it has been changed since submission to SAROAD, its status flag is 'C'. Its status flag is 'A' if it has not yet been sent to SAROAD.

Refer to Figure 2.2-1.1 for the format of the master file records.

The name of the load module is FXFILMNT.

5.2.1 ORGANIZATION

The following are the main modules of the File Maintenance program.

- MAIN-LOOP
  Performed by ROOT-SEGMENT, this routine (on the basis of the transaction action code) performs lower level segments to either add a record or data to the file, change a record in the file or delete a record from the file.

- FM-ADD
  This routine, performed by MAIN-LOOP, controls the adding of records or data to the system master file. It performs lower level routines to locate the place in the old master where the record should be added and adds the record to the new master file.

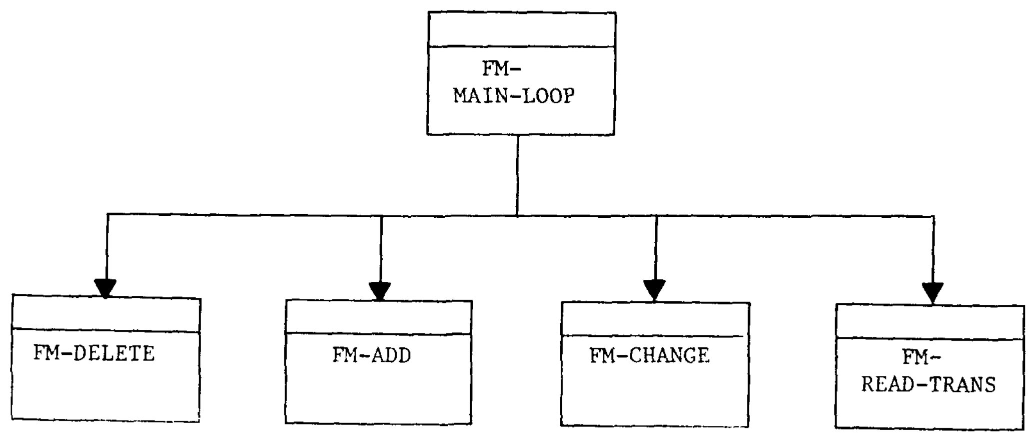
- FM-CHANGE
  This routine, performed by MAIN-LOOP, controls the changing of a record on the system file. It performs lower level routines that locate the record to be changed, makes the appropriate changes, and writes the updated record onto the file.

- FM-DELETE
  This routine, performed by MAIN-LOOP, controls the deletion of a record from the system master file. It performs lower level routines to locate the record to be deleted and deletes it from the file.

o <u>FM-READ-TRANS</u>

Performed by MAIN-LOOP, this routine reads the next record from the transaction file.

Figure 5.2.1 shows the organization of FXFILMNT.



FXFILMNT - Organization
Figure 5.2.1

5.2.2 COMMUNICATION AND DATA FORMATS

The following COBOL working storage section fields are the major control and data areas internal to the File Maintenance program.

- END-OF-FILE-SW
  When equal to TRUE, indicates that an end-of-file has occurred on the transaction file.

- LAST-TRANS-WAS-DELETE-SW
  When equal to TRUE, indicates that the preceding transaction deleted a record from the master file.

- MSTR-RECORD-LOCATED-SW
  Used by LOCATE-MSTR to indicate the positioning of the master file. When equal to 0, the master file has not been properly positioned. When equal to 1, the master file has been positioned and the key of the transaction is greater than the preceding master file record but less than the following master file record. The only valid action which may be taken is to build (add) a new record from the transaction. When equal to 2, the master file has been positioned and a matching key has been found. Data may be added to or changed in the record or the record may be deleted.

5.2.3 ROUTINES

The following are internal subroutines of the File Maintenance program.

- FM-COMPUTE-SUB
  This routine calculates the position within the master file record into which the first reading on the transaction will be placed.

- FM-COPY-MSTR
  This routine copies records from the old master file into the new master file.

- FM-LOCATE-MSTR
  This routine uses the key of the transaction to position the master file so that the action indicated on the transaction may be performed.

- FM-SCAN-RECORD
  This routine scans the new master file record before it is written to the new master file. Any readings which have not been supplied by this point will be filled with nines to indicate that they are null.

6.0 DATA RETRIEVAL

The capability to retrieve data from the AQDHS master file is provided by the Retrieval Language Processor program and the Retrieval program.

6.1 RETRIEVAL LANGUAGE PROCESSOR

The Retrieval Language Processor program generates the actual Retrieval program based on English-like requests entered by the user. These retrieval requests are edited and, if no severe errors are found, the Retrieval program is compiled and executed to extract any data meeting the user's specification. Those records in the master file which have qualified are placed in a separate answer file for subsequent processing. The records in the answer file are in the same order and have the same format as the records in the master file. Therefore, the answer file may be used by any program which accepts the master file for processing. Care should be used, however, as the answer file is a subset of the master file. Consequently, any changes to the answer file will not necessarily be reflected in the master file.

Every retrieval request contains two control cards. The $$END card signals the end of the request. The $$SELECT card signals the beginning of the request and the type of request. There are three different types of request: AQDHS retrieval language specifications, inline COBOL language specifications and copied COBOL language specifications.

Refer to Figure 6.1.4-1 for the formats of the $$SELECT and $$END cards.

Retrieval specifications written in the AQDHS retrieval language are entered immediately following the $$SELECT card, expressing one relationship per card. For example, A = B expresses a relationship where A is the subject name, = is the relational operator and B is the object name. Multiple relationships may be expressed by connecting single relationships with a Boolean operator. For example, A = B AND C = D OR E = F expresses three relationships, A = B, C = D and E = F connected by the Boolean operators AND and OR.

Refer to Figure 6.1.4-2 for the format of the AQDHS retrieval language specifications.

The subject name field and object name field may each contain either a valid data name (see Figure 6.1.4-3) or a literal enclosed in apostrophes. An invalid data name or missing apostrophes will cause the retrieval to abort. Parentheses may be used for grouping provided that the first occurrence of a repeating data name is not enclosed in parentheses. After the first occurrence of a repeating name, parentheses may be used freely. Parentheses must be balanced when the specifications are terminated.

The negation field must contain the negation character, 'N', or blank. Any other character will result in a warning message. The program will then assume that negation was intended and continue processing.

An invalid relational or Boolean operator will cause the retrieval to abort.

The Boolean operator field may contain blanks or the words AND or OR. Blanks in this field signal the end of the retrieval specification and terminate editing. The program then expects that the next card will be the $$END card. The words AND or OR indicate that another relationship follows and that editing is to continue.

Inline COBOL language specifications are entered immediately following the $$SELECT card. These cards are punched according to COBOL rules for syntax and punctuation. Any user defined paragraph or section name should begin with the prefix, USER-, to avoid conflict with other names in the program. Any valid PROCEDURE DIVISION statements, with the exception of DECLARATIVES, may be entered. Since the Retrieval Language Processor program does not examine the statements, any errors could result in COBOL diagnostics or unpredictable results. A data name named SUB (PIC 99 COMP SYNC) is provided for use as a subscript for referencing repeating data names.

Copied COBOL language specifications have the same attributes as inline COBOL specifications. However, they are stored externally and are copied in at the time the Retrieval program is compiled.

Refer to Figure 6.1.4-4 for sample retrieval specifications.

The name of the load module is LXLNGPRC.

6.1.1 ORGANIZATION

The following are the main modules in the Retrieval Language Processor program.

- o MAIN-LOOP

  This routine has a conditional branch which branches to GENERATE-COBOL, COPY-COBOL, or COPY-MEMBER depending on the option entered on the $$SELECT card.

- o GENERATE-COBOL

  This routine processes AQDHS retrieval language specifications and, after editing, generates the proper COBOL code to perform the retrieval.

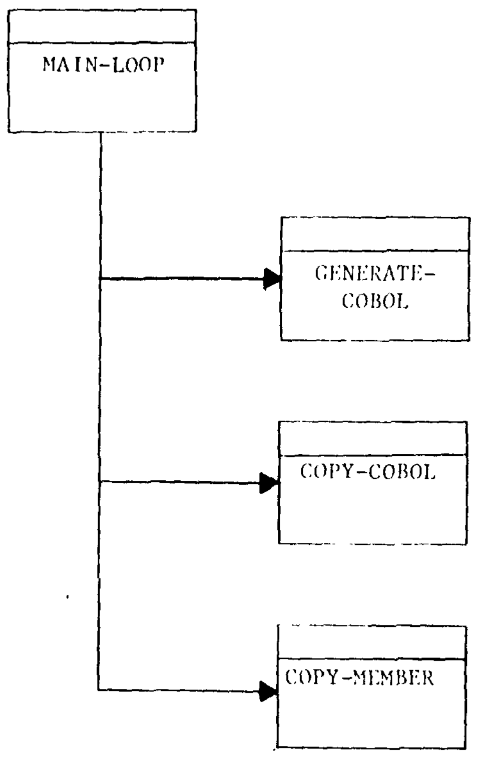
- o COPY-COBOL

  Inline COBOL language specifications are copied into the Retrieval program skeleton by this routine.

- o COPY-MEMBER

  Code to copy an external COBOL member into the Retrieval program skeleton is generated by this routine.

Figure 6.1.1 shows the organization of LXLNGPRC.



LXLNGPRC - Organization
Figure 6.1.1

6.1.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- BOOLEAN-SW

  May contain '0' (NO-PRECEDING-BOOLEAN), '1' (LAST-BOOLEAN-WAS-OR), or '2' (LAST-BOOLEAN-WAS-AND) depending on the contents of the Boolean operator field of the last control card processed.

- CHECK-REPEATING-NAME-SW

  May contain TRUE or FALSE. If true, a search is performed to determine whether or not the current data name is a repeating data name.

- CONTINUATION-PENDING-SW

  May contain TRUE or FALSE. If true, Boolean checks are made and/or an output record is written.

- DATA-NAME-FOUND-SW

  May contain TRUE or FALSE. If true, the data name has been found in the list of valid data names.

- END-CARD-FOUND-SW

  May contain TRUE or FALSE. If true, either the $$END card has been found or an end-of-file has occurred on the control card file.

- END-OF-FILE-SW

  May contain TRUE or FALSE. If true, a data set end-of-file has been detected or signaled.

- <u>ERROR-FOUND-SW</u>

  May contain TRUE or FALSE. If true, an error has been found during editing.

- <u>IF-WRITTEN-SW</u>

  May contain TRUE or FALSE. If true, the IF clause has been written to the Retrieval program skeleton.

- <u>PARENS-BALANCED-SW</u>

  May contain TRUE or FALSE. If true, the number of left parentheses are equal to the number of right parentheses.

- <u>REPEATING-NAME-SW</u>

  May contain TRUE or FALSE. If true, the current data name is a repeating data name.

6.1.3 <u>ROUTINES</u>

The following are major subroutines of the AQDHS Retrieval Language Processor program.

- <u>CHECK-DATA-NAME</u>
  This routine first performs STRIP-PARENS which strips the left and right parentheses from the current data name and replaces them with blanks. Next SEARCH-TABLE is performed which checks for a valid data name. If it is not a valid data name, then an error is printed.

- <u>CHECK-FOR-SUBSCRIPT</u>
  This routine first replaces the left and right parentheses with blanks. Then WORK-DATA-NAME is moved to DATA-NAME-BUFFER left justifying the name. Next CHECK-REPEATING-DATA-NAME is performed which checks for a repeating data name. If it is a repeating name, then '(SUB)' is moved to the output record work area.

- <u>EDIT-NON-REPEATING-DATA</u>
  This routine performs PROCESS-CONTINUATION if CONTINUATION-PENDING-SW is true. Next EDIT-NON-REPEATING-GUTS is performed which does edit checking and prints an error message for an invalid field or writes an output record.

- <u>EDIT-REPEATING-DATA</u>
  This routine performs PROCESS-CONTINUATION if CONTINUATION-PENDING-SW is true. Next EDIT-REPEATING-GUTS is performed which does edit checking and prints an error message for any invalid field. If DATA-NAME-FOUND-SW is true then CHECK-FOR-SUBSCRIPT is performed.

If END-OF-FILE-SW is false, CHECK-DATA-NAME is performed for data name and comparand.

- <u>GEN-ELSE</u>

  This routine generates an ELSE clause in the Retrieval program.

- <u>GEN-PERFORM</u>

  The routine generates a PERFORM clause in the Retrieval program to handle repeating data names.

- <u>PROCESS-CONTINUATION</u>

  This routine tests for a Boolean 'OR' or 'AND' and then tests PARENS-BALANCED-SW and REPEATING-NAME-SW generating the proper code for the switch configuration.

- <u>READ-ROUTINE</u>

  This routine reads a control card and prints it.

- <u>WRITE-READ-ROUTINE</u>

  This routine writes a record then performs READ-ROUTINE. It is used for copying inline COBOL language specifications into the Retrieval program.

Retrieval Control Cards

Columns	Format	Symbol	Description
1 - 8	A - A	Alphabetic	Card Type
9			Unused
10 - 13	AAAA	Alphabetic	User Id
14 - 15			Unused
16 - 23	X - X	Alphameric	Member Name
24 - 80			Unused

Retrieval Control Cards

Figure 6.1.4-1

6.1.4-1 DESCRIPTION OF RETRIEVAL CONTROL CARD FIELDS

Card Type: Identifies the card as either a $$SELECT card or a $$END card. The keyword must be left justified within the field.

User Id: If the keyword USER is entered in this field, it indicates that the retrieval request is in COBOL, rather than in AQDHS retrieval language. Valid only on the $$SELECT card.

Member Name: Specifies the name of the member in an external source library containing the COBOL retrieval request. Valid only in conjunction with the USER option. If blank, indicates that the COBOL specifications are inline, rather than on a library.

Retrieval Language Specifications

Columns	Format	Symbol	Description
1 - 22	X - X	Alphameric	Subject Name
23			Unused
24	A	Alphabetic	Negation Flag
25			Unused
26	X	Special Character	Relational Operator
27			Unused
28 - 67	X - X	Alphameric	Object Name
68			Unused
69 - 71	AAA	Alphabetic	Boolean Operator
72 - 80			Unused

Retrieval Language Specifications

Figure 6.1.4-2

6.1.4-2 DESCRIPTION OF RETRIEVAL LANGUAGE SPECIFICATIONS

Subject Name: The name of a field in the AQDHS master file record whose content is to be tested, or a literal (enclosed in apostrophes) to be compared against the value specified by the object name.

Negation Flag: When non-blank, indicates the Boolean NOT condition. Valid codes are N and space.

Relational Operator: Indicates the relationship to be tested between the subject name and the object name. Valid codes are:

=	equal
>	greater than
<	less than

Object Name: The name of a field in the AQDHS master file record whose content is to be tested, or a literal (enclosed in apostrophes) to be compared against the value specified by the subject name.

Boolean Operator: Connects a relationship or group of relationships to another relationship or group of relationships. Valid values are AND and OR. AND indicates that both relationships or groups must be true in order for the entire relationship to be true. OR indicates that if either relationship or group is true, the entire relationship is true.

Valid Data Names for Retrieval

ACTION-CODE	KEY-3
AGENCY	METHOD-CODE
AREA-CODE	MONTH
COMPOSITE-KEY-2	NBR-OF-READINGS
COMPOSITE-PERIOD	PARAMETER-CODE
COMPOSITE-SAMPLES	PROJECT
COMPOSITE-TIME-CODE	RPTING-SECTION*
COMPOSITE-TYPE	SITE
DATA-FIELD*	START-HOUR
DAY-CODE	STATE
DECIMAL-CODE	STATUS-FLAG*
FORM-CODE	TIME-CODE
IDENT-KEY	UNIT-CODE
KEY-1	YEAR
KEY-2	

* Repeating data names.

Valid Retrieval Data Names

Figure 6.1.4-3

6.1.4-3 DESCRIPTION OF RETRIEVAL DATA NAMES

ACTION-CODE: Indicates the last action to be performed on the master file record. The only possible values are '2' for Add and '3' for Change.

AGENCY: Identifies the agency responsible for the sampling. Valid codes are found in Appendix A, Table 1.

AREA-CODE: Identifies the area in which the samples were taken.

COMPOSITE-KEY-2: Identifies, as a group, the following fields:

COMPOSITE-PERIOD
COMPOSITE-SAMPLES
COMPOSITE-TYPE
COMPOSITE-TIME-CODE

COMPOSITE-PERIOD: Indicates the period during which the composite sample was taken. Valid codes are:

'01' - '04'	Quarterly and Seasonal Composite
'01' - '12'	Monthly Composite
'01' - '52'	Weekly Composite
'00'	Annual Composite

COMPOSITE-SAMPLES: Indicates the number of individual samples that were composited.

COMPOSITE-TIME-CODE: Indicates the interval at which the individual composited samples were taken. This time code should be taken from Code Table 3 in APTD-0663 rather than from Appendix A, Table 3 in this document.

COMPOSITE-TYPE: Indicates the interval at which the samples are composited. Valid codes are:

'1'	Quarterly Composite
'2'	Seasonal Composite
'3'	Monthly Composite
'4'	Weekly Composite
'5'	Annual Composite

DATA-FIELD: The value of the reading or sample taken.

DAY-CODE: The day of the month on which the sample was taken.

DECIMAL-CODE: A single-digit number from 0 to 4 which indicates the number of digits in the reading that are to fall to the right of the decimal point.

FORM-CODE: The number of the form used to create the master file record, 1, 2 or 3.

IDENT-KEY: Identifies, as a group, the following fields:

KEY-1
KEY-2
KEY-3

KEY-1: Identifies, as a group, the following fields:

STATE
AREA-CODE
SITE
AGENCY
PROJECT
TIME-CODE
YEAR

KEY-2: Identifies, as a group, the following fields:

MONTH

DAY-CODE

START-HOUR

KEY-3: Identifies, as a group, the following fields:

PARAMETER-CODE

METHOD-CODE

UNIT-CODE

DECIMAL-CODE

METHOD-CODE: Identifies both the collection method and the analysis method of the parameter being measured.

MONTH: The month of the year during which the sample was taken.

NBR-OF-READINGS: The number of readings or samples contained in the master file record.

PARAMETER-CODE: Identifies the parameter being measured. Refer to APTD-0633 for a full list of currently accepted parameter codes.

PROJECT: Identifies the project in association with which the sample was taken. Valid codes are found in Appendix A, Table 2.

RPTING-SECTION: Identifies, as a group, the following repeating fields:

STATUS-FLAG

DATA-FIELD

SITE: Identifies the site at which the samples were taken.

START-HOUR: The hour at which the first sample was taken.

STATE: Identifies the state (or other geographic division) in which the sampling site is located.

STATUS-FLAG: Indicates the status of the associated reading. Valid codes are:

'A'	Added since submission to SAROAD
'C'	Changed since submission to SAROAD
'S'	Sent to SAROAD

TIME-CODE: Indicates the interval at which the samples were taken. Valid codes are found in Appendix A, Table 3.

UNIT-CODE: Indicates the units in which the parameter was measured. A partial list may be found in Appendix A, Table 4.

YEAR: The year in which the sample was taken.

Examples of Retrieval Specifications

A:

```
    $$SELECT
    TIME-CODE                    = '1'
    $$END
```

B:

```
    $$SELECT
    YEAR                         > '70'              AND
    (MONTH                       < '04'              OR
    MONTH                        > '06')             AND
    TIME-CODE                    = '8'
```

C:

```
    AGENCY                       = 'F'               OR
    AGENCY                       = 'G'               OR
    AGENCY                       = 'H'               OR
    TIME-CODE                    = 'C'               AND
    COMPOSITE-TYPE               = '3'
    $$END
```

D:

```
    $$SELECT USER  RETRVL01
    $$END
```

E:

```
    $$SELECT USER
         COBOL code
    $$END
```

Sample Retrieval Specifications

Figure 6.1.4-4

6.1.4-4 EXPLANATION OF SAMPLE RETRIEVAL SPECIFICATIONS

A: This specification will select all data from the file with a time code equal to '1'. Any data collected at intervals of other than hourly will not appear in the answer file.

B: This specification will select all data collected at daily intervals (time code equal to '8') in the first, third or fourth quarter (month less than '04' or greater than '06') of any year after 1970 (year greater than '70'). A warning message will be printed because of the missing $$END card.

C: This specification will select any data collected by a state agency (agency code equal to 'F'), a county agency (agency code equal to 'G') or a city agency (agency code equal to 'H'), or any composite data (time code equal to 'C') composited at monthly intervals (composite type equal to '3'). The missing $$SELECT card will cause a warning message to printed.

D: This specification will cause a member named RETRVL01 to be copied from the user's source library into the retrieval program skeleton. This member should contain the COBOL code necessary to perform the desired selection.

E: This format would be used to test the COBOL code to be used in format D prior to its being placed into the source library. The COBOL code would simply be included between the $$SELECT and the $$END cards.

