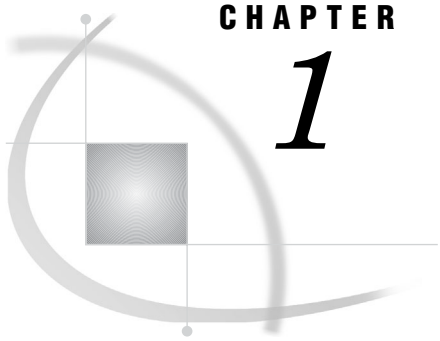

CHAPTER

1

Moving and Accessing SAS Files between Operating Environments

Deciding to Move a SAS File between Operating Environments

Moving SAS files between operating environments is a common work task. Reasons for moving a SAS file between operating environments include:

- To move SAS files to a new operating environment on a different machine; for example, HP-UX files to a RedHat Linux operating environment.
- To move a file and its processing to a high-performance operating environment that will be returned to the requesting operating environment.
- To make a static copy of a SAS file available to a physically separate operating environment for continued data processing. Files are duplicated for use in the receiving operating environment because the SAS files are not available to the receiving operating environment by means of NFS-mounted file systems.

In all of these scenarios, the move operations recognize differences between machine architectures and SAS releases, allowing the original files to be used in the receiving operating environment.

Deciding to Access a SAS File across Operating Environments

In some instances, accessing instead of owning and maintaining your own copy of a file might be preferable. Alternatively, you might need to read data from a locally mounted tape that was created elsewhere, or you might need to read, write, or update data that is remotely mounted on your network.

Note: Do not confuse the term *access* with the product SAS/ACCESS. In the context of moving or accessing SAS files across operating environments, *access* means to reach and process SAS files. SAS/ACCESS enables users to use third-party DBMS files. For a list of products that SAS/ACCESS supports, see the list on page 6. △

You can use the following methods to access remote SAS files:

- CEDA (Cross-Environment Data Access) enables you to process SAS 8 and later SAS files.
- use SAS/SHARE on your client to access a remote SAS file that resides on an operating environment that a SAS/SHARE server runs under. SAS/SHARE facilitates a transparent concurrent access to remote data among multiple users. Restrictions apply to cross-release access of SAS data.

  In addition, SAS/SHARE enables you to access certain third-party DBMS files by means of engines that are supported by SAS/ACCESS.
- without the aid of SAS/SHARE or CEDA, you can rely upon network services for access to remote files (both SAS files and third-party DBMS files). Usually, the client and the server must share a compatible architecture, and they must run the same release of SAS software. The operating environment, the network software, and the security software might control users' permissions to access specific remote files. For more information, see the SAS companion documentation that is appropriate to your operating environment, and see the third-party documentation for the network software and security software that you use.

Strategies for Moving and Accessing SAS Files

Cross-Environment Data Access (CEDA)
: This feature of SAS enables a SAS file that was created in any directory-based operating environment (for example, Solaris, Windows, HP-UX, OpenVMS) to be processed by a SAS session that is running in another directory-based environment.

CPORT and CIMPORT procedures
: In the source environment, you can use PROC CPORT to write data sets or catalogs to transport format. In the target environment, PROC CIMPORT can be used to translate the transport file into the target environment's native format.

XPORT engine with DATA step or PROC COPY
: In the source environment, you can use the LIBNAME statement with the XPORT engine and either the DATA step or PROC COPY to create a transport file from a SAS data set. In the target environment, the same method can be used to translate the transport file into the target environment's native format.

  Note: The XPORT engine does not support SAS 8 and later features, such as long file and variable names. △

XML engine with DATA step or PROC COPY
: In the source environment, you can use the LIBNAME statement with the XML engine and either the DATA step or PROC COPY to create an XML document from a SAS data set. In the target environment, the same method can be used to translate the XML document into the target environment's native format.

Data Transfer Services (DTS) in SAS/CONNECT
: This feature enables you to transfer data sets and catalogs from the source environment to the target environment. DTS dynamically translates the data between operating environment representations and SAS versions, as necessary. The transfer is accomplished using the SIGNON statement to connect two SAS sessions and then the PROC UPLOAD or PROC DOWNLOAD to move the data.

REMOTE engine and Remote Library Services in SAS/SHARE and SAS/CONNECT
These features give you transparent access to remote data using the REMOTE engine and the LIBNAME statement.

Summary of Strategy Features

Table 1.1 Summary of Strategy Features for Moving or Accessing SAS Files

Features	**Strategies That Can Be Used**					
	CEDA	PROC CPORT/ PROC CIMPORT	XPORT Engine	XML Engine	SAS/CONNECT DTS	SAS/CONNECT RLS and SAS/SHARE RLS
SAS Member Types Supported	Data File, PROC SQL views*, SAS/ACCESS views (Oracle and SYBASE), MDDB*	Library, Data Set, Catalog, Catalog entry	Library, Data Set	Data File	Library, Data Set, Catalog, Catalog entry, PROC SQL view, MDDB, External third-party databases***	Library, Data Set, Catalog**, Catalog entry**, PROC SQL view, MDDB, DATA Step view, SAS/ACCESS view, External third-party databases***

* Data set (files) can have read, write, and update access. PROC SQL views and MDDBs are read-only.

