

Introducing FOCUS Fusion ...  
the high performance database

**New Features**

# Contents

## 7.0.9 New Features

**NF575: Fusion**

**NF716: Euro Currency Support**

**Converting Currencies**

**Preparing FOCUS to Process Currency Conversions**

**Creating the Currency Database**

**Identifying Fields That Contain Currency Data**

**Activating the Currency Database**

**Processing Currency Data**

**NF744: HOLD FORMAT EXCEL**

**NF730: Hold Format PDF**

**Required Software Configuration**

**Downloading PDF Output**

**NF654: HOLD From External Sort**

**Conditions for Using External Sort to Create a HOLD File**

**NF597: Aggregation by External Sort**

**Conditions for Aggregating with an External Sort**

**NF728: Changing Retrieval Order with Aggregation**

**NF655: FOCPROF - The System Wide Profile**

**FOCUS Profiles**

**Using FOCUS Profiles**

### **NF660: Multi-volume Support in MVS FOCUS**

**Advantages of Multi-volume Data Sources**

**Allocating Multi-volume Data Sources**

**Choosing Default Sizes for FOCUS-created Files**

### **NF584: Dynamically Setting the IDMS DBNAME and DICTNAME**

### **NF673: Model 204 Interface Account Split**

### **NF720: SQLJOIN OUTER Setting for Relational Interfaces**

### **NF652: Teradata Interface Kanji Support**

### **NF722: FOCUS Client DNS Names Support**

### **NF656: Controlling REBUILD Messages**

### **NF670: DYNAM Support for Unit Count**

**Advantages of Multi-volume Data Sources**

### **NF684: PCHOLD for Non-Html Files**

**Using PCHOLD for Formats LOTUS, DIF, EXCEL, or PDF**

### **NF683: Web Interface support for Maintain Winforms**

**Prerequisites**

### **NF691: Escape Character for LIKE Predicate**

**Escape Character Capabilities**

### **NF718: DYNAM Support for Existing Relative GDG Numbers**

**Using DYNAM With Relative GDG Numbers**

### **NF735: Enhancement to ? SET**

**NF740: Changes to the REBUILD Prompt**

**NF745: ? PTF Enhancements**

**NF746: Leading Zeros**

**NF748: HOLD FORMAT WP With Carriage Control**

## **Year 2000 New Features**

### **7.0.8R New Features**

**NF557: REBUILD - Legacy Date Conversion**

**How the REBUILD Utility Converts Legacy Dates**

**Updated Master File Created by REBUILD/DATE NEW**

**Action Taken on a Date Field During REBUILD/DATE NEW**

**NF653: Displaying Base Dates in FOCUS Reports**

**NF659: CHECK FILE HOLD ALL**

**NF700: New Date Math Functions for the Year 2000**

**New Date Function Capabilities**

**Weekday Units**

**Business Day Units**

**Holidays**

**New Date Math Functions in MAINTAIN**

**NF703: Displaying Invalid Smart Dates in Reports**

**NF705: Enhancement to the YRTHRESH Command**

**NF708: Enhancement to the TODAY Subroutine**

**NF709: Displaying a Date Variable Without Separators**

**NF710: Field FORMAT=YYJUL**

**NF711: Altering Your System Date for Testing Purposes**

**NF713: MSO Log Changes**

**Sample MSO Log**

**NF714: LE Support**

**Recommended IBMLE Settings**

**Determining Proper IBMLE Settings**

## **7.0.8 New Features**

**NF550: EDA/MSO Console Display for IMS PSB**

**NF564: Pooled Tables**

**Overview**

**Memory Needs**

**Report Size Estimates**

**FOCPOOLT**

**Reporting statistics**

**Sort Selection**

**Managing Memory**

**Common Selection Criteria**

**Reporting from non-Relational Databases**

**Reporting from Relational Databases**

**Pooled Tables in Batch Mode**

**Trace Facility**

**NF564: Pooled Tables (*continued*)**

**Tuning Applications**

**Pooled Tables Example**

Single TABLE Clusters

Subpool Boundary Conditions

**Pooled Tables Installation Instructions**

Commands for the FOCPARM file

**Frequently Asked Questions**

**NF566: MSO/CICS Cooperative Processing**

**MSO FOCEXEC Cooperative Processing Service**

**MSO/CICS Cooperative Processing Services**

**CMSORCV Function Codes**

**Reconnection Capability**

**Special Considerations**

Suspend key

Previous API

**NF568: DB2 Interface IF-THEN-ELSE Optimization**

**NF571: DB2 Interface SET ISOLATION Command**

**NF572: Invisible Ordered Character and Ordered Numeric Data Type Key Support**

**NF574: System 2000 Interface Trace Facility**

**NF579: Assigning Screening Conditions to a File for Reporting Purposes**

Using Filters

Filters and JOINS

**NF583: Teradata Outer Join Optimization**

**NF586: Expanding Byte Precision for COUNT and LIST**

**NF593: IUCV CMS SU**

**NF594: JAVA Report Assist**

**NF605: Date Handling for the Year 2000 in FOCUS**

**Date Literals Interpretation Table**

**NF607: TABLA Enhancements**

**NF609: Sink Validation of Userids in CMS**

**NF617: Automatic Allocation of FOCUS Files**

**NF619: -HTMLFORM SAVE**

**NF620: Year 2000 Subroutines**

**Date Literals Interpretation Table**

**NF623: Increasing the Number of Verbs in a Report Request**

**NF626: JAVA Graph Wizard**

**NF628: Automatic Activation of Web Interface for Web Browser Users**

**NF630: Querying Which PTFs Have Been Applied for a Specific Release**

**NF631: Extended Plists**

**NF640: Dynamic Language Environment (LE) Support**

**NF642: Increased DEFINE Limitation**

**NF645: WEBHOME**

**NF647: Extended Support for Scandinavian External Sort**

**Project 2000 - Phase III**

## **7.0.7M New Features**

**MAINTAIN Updates**

## **7.0.7 New Features**

**NF580: Web Interface for FOCUS**

**Web Interface Features**

## **7.0.6 New Features**

**NF502: MSO VTAM Logon Time-out**

**NF512: IMS Interface - Automatic Index Selection Using AutoSelect**

**NF517: ADABAS Dynamic Security**

**Overriding Default Passwords in Specific Files**

**NF523: Cross-Century Dates in FOCUS Applications - Phase II**

**NF526: PRINTPLUS**

**Usage**

**Special Considerations**

**NF530: Language Environment (LE) Support**

**NF531: 31 Bit I/O**



**NF532: MSO Monitoring and Statistics**

**NF536: Multi-image FOCSORT**

**NF538: ADABAS Dynamic Database Number**

**NF539: Outer Join Optimization**

**NF540: Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS**

**NF542: Optimization of Joins Between Heterogeneous File Types**

**NF543: FOCUS Personal Agent - TCP/IP Connectivity Option for VM/CMS FOCUS**

**NF544: FOCUS Personal Agent - TCP/IP Connectivity Option for TSO FOCUS**

**NF546: Enhancements to JOIN  
Enabling Datatype Conversion**

**NF547: EDA to MSO Bridge**

**NF549: Interpreting Quotation Marks Within Quote-Delimited Literal Strings**

**NF552: Estimating SORTWORK Sizes for an External Sort**

**NF553: Enhanced Message Routing**

**NF554: Load Balancing for the Multi-Session Option (MSO)**

**Overview**

**MSO Load Balancing Requirements**

**MSO Load Balancing Approach**

**Load Balancing Sequence**

**Setting Load Balancing Defaults in the IBI Subsystem**

**MSO Load Balancing Configuration Parameters**

**Minimum Parameters for Load Balancing**

**Load Balancing Logon Procedures**

**Planning for MSO Load Balancing**

**MSO Load Balancing Requirements**

**IBI Load Balancing Defaults**

**Consistent Destination Environment**

**Define Load Balancing Group Names to SAF**

**Consistently use the APPLICATION= parameter in all Service Blocks**

**MSO Region Capacity**

**All MSO Regions should have a Console available**

**Minimize dividing the MSO Load Balancing group**

**Operational Trouble Shooting**

**CICS Problem Determination Procedures**

**Operational Benefits of MSO Load Balancing**

**MSO Glossary of Terms**

**NF576: MSO Dynamic VTAM Re-configuration**

**NF578: FOCUS Personal Agent - Interruptible Server for VM and MVS FOCUS**

**Project 2000 - Phase II**

## **7.0.5 New Features**

**NF493: Full Support for SDSF under MSO and TSO**

**NF494: FOCUS Client: Remote Data Access via EDA/SQL**

**Client/Server Computing and Middleware**

**Overview: Using FOCUS to Access Data on EDA/SQL Servers**

Establishing and Configuring the FOCUS User Environment

Prerequisites

**Remote Execution**

Logging on with REMOTE Commands

Sending Requests to a Remote Server

Viewing System and Error Messages

Terminating the Remote Session: REMOTE FIN

Checking on Remote Session Parameter Settings: ? REMOTE

**Distributed Execution: The EDA Interface (or “SUFFIX=EDA”)**

How Location Transparency Works

Logging onto the Server

Joining Files Across Platforms

Issuing SQL Commands to the EDA/SQL Server

Executing Stored Procedures

Using SQL

**NF497: Project 2000—Cross-Century Dates in FOCUS Applications**

Using DEFCENT and YRTHRESH to Establish a Century Window

Support for Date Variables in Dialogue Manager

**NF499: Scrolling Report Headings in HotScreen**

**NF500: Keyed Retrievals from FOCUS Extract Files**

**Uses for Keyed Retrieval From an Extract File**

**NF501: Public and Private DDname for MSO**

**How PRIVATEDD Works**

**Console File Allocation Displays**

**? TSO DDNAME? DDname\***

**PRIVATEDD Restrictions for User Written Subroutines**

**Planning for the Implementation of PRIVATEDD**

**NF509: MINIO - New FOCUS Database Access Method Available Under MVS**

**How MINIO Works**

**NF510: Date and Time Stamp in Reports**

**NF513: Redefining Fields in Non-FOCUS Files**

**NF518: VSAM Support**

**NF519: Field Level PFKeys**

**NF520: Dynamically Changing Attributes of MAINTAIN Winforms**

**NF521: Enhancements to Objects in MAINTAIN**

**List Boxes**

**Radio Groups**

**CDN Support**

**MAINTAIN Improved SU Performance**

**National Language Support**

## **7.0.3 New Features**

**NF491: Distinct Operator**

**NF495:ADABAS Interface: Multifetch/Prefetch Options**

**INVOKING MULTIFETCH/PREFETCH**

**ADABAS Commands Supporting the Multifetch/Prefetch Feature**

**Tracing the FETCH Feature**

**NF498: Enabling the CA-DATACOM/DB CBS Trace**

**Missing Values**

## **7.0.1 New Features**

**NF414: The MAINTAIN Facility**

**Introduction to MAINTAIN**

**Using MAINTAIN to Manage Data**

**NF415: SET SAVEMATRIX**

**NF420: Double-Byte Character Set K Format and G Prefix**

**The K Format**

**The G Prefix**

**NF421: Determining Which FOCEXEC is Running**

**NF423: Specifying up to 256 Verb Objects**

**NF425: External Indices for FOCUS Databases**

**Creating an External Index**

**Concatenating Index Databases**

**Positioning Indexed Fields**

**Activating an External Index**

**NF426: AUTOPATH**

**Initiating AUTOPATH**

**NF427: Longer Length for HOLD FORMAT LOTUS Files**

**NF428: Increased Size of FOCSORT**

**NF429: Generalized Listings of DDNAMEs**

**NF431: Universal Concatenation**

**The MORE Subcommand**

**Fieldname and Format Matching**

**MATCH FILE Considerations**

**NF432: Renaming/Rejustifying Row and Column Total Labels**

**NF433: Improved Handling of Text Fields In TED**

**Displaying or Printing a Text Field in TABLE**

**NF434: Larger FOCUS Databases**

**NF435: Increased Number of Literal Values in a File**

**NF436: Checking Current Language Settings**

**NF437: AUTOADBS**

**Documentation in the Master and Access File Descriptions**

**How to Use the AUTOADBS Facility**

**Starting AUTOADBS**

**The File Selection Screen**

**The Access File Attribute Screen**

**The Child Selection Screen**

**The IXFLD Selection Screen**

**The KEYFLD Selection Screen**

**Background Execution**

**The Generated Descriptions**

**File and Segment Attributes**

**Access File Attributes**

**Field Attributes**

**Differences Between ADABAS=OLD and NEW**

**Changes to the Generated Descriptions**

**Creating a Record of Masters Generated**

**Search Order for Parameter Log File (MVS only)**

**NF438: MODEL 204 Interface Enhancements**

**NF440: Improved Page Handling - SET TRACKIO**

**NF441: External Sort**

**Requirements**

**Displaying External Sort Messages**

**AUTOTABLEF**

**NF443: Large Packed Fields**

**NF444: ASIS Function**

**NF446: Using StyleSheets**

**NF448: FOCPARM Enhancements**

**NF449: Increased Number of Indices**

**NF452: Capturing SET Parameter Values**

**NF455: FOCUS File Date and Time Stamp**

**Date/Time Stamp**

**Changes to the REBUILD command**

**NF456: VSAM Data and Index Buffers**

**NF457: The IBI MVS Subsystem**

**NF460: Changing the Default CALLTYPE**

**NF465: Changes to the Catalog Search in FOCUS**

**NF466: Controlling IMS Access via DBCTL**

**Advantages of DBCTL**

**NF467: Automatic Indexed Retrieval (AUTOINDEX)**

**Implications When a Request Contains an Indexed View**

**NF468: HiperBudget**

**NF469: DRDA Support Enhancements to the DB2 Interface**

**NF470: Loading Access File Descriptions**

**NF471: Enhanced ? SET Command**



**NF472: Enhancement to the TED Command in MVS**

**NF473: The New ADABAS Interface**

**NF474: APF Internal Authorization  
Implementation**

**NF476: Usability Enhancements**

**ASC/DESC Synonyms for the KEYORDER Attribute**

**Sample Runtime CLISTs and JCL for the MVS Relational Interfaces**

**Full Support for the Long DECIMAL Datatype**

**SET INCLUDE SUBTREE Segment Activation Behavior is the Default**

**Enhancements to the Interface EXPLAIN Facilities for DB2 and SQL/DS**

**User Specification of DB2 Index Space Parameters**

**Enable Collection-Id Parameter for SQL/DS Interface GENUSQL EXEC**

**Relaxing of Join Field Data Length Requirement for Alphanumeric Fields**

**NF477: FASTPDS**

**NF478: MSO CONSOLE Browser**

**NF479: The New MSO/CICS Interface**

**NF480: Fast Logon Enhancements**

**NF481: DYNAM Utilities Menu**

**NF482: IMS Enhancements via SET IMS=NEW**

Accessing the New IMS Interface

The XMI Server

Comma-Delimited FOCPSB

The New Trace Facility

New SET Commands

The IMDTEST Verification Program

Improved Security

The IMS Access File

General Enhancements

**NF483: The MSO Resource Manager (MRM)**

**NF484: Allowing VSAM File Allocation in MSO JCL**

**NF485: Dynamic GETPRV Exit**

**NF486: Dynamically Setting the Addressing Mode**

**NF487: Enhancement to the ZCOMP1 User Exit**

**NF488: AUTODATACOM**

**NF490: Online Release Information**

**NF496: Static SQL for TABLE Requests**

Requirements for Static SQL for TABLE

Static TABLE Module Creation

Static TABLE Module Execution

SQL COMPILE and SQL RUN Processing

SQL Host Variable Length Considerations

Static FOCEXEC Creation for DB2

Run-Time Requirements

Processing and Security Overview

DB2 Static FOCEXEC Example

Plan Management in DB2

Basic Plan Management

Extended Plan Management

Static FOCEXEC Creation for SQL/DS

Run-Time Requirements

Processing and Security Overview

SQL/DS Static FOCEXEC Example

Verification and Control of Host Variable Placement

Usage Restrictions

Resource Restrictions

**NF504: Installation Enhancement for the SQL/DS Interface**

### **NF505: Optimization Enhancements**

**Interface-Managed Native Join Optimization**

**Selective Optimization of FST. and LST. Aggregation Operators**

**Optimization of DEFINE-Based Screening Conditions on Single Segments**

**Optimization of WHERE/MISSING Clauses as SQL WHERE/NULL Clauses**

**Optimization of WHERE/FROM-TO Clauses as SQL BETWEEN Clauses**

**Oracle Interface Optimization Enhancements**

**Optimization of IF/WHERE READLIMIT as WHERE ROWNUM**

**Enabling of Array Blocking for SELECT and INSERT Requests**

**NF507: FSTRACE for the CA-DATACOM/DB Interface**

**NF508: DYNAM Support for BUFNI and BUFND**

## **Index**

## 7.0.9 New Features

### Fusion

[NF575: Fusion](#)

### General Enhancements

[NF716: Euro Currency Support](#)

[NF655: FOCPROF - The System Wide Profile](#)

[NF735: Enhancement to ? SET](#)

[NF746: Leading Zeros](#)

[NF656: Controlling REBUILD Messages](#)

[NF740: Changes to the REBUILD Prompt](#)

[NF660: Multi-volume Support in MVS FOCUS](#)

[NF670: DYNAM Support for Unit Count](#)

[NF718: DYNAM Support for Existing Relative GDG  
Numbers](#)

[NF745: ? PTF Enhancements](#)

### Reporting Enhancements

[NF691: Escape Character for LIKE Predicate](#)

[NF744: HOLD FORMAT EXCEL](#)

[NF748: HOLD FORMAT WP With Carriage Control](#)

## Performance Enhancements

[NF654: HOLD From External Sort](#)

[NF597: Aggregation by External Sort](#)

[NF728: Changing Retrieval Order with Aggregation](#)

## Web Interface for FOCUS

[NF683: Web Interface support for Maintain Winforms](#)

[NF684: PCHOLD for Non-Html Files](#)

[NF730: Hold Format PDF](#)

## Relational Interfaces

[NF720: SQLJOIN OUTER Setting for Relational Interfaces](#)

## Teradata Interface

[NF652: Teradata Interface Kanji Support](#)

## Model 204 Interface

[NF673: Model 204 Interface Account Split](#)

## CA-IDMS Interface

[NF584: Dynamically Setting the IDMS DBNAME and  
DICTNAME](#)

## FOCUS Client

[NF722: FOCUS Client DNS Names Support](#)

## NF575: Fusion

FOCUS Version 7.0 release 9 introduces the Fusion database for CMS and MVS FOCUS. Fusion is a high performance database whose unique Multi Dimensional Indexing (MDI) architecture extends the scope of high-speed multi-dimensional query performance.

For complete details, see the [\*Fusion User's Manual for EDA 4.2 and FOCUS 7.0\*](#) (DN3700041.1198).

## NF716: Euro Currency Support

With the introduction of the euro currency, businesses need to maintain books in two currencies, add new fields to their database designs, and perform new types of currency conversions. This new feature gives FOCUS the ability to perform currency conversions according to the rules specified by the European Union. Before you can use FOCUS to process currency conversions, you must:

- Create a currency database with the currency IDs and exchange rates you will use. See [Creating the Currency Database](#).
- Identify fields in your data sources that represent currency data. See [Identifying Fields That Contain Currency Data](#).
- Activate your currency database. See [Activating the Currency Database](#).

After you complete these preliminary steps, you can perform currency conversions. See [Processing Currency Data](#).

**Note:** Operating system vendors are in the process of integrating the euro currency symbol into their environments. As the euro symbol becomes available, FOCUS will support it.



## Converting Currencies

Although the euro was introduced in 11 countries of the European Union on January 1, 1999, it will not immediately replace local currencies in those countries. During the transition period from 1999 to 2002, both traditional currencies and the euro will be used simultaneously for accounting purposes and non-cash transactions in each participating country. Euro cash will not be introduced until January 1, 2002, and by July 1, 2002 the traditional currencies will no longer be legal tender.

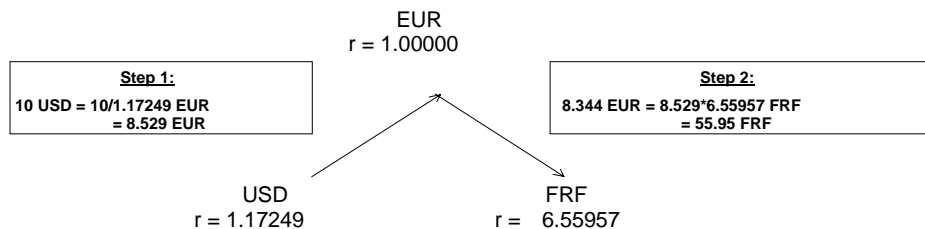
On the last day of 1998, the European Union set fixed exchange rates between the euro and the traditional national currency in each of the 11 adopting member nations. While the exchange rates within “Euroland” will remain fixed, exchange rates between the euro and non-euro countries will continue to vary freely and, in fact, several rates may be in use at one time (for example, actual and budgeted rates).

The European Union has established the following rules for currency conversion:

- The exchange rate must be specified as a decimal value,  $r$ , with six significant digits (not six decimal places). For example, 123.456 has six significant digits but not six decimal places. This rate will establish the following relationship between the euro and the particular national currency:  
1 euro =  $r$  national units
- To convert from the euro to the national unit, multiply by  $r$  and round the result to two decimal places.

- To convert from the national currency to the euro, divide by  $r$  and round the result to two decimal places.
- To convert from one national currency to another, first convert from one national unit to the euro, rounding the result to at least three decimal places (FOCUS rounds to exactly three decimal places). Then convert from the euro to the second national unit, rounding the result to two decimal places. The following diagram illustrates this two-step conversion process known as *triangulation*:

Converting 10 US Dollars to French Francs



## Preparing FOCUS to Process Currency Conversions

Although 11 or more currencies in the European Union will be converting to the euro, more than 100 currencies have a recognized status worldwide. In addition, you may need to define custom currencies for some applications.

You identify your currency codes and rates by creating a currency database. The currency database can be any type of data source that FOCUS can access.

### Creating the Currency Database

For each type of currency you need, you must supply the following values in your currency database:

- A three-character code to identify the currency, such as USD for U.S. dollars or BEF for Belgian francs. (For a partial list of recognized currency codes, see [Sample Currency Codes](#).)
- One or more exchange rates for the currency.

There is no limit to the number of currencies you can add to your currency database; the currencies you can define are not limited to official currencies and, therefore, the currency database can be fully customized for your applications.

We strongly recommend that you create a separate database for the currency data rather than adding the currency fields to another data source. A separate currency database enhances performance and minimizes resource utilization because FOCUS loads the currency database into memory before you perform currency conversions.

## Syntax      How to Specify Currency Codes and Rates in a Master File

The currency database can be any type of data source accessible by FOCUS (for example, FOCUS, FIX, DB2, or VSAM). The currency Master File must have one field that identifies each currency ID you will use and one or more fields to specify the exchange rates.

The syntax is

```
FIELD = CURRENCY_ID,      FORMAT = A3,                ACTUAL = A3 , $
FIELD = rate1,           FORMAT = {D12.6|numeric_format1}, ACTUAL = A12,$
.
.
.
FIELD = raten,           FORMAT = {D12.6|numeric_formatn}, ACTUAL = A12,$
```

where:

CURRENCY\_ID

Is the required field name. The values stored in this field are the three-character codes that identify each currency, such as USD for U.S. dollars. Each currency ID can be a universally recognized code or a user-defined code. **Note:** FOCUS automatically recognizes the code EUR; you should *not* store this code in your currency database. See [Sample Currency Codes](#) for a list of common currency codes.

*rate1,...,raten*

Are types of rates (such as BUDGET, FASB, ACTUAL) to be used in currency conversions. Each rate is the number of national units that represent one euro.

*numeric\_format1,...,numeric\_formatn*

Are the display formats for the exchange rates. Each format must be numeric. The recommended format, D12.6, ensures that the rate is expressed with six significant digits as required by the European Union conversion rules. Do not use Integer format (I).

ACTUAL An

Is required only for non-FOCUS data sources.

**Note:** The maximum number of fields in the currency database must not exceed 255 (that is, the CURRENCY\_ID field plus 254 currency conversion fields).

### Example Specifying Currency Codes and Rates in a Master File

The following Master File for a comma-delimited currency database specifies two rates for each currency, ACTUAL and BUDGET:

```
FILE = CURRCODE, SUFFIX = COM,$
FIELD = CURRENCY_ID, FORMAT = A3, ACTUAL = A3 , $
FIELD = ACTUAL, ALIAS = , FORMAT = D12.6, ACTUAL = A12 , $
FIELD = BUDGET, ALIAS = , FORMAT = D12.6, ACTUAL = A12 , $
```

The following is sample data for the currency database defined by this Master File:

```
FRF, 6.55957, 6.50000,$
USD, 1.17249, 1.20000,$
BEF, 40.3399, 41.00000,$
```

## Identifying Fields That Contain Currency Data

Once you have created your currency database, you must identify the fields in your data sources that represent currency values. To designate a field as a currency-denominated value (a value that represents a number of units in a specific type of currency) add the CURRENCY attribute to one of the following:

- The FIELD specification in the Master File.
- The left side of a DEFINE or COMPUTE.

### Syntax      How to Identify a Currency Value

Use the following syntax to identify a currency-denominated value:

- In a Master File

```
FIELD = currfield, FORMAT = numeric_format, ,  
CURR = {curr_id|codefield} , $
```

- In a DEFINE in the Master File

```
DEFINE currfield/numeric_format CURR curr_id = expression ; $
```

- In a DEFINE FILE command

```
DEFINE FILE filename  
currfield/numeric_format CURR curr_id = expression ;  
END
```

- In a COMPUTE command

```
COMPUTE currfield/numeric_format CURR curr_id = expression ;
```

where:

*filename*

Is the name of the file for which this field is defined.

*currfield*

Is the name of the currency-denominated field.

*numeric\_format*

Is a numeric format. Depending on the currency denomination involved, the recommended number of decimal places is either two or zero. Do not use I or F format.

CURR

Indicates that the field value represents a currency-denominated value. CURR is an abbreviation of CURRENCY, which is the full attribute name.

*curr\_id*

Is the three-character currency ID associated with the field. In order to perform currency conversions, this ID must either be the value EUR or match a CURRENCY\_ID value in your currency database.

*codefield*

Is the name of a field, qualified if necessary, that contains the currency ID associated with *currfield*. The code field should have format A3 or longer and is interpreted as containing the currency ID value in its first three bytes. For example:

```
FIELD = PRICE, FORMAT = P12.2C, ..., CURR = TABLE.FLD1,$  
. . .  
FIELD = FLD1, FORMAT = A3, ..., $
```

The field named FLD1 contains the currency ID for the field named PRICE.

*expression*

Is a valid expression.

### **Example** Identifying a Currency-denominated Field

Assume that the currency database contains the currency ID value BEF (Belgian francs).

If the FINANCE data source contains a field named PRICE that is denominated in Belgian francs, the description of PRICE in the FINANCE Master File could be:

```
FIELD = PRICE, ALIAS=, FORMAT = P17.2, CURR=BEF,$
```

## Activating the Currency Database

Before you can perform currency conversions, you must bring the relevant currency database into memory by issuing the SET EUROFILE command.

### **Syntax** How to Activate Your Currency Database

Issue the following command at the FOCUS command prompt, in a FOCEXEC, or in any supported profile

```
SET EUROFILE = {cdname|OFF}
```



where:

*ddname*

Is the name of the Master File for the currency database. There is no default value for EUROFILE. The ddname must refer to a data source known to FOCUS and accessible by FOCUS in read-only mode.

OFF

Deactivates the currency database and removes it from memory.

During your FOCUS session, if you want to access a different currency database, you can re-issue the SET EUROFILE command.

**Note:**

- You cannot append any additional SET parameters to the SET EUROFILE command line. For example, the PAUSE setting would be lost if you issued the following command:

```
SET EUROFILE=filename , PAUSE=OFF
```

- You cannot issue the SET EUROFILE command within a TABLE request.

## Syntax      How to Determine the Currency Database in Effect

If you want to determine which currency database is in effect, issue the ? SET ALL command or the new EUROFILE query command:

```
? SET EUROFILE
```

## Example      Determining the Currency Database in Effect

Assume the currency database is named CURRCODE.

If you issue the following commands:

```
set eurofile = currcode  
? set eurofile
```

FOCUS returns the following response:

```
EUROFILE          CURRCODE
```

## Reference SET EUROFILE Error Messages and Notes

Issuing the SET EUROFILE command when the currency database Master File does not exist generates the following error message:

```
(FOC205) THE DESCRIPTION CANNOT BE FOUND FOR FILE NAMED: ddname
```

Issuing the SET EUROFILE command when the currency Master File specifies a FOCUS database and the associated FOCUS database does not exist generates the following error message:

```
(FOC036) NO DATA FOUND FOR THE FOCUS FILE NAMED: name
```

**Note for Pooled Table users:** The SET EUROFILE command creates a pool boundary.

## Processing Currency Data

After you have created your currency database, identified the currency-denominated fields in your data sources, and activated your currency database, you can perform currency conversions.

Each currency ID in your currency database generates a virtual conversion function whose name is the same as its currency ID. For example, if you added BEF to your currency database, a virtual BEF currency conversion function will be generated.

The euro function, EUR, is supplied automatically with FOCUS. You do not need to add the EUR currency ID to your currency database.

## **Syntax**      **How to Convert Currency Data**

Use the following syntax for calling a currency conversion function

- In a TABLE, GRAPH, or MODIFY procedure:

```
DEFINE FILE filename  
result/format [CURR curr_id] = curr_id(infield, rate1 [,rate2]);  
END
```

or

```
COMPUTE result/format [CURR curr_id] = curr_id(infield, rate1  
[,rate2]);
```

- In a Master File:

```
DEFINE result/format [CURR curr_id] = curr_id(infield, rate1  
[,rate2]);$
```

where:

*filename*

Is the name of the file for which this field is defined.

*result*

Is the converted currency value.

### *format*

Must be a numeric format. Depending on the currency denomination involved, the recommended number of decimal places is either two or zero. Do not use I or F format. The result will always be rounded to two decimal places, which will display if the format allows at least two decimal places.

### *curr\_id*

Is the currency ID of the result field. This ID must be the value EUR or match a currency ID in your currency database; any other value generates the following message

```
(FOC263) EXTERNAL FUNCTION OR LOAD MODULE NOT FOUND: curr_id
```

**Note:** The CURR attribute on the left side of the DEFINE or COMPUTE identifies the result field as a currency-denominated value which can be passed as an argument to a currency function in subsequent currency calculations. Adding this attribute to the left side of the DEFINE or COMPUTE does not invoke any format or value conversion on the calculated result.

### *infield*

Is a currency-denominated value. This input value will be converted from its original currency to the *curr\_id* denomination. If the *infield* and *result* currencies are the same, no calculation is performed and the *result* value is the same as the *infield* value.

### *rate1*

Is the name of a rate field from the currency database. The *infield* value is divided by its currency's *rate1* value to produce the equivalent number of euros.

If *rate2* is not specified in the currency calculation and triangulation is required, this intermediate result is then multiplied by the *result* currency's *rate1* value to complete the conversion.

In certain cases, you may need to provide different rates for special purposes. In these situations you can specify any field or numeric constant for *rate1* as long as it indicates the number of units of the *infield* currency denomination that equals one euro.

### *rate2*

Is the name of a rate field from the currency database. This argument is only used for those cases of triangulation in which you need to specify different rate fields for the *infield* and *result* currencies. It is ignored if the euro is one of the currencies involved in the calculation.

The number of euros that was derived using *rate1* is multiplied by the *result* currency's *rate2* value to complete the conversion.

In certain cases, you may need to provide different rates for special purposes. In these situations you can specify any field or numeric constant for *rate2* as long as it indicates the number of units of the *result* currency denomination that equals one euro.

**Note:** MAINTAIN does not support these currency conversion functions.

## Example Converting Currencies

Assume that the currency database contains the currency IDs USD and BEF, and that PRICE is denominated in Belgian francs as follows:

```
FIELD = PRICE, ALIAS=, FORMAT = P17.2, CURR=BEF,$
```

- The following example converts PRICE to euros and stores the result in PRICE2 using the BUDGET conversion rate for the BEF currency ID:

```
COMPUTE PRICE2/P17.2 CURR EUR = EUR(PRICE, BUDGET);
```

- This example converts PRICE from Belgian francs to US dollars using the triangulation rule:

```
DEFINE PRICE3/P17.2 CURR USD = USD(PRICE, ACTUAL);$
```

First PRICE is divided by the ACTUAL rate for Belgian francs to derive the number of euros rounded to three decimal places. Then this intermediate value is multiplied by the ACTUAL rate for US dollars and rounded to two decimal places.

- The following example uses a numeric constant for the conversion rate:

```
DEFINE PRICE4/P17.2 CURR EUR = EUR(PRICE,5);$
```

- The next example uses the ACTUAL rate for Belgian francs in the division and the BUDGET rate for US dollars in the multiplication:

```
DEFINE PRICE5/P17.2 CURR USD = USD(PRICE, ACTUAL, BUDGET);$
```

## Reference Currency Calculation Processing and Messages

The result is always calculated with very high precision, 31 to 36 significant digits, depending on platform. The precision of the final result is always rounded to two decimal places. In order to display the result to the proper precision, its format must allow at least two decimal places.

Issuing a TABLE request against a Master File that specifies a currency code not listed in the active currency database generates the following message:

**(FOC1911) CURRENCY IN FILE DESCRIPTION NOT FOUND IN DATA**

A syntax error or undefined fieldname in a currency conversion expression generates the following message:

**(FOC1912) ERROR IN PARSING CURRENCY STATEMENT**

## Reference Sample Currency Codes

The following rates were in effect on December 31, 1998. Euroland countries as of that date are marked with an asterisk (\*). Their rates are fixed and will not change; the rates for other countries can change over time:

Country	Currency Code	Rate
Austria*	ATS	13.7603
Belgium*	BEF	40.3399
Canada	CAD	1.7978
Denmark	DKK	7.46215

<b>Country</b>	<b>Currency Code</b>	<b>Rate</b>
European Union	EUR	1
Finland*	FIM	5.94573
France*	FRF	6.55957
Germany*	DEM	1.95583
Greece	GRD	328.6
Ireland*	IEP	0.787564
Italy*	ITL	1936.27
Japan	JPY	133.149
Luxembourg*	LUF	40.3399
Netherlands*	NLG	2.20371
Norway	NOK	8.91039
Portugal*	PTE	200.482
Spain*	ESP	166.386
Sweden	SEK	9.52669



Country	Currency Code	Rate
Switzerland	CHF	1.61093
UK	GBP	0.706739
USA	USD	1.17249

### Example Converting U.S. Dollars to Euros, French Francs, and Belgian Francs

Assume PRICE is denominated in U.S. dollars and ACTUAL is the name of a rate in the currency database. Using the currency conversion rates from [Sample Currency Codes](#), the following FOCEXEC converts PRICE to euros, French francs, and Belgian francs:

```

-* CURRCODE IS THE CURRENCY DATABASE
-* CURRDATA IS THE DATA SOURCE WITH CURRENCY-DENOMINATED FIELDS

-* THE FOLLOWING FILEDEFS ARE FOR RUNNING UNDER CMS
CMS FILEDEF CURRCODE DISK CURRCODE TEXT A
CMS FILEDEF CURRDATA DISK CURRDATA TEXT A

-* THE FOLLOWING ALLOCATIONS ARE FOR RUNNING UNDER MVS
-* DYNAM ALLOC FILE CURRCODE DA USER1.FOCEXEC.DATA(CURRCODE) SHR REU
-* DYNAM ALLOC FILE CURRDATA DA USER1.FOCEXEC.DATA(CURRDATA) SHR REU

SET EUROFILE = CURRCODE

```

```
DEFINE FILE CURRDATA
PRICEEUR/P17.2 CURR EUR = EUR(PRICE, ACTUAL);
END
```

```
TABLE FILE CURRDATA
PRINT PRICE PRICEEUR AND COMPUTE
PRICEFRF/P17.2 CURR FRF = FRF(PRICE, ACTUAL);
PRICEBEF/P17.2 CURR BEF = BEF(PRICE, ACTUAL);
END
```

This request generates the following report:

PAGE 1

PRICE	PRICEEUR	PRICEFRF	PRICEBEF
-----	-----	-----	-----
5.00	4.26	27.97	172.01
6.00	5.12	33.57	206.42
40.00	34.12	223.78	1376.20
10.00	8.53	55.95	344.06

**Note:** You cannot use the derived euro value PRICEEUR in a conversion from USD to BEF. PRICEEUR has two decimal places (P17.2), not three, as the triangulation rules require. Therefore, PRICEEUR yields the following inaccurate result (see PRICEBEF above) and is not valid as the intermediate value in a currency conversion that requires triangulation:

```
COMPUTE PRICENEW/P17.2 CURR BEF = BEF(PRICEEUR, ACTUAL);
```

```
PRICENEW
```

```
-----
```

```
171.85
```

```
206.54
```

```
1376.40
```

```
344.10
```

## NF744: HOLD FORMAT EXCEL

FOCUS can now format report output as a Microsoft® Excel spreadsheet. When you use the HOLD FORMAT EXCEL syntax, FOCUS creates a binary file containing all columns of the report output with their column headings.

If you are using the Web Interface for FOCUS, the Excel spreadsheet can be downloaded automatically to your Web Browser. If you do not have the Web Interface, you can transfer the spreadsheet file to any environment that supports Excel, such as a PC running Microsoft Windows®; the recommended transfer protocol is FTP in binary mode.

You can also store the Excel file on a web server where users can display it using any browser configured with Excel as a plug-in.

### Syntax      How to Create an Excel Spreadsheet

The syntax is.

```
[ON TABLE] HOLD [AS filename] FORMAT EXCEL
```

where:

*ON TABLE*

Is required syntax if you create the Excel file within a report request.

*filename*

Assigns a name to the Hold file. The default name is HOLD.

In MVS, unless you allocate this ddname to a permanent file, FOCUS allocates it to a temporary data set.

In CMS, this name becomes the file name. The file type is XLS.

By default, the extension for Microsoft Excel files is .xls in environments that support file names with extensions. Therefore, the file name after transfer is filename.xls.

## Syntax      How to Create and Download an Excel File With the Web Interface for FOCUS

If you are using the Web Interface for FOCUS, you can create the Excel spreadsheet and automatically download it to your Web Browser with the following syntax:

```
[ON TABLE] PCHOLD FORMAT EXCEL
```

where:

```
ON TABLE
```

Is required syntax if you create the Excel file within a report request.

On the PC, the name of the Excel spreadsheet file is *hold.xls*.

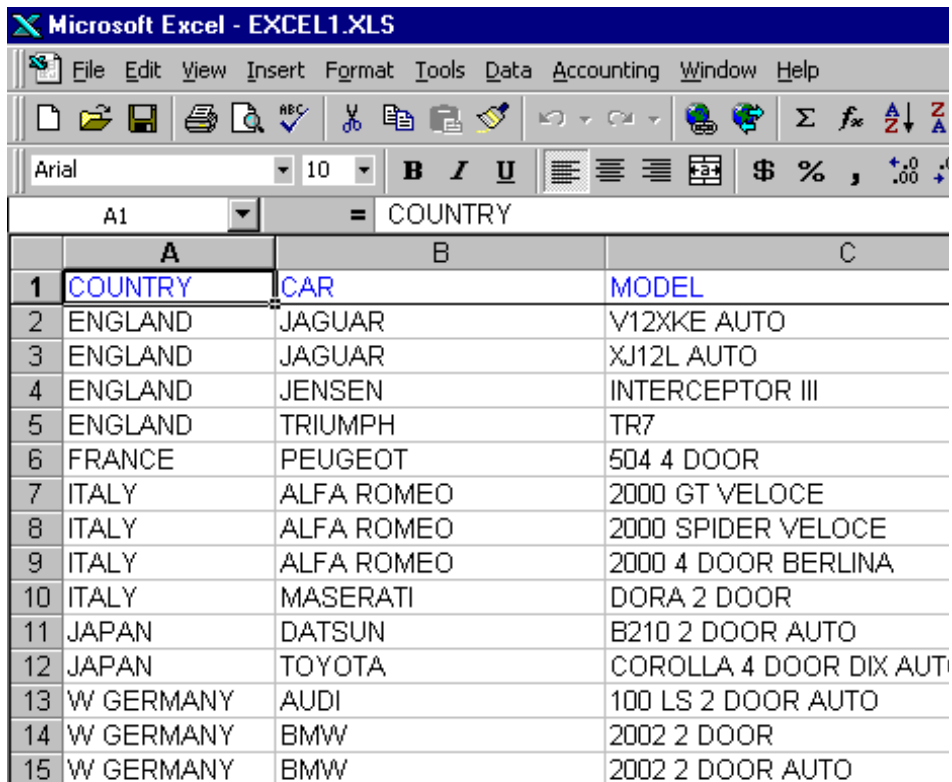
For a detailed description of how to download report output using the Web Interface for FOCUS and PCHOLD, see [NF730: Hold Format PDF](#).

## Example      Creating an Excel Spreadsheet in a Report Request

Consider the following request run in CMS:

```
TABLE FILE CAR  
PRINT CAR MODEL RETAIL_COST DEALER_COST  
BY COUNTRY  
ON TABLE HOLD AS EXCEL1 FORMAT EXCEL  
END
```

This request creates the file EXCEL1 XLS A. After transferring this file (using FTP in binary mode) to a file named *excel1.xls* on your PC, you can open it in Excel:



The screenshot shows the Microsoft Excel interface with the following data in the spreadsheet:

	A	B	C
1	COUNTRY	CAR	MODEL
2	ENGLAND	JAGUAR	V12XKE AUTO
3	ENGLAND	JAGUAR	XJ12L AUTO
4	ENGLAND	JENSEN	INTERCEPTOR III
5	ENGLAND	TRIUMPH	TR7
6	FRANCE	PEUGEOT	504 4 DOOR
7	ITALY	ALFA ROMEO	2000 GT VELOCE
8	ITALY	ALFA ROMEO	2000 SPIDER VELOCE
9	ITALY	ALFA ROMEO	2000 4 DOOR BERLINA
10	ITALY	MASERATI	DORA 2 DOOR
11	JAPAN	DATSUN	B210 2 DOOR AUTO
12	JAPAN	TOYOTA	COROLLA 4 DOOR DIX AUTO
13	W GERMANY	AUDI	100 LS 2 DOOR AUTO
14	W GERMANY	BMW	2002 2 DOOR
15	W GERMANY	BMW	2002 2 DOOR AUTO

## NF730: Hold Format PDF

FOCUS can now generate output in Adobe® Portable Document Format (PDF). This feature enables Web Interface users to produce reports with all PDF formatting options (for example, headings, footings, and titles) correctly aligned on the physical pages. PDF also supports StyleSheets incorporating drill-downs and links to URLs.

### Required Software Configuration

Adobe Acrobat Reader® Version 3.01 or higher is required to display PDF output, however, no third-party products are needed to *produce* it. The Adobe Acrobat Reader is Internet shareware for Windows 95, Windows NT, UNIX, and Macintosh, and is available free from their Web site, <http://www.adobe.com>.

Reports viewed with Adobe Acrobat Reader look precisely as if printed. By configuring the Acrobat Reader as a plug-in on your Web browser, PDF output displays directly inside the browser window in printed format without additional setup or configuration.