6.2 <u>RETRIEVAL</u>

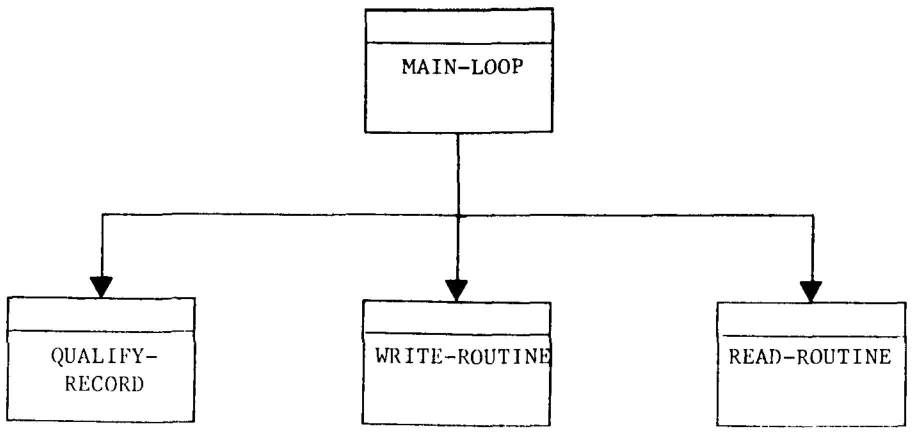
The AQDHS Retrieval program reads the master file and, based on user specifications edited by the Retrieval Language Processor program, selects those records which qualify for inclusion in an answer file.

6.2.1 ORGANIZATION

The following is the main module of the Retrieval program.

- MAIN-LOOP
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the AQDHS master file. First QUALIFY-RECORD is performed which is expanded by the Language Processor. Next, if the RECORD-QUALIFIES-SW is true, then an answer record is written. Finally, the next AQDHS master record is read.

Figure 6.2.1 shows the organization of the Retrieval program.



Retrieval - Organization
Figure 6.2.1

6.2.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- RECORD-QUALIFIES-SW
  May contain TRUE or FALSE. If true, an answer record will be written.

6.2.3 ROUTINES

The following are major subroutines of the AQDHS Retrieval program.

- READ-ROUTINE
  This routine reads an AQDHS master file record.

- WRITE-ROUTINE
  This routine moves data to the answer file work area and writes a record.

- QUALIFY-RECORD
  Contains the code generated by the Retrieval Language Processor program to perform the qualification of the master file record.

7.0 OUTPUT PROGRAMS

AQDHS provides three output programs for producing various listings of data contained in the master file. These are the Detail List program, the Sliding Average List program and the Data Analysis/Statistical List program set. The Data Analysis program also requires the use of two File Flagging programs.

7.1 DETAIL LIST

This program formats data contained in the AQDHS master file and provides a detailed listing of the data. The formats of the reports are controlled by the sampling interval of the data contained in the records being processed and by a user-supplied control card.

The control card has only two valid options. The keyword MEAN entered in columns 1 through 4, causes the number of readings, the mean reading and the maximum reading for each line and column to be printed on the report. The keyword SUM, entered in columns 1 through 3 (column 4 must be blank), causes the number of readings and the sum of the readings for each line to be printed. With the SUM option, no column footings are printed. Columns 5 through 80 of the control card are ignored by the detail list program. If the control card is missing or invalid, a warning message is issued, the MEAN option is assumed, and processing continues.

Refer to Figure 7.1.4 for the format of the control card.

The name of the load module is EXRPTLST.

7.1.1 ORGANIZATION

The following are main modules of the Detail List program.

- MAIN-LOOP
  Performed by ROOT-SEGMENT, this routine generates (based on a specified "sampling interval") an Air Quality Data Report for some hourly, daily, weekly, monthly, quarterly or composite period. The readings for each interval are always displayed as are the number of such readings. However, the user has the additional option of specifying that the report be "mean" in which case mean and maximum values are displayed (for row and column data) or that the report be "sum", in which case, the sum of the readings (row only) are displayed.

- FILL-TITLES
  Performed by MAIN-LOOP, this routine completes skeletal-type headings (titles) with certain identifying information (e.g., agency, state, project, parameter, parameter standards, sampling interval, etc.) so that they appropriately reflect the data to be listed.

- REPORT-LESS-THAN-DAILY
  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed reflects some hourly interval less than one day (i.e., the sampling interval is less than one day). The resulting report is such that for each hourly interval of each day (of a specified month) a reading is displayed. Additionally, for each interval, the number of readings and the mean and maximum values are determined and displayed or if the report is "sum", the sum.

- <u>REPORT-DAILY</u>

  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed is "daily", i.e., the sampling interval is one day. The report structure is such that readings are displayed for each day and the number of readings, and mean and maximum values (or sums) are determined and displayed for each day and each month.

- <u>REPORT-WEEKLY</u>

  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed is "weekly", i.e., the sampling interval is one week. The resulting report is similar in form to that of the report described when the data is "less than daily".

- <u>REPORT-MONTHLY</u>

  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed is "monthly", i.e., the sampling interval is one month. The resulting report is similar in form to that of the report described when the data is "less than daily".

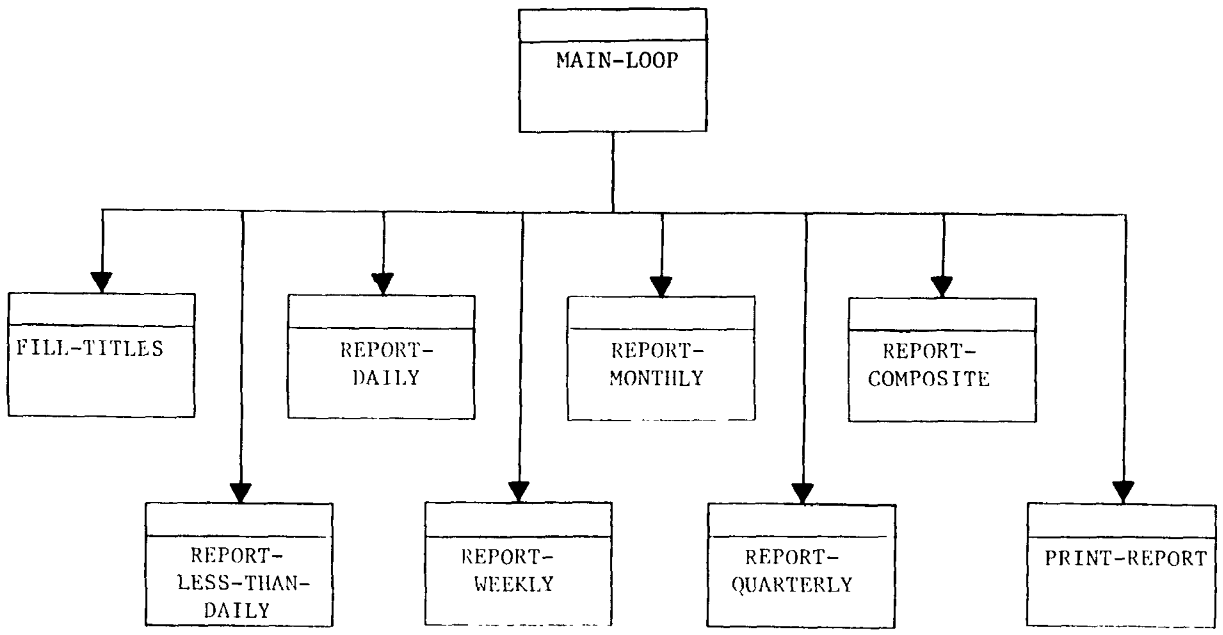
- <u>REPORT-QUARTERLY</u>

  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed is "quarterly", i.e., the sampling interval is one quarter. The resulting report is similar in form to that of the report described when the data is "less than daily".

- <u>REPORT-COMPOSITE</u>

  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the sampling interval is made up of distinct parts, i.e., is composite. The resulting report is similar in form to that of the report described when the data is "less than daily".

Figure 7.1.1 shows the organization of EXRPTLST.



EXRPTLST - Organization
Figure 7.1.1

7.1.2 COMMUNICATION AND DATA FORMATS

The following COBOL working storage section fields are the major control and data areas internal to the Detail List program.

- <u>COMPOSITE-REPORT-SW</u>
  When equal to TRUE, indicates that composite data is to be formatted.

- <u>PRINT-FOOTINGS-SW</u>
  When equal to TRUE, indicates that cross-footings are to be printed.

- <u>REPORT-IS-MEAN-SW</u>
  When equal to TRUE, indicates that the MEAN option was selected on the control card. When equal to FALSE, the SUM option was selected.

- <u>LINES-AREA</u>
  Report formatting work area.

- <u>PAGE-MATRICES</u>
  In-core buffer to hold up to two pages of report to allow column and line footings.

- <u>SAVE-IDENT-KEY</u>
  Holds the key of the first master file record in the reporting period. Used to detect a change in the sample set

- <u>WORK-AREA</u>
  Work area for formatting readings.

7.1.3 ROUTINES

The following are internal subroutines of the Detail List program.

- COMPUTE-INTERVAL
  Determines the format of less than daily reports and fills in the interval field of the titles.

- FOOTING-ROUTINE
  Formats and prints cross footings when the option on the control card was MEAN.

- FORMAT-MEAN-COLS
  Formats column NO. MEAN and MAX when the option on the control card was MEAN.

- FORMAT-MEAN-LINE
  Formats line NO. MEAN and MAX when the option on the control card was MEAN.

- FORMAT-SUM-LINE
  Formats line NO. and SUM when the option on the control card was SUM.

Detail List Control Card

Columns	Format	Symbol	Description
1 - 4	AAAA	Alphabetic	Report Type
5 - 80			Unused

Report Type: Specifies the type of report to be produced. Valid values for columns 1-4 are: MEAN AND SUM.

Detail List Control Card

Figure 7.1.4

7.2 SLIDING AVERAGE

This program formats the readings contained in each record in the AQDHS master file and computes sliding averages of the readings. The number of readings in each average is controlled by a user-supplied control card. The control card also determines whether the number of readings, the mean reading and the maximum reading in each record is printed (MEAN option) or the number of readings and the sum of the readings in the record (SUM option) is printed.

The MEAN option and SUM option of the control card occupy the same columns as used by the Detail List program. The default is also MEAN if an incorrect option is specified. If the control card is mission, the program is aborted. The number of readings to be used in each average is entered in columns 5 and 6. A leading zero must be specified for numbers less than 10. If the field is invalid (non-numeric, less than 02 or greater than 31), the program is aborted.

Refer to Figure 7.2.4 for the format of the control card.

The average for each group readings is printed directly under the last reading in the group. More than half the readings in the group must be non-null, otherwise asterisks are printed in place of the average. If the entire record contains too few non-null readings to compute averages, a message is printed to that effect. Each group of readings to be averaged must be contained within a single record as the program does not span records to accumulate readings.

Composite data is not supported by this program.

The name of the load module is IXSLDAVG.

7.2.1 ORGANIZATION

The following are the main modules of the Sliding Average program.

- MAIN-LOOP
  Performed by ROOT-SEGMENT, this routine generates (based on a specified "sampling interval") an Air Quality Data Report for some hourly, daily, weekly, monthly, or quarterly period. The readings for each interval are displayed, along with the number of such readings, mean and maximum values or when specified, the sum of the readings. Additionally, on the basis of a user-supplied integer, n, equal to some number of intervals, a sliding average is computed and displayed.

- REPORT-LESS-THAN-DAILY
  This routine, performed by MAIN-LOOP, sets up the appropriate report format when the data to be listed reflects some hourly interval less than one day. The resulting report is such that for each hourly interval of each day (of a specified month) a reading is displayed, then for each day, the number of readings, mean and maximum values or the sum of the readings (if so specified) are determined and displayed along with the sliding averages.

- REPORT-DAILY
  Performed by MAIN-LOOP, this routine sets up the appropriate report format when the data to be listed is "daily", i.e., the sampling interval is one day. The resulting report is similar in form to that of the report described when the data is "less than daily".

- <u>FILL-TITLES</u>

  As in Detail List Program

- <u>REPORT-MONTHLY</u>

  As in Detail List Program

- <u>REPORT-QUARTERLY</u>

  As in Detail List Program

- <u>REPORT-COMPOSITE</u>

  Composite data is not supported by this program.

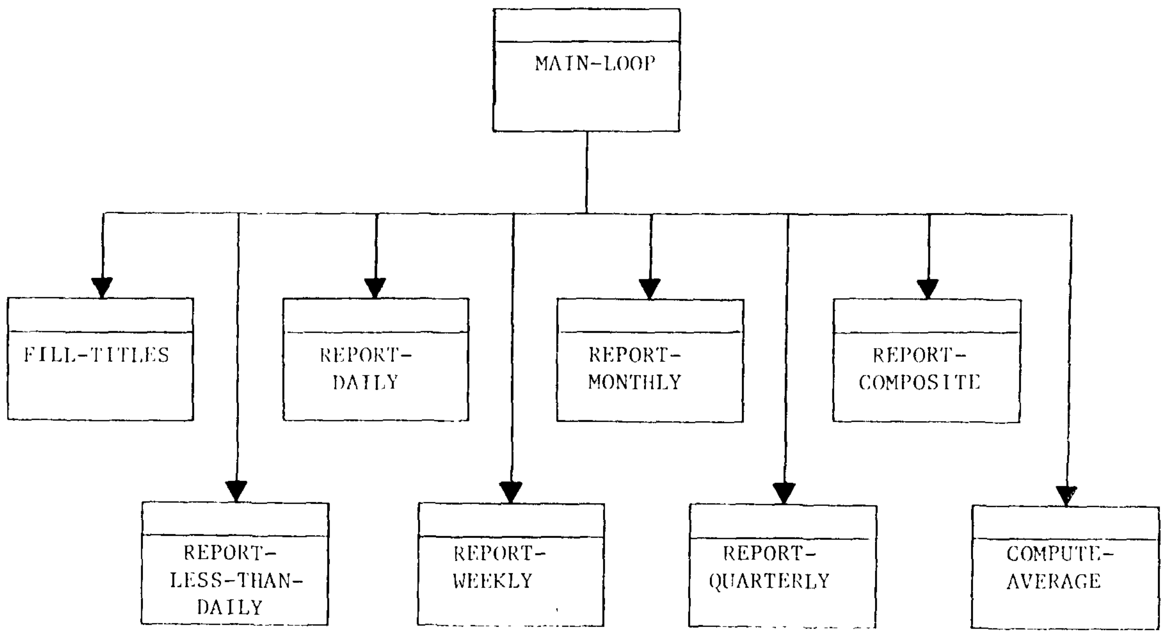
- <u>COMPUTE-AVERAGE</u>

  Performed by MAIN-LOOP, this routine uses a user-supplied value (n), where n equals some integral number of readings to compute a sliding average, i.e., beginning at the first reading after summing for "n" readings, computes an average, shifts (slides) right and repeats the process, this time beginning at the second reading; the process terminates when the final reading in the record is included in the averaging process.

- <u>READ-ROUTINE</u>

  Performed by MAIN-LOOP, this routine reads the next record from the master file.

Figure 7.2.1 shows the organization of IXSLDAVG.



IXSLDAVG - Organization
Figure 7.2.1

7.2.2 COMMUNICATION AND DATA FORMATS

The following COBOL working storage section fields are the major control and data areas internal to the Sliding Average program.

- <u>END-OF-FILE-SW</u>
  When equal to TRUE, indicates than an end-of-file has been detected on the input master file.

- <u>REPORT-IS-MEAN-SW</u>
  When equal to TRUE, indicates that the MEAN option was selected on the control card. When equal to FALSE, indicates that the SUM option was selected on the control card.

- <u>LINES-AREA</u>
  Report line formatting work area.

- <u>WORK-AREA</u>
  Reading formatting work area.

7.2.3 ROUTINES

The following are internal subroutines of the Sliding Average program.

- COMPUTE-INTERVAL
  For less than daily reports, determines the format of the report and fills in the interval field in the titles.

  AVERAGE
  the average [illegible] COMPUTE-AVERAGE and places it under the last reading in the interval.

- FORMAT-MEAN-LINE
  Formats line NO. MEAN and MAX when the option on the control card was MEAN.

- FORMAT-SUM-LINE
  Formats line NO. and SUM when the option on the control card was SUM.

- SUM-READINGS
  Sums the readings in the interval specified on the control card so that the average may be computed.

Sliding Average Control Card

Columns	Format	Symbol	Description
1 - 4	AAAA	Alphabetic	Report Type
5 - 6	99	Numeric	Number of Readings
7 - 80			Unused

Report Type: Specifies the type of report to be produced. Valid values are: MEAN and SUM.

Number of Readings: Specifies the number of samples to be averaged.

Sliding Average Control Card
Figure 7.2.4

7.3 DATA ANALYSIS

The Data Analysis program performs various statistical analyses on data contained in the AQDHS master file. The results of the calculations are written to the statistics file to be formatted by the Statistical List program.

The following statistical information is computed: minimum and maximum observations; arithmetic mean; standard deviation; geometric mean and standard deviation; percentage of readings present; and the 10th, 30th, 50th, 70th, 90th, 95th, 96th, 97th, 98th and 99th percentile occurrence.

The following statistics are used in this program (x represents a raw data value and n represent the number of values):

Arithmetic Mean:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{x_1 + x_2 + \ldots + x_n}{n}$$

Arithmetic Standard Deviation:

$$\sigma = \left[\frac{\sum_{i=1}^{n} x_i^2 - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n}}{n - 1} \right]^{1/2} =$$

$$\left[\frac{x_i^2 + x_2^2 + \ldots + x_n^2 - \frac{(x_1 + x_2 + \ldots x_n)^2}{n}}{n - 1} \right]^{1/2}$$

Geometric Mean:

$$\bar{x}_g = \text{Antilog}\left(\frac{\sum_{i=1}^{n} \log x_i}{n}\right) = \text{Antilog}\left(\frac{\log x_1 + \log x_2 + \ldots + \log x_n}{n}\right)$$

Geometric Standard Deviation:

$$\sigma_g = \text{Antilog}\left[\frac{\sum_{i=1}^{n} (\log x_i)^2 - \left(\frac{\sum_{i=1}^{n} \log x_i}{n}\right)^2}{n-1}\right]^{1/2}$$

The files input to the Data Analysis program (AQDHS master file or answer file and the key portion of the parameter, method, unit, minimum detectable table) must be flagged by the File Flagging programs before being used by Data Analysis.

Composite data is not supported by this program.

The name of the load module is DXSTATIS.

7.3.1 ORGANIZATION

Although the Data Analysis program is written in FORTRAN IV rather than in COBOL, it is still organized in the same fashion as the other programs in AQDHS. The gross structure consists of an initialization segment, a main loop and a wrap-up or termination segment. The following sub-segments are identified by the comments preceding them in the program source listing.

- Initialize the Sample Set - Before processing of a sample set begins, all accumulators are initialized to their nominal values.

- Initialize Minimum Detectable Table - This code is executed only once. It reads the key portion of the parameter, method, unit, minimum detectable table into an internal array for future reference by other program segments.

- Extract Parms for New Sample Set - This code saves the information necessary to identify the sample set.

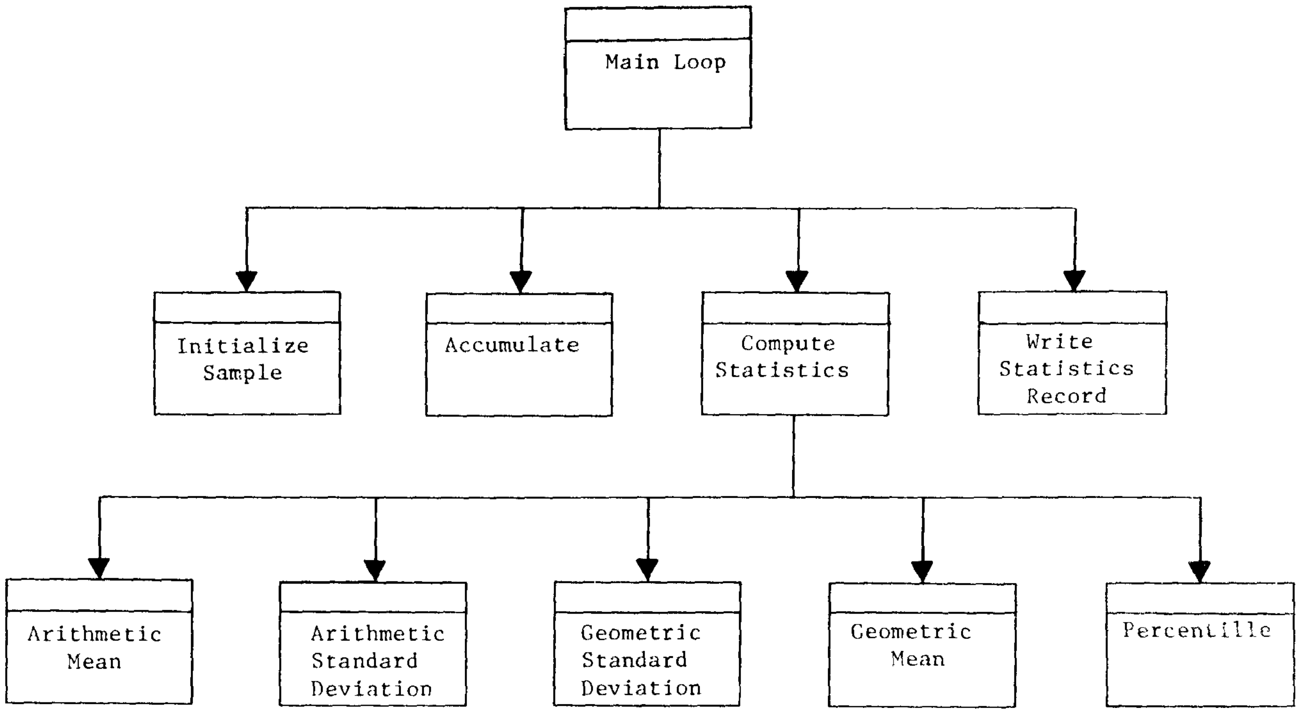
- Determine Minimum Detectable Value - The minimum detectable value array is searched until the current parameter code and unit code are matched and the associated minimum detectable value is extracted. If no match is found, the minimum detectable value is set equal to 1.0.