** SAS 9 does not support cross-operating environment access to catalog entries or catalogs in operating environments that are incompatible. For information about architecture groups, see *SAS/CONNECT User's Guide* or *SAS/SHARE User's Guide*.

***SAS/CONNECT supports external text files and binary files. SAS/CONNECT and SAS/SHARE support third-party external databases by means of the Remote SQL Pass-Through Facility, but you must have a SAS/ACCESS license to access these databases. Here is a list of external files that SAS/CONNECT and SAS/SHARE support:

- Relational databases
  - CA-OpenIngres, DB2 for OS/390, DB2 for UNIX and PC operating environments, Informix, ODBC, Oracle, Oracle Rdb, and SYBASE
- Nonrelational databases
  - ADABAS, CA-IDMS, IMS-DL/I, and SYSTEM 2000
- PC files
  - PC file formats Excel and Lotus

Features	Strategies That Can Be Used					
	CEDA	PROC CPORT/ PROC CIMPORT	XPORT Engine	XML Engine	SAS/CONNECT DTS	SAS/CONNECT RLS and SAS/SHARE RLS
Dynamic Translation or Create a File Format	Dynamic	Transport****	Transport****	XML	Dynamic	Dynamic
SAS Versions Supported	SAS 8 and later	SAS 6 and later	SAS 6 and later****	SAS 8.2 and later	SAS 6 and later	SAS 6 and later
Regression from a Later to an Earlier SAS Release	No	No	Yes	No	Yes	Yes
Limited to Operating Environments that Use Directory-Based File Structures	Yes	No	No	No	No	No
SAS Product License Required	Base SAS	Base SAS	Base SAS	Base SAS	SAS/CONNECT	SAS/CONNECT or SAS/SHARE

****The XPORT engine does not support features that were introduced in SAS 8 (such as long file and variable names). If the XPORT engine is used to regress a SAS 8 or later SAS file to an earlier release, the features that are exclusive to SAS 8 and later are removed from the SAS file. Also, the transport formats that are produced by the XPORT engine and PROC CPORT are *not* interchangeable.

For complete details about relational databases, see *SAS/ACCESS for Relational Databases: Reference*. For details about nonrelational databases, see *SAS/ACCESS Interface to CA-Datacom/DB: Reference*, *SAS/ACCESS Interface to IMS: Reference*, *SAS/ACCESS DATA Step Interface to CA-IDMS: Reference*, or *SAS/ACCESS Interface to SYSTEM 2000: Reference*, as appropriate.

Moving and Accessing SAS Files in International Environments

SAS provides National Language Support (NLS) for SAS applications and data that are created in supported operating environments. Customers who use the English language can use SAS applications and data that are created in the United States. However, without NLS, customers in other geographic regions of the world such as Asia and Europe would not be able to run SAS applications and read and write data that was created in the United States. NLS features enable customers to process data successfully in their native languages and environments, regardless of the language that the application and data were created in.

As an example, a source SAS session runs a SAS application and creates a data set, which is written in the English language, on a SAS 8 PC. A target SAS session runs a different SAS application, which is written in the German language, on a SAS 6 mainframe that needs to read from and write to the SAS data set that was created in the English language.

Before the data can be moved or accessed using the preferred strategy, (for example, CEDA or PROC CPORT and PROC CIMPORT), locale or encoding must be specified at the source session and target session to enable the source data to be translated to the format of the target session. If encodings are not accounted for in an international environment, source and target sessions cannot read and write the data. Strategies for specifying locale or encoding vary according to the version of SAS that is running on the source and target machines.

If you are moving or accessing SAS files in an international environment, see *SAS National Language Support (NLS): User's Guide*.

The Data Set Used for Examples

If you choose to experiment, you can create several simple data sets in a library. Here is a sample SAS program that creates the data set GRADES:

```
data grades;
  input student $ test1 test2 final;
  datalines;
Fred 66 80 70
Wilma 97 91 98
;
proc print data=grades;
run;
```

Here is the output:

```
     The SAS System      10:59 Friday, April 25, 2003

 Obs    student    test1    test2    final
 1      Fred        66       80       70
 2      Wilma       97       91       98
```

Naming Conventions Used for Examples

The following consistent naming conventions are used in the examples in this documentation:

WORK
is the default libref that points to the library that contains the data set GRADES.

XPORTOUT
is the libref that points to the location where the transport file is created with the XPORT engine.

XPORTIN
is the libref that points to the location on the target machine that you transferred the transport file to.

XMLOUT
is the libref that points to the location where the XML file is created with the XML engine.

XMLIN
is the libref that points to the location on the target machine that you transferred the XML file to.

CPORTOUT
is the fileref that points to the location where the transport file is created with PROC CPORT.

IMPORTIN
is the fileref that points to the location on the target machine that you transferred the transport file to.

SOURCE
is the libref that points to the location of the source file that is translated into transport or XML format.

LIST
is a catalog entry type.

GRADES
is the name of a data set.

TARGET
is the libref that points to the location where the restored SAS file is created.

TESTCAT
is the name of a catalog.

TESTNPGM
is the name of a catalog entry.