Browser users who have not installed Acrobat Reader can save PDF files to disk or download them to a PC when prompted by the browser and then transfer them later to a machine with the Acrobat Reader to display them. Users can then either run the standalone Acrobat Reader program or use a browser with the PDF plug-in to view the PDF files.

## Syntax      How to Create PDF Output

The syntax is:

```
ON TABLE {HOLD|SAVE} FORMAT PDF [AS filename]
```

where:

```
ON TABLE
```

Is required syntax if you create the PDF file within a report request.

*filename*

Assigns a name to the Hold file. The default name is HOLD. If you specify an optional AS filename the name that you supply (1-8 characters) is used in place of HOLD. If you issue a FILEDEF for the filename, the PDF output is created in the file specified in the FILEDEF.

In MVS, unless you allocate this ddname to a permanent file, FOCUS allocates it to a temporary dataset.

In CMS, this name becomes the filename. The filetype is PDF.

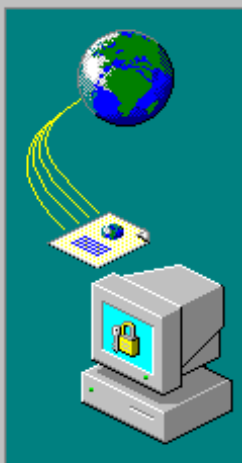
## Downloading PDF Output

The technique for downloading PDF files depends on how you accessed FOCUS. PDF-formatted Hold files created on the mainframe can be downloaded to the PC using FTP transfers. With the Web Interface, browser users can issue the following PCHOLD syntax to download PDF files to their browsers.

```
ON TABLE PCHOLD FORMAT PDF
```

The Web Interface automatically FTPs output to your PC if you choose. A browser window opens after the command is issued, inquiring whether you wish to have the file opened immediately or saved to disk.





You have chosen to download a file from this location.

webtoe916f4b18. from ibivm

What would you like to do with this file?

- Open this file from its current location
- Save this file to disk

Always ask before opening this type of file

OK

Cancel

More Info

## Example Creating a PDF-formatted Report Using the CAR File

This report request generates the PDF report that follows.

```
TABLE FILE CAR
HEADING CENTER
"CAR COSTS PER COUNTRY"
FOOTING
"---> Uncentered footing"
PRINT
MODEL DEALER_COST RETAIL_COST
BY COUNTRY
BY CAR
WHERE COUNTRY = 'ENGLAND' OR 'ITALY'
ON CAR SUBTOTAL
ON TABLE PCHOLD FORMAT PDF
END
```

Acrobat Reader - [HOLD.PDF]

File Edit View Tools Window Help

Page 1 of 1

FACE 1

COUNTRY	CAR	CAR COSTS PER COUNTRY MODEL	DEALER_COST	RETAIL_COST
ENGLAND	JAGUAR	VJ600K AUTO	7,427	8,870
		XJ61L AUTO	11,194	13,091
		*TOTAL CAR JAGUAR	18,621	22,969
	JEEP	INTERCEPTOR III	14,848	17,850
	*TOTAL CAR JEEP		14,848	17,850
	TRIUMPH	TR7	4,292	5,100
	*TOTAL CAR TRIUMPH		4,292	5,100
ITALY	ALFA ROMEO	2000 GT V616M	5,453	6,820
		2000 SPIDER V616M	5,453	6,820
		2000 4 DOOR BRIGLIA	4,919	5,926
	*TOTAL CAR ALFA ROMEO		16,235	19,560
	MAZDA	DOZA 2 DOOR	25,000	31,500
	*TOTAL CAR MAZDA		25,000	31,500
TOTAL			79,088	96,384

---> SCENTERED FOOTING

Page 1 of 1 | 0,00% | 88 x 114 | 1

## NF654: HOLD From External Sort

External sorts can be used to create HOLD files. This can lead to savings of up to twenty percent on processing time. The gains are most notable with relatively simple requests against large databases.

### Syntax      How to Create HOLD FILES with an External Sort

```
SET EXTHOLD = [OFF|ON]
```

where:

**OFF**

Disables HOLD files by an external sort. OFF is the default.

**ON**

Enables HOLD files by an external sort.

### Conditions for Using External Sort to Create a HOLD File

- The default setting of EXTSORT=ON must be in effect.
- EXTHOLD must be ON.
- Request must contain a BY field.
- Request must contain ON TABLE HOLD or ON TABLE HOLD AS.

- Your query should be simple (AUTOTABLEFable). AUTOTABLEF analyzes a query and determines whether the combination of verbs and formatting options require the internal matrix or not. In cases where it's determined that a matrix is not necessary to satisfy the query we avoid the extra internal costs associated with creating the matrix. The internal matrix is stored in a file or dataset named FOCSORT. Its default is ON so that performance gains may be realized.
- SET ALL must be OFF.
- There cannot be an IF/WHERE TOTAL or BY TOTAL in the request.
- If a request contains a SUM command, EXTAGGR must be set ON and the only column prefixes allowed are SUM and FST.
- If a request contains a PRINT command, the column prefixes allowed are SUM, AVE, MAX, MIN, FST and LST.

## NF597: Aggregation by External Sort

External sorts can be used to perform aggregation with a significant decrease in processing time in comparison to using the sort facility of FOCUS. The gains will be most notable with relatively simple requests against large databases.

### Syntax      How to Use Aggregation in Your External Sort

```
SET EXTAGGR = aggropt
```

where:

*aggropt*

can be one of the following:

OFF

disallows aggregation by an external sort.

NOFLOAT

allows aggregation if there are no floating data fields present.

ON

allows aggregation by an external sort. ON is the default.

### Conditions for Aggregating with an External Sort

- You must be using SYNCSORT or DFSORT.
- EXTAGGR cannot be set to OFF.
- Your query should be simple. (AUTOTABLEFable)
- The PRINT display command may not be used in the query.

- SET ALL must be equal to OFF.
- Only the following column prefixes are allowed: SUM, AVG, CNT, FST.
- Columns can be COMPUTED or have a ROW-TOTAL.
- CMS DFSORT does not support aggregation of numeric data types. When SET EXTAGGR = NOFLOAT and your query aggregates numeric data, the external sort is not called and aggregation is performed by the FOCUS sort.

### **Example** How Using an External Sort for Aggregation Can Change Your Output

Using an external sort for aggregation can change the output of your report request.

If you use SUM on an alphanumeric field in your report request without using an external sort, FOCUS displays the last instance of the sorted fields in the output. Turning on aggregation in the external sort results in the first record being displayed instead.

With aggregation in the external sort turned on:

```
SET EXTAGGR = ON
TABLE FILE CAR
SUM CAR BY COUNTRY
END
```

The output is:

```
COUNTRY      CAR
-----      -
ENGLAND      JAGUAR
FRANCE       PEUGEOT
ITALY         ALFA ROMEO
JAPAN         DATSUN
W GERMANY     AUDI
```

With aggregation in the external sort turned off:

```
SET EXTAGGR = OFF
TABLE FILE CAR
SUM CAR BY COUNTRY
END
```

The output is:

```
COUNTRY      CAR
-----      -
ENGLAND      TRIUMPH
FRANCE       PEUGEOT
ITALY         MASERATI
JAPAN         TOYOTA
W GERMANY     BMW
```

**Note:** The SET SUMPREFIX command in conjunction with aggregation using an external sort also affects the order of information displayed in your report. For complete information on SUMPREFIX please see New Feature Bulletin 728, SUMPREFIX.



## Reference **Special Considerations**

When aggregation is performed by an external sort the statistical variables &RECORDS and &LINES are of equal value. This is done because the external sort products do not return a line count for the answer set. This is a behavior change and affects any code that checks the value of &LINES.

## NF728: Changing Retrieval Order with Aggregation

When an external sort product performs aggregation of alphanumeric or smart date formats, the order of the answer set returned differs from the order of the FOCUS sorted answer sets.

External sort products return the first alphanumeric or smart date record that was aggregated. Conversely, FOCUS returns the last record.

The SUMPREFIX command deals with this difference in behavior by allowing users to choose which order the answer set should display.

### Syntax      Setting Retrieval Order

```
SET SUMPREFIX = {LST|FST}
```

where:

**LST**

Displays the last value in cases of data aggregation of alphanumeric or smart date data types.

**FST**

Displays the first value in cases of data aggregation of alphanumeric or smart date data types.

## NF655: FOCPROF - The System Wide Profile

FOCPROF is a new global profile for FOCUS. While the FOCPARM profile supports only FOCUS SET commands, this new profile can contain any command that is valid in a FOCEXEC, including TABLE, GRAPH, MATCH, MODIFY, MAINTAIN, REBUILD, COMPILE, LOAD, Dialogue Manager commands, CMS commands, TSO commands, and DYNAM commands.

### FOCUS Profiles

With the addition of FOCPROF, there are now three FOCUS profiles: FOCPARM, FOCPROF, and PROFILE.

The files FOCPARM and FOCPROF:

- Are members of the ERRORS PDS on MVS.
- Have filetype ERRORS on CMS.

The file PROFILE:

- Is a member of the FOCEXEC PDS on MVS.
- Has filetype FOCEXEC on CMS.

### Using FOCUS Profiles

The order of execution of FOCUS profiles is:

1. FOCPARM, which can contain FOCUS SET commands only.
2. FOCPROF, the new global profile.
3. PROFILE.

## **Procedure** How to Create a FOCPROF Profile

- For MVS, create a new member of the ERRORS PDS named FOCPROF.
- For CMS, create a file with filename FOCPROF and filetype ERRORS.

Edit the FOCPROF file to contain the commands to be executed each time FOCUS is invoked.

## NF660: Multi-volume Support in MVS FOCUS

The latest release of MVS FOCUS gives sites the option of allocating FOCUS databases, Fusion databases, and FOCUS-created sequential files across multiple volumes.

### Advantages of Multi-volume Data Sources

Many sites prefer to distribute high volume data sources across multiple volumes in order to manage:

- Use of storage on specific devices or device types.
- Run-time access to these devices.
- B37 abends.

You can now use this performance tuning technique (also known as *data striping*) with FOCUS databases, Fusion databases, and FOCUS-created sequential files in the MVS batch, TSO, and MSO environments.

### Allocating Multi-volume Data Sources

The SPACE parameter for allocating a data source can include a primary and a secondary allocation. The primary allocation is the amount of space allocated the first time data is written to the data set. The secondary allocation is the amount of space to be allocated, when necessary, for up to 15 additional extents.

For a single-volume data source, processing terminates with a B37 abend when the system detects either of the following conditions:

- A need for more than 16 extents.
- A need for a new secondary extent (below the 16-extent limit) when enough space is not available on the volume.

FOCUS returns the following message to indicate that one of these conditions has occurred:

`(FOC198) FATAL ERROR IN DATABASE I/O. FOCUS TERMINATING CODE: 00000070`

You can prevent this type of abnormal termination by allocating multiple volumes to the data source. With multiple volumes, an out of space condition on the first volume causes allocation to start on another volume.

With multiple volumes, the allocation process varies slightly for each of the following:

- The first volume.
- Intermediate volumes.
- The last volume.

The following table describes the multi-volume allocation process:

Primary Allocation	The primary allocation is applied to the first volume only. It can consist of the number of extents allowed by MVS for a primary allocation. <b>Note:</b> A data source with no secondary space allocation is limited to a single volume.
--------------------	--

Secondary Allocation	<p><b>1.First volume:</b></p> <p>As many extents as are available, up to the 16-extent limit, are allocated and filled before continuing to the second volume.</p> <p><b>2.Intermediate volume:</b></p> <p>Depending on the space available, up to 16 extents are filled before allocation begins on the next volume.</p> <p><b>3.Last volume:</b></p> <p>Once the need for a number of extents greater than the limit is detected:</p> <ul style="list-style-type: none"><li>• For a FOCUS or Fusion database, processing terminates with the following message <code>(FOC198) FATAL ERROR IN DATABASE I/O. FOCUS TERMINATING CODE 00000070</code></li><li>• For temporary FOCSORT files, after all volumes and extents are filled allocation spills to up to 15 additional temporary files, each with 16 extents. (This feature, NF536, <i>Multi-Image FOCSORT</i>, was added in FOCUS 7.0.6.) The SPACE allocation for each spill file is the same as the SPACE allocation for the original FOCSORT file.</li></ul>
----------------------	--

For example:

```
//FOCSORT DD SPACE=(TRK,(5,5))
```

This allocates a total of  $5 + (5 \times 15) = 80$  tracks. When the 81st track is needed, another temporary data set is allocated with the parameter `SPACE=(TRK,(5,5))`. If necessary, this additional step is repeated a total of 15 times yielding a total of  $80 \times 16$  tracks for FOCSORT.

If enough space is not available after filling all of the extents of all of the spill files, the FOC198 message is issued and processing terminates.

**Note:** The number of extents actually obtained on any volume may be less than 16; however, in most situations 16 will be available and used.

## Syntax      How to Allocate a Multi-volume Data Source in the MVS Batch Environment

You have two choices for statically allocating a new multi-volume FOCUS database, Fusion database, or sequential file.

You can list multiple VOLSER identifiers on the DD card for the multi-volume data source:

```
//ddname DD DSN=dsname,VOL=SER=(vol1,...,voln),...
```

Alternatively, you can ask for multiple units of a specific type:

```
//ddname DD DSN=dsname,UNIT=(type,n),...
```



where:

*ddname*

Is the DDNAME associated with the multi-volume data source.

*dsname*

Is the data set name of the multi-volume data source.

*vol1,...,voln*

Are the volume identifiers for the each of the volumes to use.

*type*

Is the type of unit to use.

*n*

Is the number of units.

## Allocating a Multi-volume Data Source in the TSO and MSO Environments

In both TSO and MSO you have two choices for dynamically allocating a multi-volume data source:

- You can list multiple volume identifiers.
- You can specify the number of units to use and let the system choose the specific volumes. All of the units will be the same type (for example, 3390).

### **Syntax**      **How to Allocate Specific Volumes in the TSO and MSO Environments**

To allocate specific volumes for a multi-volume data source, use the following syntax:

- In TSO

```
TSO ALLOC ... VOLUME('vol1,...,voln')...
```

- In MVS FOCUS or MSO

```
DYNAM ALLOC ... VOL vol1,...,voln ...
```

where:

```
vol1,...,voln
```

Are the volume identifiers for the each of the volumes to use.

## Syntax      How to Specify the Number of Units in the TSO and MSO Environments

To specify the number of volumes for a multi-volume data source and let the system choose the specific volumes, use the following syntax:

- In TSO

```
TSO ALLOC ... UCOUNT('n') UNIT('type') ...
```

- In MVS FOCUS or MSO

```
DYNAM ALLOC ... UCOUNT n UNIT type ...
```

where:

```
n
```

Is the number of volumes to use.

```
type
```

Is the type of unit to use.

**Note:**

- UNIT VIO is not supported.
- The RLSE option of the SPACE parameter is not supported.
- The DYNAM UCOUNT parameter is also discussed in NF670, *DYNAM Support for Unit Count*.

**Example**    **Allocating a Data Source to Two Volumes**

The following DYNAM command allocates two volumes to a data source called MULTVOL:

```
DYNAM ALLOC FI MULTVOL DS USER1.FOCTST.MULTVOL TRACK SPACE 4 4 REU -  
                UCOUNT 2 UNIT SYSDA CATALOG
```

With this allocation, a second volume will be used when the 17th extent is needed.

**Syntax**    **How to Display the Volume Identifiers Allocated to a Multi-volume Data Source**

To see the data set information associated with a specific DDNAME, issue the following command

```
? TSO DDNAME ddname
```

where:

*ddname*

Is the DDNAME allocated to the data set whose volume identifiers you want to see.

## Example Displaying Multi-volume Data Set Information

The following example shows how to display data set information for DDNAME MULTVOL:

```
? tso ddname multvol
```

The following information is returned. Notice that two volume serial identifiers are listed on the VOLSER line:

```
DDNAME      =  MULTVOL
DSNAME      =  USER1.FOCTST.MULTVOL
DISP        =  NEW
DEVICE      =  DISK
VOLSER      =  MFOC02,MFOC01
DSORG       =  PS
RECFM       =  F
SECONDARY   =      4
ALLOCATION   =  TRACKS
BLKSIZE     =      4096
LRECL       =      4096
TRKTOT     =      92
EXTENTSUSED =      23
BLKSPERTRK =      12
TRKSPERCYL =      15
CYLSPERDISK =     2227
BLKSWRITTEN =     1104
FOCUSPAGES =     1059
> >
```

## Choosing Default Sizes for FOCUS-created Files

IBITABLA, the file that contains default allocations for FOCUS-created files, has been enhanced to allow a unit count for FOCUS databases, Fusion databases, and all sequential FOCUS-created files. The advantages of multi-volume allocations are described in [Advantages of Multi-volume Data Sources](#).

Every FOCUS release is shipped with a member of the FOCCTL.DATA PDS named IBITABLA. This member contains default allocations for all FOCUS-created files that you do not specifically allocate in your FOCUS session, JCL, or CLIST. As part of the installation process, the FOCUS installer should copy the new version of IBITABLA to the ERRORS PDS and edit it to conform to the standards for the site.

When a new release of FOCUS is shipped, the installer should compare the latest version of IBITABLA to the prior site-specific version in order to construct a new site-specific version of IBITABLA. In this way, the installer will be aware of changes in format (such as new fields or ddnames added in the new release) that must be addressed in the customized copy.

The following is the default IBITABLA shipped with FOCUS Version 7.0 Release 9:

```
*...+...1...+...2...+...3...+...4...+
HOLD      CYLS    5  10                                3,
HOLDMAST  TRKS    5  5 36                                ,
SAVE      CYLS    5  10                                3,
REBUILD   CYLS    5  10                                3,
FOCSML    CYLS    5  5                                  2,
FOCUS     CYLS    5  5                                  1,
FOCSTACK  TRKS    5  5                                  2,
FOCSORT   CYLS    5  5                                  1,
OFFLINE   CYLS                                     A      ,
SESSION   TRKS    5  5                                  2,
FOCCOMP   TRKS    5  5 12                                ,
HOLDACC   TRKS    5  5 12                                ,
FMU       TRKS    5  5 12                                ,
TRF       TRKS    5  5 12                                ,
FOCPOOLT  CYLS    5  20                                NOHIPER 4,
FUSION    CYLS    5  50                                NOHIPER 4,
MDI       CYLS    5  20                                NOHIPER 4,
FOC$HOLD  CYLS    5  5                                  2,
EXTINDEX  CYLS    5  5                                  2,
```

The unit count field is columns 44-45. The fields are:

<b>Columns</b>	<b>Length</b>	<b>Comments</b>
01-08	8	Class of file - DDname
10-13	4	Allocation units (CYLS, TRKS)
15-17	3	Primary allocation
19-21	3	Secondary allocation
23-24	2	Number of directory blocks (blank specifies a sequential file; 0 is invalid)
26-26	1	SYSOUT class
28-33	6	Volume serial on which to allocate
35-42	8	Type of unit to allocate (for example, 3390, DASD, NOHIPER*)
44-45	2	Unit count

Note that partitioned data sets do not support multi-volume allocations.

If you enter a non-valid value for any field in IBITABLA, FOCUS substitutes the corresponding value from the original version of IBITABLA that was shipped with FOCUS.

**Example** Allocating DDNAME FOCSTACK to Two Volumes in IBITABLA

Assume you replaced the original FOCSTACK line with the following in IBITABLA:

```
|...+...1...+...2...+...3...+...4...+
FOCSTACK TRKS 5 5 3390 2,
```

This line indicates the following allocation attributes:

- DDname = FOCSTACK
- Allocation is in tracks:  
Primary tracks=5  
Secondary tracks=5
- Type of unit = 3390
- Number of units=2

The resulting dynamic allocation is equivalent to the following JCL:

```
//FOCSTACK DD SPACE=(TRK,(5,5)),UNIT=(3390,2),...
```

**Reference** Usage Notes for IBITABLA

- IBITABLA is a fixed-format file.
- Each data field must be placed in specific columns, but leading or trailing blanks are allowed. (There is at least one blank between successive data fields.)
- All lines beginning with an asterisk (\*) are comments.
- Unit count is ignored for HiperFOCUS files with DISP=(NEW,DELETE).



## NF584: Dynamically Setting the IDMS DBNAME and DICTNAME

The new IDMS Interface commands SET DBNAME and SET DICTNAME enable you to dynamically change the DBNAME and DICTNAME parameters at any time during your FOCUS session.

If you do not issue these commands, the Interface reads the DBNAME and DICTNAME from the Access File. However, once you issue them, the new DBNAME and DICTNAME values take precedence over those in your Access Files. The new values remain in effect until you either:

- Reissue the SET commands with new DBNAME and DICTNAME values.
- End your FOCUS session.
- Reinstate the Access File parameters by issuing the SET commands with the DEFAULT option.

### Syntax      How to Set the DBNAME and DICTNAME

Issue the following commands at the FOCUS session prompt, in a FOCEXEC, or in any supported profile:

```
TSO IDMSR SET DBNAME {dbname|DEFAULT}
```

```
TSO IDMSR SET DICTNAME {dictname|DEFAULT}
```

where:

*dbname*

Is the IDMS database name that you want to access.

*dictname*

Is the IDMS dictionary name that you want to access.

DEFAULT

Causes the Interface to read the value from the Access File.

### Syntax     How to Display the Current Settings

To display the settings that are currently in effect, issue the following command:

```
TSO IDMSR SET ?
```

### Example     Dynamically Changing the DBNAME and DICTNAME Values

```
TSO IDMSR SET DBNAME EMPDEMO
```

```
TSO IDMSR SET DICTNAME APPLDICT
```

## NF673: Model 204 Interface Account Split

The Model 204 Interface has been enhanced to accept an account code as part of the logon string.

The syntax for specifying the logon account in the Access File is

```
ACCOUNT=x[ y], ACCOUNTPASS=pswd, IFAMCHNL=ifchnl, $
```

The SET command syntax for specifying the logon account is

```
{TSO|MVS} M204IN SET ACCTNAME x[ y]  
{TSO|MVS} M204IN SET ACCTPASS pswd
```

where:

*TSO|MVS*

Is a required environmental prefix. TSO and MVS are synonyms and can be used interchangeably.

*x*

Is the user ID, up to 16 characters.

*y*

Is the account code, up to 15 characters, preceded by exactly one blank.

*pswd*

Is the account password.

### **Example** Setting the Model 204 Userid and Account Code

Assume the Model 204 userid is SUPERKLUGE and the account code is MIS. You can issue the following Interface command to set these values:

```
TSO M204IN SET ACCTNAME SUPERKLUGE MIS
```

You can also set them in the Access File with the following attribute:

ACCOUNT=SUPERKLUGE MIS, ...

## NF720: SQLJOIN OUTER Setting for Relational Interfaces

With the new SQLJOIN OUTER setting you can control when the relational Interfaces optimize outer joins without affecting the optimization of other operations. This parameter provides backward compatibility with prior releases of the relational Interfaces and enables you to fine-tune your applications.

When join optimization is in effect, the Interface generates one SQL SELECT statement that includes every table involved in the join. The RDBMS can then process the join. When join optimization is disabled, the Interface generates a separate SQL SELECT statement for each table, and FOCUS processes the join.

In FOCUS Version 7.0 Release 6, the relational Interfaces were enhanced to optimize outer joins. The following command causes the Interface to generate an outer join:

```
SET ALL = ON
```

Beginning with FOCUS Version 7.0 Release 6, issuing this command with OPTIMIZATION enabled invokes outer join optimization. In FOCUS releases prior to 7.0.6, this setting disabled optimization so that FOCUS always processed the outer join. Turning OPTIMIZATION OFF still causes FOCUS to process the join, but it disables *all* optimization enhancements, not just outer join processing.

Starting with FOCUS Version 7.0 Release 9, you can use the SQLJOIN OUTER setting to disable outer join optimization while leaving other optimization enhancements in effect.

## Syntax How to Control Outer Join Optimization

Issue the following command at the FOCUS prompt, in a stored procedure, or in any supported profile:

```
SQL target_db SET SQLJOIN OUTER {ON|OFF}
```

where:

*target\_db*

Indicates the target RDBMS. Valid values are DB2, SQLDS, SQLDBC, SQLORA, or SQLIDMS. Omit if you issued the SET SQLENGINE command.

ON

Enables outer join optimization.

OFF

Disables outer join optimization. OFF is the default value.

### Note:

- The SQLJOIN OUTER setting is available only when optimization is enabled (that is, OPTIMIZATION is not set to OFF).
- The SQLJOIN OUTER setting is ignored when SET ALL = OFF.

## Effects of Combinations of Settings on Outer Join Optimization

The following table describes how different combinations of OPTIMIZATION and SQLJOIN OUTER settings affect Interface behavior. It assumes that SET ALL = ON:

Settings		Results	
OPTIMIZATION	SQLJOIN OUTER	Outer Join Optimized?	Other Optimization Features
ON	ON	Yes	Enabled
ON	OFF	No	Enabled
OFF	N/A	No	Disabled
SQL	ON	Yes, in all possible cases	Enabled
SQL	OFF	No	Enabled
FOCUS	ON	Yes if results are equivalent to FOCUS managed request	Enabled
FOCUS	OFF	No	Enabled

## Reference SQLJOIN OUTER Messages

If SQLJOIN OUTER is set to OFF, the following message displays when you issue the SQL ? query command:

```
(FOC1420) OPTIMIZATION OF ALL=ON AS LEFT JOIN - : OFF
```



## NF652: Teradata Interface Kanji Support

The Teradata Interface now supports the DBCS (Double Byte Character Set) Kanji characters described by the Teradata datatypes GRAPHIC and VARGRAPHIC.

The following conversion chart shows the SQL datatypes that support the Kanji character set and their corresponding ACTUAL formats in the Master File:

SQL Datatype	ACTUAL Format	Description
GRAPHIC	Kn	DBCS Kanji character set. Fixed-length string of 'n' 16-bit characters where $0 < n \leq 127$ . The appropriate corresponding USAGE format is A(2n+2).
VARGRAPHIC	Kn	DBCS Kanji character set. Varying-length string of 'n' 16-bit characters where $0 < n \leq 127$ . The appropriate corresponding USAGE format is A(2n+2). <b>Note:</b> LONG VARGRAPHIC (or VARGRAPHIC (n) where $n > 127$ ) is not supported.

## NF722: FOCUS Client DNS Names Support

This feature enables the FOCUS Client EDACFG file to identify an EDA server by host name rather than IP address. The EDACFG file is the client communications configuration file allocated to DDNAME EDACFG. In prior versions of FOCUS Client, the EDACFG file had to supply the IP address of the EDA server to which it would connect.

The domain name system (DNS) is a global network of servers that translate host names, such as `www.ibi.com`, into IP addresses. For more information about FOCUS Client, see NF494, *FOCUS Client - Remote Data Access via EDA/SQL*, which was implemented in FOCUS 7.0.5.

### Syntax      How to Invoke DNS Names Support

The syntax for specifying a host name in the client configuration file for TCP/IP is

```
HOST = hostname
```

where:

```
hostname
```

Is the host name of the EDA server.

## **Example** Using DNS Names Support

The following client configuration file is used for connecting to the host named IBIMVS:

```
NAME = EDA/SQL CLIENT USING CS/3 TCP/IP
NODE = TCPOUT
  BEGIN
; TRACE = 31
  PROTOCOL = TCP
  CLASS = CLIENT
  HOST      = IBIMVS           ; DNS NAME OF HOST
  SERVICE   = 2459            ; PORT NUMBER OF THE EDA SERVER
  END
```

## NF656: Controlling REBUILD Messages

This feature allows for direct control over the frequency with which REBUILD issues messages. FOCUS, by default, displays a message for every 1000 records read during the database retrieval and load phases of the REBUILD utility. The message, REFERENCE..AT SEGMENT 1000, REFERENCE..AT SEGMENT 2000.. is a function of the number of records in the FOCUS file being rebuilt. The frequency of these messages can become problematic for larger FOCUS files because CMS spool space may be limited.

### Syntax     How to Control REBUILD Messages

The user can set how often the message is displayed by issuing the command:

```
SET REBUILDMSG = n
```

where:

*n*

is any 8-byte integer.

A setting of less than 1000 generates a diagnostic and keeps the current setting. The current setting will either be the default of 1000, or the last valid integer greater than 999 to which REBUILDMSG was set. A setting of 0 disables the 'REFERENCE..AT SEGMENT' messages.

## Example Controlling Display of REBUILD Messages

The following example shows a REBUILD CHECK function where REBUILDMSG has been set to 4000, and the database contains 19,753 records.

```
ENTER NAME OF FOCUS/FUSION FILE (FN FT FM)
...
STARTING..
REFERENCE..AT SEGMENT      4000
REFERENCE..AT SEGMENT      8000
REFERENCE..AT SEGMENT     12000
REFERENCE..AT SEGMENT     16000
NUMBER OF SEGMENTS RETRIEVED= 19753
CHECK COMPLETED...
```

## NF670: DYNAM Support for Unit Count

The DYNAM command now supports the unit count parameter for allocating multi-volume data sources. With the UCOUNT parameter you can allocate a file to multiple volumes of a particular type of storage device without indicating specific volume serial numbers.

### Advantages of Multi-volume Data Sources

Many sites prefer to distribute high volume data sources across multiple volumes in order to manage:

- Use of storage on specific devices or device types.
- Run-time access to these devices.

This new DYNAM parameter is particularly useful for allocating large temporary files such as FOCSORT, FOCPOOLT, and HOLD files in the TSO and MSO environments.

### Syntax      How to Specify the DYNAM Unit Count Parameter

```
DYNAM ALLOC ... UCOUNT n UNIT type ...
```

where:

*n*

Is the number of volumes to use.

*type*

Is the type of unit to use. **Note:** UNIT VIO is not supported.

**Example** Allocating a HOLD File to Two Volumes

The following DYNAM command allocates two 3390s to a HOLD file:

```
DYNAM ALLOC FILE HOLD SPACE 20,10 TRACKS UCOUNT 2 UNIT 3390
```

## NF684: PCHOLD for Non-Html Files

Web Interface support of PCHOLD enables Web browser users to extract five types of preformatted data from the mainframe and either display the output immediately on their browsers or automatically transfer the files via FTP to their PCs.

### **Using PCHOLD for Formats LOTUS, DIF, EXCEL, or PDF**

When you issue the PCHOLD command for files with LOTUS, DIF, EXCEL, or PDF formats, the Web Interface returns a notification window telling you that the file is being downloaded. You can then rename the file on the PC using the correct extension from the table below. Word processing files (Format=WP) are always displayed first on the browser and can then be saved to the PC.



## Syntax Specifying PCHOLD in a TABLE Request

ON TABLE PCHOLD FORMAT *fmt*

where:

*fmt*

Specifies the format of the PCHOLD extract file. When downloading these files to the PC you must use the appropriate extensions from the following table:

File Format	Content	Extension
LOTUS	Lotus PRN files	.prn
DIF	Spreadsheets without headings	.dif
EXCEL	Excel spreadsheets with headings	.xls
PDF	ADOBE Portable Document Format (PDF) files	.pdf
WP	Word processing files	.txt or .doc

## Syntax Specifying PCHOLD for a WP File in a TABLE Request

ON TABLE PCHOLD FORMAT WP

WP-formatted files are displayed immediately on the browser and can be saved to the PC. When the file appears, select the Save As option, providing a file name with an appropriate file extension for your word processor (for example, extension for MS Word = .doc).

```
SAVE AS myfile.doc
```

## Example Using PCHOLD for Non-Html Files

The request below creates a spreadsheet without headings (format=DIF) displayed on the screen that follows:

```
table file car
print compute salesl/i9 = sales;
profit/d10.2 = retail_cost - dealer_cost;
adate/yynd = '1999/04/29';
by model
ON TABLE PCHOLD FORMAT DIF
end
```

Microsoft Excel - testdif

File Edit View Insert Format Tools Data Window Help

Arial 10 B I U [Text Alignment Icons] \$ % , +.00 +.00 [Grid Icons] 100%

H8 Font =

	A	B	C	D	E	F	G
1	B210 2 DOOR AUTO	43000	513	19990429			
2	COROLLA 4 DOOR DIX AUTO	35030	453	19990429			
3	DORA 2 DOOR	0	6500	19990429			
4	INTERCEPTOR III	0	2910	19990429			
5	TR7	0	808	19990429			
6	V12XKE AUTO	0	1451	19990429			
7	XJ12L AUTO	12000	2297	19990429			
8	100 LS 2 DOOR AUTO	7800	907	19990429			
9	2000 GT VELOCE	12400	1160	19990429			
10	2000 SPIDER VELOCE	13000	1160	19990429			
11	2000 4 DOOR BERLINA	4800	1010	19990429			
12	2002 2 DOOR	8950	140	19990429			
13	2002 2 DOOR AUTO	8900	355	19990429			
14	3.0 SI 4 DOOR	14000	3752	19990429			
15	3.0 SI 4 DOOR AUTO	18940	3123	19990429			
16	504 4 DOOR	0	979	19990429			
17	530I 4 DOOR	14000	797	19990429			
18	530I 4 DOOR AUTO	15600	1095	19990429			

## Reference Notes for Internet Explorer Users

- LOTUS or DIF.** When using Internet Explorer to view consecutive LOTUS or DIF reports online you must take special precautions to insure getting latest version of temporary files.

This step must be taken to avoid a potential problem caused by the use of the same filename when running consecutive reports for immediate viewing (as opposed to saving the output in uniquely named files). Spreadsheet users planning to view multiple consecutive reports online must configure the Internet Explorer browser in the following manner:

- **Internet Explorer Release 5** - Under Tools/Internet Options/Temporary Internet Files, select Check for newer versions automatically.
- **Internet Explorer Release 4** - Under View options/General/Temporary Internet Files, select Check every time.

This step is not required with Netscape Navigator, which checks temporary Internet files for newer versions every time as the default. Users of other browsers must check their browser's handling of this situation and act accordingly.

- **Excel.** When using Internet Explorer to view Excel output you must use the binary MIME type for the file transfer (all other file types are transferred as text).

## NF683: Web Interface support for Maintain Winforms

This feature enables mainframe FOCUS sites to deploy applications created with the Cactus Workbench to intranet or Internet audiences without installing an EDA or Cactus server on the host machine.

Existing applications can be “webified” by using the Cactus Workbench to convert Maintain Winforms into Webforms. The Focexecs (.fex files) and Winforms (.wfm files) must then be transmitted **manually** to the mainframe using FTP (alphanumeric mode):

- For MVS, transfer the FOCEXEC files to a data set allocated to ddname FOCEXEC and the Winform files to a data set allocated to ddname WINFORMS in your FOCUS CLIST.
- For CMS, transfer the files to your A disk. The FOCEXEC files have filetype FOCEXEC and the Winform files have filetype WINFORMS.

Once these are installed, end users running the Web Interface from a Web browser can execute these applications and access data defined to mainframe FOCUS.

### Prerequisites

Web390 Server Release 3.2 is required to use this feature and end-users will need to install at least Release 4 of Netscape Navigator, Microsoft Internet Explorer, or an equivalently featured browser.

## Procedure How to Execute Maintain Applications From a Browser

There are two ways to execute Maintain applications from the browser window:

1. Use Logon scripts (*Web390 Developer's Guide and Installation Manual* DN1001035.0599 provides instructions for creating Logon scripts).

Logon scripts can bring browser users directly into the Webform equivalent of a Maintain Winform after they enter the appropriate userid and password. Thereafter, they can work interactively with the applications on their browsers.

2. Users can execute applications directly from the Web Interface Interactive screen

*EX FOCEXECname*

where:

*FOCEXECname*

Is the name of the target Maintain application.

## Example Running a Maintain Application From a Browser

This webified Maintain application issues a simple car-by-country report request, formatting the output on the Webform below:

EX CMNTX05W



The screenshot shows a web browser window with the address bar containing `http://bmvw/it_cp/ibweb/...`. The main content area has the title "CAR AND COUNTRY REPORT" in blue. Below the title, there is a list of three steps:

1. Passing a Stack with two Fields
2. Local CALL
3. CALL SMNTX05 From Carstk Into Carstk

To the right of these steps are two buttons: "Run" and "Exit". Below the list, there is a block of text: "passing a stack ( two fields inlined) AND RETURNING THE STACK AFTER PROCEDURE HAS RETRIEVED ENTIRE STACK".

At the bottom of the page is a table with two columns: "COUNTRY" and "CAR".

COUNTRY	CAR
ENGLAND	JAGUAR
ENGLAND	JENSEN
ENGLAND	TRUMPH
FRANCE	PEUGEOT
ITALY	ALFA ROMEO
ITALY	MASERATI
JAPAN	DATSUM
JAPAN	TOYOTA
W GERMANY	AUDI
W GERMANY	BMW

## NF691: Escape Character for LIKE Predicate

New syntax exists for the LIKE predicate that enables you to search for special characters in data. You can set an escape character to use in the LIKE predicate. Then, when you include this escape character in front of a special character in the mask pattern, the parser treats the special character as a normal character and searches for it in the data. This next character can be '%' or '\_' which are normally special characters. LIKE may be coded in WHERE, DEFINE, and COMPUTE statements. The mask is an alphanumeric, user-supplied pattern that FOCUS uses to compare characters in a data field value. A mask has two special characters:

1. The percent sign (%) to indicate any number of characters; and
2. The underscore (\_) to indicate a single character in a specified position.

The escape character enhancement permits FOCUS to treat these masking characters as literals within the search pattern and not as wildcards.

Any single character can be used as an escape character, but the one used must be prefaced with the word ESCAPE. The syntax is:

```
WHERE field LIKE 'phrase' ESCAPE 'c'
```

where

```
'c'
```

Is any character embedded in the phrase before a '%' or '\_'.



## Escape Character Capabilities

The escape character is supported by the SQL Interfaces, as well as FOCUS databases. The escape character is only in effect when the ESCAPE syntax is included in the LIKE predicate.

### Syntax      How to Use the Escape Character

The pattern (alphanumeric constant) that follows the LIKE predicate may contain wildcard characters ('%') and ('\_') that users may need to escape in order to use as part of the search pattern. Every LIKE predicate can provide its own escape character to be used within the pattern (or mask). The syntax is as follows:

```
WHERE expression LIKE 'abc\%' ESCAPE '\'
```

where

*expression*

Is the name of the expression to be evaluated in the selection test.

'abc\%'

Is the alphanumeric test value.

'\'

Is the escape character enclosed in single quotation marks. When the escape character is used in the pattern immediately preceding the special character '%' or '\_', FOCUS is instructed to treat the special character as a literal and not as a wildcard. The character itself can also be escaped, thus becoming a normal character in a string (for example, 'abc\%\%').

## Example Using the Escape Character

Using the Car file, assume that both Peugeot and Alfa Romeo produce 4\_door car models. To generate a report showing countries that produce 4\_door car models, the syntax would be as follows:

```
TABLE FILE CAR
PRINT CAR MODEL
BY COUNTRY
WHERE MODEL LIKE '%g_DOOR%' ESCAPE 'g'
END
```

The request produces the following report:

```
PAGE      1
```

```
COUNTRY      CAR          MODEL
-----      ---          -
FRANCE       PEUGEOT      504 4_DOOR
ITALY        ALFA ROMEO   2000 4_DOOR BERLINA
```

## Reference Special Considerations

- The use of an escape character in front of any character other than '%', '\_', and itself will be ignored.
- Only one escape character can be used per LIKE phrase.

- If a WHERE clause is used with lazy ORs, the ESCAPE must be on the first phrase and will apply to all subsequent phrases in that WHERE clause. For example:

```
WHERE field LIKE 'ABCg_' ESCAPE 'g' OR 'ABCg%' OR 'g%ABC'
```

## Reference Error Messages

(FOC36251) SYNTAX ERROR IN LIKE OPERATOR

Alphanumeric mask must follow the LIKE operator. An optional keyword ESCAPE followed by a single character alphanumeric constant in apostrophes can be used after the mask.

## NF718: DYNAM Support for Existing Relative GDG Numbers

The DYNAM command now supports allocation of existing iterations of a Generation Data Group (GDG) using relative index numbers. Existing iterations are those with index numbers less than or equal to zero.

**Note:** DYNAM does *not* support allocation of a new GDG iteration; that is, you cannot use the relative number +1 in a DYNAM allocation for a GDG.

### Using DYNAM With Relative GDG Numbers

The original definition of a GDG assigns it a group name and specifies how many generations will be maintained. Once the maximum number of generations has been reached, each new iteration replaces the oldest existing iteration. The data set name for each iteration is the group name appended with a qualifier that contains the iteration number.

For example, the following GDG, named USER1.HOLD.GDG, has three generations:

Data Set Name	Iteration Number	Index Number
USER1.HOLD.GDG.G0003V00	3 (Oldest)	-2
USER1.HOLD.GDG.G0004V00	4	-1
USER1.HOLD.GDG.G0005V00	5 (Current)	0

The current (that is, newest) iteration always has the highest iteration number and corresponds to index number zero. The next newest corresponds to index number -1, and so on.

Prior to this new feature, DYNAM could allocate only the current iteration of a GDG. Now DYNAM supports relative index numbers for allocating any existing iteration.

### Syntax      How to Use DYNAM With Relative Index Numbers

Use the following syntax to allocate an existing iteration of a GDG

```
DYNAM ALLOC FILE ddname DS 'group_name(index)' SHR REUSE
```

where:

*ddname*

Is the name of the Master File for the GDG.

*group\_name*

Is the group name of the GDG.

*index*

Is the index number for an existing iteration of the GDG. It must be less than or equal to zero.