- Accumulator Loop - Accumulates intermediate internal results until a change in sample set (new state code, new parameter code, etc.) is detected. At the beginning of each loop, the date in the record is assumed to be the last date in the sample set and is saved. Any readings less than the minimum detectable value are substituted by half of the minimum detectable value.

- Compute Statistics - Computes and formats the beginning and end of the sample set; the minimum and maximum readings; the percentile distribution; the means and standard deviations; and the percentage of readings used in the calculations.

- Write Statistics Record - Formats the information based on the decimal position indicator and writes the record to the statistics file.

Figure 2.3.1 shows the organization of DXSTATIS.



DXSTATIS - Organization
Figure 7.3.1

7.3.2 COMMUNICATION AND DATA FORMATS

The following variable names are the major control and data areas internal to the Data Analysis program.

- LNORML

  May be .TRUE or .FALSE.. When .TRUE., indicates normal program completion. When .FALSE., indicates abnormal program completion.

- LMREOF

  Set to .TRUE. upon end-of-file on the copied master file.

- LMAINS

  Set to .FALSE. upon completion of all processing. Causes the program to leave the main loop and enter termination.

- LVFRST

  Set to .FALSE. to indicate that the minimum detectable value array has been built.

- LACCUM

  When .TRUE., indicates that processing and accumulation of the current sample set is to continue.

- NRDATA

  Array used to accumulate counts for calculating percentiles.

7.3.3 ROUTINES

There are no major internal subroutines in the Data Analysis program.

7.4 STATISTICAL LIST

This program accepts the statistics file generated by the Data Analysis program as its input and formats the information contained therein. There are no user options.

The name of the load module is SXPRINTS.

7.4.1 ORGANIZATION

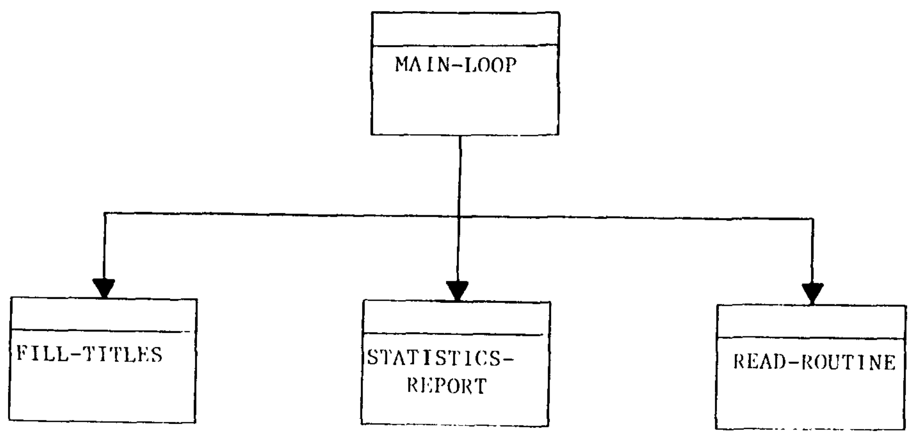
The following are main modules of the Statistical List program.

- MAIN-LOOP
  Performed by ROOT-SEGMENT, this routine generates a "Data Analysis Report" (from an input statistics file) providing such statistical information as percentiles, arithmetic and geometric means and arithmetic and geometric standard deviations.

- STATISTICS-REPORT
  Performed by MAIN-LOOP, this routine performs several lower level modules to actually format the report, (e.g., to set up and print headings) and to display it.

- READ-ROUTINE
  Performed by MAIN-LOOP, this routine reads the next record in the statistics file.

Figure 7.4.1 shows the organization of SXPRINTS.



SXPRINTS - Organization
Figure 7.4.1

7.4.2 COMMUNICATION AND DATA FORMATS

The following COBOL working storage section fields are the major control and data areas internal to the Statistical List program.

- END-OF-FILE-SW

  When equal to TRUE, indicates that an end-of-file has been detected on the input statistics file.

- RUN-ABORTED-SW

  When equal to TRUE, indicates that an error severe enough to cause the run to abort has been detected.

- PRINT-CONTROL-BLOCK

  Determines the format of the report listing.

7.4.3 ROUTINES

The following are internal subroutines of the Statistical List program.

- FILL-TITLES
  Completes skeletal headings with identifying information (e.g., agency, state, project, standards, etc.) so that they correctly identify the data in the report.

7.5 FILE FLAGGING

AQDHS supplies two programs to flag the input files used by the Data Analysis program. They flag the key portion of the parameter, method, unit, minimum detectable table and the master file or answer file to be analyzed. This flagging is required due to ANS FORTRAN restrictions.

7.5.1 ANSWER FILE FLAGGING

This program module copies the answer file and appends an end-of-file sentinel record.

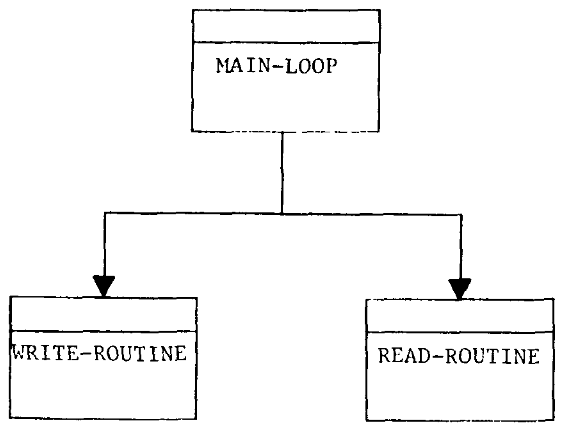
The name of the load module is MXSENTNL.

7.5.1.1 ORGANIZATION

- MAIN-LOOP

  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the answer file. First, the data is moved from the input area to the output area. Next, a new answer record is written and the old answer file is read. After all the input has been processed, an end-of-file sentinel record is written.

Figure 7.5.1.1 shows the organization of MXSENTNL.



MXSENTNL - Organization
Figure 7.5.1.1

7.5.1.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- END-OF-FILE-SW
  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- SENTINEL-CHARS
  An area of all nines which will form part of the sentinel record.

7.5.1.3 ROUTINES

The following are major subroutines of the AQDHS Data Analysis Copy program module.

- <u>READ-ROUTINE</u>
  This routine reads an answer file record.

- <u>WRITE-ROUTINE</u>
  This routine writes an answer file record.

7.5.2 PARM-CODE-KEY FILE FLAGGING

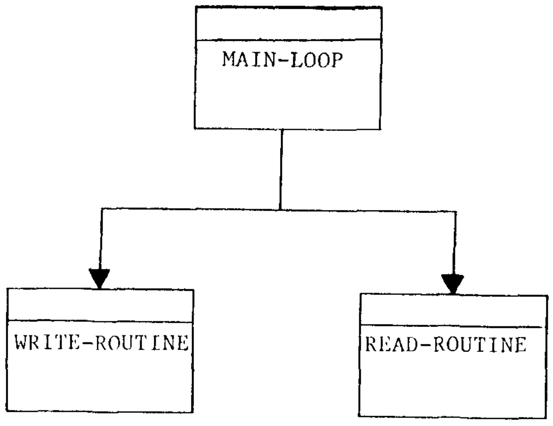
This program module copies the Parm-Code-Key file and appends an end-of-file sentinel record.

The name of the load module is NXSENTNL.

7.5.2.1 <u>ORGANIZATION</u>

- <u>MAIN-LOOP</u>
  This routine is performed by ROOT-SEGMENT until an end-of-file is detected on the Parm-Code-Key File. First, the data is moved from the input area to the output area. Next, a new Parm Code record is written and the next Parm-Code-Key file record is read. After all the input has been processed, an end-of-file sentinel record is written.

Figure 7.5.2.1 shows the organization of NXSENTNL.



NXSENTNL - Organization
Figure 7.5.2.1

7.5.2.2 COMMUNICATION AND DATA FORMATS

The following defines the usage of certain working storage names:

- o END-OF-FILE-SW

  May contain TRUE or FALSE. If true, a data set end-of-file has been detected.

- o SENTINEL-CHARS

  An area of all nines which will form part of the sentinel record.

7.5.2.3 ROUTINES

The following are major subroutines of the AQDHS Data Analysis Copy program module.

- READ-ROUTINE
  This routine reads a Parm-Code-Key file record.

- WRITE-ROUTINE
  This routine writes a Parm-Code-Flag file record.

Appendix A - CODE TABLES

Agency Codes

Code	Agency
A	EPA group responsible for atmospheric surveillance
B	EPA group responsible for meteorological activity
C	EPA group responsible for effects research
D	EPA group responsible for atmospheric research
E	EPA group responsible for abatement activity
F	State agency
G	County agency
H	City agency
I	District agency
J	Private
K	Institution (university, college, etc.)
L	Military
M	International agency
N	Other Federal nonmilitary agencies
O-Y	Open for future expansion
Z	Other

Agency Codes

Table 1

Project Codes

Long-term Surveillance

Code	Project
01	Population-oriented surveillance
02	Source-oriented ambient surveillance
03	Background surveillance

Short-term Surveillance

Code	Project
04	Complaint investigation
05	Special studies
06	Episode monitoring

Project Codes
Table 2

Time Codes

Format 1 Transaction

Code	Interval
1......	1 hour
2......	2 hours
3......	3 hours
4......	4 hours
5......	6 hours
6......	8 hours
7......	12 hours

Format 2 and 3 Transactions

Code	Interval
8.......	1 day
9.......	1 week
A.......	1 month
B.......	1 quarter
C.......	Composite data (Format 2 only)

Time Codes
Table 3

Unit Codes

Code	Units
01	micrograms/cubic meter (25^{o} C, 1013 millibars)
02	micrograms/cubic meter (0^{o} C, 1013 millibars)
03	nanograms/cubic meter (25^{o} C, 1013 millibars
04	nanograms/cubic meter (0^{o} C, 1013 millibars)
05	milligrams/cubic meter (25^{o} C, 1013 millibars)
06	milligrams/cubic meter (0^{o} C, 1013 millibars)
07	parts per million (volume/volume)
08	parts per billion (volume/volume)
09	COHS/1000 linear feet
10	RUDS/10,000 linear feet
11	meters/second
12	miles/hour
13	knots
14	degrees
20	microns
30	picocuries/cubic meter
31	microcuries/cubic meter
32	picocuries/square meter
33	microcuries/square meter
34	picocuries/cubic centimeter
35	picocuries/gram
50	number of threshold levels
70	milligrams F/100 square centimeters-day
80	milligrams SO_3/100 square centimeters-day
81	micrograms SO_2/square meter-day
90	tons/square mile-month[a]
91	milligrams/square centimeter-month[a]
92	micrograms/cubic meter-month[a]
98	milligrams SO_4^{2-}/square centimeters-30 days
99	milligrams/square centimeters-30 days

[a] On a calendar-month basis.

Unit Codes

Table 4

Appendix B - AQDHS DIAGNOSTIC MESSAGES

AQDHS DIAGNOSTIC MESSAGES

Most diagnostic message from AQDHS are self-explanatory. However, some messages refer the user to this document for further explanation. The message, with their explanation, are listed below.

*** WARNING: PARM TABLE OVERFLOW. CONSULT USER GUIDE FOR CORRECTIVE ACTION.

AQDHS currently has a built-in limit of 100 entries in the parameter, method, unit codes, minimum detectable table. This restriction is included to limit the amount of core utilized. If this message should ever appear, the limit has been exceeded. It is possible to increase the limit but only at the expense of using additional core. The following programs must be modified:

Parameter Code Table Maintenance
Transaction Editor
Detail List
Sliding Average
Data Analysis
Statistical List

Refer to Appendix C for further instructions on accomplishing these changes.

*** WARNING: SITE TABLE OVERFLOW. CONSULT USER GUIDE FOR CORRECTIVE ACTION.

AQDHS currently has a built-in limit of 100 entries in the valid site code table. This restriction is included to limit the amount of core utilized. If this message should ever appear, the limit has been exceeded. It is possible to increase the limit but only at the expense of using additional core. The following programs must be modified:

Site Code Table Maintenance
Transaction Editor
Detail List
Sliding Average
Statistical List

Refer to Appendix C for further instructions on accomplishing these changes.

*** WARNING: STANDARDS TABLE OVERFLOW. CONSULT USER GUIDE FOR CORRECTIVE ACTION.

AQDHS currently has a built-in limit of 50 entries in the parameter standards table. This restriction is included to limit the amount of core utilized. If this message should ever appear, the limit has been exceeded. It is possible to increase the limit but only at the expense of using additional core. The following programs must be modified:

Parameter Standards Table Maintenance
Detail List
Sliding Average
Statistical Liist

Refer to Appendix C for further instructions on accomplishing these changes.

Appendix C - PROGRAM MODIFICATION

INCREASING SIZE OF PARM CODE TABLE

Whenever it is desired to increase the size (number of entries) of the parameter, method, unit codes, minimum detectable table, it is necessary to modify the following programs:

Parameter Code Table Maintenance
Transaction Editor
Detail List
Sliding Average
Data Analysis
Statistical List

The current limit for the number of entries in the table is 100 entries in all programs except Data Analysis. The limit in the Data Analysis program is currently 120.

The following paragraphs discuss the changes to be made in each program described above.

- Parameter Code Table Maintenance

  Change the VALUE clause of the following sentence:
  ```
  77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 100.
  ```
  on or about sequence number 00016700, from 100 to the new limit of entries. For example:
  ```
  77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 150.
  ```
  will increase the limit to 150 entries.

- Transaction Editor

  Change the VALUE clause of the following sentence:
  ```
  77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 100.
  ```
  on or about sequence number 00037400, from 100 to the

new limit of entries. For example:

```
77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries. The value should be the same as that specified in the Parameter Code Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  PARM-CODE-KEY-ARRAY       OCCURS 100 TIMES.
```

on or about sequence number 00094300, from 100 to the new limit of entries. For example:

```
02  PARM-CODE-KEY-ARRAY       OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

o Detail List

Change the VALUE clause of the following sentence:

```
77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 100.
```

on or about sequence number 00024600, from 100 to the new limit of entries. For example:

```
77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries. The value should be the same as that specified in the Parameter Code Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  PARM-CODE-KEY-ARRAY       OCCURS 100 TIMES.
```

on or about sequence number 00118600, from 100 to the new limit of entries. For example:

```
02  PARM-CODE-KEY-ARRAY       OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

Change the OCCURS clause of the following sentence:

```
02  PARM-DESCRIPTION-TABLE    OCCURS 100 TIMES.
```

on or about sequence number 00123300, from 100 to the new limit of entries. For example:

```
02  PARM-DESCRIPTION-TABLE     OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

- o Sliding Average

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 100.
  ```

  on or about sequence number 00025000, from 100 to the new limit of entries. For example:

  ```
  77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 150.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified in the Parameter Code Table Maintenance program.

  Change the OCCURS clause of the following sentence:

  ```
  02  PARM-CODE-KEY-ARRAY        OCCURS 100 TIMES.
  ```

  on or about sequence number 00110500, from 100 to the new limit of entries. For example:

  ```
  02  PARM-CODE-KEY-ARRAY        OCCURS 150 TIMES.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

  Change the OCCURS clause of the following sentence:

  ```
  02  PARM-DESCRIPTION-TABLE     OCCURS 100 TIMES.
  ```

  on or about the sequence number 00115200, from 100 to the new limit of entries. For example:

  ```
  02  PARM-DESCRIPTION-TABLE     OCCURS 150 TIMES.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

- o Data Analysis

  Change the array size of the following variable:

  ```
  X         , MVDATA( 600 )
  ```

on or about sequence number 00015300, from 600 to the new limit of entries, incrementing by 5 for each new entry. The current value of 600 sets a limit of 120 entries. To increase the limit to 150 entries, enter a value of 750 as the array size. The number of entries should be greater than or equal to the value specified in the Parameter Code Table Maintenance program.

o Statistical List

Change the VALUE clause of the following sentence:

```
77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 100.
```

on or about sequence number 00023600, from 100 to the new limit of entries. For example:

```
77  NBR-OF-PARMS          PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries. The value should be the same as that specified in the Parameter Code Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  PARM-CODE-KEY-ARRAY        OCCURS 100 TIMES.
```

on or about sequence number 00086600, from 100 to the new limit of entries. For example:

```
02  PARM-CODE-KEY-ARRAY        OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

Change the OCCURS clause of the following sentence:

```
02  PARM-DESCRIPTION-TABLE     OCCURS 100 TIMES.
```

on or about sequence number 00122000, from 100 to the new limit of entries. For example:

```
02  PARM-DESCRIPTION-TABLE     OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-PARMS.

INCREASING SIZE OF SITE CODE TABLE

Whenever it is desired to increase the size (number of entries) of the valid site code table, it is necessary to modify the following programs:

Site Code Table Maintenance
Transaction Editor
Detail List
Sliding Average
Statistical List

The current limit for the number of entries in the table is 100 entries.

The following paragraphs discuss the changes to be made in each program described above.

o **Site Code Table Maintenance**
Change the VALUE clause of the following sentence:

```
77  NBR-OF-SITES          PIC 999 COMP SYNC VALUE 100.
```

on or about sequence number 00014800, from 100 to the new limit of entries. For example:

```
77  NBR-OF-SITES          PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries.

o **Transaction Editor**
Change the VALUE clause of the following sentence:

```
77  NBR-OF-SITES          PIC 999 COMP SYNC VALUE 100.
```

on or about sequence number 00037500, from 100 to the new limit of entries. For example:

```
77  NBR-OF-SITES          PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries. The value should be the same as that specified in the Site Code Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  SITE-CODE-TABLE        OCCURS 100 TIMES.
```

on or about sequence number 00095900, from 100 to the new limit of entries. For example:

```
02  SITE-CODE-TABLE        OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-SITES.

- Detail List

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-SITES           PIC 999 COMP SYNC VALUE 100.
  ```

  on or about sequence number 00024700, from 100 to the new limit of entries. For example:

  ```
  77  NBR-OF-SITES           PIC 999 COMP SYNC VALUE 150.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified in the Site Code Table Maintenance program.

  Change the OCCURS clause in the following sentence:

  ```
  02  SITE-CODE-TABLE        OCCURS 100 TIMES.
  ```

  on or about sequence number 00120200, from 100 to the new limit of entries. For example:

  ```
  02  SITE-CODE-TABLE        OCCURS 150 TIMES.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-SITES.

- Sliding Average

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-SITES           PIC 999 COMP SYNC VALUE 100.
  ```

  on or about sequence number 00025100, from 100 to the new limit of entries. For example:

  ```
  77  NBR-OF-SITES           PIC 999 COMP SYNC VALUE 150.
  ```

  will increase the limit to 150 entries. The value should be the same as that specified in the Site Code Table Maintenance program.

Change the OCCURS clause in the following sentence:

```
02  SITE-CODE-TABLE      OCCURS 100 TIMES.
```

on or about sequence number 00112100, from 100 to the new limit of entries. For example:

```
02  SITE-CODE-TABLE      OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified for NBR-OF-SITES.

o <u>Statistical List</u>

Change the VALUE clause in the following sentence:

```
77  NBR-OF-SITES         PIC 999 COMP SYNC VALUE 100.
```

on or about sequence number 00023700, from 100 to the new limit of entries. For example:

```
77  NBR-OF-SITES         PIC 999 COMP SYNC VALUE 150.
```

will increase the limit to 150 entries. The value should be the same as that specified in the Site Code Table Maintenance program.

Change the OCCURS clause in the following sentence:

```
02  SITE-CODE-TABLE      OCCURS 100 TIMES.
```

on or about sequence number 00088200, from 100 to the new limit of entries. For example:

```
02  SITE-CODE-TABLE      OCCURS 150 TIMES.
```

will increase the limit to 150 entries. The value should be the same as that specified in NBR-OF-SITES.

INCREASING SIZE OF PARAMETER STANDARDS TABLE

Whenever it is desired to increase the size (number of entries) of the parameter standards table, it is necessary to modify the following programs:

Parameter Standards Table Maintenance
Detail List
Sliding Average
Statistical List

The current limit for the number of entries in the table is 50.

The following paragraphs discuss the changes to be made in each program described above.

- o Parameter Standards Table Maintenance

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-STANDARDS      PIC 99 COMP SYNC VALUE 50.
  ```

  on or about sequence number 00016900, from 50 to the new limit of entries. For example:

  ```
  77  NBR-OF-STANDARDS      PIC 999 COMP SYNC VALUE 100.
  ```

  will increase the limit to 100 entires. Note that the picture clause must also be changed to allow for more digits.

- o Detail List

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-STANDARDS      PIC 99 COMP SYNC VALUE 50.
  ```

  on or about sequence number 00024800, from 50 to the new limit of entries. For example:

  ```
  77  NBR-OF-STANDARDS      PIC 999 COMP SYNC VALUE 100.
  ```

  will increase the limit to 100 entries. The value should be the same as that specified in the Parameter Standards Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  STANDARDS-TABLE       OCCURS 50 TIMES.
```

on or about sequence number 00124100, from 50 to the new limit of entries. For example:

```
02  STANDARDS-TABLE       OCCURS 100 TIMES.
```

will increase the limit to 100 entries. The value should be the same as that specified for NBR-OF-STANDARDS.

- Sliding Average

  Change the VALUE clause in the following sentence:

  ```
  77  NBR-OF-STANDARDS      PIC 99 COMP SYNC VALUE 50.
  ```

  on or about sequence number 00025200, from 50 to the new limit of entires. For example:

  ```
  77  NBR-OF-STANDARDS      PIC 999 COMP SYNC VALUE 100.
  ```

  will increase the limit to 100 entries. The value should be the same as that specified in the Parameter Standards Table Maintenance program.

  Change the OCCURS clause of the following sentence:

  ```
  02  STANDARDS-TABLE       OCCURS 50 TIMES.
  ```

  on or about sequence number 00116000, from 50 to the new limit of entries. For example:

  ```
  02  STANDARDS-TABLE       OCCURS 100 TIMES.
  ```

  will increase the limit to 100 entries. The value should be the same as that specified for NBR-OF-STANDARDS.

- Statistical List

  Change the VALUE clause of the following sentence:

  ```
  77  NBR-OF-STANDARDS      PIC 99 COMP SYNC VALUE 50.
  ```

  on or about sequence number 00023800, from 50 to the new limit of entries. For example:

  ```
  77  NBR-OF-STANDARDS      PIC 999 COMP SYNC VALUE 100.
  ```

  will increase the limit to 100 entries. The value should be the same as that specified in the Parameter Standards Table Maintenance program.

Change the OCCURS clause of the following sentence:

```
02  STANDARDS-TABLE      OCCURS 50 TIMES.
```

on or about sequence number 00122800, from 50 to the new limit of entries. For example:

```
02  STANDARDS-TABLE      OCCURS 100 TIMES.
```

will increase the limit to 100 entires. The value should be the same as that specified for NBR-OF-STANDARDS.

Appendix D - CATALOGED PROCEDURES

AQSACNVT - AQDHS FILE TO SAROAD INPUT

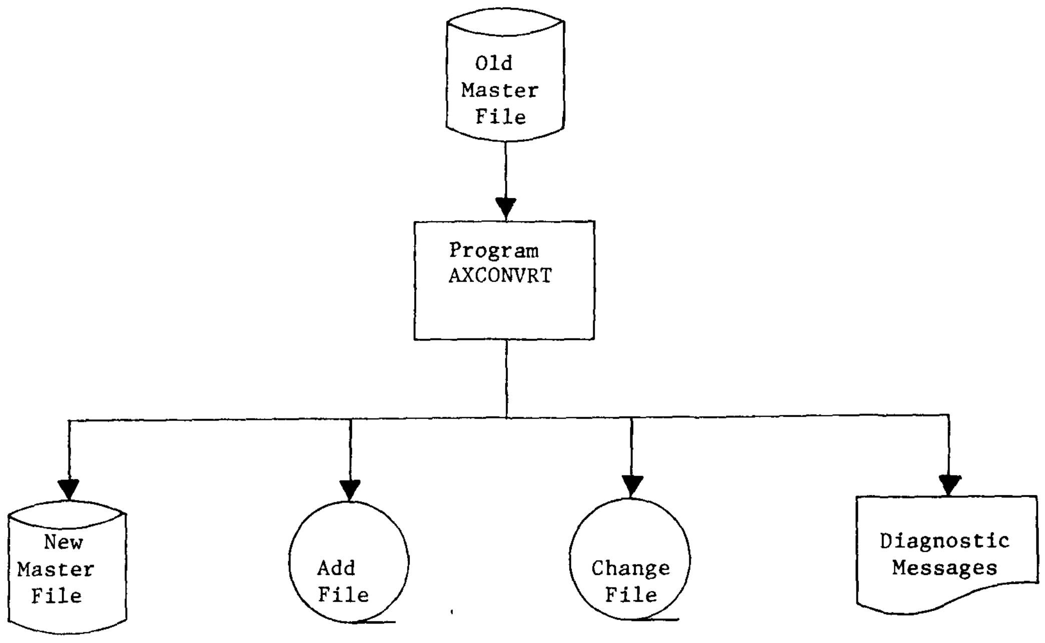
AQSACNVT is executed to convert the AQDHS master file to SAROAD input format. The input to the program consists of AQDHS master file records and is defined by DD name AQSMASTR. There are four output files created by the program. The first, defined by AQSNEWMS, contains the updated AQDHS master file. The second, defined by AQSDDFL, contains the add records. The third, defined by AQSCHGFL, contains the change records. The fourth, defined by AQSPRINT, contains diagnostic messages produced during the conversion process.

DD NAME	DESCRIPTION
AQSMASTR	AQDHS master file to be converted
AQSNEWMS	New AQDHS master file
AQSPRINT	Print file for diagnostic messages
AQSADDFL	SAROAD add file deferred to DDNAME ADDFILE
AQSCHGFL	SAROAD change file deferred to DDNAME CHNGEFL

AQSACNVT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	AXCONVRT	Program to convert AQDHS file to SAROAD input format
OLDMSTR	FTMSTRAA	Lowest level index of AQDHS file to be converted
NEWMSTR	FTMSTRAB	Lowest level index of new AQDHS file
UNIT	2314	Unit type upon which the new AQDHS master file is to reside
SERIAL	009858	Volume serial number of volume upon which AQDHS master file is to reside
DISP	'NEW, CATLG, DELETE'	Disposition of new AQDHS master file
SPCUNIT	TRK	Units in which space for the AQDHS master file is to be allocated
PRIMARY	20	Number of units to be allocated for the AQDHS master file's primary allocation
SECNDRY	10	Number of units to be allocated for the AQDHS master file's secondary allocation
OUT	A	Sysout class for all print files

AQSACNVT - Substitutable Parameters



AQSACNVT - Data Flow

AQSACNVT

```
//*             AQSACNVT                  BENFER TL         WF4   06/30/74
//*
//AQSACNVT PROC PROJECT='CDHS.AQS',
//              PROGRAM=AXCONVRT,
//              OLDMSTR=FTMSTRAA,
//              NEWMSTR=FTMSTRAB,
//              UNIT=2314,
//              SERIAL=C09858,
//              DISP='NEW,CATLG,DELETE',
//              SPCUNIT=TRK,
//              PRIMARY=20,
//              SECNDRY=10,
//              OUT=A
//*
//SUBMIT  EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* CONVERT AQDHS FILE TO SAROAD INPUT FORMAT
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - OLD AQDHS MASTER FILE
//*
//AQSMASTR  DD DSNAME=&PROJECT..DATA.&OLDMSTR,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - NEW AQDHS MASTER FILE
//*
//AQSNEWMS  DD UNIT=&UNIT,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//              DISP=(&DISP),
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//              DSNAME=&PROJECT..DATA.&NEWMSTR
//*
//* OUTPUT DATA SET - SAROAD ADD FILE
//*
//AQSADDFL  DD  DDNAME=ADDFILE
//*
//* OUTPUT DATA SET - SAROAD CHANGE FILE
//*
//AQSCHGFL  DD  DDNAME=CHNGEFL
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
```

AQSBCNVT - OLD AQDHS FILE TO NEW AQDHS INPUT

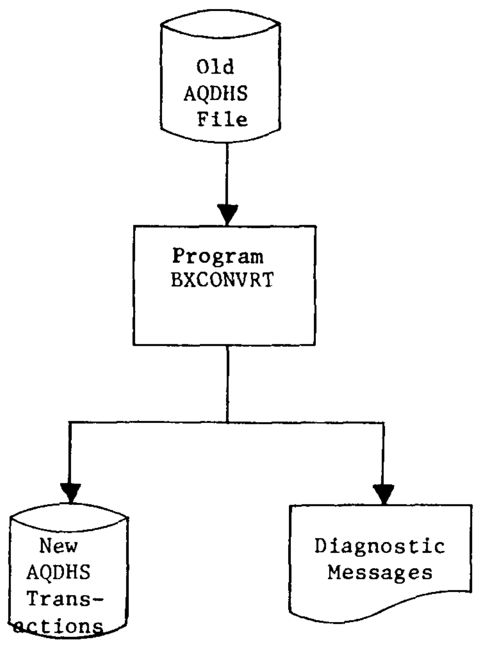
AQSBCNVT is executed to produce AQDHS input transactions from the old AQDHS master file. The input to the program consists of old AQDHS master file records and is defined by DD name AQSAROAD. There are two output files created by the program. The first, defined by AQSTRANS, contains the AQDHS input transactions. The second, defined by AQSPRINT, contains any diagnostic messages generated during the conversion process.

DD NAME	DESCRIPTION
AQSAROAD	Old AQDHS file
AQSTRANS	New AQDHS input transactions
AQSPRINT	Print file for diagnostic messages

AQSBCNVT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	BXCONVRT	Program to convert old AQDHS file to new AQDHS input transactions
OLDFILE	OLDAQDHS	Lowest level index of old AQDHS file
TEMP	SYSOUT	Unit type for temporary work space
PRIMARY	10	Number of units to be allocated for new AQDHS input transactions' primary allocation
SECNDRY	5	Number of units to be allocated for new AQDHS input transactions' secondary allocation
OUT	A	Sysout class for all print files

AQSBCNVT - Substitutable Parameters



AQSBCNVT - Data Flow

AQSBCNVT

```
//*            AQSBCNVT                   HURLEY RF         WF4  06/30/74
//*
//AQSBCNVT PROC PROJECT='CDHS.AQS',
//             PROGRAM=BXCONVRT,
//             OLDFILE=OLDAQDHS,
//             TEMP=SYSOUT,
//             SPCUNIT=TRK,
//             PRIMARY=10,
//             SECNDRY=5,
//             OUT=A
//*
//CONVERT EXEC PGM=&PROGRAM,
//             REGION=60K,
//             TIME=(1,0)
//*
//* CONVERT OLD AQDHS FILE TO NEW AQDHS INPUT TRANSACTIONS
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - OLD AQDHS FILE
//*
//AQSAROAD   DD DSNAME=&PROJECT..DATA.&OLDFILE,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - NEW AQDHS INPUT TRANSACTIONS
//*
//AQSTRANS   DD UNIT=&TEMP,
//             DISP=(NEW,PASS,DELETE),
//             SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//             DSNAME=&&OLDTRNS
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT   DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT   DD SYSOUT=&OUT
//*
//SYSOUT     DD SYSOUT=&OUT
//*
//SYSERCUT   DD SYSOUT=&OUT
//*
//SYSDTERM   DD SYSOUT=&OUT
//*
//SYSUDUMP   DD SYSOUT=&OUT
//*
```

AQSCNVRT - OLD AQDHS INPUT TO NEW AQDHS INPUT

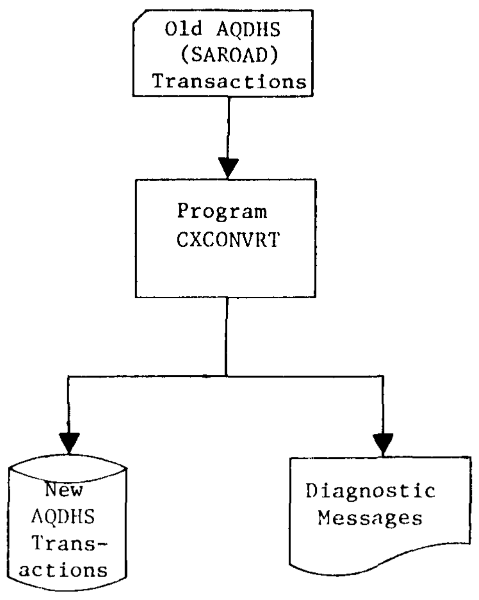
AQSCNVRT is executed to convert old format AQDHS (SAROAD) input transactions to new format AQDHS input transactions. The input to the conversion program consists of old format AQDHS input transactions and is defined by DD name AQSAROAD. There are two output files created by the program. The first, defined by AQSTRANS, contains the new format AQDHS transactions. The second, defined by AQSPRINT, contains any diagnostic messages generated during the conversion process.

DD NAME	DESCRIPTION
AQSAROAD	Old AQDHS (SAROAD) input transactions
AQSTRANS	New AQDHS transactions
AQSPRINT	Print file for diagnostic messages

AQSCNVRT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	CXCONVRT	Program to convert old AQDHS input format to new AQDHS input format
OUT	A	Sysout class for all print files

AQSCNVRT - Substitutable Parameters



AQSCNVRT - Data Flow

```
//*             AQSCNVRT                  HURLEY RF         WF4  06/30/74
//*
//AQSCNVRT PROC PROJECT='CDHS.AQS',
//              PROGRAM=CXCCNVRT,
//              OUT=A
//*
//CONVERT EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* CONVERT SAROAC INPUT FORMAT TO AQDHS INPUT FORMAT
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - SAROAC TRANSACTIONS
//*
//AQSAROAC   DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//* OUTPUT DATA SET - AQDHS TRANSACTIONS
//*
//AQSTRANS   DD DDNAME=OUTPUT
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT   DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT   DD SYSOUT=&OUT
//*
//SYSOUT     DD SYSOUT=&OUT
//*
//SYSDBOUT   DD SYSOUT=&OUT
//*
//SYSDTERM   DD SYSOUT=&OUT
//*
//SYSUDUMP   DD SYSOUT=&OUT
//*
```

AQSDANAL - DATA ANALYSIS

AQSDANAL is executed to perform various statistical analyses on the AQDHS master file or answer file. There are two input files used by this program. The first, defined by DD name FT08F001, contains the flagged master file or answer file. The second, defined by FT09F001, contains the flagged key portion of the parameter, method, unit codes table. There are two output files created by this program. The first, defined by FT10F001, contains the statistics file. The second, defined by FT06F001, contains any diagnostic messages generated during the analysis.

DD NAME	DESCRIPTION
FT08F001	AQDHS master file
FT09F001	Key portion of parameter, method, unit table
FT10F001	Statistics output file
FT06F001	Print file for diagnostic messages

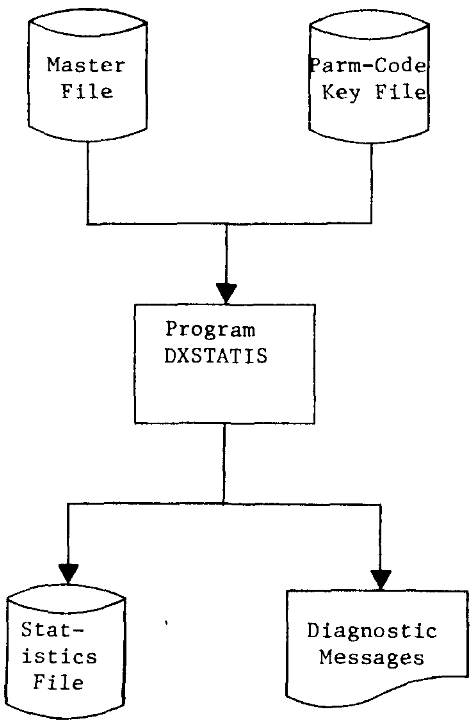
AQSDANAL - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	DXSTATIS	Program to perform statistical analyses of readings in AQDHS master file
MSTRFIL	MTMSTRAA	Lowest level index of AQDHS master file
PARMKFL	NTPRM1AA	Lowest level index of key portion of parameter, method, unit table
STATFIL	STMSTRAA	Lowest level index of statistics output file
UNIT	2314	Unit type upon which the statistics file is to reside
SERIAL	009858	Volume serial number of volume upon which statistics file is to reside
DISP	'NEW,CATLG DELETE'	Disposition of statistics file
SPCUNIT	TRK	Units in which space for the statistics file is to be allocated
PRIMARY	10	Number of units to be allocated for the statistics file's primary allocation

AQSDANAL - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
SECNDRY	5	Number of units to be allocated for the statistics file's secondary allocation
RECFM	FB	Recording format is fixed, blocked.
LRECL	141	Record length for statistics file
BLKSIZE	1692	Block size for statistics file
OUT	A	Sysout Class for all print files

AQSDANAL - Substitutable Parameters
(continued)



AQSDANAL - Data Flow

ACSDANAL

```
//*            AQSDANAL                  HURLEY RF        WF4   06/30/74
//*
//AQSDANAL PROC PROJECT='CDHS.AQS',
//             PROGRAM=DXSTATIS,
//             MSTRFIL=MTMSTRAA,
//             PARMKFL=NTPRM1AA,
//             STATFIL=STMSTRAA,
//             UNIT=2314,
//             SERIAL=C09858,
//             DISP='NEW,PASS,DELETE',
//             SPCUNIT=TRK,
//             PRIMARY=10,
//             SECNDRY=5,
//             RECFM=FB,
//             LRECL=141,
//             BLKSIZE=1692,
//             OUT=A
//*
//ANALYZE EXEC PGM=&PROGRAM,
//             REGION=100K,
//             TIME=(1,0)
//*
//* PERFORM STATISTICAL ANALYSIS OF READINGS IN AQCHS MASTER FILE
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - AQCHS MASTER FILE
//*
//FT08F001  DD DSNAME=&PROJECT..DATA.&MSTRFIL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - KEY PORTION OF PARAMETER, METHOD, UNIT TABLE
//*
//FT09F001  DD DSNAME=&PROJECT..DATA.&PARMKFL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - STATISTICS FILE
//*
//FT10F001  DD UNIT=&UNIT,
//             VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//             DISP=(&DISP),
//             SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//             DSNAME=&PROJECT..DATA.&STATFIL,
//             DCB=(RECFM=&RECFM,LRECL=&LRECL,BLKSIZE=&BLKSIZE)
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//FT06F001  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
```

00005900

AQSDANAL //*

AQSELIST - DETAIL LIST

AQSELIST is executed to produce a detailed listing of the contents of the AQDHS master file. There are six input files used by the program. The first, defined by DD name AQSINPUT, contains the control card required by the program. The second, defined by AQSMASTR, contain the master or answer file to be listed. The third, defined by AQSPARMK, contains the key portion of the parameter, method, unit codes table. The fourth, defined by AQSPARMD, contains the description portion of the parameter, method, unit codes table. The fifth, defined by AQSSITES, contains the valid site codes table. The sixth, defined by AQSTNDRD, contains the parameter standards table. There is one output file created by the program. This file, defined by AQSPRINT, contains the detailed report listing and any diagnostic messages generated during the reporting process.

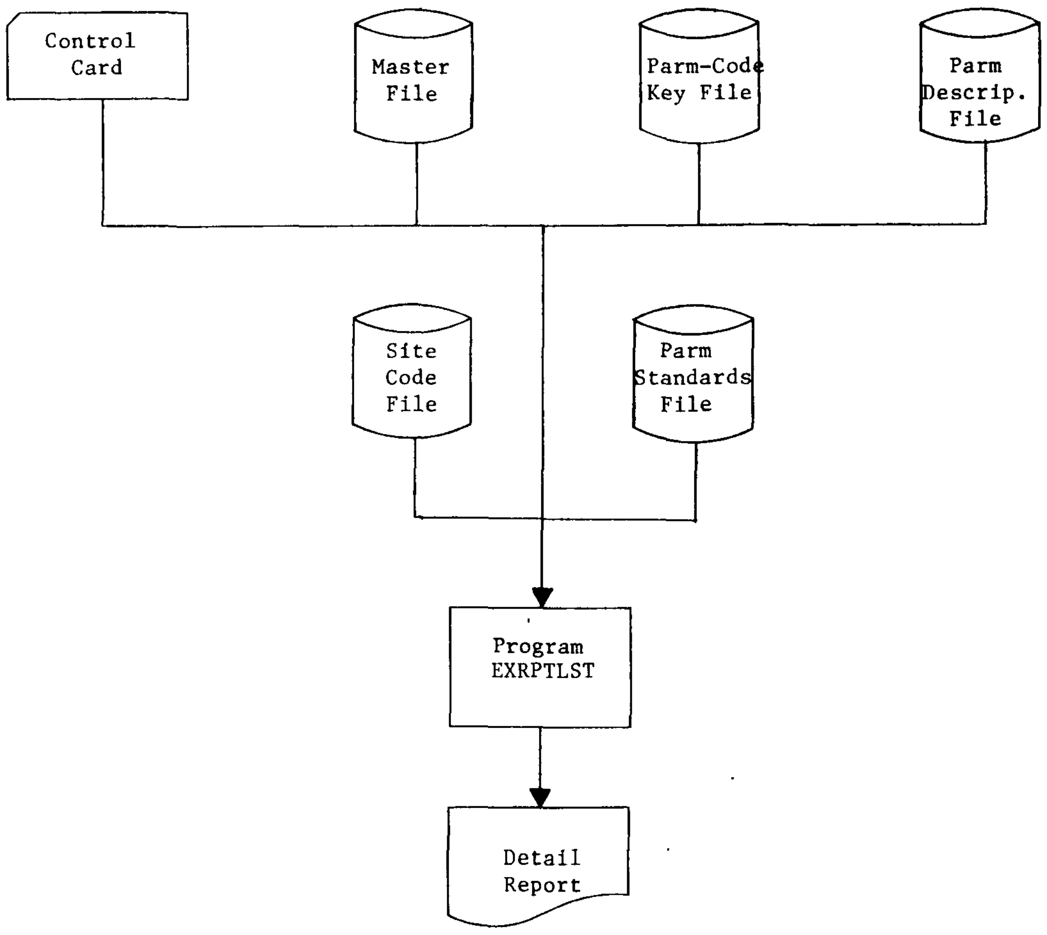
Figure 7.1.4 describes the format of the control card.

DD NAME	DESCRIPTION
AQSINPUT	AQDHS control card
AQSMASTR	AQDHS master file
AQSPARMK	Key portion of parameter, method, unit table
AQSPARMD	Description portion of table
AQSSITES	Valid site codes table
AQSTNDRD	Parameter standards table
AQSPRINT	Print file for report listing

AQSELIST - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	EXRPTLST	This program lists the contents of the AQDHS master file
MSTRFIL	FTMSTRAA	Lowest level index of the AQDHS master file
PARMKFL	HTPRM1AA	Lowest level index of the key portion of parameter, method, unit table
PARMDFL	HTPRM2AA	Lowest level index of the description portion of the above table
SITEFIL	HTSITEAA	Lowest level index of the valid site codes table
STNDFIL	HTSTNDAA	Lowest level index of the parameter standards table
OUT	A	Sysout class for all print files

AQSELIST - Substitutable Parameters



AQSELIST - Data Flow