**Example** Allocating Existing GDG Iterations Using DYNAM With Relative Index Numbers

The GDG named USER1.HOLD.GDG discussed in [Using DYNAM With Relative GDG Numbers](#), is described by Master File GDG1:

```
FILENAME=GDG1 , SUFFIX=FIX
SEGNAME=ORIGIN , SEGTYPE=S1
  FIELDNAME=FLD1 , , A10 , $
  FIELDNAME=FLD2 , , A20 , $
  FIELDNAME=FLD3 , , A30 , $
  FIELDNAME=FLD4 , , A10 , $
```

In each iteration, the value stored in FLD1 is the generation number. For example, USER1.HOLD.GDG.G0004V00 contains the following records:

```
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4444444444444AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

The following request allocates and prints FLD1 of iteration four (index number -1)

```
> dynam alloc file gdg1 ds 'user1.hold.gdg(-1)' shr reu
> > table file gdg1
> print fld1
> end
```

PAGE 1

FLD1

----

4444444444

4444444444

4444444444

4444444444

4444444444

The following DYNAM command allocates the current iteration of the GDG, and the TABLE request accesses this latest iteration:

```
> dynam free file gdg1
> > dynam alloc file gdg1 ds 'user1.hold.gdg(0)' shr reu
> > table file gdg1
> print fld1
> where recordlimit eq 1
> end
```

PAGE 1

FLD1

----

5555555555

## Syntax Determining Which GDG Iteration is Allocated

To determine which GDG iteration is allocated, issue the ? TSO DDNAME query command:

```
> > ? tso ddname gdg1

DDNAME      = GDG1
DSNAME      = USER1.HOLD.GDG.G0003V00
DISP        = SHR
DEVICE      = DISK
VOLSER      = USERME
DSORG       = PS
RECFM       = FB
SECONDARY   = 5
ALLOCATION   = TRACKS
BLKSIZE     = 8000
LRECL       = 80
TRKTOT      = 1
EXTENTSUSED = 1
BLKSPERTRK = 6
TRKSPERCYL = 15
CYLSPERDISK = 3340
BLKSWRITTEN = 1
> >
```

The DSNAME, USER1.HOLD.GDG.G0003V00, indicates that the third iteration is allocated.



## Reference Error Messages

The MESSAGE parameter must be set ON in order to receive these messages.

An attempt to allocate a new iteration of a GDG (index number +1) generates the following error message:

```
(FOC880)      DYNAMIC ALLOCATION ERROR: IKJ56871I DATA SET name
              NOT ALLOCATED, RELATIVE GENERATION NUMBER INCOMPATIBLE FOR
              SPECIFIED STATUS, #099:0C-0394
```

Allocating an index number that is out of range (too large a negative number), generates the following message:

```
(FOC855)      UNABLE TO LOCATE DATASET name, #099:04-1708
```

## NF735: Enhancement to ? SET

Two options have been added to the ? SET command. They are ? SET FOR and ? SET NOT. The FOR option lists the current state of the command queried, and details where it may be set within FOCUS. The NOT option produces a list of SET commands not settable in five specific areas.

### Syntax Querying a Command

```
? SET FOR parameter
```

where:

```
parameter
```

Is any SET parameter.

### Example Querying the Current State of EXTSORT

Entering

```
? SET FOR EXTSORT
```

yields

EXTSORT ON

```
-----  
SETTABLE FROM COMMAND LINE           : YES  
SETTABLE ON TABLE                    : YES  
SETTABLE FROM SYSTEM-WIDE PROFILE     : YES  
SETTABLE FROM HLI PROFILE             : YES  
POOL TABLE BOUNDARY                 : YES
```

&gt;

The preceding screen shows that EXTSORT is currently set ON and that it is settable from all five features.

## Syntax Determining Which Commands Are Not Settable In Each of the Five Features

```
? SET NOT functional_area
```

where:

```
functional_area
```

can be one of the following five areas:

- PROMPT (in a PROMPT command)
- ONTABLE (in a report request)
- FOCPARM (in the FOCPARM profile)
- HLIPROF (in the HLI profile)
- PT (Pooled Tables)

## Example Determining Which Commands Are Not Settable In a Report Request

Entering

```
? SET NOT ONTABLE
```

yields:

### NON-SETTABLE ON TABLE PARAMETER SETTINGS

BINS	64	LANGUAGE	AMENGLISH	REBUILDMSG	1000
BLKCALC	NEW	MAXPOOLMEM	32768	SAVEMATRIX	ON
BYPANELING	OFF	MDIBINS	8000	TCPIPINT	OFF
CACHE	0	MDIPROGRESS	100000	TEMP DISK	A
COLUMNSCROLL	OFF	MODE	CMS	TRMSD	24
DATEDISPLAY	OFF	MPRINT	NEW	TRMSW	80
DATEFNS	ON	POOL	OFF	TRMTYP	1 (3270)
DEFCENT	19	POOLBATCH	OFF	WEBHOME	OFF
EUROFILE		POOLFEATURE	OFF	WIDTH	130
FIELDNAME	NEW	POOLMEMORY	16384	WINPFKEY	OLD
FOCSTACK SIZE	8	POOLRESERVE	1024	YRTHRESH	0
HTMLMODE	OFF	PRINTPLUS	OFF		

>

The preceding screen shows a list of parameters that are not settable using ON TABLE.

## NF740: Changes to the REBUILD Prompt

With the addition of two new options, MIGRATE and MDINDEX, the REBUILD prompt no longer fits on one line. After you enter REBUILD at the FOCUS prompt a numbered list is displayed that allows you to enter either the option name or its corresponding number.

### Syntax      The REBUILD Prompt Screen

```
>REBUILD
```

```
Enter option
```

1. REBUILD                    (Optimize the database structure)
2. REORG                     (Alter the database structure)
3. INDEX                      (Build/modify the database index)
4. EXTERNAL INDEX            (Build/modify an external index database)
5. CHECK                     (Check the database structure)
6. TIMESTAMP                (Change the database timestamp)
7. DATE NEW                 (Convert old date formats to smartdate formats)
8. MDINDEX                   (Build/modify a multidimensional index. FUSION DBs only)
9. MIGRATE                   (Convert FOCUS masters/DBs to FUSION)

**Note:** A change was also made to the FILENAME prompt. Instead of prompting for the name of the FOCUS file, you are now asked for the FOCUS/ FUSION filename. Within the individual options a similar numbered list is also displayed if applicable. REBUILD commands are stackable in name or number format.

## NF745: ? PTF Enhancements

Two columns have been added to the output displayed in response to ? PTF. They are, SUPERSEDED BY, which displays a superseding PTF if one exists and PUT LEVEL, which demonstrates the order in which multiple PTFs must be applied..

### Syntax How to Identify PTFs Applied to Your Version of FOCUS

Issue the following command at the FOCUS prompt:

```
? PTF
```

A screen similar to the following is displayed:

```
>
? ptf
PTFS APPLIED TO RELEASE 70XFOC
FROM PTFTABLE LOCATED IN IBITEST LOADLIB C1

COUNT PTF NUM      CREATED      APPLIED      SUPERSEDED BY  PUT LEVEL
-----
1) 95828             .....      .....      112600         .....
2) 107164            .....      .....      112600         .....
3) 110763            .....      .....      112600         .....
4) 112600            19990427   19990513     .....         200295
>
```

**Note:** Dots denote that no information exists for a column entry in the resulting report.

## NF746: Leading Zeros

This feature may be used in Dialogue Manager for date subroutines that return a numeric integer format. It specifically addresses the case where a two-digit year is input in a Dialogue Manager string. The result of the subroutine is 00, representing the year 2000. The leading zeros are truncated when typed out.

### Syntax     Displaying Leading Zeros

```
SET LEADZERO = {ON|OFF}
```

where:

ON

Allows the display of leading zeros if they are present.

OFF

Truncates leading zeros if they are present.

### Example     Preserving Leading Zeros

In the following example the AYM subroutine is being called. The input year is 99 and the month is 12.

```
-SET &IN = '9912';  
-SET &OUT = AYM (&IN, 1, 'I4');  
-TYPE &OUT
```

This yields

1

## Adding

```
SET LEADZERO = ON
```

before the above example yields

```
0001
```

correctly indicating January, 2000.

**Note:** LEADZERO only supports expressions that make a direct call to a subroutine. Expressions that have nesting or other mathematical functions truncate leading zeros. For example,

```
-SET &OUT = AYM(&IN, 1, 'I4'/100;
```



## NF748: HOLD FORMAT WP With Carriage Control

You can now control whether carriage control characters appear in column 1 of each page of report output saved with the HOLD FORMAT WP option. When you choose to include carriage control in a format WP file in MVS, FOCUS creates the file with RECFM VBA so that each report page prints on a separate sheet of paper.

### Syntax      How to Include Carriage Control Characters in a FORMAT WP Hold File

```
[ON TABLE] HOLD AS filename FORMAT WP [CC|NOCC]
```

where:

*filename*

Is the name of the resulting file.

CC

Includes a carriage control character (1) as the first character of each report page in the extract file. In MVS, the file is created with RECFM VBA which causes the operating system to respect the carriage control via page ejects.

NOCC

Omits carriage control characters from the extract file. The file is created with RECFM VB.

When you do not include either CC or NOCC in the syntax, the value of the PAGE-NUM (or PAGE) parameter controls whether carriage control characters are supplied in column 1:

- SET PAGE-NUM=ON (the default) or SET PAGE-NUM=NOPAGE causes column 1 to be blank (no carriage control). In MVS, the file is created with RECFM=VB.
- SET PAGE-NUM=OFF, SET PAGE-NUM=TOP, or using the TABPAGENO option in the heading of the report request supplies carriage control characters. This is the same as prior behavior. However, starting with this release, these settings cause the Hold file to be created with RECFM VBA in MVS.

## Reference Directing Output to a Printer

In MVS, you can send the format WP output directly to a printer by allocating the filename to that printer. You must FREE or CLOSE the file after it is created and before it is printed. For example:

```
SET PAGE = ON
DYNAM ALLOC FILE EMPLOYEE DA PREFIX.EMPLOYEE.FOCUS SHR REU
DYNAM FREE FILE HOLD1
DYNAM ALLOC FILE HOLD1 SYSOUT A DEST IBIVM.P29C1 CLOSE REUSE
TABLE FILE EMPLOYEE
PRINT EMP_ID
BY CURR_SAL BY LAST_NAME
ON TABLE HOLD AS HOLD1 FORMAT WP CC
END
```

**Note:** Your SYSOUT allocation must contain your destination printer.  
In VM/CMS, print the HOLD file using the CC option of the CMS PRINT command to interpret the first character of each record as a carriage control character.

`PRINT fn ft fm (cc`

**Reminder:** If the HOLD file is created with carriage control, it must be printed with carriage control.

## Reference **FORMAT WP Carriage Control and MVS Record Format Options**

The following table summarizes the options for creating a FORMAT WP extract file:

Carriage Control Option	PAGE=ON or NOPAGE		PAGE=OFF or TOP or TABPAGENO in heading	
	Carriage Control?	MVS RECFM	Carriage Control?	MVS RECFM
CC	Yes	VBA	Yes	VBA
NOCC	No	VB	No	VB
none	No	VB	Yes	VBA

## Year 2000 New Features

### Version 7.0 Release 8R

NF557: REBUILD - Legacy Date Conversion

NF653: Displaying Base Dates in FOCUS Reports

NF659: CHECK FILE HOLD ALL

NF700: New Date Math Functions for the Year 2000

NF703: Displaying Invalid Smart Dates in Reports

NF705: Enhancement to the YRTHRESH Command

NF708: Enhancement to the TODAY Subroutine

NF709: Displaying a Date Variable Without Separators

NF710: Field FORMAT=YYJUL

NF711: Altering Your System Date for Testing Purposes

NF713: MSO Log Changes

### Version 7.0 Release 8

Project 2000 - Phase III

NF605: Date Handling for the Year 2000 in FOCUS

NF620: Year 2000 Subroutines

**Version 7.0 Release 6**

**Project 2000 - Phase II**

**NF523: Cross-Century Dates in FOCUS Applications -  
Phase II**

**Version 7.0 Release 5**

**NF497: Project 2000—Cross-Century Dates in FOCUS  
Applications**

## 7.0.8R New Features

Year 2000

### FOCUS

[NF557: REBUILD - Legacy Date Conversion](#)

[NF653: Displaying Base Dates in FOCUS Reports](#)

[NF659: CHECK FILE HOLD ALL](#)

[NF700: New Date Math Functions for the Year 2000](#)

[NF703: Displaying Invalid Smart Dates in Reports](#)

[NF705: Enhancement to the YRTHRESH Command](#)

[NF708: Enhancement to the TODAY Subroutine](#)

[NF709: Displaying a Date Variable Without Separators](#)

[NF710: Field FORMAT=YYJUL](#)

[NF711: Altering Your System Date for Testing Purposes](#)

[NF713: MSO Log Changes](#)

### General Enhancements

[NF714: LE Support](#)

## NF557: REBUILD - Legacy Date Conversion

A new option has been added to REBUILD for legacy date conversion. The option DATE NEW converts legacy dates to smart dates in your FOCUS databases. The utility uses 'update-in-place' technology. It updates your database and creates a new Master File, yet does not change the structure or size of the database.

You must backup the database before executing REBUILD with the DATE NEW option. We recommend that you run the utility against the copy and then replace the original file with the updated backup.

### How the REBUILD Utility Converts Legacy Dates

The utility overwrites the original legacy date field with a smart date. When the storage size of the legacy date exceeds four bytes (the storage size of a smart date), a pad field is added to the database following the date field. Formats A6YMD, A6MDY, and A6DMY are changed to formats YMD, MDY, and DMY, respectively, and have a two-byte pad field added to the Master File. The storage size of integer dates (I6YMD, I6MDY, for example) is four bytes, so no pad field is added. All packed fields and A8 dates add a four-byte pad field. When a date is a key field (but not the last key for the segment), and it requires a pad field, the number of keys in the SEGTYPE is increased by one for each date field that requires padding.

The utility only changes legacy dates to smart dates. The FORMAT in the Master File must be one of the following (Month translation edit options T and TR may be included in the format):

```
A8YYMD A8MDYY A8DMYY A6YMD A6MDY A6DMY A6YYM A6MYM A4YM A4MY
I8YYMD I8MDYY I8DMYY I6YMD I6MDY I6DMY I6YYM I6MYM I4YM I4MY
P8YYMD P8MDYY P8DMYY P6YMD P6MDY P6DMY P6YYM P6MYM P4YM P4MY
```

If you have a field that stores date values but does not have one of these formats, the utility does not change it. If you have a date with one of these formats that you do not want changed, temporarily remove the date components from the FORMAT, run the REBUILD, and then restore the date format.

### Example Using REBUILD With the DATE NEW Option

```
let clear *
set defcent = 19
set yrthresh = 50
set pass = dohide
use

employee focus f
end
rebuild
ENTER OPTION (REBUILD,REORG,INDEX,EXTERNAL INDEX,CHECK OR TIMESTAMP,DATE
NEW)
date new
ENTER THE NAME OF THE MASTER
employee
ENTER THE NEW NAME FOR THE MASTER (FN FT FM)
newemp master a
HAVE YOU BACKED UP THE DATABASE? (YES,NO)
yes
```



If you answer anything other than YES to 'HAVE YOU BACKED UP THE DATABASE', the REBUILD does not continue. Backing up your database is critical to this process.

In CMS, the new Master File is written to the file that you specified in the prompt. The default filetype is MASTER, and the default filemode is A.

In MVS, the new Master File is written to *ddname* HOLDMAST. After the new Master File is created, you should immediately copy it to a permanent dataset. For example:

```
DYNAM COPYDD HOLDMAST(newname) MASTER(newname)
```

### Notes:

- The DBA password for the file must be issued prior to issuing REBUILD.
- The original Master File cannot be encrypted.
- All files must be available locally during the REBUILD, including LOCATION files.
- The Master File cannot have GROUP fields.
- Some error numbers are available in &FOCERRNUM while all error numbers are available in &&FOCREBUILD. Test both &&FOCREBUILD and &FOCERRNUM for errors when writing procedures to rebuild your files.
- To avoid any potential problems, clear all LETs and JOINS before issuing REBUILD.
- DEFCENT/YRTHRESH are respected at the global, file, and field level.

- Correct all invalid date values in the database before executing REBUILD/DATE NEW. The utility converts all invalid dates to zero. Invalid dates used as keys may lead to duplicate keys in the database.
- Adequate workspace, such as temporary attached disk storage, must be available for the temporary REBUILD file. As a rule of thumb, have space 10 to 20% larger than the size of the existing file available.
- REBUILD/INDEX is performed automatically if an index exists.
- REBUILD/REBUILD is performed automatically after REBUILD/DATE NEW when any key is a date.
- Sort libraries and workspace must be available (as with REBUILD/INDEX). The REBUILD allocates default sortwork space in MVS, if you have not already. DDNAMEs SORTIN and SORTOUT must be allocated prior to issuing a REBUILD:

```
DYNAM ALLOC FILE SORTIN   SPACE 5,5 TRACKS NEW
DYNAM ALLOC FILE SORTOUT  SPACE 5,5 TRACKS NEW
DYNAM ALLOC FILE SYSOUT DA * MOD
```

## Restrictions

REBUILD DATE NEW is a remediation tool for your FOCUS databases and date fields only. It does not remediate:

- DEFINES in the Master File
- ACCEPTS in the Master File
- DBA restrictions (especially VALUE restrictions) in the Master File or DBAFILE
- Cross-references to other date fields in this or other Master Files.

- Any references to date fields in your application (FOCEXECs, Master Files).

## **Updated Master File Created by REBUILD/DATE NEW**

The REBUILD/DATE NEW utility creates an updated Master File that reflects the changes made to the database. Once the file has been rebuilt, the original Master File can no longer be used against the database. You must use the new Master File created by the REBUILD/DATE NEW utility.

For example:

```
USE
EMPLOYEE {FOCUS F} AS NEWEMP
END
REBUILD
DATE NEW
EMPLOYEE
NEWEMP
YES
-RUN
-IF (&FOCREBUILD NE 0)OR(&FOCERRNUM NE 0) GOTO error_case;
USE
EMPLOYEE {FOCUS F} AS NEWEMP
END
TABLE FILE NEWEMP
PRINT ....
END
```

The new Master File is an updated copy of the original Master File except:

- The USAGE format for legacy date fields is updated to remove the format and length. The date edit options are retained. For example A6YMDTR becomes YMDTR.
- Padding fields are added for those dates that need them:

```
FIELDNAME= ,ALIAS= ,FORMAT=An,$ PAD FIELD ADDED BY REBUILD
```

where *n* is the padding length (either 2 or 4). Note that the FIELDNAME and ALIAS are blank.

- The SEGTYPE attribute is updated for segments that have remediated dates as keys when the date requires padding and the date is not the last field in the key. The SEGTYPE number will be increased by the number of pad fields added to the key.
- If the SEGTYPE is missing for any segment, the following line is added immediately prior to the \$ terminator for that segment:

```
SEGTYPE=segtype,$ OMITTED SEGTYPE ADDED BY REBUILD
```

where *segtype* is determined by FOCUS.

- If the USAGE for any field, including date fields is missing, the following line is added, immediately prior to the \$ terminator for that field:

```
USAGE=fmt,$ OMITTED USAGE ADDED BY REBUILD
```

where *fmt* is the format of the previous field in the Master File. FOCUS automatically assigns the previous field's format to any field coded without an explicit USAGE= statement.

**Example** Sample Master File: Before and After Conversion

Before Conversion	After Conversion
FILE=filename	FILE=newfilename
SEGNAME=segname, SEGTYPE=S2	SEGNAME=segname, SEGTYPE=S3
FIELD=KEY1,,USAGE=A6YMD,\$	FIELD=KEY1,,USAGE= YMD,\$
	FIELD=, ,USAGE=A2,\$ PAD FIELD ADDED BY REBUILD
FIELD=KEY2,,USAGE=I6MDY,\$	FIELD=KEY2,,USAGE= MDY,\$
FIELD=FIELD3,,USAGE=A8YYMD , \$	FIELD=FIELD3,,USAGE= YYMD,\$
	FIELD=, ,USAGE=A4,\$ PAD FIELD ADDED BY REBUILD

In the conversion of the Master File:

- The SEGTYPE changes from an S2 to S3 to incorporate a 2-byte pad field.
- Format A6YMD changes to smart date format YMD.
- A 2-byte pad field with a blank fieldname and alias is added to the Master File.
- Format I6MDY changes to smart date format MDY (no pad needed).
- Format A8YYMD changes to smart date format YYMD.
- A 4-byte pad field with a blank fieldname and alias is added to the Master File.

## Action Taken on a Date Field During REBUILD/DATE NEW

A new message has been added after a REBUILD has been performed:

NUMBER OF SEGMENTS CHANGED=  $n$

where:

$n$  is the number of segments that have been changed. For example, if there are 10 fields on one segment, and 20 records, then  $n$  is 20 (the number of records/segments changed).

REBUILD/DATE NEW performs a REBUILD/REBUILD or REBUILD/INDEX automatically when a date field is a key or a date field is indexed. The following chart shows the action taken on a date field during the REBUILD/DATE NEW process.

Date is a Key	Index	Result
No	None	NUMBER OF SEGMENTS CHANGED = $n$
No	Yes	REBUILD/INDEX on date field
Yes	None	REBUILD/REBUILD is performed.
Yes	On Any field	REBUILD/REBUILD is performed. REBUILD/INDEX is performed for the indexed fields.

## **REBUILD/DATE NEW Error Messages**

40001 - THIS UTILITY IS NOT SUPPORTED ON THIS PLATFORM  
40002 - FILE NOT BACKED UP - REBUILD NOT EXECUTED  
40003 - THERE ARE NO DATE FIELDS IN THE FILE - REBUILD NOT EXECUTED  
40004 - FILE CONTAINS GROUP FIELDS - REBUILD NOT EXECUTED  
40005 - COMBINE FILE CANNOT BE REBUILT  
40006 - INTERNAL ERROR IN REBUILD/DATE NEW  
40007 - NEW MASTER NAME MUST BE DIFFERENT THAN THE ORIGINAL

## NF653: Displaying Base Dates in FOCUS Reports

You can now display base dates in a FOCUS report. Previously, TABLE always displayed a blank when:

- A date read from a file matched the base date, or
- A field with a smart date format had the value 0

The following chart shows the base date for each supported date format:

<b>FORMAT</b>	<b>Base Date</b>
YMD and YYMD	1900/12/31
YM and YYM	1901/01
YQ and YYQ	1901/Q1
JUL and YYJUL	00/365 and 1900/365 respectively

### Syntax Invoking DATEDISPLAY

```
SET DATEDISPLAY={ON|OFF}
```

where:

ON

Displays the base date if the data is the base date value.



OFF

Displays blanks if the data is the base date value. OFF is the default.  
You cannot set DATEDISPLAY with the ON TABLE command.

## NF659: CHECK FILE HOLD ALL

CHECK FILE HOLD has been enhanced so you can view all of the attributes in a HOLD file including YRTHRESH and DEFCENT. The HOLD file contains two new columns with the values of FDEFCENT and FYRTHRESH at the file level and two new columns with the values of DEFCENT and YRTHRESH at the field level.

The syntax is:

```
CHECK FILE filename HOLD ALL
```

where

*filename*

Is the name of the file whose Master specifications are to be placed in a HOLD file

Then issue the following to see the data about the Master:

```
TABLE FILE HOLD  
PRINT *  
END
```

Running this report request displays columns FDEFCENT and FYRTHRESH at the file level,

FDEFCENT	FYRTHRESH
-----	-----
19	50
19	50

and columns DEFCENT and YRTHRESH at the field level.

DEFCENT	YRTHRESH
-----	-----
19	60
19	50

The four columns shown in the previous two examples represent a small portion of the total information displayed by the TABLE FILE HOLD command.

## NF700: New Date Math Functions for the Year 2000

Several functions have been added to FOCUS enabling you to perform operations on day-based **new dates** in DEFINES, COMPUTEs and anywhere else a function can be used.

### New Date Function Capabilities

The new date functions can:

- Add or subtract date units (months, years, days, weekdays or business days) to or from a date
- Yield a difference between dates in date units
- Move a date to a specific point in the calendar, such as End-of-Month
- Convert from one date format to another (including old dates)

These new functions can help you:

- Compute payroll dates
- Track and ship orders
- Ensure correct credit card transactions

Non day-based date calculations (for example, YM, YQ) can be computed in direct operations (+, -), so they do not need these functions.

### Syntax Adding and Subtracting Date Units to or from a Date

You can add or subtract date units to or from a date by issuing the following:

```
DATEADD (YMDdate, 'unit', #units)
```

where

*YYMDdate*

Is any day-based new date, for example, YYMD, MDY, or JUL

*unit*

Can be one of the following:

- Y (Year)
- M (Month)
- D (Day)
- WD (Weekday)
- BD (Business Day)

*#units*

Is the number of date units you wish to add or subtract to or from the day-based new date

For example,

```
DATE/YYMD = '19991231';
```

```
NEWDATE/YYMD = DATEADD (DATE, 'D', 5);
```

adds 5 days to yield a value of 2000/01/05

The number of units passed to DATEADD is always a whole unit. For example,

```
DATEADD (date, 'M', 1.999)
```

adds 1 month because the number of units is less than 2. Any fractional part is ignored. If the number of units is negative, DATEADD performs subtraction instead of addition.

Invalid date units result in a zero being returned.

If the result of adding months creates an invalid day in the new month, the day is backed off to the end of the resultant month. For example, adding one month to March 31st cannot yield April 31st. Instead, it correctly yields April 30th. The same is true for adding one month to January 29th, 30th, or 31st. All three result in the last day of February (28th, or 29th if a leap year). DATEADD works with smart dates only. The following uses DATEADD to determine whether a date is a business day:

```
SET EMPTYREPORT=ON
DEFINE FILE DATE
  X/YYMD=DATEADD(D1_YYMD, 'BD', 0);
END
TABLE FILE DATE
HEADING
" USE DATEADD TO DETERMINE WHETHER A SMARTDATE FIELD IS A BUSINESS      "
" DAY.  THE DATABASE HAS THE DATE '1998/06/05' WHICH IS A FRIDAY        "
" STORED IN FIELD D1_YYMD.  AN IF TEST IS USED TO DETERMINE IF THE      "
" DATE CORRESPONDS TO A BUSINESS DAY.                                    "
  PRINT D1_YYMD X
  IF X EQ '19980605'
END
TABLE FILE DATE
HEADING
" IT WILL YIELD 0 RECORDS 0 LINES IF THE RESULTING DATE IS NOT A        "
" A BUSINESS DAY.  THE DATABASE ALSO HAS '1998/06/06, A SATURDAY.        "
  PRINT D1_YYMD X
  IF X EQ '19980606'
```

END

The preceding FOCEXEC yields the following:

PAGE 1

USE DATEADD TO DETERMINE WHETHER A SMARTDATE FIELD IS A BUSINESS DAY. THE DATABASE HAS THE DATE '1998/06/05' WHICH IS A FRIDAY STORED IN FIELD D1\_YYMD. AN IF TEST IS USED TO DETERMINE IF THE DATE CORRESPONDS TO A BUSINESS DAY.

```
D1_YYMD      X
-----      -
1998/06/05  1998/06/05
```

PAGE 1

IT WILL YIELD 0 RECORDS 0 LINES IF THE RESULTING DATE IS NOT A BUSINESS DAY. THE DATABASE ALSO HAS '1998/06/06, A SATURDAY.

```
D1_YYMD      X
-----      -
```

## Weekday Units

Weekday units (WD), by default, refer to Monday through Friday. One weekday past a Friday is the following Monday. If the input to DATEADD using WD is a Saturday or Sunday, the input is adjusted to the next weekday before doing the addition. For example,

```
DATEADD (Saturday, 'WD', 1)
```

and

```
DATEADD (Sunday, 'WD', 1)
```

both yield Tuesday as a result because Saturday and Sunday are not business days, so DATEADD begins with Monday and adds 1, yielding Tuesday.

## Business Day Units

You can direct which days are considered business days and which days are not. Business days are traditionally Monday through Friday, but not every business works the same schedule. For example, if your company does business on Sunday, Tuesday, Wednesday, Friday, and Saturday, you can tailor business day units to reflect that situation. Issue the following positionally dependent command to set business days:

```
SET BUSDAYS = {day-list|_MTWTF_}
```

where

*day-list*

is the list of days that represents your business week. The list has a position for each day from Sunday to Saturday.

MTWTF

Represents the positional days of the week from Sunday through Saturday. The underscores represent Sunday and Saturday respectively. Monday through Friday with an underscore before and an underscore afterwards is the default.



Any day that you do not wish to designate as a business day must be replaced with an underscore (\_) in its designated space. If any position within SMTWTFS is either not in its correct position or is not an underscore, an error message is displayed. Using the example of a company that does business on Sunday, Tuesday, Wednesday, Friday and Saturday, business days are represented as:

S\_TW\_FS

To view the current setting of business days issue:

? SET ALL or ? SET

## Holidays

You also have the ability to individually tailor holiday schedules that affect the calculation of business days by skipping those days when calculating offsets. For example, in a given week, if Friday is designated as a holiday, the next business day (BD) after Thursday is the following Monday. In MVS the list of holidays is loaded from a member in ERRORS called HDAYxxxx. In CMS the list is loaded from HDAYxxx ERRORS. A sample Master File, (HDAYDB), and FOCEXEC, (HDAYMAKE), that creates an errors member from a data source used to maintain a list of holidays is available on the FOCUS disk. Create a flat file of holidays as described in the FOCEXEC and execute the FOCEXEC to create the holiday file. The value of xxxx is controlled by the SET HDAY command so that a single installation can support different holiday schedules.

For example,

SET HDAY = STKM

Reads in the holidays from member HDAYSTKM. Each year for which data exists must be represented in the holiday file. Calling a date function with a date value outside the range of the holidays file returns a zero on BD requests. The current setting of HDAY can be viewed with

? SET ALL or ? SET

## **Syntax**      **Returning the Difference between Two Dates**

You can return the difference between two dates by issuing the following:

```
DATEDIF (fromYYMD, toYYMD, 'unit')
```

where

*fromYYMD*

Is the starting date from which to calculate the difference

*toYYMD*

Is the ending date from which to calculate the difference

*unit*

See *Adding and Subtracting Date Units to or from a Date* for valid units

The number of units returned from DATEDIF is always a whole number. For example,

```
DATEDIF (19960302,19970301,'Y')
```

DATEDIF calculates to zero because the difference between March 2, 1996 and March 1, 1997 is less than a whole year. If the to-date is before the from-date, a negative number is returned. For example,

```
DATEDIF (19990228, 19990128, 'M')
```

```
DATEDIF (19990228, 19990129, 'M')
```

DATEDIF (19990228, 19990130, 'M')

DATEDIF (19990228, 19990131, 'M')

all return a result of minus 1 month.

Using DATEDIF with month units yields the inverse of DATEADD. If adding one month to date X creates date Y, then the count of months via DATEDIF between date X and date Y must be one month. The rule is:

If the to-date is an end-of-month then the month difference may be rounded up (in absolute terms) to guarantee the inverse rule. For example,

DATEDIF (March31, May31, 'M') yields 2

DATEDIF (March31, May30, 'M') yields 1 (because May 30 is not the end of the month)

DATEDIF (March31, April30, 'M') yields 1

The same rules apply to year math, the only difference being that February 29th plus 1 year is equal to February 28th.

DATEDIF works with smart dates only.

## Syntax      Moving a Date to a Significant Point

You can move a date to a significant point on the calendar by issuing the following:

```
DATEMOV (YYMDate, 'move-point')
```

where

*YYMDate*

Is the date you wish to move. May be any new date format as long as it implies a day component ( for example MDYY, DMY, but not YM or MYY).

*move-point*

Is the significant point to which you wish to move. Permissible move-points are:

- EOM End of month
- BOM Beginning of month
- EOQ End of quarter
- BOQ Beginning of quarter
- EOY End of year
- BOY Beginning of year
- EOW End of week
- BOW Beginning of week
- NWD Next weekday
- NBD Next business day (Affected by BUSDAY and HDAY files)
- PWD Prior weekday
- PBD Prior business day (Affected by BUSDAY and HDAY files)
- WD- A weekday or earlier
- BD- A business day or earlier (Affected by BUSDAY and HDAY files)
- WD+ A weekday or later
- BD+ A business day or later (Affected by BUSDAY and HDAY files)

DATEMOV works with smart dates only.

The following shows an application using DATEMOV and the report it produces

```
DEFINE FILE CAR
ANUM/I5 WITH COUNTRY = ANUM +1;
ADATEX/YMD WITH COUNTRY = 19980507;
ADATE/YMD = ADATEX + ANUM;
NWD/YMDWT = DATEMOV(ADATE, 'NWD' );
PWD/YMDWT = DATEMOV(ADATE, 'PWD' );
WDP/YMDWT = DATEMOV(ADATE, 'WD+' );
WDM/YMDWT = DATEMOV(ADATE, 'WD-' );
NBD/YMDWT = DATEMOV(ADATE, 'NBD' );
PBD/YMDWT = DATEMOV(ADATE, 'PBD' );
WBP/YMDWT = DATEMOV(ADATE, 'BD+' );
WBM/YMDWT = DATEMOV(ADATE, 'BD-' );
END
SET BUSDAY = _MTWT___
TABLE FILE CAR
HEADING
"EXAMPLES OF DATEMOV"
"BUSINESS DAYS ARE MONDAY, TUESDAY, WEDNESDAY, + THURSDAY "
" "
"START DATE.. 3 MOVE POINTS....."
PRINT ADATE/WT AS 'DOW'
NWD/WT PWD/WT WDP/WT AS 'WD+' WDM/WT AS 'WD-'
NBD/WT PBD/WT WBP/WT AS 'BD+' WBM/WT AS 'BD-'
BY ADATE
END
```

yields:

EXAMPLES OF DATEMOV

BUSINESS DAYS ARE MONDAY, TUESDAY, WEDNESDAY, + THURSDAY

START DATE.. | MOVE POINTS.....

ADATE DOW NWD PWD WD+ WD- NBD PBD BD+ BD-  
 -----

98/05/08	FRI	MON	THU	FRI	FRI	TUE	WED	MON	THU
98/05/09	SAT	TUE	THU	MON	FRI	TUE	WED	MON	THU
98/05/10	SUN	TUE	THU	MON	FRI	TUE	WED	MON	THU
98/05/11	MON	TUE	FRI	MON	MON	TUE	THU	MON	MON
98/05/12	TUE	WED	MON	TUE	TUE	WED	MON	TUE	TUE

Invalid move-points result in a zero being returned.

## Syntax Converting From One Date Format to Another

Applications no longer have to use intermediate calculations to convert date formats. Instead you can issue the following:

```
DATECVT (indate, 'infmt', 'outfmt')
```

where

*indate*

Is the date whose format you wish to change

*infmt* and *outfmt*

Can be:

- Any new date format (for example, YYMD, YQ, M, DMY, JUL) that matches the format of *indate*. It can also be in the format of the output value enclosed within single quotes.
- Any old date format (such as I6YMD or A8MDYY)
- Non-date formats (such as I8, or A6). Non-date formats on *infmt* are treated as offsets from the base date (12/31/1900). Use the DAYMD function to retrieve the offset of a date.

The format of the field on the left side of the equal sign must match the *outfmt* value. For example,

```
field/DMY = DATECVT (indate, 'YYMD', 'DMY');
```

If the value of *indate* is 19991231 then the field is set to the offset, which is 311299. *Indates* with old formats obey any DEFCEINT and YRTHRESH values implied for that field when performing the conversion.

Invalid old dates passed to DATECVT cause a zero to be returned as does a DEFINE. Invalid formats in DATECVT cause a zero or blank to be returned.

## New Date Math Functions in MAINTAIN

MAINTAIN supports the new date functions DATEADD, DATEDIF, and DATEMOV with an extra parameter: the result field. In MAINTAIN, you can code:

```
COMPUTE ADATE/YYMD = ... (some expression)
```

```
COMPUTE DUE_DATE/YYMD = DATEADD(ADATE, 'BD', 20, DUE_DATE);  
COMPUTE NOTICE_DATE/YYMD = DATEMOV(DUE_DATE+1, 'EOM', NOTICE_DATE);  
COMPUTE TOTAL_DAY/I4 = DATEDIF(ADATE, NOTICE_DATE, 'BD', TOTAL_DAY);
```

**DATECVT** is not supported in **MAINTAIN**. If you attempt to use **DATECVT** in **MAINTAIN** the following message displays:

```
FUNCTION NOT FOUND ERROR
```



## NF703: Displaying Invalid Smart Dates in Reports

In previous releases of FOCUS, if a date field in a non-FOCUS file contained an invalid date, a diagnostic error was displayed and the entire record failed to display in a report. For example, if a date field contained '980450' with an ACTUAL of A6 and a USAGE of YMD, the record containing that field would not display. With the use of a new command, it is possible to display the rest of the record that contains the incorrect date.

### Syntax Invoking ALLOWCVTERR

```
SET ALLOWCVTERR = {ON|OFF}
```

where

ON

Allows the display of a field containing an incorrect date.

OFF

Generates an error if bad data is encountered, and does not display the record containing the bad data. This behavior is identical to releases prior to FOCUS Release 7.0.8R.

When it encounters a bad date, ALLOWCVTERR sets the value of the field to either MISSING or to the base date. When a field is being converted and ALLOWCVTERR is set on, FOCUS first checks to see if MISSING=ON. The following chart shows the results of interaction between DATEDISPLAY and MISSING assuming ALLOWCVTERR=ON and the presence of a bad date.

	MISSING=OFF	MISSING=ON
DATEDISPLAY=ON	Displays Base Date 19001231 or 1901/1	.
DATEDISPLAY=OFF	Displays Blanks	.

DATEDISPLAY only affects how the base date is displayed.

**Note:** See New Feature Bulletin 653, *Displaying Base Dates in FOCUS Reports*, for detailed information concerning the setting of DATEDISPLAY.

## NF705: Enhancement to the YRTHRESH Command

You can now set YRTHRESH as an offset from the current year in addition to specifying a year. This technique creates a moving century window that increments itself each year without modifying your application.

You decide the number of years to offset in YRTHRESH. For example, if the current year is 1998 and you wish to set YRTHRESH to 60, you can set YRTHRESH to -38 (1998 - 38 = 60). By setting YRTHRESH to a negative number FOCUS subtracts, in this example, 38 from whatever the current year is. In the year 1999 YRTHRESH is 61 instead of 60 (1999 - 38 = 61) illustrating how the moving window application functions without outside intervention.

If you set YRTHRESH to a large enough value that crosses a century boundary, the value of DEFCENT is recalculated. For example, if you set YRTHRESH to minus 99 (1998-99 = -1), DEFCENT is calculated to 18 and YRTHRESH becomes 99. The 100- year span begins with a pivot year of 1899 and ends with year 1998. ? SET and ? SET ALL now reflect the new settings of DEFCENT.

The following request shows the use of an offset with DEFCENT set to 19 and YRTHRESH set to -38 (where the current year is 1998), followed by the output:

```
SET DEFCENT=19, YRTHRESH=-38
TABLE FILE DATE
PRINT D2_I6YMD AND COMPUTE
NEWDATE/I8YMD=AYMD(D2_I6YMD,1,NEWDATE);
END
```

## **NF705: Enhancement to the YRTHRESH Command**

---

D2_I6YMD	NEWDATE
-----	-----
60/04/02	1960/04/03
66/06/06	1966/06/07
60/12/13	1960/12/14
53/06/06	2053/06/07
59/08/11	2059/08/12
60/02/28	1960/02/29

## NF708: Enhancement to the TODAY Subroutine

The TODAY subroutine is year 2000 compatible and is useful in a compiled MODIFY. It can return a 4-digit year when you declare a DEFINE or COMPUTE field as 10 bytes. TODAY can continue returning a 2-digit year when you declare the output format as 8 bytes.

For example,

```
DEFINE FILE EMPLOYEE
NOWDATE/A10 WITH EMP_ID = TODAY (NOWDATE)
END

TABLE FILE EMPLOYEE
HEADING
" SALARY REPORT RUN ON DATE <NOWDATE>"
" "
PRINT DEPARTMENT CURR_SAL
BY LAST_NAME BY FIRST_NAME
END
```

The DEFINE may also be coded as

```
NOWDATE/A10 WITH EMP_ID = TODAY('A10');
```

The request produces the following report:

```
SALARY REPORT RUN ON DATE 06/08/1998
```

LAST_NAME	FIRST_NAME	DEPARTMENT	CURR_SAL
-----	-----	-----	-----
BANNING	JOHN	PRODUCTION	\$29,700.00
BLACKWOOD	ROSEMARIE	MIS	\$21,780.00
CROSS	BARBARA	MIS	\$27,062.00
GREENSPAN	MARY	MIS	\$9,000.00
IRVING	JOAN	PRODUCTION	\$26,862.00
JONES	DIANE	MIS	\$18,480.00
MCCOY	JOHN	MIS	\$18,480.00
MCKNIGHT	ROGER	PRODUCTION	\$16,100.00
ROMANS	ANTHONY	PRODUCTION	\$21,120.00
SMITH	MARY	MIS	\$13,200.00
	RICHARD	PRODUCTION	\$9,500.00
STEVENS	ALFRED	PRODUCTION	\$11,000.00

**Note:** DATEFNS must be set to ON to retrieve the extended TODAY value.

## NF709: Displaying a Date Variable Without Separators

You can now display a date variable containing a 4-digit year without separators. The variables are &YYMD, &MDYY and &DMYY. These variables complement the existing 2-digit year variables &YMD, &MDY, and &DMY.

The following example shows a report using &YYMD:

```
TABLE FILE EMPLOYEE
HEADING
"SALARY REPORT RUN ON DATE &YYMD"
" "
PRINT DEPARTMENT CURR_SAL
BY LAST_NAME BY FIRST_NAME
END
```

The resulting output for May 18, 1998 is:

---

SALARY REPORT RUN ON DATE 19980518

LAST_NAME	FIRST_NAME	DEPARTMENT	CURR_SAL
-----	-----	-----	-----
BANNING	JOHN	PRODUCTION	\$29,700.00
BLACKWOOD	ROSEMARIE	MIS	\$21,780.00
CROSS	BARBARA	MIS	\$27,062.00
GREENSPAN	MARY	MIS	\$9,000.00
IRVING	JOAN	PRODUCTION	\$26,862.00
JONES	DIANE	MIS	\$18,480.00
MCCOY	JOHN	MIS	\$18,480.00
MCKNIGHT	ROGER	PRODUCTION	\$16,100.00
ROMANS	ANTHONY	PRODUCTION	\$21,120.00
SMITH	MARY	MIS	\$13,200.00
	RICHARD	PRODUCTION	\$9,500.00
STEVENS	ALFRED	PRODUCTION	\$11,000.00

---



## NF710: Field FORMAT=YYJUL

A new date field formatting option, FORMAT=YYJUL, lets you print a Julian date in the format YYYY/DDD. The 7-digit format displays the four-digit year and the number of days counting from January 1. For example, January 3, 2001 in Julian format is 2001003.

FORMAT=JUL is still supported; however, only the last two digits of the year display (YY/DDD).

### Example Displaying a Date in YYJUL Format

The example displays expiration dates in YYJUL format. It also illustrates the definition of a new date field that converts the Julian date and displays it in YYMD format.