```
//*             AQSELIST                HURLEY RF         WF4  06/30/74
//*
//AQSELIST PROC PROJECT='CDHS.ACS',
//              PROGRAM=EXRPTLST,
//              MSTRFIL=FTMSTRAA,
//              PARMKFL=HTPRM1AA,
//              PARMDFL=HTPRM2AA,
//              SITEFIL=HTSITEAA,
//              STNDFIL=HTSTNDAA,
//              OUT=A
//*
//REPORT   EXEC PGM=&PROGRAM,
//              REGION=100K,
//              TIME=(1,0)
//*
//* LIST CONTENTS OF ACDHS MASTER FILE
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - CONTROL CARD
//*
//AQSINPUT   DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//* INPUT DATA SET - ACDHS MASTER FILE
//*
//AQSMASTR   DD DSNAME=&PROJECT..DATA.&MSTRFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - KEY PORTION OF PARAMETER, METHOD, UNIT TABLE
//*
//AQSPARMK   DD DSNAME=&PROJECT..DATA.&PARMKFL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - DESCRIPTION PORTION OF TABLE
//*
//AQSPARMD   DD DSNAME=&PROJECT..DATA.&PARMDFL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - VALID SITE CODE TABLE
//*
//AQSSITES   DD DSNAME=&PROJECT..DATA.&SITEFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - PARAMETER STANDARDS TABLE
//*
//AQSTNDRD   DD DSNAME=&PROJECT..DATA.&STNDFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
```

AQSFLIST

```
//* OUTPUT DATA SET - REPORT LISTING                                  00005900
//*                                                                   00006000
//AQSPRINT  DD SYSOUT=&OUT                                            00006100
//*                                                                   00006200
//* OUTPUT DATA SETS - SYSTEM OPERATION                               00006300
//*                                                                   00006400
//SYSPRINT  DD SYSOUT=&OUT                                            00006500
//*                                                                   00006600
//SYSOUT    DD SYSOUT=&OUT                                            00006700
//*                                                                   00006800
//SYSCBOUT  DD SYSOUT=&OUT                                            00006900
//*                                                                   00007000
//SYSDTERM  DD SYSOUT=&OUT                                            00007100
//*                                                                   00007200
//SYSUDUMP  DD SYSOUT=&OUT                                            00007300
//*                                                                   00007400
```

AQSFUPDT - FILE MAINTENANCE

AQSFUPDT is executed to create and maintain the AQDHS master file. The process occurs in three stages. First, the input transactions are edited and converted to an internal format. Next, these internal transactions are sorted into file sequence. Finally, the File Maintenance program reads the sorted transactions and uses them to create or maintain the master file.

There are three input files used in the editing phase. The first, defined by DD name AQSTRANS, contains the transactions to be edited. The second, defined by AQSPARMS, contains the key portion of the parameter, method, unit codes table. The third, defined by AQSSITES, contains the valid site codes table. There are two output files created during the editing phase. The first, defined by AQSINTRN, contains the edited internal format transactions. The second, defined by AQSPRINT, contains any diagnostic messages generated during the editing process.

Figure 2.2-2.1 shows the formats of the file maintenance transactions.

There are two input files used by the maintenance phase. The first, defined by AQSINTRN, contains the sorted internal format transactions. The second, defined by AQSOLDMS, contains the old AQDHS master file to be updated. There are four output files created during the maintenance phase. The first, defined by AQSNEWMS, contains the new AQDHS master file. The second, defined by AQSPRINT, contains any diagnostic messages generated during the maintenance process. The third, defined by AQSOLDRC, contains a listing of any old master file records which were changed or deleted. The fourth, defined by AQSNEWRC, contains a listing of any new master file records added or changed.

DD NAME	DESCRIPTION
AQSTRANS	AQDHS input transactions
AQSPARMS	Key portion of valid parm, method, unit table
AQSSITES	Valid site codes table
AQSINTRN	AQDHS output internal transactions
AQSPRINT	Print file for diagnostic messages
SORTIN	AQDHS internal transactions
SORTOUT	Sorted AQDHS internal transactions
AQSINTRN	Sorted AQDHS internal transactions
AQSOLDMS	Old AQDHS master file to be updated
AQSNEWMS	Updated AQDHS master file
AQSPRINT	Print file for diagnostic messages
AQSOLDRC	Old master listing
AQSNEWRC	New master listing

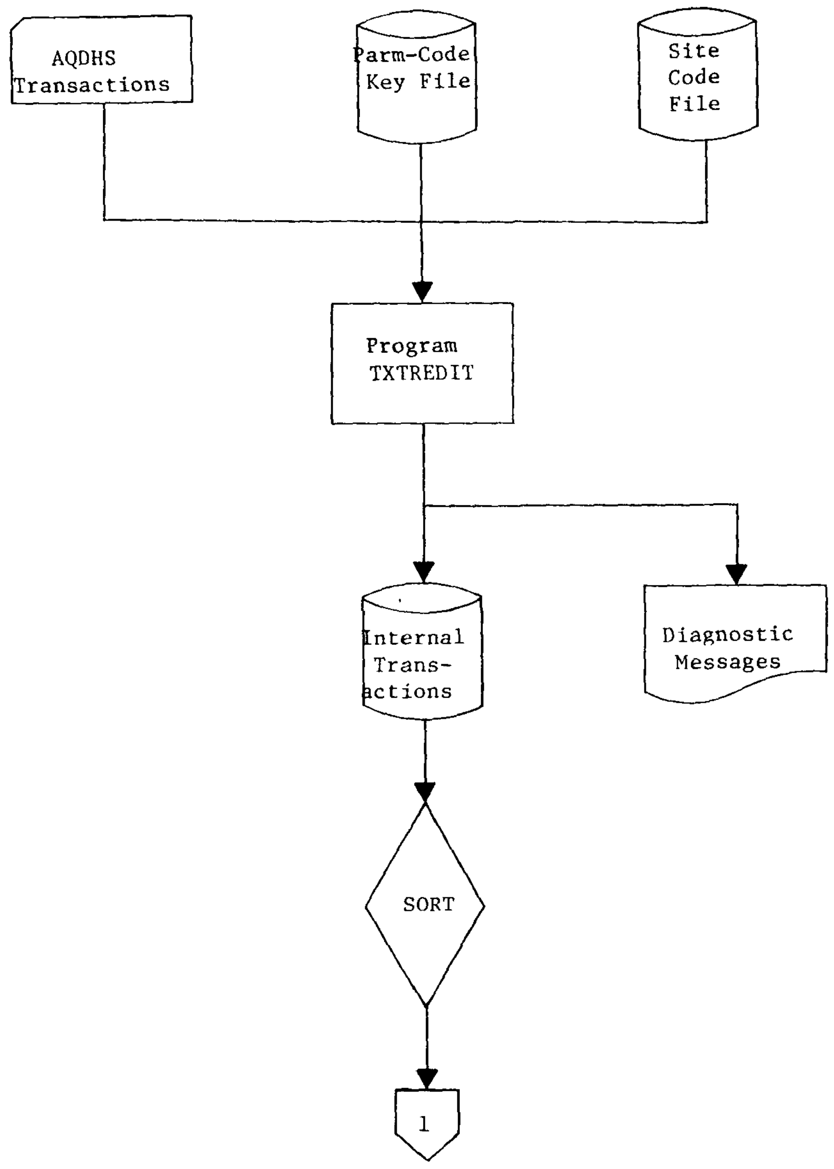
AQSFUPDT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROG1	TXTREDIT	Program to edit AQDHS file maintenance transactions
PROG2	FXFILMNT	Program to update AQDHS master file
OLDMSTR	FTMSTRAA	Lowest level index of AQDHS file to be updated
NEWMSTR	FTMSTRAB	Lowest level index of new AQDHS master file.
PARMKFL	HTPRM1AA	Lowest level index of the key portion of valid parm, method, unit table
SITEFIL	HTSITEAA	Lowest level index of the valid site codes table
UNIT	2314	Unit type upon which the new AQDHS master file is to reside
SERIAL	009858	Volume serial number of volume upon which AQDHS master file is to reside
DISP	'NEW,CATLG, DELETE'	Disposition of new AQDHS master file
SPCUNIT	TRK	Units in which space for the AQDHS master file is to be allocated

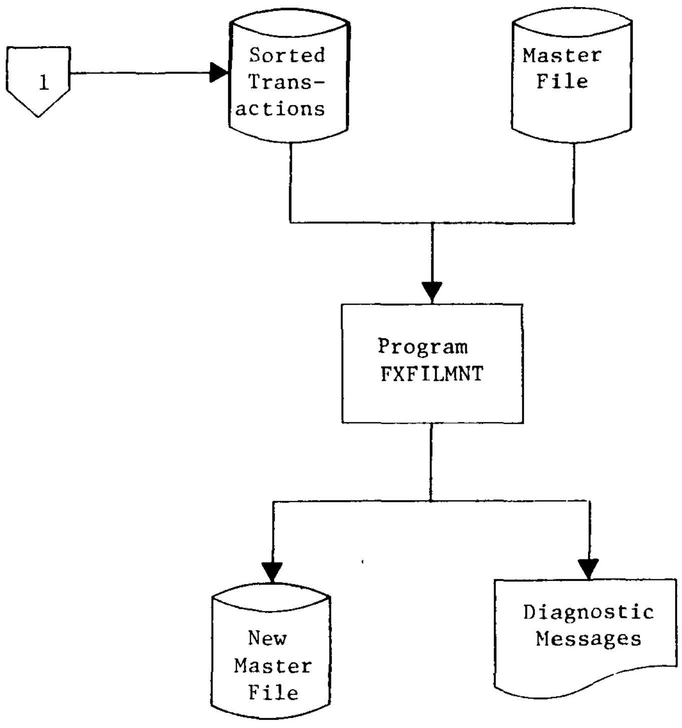
AQSFUPDT - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PRIMARY	20	Number of units to be allocated for the AQDHS master file's primary allocation
SECNDRY	10	Number of units to be allocated for the AQDHS master file's secondary allocation
TEMP	SYSOUT	Unit type for temporary work space
WORKSPC	50	Number of units to be allocated for temporary work space
SORTCTL	TSORTCTL	Sort control card
OUT	A	Sysout class for all print files

AQSFUPDT - Substitutable Parameters
(continued)



AQSFUPDT - Data Flow



AQSFUPDT - Data Flow
(continued)

AQSFUPDT

```
//*              AQSFUPDT                    HURLEY RF          WF4   06/30/74
//*
//AQSFUPDT PROC PROJECT='CDHS.AQS',
//               PROC1=TXTREDIT,
//               PROG2=FXFILMNT,
//               OLDMSTR=FTMSTRAA,
//               NEWMSTR=FTMSTRAB,
//               PARMKFL=HTPRMIAA,
//               SITEFIL=HTSITEAA,
//               UNIT=2314,
//               SERIAL=CO9858,
//               DISP='NEW,CATLG,DELETE',
//               SPCUNIT=TRK,
//               PRIMARY=20,
//               SECNDRY=10,
//               TEMP=SYSOUT,
//               WORKSPC=50,
//               SORTCTL=TSORTCTL,
//               OUT=A
//*
//EDIT     EXEC PGM=&PROG1,
//               REGION=60K,
//               TIME=(1,0)
//*
//* EDIT AQDHS FILE MAINTENANCE TRANSACTIONS
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//               VOLUME=(PRIVATE,RETAIN),
//               DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//               DISP=(SHR,PASS)
//*
//* INPUT DATA SET - AQDHS TRANSACTIONS
//*
//AQSTRANS  DD DDNAME=INPUT,
//               DCB=BLKSIZE=80
//*
//* INPUT DATA SET - KEY PORTION OF VALID PARM, METH, UNIT TABLE
//*
//AQSPARMS  DD DSNAME=&PROJECT..DATA.&PARMKFL,
//               VOLUME=(PRIVATE,RETAIN),
//               DISP=(SHR,PASS)
//*
//* INPUT DATA SET - VALID SITE CODE TABLE
//*
//AQSSITES  DD DSNAME=&PROJECT..DATA.&SITEFIL,
//               VOLUME=(PRIVATE,RETAIN),
//               DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - AQDHS INTERNAL TRANSACTIONS
//*
//AQSINTRN  DD UNIT=&TEMP,
//               DISP=(NEW,PASS,DELETE),
//               SPACE=(TRK,(&WORKSPC),RLSE),
//               DSNAME=&&TRANS
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
```

AQSFUPDT

```
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
//SORT    EXEC PGM=IERRCO00,
//             REGION=60K,
//             TIME=(1,0)
//*
//* SORT INTERNAL TRANSACTIONS INTO FILE SEQUENCE
//*
//SORTLIB   DD DSNAME=SYS1.SORTLIB,
//             DISP=(SHR,PASS)
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SORTWK01  DD UNIT=&TEMP,
//             SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTWK02  DD UNIT=(&TEMP,SEP=SORTWK01),
//             SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTWK03  DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02)),
//             SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTIN    DD DSNAME=&&TRANS,
//             DISP=(OLD,DELETE)
//*
//SORTOUT   DD UNIT=&TEMP,
//             DISP=(NEW,PASS,DELETE),
//             SPACE=(TRK,(&WORKSPC),RLSE),
//             DSNAME=&&SORTED,
//             DCB=(RECFM=FB,LRECL=68,BLKSIZE=1632)
//*
//SYSIN     DD DSNAME=&PROJECT..SYSIN(&SORTCTL),
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//UPDATE  EXEC PGM=&PROG2,
//             REGION=60K,
//             TIME=(1,0)
//*
//* MAINTAIN AQDHS MASTER FILE
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//             DISP=(SHR,PASS)
```

AQSFUPDT

```
//*
//* INPUT DATA SET - SORTED INTERNAL TRANSACTIONS
//*
//AQSINTRN  DD DSNAME=&&SORTED,
//             DISP=(OLD,DELETE)
//*
//* INPUT DATA SET - AQDHS OLD MASTER FILE
//*
//AQSOLDMS  DD DSNAME=&PROJECT..DATA.&OLDMSTR,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - AQDHS NEW MASTER FILE
//*
//AQSNEWMS  DD UNIT=&UNIT,
//             VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//             DISP=(&DISP),
//             SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//             DSNAME=&PROJECT..DATA.&NEWMSTR
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SET - OLD MASTER LISTING
//*
//AQSOLDPC  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SET - NEW MASTER LISTING
//*
//AQSNEWRC  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSCTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSHUPD1 - PARAMETER CODE TABLE MAINTENANCE

AQSHUPD1 is executed to create or maintain the parameter, method, unit codes tables. The process occurs in two phases. First, the transactions are sorted into table sequence and then the actual maintenance takes place.

There are three input files used in the maintenance phase. The first, defined by DD name AQSCARDS, contains the sorted table maintenance transactions. The second, defined by AQSOLDT1, contains the key portion of the parameter, method, unit codes table to be updated. The third, defined by AQSOLDT2, contains the description portion of the same table. Three output files are created during this phase. The first, defined by AQSNEWT1, contains the key portion of the new parameter, method, unit codes table. The second, defined by AQSNEWT2, contains the description portion of the new table. The third, defined by AQSPRINT, contains any diagnostic messages generated during the table maintenance.

Figure 4.1.4 shows the formats of the parameter code table maintenance transactions.

DD NAME	DESCRIPTION
AQSCARDS	Control card
AQSOLDT1	Old master key portion parameter code table
AQSOLDT2	Old master description portion parameter code table
AQSNEWT1	New master key portion parameter code table
AQSNEWT2	New master description portion parameter code table
AQSPRINT	Print file for diagnostic messages

AQSHUPD1 - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	HXTABLE1	Program to generate parameter code table
OLDMST1	HTPRM1AA	Lowest level index of old key portion parameter code table
OLDMST2	HTPRM2AA	Lowest level index of old description portion parameter code table
NEWMST1	HTPRM1AB	Lowest level index of new key portion parameter code table
NEWMST2	HTPRM2AB	Lowest level index of new description portion parameter code table
UNIT1	2314	Unit type upon which the new key portion parameter code table is to reside
UNIT2	2314	Unit type upon which the new description is to reside
SERIAL1	009858	Volume serial number of volume upon which key portion parameter code table is to reside
SERIAL2	009858	Volume serial number of volume upon which description portion parameter code table is to reside

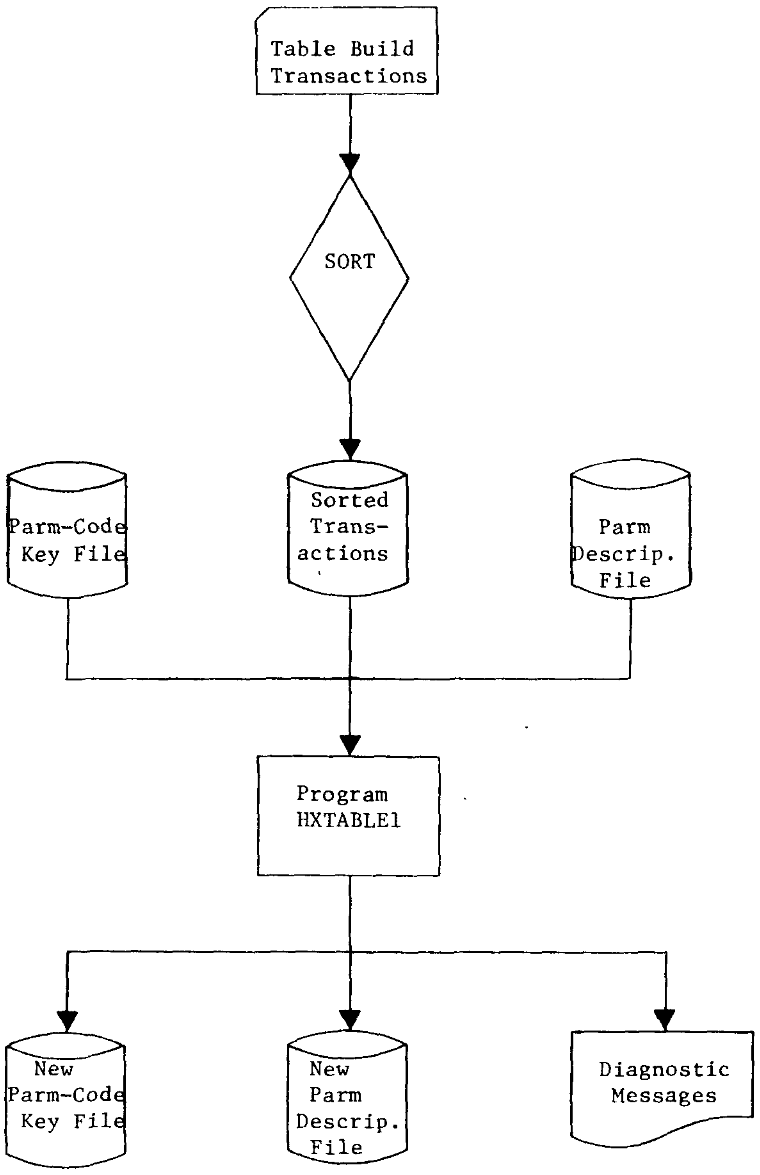
AQSHUPD1 - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
DISP1	'NEW,CATLG, DELETE'	Disposition of new key portion parameter code table
DISP2	'NEW,CATLG, DELETE'	Disposition of new description portion parameter code table
SPUNIT1	TRK	Units in which space for the key portion parameter code table is to be allocated
SPUNIT2	TRK	Units in which space for the description portion parameter code table is to be allocated
PRIMRY1	10	Number of units to be allocated for the key portion parameter code table's primary allocation
PRIMRY2	10	Number of units to be allocated for the description portion parameter code table's primary allocation
SCNDRY1	5	Number of units to be allocated for the key portion parameter code table's secondary allocation
SCNDRY2	5	Number of units to be allocated for the description portion parameter code table's secondary allocation

AQSHUPD1 - Substitutable Parameters
(continued)

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
TEMP	SYSOUT	Unit type for temporary work space
SORTSPC	10	Number of units to be allocated for the sort work space
SORTCTL	HSRTCTL1	Sort control card
OUT	A	Sysout class for all print files

AQSHUPD1 - Substitutable Parameters
(continued)



AQSHUPD1 - Data Flow

AQSHUPD1

```
//*              AQSHUPD1                HURLEY RF          WF4    06/30/74
//*
//AQSHUPD1 PROC PROJECT='CDHS.AQS',
//               PROGRAM=HXTABLE1,
//               OLDMST1=HTPRM1AA,
//               OLDMST2=HTPRM2AA,
//               NEWMST1=HTPRM1AB,
//               NEWMST2=HTPRM2AB,
//               UNIT1=2314,
//               UNIT2=2314,
//               SERIAL1=009858,
//               SERIAL2=009858,
//               DISP1='NEW,CATLG,DELETE',
//               DISP2='NEW,CATLG,DELETE',
//               SPUNIT1=TRK,
//               SPUNIT2=TRK,
//               PRIMRY1=10,
//               PRIMRY2=10,
//               SCNDRY1=5,
//               SCNDRY2=5,
//               TEMP=SYSOUT,
//               SORTSPC=10,
//               SORTCTL=HSRTCTL1,
//               OUT=A
//*
//SORT     EXEC PGM=IERRCO00,
//               REGION=60K,
//               TIME=(1,0)
//*
//* SORT INCOMING TRANSACTIONS INTO FILE SEQUENCE
//*
//SORTLIB   DD DSNAME=SYS1.SORTLIB,
//               DISP=(SHR,PASS)
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SORTWK01  DD UNIT=&TEMP,
//               SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK02  DD UNIT=(&TEMP,SEP=SORTWK01),
//               SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK03  DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02)),
//               SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTIN    DD DDNAME=INPUT,
//               DCB=BLKSIZE=80
//*
//SORTOUT   DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02,SORTWK03)),
//               DISP=(NEW,PASS,DELETE),
//               SPACE=(TRK,&SORTSPC,RLSE),
//               DSNAME=&&TRANS,
//               DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//*
//SYSIN     DD DSNAME=&PROJECT..SYSIN(&SORTCTL),
//               VOLUME=(PRIVATE,RETAIN),
//               DISP=(SHR,PASS)
//*
```

AQSHUPD1

```
//UPDATE   EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* MAINTAIN PARAMETER, METHOD, UNIT, MIN DETECTABLE TABLES
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.CORSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - CONTROL CARD AND, OPTIONALLY, TRANSACTIONS
//*
//AQSCARDS  DD DSNAME=&&TRANS,
//              DISP=(OLD,DELETE)
//*
//* INPUT DATA SET - OLD MASTER KEY PORTION TABLE
//*
//AQSOLDT1  DD DSNAME=&PROJECT..DATA.&OLDMST1,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - OLD MASTER DESCRIPTION PORTION TABLE
//*
//AQSOLDT2  DD DSNAME=&PROJECT..DATA.&OLDMST2,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - NEW MASTER KEY PORTION TABLE
//*
//AQSNEWT1  DD UNIT=&UNIT1,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL1),
//              DISP=(&DISP1),
//              SPACE=(&SPUNIT1,(&PRIMRY1,&SCNDRY1),RLSE),
//              DSNAME=&PROJECT..DATA.&NEWMST1
//*
//* OUTPUT DATA SET - NEW MASTER DESCRIPTION PORTION TABLE
//*
//AQSNEWT2  DD UNIT=&UNIT2,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL2),
//              DISP=(&DISP2),
//              SPACE=(&SPUNIT2,(&PRIMRY2,&SCNDRY2),RLSE),
//              DSNAME=&PROJECT..DATA.&NEWMST2
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
```

```
ACSHUPD1
          //*                                                         00011700
          //SYSUDUMP  DD SYSOUT=&OUT                                  00011800
          //*                                                         00011900
```

AQSHUPD2 - SITE CODE TABLE MAINTENANCE

AQSHUPD2 is executed to create and maintain the valid site codes table. The process occurs in two stages. First, the transactions are sorted into table sequence and then the actual maintenance takes place.

The maintenance phase uses two input files. The first, defined by DD name AQSTRANS, contains the sorted transactions. The second, defined by AQSOLDMS, contains the old valid site codes table. There are two output files created during the maintenance phase. The first, defined by AQSNEWMS, contains the new valid site codes table. The second, defined by AQSPRINT, contains any diagnostic messages generated during the maintenance process.

Figure 4.2.4 shows the format of the site code table maintenance transaction.

DD NAME	DESCRIPTION
AQSTRANS	Site code input transactions
AQSOLDMS	Old valid site codes table
AQSNEWMS	New valid site codes table
AQSPRINT	Print file for diagnostic messages

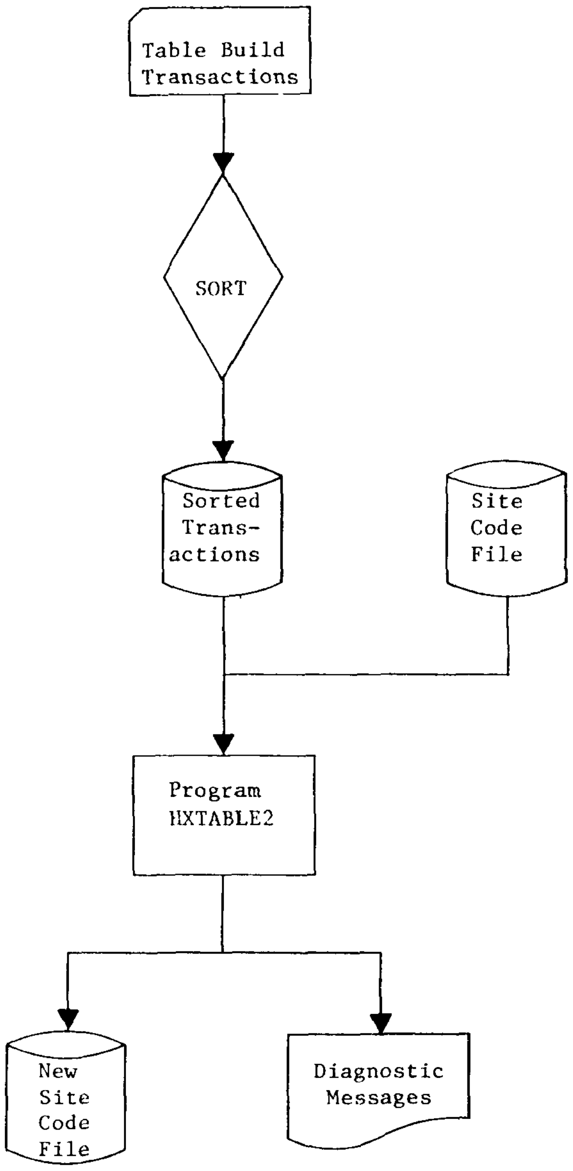
AQSHUPD2 - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	HXTABLE2	Program to generate valid site codes table
OLDMSTR	HTSITEAA	Lowest level index of the valid site codes table
NEWMSTR	HTSITEAB	Lowest level index of the new valid site codes table
UNIT	2314	Unit type upon which the new valid site codes table is to reside
SERIAL	009858	Volume serial number of volume upon which valid site codes table is to reside
DISP	'NEW,CATLG, DELETE'	Disposition of new valid site codes table
SPCUNIT	TRK	Units in which space for the valid site codes table is to be allocated
PRIMARY	10	Number of units to be allocated for the valid site codes table's primary allocation
SECNDRY	5	Number of units to be allocated for the valid site codes table's secondary allocation

AQSHUPD2 - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
TEMP	SYSOUT	Unit type for temporary work space
SORTSPC	10	Number of units to be allocated for the sort work space
SORTCTL	HSRTCTL2	Sort control card
OUT	A	Sysout class for all print files

AQSHUPD2 - Substitutable Parameters
(continued)



AQSHUPD2 - Data Flow

AQSHUPD2

```
//*            AQSHUPD2                HURLEY RF          WF4   06/30/74
//*
//AQSHUPD2 PROC PROJECT='CDHS.AQS',
//              PROGRAM=HXTABLE2,
//              OLDMSTR=HTSITEAA,
//              NEWMSTR=HTSITEAB,
//              UNIT=2314,
//              SERIAL=C09858,
//              DISP='NEW,CATLG,DELETE',
//              SPCUNIT=TRK,
//              PRIMARY=10,
//              SECNDRY=5,
//              TEMP=SYSCUT,
//              SORTSPC=10,
//              SORTCTL=HSRTCTL2,
//              OUT=A
//*
//SORT     EXEC PGM=IERRCO00,
//              REGION=60K,
//              TIME=(1,0)
//*
//* SORT INCOMING TRANSACTIONS INTO FILE SEQUENCE
//*
//SORTLIB   DD DSNAME=SYS1.SORTLIB,
//              DISP=(SHR,PASS)
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SORTWK01  DD UNIT=&TEMP,
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK02  DD UNIT=(&TEMP,SEP=SORTWK01),
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK03  DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02)),
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTIN    DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//SORTOUT   DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02,SORTWK03)),
//              DISP=(NEW,PASS,DELETE),
//              SPACE=(TRK,&SORTSPC,RLSE),
//              DSNAME=&&TRANS,
//              DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//*
//SYSIN     DD DSNAME=&PROJECT..SYSIN(&SORTCTL),
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//UPDATE  EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* MAINTAIN VALID SITE CODE TABLE
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
```

ACSHUPC2

```
//              DISP=(SHR,PASS)                                           00005900
//           DD DSNAME=SYS1.ANS.COBSUBR,                                  00006000
//              DISP=(SHR,PASS)                                           00006100
//*                                                                       00006200
//* INPUT DATA SET - TRANSACTIONS                                         00006300
//*                                                                       00006400
//AQSTRANS  DD DSNAME=&&TRANS,                                            00006500
//              DISP=(OLD,DELETE)                                         00006600
//*                                                                       00006700
//* INPUT DATA SET - OLD VALID SITE CODE TABLE                            00006800
//*                                                                       00006900
//AQSOLDMS  DD DSNAME=&PROJECT..DATA.&OLDMSTR,                            00007000
//              VOLUME=(PRIVATE,RETAIN),                                  00007100
//              DISP=(SHR,PASS)                                           00007200
//*                                                                       00007300
//* OUTPUT DATA SET - NEW VALID SITE CODE TABLE                           00007400
//*                                                                       00007500
//AQSNEWMS  DD UNIT=&UNIT,                                                00007600
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),                      00007700
//              DISP=(&DISP),                                             00007800
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),                00007900
//              DSNAME=&PROJECT..DATA.&NEWMSTR                            00008000
//*                                                                       00008100
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES                                 00008200
//*                                                                       00008300
//AQSPRINT  DD SYSOUT=&OUT                                                00008400
//*                                                                       00008500
//* OUTPUT DATA SETS - SYSTEM OPERATION                                   00008600
//*                                                                       00008700
//SYSPRINT  DD SYSOUT=&OUT                                                00008800
//*                                                                       00008900
//SYSOUT    DD SYSOUT=&OUT                                                00009000
//*                                                                       00009100
//SYSDBOUT  DD SYSOUT=&OUT                                                00009200
//*                                                                       00009300
//SYSOTERM  DD SYSOUT=&OUT                                                00009400
//*                                                                       00009500
//SYSUDUMP  DD SYSOUT=&OUT                                                00009600
//*                                                                       00009700
```

AQSHUPD3 - PARAMETER STANDARDS TABLE MAINTENANCE

AQSHUPD3 is executed to create and maintain the parameter standards table. The process occurs in two phases. First, the transactions are sorted into table sequence and then the actual maintenance takes place.

The maintenance phase uses two input files. The first, defined by DD name AQSCARDS, contains the sorted transactions. The second, defined by AQSOLDTB, contains the old parameter standards table. There are two output files created during the maintenance phase. The first, defined by AQSSTDTB, contains the new parameter standards table. The second, defined by AQSPRINT, contains any diagnostic messages generated during the maintenance process.

Figure 4.3.4 shows the format of the parameter standards table maintenance transaction.

DD NAME	DESCRIPTION
AQSCARDS	Parameter standards input transactions
AQSOLDTB	Old standards table
AQSSTDTB	New standards table
AQSPRINT	Print file for diagnostic messages

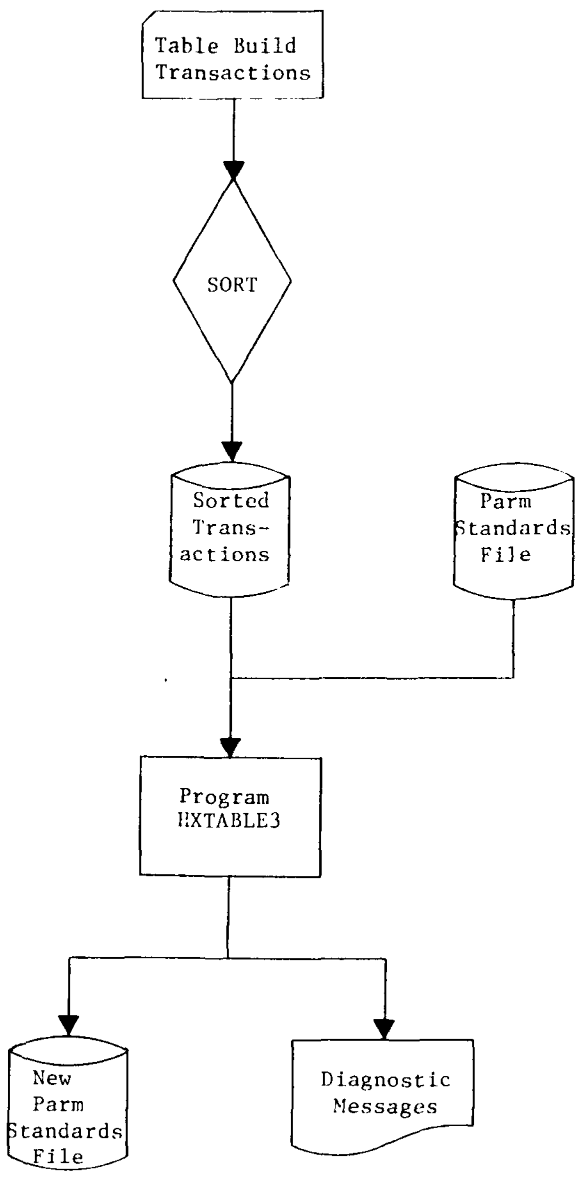
AQSHUPD3 - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	HXTABLE3	Program to generate standards table
OLDMSTR	HTSTNDAA	Lowest level index of the standards table
UNIT	2314	Unit type upon which the new standards table is to reside
SERIAL	009858	Volume serial number of volume upon which standards table is to reside
DISP	'NEW,CATLG, DELETE'	Disposition of the new standards table
SPCUNIT	TRK	Units in which space for the standards is to be allocated
PRIMARY	10	Number of units to be allocated for standards table's primary allocation
SECNDRY	5	Number of units to be allocated for standards table's secondary allocation
TEMP	SYSOUT	Unit type for temporary work area
SORTSPC	10	Number of units to be allocated for the sort work space

AQSHUPD3 - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
SORTCTL	HSRTCTL3	Sort control card
OUT	A	Sysout class for all print files

AQSHUPD3 - Substitutable Parameters
(continued)



AQSHUPD3 - Data Flow

```
//*           AQSHUPD3                HURLEY RF        WF4   06/30/74
//*
//AQSHUPD3 PROC PROJECT='CDHS.AQS',
//              PROGRAM=HXTABLE3,
//              OLDMSTR=HTSTNDAA,
//              NEWMSTR=HTSTNDAB,
//              UNIT=2314,
//              SERIAL=CC9858,
//              DISP='NEW,CATLG,DELETE',
//              SPCUNIT=TRK,
//              PRIMARY=10,
//              SECNDRY=5,
//              TEMP=SYSOUT,
//              SORTSPC=10,
//              SORTCTL=HSRTCTL3,
//              OUT=A
//*
//SORT     EXEC PGM=IERRCO00,
//              REGION=60K,
//              TIME=(1,0)
//*
//* SORT INCOMING TRANSACTIONS INTO FILE SEQUENCE
//*
//SORTLIB    DD DSNAME=SYS1.SORTLIB,
//              DISP=(SHR,PASS)
//*
//SYSOUT     DD SYSOUT=&OUT
//*
//SORTWK01   DD UNIT=&TEMP,
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK02   DD UNIT=(&TEMP,SEP=SORTWK01),
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTWK03   DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02)),
//              SPACE=(TRK,&SORTSPC,,CONTIG)
//*
//SORTIN     DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//SORTOUT    DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02,SORTWK03)),
//              DISP=(NEW,PASS,DELETE),
//              SPACE=(TRK,&SORTSPC,RLSE),
//              DSNAME=&&TRANS,
//              DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//*
//SYSIN      DD DSNAME=&PROJECT..SYSIN(&SORTCTL),
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//UPDATE   EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* MAINTAIN PARAMETER STANDARDS TABLE
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//     .        VOLUME=(PRIVATE,RETAIN),
```

AQSHUPD3

```
//             DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - TRANSACTIONS
//*
//AQSCARDS  DD DSNAME=&&TRANS,
//             DISP=(OLD,DELETE)
//*
//* INPUT DATA SET - OLD STANDARDS TABLE
//*
//AQSOLDTB  DD DSNAME=&PROJECT..DATA.&OLDMSTR,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - NEW STANDARDS TABLE
//*
//AQSSTDTB  DD UNIT=&UNIT,
//             VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//             DISP=(&DISP),
//             SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//             DSNAME=&PROJECT..DATA.&NEWMSTR
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSCPOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSILIST - SLIDING AVERAGE

AQSILIST is executed to produce a detailed listing of each record in the master or answer file and to calculate a sliding average of the readings in the record. There are six input files used by this program. The first, defined by DD name AQSINPUT, contains the control card required by the program. The second, defined by AQSMASTR, contains the answer or master file to be listed. The third, defined by AQSPARMK, contains the key portion of the parameter, method, unit codes table. The fourth, defined by AQSPARMD, contains the description portion of the parameter, method, unit codes table. The fifth, defined by AQSSITES, contains the valid site codes table. The sixth, defined by AQSTNDRD, contains the parameter standards table. There is one output file created by this program. This file, defined by AQSPRINT, contains the detailed report listing and any diagnostic messages generated during the reporting process.

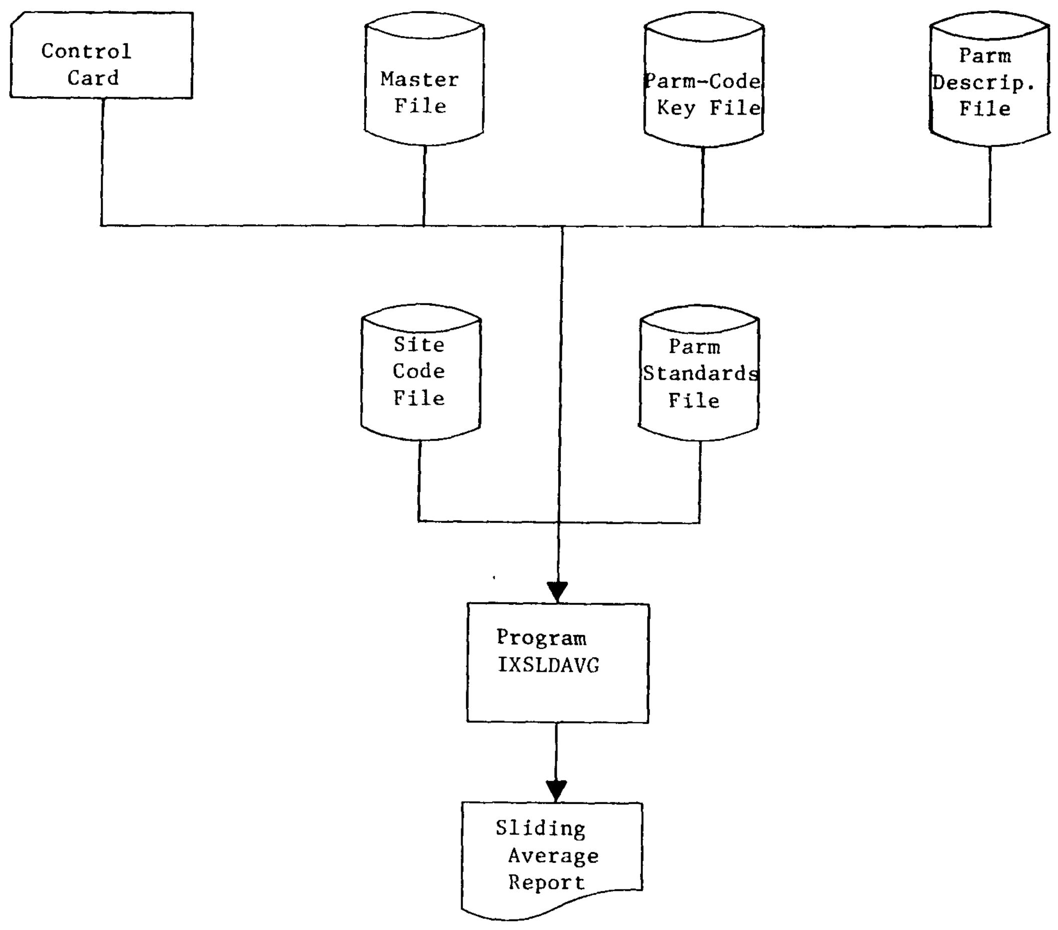
Figure 7.2.4 shows the format of the sliding average control card.

DD NAME	DESCRIPTION
AQSINPUT	AQDHS control card
AQSMASTR	AQDHS master file
AQSPARMK	Key portion of parameter, method, unit table
AQSPARMD	Description portion of table
AQSSITES	Valid site codes table
AQSTNDRD	Parameter standards table
AQSPRINT	Print file for report listing

AQSILIST - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	IXSLDAVG	This program lists the contents of the AQDHS master file and computes sliding averages
MSTRFIL	FTMSTRAA	Lowest level index of the AQDHS master file
PARMKFL	HTPRM1AA	Lowest level index of the key portion of parameter, method, unit codes table
PARMDFL	HTPRM2AA	Lowest level index of the description portion of table
SITEFIL	HTSITEAA	Lowest level index of the valid site codes table
STNDFIL	HTSTNDAA	Lowest level index of the parameter standards table
OUT	A	Sysout class for all print files

AQSILIST - Substitutable Parameters



AQSILIST - Data Flow

AQSILIST