```
FILENAME=EXAMPLE, SUFFIX=FOC
SEGNAME=ROOT, SEGTYPE=S1
FIELDNAME=DRIVER_ID, ALIAS=, FORMAT=A9, $
FIELDNAME=EXP_DATE, ALIAS=, FORMAT=YYJUL, $
FIELDNAME=CLASS, ALIAS=, FORMAT=A2, $

DEFINE FILE EXAMPLE
  NEWDATE/YYMD=EXP_DATE;
  END
TABLE FILE EXAMPLE
  PRINT EXP_DATE NEWDATE CLASS
  BY DRIVER_ID
END
```

PAGE 1

DRIVER_ID	EXP_DATE	NEWDATE	CLASS
-----	-----	-----	-----
123254365	2000/139	2000/05/18	A4
254069503	1999/068	1999/03/09	R4
675678904	2003/253	2003/09/10	W9

## NF711: Altering Your System Date for Testing Purposes

TESTDATE is a new SET command that allows you to temporarily alter your FOCUS system date for a given application program. This allows you to determine what impact the year 2000 will have on your application programs.

**Note: Only use TESTDATE for testing purposes with a test database.**

The syntax for TESTDATE is:

```
SET TESTDATE = {yyyymmdd|TODAY}
```

where

*yyyymmdd*

Is an 8-digit date in the format YYYYMMDD.

TODAY

Is today's date. TODAY is the default.

The value of TESTDATE affects all reserved variables that retrieve the current date from the system. Setting TESTDATE also affects anywhere in FOCUS that a date is used (such as CREATE, MODIFY, MAINTAIN) but does not affect the date referenced directly from the system.

TESTDATE can either be equal to TODAY or a date in the format YYYYMMDD.

If anything else is entered the following message is displayed:

```
TESTDATE MUST BE YYYYMMDD OR TODAY
```

## NF713: MSO Log Changes

The MSO logs now display dates with four-digit years.

### **Sample MSO Log**

Following is a portion of a sample MSOPRINT log. Note that the dates display with four-digit years:

```

06/10/1998 12.12.13 (MSD13137) PROCESSING EDACPG CONFIGURATION FILE
06/10/1998 12.12.13 (MSD13137) PROCESSING FOOMSO CONFIGURATION FILE
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 1 *-----
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 2 * Sample MSO configuration file.
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 3 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 4 * This file is allocated to dname FOOMS
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 5 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 6 * The records and options which may be sp
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 7 * documented in the MSO Installation and
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 8 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ... 9 * Not all options which may be specified
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...10 * file; the options and values listed are
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...11 * be tailored to meet your installation's
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...12 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...13 * Please note that some of the options wh
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...14 * sample require APF authorization and/or
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...15 * steps; please check the manual for furt
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...16 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...17 *-----
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...18 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...19 *-----
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...20 * Global processing and options:
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...21 *-----
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...22 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...23 * Enable use of an external security packa
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...24 * EXTSEC = NO
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...25 * EXTSEC = YES
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...26 *
06/10/1998 12.12.14 (MSD13136) CONFIG LINE ...27 * This installation of processing statisti

```

## NF714: LE Support

You can now control use of IBM's Dynamic Language Environment for IBI-supplied and user-written subroutines by setting an option in FOCPARM, in a FOCUS application, or in your FOCUS session.

### Syntax      How to Control the LE Environment Setting

The syntax for specifying the type of LE support you need is

```
SET IBMLE = {OFF|ON|ALL}
```

where:

OFF

Does not initialize the LE environment. OFF is the default value and is the recommended setting for applications using only IBI-supplied subroutines.

ON

Establishes the LE pre-initialization environment with the IBM default configuration. This configuration initializes the LE environment for subroutines coded in COBOL, PL/I, C, and ASMH if the routines are linked with the LE environment. If the application calls a module not supported under LE, it runs without LE. For a list of languages supported under LE, see [LE Language Support for User-Written Subroutines](#).

ON is the recommended setting for applications that call user-written subroutines linked with the LE environment and not coded in FORTRAN. ON is also recommended for applications that call a combination of these subroutines and IBI-supplied subroutines. Running IBI-supplied subroutines with this setting requires LE version 1.5 or above.

### ALL

Should be used only for user-created FORTRAN subroutines that need the LE environment. **Note:** This is not supported by IBM. The ALL setting adds FORTRAN to the list of languages supported for LE pre-initialization. The FORTRAN run-time libraries must be installed under LE. However, FORTRAN modules do not run under LE. ALL is the recommended setting for applications that call user-written subroutines written in FORTRAN if the FORTRAN run-time libraries were installed under LE. This setting requires LE version 1.5 and above.

### Reference LE Language Support for User-Written Subroutines

FOCUS supports LE access to user-written subroutines coded in the following LE-supported languages:

- C/MVS™.
- COBOL for MVS & VM (Release 2).
- COBOL/370™ (Release 1).
- VS FORTRAN (Version 2).
- PL/I for MVS & VM.
- ASMH (with macro support).

## Recommended IBMLE Settings

LE pre-initialization may be beneficial for application performance when using 3GL user-written subroutines. However, users should be aware that IBM may modify the LE environment at any time; its use is the responsibility of the user based on system analysis and resource requirements. IBM provides extensive documentation regarding LE at the following URL: [www1.s390.ibm.com/os390](http://www1.s390.ibm.com/os390).

Use the guidelines in the following table to determine the appropriate setting for each FOCUS application. If you need to change the setting in a particular application or in a FOCUS session, issue the SET IBMLE command prior to executing the first subroutine call. Results can be unpredictable if you change the setting between FOCEXECs or subroutine calls.

<b>The application calls ...</b>	<b>Setting</b>
At least one FORTRAN module, and the FORTRAN runtime libraries were installed under LE.	ALL
Modules linked with the LE environment and coded in languages other than FORTRAN.	ON
Only IBI-supplied subroutines and modules not linked with LE.	OFF

## Determining Proper IBMLE Settings

The following table describes the results of various IBMLE settings on supported LE subroutines:



<b>Type of Subroutine</b>	<b>Required Setting</b>	<b>Effects of Changing the IBMLE Setting</b>
IBI-supplied	any setting	None, works with any IBMLE setting.
FORTRAN/LE	ALL	ALL is the only supported setting for FORTRAN/LE. Requires LE release 1.5 or above.

COBOL/LE	ON or ALL	<p>For performance reasons, link-edit the subroutine as reusable (<code>reus</code>).</p> <p>If the subroutine must be executed with IBMLE=OFF, apply the COBOL run-time option <code>rtreus</code> to the routine.</p> <p>Without this option, the subroutine returns invalid data and generates the following messages:</p> <pre>IGZ0044S There was an attempt to call the COBOL main program [xxxxxxxx] that was not in initial state.  The traceback information could not be determined.</pre> <p>The job does not abend but produces a CEEDUMP prefaced by:</p> <pre>CEE3DMP[release]: Condition processing resulted in the unhandled condition.</pre>
PL/I - LE	ON or ALL	<p>If executed with IBMLE=OFF, results in an 0C1 abend.</p>

**Note:**

- Non-LE Assembler subroutines require source code changes in order to be LE compliant.

- Mixed-mode applications calling both LE and non-LE subroutines in the same FOCEXEC or FOCUS session are not supported and may produce unpredictable results.

## 7.0.8 New Features

### Year 2000

Project 2000 - Phase III

NF605: Date Handling for the Year 2000 in FOCUS

NF620: Year 2000 Subroutines

### Performance Enhancements

NF564: Pooled Tables

NF593: IUCV CMS SU

NF617: Automatic Allocation of FOCUS Files

### Raised Limits

NF642: Increased DEFINE Limitation

### Reporting Enhancements

NF579: Assigning Screening Conditions to a File for Reporting Purposes

NF586: Expanding Byte Precision for COUNT and LIST

NF623: Increasing the Number of Verbs in a Report Request

## General Enhancements

[NF607: TABLA Enhancements](#)

[NF609: Sink Validation of Userids in CMS](#)

[NF630: Querying Which PTFs Have Been Applied for a Specific Release](#)

[NF631: Extended Plists](#)

[NF640: Dynamic Language Environment \(LE\) Support](#)

## The Multi-Session Option

[NF566: MSO/CICS Cooperative Processing](#)

## Web Interface for FOCUS

[NF619: -HTMLFORM SAVE](#)

[NF626: JAVA Graph Wizard](#)

[NF594: JAVA Report Assist](#)

[NF628: Automatic Activation of Web Interface for Web Browser Users](#)

[NF645: WEBHOME](#)

## Relational Interfaces

[NF568: DB2 Interface IF-THEN-ELSE Optimization](#)

[NF571: DB2 Interface SET ISOLATION Command](#)

## **Model 204 Interface**

**NF572: Invisible Ordered Character and Ordered Numeric Data Type Key Support**

## **IMS Interface**

**NF550: EDA/MSO Console Display for IMS PSB**

## **System 2000 Interface**

**NF574: System 2000 Interface Trace Facility**

## **Teradata Interface**

**NF583: Teradata Outer Join Optimization**

## **National Language Support**

**NF647: Extended Support for Scandinavian External Sort**

## NF550: EDA/MSO Console Display for IMS PSB

The EDA/MSO Console for the DU (Display Users) screen includes a new column heading named **PSB**. This column displays the name of the IMS PSB scheduled for each TCB that is accessing IMS. IMS PSB names are up to 8 characters in length.

### Usage

When an EDA/MSO user subtask is accessing IMS, a PSB is used. By selecting DU from the Console and scrolling the display to the third screen to the right, a column named PSB will display the IMS PSB scheduled by a TCB. This is especially useful when a user subtask is not responding due to a runaway query. If the operator decides to, he can cancel that particular subtask.

In the case where an IMS error condition is encountered, the PSB column remains populated; however, an asterisk (\*) is placed in the first position as an indicator of an error condition. This can be useful for debugging purposes and/or notification of other users attempting to use that particular PSB.

### Example Sample Console DU Screen

Below is an example of the CONSOLE DU screen display:

```
<PMSLCCM JOB04558----- .CONSOLE DISPLAY USERS PANEL. -----
Line:001(002) Col:153
COMMAND ==>

C Logon_ID Runcount MVSDD USER-DD OPN Dds QueryID1 QueryID2 PSB
*REGION*
```

In the above example, userid USER01 has scheduled a PSB named ALLPSB. If an error was encountered, the PSB display for the user would have read:

**\*LLPSB.**

## Reference **Special Considerations**

The PSB name will be populated at the time that the PSB is scheduled by the user subtask, and is removed after the DBCTL thread connection is successfully de-allocated. If an IMS error is encountered, the PSB column will remain populated until another PSB is scheduled. The PSB column is cleared whenever an END THREAD command is processed, or an OPEN thread is attempted. This feature requires no action to be activated.

## **Error Messages**

None.



## NF564: Pooled Tables

Pooled Tables is a FOCUS performance feature for reporting applications that enables many reports or extract files to be created from a single pass of a database. Requests from any database, file, or JOINed structure that FOCUS reads can be pooled, reducing all of the normal reporting costs including database I/O, CPU and elapsed time. Performance gains with Pooled Tables can be dramatic; there is no penalty for its use -- even with applications that do not take advantage of it.

Pooled Tables is simple to use: by just adding a few lines to your application, Pooled Tables does the work of identifying reports that can share database I/O and running them concurrently.

Pooled Tables is ideal for large applications with many reports and batch reporting applications. Additionally, reports where data retrieval costs are significant compared to formatting costs benefit greatly from Pooled Tables.

There are several additional efficiencies that users can employ to maximize performance gains achievable through Pooled Tables.

This document describes how to use Pooled Tables. Refer to the *Pooled Tables White Paper* (DN1100978.0498) for additional information describing the internal logic.

This document covers the following topics:

- Overview - a general overview of Pooled Tables.
- Memory Needs - covers memory requirements for Pooled Tables.
- Report Size Estimates - describes record and line estimates for Pooled Tables.

- FOCPOOLT - describes a temporary work file Pooled Tables sometimes creates.
- Reporting Statistics - describes changes to the ? STAT output.
- Sort Selection - describes criteria for sort selection.
- Managing Memory - describes how you can control memory usage.
- Common Selection Criteria - describes efficiencies developers can employ.
- Reporting from Non-Relational databases - provides general information for using Pooled Tables with Non-Relational databases.
- Reporting from Relational databases - provides general information for using Pooled Tables with Relational databases.
- Trace Facility - describes the Pooled Tables trace facility.
- Pooled Tables in Batch Mode - provides general information for using Pooled Tables in batch mode.
- Tuning applications - provides suggestions for using Pooled Tables more efficiently.
- Syntax - provides syntax reference.
- Pooled Tables Example - illustrates the use of Pooled Tables commands.
- Single TABLE Clusters - describes situations when pooling is not done.
- Subpool Boundary Conditions - describes situations that constitute boundaries.
- Pooled Tables Installation Instructions
- Usage Notes
- Frequently Asked Questions

- Error Processing

## Overview

Pooled Tables should be used whenever two or more consecutive reports are executed against the same database. It is ideal for use with large batch or canned FOCUS reporting runs and data-extract applications. The feature is implemented with only a few simple SET commands. There is no need to change an application. A sample program that uses Pooled Tables is illustrated in the section [\*Pooled Tables Example\*](#).

A pool begins with the command `SET POOL=ON` and continues until `SET POOL=OFF` is encountered. Within a pool, FOCUS reads ahead in an application searching for consecutive TABLE requests that access the same file using the same access method. This read ahead feature extends across FOCEXECs and divides commands into retrieval and non-retrieval categories called subpools. A subpool is a collection of TABLE requests and related commands. Only report requests within a subpool can be combined.

Commands that alter data or the processing environment create subpool boundaries. For example, in the sequence

```
TABLE FILE CAR
... END
TABLE FILE CAR
... END
MAINTAIN FILE CAR
... END
TABLE FILE CAR
... END
```

a subpool boundary occurs at the MAINTAIN command. A list of subpool boundary commands appears in the section [Subpool Boundary Conditions](#).

Subpools are further divided into clusters. A cluster is a set of consecutive TABLE requests that share the same logical database and access method. For example, two TABLE requests against a VSAM file, one using sequential access and the other using indexed retrieval, are placed in separate clusters because of the different access methods. Reports that cannot be pooled because of syntax or environmental conditions are executed as single TABLE clusters. A list of these conditions appears in the section “*Single TABLE Clusters*.”

All TABLE requests in a single cluster are executed concurrently and share the same data retrieval and screening processes. Sorting and output formatting are not shared.

The figure below diagrams the breakdown of pools into sub-pools and clusters:

```

SET POOL=ON

TABLE FILE EMPLOYEEE
END
TABLE FILE EMPLOYEEE
END
TABLE FILE EMPLOYEEE
END
TABLE FILE CAR
END
TABLE FILE CAR
END
-RUN
TABLE FILE CAR
END
-RUN
TABLE FILE EMPLOYEEE
END
TABLE FILE EMPLOYEEE
END
SET POOL=OFF

```

Diagram illustrating the structure of pooled tables:

- The first three tables (EMPLOYEEE) are grouped under a **SUBPOOL**.
- The next three tables (CAR) are grouped under a **POOL**.
- The final two tables (EMPLOYEEE) are grouped under a **SUBPOOL**.
- The **POOL** label is positioned to the right of the middle section, encompassing the CAR tables and the second SUBPOOL.

## Memory Needs

The number of reports that can be executed in a single cluster is limited only by the amount of memory the user allocates to Pooled Tables (POOLMEMORY). More reports can be run concurrently with Pooled Tables when larger amounts of memory are available. Memory requirements for each report depend on the number of records included in the report, the number of lines of output, and the width of the report. Pooled Tables calculates these memory requirements. In general, the memory needed for small summary reports can be estimated as `NUMBER OF LINES OF OUTPUT * REPORT WIDTH`. The memory needed for large summary reports and detail reports can be estimated as `NUMBER OF RECORDS SELECTED * REPORT WIDTH`. The Pooled Tables trace facility displays the actual amount of memory allocated for each report and the statistics used to calculate it.

**When available memory is insufficient for simultaneously executing all of the requests in a cluster, Pooled Tables executes them in a series of steps, called iterations.**

When multiple iterations are required, Pooled Tables produces as many reports as it can directly in memory during the first iteration. Concurrently, data for the remaining reports in the cluster are staged in a work file called FOCPOOLT. The rest of the reports are then produced from the data in FOCPOOLT in subsequent iterations. The source database is only accessed once at the beginning of the process.

## Report Size Estimates

To calculate memory needs, Pooled Tables requires accurate estimates of the size of each report to deliver optimal performance. These estimates are used to select the appropriate sort and distribute memory resources equitably across the several reports in the pool. Input report size equals the number of records in the report following selection (ESTRECORDS); output report size is the number of output lines after aggregation is complete (ESTLINES). These estimates apply to the individual reports, not the size of the set of reports in the cluster.

ESTLINES and ESTRECORDS estimates can be gathered from:

- The statistical message: NUMBER OF RECORDS IN TABLE= LINES=
- The RECORDS and LINES information available on the ? STAT output.
- Previously gathered information from the &RECORDS and &LINES variables.
- When ACROSS is used, ESTLINES = number of lines X number of unique ACROSS columns.
- When IF TOTAL or WHERE TOTAL is used, ESTLINES is the number of lines before the TOTAL selection is made.

These estimates should be set individually for every request. Global settings can be issued, however. If ESTRECORDS is set for a group of requests, the estimate should be representative of the most common reports and need not exceed the size of the database.

## FOCPOOLT

The temporary work file FOCPOOLT is created only when a cluster contains more reports than can be executed in available memory (POOLMEMORY). If a cluster can be produced directly from memory, the FOCPOOLT file is not created.

For example, a cluster has 30 reports, each of which requires 1 megabyte of memory. There are 10 megabytes of memory available (POOLMEMORY). Pooled Tables retrieves all of the data once and produces the first 10 reports from memory (this is the first iteration). The records for the remaining 20 reports are written to the work file FOCPOOLT. In the second iteration, Pooled Tables reads data for the next 10 reports from FOCPOOLT and produces them. The final 10 reports are produced in the third iteration.

It is more efficient to get data from FOCPOOLT than the database, because data in FOCPOOLT has already been screened and formatted. In addition, Pooled Tables determines accurate record counts (ESTRECORDS) for all reports in the second and subsequent iterations. Memory needs for these reports are more accurately calculated, further optimizing Pooled Tables performance.

The size of FOCPOOLT depends on the volume of data in the reports that are executed in the second and subsequent iterations. Data required only for reports in the first iteration are not stored in FOCPOOLT. Overlapping data required for more than one report is stored only once. The size of FOCPOOLT will never exceed the size of the logical database used for the cluster. Typical size requirements for FOCPOOLT are the same as those for the largest FOCSORT file for any report in the pool.



For MVS users, the default allocation for FOCPOOLT is 5 primary and 20 secondary cylinders. FOCUS uses a second volume if all extents on the first volume are used. It is recommended that the user pre-allocate FOCPOOLT with the necessary space attributes under MVS. DCB information will be determined by FOCUS.

In CMS, adequate temp disk space must be made available for this file.

## **Reporting statistics**

? STAT has been enhanced to display pooling statistics for each pooled report. It can be used to identify pooling characteristics in an application and tune the application.

Below is an annotated sample of the output for ? STAT with only the information for Pooled Tables shown.

STATISTICS OF LAST COMMAND									
	RECORDS	=	50000	.					
	LINES	=	50000	.					
	.			.					
	.			.					
	.			.					
1.	READS	=	250000	.					
	.			.					
	.			.					
	.			.					
	.			.					
2.	SUBPOOL	=	1						
3.	CLUSTER	=	2		ITERATION	=	1	8.	
4.	#CLUSTER ITEMS	=	25		#ITER ITEMS	=	16	9.	
5.	SEQ# IN CLUSTER	=	5		SEQ# IN ITER	=	5	10.	
6.	ESTIMATED RECS	=	50000		ESTIMATED LINES	=	50000	11.	
7.	REPORT WIDTH	=	148						

Where, for the report just run:

1. **READS**

The total number of records retrieved for the cluster.

2. SUBPOOL                      The report is in the first subpool.
3. CLUSTER                      The report is in the second cluster.
4. # CLUSTER ITEMS              There are 25 reports in the cluster.
5. SEQ# IN CLUSTER              This is the fifth report in the cluster.
6. ESTIMATED RECS              ESTRECORDS has been set to 50,000. This number should be compared with RECORDS at the top of this ? STAT. If there is a discrepancy, change the ESTRECORDS value for this report.
7. REPORT WIDTH                The report width is 148 bytes.
8. ITERATION                    This report was produced in the first iteration.
9. # ITER ITEMS                There are 16 reports produced in the first iteration. The next 9 reports are executed during subsequent iterations.
- 10 SEQ# IN ITER                This is the fifth report in this iteration.  
.
- 11 ESTIMATED LINES              ESTLINES has been set to 50,000. This number should be compared with LINES at the top of this ? STAT. If there is a discrepancy, change the ESTLINES value for this report.  
.

## Sort Selection

Pooled Tables uses the report size estimates to choose the appropriate sort: an in-memory FOCUS sort or an external sort. The FOCUS sort is used for all reports whose memory needs are less than 1 megabyte. In general, an external sort is used for all other cases. The maximum number of concurrently executing sorts, and thus the maximum number of concurrently executing reports, is limited by the amount of memory available to Pooled Tables. The maximum number of external sorts that can be used by Pooled Tables is 26. This number can be decreased with the MAXEXTSRSTS setting. The number of external sorts can also be limited by the amount of available memory below the 16 megabyte line in MVS. In VM, only one version of the external sort can be executed when the sort package is SyncSort. When practical, the FOCUS sort is substituted for the external sort when the number of external sorts is limited but memory is available.

## Managing Memory

The maximum amount of memory used by Pooled Tables can be limited with the POOLMEMORY setting. In MVS, the number represents memory above the 16 megabyte line. In VM, the number represents total virtual memory. The default value for POOLMEMORY is 16,384 K (16 M). The minimum value is 1,024 K. A maximum bound can be placed on POOLMEMORY when Pooled Table is installed. In MVS, you can also control the total amount of memory available from the operating system above the 16 megabyte line by coding REGION=nM in your JCL job card , where n is greater than 16. POOLMEMORY can be set from the command line, during FOCUS initialization (in the PROFILE FOCEXEC), or within an application.

Memory is reserved by using the POOLRESERVE setting. This reserves a portion of available memory for system or FOCUS use that is not to be used by Pooled Tables. In MVS, the default is 100K. In VM the default is 1,024 K. The default value can be changed at installation time. POOLRESERVE can be set from the command line, during FOCUS initialization (in the PROFILE FOCEXEC), or within an application.

The purpose of POOLRESERVE is to reserve memory during Pooled Tables' parsing and decision making process for other modules. For example, first time access to SQL/DS requires loading of IBI interface code and IBM modules. The memory needed for these is not used in the Pooled Tables case until the common read is executed. After these modules are loaded, POOLRESERVE can be reduced, possibly to zero. If the IBI interface and IBM load modules are stored in a saved segment, POOLRESERVE can be reduced prior to execution of Pooled Tables.

Suggested values for POOLRESERVE are :

```
Running an interface (not in saved segment): 1024 K
Running an interface (in saved segment):      256 K
Using SyncSort as the external sort:          512 K
Using any other sort:                          128 K
```

## Common Selection Criteria

Common selection statements that appear in every report in the cluster are applied during Pooled Tables retrieval. The common test must refer to the same field and use an equality screening relational operator (`EQ` or `IS`). The selected values do not need to be the same in all reports. For example, if the first report has the test `WHERE FISCAL_YEAR EQ 1997` and the second request has the test `WHERE FISCAL_YEAR EQ 1998`, the test `WHERE FISCAL_YEAR EQ 1997 OR 1998` is evaluated during Pooled Tables retrieval. Common selection tests are included to reduce the size of the answer set returned for a pool.

Common selection criteria that do not use equality can be evaluated by Pooled Tables using another FOCUS feature: Filters. Filters allow you to specify simple or complex selection tests for all reports against the same file. Filters in effect for all reports in a cluster are also applied during Pooled Tables retrieval. The use of Filters allows you to reduce the size of the pooled answer set, even when there are no common equality selection tests.

## Reporting from non-Relational Databases

Reports against non-relational databases, such as VSAM, IMS, IDMS, FOCUS, and sequential files, must meet several simple criteria in order to be pooled together into one cluster. First, all reports must access the same database, using the same Master File Description. Next, the reports in a cluster must share the same access method. For example, reports that use sequential access can be pooled together; reports that use indexed access can be pooled together. Finally, all reports in a cluster must share the same entry point. That is, the reporting view must be from the same segment and, in the case of indexed access, from the same field. Reports against sequential files always meet these criteria so they always pool. Reports against JOINed structures are pooled together based on the access method to the host file.

## Reporting from Relational Databases

Reports against relational databases, such as DB2 and SQL/DS, can be pooled into the same cluster when they share several common attributes. Like non-relational files, all reports must access the same Master File Description from the same entry point. Reports that require SQL aggregation (that is, the generated SQL statements contain the `GROUP BY` phrase) are not pooled. This assures that the set presented to each report in the pool is accurate.

Further, all requests against a multi-table relational view must reference the same tables to be pooled into the same cluster. Consider, for example, a view that contains table 'A' and table 'B'. All reports that reference only fields from table 'A' can be pooled together; all reports that reference only fields from table 'B' can be pooled together, and all reports that reference fields from both tables 'A' and 'B' can be pooled together. However, none of the reports in each of these three preceding sets can be pooled with reports from another set. This limitation is imposed to assure that the same optimization logic is used by the RDBMS retrieval engine for each report in the set.

Pooling requirements for relational databases are less stringent when optimization is turned off (`SQL SET OPTIMIZATION OFF`). In this case, FOCUS manages the retrieval and aggregation. Therefore, pooling conditions are the same with optimization off as for non-relational databases. Restrictions regarding common accessed tables and SQL aggregation do not apply. The benefits of pooling reports with optimization off versus allowing the RDBMS to optimize retrieval vary from case to case. For example, a request that requires an area sweep and returns a large answer set, even with optimization, would be a good candidate to pool with other requests by turning optimization off.

When using the interface trace facility for a relational database, the generated SQL for each request is echoed. The SQL is generated during the Pooled Tables parsing phase but is not submitted to the RDBMS. Instead, Pooled Tables constructs an internal request to retrieve all of the data required for the cluster. The SQL SELECT statements generated for the cluster are echoed in the trace. These are the statements that are passed to the RDBMS.



The SQL SELECT statements generated by Pooled Tables are the ones optimized by the RDBMS. Therefore, the best optimization occurs when all requests in a cluster contain the same equality screening conditions or Filters (see [Common Selection Criteria](#)). In these cases, the screening tests are included in the SQL and passed to the RDBMS for optimization. Without common selections or Filters, it is possible that efficiencies gained from RDBMS optimization may be lost when pooling individual requests. For example, consider two requests: the first request returns a small answer set based on a selection against a key field named KEY1. The second request returns a small answer set based on a selection against a different key field named KEY2. The independent screening conditions are not included in the SQL generated by Pooled Tables, resulting in an area sweep and large answer set for the cluster. If the two tests are included as an OR condition in a Filter, the screening tests will be passed to the RDBMS. A much smaller answer set will be returned to Pooled Tables.

## **Pooled Tables in Batch Mode**

Pooled Tables can automatically pool all batch requests. For batch jobs to become pools, issue the SET command POOLBATCH, from either users PROFILE or in FOC Parm. Wherever possible, pooling automatically occurs. In the context of Pooled Tables, 'batch' means any non-interactive FOCUS session. In MVS, this occurs in batch jobs, or when ddname SYSIN is allocated to a dataset. In VM, a non-interactive job occurs when ddname SYSIN is FILEDEFed to a file, FOCUS is invoked with the syntax `FOCUS IN fileid`, or the VM session is running disconnected.

## Trace Facility

In general, the trace facility displays the reasons for segregation of a pool into subpools and clusters, warnings when allocating insufficient memory, and completion statistics for pooled reports. The purpose of the trace facility is to assist the application developer in determining how a pool was executed so that the information can be used in tuning the application.

The trace facility is started by issuing the command `SET TRACEON=POOLTABL`. By default, the trace output is routed to the ddname `PTTRACE` which is allocated to `SYSOUT` (MVS) or the terminal (VM). You can select a different ddname by issuing the command `SET TRACEON=POOLTABL//ddname`. To route the output to disk, allocate or `FILEDEF` ddname `PTTRACE` (or the optional ddname you selected) to a file with `LRECL 160` and disposition `MOD`. You may also allocate the ddname to the terminal. Stop the trace by issuing the command `SET TRACEOFF=POOLTABL`.

The following messages are displayed when a subpool boundary is encountered:

```
Subpool boundary--prior output required as input
Subpool boundary--FOCUS/SET command
Subpool boundary--DEFINE ADD
Subpool boundary--new MASTER name
Subpool boundary--new DEFINE clears pre-pool DEFINE
This command will run now, outside of pooling:
A DEFINE ADD will run now, outside of pooling.
```

The following messages are displayed when a cluster boundary is encountered:

```
Cluster boundary--new master name  
Cluster boundary--single-table cluster  
Cluster boundary--new alternate view  
Cluster boundary--new pool flag  
Cluster boundary--new pool condition  
Cluster boundary--mid-stream DEFINE  
Cluster boundary--new entry segment  
Cluster boundary--too many verb objects
```

The following messages are displayed for reports that cannot be pooled (they are single-table clusters):

Single-table cluster--REDEFINED real field  
Single-table cluster--User subroutine not known safe  
Single-table cluster--self-referential DBA/filter  
Single-table cluster--INCLUDES/EXCLUDES selection  
Single-table cluster--too many test literals  
Single-table cluster--complex test on index  
Single-table cluster--\$ORTPARM allocated  
Single-table cluster--REDEFINED constant real field  
Single-table cluster--RANKED BY  
Single-table cluster--COUNT DISTINCT  
Single-table cluster--RECAP  
Single-table cluster--COUNT is a verb object  
Single-table cluster--indexed view via AUTOINDEX  
Single-table cluster--EMR  
Single-table cluster--ON TABLE SET  
Single-table cluster--TEXT field  
Single-table cluster--PREVIEW mode  
Single-table cluster--ALL = ON/PASS  
Single-table cluster--per message above  
Single-table cluster--indexed view for FOCUS database  
Single-table cluster--non-poolable interface request  
Single-table cluster--too many verb objects

The following messages appear in the trace during the creation and execution of clusters and iterations:

```
Building cluster x...
Cluster contains n table(s)
Cluster n dedicated to command x
Clusters built; subpool contains x cluster(s).
***** Stack before 1st cluster: *****
***** Stack before nth cluster: *****
***** Begin union table *****
**** Stack before nth iteration: ****
```

During the parsing phase of the Pooled Table process, the following statistics are displayed for each report. They are used to determine if a report is poolable and under what conditions. All reports that have the same pooling criteria can be pooled with each other.

```
Entry Segment   : x
Relational Flag : y
Pool Flag       : z
Condition Length: n
Condition       : c
```

After a pooled report is executed, the output from ? STAT is included in the trace. The entries for TRACKIO and MINIO are included in the output but their values are not populated. In addition, the following statistics are included:

```
TRAVERSAL MTHD =          x          ENTRY SEGMENT =          i
FOCUS SORT MEM =         y1         EXTSORT MEMORY =         y2
ALGORITHM USED =          z
```

The following messages are displayed to indicate limitations imposed on Pooled Tables to execute reports under the most favorable conditions, based on parameters provided by the user (POOLMEMORY, POOLRESERVE, ESTRECORDS, and ESTLINES) or the available memory. These messages will not inhibit the execution of Pooled Tables. To correct these situations, replace the values for ESTRECORDS and ESTLINES with accurate values or make more memory available for Pooled Tables.

```
# concurrent external sorts reduced from x to y by below-16M shortage
Minimum sort memory forces iterations
Warning--POOLMEMORY desired = x but only y is available
Warning: actual line count (x) exceeds lines estimate (y) in heavy
aggregation case
Warning: records estimate (x) off by more than 10%-actual record count=y
Warning: lines estimate (x) off by more than 10%-actual line count = y
```

## Tuning Applications

Pooled Tables will always pool any application when POOL is set ON. Pooled Tables works best when accurate estimates for ESTRECORDS, ESTLINES, and POOLMEMORY are given for each request. These numbers can be determined by reviewing the statistics from previous runs. If these estimates are not provided, FOCUS uses the defaults: ESTRECORDS=100000, ESTLINES=0, and POOLMEMORY=16,384K. When ESTLINES is 0, Pooled Tables uses the current value of ESTRECORDS for ESTLINES. While these defaults are adequate for large extract reports, they may provide minimal benefit if they are grossly inaccurate.

To optimize pooling capacity, provide ample memory to Pooled Tables. Increase POOLMEMORY to an adequate size. Provide a sufficient region size (MVS) or virtual memory (VM). Reduce POOLRESERVE once interface and other modules are loaded. Furnish accurate estimates for ESTRECORDS and ESTLINES.

To optimize pooling capability, remove all unnecessary subpool boundary commands such as extraneous -RUNs. Consolidate the necessary boundary commands such as the DYNAMs, SETs, etc. Organize the requests for optimal cluster usage by putting all requests for the same database with the same entry point and retrieval method together. This will increase the opportunity for pooling more requests in one cluster.

To optimize retrieval and reduce the size of the answer set returned by Pooled Tables, use Filters to screen data. For example, if all reports in a cluster use `WHERE DELETE_FLAG NE 'Y'`, create a filter with this test. Alternately, change the test to read `WHERE DELETE_FLAG EQ 'N'` so that the common selection statement is used in the Pooled Tables common read.

## **Syntax**      **How to Use Pooled Tables**

To activate Pooled Tables, issue the following command

```
SET POOL = {OFF|ON}
```

where:

`OFF`

Ends Pooled Tables and executes any queued requests.

`ON`

Begins Pooled Tables.

Issue the following command in a report request to supply an estimate for the number of input records for that report:

```
ON TABLE SET ESTRECORDS m
```

where:

*m*

Is the estimate of the number of records being retrieved for a report.

You can assign a global value for each report in a pool with the following command

```
SET ESTRECORDS=m
```

The default value is 100,000.

Issue the following command in a report request to supply an estimate for the number of output lines for that report:

```
ON TABLE SET ESTLINES n
```

where:

*n*

Is the user's estimate of the number lines of output for a report.

You can assign a global value for each report in a pool with the following command

```
SET ESTLINES=n
```

The default value is 0. If no value is given, Pooled Tables assumes there is no aggregation and the number of lines is the same as the number of records.

To set a limit on the amount of memory that FOCUS can use for pooling a cluster for a user, issue the following command

```
SET POOLMEMORY=n
```



where:

*n*

Is the upper limit on the number of kilobytes of memory that FOCUS may use during any cluster for this user. In MVS, the number represents memory above the 16 megabyte line. In VM, the number represents total virtual memory.

The default value is 16,384 K (16 M). The minimum value is 1,024 K.

To reserve memory for other modules, issue the following command

```
SET POOLRESERVE =n
```

where:

*n*

Is the amount of memory in kilobytes to reserve for other modules and restrict Pooled Tables from using.

In VM, the default is 1,024K. In MVS, the default is 100K.

To set a limit on the number of concurrent external sorts that can run, issue the following command

```
SET MAXEXTSRTS=n
```

where:

*n*

Is the number of concurrent external sorts that can run.

Is a number from 1 to 26. The default is 26. In VM, only one version of SyncSort can run concurrently. If you use SyncSort in VM, the value of MAXEXTSRTS is assumed to be 1.

To control whether Pooled Tables is used automatically for batch processing, issue the following command

```
SET POOLBATCH = {OFF|ON}
```

where:

**OFF**

Does not enable automatic use of Pooled Tables for batch processing. This is the default.

POOLBATCH can be included in the FOCPARM ERRORS, FOCUS PROFILE, a FOCEXEC, or issued in the SYSIN input stream.

`SET POOLBATCH=ON` has the effect of automatically setting `POOL=ON` for batch execution. `SET POOLBATCH=OFF` will not reverse this setting. To disable pooling when `POOLBATCH=ON`, issue the command `SET POOL=OFF`.

**ON**

Enables automatic use of Pooled Tables for batch processing.

To identify an external sort utility to use for Pooled Tables, issue the following command

```
SET SORTLIB = sorttype
```

where:

*sorttype*

Can be one of the following

**SYNCSORT**

Identifies the external sort utility as SYNCSORT.

DFSORT	Identifies the external sort utility as DFSORT.
VMSORT	Identifies the external sort utility as VMSORT.
MVMSGSS	Identifies the external sort utility as SYNCSORT and its messages are displayed (MVS only).
MVMSGDF	Identifies the external sort utility as DFSORT and its messages are displayed (MVS only).

To direct the trace output, issue the following command

```
SET TRACEON=POOLTABL // {PTTRACE|ddname}
```

where:

PTTRACE

Is the default *ddname* where the trace output is directed.

*ddname*

Is an optional *ddname* where the trace output can be directed.

To turn off the trace facility, issue the following command

```
SET TRACEOFF=POOLTABL
```

where:

POOLTABL

Ends the Pooled Tables Trace facility.

## Pooled Tables Example

The following example illustrates the ease in which Pooled Tables can be implemented. In it, a small amount of memory is made available for Pooled Tables (4,000K), pooling is turned on, and report size estimates are provided for each report. The reports will be queued until pooling is turned off. At that time, data will be retrieved only once for all of the reports in the pool. The reports will be executed concurrently and the output printed one after the other.

```
SET POOLMEMORY = 4000
SET POOL = ON
TABLE FILE EMPLOYEE
PRINT LN FN BY DPT IF HIRE_DATE GE 860101
ON TABLE SET ESTLINES 1000 AND ESTRECORDS 1000
END
TABLE FILE EMPLOYEE
SUM CURR_SAL BY CURR_JOBCODE IF CURR_JOBCODE EQ 'A$*'
ON TABLE SET ESTLINES 5 AND ESTRECORDS 400
END
TABLE FILE EMPLOYEE
SUM GROSS BY PAY_DATE
IF PAY_DATE FROM 960101 TO 961231
ON TABLE SET ESTLINES 52 AND ESTRECORDS 1200
END
SET POOL = OFF
```

## Single TABLE Clusters

There are several instances when reports will not be pooled because of syntactical or environmental conditions. The reports will be executed as single TABLE clusters. Reports that fall into this category include:

- TABLEF requests.
- MATCH requests.
- Extended Matrix Reports (EMR).
- Reports using SET ALL=ON or PASS.
- Reports against FOCUS databases using an explicit indexed view or an implicit indexed view via AUTOINDEX.
- Reports against relational databases where aggregation is passed to the DBMS.
- Reports which use MORE, ON field RECAP, COUNT DISTINCT, DST., INCLUDES, EXCLUDES, or COUNT as a verb object.
- Reports that use a redefined database field.
- Reports issued from the FOCUS command line.
- Reports that use a self-referential Filter or DBA value restriction.
- Reports that have more than 256 values in an equality IF or WHERE test.
- Reports executed when \$ORTPARM is allocated.
- Reports that use a user written subroutine except those found in Table 1. In general, subroutines that require initialization and are then reused are not poolable. Random number generator subroutines are an example of these.

ARGLEN	ATODBL	AYM	AYMD	BAR
BITSON	BITVAL	BYTVAL	CHGDAT	CHKFMT
CHKPCK	CTRAN	CTRFLD	DADMY	DADYM
DAMDY	DAMYD	DAYDM	DAYMD	DMOD
DOWK	DOWKL	DTDMY	DTDYM	DTMDY
DTMYD	DTYDM	DTYMD	EXP	FEXERR
FINDMEM	FMOD	FTOA	GETPDS	GETTOK
GETUSER	GREGDT	HEXBYT	HHMSS	IMOD
ITONUM	ITOPACK	ITOZ	JULDAT	LCWORD
LJUST	LOCASE	OVLAY	PARAG	PCKOUT
POSIT	RJUST	SOUNDEX	SUBSTR	TODAY
UFMT	UPCASE	YM		

**Table 1. Poolable User Written Subroutines**

## Subpool Boundary Conditions

A subpool is a collection of TABLE or GRAPH requests and their related commands. Subpool boundaries are imposed by non-retrieval commands. Only reports within a subpool can be pooled together to share the same I/O. Commands that cause subpool boundaries can change the data or retrieval method for the database. Therefore, reports on either side of a subpool boundary cannot be pooled together reliably. When a subpool boundary command is encountered, pooling is temporarily stopped and all queued requests are executed.

A subpool boundary is created when:

- A FOCEXEC completes execution and control is returned to the command line.
- A -RUN or -EXIT command is issued in a FOCEXEC.
- A `DEFINE FILE filename ADD` command is issued.
- Any non-TABLE or GRAPH command is issued that could change the data (MAINTAIN, MODIFY, SQL), change the source of the data (DYNAM, USE), change the retrieval method (JOIN, PASS, FILTER), or change the operating environment (TSO, MVS, CMS), Table 2 lists the retrieval commands that are part of a subpool. Table 3 lists the commands that cause subpool boundaries.
- Any SET or ON TABLE SET command that can alter retrieval or the Pooled Tables environment. Table 4 lists the SET commands that cause subpool boundaries. SET commands that appear in ? SET ALL and not on this list will not cause a subpool boundary. This list is accurate for FOCUS release 7.0.8 and is subject to change in subsequent releases.

?	?F	?FF	CHECK	DEFINE
GRAPH	HELP	HOLD	OFFLINE	ONLINE
PCHOLD	REPLOT	RETYPE	SAVB	SAVE
TABLE	TABLEF			

**Table 2. Commands Included in a Subpool**

ACE	ANALYSE	CALC	CMS	COMBINE
COMPILE	CREATE	DECRYPT	DYNAM	ENCRYPT
EX	EXEC	FILETALK	FILTER	FIN
FINISH	FIXPACK	FS	FSCAN	GRAPHTALK
JOIN	LET	LOAD	MAINTAIN	MATCH
MODIFY	MODIFYTALK	MPAINT	MVS	PASS
PLOTTALK	REBUILD	RECALC	REMOTE	RESTRICT
RUN	SCAN	SET	SQL	TABLETALK
TED	TSO	UNLOAD	USE	WINDOW
XFER				



**Table 3. Commands That Cause Subpool Boundaries**

ADABAS	AGRRATIO	ALL.	AUTOINDEX
AUTOPATH	AUTOSTRATEGY	AUTOTABLEF	BINS
BLKCALC	BYPANEL 2	BYSCROLL	CACHE
CALC	CALCMEMORY	CALCROWS	CALCWAIT
CARTESIAN	CDN	COLUMNSCROLL2	COMMIT
COMPUTE	CONSULTOPTN	CURRENCY	DATETIME
DEFCENT	ESTLINES 1	ESTRECORDS 1	EXTSORT
FIELDNAME	FILENAME	FIXRETRIEVE	FOCSTACK
FOC144 1	HIPERFOCUS	HTMLMODE	ICUFORM
IMPLIEDLOAD	IMS	LABELPROMPT	LANGUAGE
LE370	LOADLIMIT	LOOKGRAPH	MAXLRECL
MAXPOOLMEM	MINIO	MODE XXXXXX	MPRINT
PASS	POOL	POOLBATCH	POOLFEATURE
POOLMEMORY	POOLRESERVE	PREFIX	PREVIEW
PRINTPLUS 2	QUALCHAR	RECORDLIMIT 1	SAVEMATRIX 2
SHIFT	SM	SQLENGINE	SQLTCARTES

SQLTOPTTF	STYLEMODE	SUSI	SUTABSIZE
TCPIPINT	TEMP DISK	TERMINAL	TEXTFIELD
TRACKIO	TRMSD	TRMSW	TRMTYP
USER	WINPFKEY	XRETRIEVAL	YRTHRESH
3DGRAPH			

**Table 4. SETs That Cause Subpool Boundaries**

- 1 - Subpool boundary with SET only
- 2 - Subpool boundary with ON TABLE SET only

**Pooled Tables Installation Instructions**

This section describes installation instructions for all systems, IMS, MVS, and VM/CMS.

## Procedure Installation Instructions for All Systems

Pooled Tables is enabled for your release of FOCUS by including the command

`SET POOLFEATURE = ON` in FOCPARM. To disable Pooled Tables, include the command

`SET POOLFEATURE = OFF` in the FOCPARM file. If there is no `SET POOLFEATURE` in FOCPARM, FOCUS assumes Pooled Tables is disabled.

The maximum amount of memory above 16 megabytes that can be requested by a user with the `SET POOLMEMORY` command can be restricted by including the

`SET MAXPOOLMEM = n` command in FOCPARM.

To make `POOL = ON` the default for all batch jobs, include the command `SET POOLBATCH = ON`. This must follow the `SET POOLFEATURE = ON` command in FOCPARM.

Each of the commands is also included in the member FOCPARM in ERRORS.DATA (MVS) or the file FOCPARM ERRORS (CMS).

## Procedure Installation Instructions for MVS

Include the POOLFEATURE, POOLBATCH, and MAXPOOLMEM commands in the member FOCPARM in ERRORS.DATA as outlined above. Refer to the MVS Installation Guide for FOCUS Release 7.0 (DN1000994.1097) or New Feature Memo 607, *TABLA Enhancements*, to change the default allocation for the file FOCPOOLT. If you use DFSort, refer to the section *"Usage Notes" for information about a required IBM PTF*.

## Procedure Installation Instructions for VM/CMS

Include the POOLFEATURE, POOLBATCH, and MAXPOOLMEM commands in the file FOCPARM ERRORS as outlined above. Change the value of POOLRESERVE in FOCPARM ERRORS if appropriate for your installation. See section *"Managing Memory" for recommended values*.

## Commands for the FOCPARM file

To configure Pooled Tables, include the following commands in the FOCPARM file.