```
//*             AQSILIST                 HURLEY RF          WF4   06/30/74
//*
//AQSILIST PROC PROJECT='CDHS.AQS',
//              PROGRAM=IXSLDAVG,
//              MSTRFIL=FTMSTRAA,
//              PARMKFL=HTPRM1AA,
//              PARMDFL=HTPRM2AA,
//              SITEFIL=HTSITEAA,
//              STNDFIL=HTSTNDAA,
//              OUT=A
//*
//REPORT   EXEC PGM=&PROGRAM,
//              REGION=100K,
//              TIME=(1,0)
//*
//* LIST CONTENTS OF AQDHS MASTER FILE WITH SLIDING AVERAGE OF READINGS
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.CORSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - CONTROL CARD
//*
//AQSINPUT   DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//* INPUT DATA SET - AQDHS MASTER FILE
//*
//AQSMASTR   DD DSNAME=&PROJECT..DATA.&MSTRFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - KEY PORTION OF PARAMETER, METHOD, UNIT TABLE
//*
//AQSPARMK   DD DSNAME=&PROJECT..DATA.&PARMKFL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - DESCRIPTION PORTION OF TABLE
//*
//AQSPARMD   DD DSNAME=&PROJECT..DATA.&PARMDFL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - VALID SITE CODE TABLE
//*
//AQSSITES   DD DSNAME=&PROJECT..DATA.&SITEFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - PARAMETER STANDARDS TABLE
//*
//AQSTNDRD   DD DSNAME=&PROJECT..DATA.&STNDFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
```

ACSILIST

```
//* OUTPUT DATA SET - REPORT LISTING
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSMSENT - ANSWER FILE FLAGGING

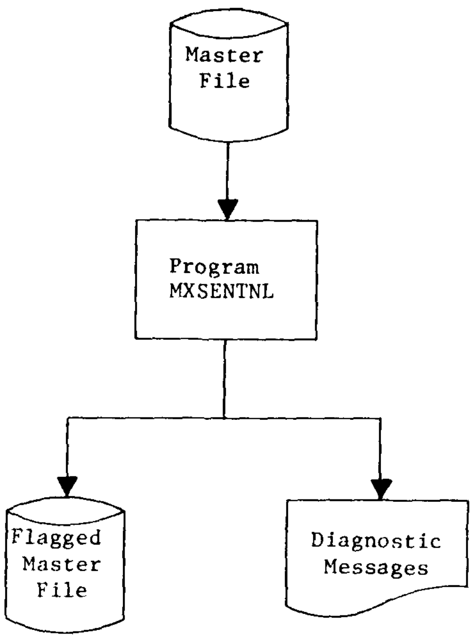
AQSMSENT is executed to append an end-of-file sentinel record to the AQDHS master or answer file before it is input to Data Analysis. The input to the program consists of the AQDHS master of answer file and is defined by DD name AQSMASTR. There are two output files created by this program. The first, defined by AQSENTNL, contains the flagged file. The second, defined by AQSPRINT, contains any diagnostic messages generated during the flagging process.

DD NAME	DESCRIPTION
AQSMASTR	AQDHS master file
AQSENTNL	Flagged AQDHS master file
AQSPRINT	Print file for diagnostic messages

AQSMSENT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	MXSENTNL	Program to copy master file and append end-of-file sentinel record
MSTRFIL	FTMSTRAA	Lowest level index of master file
STATFIL	MTMSTRAA	Lowest level index of flagged file
UNIT	2314	Unit type upon which the flagged file is to reside
SERIAL	009858	Volume serial number of volume upon which flagged file is to reside
DISP	'NEW,CATLG, DELETE'	Disposition of flagged file
SPCUNIT	TRK	Units in which space for the flagged file is to be allocated
PRIMARY	20	Number of units to be allocated for the flagged file's primary allocation
SECNDRY	10	Number of units to be allocated for the flagged file's secondary allocation
OUT	A	Sysout class for all print files

AQSMSENT - Substitutable Parameters



AQSMSENT - Data Flow

AQSMSENT

```
//*             AQSMSENT                HURLEY RF           WF4    06/30/74
//*
//AQSMSENT PROC PROJECT='COHS.AQS',
//              PROGRAM=MXSENTNL,
//              MSTRFIL=FTMSTRAA,
//              STATFIL=MTMSTRAA,
//              UNIT=2314,
//              SERIAL=CC9858,
//              DISP='NEW,PASS,DELETE',
//              SPCUNIT=TRK,
//              PRIMARY=20,
//              SECNDRY=10,
//              OUT=A
//*
//SENTINEL  EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* COPY MASTER FILE AND APPEND SENTINEL RECORD
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.CORSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - ACEHS MASTER FILE
//*
//AQSMASTR  DD DSNAME=&PROJECT..DATA.&MSTRFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - FLAGGED MASTER FILE
//*
//AQSENTNL  DD UNIT=&UNIT,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//              DISP=(&DISP),
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//              DSNAME=&PROJECT..DATA.&STATFIL
//*
//* OUTPUT DATA SET - MESSAGE LISTING
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSNSENT - PARM-CODE-KEY FILE FLAGGING

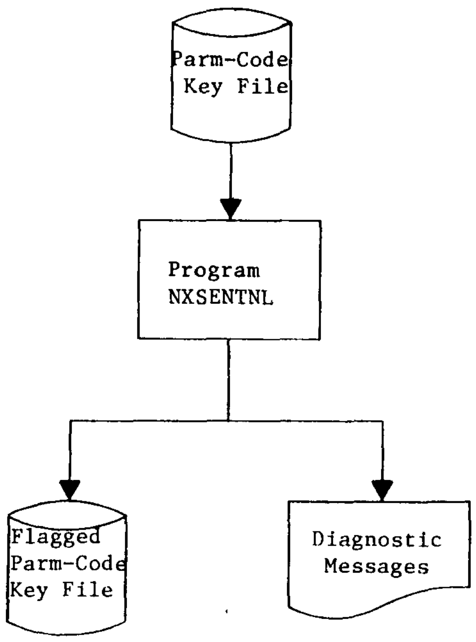
AQSNSENT is executed to append an end-of-file sentinel record to the key portion of the parameter, method, unit codes, minimum detectable table before it is input to Data Analysis. The input to the program consists of the key portion of the table and is defined by DD name AQSPARMK. There are two output files created by the program. The first, defined by AQSPARMF, contains the flagged table. The second, defined by AQSPRINT, contains any diagnostic messages generated during the flagging process.

DD NAME	DESCRIPTION
AQSPARMK	Key portion of parameter, method, unit codes table
AQSPARMF	Flagged key portion
AQSPRINT	Print file for diagnostic messages

AQSNSENT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	NXSENTNL	Program to copy parm-code-key file and append end-of-file sentinel record
PARMKFL	HTPRM1AA	Lowest level index of parm-code-key file
PARMFFL	NTPRM1AA	Lowest level index of flagged file
UNIT	2314	Unit type upon which flagged file is to be allocated
SERIAL	009858	Volume serial number of volume upon which flagged file is to be allocated
DISP	'NEW,CATLG, DELETE'	Disposition of flagged file
SPCUNIT	TRK	Units in which space for the flagged file is to be allocated
PRIMARY	10	Number of units to be allocated to the flagged file's primary allocation
SECNDRY	5	Number of units to be allocated to the flagged file's secondary allocation
OUT	A	Sysout class of all print files

AQSNSENT - Substitutable Parameters



AQSNSENT - Data Flow

AQSNSENT

```
//*              AQSNSENT               HURLEY RF          WF4   06/30/74
//*
//AQSNSENT PROC PROJECT='CDHS.AQS',
//              PROGRAM=NXSENTNL,
//              PARMKFL=HTPRM1AA,
//              PARMFFL=NTPRM1AA,
//              UNIT=2314,
//              SERIAL=CC9858,
//              DISP='NEW,PASS,DELETE',
//              SPCUNIT=TRK,
//              PRIMARY=10,
//              SECNDRY=5,
//              OUT=A
//*
//SENTINEL EXEC PGM=&PROGRAM,
//              REGION=60K,
//              TIME=(1,0)
//*
//* COPY PARAMETER KEY FILE AND APPEND SENTINEL RECORD
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - KEY PORTION OF PARAMETER, METHOD, UNIT TABLE
//*
//AQSPARMK  DD DSNAME=&PROJECT..DATA.&PARMKFL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - FLAGGED KEY FILE
//*
//AQSPARMF  DD UNIT=&UNIT,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//              DISP=(&DISP),
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//              DSNAME=&PROJECT..DATA.&PARMFFL
//*
//* OUTPUT DATA SET - MESSAGE LISTING
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSRETVR - RETRIEVAL

AQSRETVR is executed to generate an AQDHS Retrieval program, compile and link-edit it, and execute it to retrieve an answer file from the AQDHS master file.

There are two input files to the generation phase of this procedure. The first, defined by DD name AQSINPGM, contains the skeleton of the Retrieval program. The second, defined by AQSINPUT, contains the control cards specifying the qualification to be performed on the master file. Two output files are created by the generation phase. The first, defined by AQSRTRVR, contains the Retrieval program to be compiled and link-edited. The second, defined by AQSPRINT, contains any diagnostic messages generated during the program generation phase.

The actual retrieval phase uses one input file. The file is the AQDHS master file and is defined by AQSMASTR. There are two output files created in this phase. The first, defined by AQSANSWR, contains the records in the master file which met the qualification criteria. The second, defined by AQSPRINT, contains any diagnostic messages generated during retrieval.

Figures 6.1.4-1 and 6.1.4-2 show the formats of the retrieval control and specification cards.

DD NAME	DESCRIPTION
AQSINPGM	Retrieval program skeleton
AQSINPUT	Retrieval specifications
AQSRTRVR	Retrieval program
AQSPRINT	Print file for diagnostic messages
AQSMASTR	AQDHS master file
AQSANSWR	AQDHS answer file
AQSPRINT	Print file for diagnostic messages

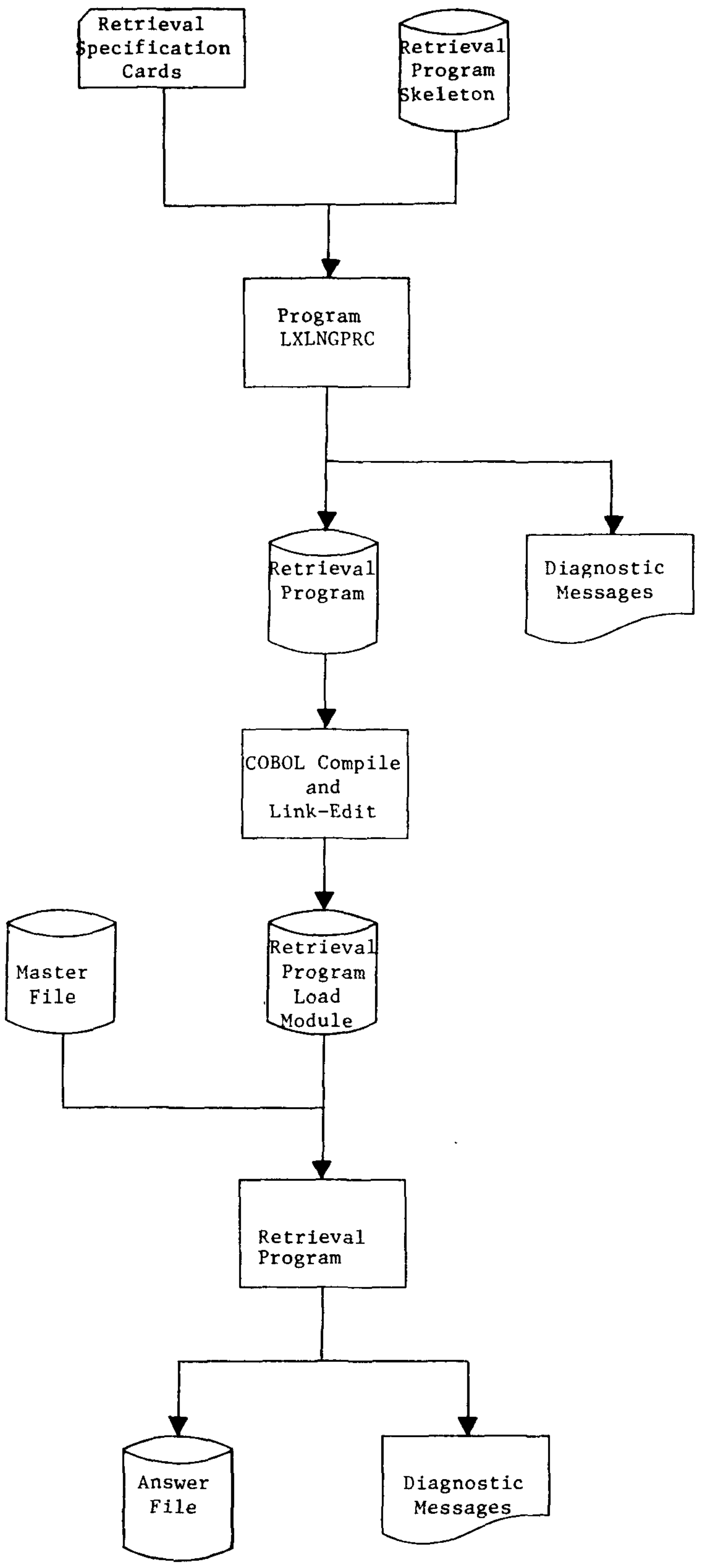
AQSRETVR - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	LXLNGPRC	Program to produce an AQDHS retrieval program
TEMP	SYSOUT	Unit type for temporary work space
MSTRFIL	FTMSTRAA	Lowest level index of master file
ANSWRFL	RTANSRAA	Lowest level index of answer file
UNIT	2314	Unit type upon which the answer file is to reside
SERIAL	009858	Volume serial number of volume upon which answer file is to reside
DISP	'NEW,CATLG, DELETE'	Disposition of answer file
SPCUNIT	TRK	Units in which space for answer file is to be allocated
PRIMARY	20	Number of units to be allocated for answer file's primary allocation
SECNDRY	10	Number of units to be allocated for answer file's secondary allocation

AQSRETVR - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
MEMBER	RTRETRVR	Retrieval program skeleton
OUT	A	Sysout class for all print files

AQSRETVR - Substitutable Parameters
(continued)


Retrieval Specification Cards
Retrieval Program Skeleton
Program LXLNGPRC
Retrieval Program
Diagnostic Messages
COBOL Compile and Link-Edit
Master File
Retrieval Program Load Module
Retrieval Program
Answer File
Diagnostic Messages

AQSRETVR - Data Flow

```
//*              AQSRETVR                   HURLEY RF          WF4  06/30/74
//*
//AQSRETVR PROC PROJECT='CDHS.AQS',
//             PROGRAM=LXLNGPRC,
//             TEMP=SYSOUT,
//             MSTRFIL=FTMSTRAA,
//             ANSWRFL=RTANSRAA,
//             UNIT=2314,
//             SERIAL=CC9858,
//             DISP='NEW,PASS,DELETE',
//             SPCUNIT=TRK,
//             PRIMARY=20,
//             SECNDRY=10,
//             MEMBER=RTRETRVR,
//             OUT=A
//*
//COMPILE EXEC PGM=&PROGRAM,
//             REGION=60K,
//             TIME=(1,0)
//*
//* PRODUCE AQDHS RETRIEVAL PROGRAM
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//          DD DSNAME=SYS1.ANS.COBSUBR,
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - RETRIEVAL PROGRAM SKELETON
//*
//AQSINPGM  DD DSNAME=&PROJECT..TEST(&MEMBER),
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - RETRIEVAL SPECIFICATION CARDS
//*
//AQSINPUT  DD DDNAME=INPUT,
//             DCB=BLKSIZE=80
//*
//* OUTPUT DATA SET - RETRIEVAL PROGRAM
//*
//AQSRTRVR  DD UNIT=&TEMP,
//             DISP=(NEW,PASS),
//             SPACE=(TRK,(5,2),RLSE),
//             DSNAME=&&PROGRAM
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
```

ACSRETVR

```
//SYSDTERM  CC SYSCUT=&CUT
//*
//SYSUDUMP  CD SYSCUT=&CUT
//*
//COBOL    EXEC PGM=IKFCBLOO,
//              PARM='SIZE=94K,BUF=10K,NOSEQ,NOSOURCE,SUPMAP',
//              REGICN=100K,
//              TIME=(2,0)
//*
//* CCMPILE RETRIEVAL PRCGRAM
//*
//STEPLIB   CD DSNAME=SYS1.ANS.COBLIB,
//              DISP=(SHR,PASS)
//*
//* INPUT CATA SET - SCURCE LIBRARY
//*
//SYSLIB    CD DSNAME=&PRCJECT..SCURCE,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - CCBCL RETRIEVAL PROGRAM SCURCE CODE
//*
//SYSIN     CD CSNAME=&&PRCGRAM,
//              DISP=(CLC,DELETE)
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//SYSPRINT  DD SYSOUT=&CUT
//*
//* OUTPUT CATA SET - CCBOL RETRIEVAL PRCGRAM OBJECT CODE
//*
//SYSLIN    CC UNIT=&TEMP,
//              DISP=(NEW,PASS),
//              SPACE=(TRK,(5,2),RLSE),
//              DSNAME=&&OBJMOD
//*
//* UTILITY DATA SETS
//*
//SYSUT1    CD UNIT=&TEMP,
//              SPACE=(TRK,(50,100))
//*
//SYSUT2    DD UNIT=(&TEMF,SEP=SYSUT1),
//              SPACE=(TRK,(50,100))
//*
//SYSUT3    DC UNIT=(&TEMF,SEP=(SYSUT1,SYSUT2)),
//              SPACE=(TRK,(50,100))
//*
//SYSUT4    DD UNIT=(&TEMF,SEP=(SYSUT1,SYSUT2,SYSUT3)),
//              SPACE=(TRK,(5C,100))
//*
//LKED     EXEC PGM=IEWL,
//              PARM='LIST,LET,XREF',
//              CONC=(5,LT,COBOL),
//              REGICN=100K,
//              TIME=(1,0)
//*
//* LINK-EDIT RETRIEVAL
//*
```

AQSRETVR

```
//*
//* INPUT DATA SET - AUTO-CALL LIBRARY
//SYSLIB     DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - COBOL RETRIEVAL PROGRAM OBJECT CODE
//*
//SYSLIN     DD DSNAME=&&OBJMOD,
//              DISP=(OLD,DELETE)
//           DD DDNAME=INPUT,
//              DCB=BLKSIZE=80
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//SYSPRINT   DD SYSOUT=&OUT
//*
//* OUTPUT DATA SET - COBOL RETRIEVAL PROGRAM LOAD MODULE
//*
//SYSLMOD    DD UNIT=&TEMP,
//              DISP=(MOD,PASS),
//              SPACE=(TRK,(10,5,1)),
//              DSNAME=&&LOADMOD(RETRIEVE)
//*
//* UTILITY DATA SET
//*
//SYSUT1     DD UNIT=&TEMP,
//              SPACE=(TRK,(10,5))
//*
//RETRIEVE EXEC PGM=*.LKED.SYSLMOD,
//              COND=((5,LT,COBOL),(5,LT,LKED)),
//              REGION=60K,
//              TIME=(1,0)
//*
//* RETRIEVE ANSWER FILE FROM ACDHS MASTER FILE
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//              DISP=(SHR,PASS)
//*
//* INPUT DATA SET - ACDHS MASTER FILE
//*
//AQSMASTR   DD DSNAME=&PROJECT..DATA.&MSTRFIL,
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - ANSWER FILE
//*
//AQSANSWR   DD UNIT=&UNIT,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//              DISP=(&DISP),
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//              DSNAME=&PROJECT..DATA.&ANSWRFL
//*
```