```
SET POOLFEATURE = {OFF|ON}
```

where:

OFF

Disables Pooled Tables for this FOCUS site.

ON

Enables Pooled Tables for this FOCUS site.

```
SET POOLBATCH = {OFF|ON}
```

where:

OFF

Does not enable automatic use of Pooled Tables for batch processing. This is the default.

POOLBATCH can be included in the FOCPARM ERRORS, FOCUS PROFILE, a FOCEXEC, or issued in the SYSIN input stream.

SET POOLBATCH=ON has the effect of automatically setting POOL=ON for batch execution. SET POOLBATCH=OFF will not reverse this setting. To disable pooling when POOLBATCH=ON, issue the command SET POOL=OFF.

ON

Enables automatic use of Pooled Tables for batch processing.

SET MAXPOOLMEM = *n*

where:

*n*

Sets upper limit in Kilobytes of memory above 16 megabytes available for users to set in the SET POOLMEMORY command. The default is 32,768 K (32 M). Minimum is 1,024K.

## Reference Usage Notes

- With pooling, there may be differences in the order of output records in unsorted reports (PRINT with no BYs).
- In MVS batch jobs that have set MSG=ON, the TABLE request appears twice in the output.

- ? SET ALL has been enhanced to display the values of Pooled Tables settings. No Pooled Tables settings have been added to ? SET.
- If it is needed, adequate temporary disk space must be made available for FOCPOOLT under CMS.
- If it is needed, it is recommended that the user pre-allocate FOCPOOLT with the necessary space attributes under MVS. DCB information will be determined by FOCUS.
- KX following the attention interrupt is not supported during a pooled request.
- Pooled Tables memory in MVS is generally restricted using the POOLMEMORY setting. Pooled Tables memory in VM is generally restricted by using the POOLRESERVE setting.
- An implied `SET POOL=OFF` is issued and all queued requests are executed when an explicit or implicit FIN command is encountered and POOL is still ON.
- When SyncSort is the external sort package, the ddname \$ORTPARM must not be allocated. If \$ORTPARM is allocated, pooling is disabled for **all** requests, not only those that require the external sort. A warning message is issued when pooling is attempted and \$ORTPARM is allocated.
- DFSort Release 13.0 has a limitation where only 10 sorts can be run concurrently in MVS. If you exceed this limit, DFSort will display the message:

```
ICE149A DFSORT IS NOT LICENSED FOR USE ON THIS SYSTEM.RETURN CODE 12,  
REASON CODE 4.
```

This will cause FOCUS to abend. Issue the command SET MAXEXTSRTS = 10 to temporarily avoid this symptom. This problem has been fixed by IBM with APAR OW29152. Order IBM PTF UW41671 if you are running SMS Release 1.3. Order IBM PTF UW41672 if you are running SMS Release 1.4.

## Frequently Asked Questions

- *"How can I control how much memory Pooled Tables uses? I am afraid that users will abuse memory resources."*

The command SET POOLMEMORY=n, where n is in kilobytes, sets the upper bound of memory that FOCUS will use for pooling on a per user basis. The default is 16 Megabytes and the minimum is 1 Megabyte. A maximum limit per user can be established during Pooled Tables installation by including the command SET MAXPOOLMEM=n in the FOCPARM file. A user will not be able to request more memory than this limit.

- *"I already create a HOLD file from my database and then report from that. Why do I need Pooled Tables?"*

With Pooled Tables, it is possible to receive even better results than this technique, without any pre-planning on the developers part! Rather than create the HOLD file, Pooled Tables will read the data only once and pass it to each of the reports in the pool. There is no I/O to write the HOLD file and, more importantly, no I/O to read the HOLD file once for each of the reports in the pool. If you already use this technique, there is no immediate need to change your application to benefit from Pooled Tables. All of the reports from the HOLD file can be pooled. This will alleviate all but the I/O to create the HOLD file and one set of I/O to read the file once for all of the reports in the pool.

- *"I have reports in two separate FOCXECs. Will they be pooled?"*

Pooling will occur across FOCXECs, as long as there are no intermediate commands between them that would create a subpool boundary. The most common subpool boundary command that would be encountered in this situation is the use of -RUN or -EXIT in the first FOCXEC. Although this is good practice in the non-Pooled Tables case, it will cause reports that could benefit from Pooled Tables to be executed separately, as they are without Pooled Tables.

- *"What happens if I provide incorrect estimates for the number of records and lines in a pooled report?"*



Your reports will still execute and you will still receive the benefit of reduced database I/O and the CPU associated with it. However, the report sorting and formatting costs may increase. With incorrect estimates, Pooled Tables may select the wrong sort or allocate too little or too much memory for the report within the pool. If too much memory is allocated, fewer reports can be executed concurrently. If too little memory is allocated, fewer records can be sorted in memory, causing additional sort work I/O. If a FOCUS sort is selected, you lose the benefit an external sort when it is more appropriate. Note, however, that Pooled Tables will have an accurate count of the selected records (ESTRECORDS) for reports in the second and subsequent iterations.

## Reference Error Processing

- Any error detected during parsing causes pooled tables to flush the remaining commands through `SET POOL = OFF`.
- Any error detected during the retrieval phase will cause Pooled Tables to abort the FOCUS session. This occurrence will display both the error message that caused the problem and error message FOC1897. The following is an example of when this may occur:

```
(FOC1070) VALUE FOR JOIN 'FROM' FIELD OUT OF SEQUENCE. RETRIEVAL ENDED  
(FOC1897) FATAL ERROR DURING POOLED TABLES RETRIEVAL. FOCUS  
TERMINATING.
```

- Any error detected during the output phase for a given TABLE terminates processing for that TABLE. Output processing continues for subsequent TABLE requests.

- Line numbers and FOCEXEC name may be missing in the message:  
ERROR AT OR NEAR LINE n IN PROCEDURE a

## Reference Warning/Error Messages

FOCUS warning messages (normally generated without Pooled Tables) may appear twice when running with Pooled Tables active.

Below is a list of messages generated by Pooled Tables. These appear only when FOCUS is about to be terminated.

(FOC1897) FATAL ERROR DURING POOLED TABLES RETRIEVAL. FOCUS  
TERMINATING.

An error was encountered in a report during Pooled Tables retrieval. FOCUS cannot recover from this error and will return to the host environment.

(FOC1899) LOAD FAILED FOR EXTERNAL SORT

In preparation for an external sort under Pooled Tables, FOCUS tried to LOAD the external-sort module. The LOAD failed. (This message is produced only under VM.)

(FOC1900) NOT ENOUGH MEMORY FOR EXTERNAL SORT

In preparation for an external sort, available memory was queried. Not enough memory is available for an external sort. Increase memory and execute the request again.

## NF566: MSO/CICS Cooperative Processing

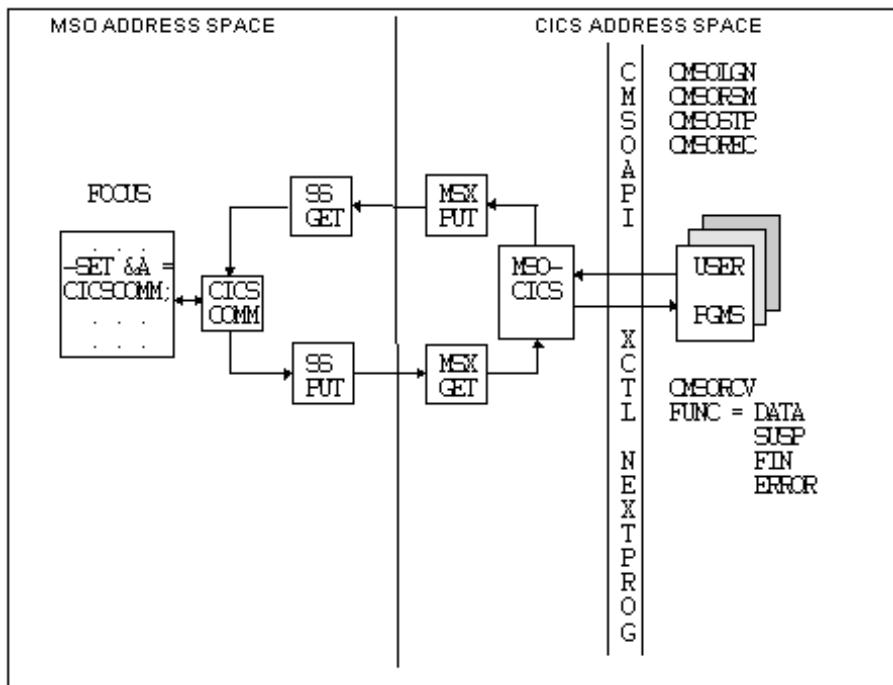
CICS transactions and MSO FOCXECs may now communicate directly with each other in a synchronous mode. A CICS pseudo conversational transaction may start an MSO session on behalf of a CICS user. Once a cooperative processing session is started, data can be handed back and forth between FOCXECs and CICS transactions in packets containing up to 256 bytes. Additionally, a suspend function is available. When this is invoked in MSO, the CICS transaction is given control. This allows a “hot” MSO to be available to the CICS user. The CICS user may go in and out of MSO without terminating their MSO session.

These new facilities allow CICS transactions to get and retrieve data from the MSO region and the reverse is also true. The FOCXEC interaction is implemented in MSO via a FUSELIB routine, CICSComm.

The CICS transaction functions are implemented by linking the installation's transaction module with the IBI supplied module, CMSOAPI. CMSOAPI provides five function calls that supply the MSO/CICS communications.

Also, a reconnect facility is available for cooperative processing sessions and for standard MSO connections from CICS. This allows the reconnection of an MSO session when the logical connection from a CICS terminal is lost.

The figure below illustrates the relationship between a FOCXEC running in the MSO address space and user programs running in the CICS address space when a cooperative session has been established.



## MSO FOCEXEC Cooperative Processing Service

The CICSCOM FUSELIB routine supplies the cooperative processing facility to FOCEXECs executing in the MSO region. It may be called from wherever a FOCUS user-written subroutine is supported. This routine is used to communicate and synchronize activity with the CICS portion of the dialog. When invoked, the FOCEXEC is placed into a wait until action is taken on the CICS side of the conversation. Data specified by the outlen/outbuf parameters is passed to CICS. When this subroutine is called it causes a CICS transaction specified in the CMSLOGN to be started in the CICS region. When CICSCOM completes, the inlen/inbuf parameters contain data that was passed back from CICS.

CICSCOM supports the following syntax:

```
CICSCOM(timeout, outlen, outbuf, inlen, inbuf);
```

where:

*timeout*

Is the number of seconds to wait for response before timing out FOCUS. If this timeout duration is reached before the CICS portion of the session responds, the MSO FOCUS session is terminated. This action is represented as a FIN function code to the CMSRCV call.

*outlen*

Is length (0-256) of the outbound message to CICS.

*outbuf*

Is the field containing the outbound message to CICS.

*inlen*

Is the length (0-256) of the inbound message buffer.

This value is the maximum amount that can be returned by CICS. It is presented to the CICS portion of the conversation in the *outlen* parameter of the CMSORCV call.

*inbuf*

Is the field to contain the inbound message from CICS.

## MSO/CICS Cooperative Processing Services

The CICS portion of the cooperative processing functions is provided in the IBI supplied module CMSOAPI. This module contains the functional code that supports the individual calls available to CICS transactions. When this module is link edited with a CICS transaction module, five functions become available to the module.

<i>CMSOLGN</i>	Logs a CICS user as an MSO FOCUS user and establishes a session between MSO FOCUS and CICS
<i>CMSRCV</i>	Interrogates MSO/CICS to identify the session that requires servicing, receiving data if sent.
<i>CMSREC</i>	Reconnects to a session that is in an indeterminant state.
<i>CMSOSTP</i>	Stops or terminates the session. This cancels the users MSO FOCUS task.
<i>CMSORSM</i>	Sends data back to the FOCUS portion of a session or restarts a suspended MSO FOCUS session.

Syntax and descriptions of the functions follow:

## **Syntax**      **How to Use the CMSOLGN Function**

This function starts an MSO session for the user by logging on to MSO. It defines the details of the MSO/CICS conversation processing that will take place.

```
CALL CMSOLGN(EIB, COMMAREA, MSONAME, USERWD, NEXTPROG, LOGONBUF)
```

where:

**EIB**

is the CICS Exec Interface control Block

**COMMAREA**

is the CICS Communications Area

**MSONAME**

4 byte name of MSO CICS transaction. Each MSO transaction name corresponds to a single MSO region that may be connected to.

**USERWD**

4 byte user word to associate with an MSO session

It is an arbitrary 4 byte value that is returned by CMSOLGN. It represents a unique identifier for the conversation that this service just established. It will be returned when the CMSRCV is issued to identify the specific conversation that needs to be serviced

**NEXTPROG**

8 byte name of next program to call

It identifies the CICS user program that will be called when the CMSCOMM FUSELIB is invoked on the MSO side of the conversation. NEXTPROG is mutually exclusive with the LOGTRAN entry in the LOGONBUF control block. Only one of these parameters should be specified. They function identically. Cooperative processing support, that is, the ability to be called back at all, is enabled by specifying a **NEXTPROG** in **CMSOLGN**. Fill **NEXTPROG** with blanks or nulls if no callbacks are desired.

**LOGONBUF**

Is a required control block. It should be completely initialized to blanks (x'40') before individual fields are set. Some of the fields support the MSO Load Balancing feature. The purpose of these fields are fully described in the new feature documentation for that feature.

An assembler copy file, CMSOAPIA, is supplied in MSO.DATA. This file maps the LOGONBUF. The fields and their meanings are listed in the table below. All fields are alphabetic.

<b>Field Name</b>	<b>length</b>	<b>contents</b>
<b>LOGAPPL</b>	8 bytes	A load balancing parameter that limits MSO region selection to those service groups that specify the same application name.



LOGSERV	8 bytes	Specifies the particular MSO Service Group that this user should be started in. It applies to load balancing as well as a single region MSO.
LOGTRAN	4 bytes	This defines the CICS transaction that will be invoked when the CICSCOMM FUSELIB routine is called in the MSO region. It is mutually exclusive with NEXTPROG in the invocation parameters. Only one should be specified.
LOGBREAK	4 bytes	Break key ( <i>PF/PAnn</i> ) Identifies the key that will unconditionally terminate the active MSO session.
LOGSUSP	4 bytes	Suspend key ( <i>PF/PAnn</i> ) Identifies the key that will cause the active MSO session to suspend operation. When this key is pressed, the LOGTRAN or NEXTPROG CICS transaction (whichever was specified) will be invoked in the CICS region.
LOGELVL	4 bytes	Error level ( <i>ALL, ERR, NONE</i> ) Controls what messages are displayed to the user when the MSO session ends. ALL - All messages are displayed ERR - Only error messages are displayed NONE - No messages are displayed

LOGGROUP	8 bytes	Logon group name The load balancing group name. It controls the MSO load balancing group that the user will be started in.
LOGVALID	1 byte	Security check flag (N/Y) Y must be specified for the reconnect service (CMSOREC call) to work. In addition, UNIQUE=LOGONID must be specified in the MSO configuration file.
LOGINITM	1 byte	Suppress initialization message flag (N/Y) Y suppresses the MSO initialization message.
LOGFLAG1	1 byte	reserved flag
LOGFLAG2	1 byte	reserved flag
LOGRESV	20 bytes	reserved
LOGACCT	40 bytes	account Specifies the MSO account field. This field will be recorded in the MSO SMF records when that feature is active.

LOGUPRM	256 bytes	Logon parameter The contents of this field is available to FOCEXECs via the MSOINFO subroutine. It is generally used to provide control information to the MSO profile exec so that specific FOCUS applications may be invoked in the MSO region.
---------	-----------	--

## Syntax      How to Use the CMSORCV Function

This function is used to interrogate the MSO/CICS control program. The returned parameters identify the particular conversation that had a status change, the current status of the conversation, and any data that may have been received from MSO. It is usually the first MSO/CICS Cooperative Processing service used in the transaction that is triggered by the CICSCOMM subroutine (NEXTPROG or LOGTRAN).

The `FUNCTION` field is set based on either an event in the MSO region or if an event in the CICS region caused the status. The `FUNCTION` codes that are returned are described in CMSORCV Function Codes.

```
CALL CMSORCV(EIB, COMMAREA, FUNCTION, CONNID, USERWD, INLEN, OUTLEN,
BUFFER, ERRNUM)
```

where

`EIB`

Is the CICS Exec Interface control Block

`COMMAREA`

Is the CICS Communications Area

FUNCTION

4 byte callback function code See CMSORCV Function Codes

CONNID

4 byte connect id of MSO session. This value together with USERWD uniquely define each MSO/CICS session.

USERWD

4 byte user word to associate with an MSO session. This value together with CONNID uniquely define each MSO/CICS session.

INLEN

4 byte length of buffer inbound from FOCUS

OUTLEN

4 byte length of return buffer expected by FOCUS

When using the CMSORSM, the `LENGTH` parameter may not exceed the value of `OUTLEN`. If it does, the data presented to the MSO FOCEXEC is truncated to the value of `OUTLEN`.

BUFFER

256 byte inbound data buffer

ERRNUM

4 byte FOCUS ending error number (function FIN) or 4 This field is a binary number and is mapped by the CMSOAPIA member of MSO.DATA. Possible returned values are described in section: CMSORCV Function Codes.

## **Syntax**      **How to Use the CMSOREC Function**

This function re-establishes a MSO/CICS session based upon the current user's CICS id. The connect id may be supplied if known. Otherwise, the function uses a connect id of 0 and the userid to identify the session. This condition may be caused by the user powering off their terminal while an MSO/CICS session is active and then logging on to CICS again.

Security flag - If the security flag in the logon buffer is set to yes, then the resuming or reconnecting userid is validated against the known userid. The reconnection is rejected if they do not match. If present userid determination is subject to the `MSCXUID` exit. The default for the security flag in the logon buffer is no.

```
CALL CMSOREC(EIB, COMMAREA, MSONAME, CONNID, NEXTPROG)
```

where:

`EIB`

is the CICS Exec Interface control Block

`COMMAREA`

Is the CICS Communications Area

`MSONAME`

4 byte name of MSO transaction

`CONNID`

4 byte connect id of MSO session

`NEXTPROG`

8 byte name of next program to call

## Syntax      How to Use the CMSOSTP Function

This function stops an MSO/CICS session immediately. MSO must have passed control to the CICSCOMM FUSELIB program before a stop can be issued. A CMSOSTP received while MSO is still in control is treated as a protocol error.

CALL CMSOSTP(EIB, COMMAREA, MSONAME, CONNID, NEXTPROG)

where:

EIB

Is the CICS Exec Interface control Block

COMMAREA

Is the CICS Communications Area

MSONAME

4 byte name of MSO transaction

CONNID

4 byte connect id of MSO session

NEXTPROG

8 byte name of next program to call. This field specifies a CICS transaction to start if the current invocation fails and cannot be associated with a known session. If a session is identified then the nextprog that was specified in the CMSOLGN service for the identified session will be called.

## Syntax      How to Use the CMSORSM Function

This function sends data to an MSO FOCUS session that previously issued the CICSCOMM FUSELIB routine or resumes an existing MSO/CICS session that was suspended by the user with the suspend key. Data may be sent in both cases but will be ignored if the session is in a suspended state. A `CMSORSM` received while MSO is still in control is a protocol error. If there is data, a buffer is allocated which is freed by MSOCICS.

```
CALL CMSORSM(EIB,COMMAREA,MSONAME,CONNID,NEXTPROG,LENGTH,BUFFER)
```

where:

`EIB`

Is the CICS Exec Interface control Block

`COMMAREA`

Is the CICS Communications Area

`MSONAME`

4 byte name of MSO transaction

`CONNID`

4 byte connect id of MSO session

`NEXTPROG`

8 byte name of next program to call

`LENGTH`

4 byte length of outbound buffer. Data sent to MSO FOCUS will be truncated to the original length specified by `INLEN` on the MSO FOCEXEC call to CICSCOMM.

BUFFER

Outbound data buffer (up to 256 bytes)

## CMSORCV Function Codes

The possible function codes that may be returned on the CMSORCV call are listed below:

Value name	value	Meaning	Returned Parameters
IB_DATA	9	Data returned This is the result of CICSCOMM being invoked in MSO.	CONNID USERWD INLEN OUTLEN BUFFER
IB_SUSP	10	suspend key struck The users MSO session is dormant until CMSORSM is issued for it.	CONNID USERWD
IB_FIN	11	FOCUS session ended ERRNUM contains return code from the MSO FOCUS session.	CONNID USERWD ERRNUM



IB_ERROR	12	API protocol error ERRNUM contains the value for the error. See the following table for a list of errors and their meanings.	CONNID USERWD ERRNUM
IB_NOAPI	17	Program not called by MSO\CICS	None

The ERRNUM values that may be associated with a function code of IB\_ERROR are:

Error	Value	Description
IBERR_STATE	1	MSO called in invalid state
IBERR_INVFUNC	2	MSO called w/invalid function
IBERR_SECURE	3	id verification failed.
IBERR_NORECON	4	RECON called but not supported
IBERR_NOTFOUND	5	user not found for connect id

## Examples

The following sample members are supplied in MSO.DATA to aid in developing installation applications to use this feature:

<a href="#">CCDEMO</a>	A FOCEXEC that implements a sample MSO menu to illustrate the function supplied by the CICS COMM FUSELIB routine
<a href="#">CCDEMOAS</a>	Assembler source of CICS API program
<a href="#">CCDEMOAJ</a>	JCL to build CCDEMOA module
<a href="#">CCMAPS</a>	map source for demo program
<a href="#">CCMAPSAJ</a>	JCL to build mapset and MAP DSECT

## Reconnection Capability

A CICS connection may be lost by powering off a terminal or closing an emulator session. This feature adds the capability of reconnecting to that session. Previously there was no way to do this and the user would not be allowed to re-logout until the original session had timed out or been canceled by the MSO operator.

Now, a standard MSO session may be reconnected with, by specifying MSO RECON (where MSO is the CICS transaction that invokes the MSOCICS program).

An MSO/CICS cooperative processing session, one that was established via the CMSOLGN call, may be reconnected using the CMSREC call. CMSREC allows the use of conid or userid to be specified. If conid is specified then the userid of the MSO FOCUS session has to match the CICS userid only if the LOGVALID security flag was set to Y in the LOGONBUF. The reconnection is found using the userid associated with the CICS session. UNIQUE=LOGONID must have been specified in the MSO configuration file for this service to be able to reconnect.

## Special Considerations

### Suspend key

If the suspend key is activated in the MSO FOCUS session, it is subject to the MSO configuration setting of IDLELIM. If the session is in the suspended state long enough to set off the IDLELIM limit, the session is terminated. This condition can be avoided by defining a separate service group that has IDLELIM set to a high value and placing the cooperative processing MSO sessions into these service groups by specifying the appropriate LOGSERV parameter.

### Previous API

The `CMSOLGN` is an alternate method of starting an MSO/CICS session. The original method was documented with the Load Balancing new feature of FOCUS 7.0.5 & 7.0.6. Logon according to those specifications is still supported. More information is available in New Feature Bulletin NF554.

## **Error Messages**

Not applicable

## NF568: DB2 Interface IF-THEN-ELSE Optimization

The DB2 interface has been enabled to improve the performance of FOCUS TABLE requests that include IF/THEN/ELSE define statements. Where applicable, the defined statements will be passed to DB2 as expressions allowing DB2 to optimize its own execution and minimize the size of the answer set returned to FOCUS.

### Usage

By issuing the new DB2 interface set command OPTIFTHENELSE, the interface will attempt to deliver as an expression to DB2 the construct of FOCUS IF/THEN/ELSE defines. The defined field must be an object of a selection test or an object of an aggregation request. The define definition may be specified in the table request or in the master file description.

### Syntax      How to Enable IF-THEN-ELSE Optimization

```
SQL {DB2} SET OPTIFTHENELSE {ON|OFF}
```

where

ON

Enables the feature

OFF

Disables the feature and is the default

**Note:** Omit the DB2 target RDBMS qualifier to issue the command if you previously issued the SET SQLENGINE command for DB2.

**Example Using IF-THEN-ELSE Optimization Without Aggregation**

```
SQL DB2 SET OPTIFTHENELSE ON
DEFINE FILE DB2TABLE
DEF1 = IF (NAME EQ ' ') AND (NAME EQ 'XYZ') AND (SSNO EQ ' ') THEN 1
ELSE 0;
END
TABLE FILE DB2TABLE
PRINT SSNO NAME
WHERE DEF1 EQ 1
END

>SELECT T1.SSNO,T1. NAME FROM Creator.table T1 WHERE
>      (((((T1.TOTAL_NAME = ' ') AND (T1.NAME = 'XYZ')) AND
>          (T1.SSNO = ' ')))) FOR FETCH ONLY;
```

**Example Using IF-THEN-ELSE Optimization With Aggregation**

```
DEFINE FILE DB2TABLE
DEF2 = IF NAME EQ 'NAME1' THEN 1 ELSE IF NAME EQ 'NAME2' THEN 2
ELSE IF NAME EQ 'NAME3' THEN 3 ELSE 0 ;
END
TABLE FILE DB2TABLE
WRITE MAX.NAME IF DEF2 EQ 1
END

>      SELECT  MAX(T1. NAME) FROM creator.table T1 WHERE
>          (((T1. NAME = 'NAME1')))) FOR FETCH ONLY;

TABLE FILE DB2TABLE
WRITE MAX.NAME IF DEF2 EQ 2
END

>      SELECT  MAX(T1. NAME) FROM creator.table T1 WHERE (((NOT
(T1. NAME = 'NAME1')) AND (T1. NAME = 'NAME2')))) FOR FETCH ONLY;
```

## Example Using IF-THEN-ELSE Optimization With Selection Criteria That Is Always False

```
DEFINE FILE DB2TABLE
DEF3=IF NAME EQ 'RITA' THEN 1 ELSE 0;
END

TABLE FILE DB2TABLE
PRINT NAME IF DEF3 EQ 2
END

> SELECT T1.NAME FROM creator.table T1 WHERE (1 = 0) FOR FETCH ONLY;
```

**Note:** Please note that DEF3 EQ 2 will never be true, thus the interface passes the where test 1=0 to DB2, denoting a not true condition and returning zero records from DB2.

## Reference Special Considerations

This new feature is enabled for SELECT statements only created as a result of FOCUS TABLE requests.

The following features are not supported:

- Decode defines
- Self-referencing or “recursive” defines
- STATIC SQL requests
- IF/WHERE DDname defines
- Partial Date selection

There is no guarantee that the SQL that is generated will improve performance for all requests. If it's found that this feature does not improve performance, OPTIFTEELSE OFF will disable this feature to initiate previous behavior.

## **Error Messages**

None.



## NF571: DB2 Interface SET ISOLATION Command

Starting with FOCUS release 7.0.8, the interface has been enabled to take advantage of the DB2 version 4 (and higher) ability to pass an SQL statement isolation level. This allows the interface to modify the isolation level for a TABLE request with the new interface SET command. The SET ISOLATION command will override the isolation level that was established at interface installation bind time. The new setting removes the requirement of binding multiple DB2 plans, each bound with different isolation levels.

### Usage

DB2 protects data being read by one user from changes (INSERT, UPDATE, or DELETE) made by other users; the isolation level setting governs the duration of the protection. The isolation level determines when shared locks on rows or data pages are released, so that rows or pages become available for updates by other users. DB2 version 4 and above allows this isolation setting to be changed at the SQL statement level. Besides the earlier DB2 supported isolation levels of CS and RR, DB2 version 4 and above has introduced the uncommitted read isolation level (UR) and in DB2 5.1, introduced the read stability isolation level (RS) and the isolation level of NC.

Please refer to the *DB2 Read/Write Interface Users Manual* for a more detailed definition of isolation level and to the appropriate DB2 manuals for a complete description of CS, RR, UR, RS and NC isolation levels and their impact on concurrency and database implications.

## Syntax How to Set the DB2 Isolation Level

```
SQL {DB2} SET ISOLATION level
```

where:

*level*

Can be one of the following DB2 isolation levels:

RR

UR

RS

NC

Omit the DB2 target RDBMS qualifier to issue the command if you previously issued the SET SQLENGINE command for DB2.

To display the isolation level setting, issue the command SQL DB2 ?.

Issue the command SQL DB2 SET ISOLATION (without an isolation level) to revert back to the bound isolation level of the interface plan.

### Example Setting the DB2 Isolation Level

This example shows the SQL passed to DB2 as a result of the SET ISOLATION command:

```
SQL SET ISOLATION UR
SELECT T1.field1 FROM "creator"."table1" T1 FOR FETCH ONLY WITH UR;
SQL SET ISOLATION CS
SELECT T1.field1 FROM "creator"."table1" T1 FOR FETCH ONLY WITH CS;
```

The following is a result of the SQL DB2 ? command:

```
(FOC1424) ISOLATION LEVEL FOR TABLE REQUEST IS : CS
```

To reset the isolation to the Interface default:

```
SQL SET ISOLATION (blank)
SELECT T1.field1 FROM "creator"."table1" T1 FOR FETCH ONLY;
```

The following is a result of the SQL DB2 ? command:

```
(FOC1424) ISOLATION LEVEL FOR TABLE REQUEST IS :
```

The blank isolation level denotes the default isolation level will be used.

### Reference Special Considerations

This new feature is enabled for SELECT requests only created as a result of FOCUS TABLE requests.

This new setting cannot be used to override the required isolation level of RR for FOCUS MODIFY requests, COPY MANAGER or CACTUS/MAINTAIN applications.

RS isolation level is only available for DB2 version 5.1 (4.2).

The interface does not validate the isolation level values, so be sure that they are one of the acceptable isolation levels for the version of DB2 that is being accessed, otherwise an SQL code of -104 will occur signifying a SQL syntax error.

### Error Messages

None.

## NF572: Invisible Ordered Character and Ordered Numeric Data Type Key Support

This new feature will allow access and selection of Model 204 invisible ordered character and invisible ordered numeric fields. The interface supports these key designations with comparable abbreviations called *suffix operators*. These operators are described with the TYPE attribute in the Access File Description.

### Usage

A TYPE=IOA in the Access File will designate a Model 204 Key of invisible ordered character. A TYPE=ION would denote an invisible ordered numeric field.

Any FOCUS or EDA selection request against a field with TYPE=IOA or ION will result in a Model 204 IFFIND command with the appropriate IFFIND specification.

### Example Using TYPE=IOA and TYPE=ION to Produce an IFFIND Specification

For example, an equality test produces the following IFFIND specification:

Suffix Operator	Model 204 KEY Type	IFFIND Specification
IOA	Ordered Character, invisible	IS ALPHA

ION	Ordered Numeric, invisible	IS NUM
-----	----------------------------	--------

A range test using LT or GT produces the following IFFIND specification:

Suffix Operator	Model 204 KEY Type	IFFIND Specification
IOA	Ordered Character, invisible	IS ALPHA BEFORE/AFTER
ION	Ordered Numeric, invisible	IS NUM LT/GT

A range test using FROM-TO produces the following IFFIND specification:

Suffix Operator	Model 204 KEY Type	IFFIND Specification
IOA	Ordered Character, invisible	IS ALPHA BEFORE/AFTER
ION	Ordered Numeric, invisible	IS NUM LT/GT

## Special Considerations

For a complete list and chart of all Model 204 KEY types and their corresponding IFFIND specifications, please refer to the *Model 204 Interface Users Manual*.

## Error Messages

N/A

## NF574: System 2000 Interface Trace Facility

Two new trace levels have been enabled. The traces can be used for informational or debugging purposes. You can access the SYSTEM 2000 database two ways with FOCUS or EDA: Selective or Sequential. FSTRACE will display all SYSTEM 2000 calls made by the Interface. If the selective strategy has been used, then FSTRACE1 can be used to display the SYSTEM 2000 LOCATE command with the associated parameters.

### Usage

You can store the trace information in an MVS sequential data set or CMS file, to SYSOUT in a batch job, or you can display it online at the terminal.

### Syntax      How to Invoke the System 2000 Interface Trace Facility

For Online to the screen (do not use for EDA or MSO):

```
DYNAM ALLOC F(FSTRACE) DA(*)
```

To write the trace to a file, use the appropriate allocation:

```
MVS ALLOC F(FSTRACE) DA('userid.FSTRACE') SHR REUSE LRECL(80) RECFM(F)
TSO ALLOC F(FSTRACE) DA('userid.FSTRACE') SHR REUSE LRECL(80) RECFM(F)
DYNAM ALLOC FILE FSTRACE DATASET userid.FSTRACE SHR REUSE LRECL 80 RECFM
F
```

For a batch job, to write the trace information to SYSOUT:

```
//FSTRACE DD SYSOUT=*,DCB=(LRECL=*,BLKSIZE=80,RECFM=F)
```

For CMS:

```
CMS FILEDEF FSTRACE DISK FSTRACE DATA A (LRECL 80 RECFM F
```

**Note:** Depending on the trace desired, specify FSTRACE or FSTRACE1. You must specify the MOD parameter in order to produce a complete trace listing. The FSTRACE and FSTRACE1 data sets are opened and closed for each SYSTEM 2000 request. Without the MOD disposition parameter, requests that produce multiple database accesses store only the last statement in the dataset.

## Reference Special Consideration

FSTRACE1 will contain less information than FSTRACE since it is restricted to SYSTEM 2000 LOCATE calls. It should contain enough information to determine interface to database communication and/or possible causes of performance degradation. When utilizing FSTRACE, it is recommended to use a RECORDLIMIT of 1 to begin with and increase this limit when needed since the FSTRACE file produced can be quite voluminous.

## Error Messages

N/A



## NF579: Assigning Screening Conditions to a File for Reporting Purposes

A filtering mechanism that assigns screening conditions to a file has been added to the functionality of TABLE. This enables you to declare a set of screening conditions, and assign it to a specific file, instead of constantly having to rewrite these screening conditions every time you need them.

The following example illustrates the use of filters. Both sides yield the same results.

### Without filters

```
TABLE FILE CAR...  
WHERE...SEATS GT 5  
  
TABLE FILE CAR...  
WHERE SEATS GT 5...  
  
TABLE FILE CAR...  
WHERE SEATS GT 5
```

### With Filters

```
FILTER FILE CAR...  
SET FILTER= WHERE SEATS GT 5...  
  
TABLE FILE CAR...  
TABLE FILE CAR...  
TABLE FILE CAR...  
TABLE FILE CAR...
```

Whenever a TABLE request is made against a file, all filters that have been activated for that file are in effect. A filter is a packet of definitions that resides at the file level, containing IF and/or WHERE statements. Once these conditions have been declared, you may deactivate and reactivate them as needed for your reporting purposes.

## Using Filters

A filter is an IF or WHERE statement that is automatically added to every TABLE against that file as if the IF or WHERE were actually coded by the user. All IF/WHERE syntax that is valid in a TABLE is valid in a filter. A filter can be declared at any time before the TABLE request, and remains in effect after the TABLE request has been executed. There can be one or more filters declared for a file.

Just declaring a filter for a file does not make it active. A filter must be activated with a SET command to be in effect.

Filters allow you to:

- Declare a common set of screening conditions that apply to all extracts from a file.
- Declare a set of screening conditions and dynamically turn them on and off.
- Reduce repetitive ad hoc typing.
- Implement DBA capabilities that are not tied to the Master File Description.

### **Syntax**      **How to Declare a Filter**

A filter can be described by the following declaration:

```
FILTER FILE filename [CLEAR/ADD]
  [filter-defines;]
  NAME=filtername1 [,DESC=text]
  if-where-statements
  ...
  NAME=filternamen [,DESC=text]
  if-where-statements
END
```

where:

*filename*

Is the name of master to be used in subsequent TABLE commands

*filter-defines*

Are virtual fields declared for use in filters. For more information, see [Filter Defines](#).

NAME

Identifies the start of the declaration of a new filter

*filtername*

Is the name by which the filter is referenced in subsequent SET FILTER commands. Filtername may be up to 8 characters in length and must be unique for a particular filename.

CLEAR

Releases any previously declared filters.

ADD

Enables you to specify additional filters without releasing existing ones.

DESC

Describes the filter. Text must fit on one line for documentation purposes.

END

Terminates the filter

*if-where-statements*

Are screening conditions that can include all valid syntax. May not refer to defined fields declared via DEFINE FILE. May refer to Database fields, defines in the Master. May not refer to other filternames.

## Reference Filter Defines

- Are exclusively local to (usable by) filters in the filter block.
- Are not referenceable by DEFINE FILE DEFINES or TABLE.
- Support any syntax valid for DEFINES in DEFINE FILE.
- Cannot reference DEFINES from DEFINE FILE but can reference DEFINES in the Master File Description.
- Do not count toward the 256 verb object limit of TABLE unlike DEFINES from DEFINE FILE when referenced explicitly or implicitly.
- Must all be declared before the first named filter.
- Must each end with a semi-colon
- Cannot be enclosed between DEFINE FILE/END commands.

## **Example** Using Filters

The following example replaces the filter UK, with a new WHERE condition. It also adds to the CAR file's set of filter-defines, a definition for "MARK\_UP". When the TABLE command is issued for CAR, and UK is activated, the condition WHERE MARK\_UP is greater than 1000 is automatically added to the TABLE request.

Note: The field MARK\_UP cannot be explicitly displayed or referenced in the TABLE..

```
FILTER FILE CAR ADD
MARK_UP/A16=RCOST-DCOST;
NAME=UK
WHERE MARKUP GT 1000
END
TABLE FILE CAR
PRINT
```

The following example declares three named filters for the CAR file; ASIA, UK, and LUXURY. The filter ASIA is given a textual description, for documentation purposes only. The CLEAR on the first line erases any named filters that had existed for CAR, as well any filter DEFINES for CAR, before it processes the new definitions.

```
FILTER FILE CAR CLEAR
NAME=ASIA,DESC=Asian cars only
IF COUNTRY EQ JAPAN
NAME=UK
IF COUNTRY EQ ENGLAND
NAME=LUXURY
IF RETAIL_COST GT 50000
END
```

## Syntax How to Activate/Deactivate Filters

Filters can be activated and deactivated with the following SET command:

```
SET FILTER {*|xx[ yy zz]} IN file {ON|OFF}
```

where:

\*

Denotes all declared filters (default)

ON

Activates the filter.

OFF

Deactivates the filter (default).

xx

Is the name of a filter as declared in the NAME = syntax of the FILTER FILE block

The following is an example of filter activation and deactivation:

```
SET FILTER = UK LUXURY IN CAR ON
...
TABLE FILE CAR
PRINT COUNTRY NAME MODEL RETAIL_COST
END
...
SET FILTER = LUXURY IN CAR OFF
TABLE FILE CAR
PRINT COUNTRY NAME MODEL RETAIL_COST
END
```

The first SET FILTER activates CAR's filters, UK and LUXURY, and applies the conditions their filters contain to any subsequent TABLE request of CAR. The second SET FILTER, deactivates the filter named LUXURY of CAR. Any subsequent TABLE request (unless LUXURY is activated again) of CAR will not apply the conditions in LUXURY but continues to apply UK.

## Syntax Filter Query

In order to find out the status of any existing filters, use the following syntax:

```
? FILTER [{file|*}] [SET] [ALL]
```

where:

*file*

Is the name of a database master

\*

Displays filters for all files that have filters declared

SET

Displays only active filters

ALL

Displays all information about the filter including its description and the exact IF/WHERE definition.

## Example Querying Filters

The following is an example of querying filters:

```
? FILTER
```

```
NO FILTERS DEFINED
```

OR

Set	File	Filter name	Description
	CAR	ROB	Rob's selections
*	CAR	PETER	Peter's selections for CAR
*	EMPLOYEE	DAVE	Dave's tests
	EMPLOYEE	BRAD	Brad's tests

? FILTER CAR

NO FILTERS DEFINED FOR FILE NAMED CAR

OR

Set	File	Filter name	Description
	CAR	ROB	Rob's selections
*	CAR	PETER	Peter's selections for CAR

? FILTER \* SET

Set	File	Filter name	Description
*	CAR	PETER	Peter's selections for CAR
*	EMPLOYEE	DAVE	Dave's tests



## **Filters and JOINS**

Filters against a file are suspended (but not erased) when that file is the object of a JOIN. Filters against the primary file of a JOIN may be re-declared or be entirely different, may reference only fields in the JOINed structure and are in effect until the JOIN is cleared. At that time the pre-JOIN filters are brought back. Filters against the secondary JOIN file(s) remain alive as originally declared.