ACSRETVR

```
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//ACSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSCBOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
//DELETE  EXEC PGM=IEFBR14,
//             REGION=4K,
//             TIME=(0,5)
//LOADMOD   DD DSNAME=&&LOADMOD,
//             DISP=(OLD,DELETE)
//*
```

AQSSLIST - STATISTICAL LIST

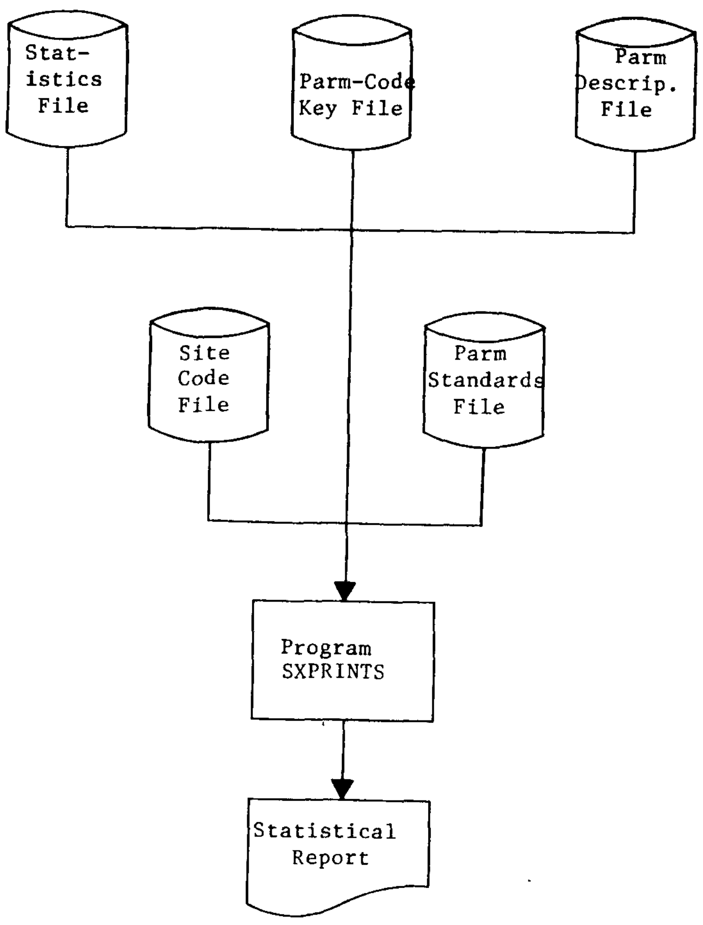
AQSSLIST is executed to format the statistics file created by Data Analysis. There are five input files. The first, defined by DD name AQSTATIS, contains the statistics file to be formatted. The second, defined by AQSPARMK, contains the key portion of the parameter, method, unit codes table. The third, defined by AQSPARMD, contains the description portion of the parameter, method, unit codes table. The fourth, defined by AQSSITES, contains the valid site codes table. The fifth, defined by AQSTNDRD, contains the parameter standards table. There is one output file created by this program. The file, defined by AQSPRINT, contains the formatted listing of the statistics file and any diagnostic messages generated during the formatting process.

DD NAME	DESCRIPTION
AQSTATIS	Defines statistics file to be formatted
AQSPARMK	Defines key portion of parameter, method, unit, minimum detectable table file
AQSPARMD	Defines description portion of above file
AQSSITES	Defines valid site codes table file
AQSTNDRD	Defines parameter standards file
AQSPRINT	Statistical report and diagnostic messages

AQSSLIST - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	SXPRINTS	Program to format and print statistical report
PARMKFL	HTPRM1AA	Lowest level index of key portion of parameter method, unit, minimum detectable table file
PARMDFL	HTPRM2AA	Lowest level index of description portion of above file
SITEFIL	HTSITEAA	Lowest level index of valid site codes table file
STNDFIL	HTSTNDAA	Lowest level index of parameter standards table file
STATFIL	STMSTRAA	Lowest level index of statistic file
OUT	A	Sysout class of all print files

AQSSLIST - Substitutable Parameters



AQSSLIST - Data Flow

AQSSLIST

```
//*              AQSSLIST                     STROUP EB         WF4  06/30/74
//*
//AQSSLIST PROC PROJECT='CDHS.AQS',
//             PROGRAM=SXPRINTS,
//             PARMKFL=HTPRM1AA,
//             PARMDFL=HTPRM2AA,
//             SITEFIL=HTSITEAA,
//             STNDFIL=HTSTNDAA,
//             STATFIL=STMSTRAA,
//             OUT=A
//*
//PRINTS EXEC PGM=&PROGRAM,
//             REGION=60K,
//             TIME=(1,0)
//*
//* LIST DATA ANALYSIS STATISTICS FILE
//*
//STEPLIB   DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)'
//          DD DSNAME=SYS1.ANS.COBSUBR,
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - AQDHS STATISTICS FILE
//*
//AQSTATIS  DD DSNAME=&PROJECT..DATA.&STATFIL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - KEY PORTION OF PARAMETER, METHOD, UNIT TABLE
//*
//AQSPARMK  DD DSNAME=&PROJECT..DATA.&PARMKFL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - DESCRIPTION PORTION OF TABLE
//*
//AQSPARMD  DD DSNAME=&PROJECT..DATA.&PARMDFL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - VALID SITE CODE TABLE
//*
//AQSSITES  DD DSNAME=&PROJECT..DATA.&SITEFIL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - PARAMETER STANDARDS TABLE
//*
//AQSTNDRD  DD DSNAME=&PROJECT..DATA.&STNDFIL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - REPORT LISTING
//*
//AQSPRINT  DD SYSOUT=&OUT
//*
//* OUTPUT DATA SETS - SYSTEM OPERATION
```

AQSSLIST

```
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSCPOUT  DD SYSOUT=&OUT
//*
//SYSDTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
```

AQSTEDIT - TRANSACTION EDITOR

AQSTEDIT is executed to edit AQDHS file maintenance transactions. There are three input files used by this program. The first, defined by DD name AQSTRANS, contains the AQDHS transactions. The second, defined by AQSPARMS, consists of the key portion of the valid parameter, method, unit code table. The third, defined by AQSSITES, contains the valid site codes table. There are two output files created by this program. The first, defined by AQSINTRN, consists of the AQDHS internal transactions. The second, defined by AQSPRINT, contains a listing of diagnostic messages. Afterwards, the AQDHS internal transactions are sorted into file sequence.

Figure 2.2-2.1 shows the formats of the file maintenance transactions.

DD NAME	DESCRIPTION
AQSTRANS	Defines input data set containing transactions to be edited
AQSPARMS	Defines key portion of parameter, method, unit, minimum detectable table file.
AQSSITES	Defines valid site code table file
AQSINTRN	Defines output data set to contain edited transactions
AQSPRINT	Diagnostic messages

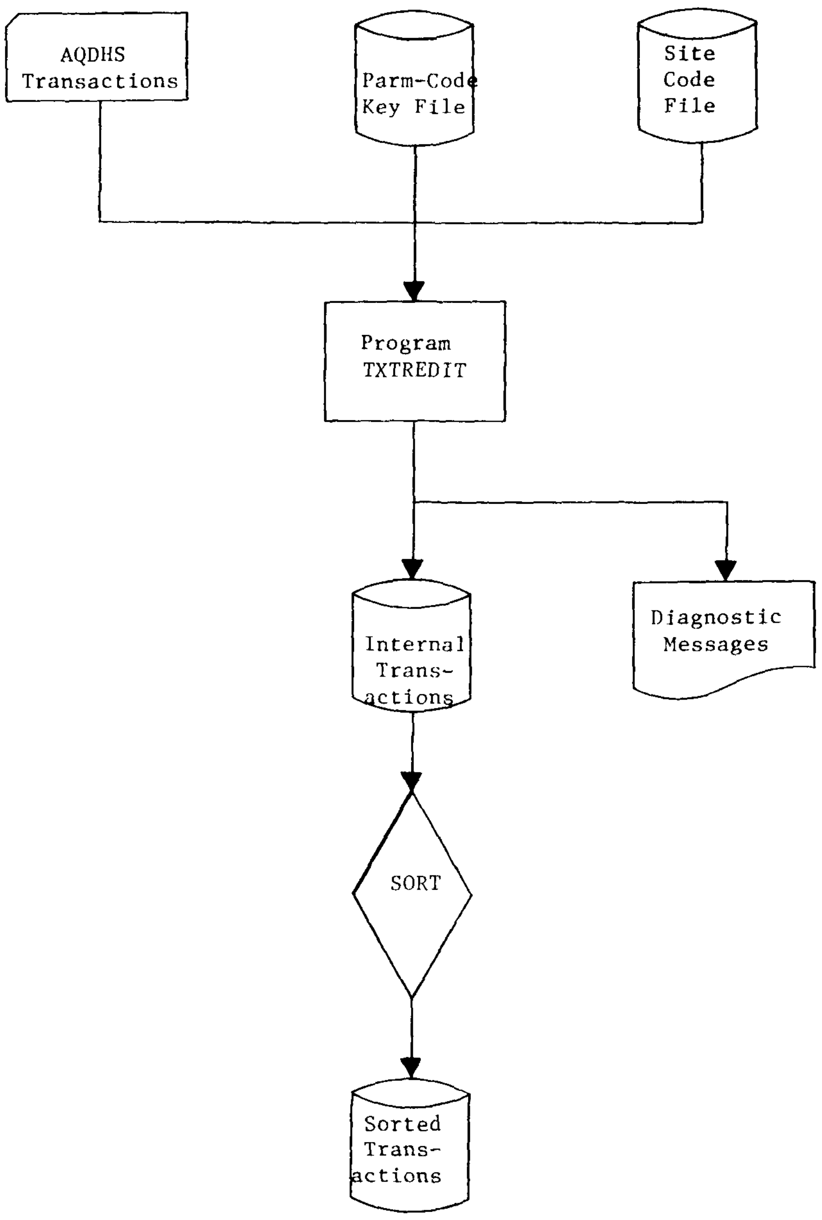
AQSTEDIT - DDNAMES

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
PROJECT	'CDHS.AQS'	Highest level index of data set names (e.g., CDHS.AQS.DATA.FTMSTRAA)
PROGRAM	TXTREDIT	Program to edit AQDHS file maintenance transactions
UNIT	2314	Unit type to which edited transactions are to be written
SERIAL	009858	Volume id to which edited transactions are to be written
DISP	'NEW,PASS, DELETE'	Disposition of edited transactions
SPCUNIT	TRK	Units in which space for the edited transactions is to be allocated
PRIMARY	10	Primary space allocation
SECNDRY	5	Secondary space allocation
TRANS	TRANS	Lowest level index of data set to contain edited transactions
PARMKFL	HTPRM1AA	Lowest level index of parameter code key file
SITEFIL	HTSITEAA	Lowest level index of valid site code file

AQSTEDIT - Substitutable Parameters

PARAMETER NAME	DEFAULT VALUE	DESCRIPTION
TEMP	SYSOUT	Unit type for temporary allocations
WORKSPC	20	Space allocation, in tracks, for work space
SORTCTL	TSORTCTL	Member of SYSIN section containing sort control card
OUT	A	Sysout class for all print files

AQSTEDIT - Substitutable Parameters
(continued)



AQSTEDIT - Data Flow

```
//*              AQSTEDIT               HURLEY RF          WF4   06/30/74
//*
//AQSTEDIT PROC PROJECT='CCHS.AQS',
//             PROGRAM=TXTREDIT,
//             UNIT=2314,
//             SERIAL=CC9858,
//             DISP='NEW,PASS,DELETE',
//             SPCUNIT=TRK,
//             PRIMARY=10,
//             SECNDRY=5,
//             TRANS=TRANS,
//             PARMKFL=HTPRM1AA,
//             SITEFIL=HTSITEAA,
//             TEMP=SYSOUT,
//             WORKSPC=20,
//             SORTCTL=TSCRTCTL,
//             OUT=A
//*
//EDIT     EXEC PGM=&PROGRAM,
//             REGION=60K,
//             TIME=(1,0)
//*
//* EDIT AQDHS FILE MAINTENANCE TRANSACTIONS
//*
//STEPLIB    DD DSNAME=&PROJECT..LOAD,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//           DD DSNAME=SYS1.ANS.COBSUBR,
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - AQDHS TRANSACTIONS
//*
//AQSTRANS   DD DDNAME=INPUT,
//             DCB=BLKSIZE=80
//*
//* INPUT DATA SET - KEY PORTION OF VALID PARM, METH, UNIT TABLE
//*
//AQSPARMS   DD DSNAME=&PROJECT..DATA.&PARMKFL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* INPUT DATA SET - VALID SITE CODE TABLE
//*
//AQSSITES   DD DSNAME=&PROJECT..DATA.&SITEFIL,
//             VOLUME=(PRIVATE,RETAIN),
//             DISP=(SHR,PASS)
//*
//* OUTPUT DATA SET - AQDHS INTERNAL TRANSACTIONS
//*
//AQSINTRN   DD UNIT=&TEMP,
//             DISP=(NEW,PASS,DELETE),
//             SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//             DSNAME=&&TRANS
//*
//* OUTPUT DATA SET - DIAGNOSTIC MESSAGES
//*
//AQSPRINT   DD SYSOUT=&OUT
//*
```

ACSTECIT

```
//* OUTPUT DATA SETS - SYSTEM OPERATION
//*
//SYSPRINT  DD SYSOUT=&OUT
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SYSDBOUT  DD SYSOUT=&OUT
//*
//SYSCTERM  DD SYSOUT=&OUT
//*
//SYSUDUMP  DD SYSOUT=&OUT
//*
//SORT     EXEC PGM=IERRCO00,
//              REGION=60K,
//              TIME=(1,0)
//*
//* SORT INTERNAL TRANSACTIONS INTO FILE SEQUENCE
//*
//SORTLIB   DD DSNAME=SYS1.SORTLIB,
//              DISP=(SHR,PASS)'
//*
//SYSOUT    DD SYSOUT=&OUT
//*
//SORTWK01  DD UNIT=&TEMP,
//              SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTWK02  DD UNIT=(&TEMP,SEP=SORTWK01),
//              SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTWK03  DD UNIT=(&TEMP,SEP=(SORTWK01,SORTWK02)),
//              SPACE=(TRK,(&WORKSPC),,CONTIG)
//*
//SORTIN    DD DSNAME=&&TRANS,
//              DISP=(OLD,DELETE)
//*
//SORTOUT   DD UNIT=&UNIT,
//              VOLUME=(PRIVATE,RETAIN,SER=&SERIAL),
//              DISP=(&DISP),
//              SPACE=(&SPCUNIT,(&PRIMARY,&SECNDRY),RLSE),
//              DSNAME=&PROJECT..DATA.&TRANS,
//              DCB=(RECFM=FB,LRECL=68,BLKSIZE=1632)
//*
//SYSIN     DD DSNAME=&PROJECT..SYSIN(&SORTCTL),
//              VOLUME=(PRIVATE,RETAIN),
//              DISP=(SHR,PASS)
//*
```

Appendix E - AQDHS LOAD SHEETS

AQDHS LOAD SHEETS

The following load sheets are suggested for use with AQDHS. Data encoded upon them may be easily keypunched and entered into the system. A separate load sheet is provided for each different type of transaction (form #1, form #2, and form #3) as well as for composite data.

AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS)
COMPREHENSIVE DATAHANDLING
SYSTEM (CDHS)

FORMAT 1
INPUT FORM
(Corresponds to
SAROAD Form 1)

Name of Person
Completing Form ______________________

Date ______________________

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
1													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Reading				Reading				Reading				Reading				Reading				Reading				Reading																		Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
1													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Reading				Reading				Reading				Reading				Reading				Reading				Reading																		Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	29	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
1													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Reading				Reading				Reading				Reading				Reading				Reading				Reading																		Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS)
COMPREHENSIVE DATAHANDLING
SYSTEM (CDHS)

Format 2
Input Form
(Corresponds to
SAROADForm2)

Name of Person
Completing Form ____________________

Date ____________________

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Month		Day		Start Hour		Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS)
COMPREHENSIVE DATA HANDLING
SYSTEM (CDHS)

Format 2
Composite Input Form
(Corresponds to
SAROAD Composite Form 2)

Name of Person
Completing Form ______________________

Date ______________________

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Period		Number of Samples		Composite Type	Sample Time	Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Period		Number of Samples		Composite Type	Sample Time	Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Area				Site			Agency	Project		Time
1	2	3	4	5	6	7	8	9	10	11	12	13	14
2													

Year		Period		Number of Samples		Composite Type	Sample Time	Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Parameter Code					Method		Units		Decimal Position	Reading				Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

AIR QUALITY DATA HANDLING SUBSYSTEM (AQDHS)
COMPREHENSIVE DATA HANDLING
SYSTEM (CDHS)

Format 3
Input Form
(Corresponds to
SAROAD Form 3)

Name of Person
Completing Form ______________________

Date ______________________

Form	State		Agency	Project		Time	Parameter Code					Method	
1	2	3	4	5	6	7	8	9	10	11	12	13	14
3													

Units		Decimal Position	Year		Month		Day		Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading							Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Agency	Project		Time	Parameter Code					Method	
1	2	3	4	5	6	7	8	9	10	11	12	13	14
3													

Units		Decimal Position	Year		Month		Day		Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading							Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

Form	State		Agency	Project		Time	Parameter Code					Method	
1	2	3	4	5	6	7	8	9	10	11	12	13	14

Units		Decimal Position	Year		Month		Day		Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading				Start Hour		Area				Site			Reading							Status	Action
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80

TECHNICAL REPORT DATA
(Please read Instructions on the reverse before completing)

1. REPORT NO. EPA - 450/3-74-045	2.	3. RECIPIENT'S ACCESSION NO.
4. TITLE AND SUBTITLE Comprehensive Data Handling System, Air Quality Data Handling Subsystem (AQDHS-II) Program Documentation and Users Guide		5. REPORT DATE July 1974
		6. PERFORMING ORGANIZATION CODE
7. AUTHOR(S) The IBM Corporation Federal Systems Division Gaithersburg, Maryland 20760		8. PERFORMING ORGANIZATION REPORT NO.
9. PERFORMING ORGANIZATION NAME AND ADDRESS Office of Air Quality Planning and Standards Control Programs Development Division Research Triangle Park, N.C. 27711		10. PROGRAM ELEMENT NO. 2AH-137
		11. CONTRACT/GRANT NO. 68-02-0638
12. SPONSORING AGENCY NAME AND ADDRESS U.S. Environmental Protection Agency Office of Air and Waste Management Office of Air Quality Planning and Standards Research Triangle Park, North Carolina 27711		13. TYPE OF REPORT AND PERIOD COVERED Final
		14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

16. ABSTRACT

When dealing with atmospheric pollution, it is necessary to amass, catalog, sort, evaluate, and perform calculations upon large volumes of data. The Air Quality Data Handling System (AQDHS-II) provides air pollution control agencies with the capability to create and maintain their own air quality data base and to retrieve data and generate reports from that data base. This report contains documentation for the computer programs which comprise AQDHS-II. It is also written as a Users Guide with each program described, input requirements described, field descriptions, etc. In addition to the basic system, several preprocessor and postprocessor programs are provided which perform functions necessary to make this system compatible with existing systems such as SAROAD and the original AQDHS. The system has a powerful retrieval capability which allows the user to retrieve virtually any piece of data in his file. They system also allows the user to automatically generate his quarterly air quality progress report in SAROAD format.

17. KEY WORDS AND DOCUMENT ANALYSIS

a. DESCRIPTORS	b. IDENTIFIERS/OPEN ENDED TERMS	c. COSATI Field/Group
Computer Programs Computer Software Data Processing Air Pollution Data Handling Computer Systems Programs	CDHS AQDHS-II Atmospheric Pollution SAROAD	

18. DISTRIBUTION STATEMENT Unlimited	19. SECURITY CLASS *(This Report)* Unclassified	21. NO. OF PAGES 269
	20. SECURITY CLASS *(This page)* Unclassified	22. PRICE

EPA Form 2220-1 (9-73)

ENVIRONMENTAL PROTECTION AGENCY
Technical Publications Branch
Office of Administration
Research Triangle Park, N.C. 27711

OFFICIAL BUSINESS

AN EQUAL OPPORTUNITY EMPLOYER

POSTAGE AND FEES PAID
ENVIRONMENTAL PROTECTION AGENCY
EPA - 335

SPECIAL FOURTH-CLASS RATE
BOOK

U.S.MAIL

Return this sheet if you do NOT wish to receive this material ☐, or if change of address is needed ☐. (Indicate change, including ZIP code.)

PUBLICATION NO. EPA-450/3-74-045