```
*****
-* JOIN AND FILTER INTERACTION
*****

-* DECLARE A FILTER
FILTER FILE EMPLOYEE CLEAR
  NAME=XXX WHERE JOBCODE EQ 'A01'
END
SET FILTER = XXX IN EMPLOYEE ON
-* EMPLOYEE FILE SHOWS JOBCODE A01 ONLY
TABLE FILE EMPLOYEE PRINT EMP_ID JOBCODE
END
-* -----
-
-* NOW JOIN TO JOBFILE AND REDECLARE THE SAME FILTER TO A DIFFERENT VALUE
-* -----
-
JOIN JOBCODE IN EMPLOYEE TO JOBCODE IN JOBFILE
FILTER FILE EMPLOYEE
  NAME=XXX WHERE JOBCODE EQ 'A07'
END
-* (NOTE: NEW FILTER FOR JOIN STRUCTURE IS NOT ACTIVATED YET)
-* EMPLOYEE FILE SHOWS **ALL** JOBCODES (ORIGINAL FILTER TURNED OFF BY
JOIN)
TABLE FILE EMPLOYEE PRINT EMP_ID JOBCODE
END
-* -----
-* NOW TURN ON THE NEW FILTER THAT APPLIES TO THE JOIN STRUCTURE
-* -----
-
SET FILTER = XXX IN EMPLOYEE ON
-* SHOWS JOBCODE A07 (NOT A01) (NEW FILTER APPLIES TO JOIN ONLY)
TABLE FILE EMPLOYEE PRINT EMP_ID JOBCODE
```

```
END
-* NOW CLEAR THE JOIN TO RE-ESTABLISH THE ORIGINAL FILTER
JOIN CLEAR *
-* NOW SHOWS JOBCODE A01 ONLY, AS BEFORE (ORIGINAL FILTER REACTIVATED)
TABLE FILE EMPLOYEE PRINT EMP_ID JOBCODE
END
```

## Reference Special Consideration

The maximum number of filters set 'ON' for a file is limited by the number of IF/ WHERE statements in these filters, not to exceed the standard FOCUS limit of IF/WHERE statements in any single TABLE.

The SET FILTER command is limited to one line. To activate more filters than fit on one line, repeat the SET FILTER command. As long as you specify 'ON' the effect is additive, not one of replacement. For example,

```
SET FILTER A B C IN CAR ON
SET FILTER D E F IN CAR ON
SET FILTER G IN CAR OFF
```

activates A, B, C, D, E, F and deactivates G (assuming that it was set ON previously).

## Reference Error Messages

```
(FOC 36237) SYNTAX ERROR SETTING FILTER
(FOC36241) FILTERS DON'T EXIST FOR FILE NAMED:
(FOC36242) FILTER DOESN'T EXIST
```

## NF583: Teradata Outer Join Optimization

This feature improves the Teradata Relational Interface performance by enabling the interface to deliver better optimized SQL to the Teradata RDBMS, permitting the RDBMS to optimize its own join processing.

### Usage

The interface now passes left outer joins to Teradata when you issue the FOCUS command

SET ALL=ON (**Note:** SET ALL=ON previously disabled optimization, leaving FOCUS to handle the join).

Optimization of outer joins is available for the interface installed with Teradata TOS version 1 release 5.1 and higher or Teradata version 2 release 2 and higher. The Teradata version is specified by the REL= parameter of the Teradata Installation procedure GENFDBC.

### Syntax      How to Invoke Teradata Outer Join Optimization

```
SET ALL=ON
```

```
SQL SET OPTIMIZATION ON
```

### Example      Invoking Teradata Outer Join Optimization

This example shows the SQL passed to Teradata for a FOCUS dynamic join with SET ALL=ON, SQL SET OPTIMIZATION ON:

```
JOIN field1 IN file1 TO field2 IN file25
```

```
SELECT T1.field1,T2.field2 FROM "creator.table1" T1 LEFT OUTER JOIN  
"creator.table2" T2 ON T2.field2 = T1.field1 FOR FETCH ONLY;
```

### Special Considerations

- The SET ALL=ON command also controls processing of short paths. The interface default setting is OFF. For a complete description of short path processing, please refer to your Interface Users Guide.
- Please be aware that in passing requests to Teradata with SET ALL=ON, you may get correct, yet subtly different, sequences of report rows than you had with earlier releases of the Interface. These differences are due to differences between sorting algorithms used by FOCUS and by Teradata, and do not indicate upward compatibility problems.

### Error Messages

N/A

## NF586: Expanding Byte Precision for COUNT and LIST

The COUNT and LIST verbs may now optionally be expanded from 5 to 9 characters on display. This internally reformats COUNT and LIST from I5 to I9.

### Usage

Before FOCUS Release 7.0.8, if the number of records retrieved for a field exceeded 5 bytes, asterisks were displayed in the report. This indicates an overflow condition, meaning that the display must be increased. The new maximum value for COUNT and LIST is 999,999,999.

### Syntax      How to Set the Precision for COUNT and LIST

`SET COUNTWIDTH = ON/OFF`

where OFF is the default.

### Example      Setting Precision for COUNT and LIST

The following example shows the COUNT verb with behavior prior to FOCUS Release 7.0.8:

```
TABLE FILE filename
COUNT Fldxx
BY Fldyy
END
```

```
          FLDxx  
Fldyy  COUNT  
value  *****
```

The following example shows the COUNT verb with behavior as of FOCUS Release 7.0.8 with SET COUNTWIDTH = ON:

```
TABLE FILE filename  
COUNT Fldxx  
BY Fldyy  
END
```

```
          FLDxx  
Fldyy  COUNT  
value  999999999
```

## Special Considerations

This feature will affect the width of a report when the COUNTWIDTH is set to ON. Calculating the LRECL of a report will now require an additional 4 bytes for each COUNT and LIST column.

## Error Messages

None.

## NF593: IUCV CMS SU

Inter-user communications vehicle is now supported for CMS SU. This is a desirable protocol as the master processor is not enqueued serially per request. IUCV also offers performance gains when compared to the VMCF communications protocol.

IUCV is not supported for any release prior to R7.0.8. If any earlier release of FOCUS is used with an IUCV server the results are unpredictable. Please see the Simultaneous Usage Reference Manual, CMS Version (DN 1000015.0797)



## NF594: JAVA Report Assist

Java Report Assist provides a user-friendly environment for creating ad hoc reports in HTML, WP, DF or Lotus formats. The Report Assistant supports automatic generation of complete record selection criteria, sort fields, headings and footings, subtotals, and calculations.

Complete documentation for the Web Interface product can be found in the Web Interface User's Manual and Installation Guide Release 7.0.8 (DN1001038.1097).

## NF605: Date Handling for the Year 2000 in FOCUS

As part of our year 2000 compliance effort we have changed the default date format display in FOCUS. The new format is MMDDCCYY.

### Usage

The two digit century, has been added to the year portion of the display. This applies to all areas within FOCUS that display a date. They include:

- The FOCUS Banner
- ? REL
- ? FILE
- ? FDT
- MODIFY FILE FN
- CREATE FILE FN
- FSCAN FILE FN
- REBUILD TIMESTAMP

A 4 digit year is written into the FOCUS file in the format of CCYY. Prior to Release 7.0.8, YY was written into page 1 of the FOCUS file.

### Example Displaying Four-digit Years in FOCUS

Entering ? REL at the FOCUS prompt displays a screen similar to the following:

>

? REL

FOCUS 7.0.8

CREATED 11/20/1997

## Date Literals Interpretation Table

This table illustrates the behavior of FOCUS date formats. The columns indicate the number of input digits for a date format. The rows indicate the usage or format of the field. The intersection of row and column describes the result of input and format.

	1	2	3	4
YYMD	*	*	CC00/0m/dd	CC00/mm/dd
MDYY	*	*	*	*
DMYY	*	*	*	*
YMD	*	*	CC00/0m/dd	CC00/mm/dd
MDY	*	*	*	*
DMY	*	*	*	*
YYM	CC00/0m	CC00/mm	CC0y/mm	CCyy/mm
MYY	*	*	*	*
YM	CC00/0m	CC00/mm	CC0y/mm	CCyy/mm
MY	*	*	0m/CCyy	mm/CCyy

M	0m	mm	*	*
YYQ	CC00/q	CC0y/q	CCyy/q	0yyy/q
QYY	*	*	q/CCyy	*
YQ	CC00/q	CC0y/q	CCyy/q	0yyy/q
QY	*	*	q/CCyy	*
Q	q	*	*	*
JUL	CC00/00d	CC00/0dd	CC00/ddd	CC0y/ddd
YY	000y	00yy	0yyy	yyyy
Y	0y	yy	*	*
D	0d	dd	*	*
W	w	*	*	*

	5	6	7	8
YYMD	CC0y/mm/dd	CCyy/mm/dd	0yyy/mm/dd	yyyy/mm/dd
MDYY	0m/dd/CCyy	mm/dd/Ccyy	0m/dd/yyyy	mm/dd/yyyy

DMYY	0d/mm/CCyy	dd/mm/Ccyy	0d/mm/yyyy	dd/mm/yyyy
YMD	CC0y/mm/dd	CCyy/mm/dd	0yyy/mm/dd	yyyy/mm/dd
MDY	0m/dd/CCyy	mm/dd/Ccyy	0m/dd/yyyy	mm/dd/yyyy
DMY	0d/mm/CCyy	dd/mm/Ccyy	0d/mm/yyyy	dd/mm/yyyy
YYM	0yyy/mm	yyyy/mm	*	*
MYY	0m/yyyy	mm/yyyy	*	*
YM	0yyy/mm	yyyy/mm	*	*
MY	0m/yyyy	mm/yyyy	*	*
M	*	*	*	*
YYQ	yyyy/q	*	*	*
QYY	q/yyyy	*	*	*
YQ	yyyy/q	*	*	*
QY	q/yyyy	*	*	*
Q	*	*	*	*
JUL	CCyy/ddd	*	*	*
YY	*	*	*	*

Y	*	*	*	*
D	*	*	*	*
W	*	*	*	*

- CC stands for two century digits provided by DFC/YRT settings.
- \* stands for error message FOC177 (invalid date constant).
- FOCUS reads date literals from right to left.

### Special Considerations

N/A

### Error Messages

(FOC177)      **INVALID DATE CONSTANT:**

The constant in the calculation is not a valid date. Enter a valid date.

## NF607: TABLA Enhancements

### (Default Space Allocation Table for Work Files)

FOCUS output datasets not allocated by the user are allocated dynamically by FOCUS itself. The default space attributes associated with each dynamically allocated ddname are now set by editing the member IBITABLA in the PDS called FOCCTL.DATA.

### Usage

In order to change the defaults for these FOCUS output datasets, the file IBITABLA must be copied to a dataset allocated to the DDname ERRORS. The file is a fixed columnar file. All changes must be made in the appropriate column. The columns are:

<b>Column name</b>	<b>Starting column</b>	<b>length</b>
DDname	01	8
Allocation units ( <u>CYLS</u> ,TRKS)	10	4
Primary space	15	3
Secondary space	19	3
Number of Directory entries (PDS)	23	2

Sysout class (OFFLINE only)	26	1
Volume	28	6
Unit (SYSDA,DASD,HIPER,NOHIPER*,etc)	35	8
Unit Count (FOCPOOLT only)	44	2

- \* Use NOHIPER as a unit name to exclude particular datasets from HiperFOCUS.

If the file IBITABLA is not available in a PDS allocated to the ddname ERRORS, the defaults are in affect for all FOCUS output datasets. The default for all FOCUS output datasets except FOCPOOLT is 5 CYLs with secondary extent size of 5. The default for FOCPOOLT is 5 CYLs with secondary extent size of 20.

### **Example**    **Sample IBITABLA**

This is a copy of IBITABLA as shipped.



```

* DDNAME*A.UN*SP1*SP2*DR*C*VOLUME* UNIT *UC* FIELD NAME
*0-----1---1---1---2--2-2-----3-----4-* STARTING
*1-----0---5---9---3--6-8-----5-----4-* COLUMN
*---8-----4-- -3- -3- 2- 1 --6--- ---8--- 2-* LENGTH
HOLD CYLS 5 5 , /* 1 */
HOLDMAST TRKS 5 5 36 , /* 2 */
SAVE CYLS 5 5 , /* 3 */
REBUILD CYLS 5 5 , /* 4 */
FOCSML CYLS 5 5 , /* 5 */
FOCUS CYLS 5 5 , /* 6 */
FOCSTACK TRKS 5 5 , /* 7 */
FOCSORT CYLS 5 5 , /* 8 */
OFFLINE CYLS 5 5 A , /* 9 */
SESSION TRKS 5 5 , /*10 */
FOCCOMP TRKS 5 5 12 , /*11 */
HOLDACC TRKS 5 5 12 , /*12 */
FMU TRKS 5 5 12 , /*13 */
TRF TRKS 5 5 12 , /*14 */
FOCPOOLT CYLS 5 20 2, /*15 */

```

\*  
\*

\* The UC or unit count column may be specified for FOCPOOLT only.

\*\*\*\*\* Bottom of Data  
\*\*\*\*\*

## Special Consideration

This New Feature document supersedes Section 3.14 'Default Space Allocation Table for Work Files: TABLA' in the *FOCUS MVS/TSO Installation Guide* (DN1000994.0295, DN1000994.0896, or DN1000994.1097).\_

## **Error Messages**

If the old method is used and the installer attempts to link TABLA into module FOCUS, there is now a U593 abend at FOCUS initialization.

## NF609: Sink Validation of Userids in CMS

A new file has been created in CMS to help verify who may connect to a specific CMS sinkid.

### Usage

A filename called FOCSUACC, with FILETYPE = DATA and FILEMODE = A1, can be created on the A disk of the CMS sink id. The DCBs of the file are as follows: LRECL 80 RECFM F BLKSIZE 80. You may code up to eight-character userids in this file. They are coded one per line in the file. When HLIMAIN or IUCVMAIN programs initialize, a search is performed for this file. If it's found, all of the userids stored in the file are read into a list in memory. This list is then used to verify who may connect to this CMS sink id with READ and/or WRITE access. If a client tries to connect to the sinkid and their userid doesn't exist in the FOCSUACC DATA file, an error message is displayed. This provides an equivalent of MVS SUSI for the CMS SU product.

### Syntax

Not Applicable

### Examples

Verification that this feature is working can be simply tested. For example, if FOCSUACC DATA A has two userids coded on two lines, and you start the sink, and attempt to connect with a third userid that is not coded in FOCSUACC DATA an error will display after the first request for data from the sink.

## Special Considerations

ENCRYPT FILE is not supported for this feature.

## Error Messages

`(FOC517) SU. ACCESS DENIED BY EXTERNAL SECURITY SYSTEM:`

## NF617: Automatic Allocation of FOCUS Files

The automatic allocation of FOCUS files in MVS FOCUS was removed from FOCUS Release 7.0.1. This was a dramatic change from prior releases where TABLE, MATCH, MODIFY, etc. commands would search the catalog for 'prefix.master.FOCUS'. If the file is found, the allocation for the file would be made automatically. This change was made for performance reasons.

### Usage

With the new SET command activated before running a TABLE, MODIFY, or MATCH, etc. request, FOCUS dynamically issues the equivalent of a DYNAM ALLOC or TSO ALLOC.

### Syntax      How to Activate or Deactivate Automatic Allocation of FOCUS Files

```
SET FOCALLOC = {ON|OFF}
```

where OFF is the default.

### Example      Activating and Deactivating Automatic Allocation of FOCUS Files

The following shows a TABLE request and its result in any FOCUS Release between 7.0.1. and 7.0.7, without any previous allocation to the EMPLOYEE file:

```
TABLE FILE EMPLOYEE_  
PRINT EMP_ID  
END
```

(FOC036) NO DATA FOUND FOR THE FOCUS FILE NAMED: EMPLOYEE

The same TABLE request in FOCUS Release 7.0.8, without any previous allocation to the EMPLOYEE file, but after setting FOCALLOC on, yields a report with all of the EMP\_ID records in the EMPLOYEE file.

## Reference Special Considerations

? SET ALL shows this feature in its list. ? SET does not show this feature in its list. This is an MVS only feature.

## Error Messages

None.

## NF619: -HTMLFORM SAVE

FOCUS 7.0.8 introduces a new -HTMLFORM SAVE feature for the WEB Interface. With this feature, you can save html content generated by the -HTMLFORM command to a file, rather than to the screen. All FOCUS amper variables (&var) and escape sequences (e.g., '!IBI.AMP.varname') will be fully resolved in the external file.

### Usage

This feature may be useful for those wishing to use FOCUS to generate HTML content in batch mode, rather than interactively.

### Syntax      Using -HTMLFORM SAVE

```
-HTMLFORM BEGIN SAVE AS filename
```

or

```
-HTMLFORM htmlfile SAVE AS filename
```

where:

*filename*

Is a 1 to 8 character filename of the file that will receive the html output from -HTMLFORM.

*htmlfile*

Is a 1 to 8 character filename of a source file containing HTML that you wish to save as output of -HTMLFORM.

In CMS FOCUS, this creates an html file named filename with a filetype of HTML.

In MVS FOCUS (TSO or MSO), this creates an html extract file in one of three ways:

- If ddname 'filename' is allocated, to either a sequential file or a member of a partitioned dataset, the html is written to that file.
- If ddname 'filename' is not allocated, but ddname 'HTML' is allocated to a partitioned dataset, the file is written to the HTML PDS as member 'filename.'
- If neither 'filename' nor 'HTML' is allocated, the file is written to a temporary sequential file using ddname 'filename'.

In each case, this HTML file can then be copied to a Web server for display to end users' browsers.

## **Special Consideration**

Complete documentation for the Web Interface product can be found in the Web Interface User's Manual and Installation Guide Release 7.0.8 (DN1001038.1097).

## **Error Messages**

**FOC36235 - AS KEYWORD NOT SPECIFIED FOLLOWING SAVE**



## NF620: Year 2000 Subroutines

Enhancements have been made to subroutines that handle dates. All subroutines that perform date calculations or perform date format conversions support dates including and after the year 2000. This change was made to coincide with our Year 2000 project.

The subroutines that were re-written for Year 2000 are:

AYMD	Adds and subtracts days from a date.
AYM	Adds and subtracts months from a date.
YM	Finds the number of months between two dates.
CHGDAT	Rearranges the year, month and day portions of dates, and connects dates between long and short formats.
JULDT	Converts dates in year-month-day format into Julian format.
GREGDT	Converts Julian dates to year-month-day dates.
DAxxx	Converts dates into number of days elapsed.
DTxxx	Converts the number of days elapsed into date usage.

### Usage

The new versions of these date subroutines are used by default.

## **Syntax**      How to Choose a Version of a Subroutine

```
SET DATEFNS = {ON|OFF}
```

where:

```
ON
```

is the default.

If you require the older version which does not have Year 2000 capabilities, deactivate this feature by setting DATEFNS = OFF.

## **Example**      Using Subroutines With Year 2000 Capabilities

The following FOCEXEC shows a straight conversion on the input to the subroutine.

```
SET DEFCEINT=19,YRTHRESH=50
  DYNAM ALLOC FILE DATE DS PMSPAK.DATE.FOCUS SHR REU
  TABLE FILE DATE
  ON TABLE SUBHEAD
  " THIS EXAMPLE ILLUSTRATES THE USE OF THE THIRD PARAMETER IN THE
  "
  " SUBROUTINE CALL. THE INPUT FIELD IS AN I6YMD FORMAT. IT CONTAINS
  "
  " 2 DIGIT YEARS. SETTING A PIVOT YEAR OF 1950 RESULTS IN CONVERSION
  "
  " TO A FULL 4 DIGIT YEAR VIA THE FORMAT OF 'I8'.
  PRINT D2_I6YMD AND COMPUTE
  X/I8YYMD=AYMD(D2_I6YMD,1,'I8');
END
```

## Output:

PAGE 1

THIS EXAMPLE ILLUSTRATES THE USE OF THE THIRD PARAMETER IN THE SUBROUTINE CALL. THE INPUT FIELD IS AN I6YMD FORMAT. IT CONTAINS 2 DIGIT YEARS. SETTING A PIVOT YEAR OF 1950 RESULTS IN CONVERSION TO A FULL 4 DIGIT YEAR VIA THE FORMAT OF 'I8'.

```
D2_I6YMD          X
-----          -
97/09/16  1997/09/17
00/02/29  2000/03/01
01/02/28  2001/03/01
00/02/28  2000/02/29
```

The following example demonstrates giving the subroutine a 4 digit year as input.

```
DYNAM ALLOC FILE DATE DS PMSPAK.DATE.FOCUS SHR REU
TABLE FILE DATE
ON TABLE SUBHEAD
" START WITH A COMPUTED FIELD CALLED A. IT HAS A PIVOT YEAR OF 1950."
" CONVERTING IT FROM I6YMD TO YMD SETS THE CENTURY DIGITS TO '20'  "
" FOR THOSE YEARS WHICH ARE LT 50. CREATE OLD DATE FIELD B.      "
" PASS B INTO THE AYMD SUBROUTINE ADDING 1 DAY TO THE DATE.      "
PRINT D2_I6YMD AND COMPUTE
A/YYMD DFC 19 YRT 50 = D2_I6YMD;
B/I8YYMD=A;
C/I8YYMD=AYMD(B,1,C);
END
```

**Output:**

PAGE 1

Start with a computed field called A. It has a pivot year of 1950. Converting it from I6YMD to YYMD sets the century digits to '20' for those years which are less than 50. Create old date field B. Pass B into the AYMD subroutine adding 1 day to the date.

D2_I6YMD	A	B	C
----- -		-	-
97/09/16	1997/09/16	1997/09/16	1997/09/17
00/02/29	2000/02/29	2000/02/29	2000/03/01
01/02/28	2001/02/28	2001/02/28	2001/03/01
00/02/28	2000/02/28	2000/02/28	2000/02/29

**Date Literals Interpretation Table**

This table illustrates the behavior of FOCUS date formats. The columns indicate the number of input digits for a date format. The rows indicate the usage or format of the field. The intersection of row and column describes the result of input and format.

	1	2	3	4
YYMD	*	*	CC00/0m/dd	CC00/mm/dd
MDYY	*	*	*	*
DMYY	*	*	*	*

YMD	*	*	CC00/0m/dd	CC00/mm/dd
MDY	*	*	*	*
DMY	*	*	*	*
YYM	CC00/0m	CC00/mm	CC0y/mm	CCyy/mm
MYY	*	*	*	*
YM	CC00/0m	CC00/mm	CC0y/mm	CCyy/mm
MY	*	*	0m/CCyy	mm/CCyy
M	0m	mm	*	*
YYQ	CC00/q	CC0y/q	CCyy/q	0yyy/q
QYY	*	*	q/CCyy	*
YQ	CC00/q	CC0y/q	CCyy/q	0yyy/q
QY	*	*	q/CCyy	*
Q	q	*	*	*
JUL	CC00/00d	CC00/0dd	CC00/ddd	CC0y/ddd
YY	000y	00yy	0yyy	yyyy
Y	0y	yy	*	*

D	0d	dd	*	*
W	w	*	*	*

	5	6	7	8
YYMD	CC0y/mm/dd	CCyy/mm/dd	0yyy/mm/dd	yyyy/mm/dd
MDYY	0m/dd/CCyy	mm/dd/Cyy	0m/dd/yyyy	mm/dd/yyyy
DMYY	0d/mm/CCyy	dd/mm/Cyy	0d/mm/yyyy	dd/mm/yyyy
YMD	CC0y/mm/dd	CCyy/mm/dd	0yyy/mm/dd	yyyy/mm/dd
MDY	0m/dd/CCyy	mm/dd/Cyy	0m/dd/yyyy	mm/dd/yyyy
DMY	0d/mm/CCyy	dd/mm/Cyy	0d/mm/yyyy	dd/mm/yyyy
YYM	0yyy/mm	yyyy/mm	*	*
MYY	0m/yyyy	mm/yyyy	*	*
YM	0yyy/mm	yyyy/mm	*	*
MY	0m/yyyy	mm/yyyy	*	*
M	*	*	*	*

YYQ	yyyy/q	*	*	*
QYY	q/yyyy	*	*	*
YQ	yyyy/q	*	*	*
QY	q/yyyy	*	*	*
Q	*	*	*	*
JUL	CCyy/ddd	*	*	*
YY	*	*	*	*
Y	*	*	*	*
D	*	*	*	*
W	*	*	*	*

- CC stands for two century digits provided by DFC/YRT settings.
- \* stands for error message FOC177 (invalid date constant).
- FOCUS reads date literals from right to left.

## **Reference** Special Considerations

You may need to deactivate the new subroutines if you are hardcoding the century digits based on specific years. The windowing technique determines what the century digits should be for each subroutine call.



## NF623: Increasing the Number of Verbs in a Report Request

The number of verbs for a multi-verb request, has been increased from 6 to 16.

### Usage

This is extremely useful for executing complex reports. The original requirement of having the detail verb last still applies. In other words, PRINT or LIST must be coded last in the list. The aggregation verb (SUM or COUNT) is coded prior to PRINT or LIST.

### Syntax

Not applicable

### Example Repeating Aggregation of the PCT\_INC Field

The following example shows a repeating aggregation of the PCT\_INC field out of the EMPLOYEE FOCUS file. This example has 15 verb objects, which are aggregated, and 1 verb object, which lists the detail records for LAST\_NAME and FIRST\_NAME. The CURR\_SAL field is also counted for the total number of records, which passed the IF on DEPARTMENT. The resulting report is an effective illustration of how this feature is used.

```
TABLE FILE EMPLOYEE
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM PCT_INC
SUM GROSS
SUM DED_AMT
SUM SALARY
SUM ED_HRS
SUM CNT.CURR_SAL BY DEPARTMENT
LIST LAST_NAME FIRST_NAME
BY DEPARTMENT BY EMP_ID
IF DEPARTMENT EQ 'PRODUCTION'
END
```

## Special Considerations

Not Applicable

## Reference Error Messages

If another SUM PCT\_INC is added to the above request a diagnostic displays:

```
(FOC019) THERE ARE TOO MANY VERBS IN THE REQUEST BYPASSING TO END OF
          COMMAND
```

The total number of verbs in the request would be 17 which exceeds the new limit. This is the same error message that displays in releases prior to R7.0.8 when the old limit of 6 is exceeded.

## NF626: JAVA Graph Wizard

Java Graph Wizard guides a user step-by-step through creating a graph. The Graph Wizard is an alternative to stored graph procedures and generates FOCUS graph syntax from the user's input.

Complete documentation for the Web Interface product can be found in the *Web Interface User's Manual and Installation Guide Release 7.0.8* (DN1001038.1097).

# NF628: Automatic Activation of Web Interface for Web Browser Users

Beginning with Release 7.0.8, FOCUS automatically activates the interactive Web environment for FOCUS Web Interface users entering via the Web Interface Server or WEB390.

## Usage

FOCUS automates the activation of this environment by internally issuing two SET commands (see “Syntax” below).

## Syntax      How to Activate the Web Interface

```
SET HTMLMODE=ON  
SET ONLINE-FMT=HTML
```

## Examples

N/A - see “Syntax.”

## Special Consideration

Sites wishing to activate the Web environment for FOCUS Web Interface users on a selective basis, can do so by turning the SET commands off and reissuing them selectively to activate the environment for only certain users. Complete documentation for the Web Interface product can be found in the Web Interface User’s Manual and Installation Guide Release 7.0.8 (DN1001038.1097).

## **Error Messages**

None

## NF630: Querying Which PTFs Have Been Applied for a Specific Release

By issuing a command at the FOCUS prompt, you are able to view a list of ptf's that have been applied to the version of FOCUS you are currently using.

### Example Querying the PTFs

```
? ptf
```

### Example Using the ? PTF Command

The following displays a screen containing the entering of the ? ptf command, followed by a sample result to the query.

```
FOCUS 7.0.8 10/20/1997 11.44.11 9999.01
>
? ptf
PTFS APPLIED TO RELEASE 7.0.8
FROM PTFABLE LOCATED IN FOCLIB LOADLIB F
```

COUNT	PTF NUM	CREATED	APPLIED
1)	12345	19970403	19971020
2)	12444	19970702	19971020
3)	12499	19970808	19971020

## **Special Considerations**

Not applicable

## **Error Messages**

Not applicable



## NF631: Extended Plists

VM FOCUS now issues commands to CMS using CMS's 'Extended Plist'. This will enable FOCUS users to issue CMS commands such as STORMAP and PIPE that require the extended plist. It will also enable FOCUS users to specify FILEDEF, ACCESS and other CMS commands with parameters that are longer than eight characters.

### Usage

The standard plist or tokenized plist uppercases each token, left-justifies it, and truncates it to a length of 8 bytes. Extended plist has no limit on the length of the plist passed.

### Syntax

N/A

### Reference Examples

Without extended plist the command:

```
FILEDEF MINE DSN TEST.SAMPLE.MAY
```

would read as:

```
FILEDEF  
MINE  
DSN  
TEST.SAM
```

In this case the Data Set Name has lost the 'PLE.MAY' part of the qualifier. A tokenized plist is limited to 32 tokens, each of which may be between 1 and 8 bytes. The extended plist consists of a control block that points to an uppercase command token, a pointer to the start of the option list, and a pointer to the end of the option list. The option list is left unchanged. Using extended plist with the previous example, would pass the following:

```
FILEDEF  
MINE DSN TEST.SAMPLE.MAY
```

With extended plist, all of the Data Set Name is preserved.

### Reference Special Considerations

FOCUS uses a FILEDEF for SYSIN to read from the terminal and from FOCXECs. By default FILEDEF uppercases all data. Reissuing FILEDEF with the LOWCASE option, allows a lowercase option list to be passed to VM. This requires all FOCUS commands to be typed in uppercase. This may be used for special cases when a lowercase plist is required

### Error Messages

N/A

## NF640: Dynamic Language Environment (LE) Support

IBM's recommended platform for high-level language products is known as Language Environment for VM and MVS. It provides a unified platform for runtime services used by LE supported languages. FOCUS user-written subroutines can now be linked using IBM's LE environment. LE support is available for both the MVS and VM operating systems.

This feature enhances our currently announced support for IBM Language Environment. It incorporates the automatic pre-initialization of a Language Environment Enclave or Sub-Enclave as required. This allows customer written HLL subroutines that were linked with LE libraries to run most efficiently. The implementation also allows non-LE-linked subroutines to run as well. FOCUS determines the characteristics of these subroutines and invokes them using LE interface module CEEPIPI if they are LE linked or in the traditional manner if they are not.

### Usage

All standard FOCUS subroutines, whether linked prior to LE or using the LE single runtime library are supported depending on Language Environment restrictions. Existing FOCUS subroutine libraries need not be recompiled and relinked unless your site converts to a single run-time library.

### Syntax

Not Applicable

## Examples

Not Applicable

## Reference Special Considerations

Refer to IBM manual , SC28-1944 *Language Environment for OS/390 and VM Run-Time Migration Guide*, for instructions on how to use this environment and conversion limitations.

## Error Messages

Not Applicable

## NF642: Increased DEFINE Limitation

The limit of 256 for the number of DEFINES allowed in FOCUS has been removed. The limit now is dependent on the amount of memory available. The number of fields, both real and defined that can be referenced in a single request is still 256. It is possible to define as many fields as you need, as long as you have enough memory and the total number of fields referenced in a request is no more than 256.

## NF645: WEBHOME

FOCUS 7.0.8 introduces the Web Interface feature 'WEBHOME' which enables Web application developers to specify execution of a default FOCEXEC procedure in situations where FOCUS would normally return to command level. This allows them to prevent application users from accidentally or intentionally accessing command level FOCUS from within a Web application. Previously, FOCUS always displayed the 'FOCUS Interactive' screen by default whenever a Web Interface user entered FOCUS or after they ran a drill-down report or graph that cleared FOCSTACK. With this feature, developers can now display a menu or form that they choose, instead of returning to the FOCUS command level. This will ensure that the FOCUS application environment remains intact and consistent even when Web applications include drill-downs and/or allow users to access the ad hoc Java report and graph generation tools.

### Usage

WEBHOME is activated via a FOCUS SET command that you can specify globally for a FOCUS session or issue in a FOCEXEC for a specific application.

### Syntax      Identifying a FOCEXEC to Run Automatically

```
SET WEBHOME= {focexecname|OFF}
```

where:

*focexecname*

Is the 1 to 8-character filename of the FOCEXEC to be called instead of returning to the FOCUS command level and displaying the FOCUS Interactive screen

In CMS FOCUS, the focexecname must have a filetype of FOCEXEC.

In MVS FOCUS (TSO or MSO) the FOCEXEC must be a member of a partitioned dataset allocated to ddname FOCEXEC.

#### OFF

The default, OFF, disables WEBHOME and restores the default behavior - the FOCUS Interactive screen is displayed whenever FOCUS returns to command level.

### Example Automatically Running a FOCEXEC

```
SET WEBHOME=APPMENU1
```

The above SET command causes the FOCEXEC named APPMENU1 to be called each time FOCUS would normally return to command level.

### Reference Special Consideration

This SET command is valid only in the Web Interface environment. It has no effect in a standard FOCUS 3270 session.

### Error Messages

```
This SET command only valid in Web Interface Environment.
```

## NF647: Extended Support for Scandinavian External Sort

FOCUS supports external sort with the Scandinavian National Languages Character set, and is able to pass the sort sequences for Swedish , Danish, Finnish, and Norwegian to the external sorting products. To specify the National Language Support Environment, use the LANG parameter as described in Section 21 of the FOCUS 7.0 Users Manual.



## Project 2000 - Phase III

The third phase of our year 2000 project for FOCUS includes rewritten user subroutines which perform date manipulations. They are: AYMD, AYM, YM, CHGDAT, GREGDT, JULDAT, DAxxx, DTxxx and FOCUS functions YMD,DMY,MDY. These subroutines and functions will now allow for century digit interpretation via the DEFCENT and YRTHRESH FOCUS settings. Date calculations for these subroutines may now extend beyond year 1999. The subroutines have also been enhanced to respect the last argument which may contain the output format from the subroutine.

## 7.0.7M New Features

**MAINTAIN**

**MAINTAIN Updates**

## MAINTAIN Updates

Release 7.0.7M of CMS and MVS FOCUS contains over 150 fixes for the MAINTAIN product. This is a full-function FOCUS release which was created primarily to deliver MAINTAIN fixes. Please refer to the Known Problems section of READMEF for a listing of MAINTAIN problems.

## 7.0.7 New Features

### **Web Interface for FOCUS**

#### **NF580: Web Interface for FOCUS**

## NF580: Web Interface for FOCUS

The Web Interface delivers access to mainframe FOCUS applications from a standard Web browser (i.e., Netscape Navigator or Microsoft Internet Explorer), rather than (or, as well as) 3270 terminals. This enables you to “webify” existing applications with few if any changes, while permitting you to create new applications for the Internet or a corporate intranet that can fully exploit the Web’s universal accessibility and the full range of HTML output styling options.

### Web Interface Features

The Web Interface includes the following features:

- **HTML Formatted Reporting.** You can style FOCUS reports in standard HTML (Hypertext Markup Language) for output via Web browsers. HTML is the formatting language for all Web communications. You activate HTML styling through traditional StyleSheet syntax, which has extensions designed specifically for the Web. Add colors, fonts, boldface, italics, even pictures (GIF or JPEG files), and other advanced features to give your reports high impact on the browser screen. Reports can be viewed on the browser without adding **any** specialized desktop software. You can also add drill-down capabilities, allowing end users to click on particular elements of a report to get more information about a particular category or individual line item. Drill-down support allows users to click on virtually any element, including headings or graphical images, to get more information or take some other Web-based action such as FTP or E-MAIL. Reports can be viewed immediately or saved for viewing or distribution via the new HOLD FORMAT HTML feature.

- **FOCUS Interactive:** The Web Interface can be deployed in an online, fully interactive environment, allowing application users to access existing FOCUS applications on the Internet or an intranet-based environment. The “Web Interface Server” is an advanced version of our Web3270 server, which provides a logon and user-authentication environment and form translation services for the Web Interface. FOCUS forms can be viewed in standard 3270 mode on your Web browser or translated into HTML to give them the “look and feel” of other Web applications.
- **New -HTMLFORM command:** This new Dialog Manager command permits you to create new FOCUS reporting applications using customized HTML forms to generate reports. The new HTMLFORMs support embedded images, highlighting, drill-downs, hyperlinks to other FOCEXECs, and many other advanced features.

The Web Interface for mainframe FOCUS is documented in a new *Web Interface Users Manual and Installation Guide*.

## Syntax      How to Produce HTML Output From a FOCUS Report

```
SET HTMLMODE=ON
```

```
SET ONLINE-FMT=HTML
```

These SET commands prepare FOCUS and StyleSheets for accepting input from a browser and producing HTML output. Setting both OFF returns you to the normal FOCUS operating environment.

## Examples

See the *Web Interface Users Manual*.

## Special Consideration

N/A

## Error Messages

See the *Web Interface Users Manual*.

## 7.0.6 New Features

### Year 2000

#### Project 2000 - Phase II

#### NF523: Cross-Century Dates in FOCUS Applications - Phase II

### Raised Limits

#### NF536: Multi-image FOCSORT

### Reporting Enhancements

#### NF526: PRINTPLUS

### General Enhancements

#### NF546: Enhancements to JOIN

#### NF549: Interpreting Quotation Marks Within Quote-Delimited Literal Strings

#### NF552: Estimating SORTWORK Sizes for an External Sort

### The Multi-Session Option

#### NF502: MSO VTAM Logon Time-out

#### NF530: Language Environment (LE) Support

#### NF531: 31 Bit I/O



## **The Multi-Session Option (Continued)**

[NF532: MSO Monitoring and Statistics](#)

[NF547: EDA to MSO Bridge](#)

[NF553: Enhanced Message Routing](#)

[NF554: Load Balancing for the Multi-Session Option \(MSO\)](#)

[NF576: MSO Dynamic VTAM Re-configuration](#)

## **FOCUS Personal Agent**

[NF543: FOCUS Personal Agent - TCP/IP Connectivity Option for VM/CMS FOCUS](#)

[NF544: FOCUS Personal Agent - TCP/IP Connectivity Option for TSO FOCUS](#)

[NF578: FOCUS Personal Agent - Interruptible Server for VM and MVS FOCUS](#)

## **Relational Interfaces**

[NF539: Outer Join Optimization](#)

[NF540: Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS](#)

[NF542: Optimization of Joins Between Heterogeneous File Types](#)

## **ADABAS Interface**

**NF517: ADABAS Dynamic Security**

**NF538: ADABAS Dynamic Database Number**

## **IMS Interface**

**NF512: IMS Interface - Automatic Index Selection Using  
AutoSelect**

## NF502: MSO VTAM Logon Time-out

MSO administrators can now specify time-outs for MSO VTAM Logon screens by adding a parameter in their MSO Configuration files (ddname FOCMSO).

The configuration parameter, LOGON\_TIMEOUT, allows MSO administrators to set a maximum number of seconds that the MSO Logon screen is displayed.

When that period expires the connection is broken and the user is returned to the VTAM Logon screen. This frees up the VTAM session.

### **Syntax**      **How to Set the MSO VTAM Logon Time-out**

The format of the new configuration parameter is:

```
LOGON_TIMEOUT = nnn
```

where:

nnn

Is a positive number between 30 and 100000 representing the number of seconds the MSO VTAM Logon screen will be displayed before being canceled.

## NF512: IMS Interface - Automatic Index Selection Using AutoSelect

The AutoSelect feature modifies the way IMS secondary indexes are defined in FOCUS Master File Descriptions, improving IMS Interface performance by reducing the size of data sets retrieved from IMS through exploitation of IMS secondary indexes. This modification alleviates the need to create multiple FOCUS Master File Descriptions to describe IMS databases with secondary indexes.

### Usage

Previously, in creating FOCUS Master File Descriptions to describe IMS database structures to FOCUS, database administrators had to maintain several versions of the Master Files to accommodate descriptions of the IMS secondary indexes used for data retrieval. One IMS DBD might have several FOCUS Master Files associated with it. The new AutoSelect feature removes the need for these additional Master Files.

Now, when describing an IMS DBD to FOCUS, you describe all associated indexes as the last set of fields in the root segment of your Master File Description.

### Syntax

Please refer to the new *IMS/DB Interface Users Manual and Installation Guide* for Release 7.0 (DN1000977.0997) for complete information on syntax and coding examples.

## **Examples**

See manual.

## **Special Considerations**

See manual.

## **Error Messages**

See manual.

## NF517: ADABAS Dynamic Security

It is now possible to set ADABAS user passwords from the command level in FOCUS and EDA/SQL. Previous versions of the ADABAS Interface required that the password be specified in the FOCADBS Access file via the 'PASS=' parameter. This required duplicate copies of the FOCADBS Access files for each user or group with different ADABAS passwords.

### Usage

A new SET command allows the password in the FOCADBS file to be overridden, or omitted entirely. Specifying the password in the Access file is still supported.

The new SET command is only available with the new ADABAS Interface (ADABAS=NEW), and remains valid throughout the user's session.

### Syntax      How to Set the ADABAS User Password Dynamically

The SET PASSWORD command can clear previous SET PASSWORD COMMANDS or establish a password.

The syntax of the SET command for establishing a password for *all* files is:

```
{CMS|MVS} ADBSIN SET PASSWORD pass FILENO ALL DBNO {ALL|dbno}
```

The syntax of the SET command for establishing a password for *specific* files is:

```
{CMS|MVS} ADBSIN SET PASSWORD pass FILENO fileno DBNO dbno
```

The syntax of the SET command for establishing a password for *specific* files is:

```
{CMS|MVS} ADBSIN SET PASSWORD {OFF|DEFAULT}
```

where:

pass

Is the 1- to 8-character password

FILENO

Specifies the file or files to which the password applies.

DBNO

Specifies the database or databases to which the password applies.

ALL

Is used with the FILENO and DBNO parameters to indicate ALL files and/or all databases. If you want use ALL for both FILENO and DBNO, it should be applied before any specific FILENO and DBNO information, as this combination overrides any prior settings.

*fileno*

Is used with the FILENO parameter to provide a file number, list of file numbers, and/or a range of file numbers. Numbers and ranges can be combined by separating items with commas. The maximum file number is 255.

*dbno*

Is used with the DBNO parameter to indicate the actual database number. The maximum database number is 255.

OFF

Clears all previous ADBSIN SET commands. The Interface will not use any passwords at all, even overriding passwords specified in the Access file. This command allows the user to only access files that have no password security.

### DEFAULT

Clears all previous ADBSIN SET commands and causes the Interface to use the password in the Access file.

### Example Setting the ADABAS User Password Dynamically

1. To set the password to 'MARY' for all databases and all files:

```
CMS ADBSIN SET PASSWORD MARY FILENO ALL DBNO ALL
```

2. Set the password to 'MARY' for all files of database number 150:

```
CMS ADBSIN SET PASSWORD MARY FILENO ALL DBNO 150
```

3. Set the password to 'MARY' for specific files of database number 123:

```
MVS ADBSIN SET PASSWORD MARY FILENO 1,3-5,23,89-93 DBNO 123
```

4. Clear all previous ADBSIN SET commands, only allowing the user to access files that have no password security:

```
CMS ADBSIN SET PASSWORD OFF
```

5. Clear all previous ADBSIN SET commands, causing the Interface to use the password in the Access file:

```
MVS ADBSIN SET PASSWORD DEFAULT
```



## Overriding Default Passwords in Specific Files

It is possible to set default passwords for use in all files and/or all databases, and set specific passwords to override the default in particular files. In order to do so, first issue the SET PASSWORD command using the ALL keyword for FILENO and/or DBNO. The ALL keyword clears any existing passwords in effect for the database (or for ALL databases if this keyword is used for DBNO). Then issue specific password requests by specifying particular file and/or database numbers (or ranges) in a subsequent SET command.

### **Example**    **Setting Default Passwords and Overriding Them in Specific Files**

For example:

```
MVS ADBSIN SET PASSWORD FRED FILENO ALL DBNO ALL
```

sets the password to FRED in all files and in all databases.

Then set the password to MARY in database number 123, file numbers 1,3,4 and 5. However, the default value of FRED remains in effect for all other files:

```
MVS ADBSIN SET PASSWORD MARY FILENO 1,3-5 DBNO 123
```

### Special Considerations

The new SET command is supported only with the new ADABAS Interface (ADABAS=NEW), and remains valid throughout the user's session. This feature works only with Release 7.0 and above.

### Error Messages

```
(FOC4507) ERROR - DBNO IS NOT SPECIFIED
```

(FOC4508) MAXIMUM NUMBER OF SET PASSWORDS IS REACHED

(FOC4509) ERROR IN ADABAS SETTING

(FOC4511) SET FOR

(FOC4517) INVALID KEYWORD FOR DBNO SPECIFIED

(FOC4518) INVALID KEYWORD FOR FILENO SPECIFIED

(FOC4519) INVALID VALUE FOR FILENO

(FOC4549) MAXIMUM NUMBER OF SET DBNOs IS REACHED

## NF523: Cross-Century Dates in FOCUS Applications - Phase II

With the year 2000 approaching, application managers must cope with the fact that 19 is assumed as the first two digits of the year for most of their applications. Applications receiving transaction years of 00 will typically interpret that as 1900. There is considerable risk that date-sensitive calculations in existing applications will be wrong unless an apparatus is provided for determining the century in question. This will impact almost every type of application, including those that process mortgages, insurance policies, anniversaries, bonds, inventory replenishment, contracts, leases, pensions, receivables and customer records.

The second phase of the Cross-century-dates feature enables users to solve this problem at the file and field level of their applications (a solution at the global level was introduced in release 7.0.5). At the file level, you can now retain your global settings while changing the file level settings for greater flexibility. For more information about global settings see New Feature Bulletin 497, *Cross Century Dates in FOCUS Applications*.

### **New Settings**

Four new settings were added for Master File Descriptions, two each for specifying file and field level handling. Settings made at the field level take precedence over file level settings, which in turn take precedence over global settings. DEFINES and COMPUTEs are supported.

## Syntax      How to Establish Cross-Century Dates in a Master File

At the file level, the settings FDEFCENT and FYRTHRESH were added. The FDEFCENT syntax is

`{FDEFCENT|FDFC} = nn`

where:

*nn*

Is 19 unless otherwise specified.

The FYRTHRESH syntax is

`{FYRTHRESH|FYRT} = nn`

where:

*nn*

Is zero unless otherwise specified.

At the field level, DEFCECENT and YRTHRESH have been added. The syntax is

`{DEFCECENT|DFC} = nn`

where:

*nn*

Is 19 unless otherwise specified.

`{YRTHRESH|YRT} = nn`

where:

*nn*

Is zero unless otherwise specified.

## Syntax How to Establish Cross-Century Dates in COMPUTE and DEFINE

The following is the syntax for COMPUTE and DEFINE:

```
DEFINE FILE EMPLOYEE
  fld/fmt [{DEFCENT|DFC} nn {YRTHRESH|YRT} nn] [MISSING...] = exp;
  ...
END
```

The DFC and YRT syntax must follow the field format information.

### Example Establishing Cross-Century Dates in a Master File

```
FILENAME=EMPLOYEE, SUFFIX=FOC,FDEFCENT=20,FYRTHRESH=66,$
SEGNAME=EMPINFO, SEGTYPE=S1
FIELDNAME=EMP_ID, ALIAS=EID, FORMAT=A9, $
FIELDNAME=LAST_NAME, ALIAS=LN, FORMAT=A15, $
FIELDNAME=FIRST_NAME, ALIAS=FN, FORMAT=A10, $
FIELDNAME=HIRE_DATE, ALIAS=HDT, FORMAT=I6YMD,
DEFCENT=19, YRTHRESH=75, $
```

To see the application of DEFCENT and YRTHRESH to interpret two-digit years, consider the following:

```
DEFCENT=19, YRTHRESH=80
```

describes a range from 1980 to 2079. If a two-digit year field contains 99 then the year is 1999. If it's 79 then the year is 2079. If it's 00 then the year is 2000.

## Example Establishing Cross-Century Dates With DEFINE and COMPUTE

The following two examples illustrate the conversion of a two-digit year field with DEFINE and COMPUTE respectively:

```
DEFINE FILE EMPLOYEE
ESHIRE_DATE/YYMD = HIRE_DATE; (The format of HIRE_DATE is I6YMD).
ESHIRE DFC 19 YRT 80 = HIRE_DATE
END
```

```
TABLE FILE EMPLOYEE
SUM SALARY AND COMPUTE
ESHIRE_DATE/YYMD = HIRE_DATE; (YYMD uses whatever is specified).
END
```

## Reference Special Considerations

- Compiled MODIFYs from releases earlier than 7.0.6 must be recompiled to use this feature.
- ON TABLE SET is not allowed with DEFCENT and YRTHRESH.
- In HOLD Masters the concatenation of FILE and FIELD level information is propagated as FIELD level.
- With DEFINE and COMPUTE, both DFC and YRT must be specified. Each can be specified alone in the Master File Description.

## Reference Error Messages

The following error messages may be displayed when attempting to use a MODIFY created before FOCUS Release 7.0.6.

(FOC1885)      WARNING: MODIFY USES CENTURY INFO. WILL NOT RUN BELOW 7.0.6

The user has set Year 2000 information that relates to fields in this MODIFY. MODIFYS compiled with such information will not run in versions of FOCUS prior to 7.0.6.

(FOC1886)      ERROR: %1 FOCCOMP NEEDS CENTURY INFO. PLEASE RECOMPILE.

The user is using the Project 2000 feature, which affects some fields in the named compiled MODIFY. To prevent entry of inconsistent data, we require that the user recompile their MODIFY.

## NF526: PRINTPLUS

PRINTPLUS introduces enhancements to the display alternatives offered by the FOCUS Report Writer. For example, you might wish to place a FOOTING after a SUBFOOT in your report. You now have the flexibility to produce the exact report you desire.

### Usage

A new PRINTPLUS SET command must be on (the default) to use these new TABLE capabilities. With PRINTPLUS on, the following occur:

- PAGE-BREAK is handled internally to provide the correct spacing of pages. For example, if a new report page is started and an instruction to skip a line at the top of the new page is encountered, FOCUS now knows to suppress the blank line and start at the top of the page.
- NOSPLIT is handled internally. (See Special Considerations).
- You can now perform RECAPs in cases where pre-specified conditions are met.
- A Report SUBFOOT now prints above the footing instead of below it.

### Syntax      How to Enable PRINTPLUS

```
SET PRINTPLUS = {ON|OFF}
```



**Example Using PRINTPLUS**

With PRINTPLUS on (the default), the SUBFOOT prints first, followed by the FOOTING. Old behavior printed the FOOTING followed by SUBFOOT.

```
USE CAR FOCUS F
END

TABLE FILE CAR
PRINT CAR MODEL
BY SEATS BY COUNTRY
IF COUNTRY EQ ENGLAND OR FRANCE OR ITALY
ON TABLE SUBFOOT
" "
" SUMMARY OF CARS IN COUNTRY BY SEATING CAPACITY"
FOOTING
" RELPMEK CAR SURVEY "
END
```

The output is:

SEATS	COUNTRY	CAR	MODEL
-----	-----	---	-----
2	ENGLAND	TRIUMPH	TR7
	ITALY	ALFA ROMEO	2000 GT VELOCE
		ALFA ROMEO	2000 SPIDER VELOCE
		MASERATI	DORA 2 DOOR
4	ENGLAND	JAGUAR	V12XKE AUTO
		JENSEN	INTERCEPTOR III
	ITALY	ALFA ROMEO	2000 4 DOOR BERLINA
5	ENGLAND	JAGUAR	XJ12L AUTO
	FRANCE	PEUGEOT	504 4 DOOR

```
SUMMARY OF CARS IN COUNTRY BY SEATING CAPACITY
RELPMEK CAR SURVEY
```

## **Special Considerations**

- To force a break at a specific spot, you must use NOSPLIT.
- PRINTPLUS is not supported with StyleSheets. A warning message is generated in this case.
- Problems may be encountered if HOTSCREEN is set OFFLINE. A warning message is generated.

## **Error Messages**

N/A

## NF530: Language Environment (LE) Support

IBM's recommended platform for high level language products is known as Language Environment for VM and MVS. It provides a unified platform for runtime services used by LE supported languages. FOCUS user-written subroutines can now be linked using IBM's LE environment. LE support is available for both the MVS and VM operating systems.

### Usage

All standard FOCUS subroutines, whether linked prior to LE or using the LE single runtime library are supported depending on Language Environment restrictions. Existing FOCUS subroutine libraries need not be recompiled and relinked unless your site converts to a single run-time library.

### Syntax

Not Applicable

### Examples

Not Applicable

### Special Considerations

Refer to IBM manual SC26-4818, "Language Environment for MVS and VM Release 5 Programming Guide", for instructions on how to use this environment and conversion limitations.

## **Error Messages**

Not Applicable

## NF531: 31 Bit I/O

By exploiting this MVS 5.1 feature, we halved the virtual storage requirements below the 16 Meg line needed to support a typical FOCUS user. This means more FOCUS users can operate in a single address space. The actual benefits gained by supporting 31 bit I/O are visible via your MSO console.

### Usage

All FOCUS-generated sequential I/O routines exploit this feature. User-written routines that do their own sequential I/O must be recoded in order to exploit it.

**Note:** The storage utilization for HiperFOCUS does not use below-the-line storage. HiperFOCUS and 31 bit I/O are mutually exclusive.

### Syntax

N/A

### Examples

N/A

### Special Considerations

MVS 5.1 and higher and FOCUS 7.0.6. and higher are required.

### Error Messages

N/A

## NF532: MSO Monitoring and Statistics

MSO monitoring and statistics provide detailed MVS resource utilization reports for a region or for all users. The information can be viewed immediately online or evaluated in MSOPRINT files.

Four kinds of information are available:

- On Demand -- reports MVS resource statistics for users on an 'on demand basis'
- Monitoring -- reports MVS resource statistics for users at user specified intervals
- Short on Storage -- displays console message when region reaches site-defined threshold
- Shutdown -- reports summary statistics by service at termination for the MSO region

### Usage

"On Demand" statistics are produced whenever requested. To initiate monitoring, issue the MONITOR=ON command in either the Configuration file or from the MSO or MVS console. Any command can be turned on or off via the console. The extent of information collected is determined by the specific command set issued from the console or configuration file (see Syntax section for a complete listing). All output defaults to MSOPRINT except Short on Storage messages which are also sent as non-scrollable messages to the operator console (WTO). Shutdown statistics are always produced and cannot be disabled

## Syntax     Displaying MSO Statistics On Demand

These commands can be issued from MSO/MVS console only. To display statistics based on CONFIG file specifications, type:

```
F servername,DISPLAY STATS
```

To display ALL statistics regardless of CONFIG file specifications, type:

```
F servername,DISPLAY STATS,ALL
```

## Syntax     Displaying MSO Short On Storage Messages

```
SOS_PERCENT = {nn|0}
```

where:

*nn*

Is an integer representing a percentage between 0 and 99 percent. The default is 0.

Short on Storage specifies the percentage of storage utilization, which when exceeded, will cause a warning message to be issued via non-scrollable messages to WTO and MSOPRINT. The message destination can be altered with the "Enhanced Message Routing Facility"(NF553). If storage utilization falls below this value, a message is issued stating that the storage constraint has been relieved and the WTO message is deleted (unless suppressed by the "Enhanced Message Routing Facility"). The SOS\_PERCENT check, if active, is made at every MRM\_INTERVAL interval (MSO Resource Manager Interval).

Specifying 0 disables the SOS monitoring facility. SOS monitoring may be enabled or disabled dynamically by setting SOS\_PERCENT to a non-0 or 0 value, respectively.

APF authorization is required.

## Syntax      Monitoring MSO Execution

These commands can be issued from either the CONFIG file or the MVS or MSO consoles.

Issue In ...	Syntax
CONFIG FILE:	<code>MONITOR = {ON OFF}</code>
MSO CONSOLE:	<code>/MONITOR {ON OFF}</code>
MVS CONSOLE:	<code>F <i>servname</i>,MONITOR {ON OFF}</code>

where:

### MONITOR

Turns monitoring on or off. OFF is the default.

MONITOR OFF resets current setting for MN\_USER, and for MN\_INTERVAL, etc. to original values in the CONFIG file.

APF authorization is required

To set the interval of time for monitoring, issue one of the following commands.



Issued In ...	Syntax
CONFIG FILE:	<code>MN_INTERVAL = nnn</code>
MSO CONSOLE:	<code>/MN INTERVAL nnn</code>
MVS CONSOLE:	<code>F servname, MN INTERVAL nnn</code>

where

`MN_INTERVAL`

Specifies the interval at which to perform monitoring (when active).

`nnn`

Is an integer indicating the interval in seconds. The minimum is 30 and the maximum is 3600.

To turn on monitoring, use `MONITOR=ON`. Monitoring is then performed at integer multiples of the `MRM_INTERVAL` (MSO Resource Manager Interval).

Default: 1800 (30 minutes)

APF authorization is required

To specify whether monitoring will include storage statistics, issue one of the following commands.

Issued In ...	Syntax
CONFIG FILE:	<code>MN_STORAGE = {<u>OFF</u>   ON}</code>

MSO CONSOLE:	/MN_STORAGE { <u>OFF</u>  ON}
MVS CONSOLE:	F <i>servname</i> ,MN_STORAGE { <u>OFF</u>  ON}

where:

MN\_STORAGE

Specifies whether storage statistics will be included if monitoring is active (the default setting is OFF). To turn monitoring on, use MONITOR=ON.

APF authorization is required.

To specify whether monitoring will include region statistics, issue one of the following commands.

Issued In ...	Syntax
CONFIG FILE:	MN_REGION = { <u>OFF</u>  ON}
MSO CONSOLE:	/MN_REGION { <u>OFF</u>  ON}
MVS CONSOLE:	F <i>servname</i> ,MN_REGION { <u>OFF</u>  ON}

where:

MN\_REGION

Specifies whether region statistics are required when monitoring is active (the default setting is OFF).

APF authorization is required.

To specify whether monitoring will be by userid, issue one of the following commands.

Issued In ...	Syntax
CONFIG FILE:	<code>MN_USERS = {<u>OFF</u> ALL}</code>
MSO CONSOLE:	<code>/MN_USERS {<u>OFF</u> ALL}</code>
MVS CONSOLE:	<code>F <i>servname</i>,MN_USERS {<u>OFF</u> ALL}</code>

where:

#### MN\_USERS

Specifies whether or not to do monitoring by userid. To turn on monitoring, issue MONITOR=ON (the default is OFF).

Specifying ALL produces usage statistics for all logged on userids.

Specifying OFF curtails monitoring of userids.

APF authorization is required

## Special Considerations

None

### Example Monitoring MSO Execution

The following initiate monitoring:

CONFIG FILE:	MONITOR=ON
MSO CONSOLE:	/MONITOR ON
MVS CONSOLE:	F <i>servname</i> ,MONITOR ON

These values initiate monitoring, but reset the monitoring interval to 10 minutes:

CONFIG FILE:	MN_INTERVAL=600 MONITOR=ON
MSO CONSOLE:	/MN_INTERVAL 600 /MONITOR ON
MVS CONSOLE:	F <i>servname</i> ,MN_INTERVAL 600 F <i>servname</i> ,MONITOR ON

This turns ALL monitoring off AND resets monitoring controls to the values in the CONFIG FILE:

CONFIG FILE:	N/A
MSO CONSOLE:	/MONITOR OFF
MVS CONSOLE:	F <i>servname</i> ,MONITOR OFF

Assuming that monitoring keywords are not specified in the CONFIG file, you could selectively turn on only STORAGE monitoring (using the default interval) as follows:

CONFIG FILE:	N/A
MSO CONSOLE:	<code>/MN_STORAGE YES</code> <code>/MONITOR ON</code>
MVS CONSOLE:	<code>F servname,MN_STORAGE YES</code> <code>F servname,MONITOR ON</code>

Again, assuming monitoring keywords are not specified in the CONFIG file, you issue the following to monitor all users every 10 minutes:

CONFIG FILE:	N/A
MSO CONSOLE:	<code>/MN_INTERVAL 600</code> <code>/MN_USERS ALL</code> <code>/MONITOR ON</code>
MVS CONSOLE:	<code>F servname,MN_INTERVAL 600</code> <code>F servname,MN_USERS ALL</code> <code>F servname,MONITOR ON</code>

With monitoring on, to raise the monitoring interval to 15 minutes immediately, issue:

CONFIG FILE:	N/A
MSO CONSOLE:	<code>/MN_INTERVAL 900</code>
MVS CONSOLE:	<code>F servname,MN_INTERVAL 900</code>

With monitoring on, issue the following to selectively turn off storage monitoring:

CONFIG FILE:	N/A
MSO CONSOLE:	<code>/MN_STORAGE OFF</code>
MVS CONSOLE:	<code>F servname,MN_STORAGE OFF</code>

With monitoring on, issue the following to selectively turn off userid monitoring:

CONFIG FILE:	N/A
MSO CONSOLE:	<code>/MN_USERS OFF</code>
MVS CONSOLE:	<code>F servname,MN_USERS OFF</code>

With monitoring started, issue the following to monitor ALL userids:

CONFIG FILE:	N/A
MSO CONSOLE:	/MN_USERS ALL
MVS CONSOLE:	F <i>servname</i> ,MN_USERS ALL

## Reference Messages for ON DEMAND and MONITORING

- ON DEMAND: Messages are issued according to the CONFIG FILE values, unless the user issues "display stats,ALL"
- MONITORING: Messages are issued according to the CONFIG FILE values as (currently) modified by (MSO/MVS) console commands.

```
(MSO13373)  MONITORING STATISTICS FOR xxxxxxxx (JOBxxxxxx)
(MSO13373)  ON DEMAND STATISTICS FOR xxxxxxxx (JOBxxxxxx)
(MSO13374)  STATISTICS FOR STORAGE
(MSO13372)  BELOW: REGION =   xxxK IN USE =   xxxK USED=xxx% SOS=xx%
(MSO13372)  ABOVE: REGION = xxxxxxK IN USE = xxxxxxK USED=xxx% SOS=xx%
(MSO13375)  STATISTICS FOR REGION
(MSO13383)  CPU_T=xxxxx.xx CPU%=xx.xx EXCP_T=xxxxxxxx EXCP/S=xxxx
(MSO13384)  LU2_NAME=xxxxxxxx ACTIVE USERS=xxxx APPLGROUP=xxxxxxxx
(MSO13377)  STATISTICS FOR USERS
```

The following messages appear one time per logged on user.

```
(MSO13386)  STATISTICS FOR LOGON ID=xxxxxx LOGON: yy.ddd hh:mm
```

```
(MSO13387)  SERVER=xxxxxxxxx  TCBADR=xxxxxxx  TERMADR=xxxxxxxxx
(MSO13383)  CPU_T=xxxxxx.xx  CPU%=xx.xx  EXCP_T=xxxxxxxxxx  EXCP/S=xxxxx
(MSO13388)  STG_BELOW=xxxxxxxK  STG_ABOVE=xxxxxxxK
(MSO13389)  WAIT_TIME=xxxxxx SECONDS  (SINCE LAST "ENTER")
(MSO13392)  END OF DISPLAY
```

## Reference Messages for Short on Storage

Only the address space/storage messages are issued, along with a "critical shortage" or "critical shortage relieve message

```
(MSO13371)  CRITICAL STORAGE SITUATION BELOW/ABOVE THE LINE FOR JOB
            xxxxxxxxx (JOBnnnnn)
```

followed by MSO13372 giving details

```
(MSO13378)  CRITICAL STORAGE SITUATION BELOW/ABOVE THE LINE RELIEVED FOR
            JOB xxxxxxxxx (JOBnnnnn)
```

followed by MSO13372 giving details

## Reference Messages for Shutdown

```
(MSO13373)  SHUTDOWN STATISTICS FOR xxxxxxxxx (JOBxxxxxx)
(MSO13374)  STATISTICS FOR STORAGE
(MSO13380)  BELOW: MAX_USED = xxxxxxK  OUT_OF xxxxxxK      USED=xxx%
(MSO13380)  ABOVE: MAX_USED = xxxxxxK  OUT_OF xxxxxxK     USED=xxx%
(MSO13375)  STATISTICS FOR REGION
(MSO13381)  MAX_CPU%=xx  MAX_EXCP/SEC=xxxxx  (PER MRM_INTERVAL)
(MSO13385)  USERS FAILED:  BAD_PSWD=xxxxxx  BAD_ACCT=xxxxxx  NO_STORAGE=x
(MSO13398)  USERS FAILED:  NO_SERVICE=xxxxxxx
```



```
(MSO13382)  SERVER WAS ACTIVE FOR xx DAY(S), xx HOUR(S) and xx.xx MINUTE(  
(MSO13376)  STATISTICS FOR SERVICES
```

The following messages appear one time per service

```
(MSO13390)  SERVICE=xxxxxxxx PROGRAM=xxxxxxxx MAXIMUM=xxxx  
(MSO13391)  MAX_ACTUALLY_USED=xxxx  TOTAL_TASKS=xxxxxxxx  
(MSO13394)  USERS FAILED:  EXCEEDED_MAXIMUM=xxxxxx QUIESCED_SERV=xxxxxx  
(MSO13395)  USERS FAILED:  CNCLD=xxxxxx  ABENDED=xxxxxx  
(MSO13397)  USERS FAILED:  IDLELIM=xxxxxx  TIMEOUT=xxxxxx  
(MSO13396)  CPU_T=xxxxxx  CPU%=xx.xx  
(MSO13392)  END OF DISPLAY
```

## NF536: Multi-image FOCSORT

The MVS FOCSORT work file may now expand upon its initial allocation up to a total of 16 temporary data sets. A new temporary data set is allocated when the initial FOCSORT space is exhausted. The size of these temporary data sets is based upon the first FOCSORT allocation.

## NF538: ADABAS Dynamic Database Number

To improve ease of use, you can now set ADABAS database numbers (DBNOs) from the command level in FOCUS and EDA/SQL. Previous versions of the ADABAS Interface required specification of DBNO in the FOCADBS Access file via the 'DBNO=' parameter. This necessitated keeping duplicate copies of the FOCADBS Access file for each database accessed. A new SET command allows users to override the DBNO in the FOCADBS file. Specifying database numbers in Access files is still supported.

### Usage

This feature makes Master and Access files sharable among databases.

### Syntax      How to Set ADABAS Database Numbers From the Command Line

The syntax of the SET command is:

```
{CMS|MVS} ADBSIN SET DBNO = xxx
```

```
{CMS|MVS} ADBSIN SET DBNO = xxx AFD = yyy
```

```
{CMS|MVS} ADBSIN SET DBNO = xxx AFD = yyy segname= zzz
```

where:

*xxx*

Is any valid numeric database value between 0 - 255.

*yyy*

Is any valid Access file name.

zzz

Is any valid ADBS segname within the Access file.

To display the current settings, issue the following command:

```
{CMS|MVS} ADBSIN SET ?
```

The following command replaces current DBNO parameter with the original Access file default DBNO value.

```
{CMS|MVS} ADBSIN SET DBNO DEFAULT
```

### Example Setting the ADABAS Database Number

To run all requests from database number 5:

```
MVS ADBSIN SET DBNO 5
```

To run requests for the EMPFILE master and access files in database 2:

```
MVS ADBSIN SET DBNO 2 AFD EMPFILE
```

To run requests against the EMPFILE master and access files for a particular ADBS segment:

```
MVS ADBSIN SET DBNO 2 AFD EMPFILE SEG S02
```

To check settings:

```
MVS ADBSIN SET ?
```

To return to default DBNO setting (which will be read from access file):

```
MVS ADBSIN SET DBNO DEFAULT
```

## Reference Special Considerations

The new SET command is supported only with the new ADABAS Interface (ADABAS=NEW), and remains valid throughout the user's session. This feature works only with Release 7.0 and above.

## Reference Error Messages

(FOC4507) ERROR - DBNO IS NOT SPECIFIED

(FOC4508) MAXIMUM NUMBER OF SET PASSWORDS IS REACHED

(FOC4509) ERROR IN ADABAS SETTING

(FOC4511) SET FOR

(FOC4517) INVALID KEYWORD FOR DBNO SPECIFIED

(FOC4518) INVALID KEYWORD FOR FILENO SPECIFIED

(FOC4519) INVALID VALUE FOR FILENO

(FOC4549) MAXIMUM NUMBER OF SET DBNOs IS REACHED

## NF539: Outer Join Optimization

This feature improves FOCUS Relational Interface performance by enabling the interfaces to deliver better optimized SQL to RDBMSes, permitting them to optimize their own join processing.

### Usage

The interfaces now pass left outer joins to the RDBMS when you issue the FOCUS command SET ALL=ON (note: SET ALL=ON previously disabled optimization, leaving FOCUS to handle joins). All other rules governing joins are as documented in the *DB2 Read/Write Interface Users Manual* (see Section 7.2.2).

### Note:

- The SET ALL=ON command also controls processing of short paths. The interface default setting is OFF. For a complete description of short path processing, please refer to your Interface Users Guide.
- Please be aware that in passing requests to an RDBMS with SET ALL=ON, you may get correct, yet subtly different, sequences of report rows than you had with earlier releases of the Interface. These differences are due to differences between sorting algorithms used by FOCUS and by other database management systems, and do not indicate upward compatibility problems.

## **Syntax**      **How to Optimize a Left Outer Join**

When you issue the following commands, the Interface creates SQL for a left outer join and passes the join to the RDBMS for processing:

```
SET ALL=ON
SQL SET OPTIMIZATION ON
```

### **Example**      **Passing a Left Outer Join to DB2**

This example shows the SQL passed to DB2 for a FOCUS dynamic join with SET ALL=ON, SQL SET OPTIMIZATION ON: \_

```
JOIN field1 IN file1 TO field2 IN file25
SELECT T1.field1,T2.field2 FROM "creator.table1" T1 LEFT OUTER JOIN
"creator.table2" T2 ON T2.field2 = T1.field1 FOR FETCH ONLY;
```

### **Example**      **Passing a Left Outer Join to Oracle**

This example shows the SQL passed to Oracle for a FOCUS dynamic join with SET ALL=ON, SQL SET OPTIMIZATION ON: \_

```
JOIN field1 IN file1 TO field2 IN file2
SELECT T1. field1,T2.field2, FROM "user"."table1" T1,
"user"."table2" T2 WHERE (T2.field2 (+) = T1.field1);
```

## **Reference**      **Special Considerations**

- Optimization of outer joins is available for FOCUS Interfaces installed with DB2 Versions 4 and higher or Oracle Release 6 and higher.

- For Sybase and Informix the answer sets returned differ from DB2 or Oracle when the FOCUS request contains selection tests on columns of a cross-referenced table.

With Sybase or Informix, selection criteria for missing data in cross-referenced rows is ignored, so a row is always returned for a missing column. This is consistent with normal FOCUS SET ALL=PASS behavior. FOCUS SET ALL=ON only invokes an outer join when no selection tests are made on columns of any cross-referenced tables. Specify SET ALL=PASS to instruct the interface to pass outer joins when there are selection tests against cross-referenced tables.

### Error Messages

N/A



## NF540: Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS

This relational interface enhancement delivers optimized SQL to each RDBMS, allowing the RDBMS to optimize its own execution and minimizing the size of the answer sets returned to requesters - an extremely important capability for client-server applications.

### Usage

Aggregated TABLE requests containing DEFINE field objects of FOCUS BY clauses can now be passed to DB2, Oracle and Teradata as aggregated requests. The verbs in these requests must be FOCUS aggregation verbs (SUM, COUNT or WRITE) or direct operators such as MIN., MAX., and AVE. Previously, the interfaces could pass only certain DEFINE expressions to SQL as part of record selection.

Note:

- When the interfaces pass expressions as objects of SQL GROUP BY or ORDER BY phrases, you may get correct yet subtly different results than with previous releases of the interface - that is, different sequences of rows within a given range of sort field values. These differences are due to subtle differences between the sorting algorithms used by FOCUS and the other database management systems and do not represent upward compatibility problems.

- With interface optimization set OFF, the interface generates an individual SQL statement for each table, leaving FOCUS to handle all aggregation, sorts and join operations to produce the report. This switch allows you to invoke previous interface behavior if the differences due to using different aggregation/sorting algorithms is unacceptable.

## **Syntax**      **How to Invoke DEFINE Optimization**

Turn optimization on with the following command

```
SET OPTIMIZATION ON
```

## **Example**      **Passing Aggregation on DEFINE Fields to the RDBMS for Processing**

This example shows the SQL passed to DB2 following an aggregation on a DEFINE field:

```
DEFINE FILE FILE1
TAX = 0.085 * PRICE
END

TABLE FILE FILE1
SUM PRICE TAX
BY TAX NOPRINT
END

SELECT SK001, SUM(VB001), SUM(VB002) FROM (SELECT (.085 *
T1.PRICE) AS SK001,T1.PRICE AS VB001, (.085 * T1.PRICE) AS VB002
FROM TABLE1 T1 ) X GROUP BY SK001 ORDER BY SK001 FOR
FETCH ONLY;
```

This example shows the SQL passed to the Oracle DBMS: \_

```
SELECT (.085 * T2.PRICE), SUM((.085 * T2.PRICE)),  
SUM(T2.PRICE) FROM TABLE1 T2 GROUP BY (.085 * T2.PRICE)  
ORDER BY (.085 * T2.PRICE);
```

This example shows the SQL passed to the Teradata DBMS: \_

```
SELECT (.085 * T2.PRICE)(FLOAT), SUM((.085 * T2.PRICE))(FLOAT),  
SUM(T2.PRICE)(DECIMAL(15,02)) FROM TABLE1 T2 GROUP BY (.085  
T2.PRICE) ORDER BY (.085 * T2.PRICE);
```

## **Reference Special Consideration**

For DB2, this new optimization enhancement applies only to DB2 Interfaces installed with DB2 version 4 and higher.

## **Error Messages**

N/A

## NF542: Optimization of Joins Between Heterogeneous File Types

This Relational Interface performance feature improves the SQL that the interfaces pass to relational database servers, improving their native abilities to optimize their own execution. Through this feature, the FOCUS relational interface JOIN mechanism can now pass a single SELECT statement per TABLE request when all active segments of the given SQL suffix in a file comprise a contiguous single-path subtree. Formerly, our relational interfaces could not optimize JOIN operations linking two or more SQL segments if the retrieval path included segments of different suffix types (for example, several DB2 segments and an IMS segment). Instead, optimization was turned off, forcing the Interfaces to generate a single SQL SELECT statement for each segment, and leaving FOCUS to process the returned answer sets. This could result in repeated tablespace scans by the RDBMS.

This feature removes the previous limitation that all active segments in the file have the same suffix. It is initiated through a SET statement and is set ON by default.

## Usage

The new JOIN mechanism passes a single SELECT statement per TABLE request when all active segments of a given SQL suffix in a file comprise a contiguous single-path subtree. Single path means no non-unique SQL segment can have an SQL parent and a non-unique twin. Active segments are those containing fields referenced in the request, or fields involved in JOIN operations. Contiguous segments are connected SQL segments with a common suffix. There may be segments with different suffixes above or below. Segments occupying intermediate positions in a file hierarchy between explicitly active segments are themselves also implicitly activated.

### **Note:**

- If the request specifies a sort operation, the Interface transfers the sort operation to the RDBMS only if the root of the SQL subtree is the topmost active segment in the file. Additionally, all BY fields must be contained within the segments of the SQL subtree. If the multiplicative effect is detected and the Interface-managed native join is executed, the Interface passes an SQL ORDER BY clause on all columns of each segment's primary key, in top-to-bottom order. The resulting sort on the returned answer set enables the Interface to eliminate duplicate rows before passing the data to FOCUS.

- All rules governing joins apply as documented in the DB2 Read/Write Interface Users Manual - section 7.2.2. However, the restriction on non-SQL structures has been lifted. When a sort is now passed to the RDBMS, slightly different, although correct reports may be produced - this does not indicate upward compatibility problems between releases of the Interface. Possible slight changes in report row sequencing within a given range of a sort key value reflect minor differences between the sorting/aggregation algorithms used by FOCUS and by the other database management systems. If such differences are unacceptable, simply set Interface optimization OFF, leaving the Interface to generate an individual SQL statement for each table. FOCUS will then handle all aggregation, sorts and join operations required to produce the report.

### Syntax      How to Invoke Join Optimization With Heterogeneous File Types

To invoke Interface optimization, issue the following command:

```
SET OPTIMIZATION ON
```

### Example      Optimizing Joins Between Heterogeneous File Types

The following examples illustrate SQL passed to DB2 as the result of a FOCUS dynamic join between two DB2 tables and an IDMS record. There is also an example of the previous behavior, which is what you get with Interface optimization set OFF.

```
JOIN field1 in DB2file1 TO field2 in DB2file2 AS join1  
JOIN field1 in DB2file1 TO field3 in IDMSfile AS join2
```

### New behavior:

```
SELECT T1.field1,T2.Field2 FROM "creator"."table1" T1,"creator"."table2"  
T2  
WHERE (T2.field1 = T1.field2) FOR FETCH ONLY
```

For each DB2 joined row that is fetched, the join field is used as input to an IDMS OBTAIN record command utilizing an IDMS index or a Calc key.

### Old behavior:

```
SELECT T1.field1 FROM "creator.table1" T1 FOR FETCH ONLY
```

After a row is fetched from table1, the join field is used as input to the second select .

```
SELECT T2.field2 FROM "creator.table2" T2  
WHERE (T2.field2 = ?) FOR FETCH ONLY
```

For each DB2 joined row that is fetched, the joined field is used as input to an IDMS OBTAIN record command using an IDMS index or a Calc key.

## **Special Considerations**

N/A

## **Error Messages**

N/A

## NF543: FOCUS Personal Agent - TCP/IP Connectivity Option for VM/CMS FOCUS

FOCUS Personal Agent allows the user or application developer to use FOCUS For Windows as a graphical user interface. You can generate code and produce styled Reporting and Graph applications against character-based FOCUS, running on a number of different platforms. A new feature in the 6.0 release of FOCUS Personal Agent is the ability to connect to host FOCUS via the highly efficient TCP/IP protocol. Using the Sockets interface of TCP/IP (Winsock 1.1 protocol), FOCUS Personal Agent allows you to automate the startup of your host-based FOCUS “server” session via the “rexec” (Remote Exec) function of TCP/IP. This enables you to access data from that remote FOCUS session as if it were data local to the PC.

### Reference Usage

In order to enable this feature on the host FOCUS side, you need several new source files:

- **FPAIP TEXT** - A new routine that manages the TCP/IP conversation between the client (FOCUS For Windows) and the server (host FOCUS). This file is a standard text deck and should be installed onto the FOCUS disk.



- **FPX FOCEXEC** - A FOCEXEC that is used to establish the server environment. Among other things, this includes the allocation of all temporary work files. In FOCUS 7.0.6 and above, this is included on the VM/CMS FOCUS tape. In FOCUS 7.0.5 and below, it should be placed on the FOCUS disk.
- **FPAFOCUS EXEC** - A REXX EXEC that starts your remote FOCUS session from FOCUS Personal Agent. The first section of this EXEC may need to be customized for your site to include LINK/ACCESS commands to FOCUS and any other disks needed by your applications. Also, your PROFILE EXEC will need to include the LINK/ACCESS command to access your TCP/IP maintenance disk.
- **SASC RUNTIME LIBRARY** - In order to make use of C routine calls to TCP/IP on the host, it is necessary that you include the SASC runtime library (release 5.5.0). The runtime library is distributed as part of this new feature and should be installed onto the FOCUS disk.

The use of this feature is initiated from the FOCUS Personal Agent on the PC side in the Remote Connect profile. Once you have verified that your FPAFOCUS EXEC and PROFILE EXEC link to all of the necessary disks, specify the FPAFOCUS name in the 'Procedure' line of your FPA Remote Connect Profile

## Syntax      How to Connect to a VM FOCUS Host via TCP/IP With FOCUS Personal Agent

The syntax to call FPAFOCUS is:

```
FPAFOCUS CLIENT(123.45.67.89) {PORT(pc_port_number)}  
{LOCSETUP(fexname)}
```

where:

`123.45.67.89`

Is the client IP address (alternatively, %client% can be specified and this will be resolved at run-time).

`pc_port_number`

Is an optional parm if you would like to connect back to a PC port other than port 4222 (the default). Alternatively, %port% can be specified if you are using a random port address on the PC.

`fexname`

Is an optional host-based FOCEXEC (for example., a profile) which is to be executed on the host to setup the environment. The LOCSETUP parm can be specified via any of the following methods:

- FOCEXEC name up to eight characters (e.g., MYFEX).
- FOCEXEC name and filetype (e.g., MYFEX FOCEXEC).
- FOCEXEC name, filetype, and filemode (e.g., MYFEX FOCEXEC A).

A connection can also be initiated from the CMS Ready prompt when the Personal Agent is in “Awaiting Connect” mode. In that case the IP address of the PC must be specified, and %client% cannot be used.

## **Example** Connecting to a VM FOCUS Host Via TCP/IP With FOCUS Personal Agent

From the PC, on the Procedure line in the Remote Connect Window, type:

```
ex fpafocus client(%client%) locsetup(myfex)
```

From the CMS Ready prompt, type:

```
ex fpafocus client(123.45.67.89) locsetup(myfex)
```

## **Special Considerations**

N/A

## **Error Messages**

N/A

## NF544: FOCUS Personal Agent - TCP/IP Connectivity Option for TSO FOCUS

FOCUS Personal Agent allows the user or application developer to use FOCUS For Windows as a graphical user interface to generate code and produce styled Reporting and Graph applications against character-based FOCUS running on a number of different platforms. A new feature in the 6.0 release of FOCUS Personal Agent is the ability to connect to host FOCUS via the highly efficient TCP/IP protocol. Using the Sockets interface of TCP/IP (Winsock 1.1 protocol), FOCUS Personal Agent now allows you to automate the startup of your host-based FOCUS “server” session via the “rexec” (Remote Exec) function of TCP/IP, and access data from that remote FOCUS session as if it were data local to the PC. To use the rexec function of TCP/IP, you need to have a remote execution server running in your TSO environment. For more information about remote execution servers, contact your TCP/IP system administrator.

### Reference Usage

In order to enable this feature on the host FOCUS side, you need several new source files:

- **FPAIP** - a new routine that manages the TCP/IP conversation between the client (FOCUS For Windows) and the server (host FOCUS).

- **FOCEXEC(FPX)** - a FOCEXEC (FOCUS Program) which is used to establish the server environment. Among other things, this includes the allocation of all temporary work files. In FOCUS 7.0.3 and above, this should be included in a partitioned dataset allocated to ddname ERRORS. In FOCUS 7.0.1 and below, it should be included in a partitioned dataset allocated to ddname FOCEXEC.
- **FPAFOCUS** - a sample CLIST that includes all of the allocations necessary to start your remote FOCUS session from FOCUS Personal Agent. This will need to be customized for your site and should be placed in a location that is accessible to all users, either as a member of a partitioned dataset or as a sequential dataset.
- **SASC RUNTIME LIBRARY** - In order to make use of C routine calls to TCP/IP on the host, it is necessary that you include the SASC runtime library (Release 5.5.0) in your startup CLIST, allocated to ddname 'CTRANS.' The runtime library is distributed as part of this new feature.

## Syntax      How to Connect to a TSO FOCUS Host via TCP/IP With FOCUS Personal Agent

The use of this feature is initiated from the FOCUS Personal Agent on the PC side in the Remote Connect profile. However, in order to use this feature, you will need to verify that you have all of the necessary host files in the appropriate places on your MVS system. The location of each of the necessary files is documented in the previous section. Once you have verified that your FPAFOCUS CLIST includes all of the necessary allocations and you have noted the location of this dataset, you will specify the FPAFOCUS CLIST name in the 'Procedure' line of your FPA Remote Connect Profile. The syntax to call FPAFOCUS is as follows:

```
ex 'pdsname(FPAFOCUS)' 'CLIENT(123.45.67.89) {PORT(pc_port_number)}  
{LOCSETUP(fexname)}'
```

where:

*pdsname*

Is the fully qualified dataset name, if FPAFOCUS is stored in a partitioned dataset and 'FPAFOCUS' is the membername.

*'123.45.67.89'*

Is the client IP address (alternatively, %CLIENT% can be specified and this will be resolved at run-time).

*pc\_port\_number*

Is an optional parm if you would like to connect back to a PC port other than port 4222 (the default). Alternatively, %port% can be specified if you are using a Random port address.

fexname

Is an optional host-based FOCEXEC (e.g., a profile) which is to be executed on the host to setup the environment. The LOCSETUP parm can be specified via any of the following four methods:

- Eight character FOCEXEC name where the FOCEXEC is a member of a dataset allocated to ddname FOCEXEC (e.g., MYFEX).
- Eight character FOCEXEC name where the FOCEXEC is allocated as sequential file to the ddname indicated by the FOCEXEC name (e.g., MYFEX).
- Forty-four character fully qualified dataset name of a sequential dataset that contains the FOCUS code. This data set name does not have to be allocated in the CLIST by the user. It will be allocated in the FPAFOCUS CLIST (e.g., userid.FOCEXEC.SEQUENT).
- Fifty-four character fully qualified dataset name of a PDS with the member name in parenthesis (e.g., userid.FOCEXEC.DATA(MYFEX)).

In methods 2 and 3 (sequential files), the FOCEXEC must follow the rules for sequential FOCEXECs and not contains any Dialogue Manager commands. In case 3 and 4 (dataset name provided), the dataset name must be fully qualified and not be enclosed in quotes.

### **Example** Connecting to a TSO FOCUS Host Via TCP/IP With FOCUS Personal Agent

```
ex 'prefix.clist.data(fpafocus)' 'client(%client%) locsetup(MYFEX)'
```

## NF546: Enhancements to JOIN

The JOIN command has been enhanced to permit joining two files containing different numeric datatypes. For example, a short packed field can now be joined to a long packed field, or an integer field joined to a packed field. This provides enormous flexibility for creating reports from joined files.

### Enabling Datatype Conversion

In order for datatype conversions to take place, FOCUS must access the interface module GNTINT. This allows for conversion of the USAGE in the FROM file to match those of the TO file. When joining to a FOCUS file, it is necessary to issue the command SET JOINOPT = GNTINT.

When joining a shorter field to a longer field, the FROM value is padded to the length of the TO field, adding spaces (for alpha fields) or hexadecimal zeroes (for numeric fields). This new value is used for searches in the cross-referenced file.

When joining a longer field to a shorter field, the FROM value is truncated. If part of your value is truncated due to the length of the USAGE in the cross reference file, only records matching the truncated value will be found in the cross-reference file.

### Syntax      How to Enable Joins With Datatype Conversion

When joining to a FOCUS file where datatype conversion must take place prior to issuing the JOIN, it is necessary to issue the following command:

```
SET JOINOPT = GNTINT
```



## Example Issuing Joins With Datatype Conversion

As stated earlier, you can now join a short packed field to a long packed field. The following two Master File Descriptions describe two possible files that can be joined:

```
FILE=PACKED , SUFFIX=FIX , $  
  SEGNAME=ONE , SEGTYPE=S0  
  FIELD=FIRST , , P8 , P4 , INDEX=I , $  
FILE=PACKED2 , SUFFIX=FIX , $  
  SEGNAME=ONE , SEGTYPE=S0  
  FIELD=PFIRST , , P31 , P16 , INDEX=I , $
```

## Reference Special Considerations

When joining packed fields the preferred sign format of X'C' for positive values and X'D' for negative values is still required. All other non-preferred signs are converted to either X'C' or X'D'.

## Error Messages

N/A

## NF547: EDA to MSO Bridge

Any FOCUS client (FFW, FPA, etc.) connected to an MVS EDA server can gain direct access to full functioning FOCUS using MSO. Access is simple and transparent to the user.

### Usage

The user, from their EDA FOCUS client issues the command SET EXTFOC. All subsequent commands are sent directly to MSO FOCUS where they are processed. Control reverts to EDA when a FIN or ??? is sent.

### Syntax      How to Initiate MSO FOCUS From EDA

The command to begin using MSO FOCUS can be issued from any of the following:

- EDA Global Profile:EDASPROF
- EDA user Profile:EDAPROF(userid)
- Client: Any Remote Procedure
- From any stored procedure:On the EDA server

The syntax is

```
SET EXTFOC = {ON|appl.group|.group}
```

where:

ON

Begins sending all subsequent commands directly to MSO FOCUS. The default MSO APPLGROUP and GROUP name as defined to MSO are used..

*appl.group*

Begins sending all subsequent comments directly to MSO FOCUS using this APPLGROUP and GROUP name in the command.

*.group*

Begins sending all subsequent comments directly to MSO FOCUS. The default MSO APPLGROUP as defined to MSO and this GROUP name are used.

All EDA Service Blocks default to allow use of the EDA to FOCUS Bridge. The following attribute in a service block either allows or disallows use of the bridge.

EXTFOC = {ON|OFF}

where:

ON

The Service Block in which this keyword is included allows access to the EDA to FOCUS Bridge. This is the default if EXTFOC is not specified in the server configuration file.

OFF

The Service Block in which this keyword is included does not allow access the EDA to FOCUS Bridge.

## Examples

Not Applicable

## **Reference Special Consideration**

The EDA - MSO Bridge requires a working FOCUS client with a working connection to a MVS EDA 3.2 server and FOCUS 7.0.6. The IBI Subsystem must be installed. EDA and FOCUS must be installed in the same MVS image. The FOCUS client must be connected to an EDA service that permits access to the Bridge.

MSO must be installed with an APPLGROUP. For more information about MSO, see documents “MSO Installation and Technical Reference Guide” DN1000966.1095, “Load Balancing” NF503 and “Load Balancing Enhancements” NF554.

## **Error Messages**

## NF549: Interpreting Quotation Marks Within Quote-Delimited Literal Strings

FOCUS can now interpret quote-delimited strings containing embedded quotation marks. FOCUS treats two contiguous quotes within a quote-delimited string as a single literal quote.

### Usage

Quote delimited strings are supported in the following :

- IF and WHERE statements containing multiple quotes
- WHERE statements containing: field {IS, IS-NOT, IN, IN FILE or NOT IN FILE}
- EDIT
- WHEN field EQ embedded quote in a literal
- DEFINE statements
- DEFINE statements in Master File Descriptions
- DBA expressions in Master File Descriptions (e.g., VALUE = IF field EQ embedded quotes in literal)
- ACCEPT=, DESCRIPTION=, TITLE= in Master File Descriptions
- AS
- DECODE

### Syntax

N/A

## **Example** Using Quote-Delimited Literal Strings

For example, if a report request contains the following statement:

```
IF AIRPORT EQ 'O'HARE'
```

FOCUS will interpret the literal string 'O'HARE and look for a database values of O'HARE.

### **Special Considerations**

N/A

### **Error Messages**

N/A

## NF552: Estimating SORTWORK Sizes for an External Sort

Specifying the size of your SORTWORK files is usually done when you install an external sort product. Large FOCUS queries that use external sorts can rapidly exhaust SORTWORK space, resulting in FOC909 errors. This problem is solved if the sorts include the parameter 'FILSZ=En' when invoked. This parameter enables the sorting algorithms to estimate SORTWORK space requirements for each sort parameter request.

### Usage

A new FOCUS set switch, ESTRECORDS, is used to pass the estimated number of records to be sorted in the request. In order to make an accurate estimate for your ESTRECORDS setting, it is suggested that you run the report without an external sort in order to get a record count.

### Syntax      How to Provide an Estimate of Input Records for Sorting

The syntax is

```
ON TABLE SET ESTRECORDS n
```

where:

*n*

Is the estimated number of records to be sorted.

### Examples

N/A

## Reference Special Considerations

ESTRECORDS can only be set with the ON TABLE SET command within the TABLE, MATCH, or GRAPH request.

For CMS/SyncSort the 'FILSZ=En' parameter is ignored. Therefore, SET ESTRECORDS n has no effect.

## Reference Error Messages

If an attempt is made to SET ESTRECORDS from the FOCUS prompt, FOCPARAM, or PROFILE FOCEXEC the following error is generated:

```
SET ESTRECORDS = n
```

```
(FOC36210) THE SPECIFIED PARAMETER CAN ONLY BE SET ON TABLE: ESTRECORDS
```



## NF553: Enhanced Message Routing

Enhanced message routing allows MSO installations to control the printing/ listing destination of all MSO messages. MSO installations can choose to route messages to the operator's console (WTO) or MSOPRINT or both.

### Usage

Enhanced message routing is implemented by adding a control character to the MSO messages found in the the members MSOERR1 and CMSO1ERR of the PDS prefix.ERRORS.DATA.

### Syntax      How to Route MSO Messages

Standard MSO error messages look like this: 00000(MSO12345) TEXT

Enhanced routing replaces the blank field in position 16 (before "text") with:

- b,B      to route the message to both the console and MSOPRINT
- p,P      to route the message to MSOPRINT only
- w,W      to route the message to the console only
- \_        blank defers to the default output destination

### Example      Routing MSO Messages

The following is an error message found in prefix.ERRORS.DATA(MSOERR1).

```
(MSO13001) INSUFFICIENT CORE FOR GETMAIN
```

This example shows how to route it to just the console:

```
(MSO13001)wINSUFFICIENT CORE FOR GETMAIN
```

This example shows how to route it to just MSOPRINT:

```
(MSO13001)pINSUFFICIENT CORE FOR GETMAIN
```

This example shows how to route it to both the console and MSOPRINT.

```
(MSO13001)bINSUFFICIENT CORE FOR GETMAIN
```

## Reference Special Consideration

Any changes to the ERRORS datasets must be reapplied with each FOCUS update.

## Error Messages

Not Applicable

## NF554: Load Balancing for the Multi-Session Option (MSO)

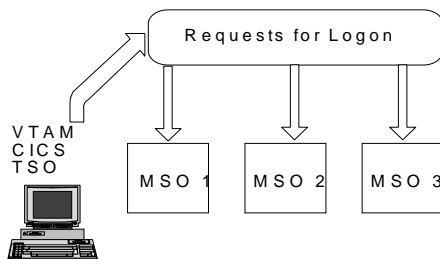
### (Enhanced for Release 7.0.6)

Multi-Session Option users can now benefit from Load Balancing, a feature that allows MSO administrators to distribute FOCUS users across multiple MSO regions to reduce contention for system resources. Information Builders developed MSO to allow mainframe sites to put many FOCUS users in a single address space called a MSO region. As users increase, it's possible to add additional MSO regions. Rather than ask users to determine which MSO region to Logon to or require that your system programmers develop extensive CICS/VTAM applications, we developed Load Balancing to provide a transparent and efficient way to distribute users across multiple MSO regions.

With Load Balancing, the MSO administrator can control access to and the use of resources. The administrator's job is simplified with Load Balancing; users are routed only to the MSO regions which are set up for their applications, all users have access to MSO FOCUS with a single profile, and users are automatically routed to the region which has capacity.

Users find MSO easier to use. They are routed to the MSO region with the applications they wish to run. For MSO administrators, all communications setup is done once. Implementing Load Balancing is as simple as adding a keyword to the configuration file.

With Load Balancing implemented at your site, FOCUS growth is easily managed. As your FOCUS needs expand, this architecture can easily accommodate your growing requirements.



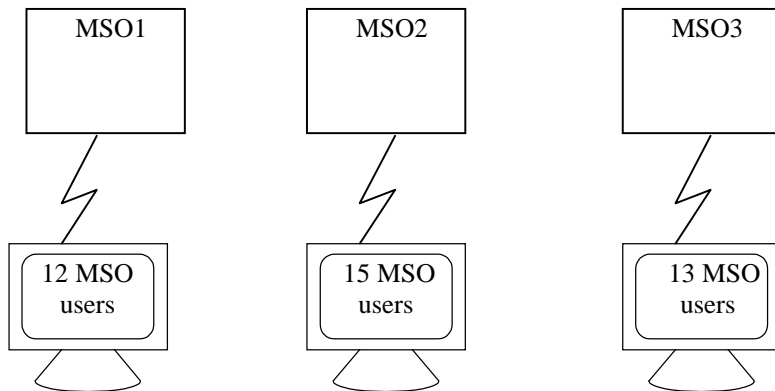
## Load Balancing

### Overview

As MSO users Logon, Load Balancing has the capability of automatically distributing them across a group of up to 32 MSO regions within a single MVS image. Users may connect to MSO via TSO, VTAM, or CICS. MSO, in conjunction with the IBI Subsystem at the time of initial connection, determines which MSO address space in the group of defined and started address spaces is the “best one” to connect this MSO user. Refer to the *Multi-Session Option Installation and Technical Reference Guide Release 7.0* for a full discussion of MSO.

The actual routing is based upon the number of users currently logged on to each MSO region, the level of CPU activity each MSO region has incurred during its last MSO Resource Monitor (MRM) interval, and finally, all things being equal, a “Round Robin” selection.

Load Balancing simplifies the setup and installation of multiple MSO regions. Communication is set up with one primary MSO region. Load Balancing also makes it easier for MSO users. The MSO Logon procedure is now just one VTAM applid or VTAM menu selection, or one CICS transaction. The search for an available MSO region is transparent to the user. See Figure 1: Effect of MSO Load Balancing. The MSO user may access any Load Balancing Group that is active at logon time.



**Figure 1: Effect of MSO Load Balancing**

Load Balancing introduces the concepts of groups or application groups and applications. Put simply, a group is a set of MSO regions participating in Load Balancing. The group is defined by the MSO administrator. There can be one or more groups but there must be at least one. The MSO administrator chooses a single group when the user community is fairly homogeneous. There can be up to 32 MSO regions in a group.

MSO administrators can use application groups to segregate users into homogenous groups of MSO regions. For example, you might create TEST, PROD, and STAGE groups. Or, an application group may be created to segregate users into non-competing (for resources) groups. If, for example, a company has several divisions (Marketing, Engineering, and Accounting) all with very active users, to keep users in Accounting from competing for resources with Engineering, the MSO administrator could set up three application groups, giving each the number of MSO regions required for that user community.

A given MSO region can belong to only one group. If it does not belong to any group, that region does not participate in Load Balancing.

Groups can be further subdivided into mutually exclusive applications. An application is a name representing users with similar needs and Logon Profiles. It is not necessary for the MSO administrator to assign applications. If no application names are assigned, all users will be Balanced across all MSO regions assigned to the application group.

If a single group has many applications, the MSO administrator can assign application names to one or many of the participating MSO regions, effectively restricting those users with the same application to a subset of the total number of MSO regions available.

Applications further restrict competition between MSO regions. If, for example the engineering group has a very important application (Cost Analysis), the MSO administrator can assign that application to many of the MSO regions to optimize Load Balancing. Less important applications (Vacation Requests) can be assigned to only a few of the group's regions.

An application name can appear in multiple groups and has different characteristics in each group. If FOCUS users are updating an existing application, that application name can exist in the PROD, TEST, and STAGE groups. For example, the 'On-line Ordering System' has been extensively changed. Some users are using the new system (STAGE), some users are using the old system (PROD) and there are developers making changes (TEST). Each set of users is running the application named 'On-line Ordering System' but the 'On-line Ordering System' is different in each group.

The rest of this document describes how plan for and setup your MSO FOCUS site to most effectively use Load Balancing.

## **MSO Load Balancing Requirements**

The IBI Subsystem must be active and the MSO program libraries must be APF authorized to support this feature.

## **MSO Load Balancing Approach**

MSO Load Balancing depends upon a repository of information that pertains to all MSO regions in the MVS system. How MSO stores information required to implement Load Balancing is illustrated in Figure 2: Capturing Load Balancing Information.



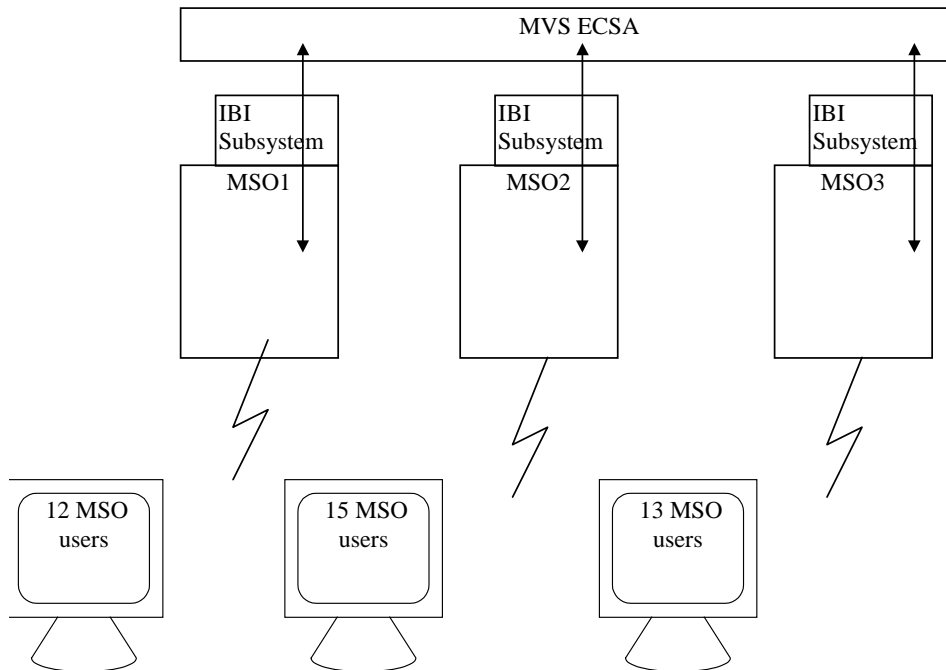


Figure 2: Capturing Load Balancing Information

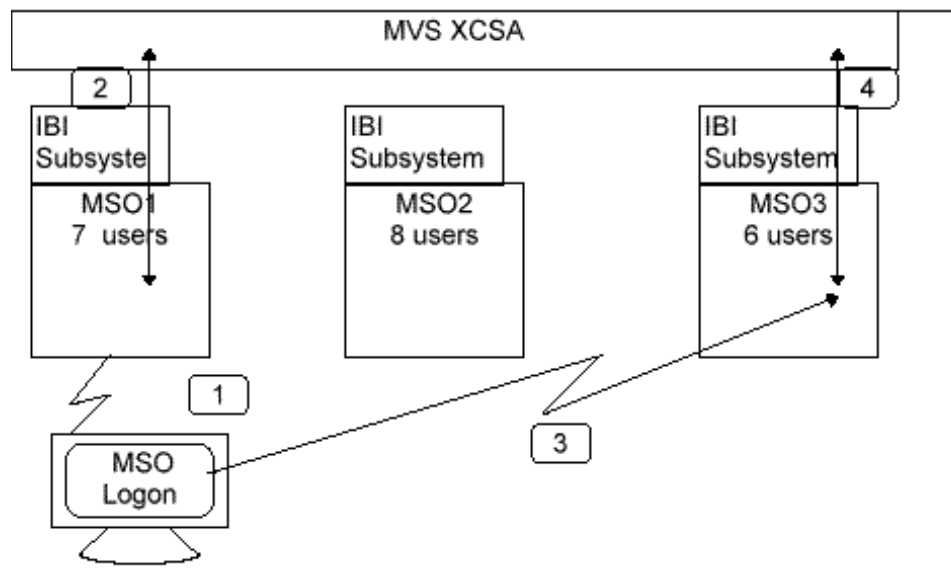
Each MSO region updates its own status information stored in ECSA. In this Common Storage Area, above the 16 megabyte virtual storage line, storage is not taken up in what is termed the Private area. It does not affect the amount of private storage below the 16 megabyte line available to all MVS programs. MSO uses the IBI Subsystem to update this information in ECSA. It stores information about the number of users currently logged on and the amount of CPU that it consumed during its last MRM interval.

## **Load Balancing Sequence**

The sequence of events in a Load Balancing Logon is shown in Figure 3: Load Balancing Sequence.

- 1. Logon Request: to Primary or initially contacted MSO Region**

The user initiates the MSO Logon request to an MSO region that is part of an MSO Load Balancing group. This initial connection is based upon the MSO 7.0 communication methods supported from TSO, VTAM, and CICS. The initial logon request may specify the Load Balancing Application Group and application name to override the settings in the IBI Subsystem.



**Figure 3: Load Balancing Sequence**

- CICS users make initial contact based upon the MSO transaction name they specified to invoke MSO.
- TSO users are connected to the primary MSO region based upon the current allocation of the MSGET and MSPUT DDnames when the MSUSER command is executed.

- The VTAM user establishes communication based upon the VTAM applid specified by the user in a VTAM Logon command or supplied by a VTAM selection menu displayed to the user.
- If the VTAM Logon request did not contain a userid, the MSO Logon Screen is displayed to the user by the primary or initially contacted region. Figure 4: MSO FOCUS Logon Screen illustrates this screen.

All three communication environments allow the user to specify the Load Balancing Group name and the application name that controls MSO region selection by Load Balancing. If the connection request does not specify an MSO Application Group or application, the defaults that are set in the IBI subsystem are used. The initially contacted MSO region does not do any Logon verification of the user. After entering a userid, the next step begins. The CICS and TSO user does not specify userid or password as this is carried over from the originating environment.

2. The primary or initially contacted MSO region now calls the IBI Subsystem which selects the Load Balancing Group and particular MSO region to route the user to based upon the Logon distribution algorithm described previously. The selected region may be the one initially contacted. If there are no MSO regions with the specified application name, the Logon ends and the user returns to their initial environment with an explanatory message.
3. The Primary MSO region dynamically “passes” the user over to the selected MSO region. The destination MSO region now processes the Logon request normally.

The VTAM user sees the MSO Logon Screen if the minimum required information for Logon to the target region was not specified.

For the VTAM Logon, the minimum information required is Userid and Password if EXTSEC=YES is specified in the MSO Configuration File and Userid is required if EXTSEC=NO.

A security check is done for all Logons specifying the class, APPL, using the entity name of the application name.

At this point Logon processing usually completes and the TOE screen is displayed. However, the Logon request may fail because of resource constraints in the destination MSO region. See the section titled "Planning for MSO Load Balancing".

```

      FFFFFFFF      00000      CCCCC      UU      UU      SSSS
      FFFFFFFF      0000000      CCCCCC      UU      UU      SSSSSS
      FF      00      00      CC      CC      CC      UU      UU      SS      SS      INFORMATION
      FFFF      00      00      CC      UU      UU      SSSS
      FFFF      00      00      CC      UU      UU      SSS      BUILDERS ,
      FF      00      00      CC      CC      UU      UU      SS      SS
      FF      0000000      CCCCCC      UUUUU      SSSSSS      INCORPORATED
      FF      00000      CCCCC      UU      SSSS

      USERID      ==>
      PASSWORD      ==>
      SERVICE      ==>
      APPLICATION      ==>
      APPL GROUP      ==>
      ACCOUNT      ==>
      USER PARM      ==>

      NEW PASSWORD      ==>
      REPEAT NEW PSWD      ==>
      SECURITY GROUP      ==>

      DEFAULT SERVICE      FOCUS
      DEFAULT APPLICATION
      DEFAULT APPL GROUP

      SERVER REGION      PGMRAMM2
      VTAM APPLID      PMRMM203
      TERMINAL      T37CT01B
  
```

PRESS PF3/PF15 TO EXIT

**Figure 4: MSO FOCUS Logon Screen**

- The Destination MSO region calls the IBI Subsystem to update the Load Balancing information stored by the subsystem to reflect the addition of an another MSO user.

## Setting Load Balancing Defaults in the IBI Subsystem

Defaults for both the Load Balancing Group name and the application may be set in the IBI subsystem. These defaults take effect whenever a user accesses an MSO Load Balancing Region without specifying an APPLGROUP name or an APPL name. When these defaults are set in the IBI Subsystem for the group or application name it takes effect for all non specific MSO access requests. The initially contacted MSO region, in this case, does not control the Load Balancing Group that user is placed into.

The Load Balancing Subsystem defaults are set via subsystem commands that are entered as follows:

- MVS Console

```
IBIS SET APPLNAME = applname  
IBIS SET APPLGROUP = groupname
```

- SYSIN to the SUBSYSI job when starting the IBI subsystem or while it is active

```
SET APPLNAME = applname  
SET APPLGROUP = groupname
```

The current IBI subsystem settings for the Load Balancing defaults may be displayed by executing the command in any of the above formats:

```
IBIS DISPLAY LOADBALANCING
```

## **MSO Load Balancing Configuration Parameters**

MSO Load Balancing provides the capability of grouping multiple MSO regions into a logical entity. Dividing the MSO Load Balancing group into subgroups is also possible. This mapping of a MSO Load Balancing group of MVS regions and routing of users into one or more regions within this group is done through the specification of three parameters.

- **APPLGROUP=MSO\_group\_name**

This specification appears once in the global section of the MSO Configuration File and activates the Load Balancing function. All MSO regions that have an identical MSO\_group\_name are grouped together and become selection candidates for MSO users. A user “enters” an MSO group by establishing an initial connection to a MSO region that has APPLGROUP specified. This results in the IBI Subsystem scanning the current MSO group population of MSO regions to properly place the Logon request of the user unless the SERVICE(...) parameter was specified at Logon.

- **SERVICE=name**

Specifying this in a Logon request is mutually exclusive with specifying “APPL(...)”.

This specification is not new with MSO Load Balancing. The functional characteristics of using the Logon parameter SERVICE(...) change when Load Balancing is active for the MSO region. The SERVICE(...) Logon request will not be able to locate the Service Block of the same name unless APPLNAME=NONE has been specified in that Service Block.



The specified Service Block name will only be used if it has PROGRAM=FOCUS specified.

When APPLNAME= is not specified for the Service Block the service name becomes the default application name for that Service Block.

```
APPLNAME=name1
APPLNAME=name2
.....
APPLNAME=name8 /* max of 8 appl names per Service Block */
APPLNAME=NONE |
```

This is a new Load Balancing keyword. It is specified within a service block in the MSO Configuration File. It identifies the Service Block as being a valid Load Balancing Logon destination for those Logon requests that specify an “APPL(name)” that is one of the ones specified. Specifying APPL(..) is mutually exclusive with “service” in the Logon request.

When an application name is specified in the Logon request, a subset of MSO regions within the Load Balancing group is determined on the basis of which ones have a Service Block with the name specified as an “appliance”.

The application name of “NONE” is reserved. If this is specified, the Service Block is not eligible for Load Balancing and is ignored for routing purposes. This is also the only way to be able to connect to a Service Block in a Load Balancing region by ‘SERVICE(...)’ in the Logon request.

**Note:** If APPLNAME is not specified for a Service Block that is in an MSO Load Balancing region, a default application name is assumed for the Service Block using the Service Block name.

## Minimum Parameters for Load Balancing

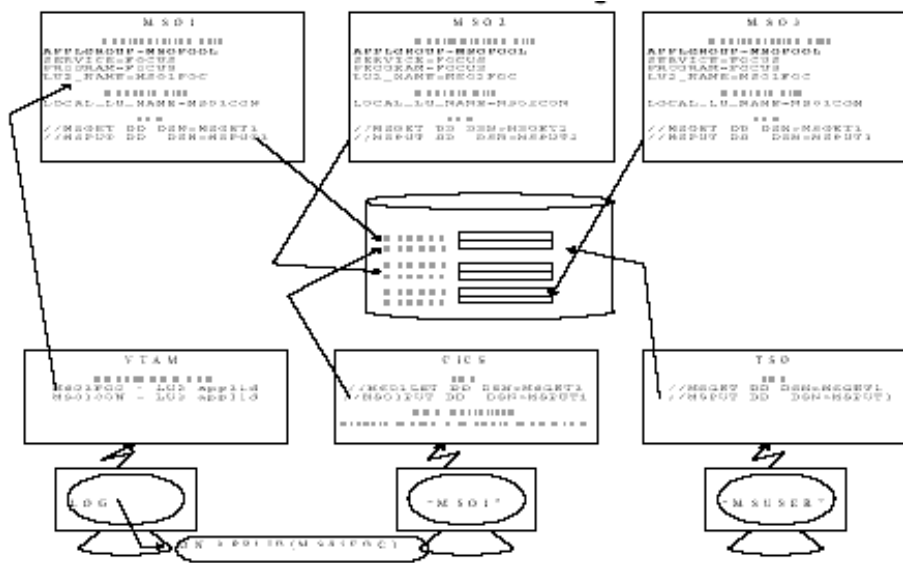


Figure 5: Minimum Parameters for Load Balancing

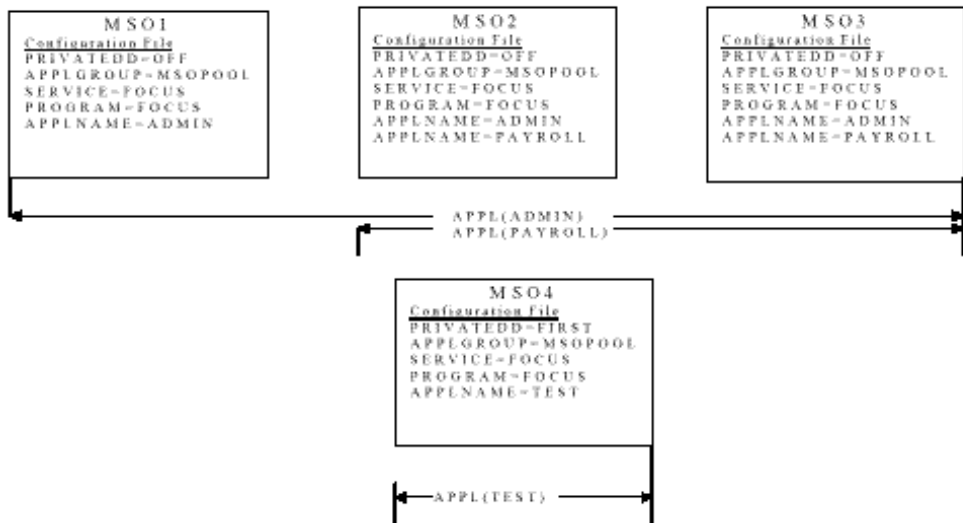
The preceding figure illustrates the minimum entries that must be present in the MSO Configuration File to implement Load Balancing. It also shows the connectivity methodology used by MSO 7.0 for CICS, VTAM, and TSO. The “APPLGROUP=groupname” parameter must be present in all MSO Configuration Files of those MSO regions that are to participate in the same MSO Load Balancing group. This is the **minimum** entry required to activate MSO Load Balancing. The IBI Subsystem must be active at the time that these MSO regions are started.

Notice that “APPLNAME”, a Service Block parameter, is described in the section titled, “Setting Load Balancing Defaults in the IBI Subsystem

MSO Load Balancing Configuration Parameters, do not have to be specified. This works because the IBI subsystem defaults for Load Balancing group and application are used if set, and if not, the default APPLNAME for a Service Block is its Service Name (SERVICE=). The user trying to Logon who does not specify APPL(..) picks up the default APPLNAME of the first Service Block of the MSO region that is initially contacted. If all MSO Users intend to run in the “same” environment, then there is no need to segment the Load Balancing group. APPLNAME then serves no purpose, as all users would be specifying the same application name. This also simplifies the MSO Logon specification for the user and bypasses the requirement in CICS for the installation to supply a parameter list to the CICS MSO transaction when a selection menu is presented to the CICS user; see the section titled, “Load Balancing Logon Procedures”.

In Figure 5: Minimum Parameters for Load Balancing the MSO region, MSO1, is only the primary or initially-connected MSO region. Using a single Load Balancing connection region is not required. Any of the MSO regions could be connected to by the user as the initial connection. Dedicating a single region has the advantage of not having to define the other regions to the connection logic. For the TSO user only a single set of communication data sets has to be defined. This is also true for the CICS region and in addition, only one CICS MSO transaction has to be defined. The VTAM user only has to know a single applid to enter on the LOGON command.

## Segmenting an MSO Load Balancing group



**Figure 6: Segmenting an MSO Load Balancing group**

Figure 6 illustrates an MSO Load Balancing group that has been segmented into several target sub-groups. All of the MSO regions are in the same Load Balancing group. This is indicated by the APPLGROUP parameter in each MSO Configuration File. All four have "MSOPOOL" specified.

It can be advantageous to do this type of segmenting. Frequently there is a need to set up a special environment for a group of users or a particular application. It may be necessary to separate these different environments because of setup conflicts or to reserve certain resources to a group of users or to a particular application.

A user first enters the MSO Load Balancing group by initiating a Logon request to an MSO region that is part of the group. See the section titled “Load Balancing Logon Procedures”. Any of the four MSO regions shown above may be the one that the user initially contacts. The destination or target sub-group of regions is based upon the parameters specified in the Logon request. When that sub-group is determined, an individual MSO region in that sub-group is selected based upon the Load Balancing selection criteria described in section titled “MSO Load Balancing Requirements”

Figure 6 has three separate sub-groups an MSO user can Logon to. The following are the different groups and the parameters that determine them. None of the three sub-groups includes all four MSO regions that make up this Load Balancing group.

1. 3 MSO regions, MSO1 - MSO3

APPLNAME=ADMIN

MSO1 through MSO3 have “ADMIN” specified as one of their eligible application names. When the Logon request specifies APPL(ADMIN), these regions are selected for Load Balancing routing.

This sub-group is also selected if the Logon request does not have an application name. Default processing is to inspect the first Service Block in the initially contacted MSO region. The first application name in the first Service Block is selected as the default application name. When there are no application names defined to the first Service Block, the Service Block name is selected as the default. In this case, ADMIN defaults as the application name.

**2. 2 MSO regions, MSO2 - MSO3**

APPLNAME=PAYROLL

For this sub-group to be selected for Logon routing, APPL(PAYROLL) must be present in the Logon request. There is no way to default to these groups as the default application name for the server block is "ADMIN".

**3. 1 MSO region, MSO4**

APPLNAME=TEST

For this sub-group to be selected for Logon routing, APPL(TEST) must be present in the Logon request when the initially contacted MSO region is not MSO4. When the initially contacted MSO region is MSO4 the default application name is "TEST" as this is the first (and only) application name specified.

In this example, MSO4 is intended to be used as a maintenance or test MSO region. Another new 7.0.5 feature, PRIVATEDD has been set to "FIRST" so a private set of libraries may be allocated to over-ride the general set. See New Feature 501 for a complete description of this new MSO Configuration File parameter.

## Load Balancing Logon Procedures

An initial MSO Logon request may contain any of the following parameters:

- Applgroup - This is the name of the MSO Load Balancing Group that is to be accessed.
- Application - This is the application name used to sub-divide the total MSO Load Balancing group.
- Service - This specifies the name of the Service Block of the initially contacted region. This service block name is only searched for in the initially contacted region. The Service Block will be found only if APPLNAME=NONE has been coded for it.
- Account - This is the MSO account number for this Logon. To validate or alter the supplied account number, use Exit SSUSRACT. MSO.DATA includes this member. A maximum of 40 characters may be specified in this field.
- Userparm - This is an arbitrary string of data that is available in the FOCUS environment using the MSOINFO module. A maximum of 256 characters may be specified for userparm.

The following FOCEXEC lines will retrieve the value that was specified at Logon into the variable, USERPARM.

```
-SET &USERPARM = MSOINFO('USERPARM', 'A256');  
-TYPE USERPARM IS: &USERPARM
```



## Syntax VTAM Logon Request

The VTAM user initiates the MSO Logon by entering the VTAM LOGON command and specifying an applid as follows:

```
LOGON APPLID(msoapplid)
```

The user sees the MSO Logon Screen as shown in Figure 4: MSO FOCUS Logon Screen. At that time the lower right hand section of the screen displays the default service and application. When Load Balancing is turned on for the region presenting the screen, application is the field that controls the connection possibilities for the user. The default application is the APPLNAME= setting for the first Service Block in the MSO region that is displaying the screen.

Only one of the fields, SERVICE or APPLICATION may be specified. If there are no entries in these fields, the Logon request is processed using the default application name displayed on the panel. If using Fast Logon with only the userid specified then the MSO Logon Screen is displayed with both the default service and default application names in the input fields. The user should blank out one of these fields, enter any of the other fields they desire and press Enter.

VTAM users may also use a Fast Logon to specify one or more of the fields that appear on the MSO VTAM Logon Screen. The following is the syntax for VTAM fast Logon:

```
LOGON APPLID(msoapplid) DATA(userid/password/service/account/newpass/  
group/  
application/userparm/applgroup)
```

Parameters not specified are indicated by a double /. Below is an illustration of a Fast Logon request specifying userid, password, account number, application name, and user information.

```
LOGON APPLID(MMMMM203) DATA(MYUSR1/PASSWORD//ACCT99476///PAYROLL/  
YEAREND/MSOGRP1)
```

## Syntax TSO Logon Request.

TSO users must pre-allocate certain data sets and then invoke the “MSUSER” module to connect to MSO. See the *MSO Installation and Technical Reference Guide Release 7.0*, for a CLIST example that allocates the required data sets.

The “MSUSER” module may be invoked as a command or via the CALL command. The format for the command invocations is:

```
MSUSER application,applgroup,service,account,userparm
```

or

```
MSUSER `application,applgroup,service,account,userparm`
```

The fields are positional and must be specified in the order listed. Skipped fields are represented by null commas. To specify only account and user parm use the following syntax.

```
MSUSER `,,,ACCT123,USERDATA`
```

The parameter string must be enclosed in single quotes if initial input fields are skipped. The commas are unconditional field delimiters and may not be specified as part of the data. Single quotes may be included within the data by entering two single quotes. Entering single quotes around a comma insures that the comma is not treated as data.

The format for invoking MSUSER with the CALL command is below. The parameter string must always be enclosed in single quotes. The parameter parsing is the same as described for the command invocation above.

```
CALL 'prefix.FOCLIB.LOAD(MSUSER)'  
'application,applgroup,service,account,userparm'
```

## **Syntax** CICS Terminal Logon Request

The IBI provided module, MSOCICS, now provides the capability of specifying the same four parameters. The format for the command is:

```
MSO APPL(application_name) | SERVICE(service_name)  
    APPLGROUP(groupname)  
    ACCT(account_information)  
    USERPARAM(user_information)
```

The MSOCICS module must have a CICS transaction ID assigned to it. The screen should be cleared before entering the MSO transaction.

## **Syntax** CICS Load Balancing API

Those installations that call the IBI MSOCICS transaction from a program must pass the control block structure illustrated below. The CICS-MSO Interface block should be passed in the CICS commarea when the MSO module MSOCICS is invoked via EXEC CICS XCTL. The storage for the second block, CICS - MSO Logon Buffer, should be obtained using EXEC CICS GETMAIN.

Prior to entering data into any of the fields in this structure, the Interface Block should be initialized to nulls (X'00') and the Log buffer should be initialized to blanks (X'40').

## MSO-CICS Interface Block

+0	func set to x'01'	msoname	reserved	reserved
+10	reserved			
+20	logbuf->	reserved		
+30	reserved			
+40	reserved			
MSO-CICS Logon Buffer				
+0	application name		service name	
+10	reserved			
+20	group name length of X'08'		reserved	
+30	reserved			
+40	account length of X'28'			
+68	user parm length of X'100'			

The field contents for these two control blocks are:

Field Name	length	type	contents
func	4 bytes	numeric	1 - signifies MSO Logon

msoname	4 bytes	character	MSO CICS transaction name
logbuff	4 bytes	31 bit address	Pointer to MSO log buffer
application	8 bytes	character	Load Balancing application name
service	8 bytes	character	Service Block name
group name	8 bytes	character	Load Balancing Group name
account	40 bytes	character	MSO account information
user parm	256 bytes	character	User information available to executing FOCEXEC

## Planning for MSO Load Balancing

The most important consideration in planning for Load Balancing is to make sure that when users “arrives at their destination” it is where they expected to be. MSO users expect the characteristics of their environment to be the same **every** time they Logon with the same application name. Application names are merely addresses to MSO Load Balancing. They are a way of ensuring that the user is serviced by an MSO region that has the application as a valid address. The actual names have no meaning beyond this.

## **MSO Load Balancing Requirements**

The IBI Subsystem must be active and the MSO program libraries must be APF authorized to support this feature.

## **IBI Load Balancing Defaults**

When the Load Balancing defaults are set in the IBI Subsystem for the group or application name it takes effect for all non specific MSO access requests. The initially contacted MSO region, in this case, does not control the Load Balancing Group that user is placed into.

## **Consistent Destination Environment**

All MSO regions running the same applications should have the same operating environment. The same application must run identically in each region.

For planning purposes, the following considerations affect the execution characteristics of an MSO region.

- **Global Allocations**  
The libraries allocated for FOCUS libraries should be identical. If different Master File allocations are made, for example, then the same application may behave differently.
- **Profile Data Sets**

The need for profile data sets is reduced with the availability of Load Balancing. Profiles tailor the environment when users were logging on to the same MSO region. If used, it is imperative that all MSO regions in a target sub-group have identical profiles for each user. The easiest way to effect this is to allocate the same MSO profile data set.

- **Configuration Parameters**

The following MSO Configuration File parameters affect the execution environment of an MSO region. They should be identical across Load Balancing sub-groups.

- **EXTSEC**

When set to NO, all users have the same security access as the MSO region in which they are executing. When set to YES, each user executes with the established security for the userid.

- **PRIVATEDD**

This parameter effects the logical search order of FOCUS libraries and files. It is a new feature with MSO 7.0.5. New Feature NF501 has a description of its operation

- **MSO Exits**

Ensure that all MSO exits are identical.

## **Define Load Balancing Group Names to SAF**

Define Load Balancing APPLGROUP names as entities in the APPL class. There is a SAF READ access check based upon the APPLGROUP name for jobs attempting to start a MSO Load Balancing Region.

The IBM RACF ® product does not fail the request for an undefined entity. However, the security check is effectively nullified.

The CA ACF2 ® SAF product fails a request for an undefined entity. In this environment, Load Balancing requires defined application names and users with READ access.

## **Consistently use the APPLICATION= parameter in all Service Blocks**

When the APPLICATION= parameter is introduced into a MSO Load Balancing group all Service Blocks should have this parameter specified. This avoids confusion resulting from the application name defaulting the Service Block name when the parameter is not used.

## **MSO Region Capacity**

The MAXIMUM setting in the MSO Configuration File for each MSO region should be conservative as compared to the amount of virtual storage that is available below and above the 16 megabyte line. This avoids the operational difficulty of having a MSO Logon request denied for a particular region when less than the minimum amount of storage is left above or below the 16m line.

## **All MSO Regions should have a Console available**

The console provides a look into the MSO region showing very useful information about the MSO users' activities. It is a valuable aid in problem determination.



## Minimize dividing the MSO Load Balancing group

To make administration easier, keep the number of subsets of MSO regions in an APPLGROUP to a minimum. The less subsetting, the more effectively the Load Balancing algorithm can balance Logons across MSO regions.

## Operational Trouble Shooting

- Why don't my users end up in the regions in which I expect them?  
Ensure all the regions have the same name specified for APPLGROUP.  
If the Logon specifies an application name, make sure the name is in a Service Block in each MSO region you want available to the MSO Load Balancing group.  
If the Logon does not specify application name ensure that all targeted MSO regions in the group have the same service name as specified for the first Service Block in the initially contacted MSO region.  
If the logon does not specify a Load Balancing Group name or application name, then the defaults may be set in the IBI Subsystem. Issue the subsystem command:  

```
IBIS, DISPLAY LOADBALANCING
```

  
This will show the current default settings. Once default settings are set in the IBI subsystem, all access requests that are not supposed to use these defaults will require that the group and application name be specified.
- The distribution of users does not seem to be even across MSO Load Balancing regions.

Load Balancing only has an effect at Logon time. If users FIN out of one or more regions faster than the rate of new users logging on, then the distribution will become unbalanced.

Dividing the Load Balancing group can produce imbalances across MSO regions in the APPLGROUP.

- One region is rejecting all MSO Logons and users cannot get on even when other regions are available.

The placement algorithm for Logon routing MSO users into load balanced regions does not guarantee the region can handle it. If one region becomes overstressed due to the amount of activity but it has the lowest number of users, users are routed to it and fail in Logon. This can be remedied in one of two ways:

1. Start a new MSO region with identical parameters to the target set that is having problems. A region is then available with 0 users logged on. All new users are routed to this address space.
  2. Issue the MSO QUIESCE command for the region rejecting Logons. This region then removes itself from eligibility in the Load Balancing group. The other regions are selected for Logons.
- My payroll application fails in one Load Balancing region.  
Ensure that the environment for the failing region is identical to the region in which the application works properly. Before you make any changes, review the other applications assigned to that region making sure the correction will not affect those applications adversely.
  - A user is having a problem. How do I find out in which MSO region they are executing.

Have the user execute a FOCEXEC that calls the MSOINFO module. This module now returns the application name in addition to the service block name and server name. The following lines in a FOCEXEC accomplish this:

```
-SET &SERVER = MSOINFO('SERVER ', 'A8');  
-TYPE SERVER JOBNAME IS &SERVER  
-SET &SERVICE = MSOINFO('SERVICE ', 'A8');  
-TYPE SERVICE BLOCK NAME IS &SERVICE  
-SET &APPLICAT = MSOINFO('APPLICAT', 'A8');  
-TYPE APPLICATION NAME IS &APPLICAT  
-SET &APPLGROUP = MSOINFO('APPLGROUP', 'A8')  
-TYPE GROUP NAME IS &APPLGROUP
```

The first parameter in the invocation of the MSOINFO subroutine must be padded to 8 characters

- A Logon request with SERVER(FOCUS) is being rejected with the message “LOGON FAILED: SPECIFIED SERVICE UNKNOWN” when there is a Service Block with FOCUS as its service name.

The ability to access a Service Block with the SERVER(...) parameter on an MSO Logon request becomes disabled when Load Balancing is active in the region. The Service Block can only be reached this way if APPLNAME=NONE is specified and the region is initially contacted. Setting Load Balancing Defaults in the IBI Subsystem

## CICS Problem Determination Procedures

MSO CICS 7.0 users may capture a dump of an IBI internal trace file. CSS researchers or IBI programming may request this trace for problem determination. This function requires additional DDnames be allocated to the CICS region. FOCDMP00 through FOCDMPnn may be allocated. At any time a CICS user may issue the following command to cause the current contents of the MSO CICS trace to be written to the selected DDname.

```
MSMT DUMP(*|transaction_name) DDSUFFIX(nn)
```

where

\*

Denotes printing the trace information for all MSO regions connected to this CICS region. Specify the associated MSO transaction ID or the MSO region you wish to trace to limit the listing to that one connection.

DDSUFFIX

Specifies the corresponding FOCDMPnn pre-allocated file to which the trace information is written. DDSUFFIX defaults to 00.

Allocate the FOCDMPnn data sets to a SYSOUT class or to a sequential file of 4 tracks. These data sets should have an LRECL of 102.

## **Operational Benefits of MSO Load Balancing**

The primary benefit derived from MSO Load Balancing is the ability to evenly distribute MSO Logon requests across multiple MSO regions. This makes monitoring of MSO regions less costly to administrative personnel. It also ensures that MSO Users will get more even handed performance characteristics when they Logon to MSO.

There is more flexibility for the MSO Administrator in controlling how many resources to give to MSO Users. The connection to all but the primary or initially contacted MSO region need not be known by the connecting environments, including the primary MSO region. Additional MSO regions may be started via the console or by submitting a job. These new regions must have their own unique connection data sets allocated (MSGET/MSPUT) and VTAM applids if they are to be accessible from VTAM. However, they become automatically available to the APPLGROUP when they have completely initialized.

The connecting environments, TSO, CICS, and VTAM only have to be set up to connect to the primary or initially contacted MSO region.

- The TSO CLIST/EXEC only has to allocate the MSGET/MSPUT communication data sets of one of the MSO regions in the MSO Load Balancing group.
- The VTAM Logon command/session manager screen only has to specify the MSO applid (lu2\_name in the MSO Configuration File) of one of the MSO regions in the Load Balancing group.

- The CICS regions that connect MSO Users to the MSO Load Balancing group only have to have one pair of communication data sets allocated. In addition, only one MSO transaction id has to be defined.

## MSO Glossary of Terms

MSO	Multi-Session Option, is an IBI product offering that provides a multi-user MVS address space for executing FOCUS commands and applications.
region	An alternate term for an MVS address space
Load Balancing	An MSO feature that automatically routes an MSO user to one of several MSO regions.
Group, Load Balancing	A named collection of MSO regions that comprise a Load Balancing group.
Primary or initially contacted MSO region	An MSO user initiates a Logon request to a Load Balancing group by issuing a normal MSO Logon to one of the MSO regions in the group. This first step is referred to as the primary MSO region or the initially contacted region.
destination region	This is the MSO region that the user is routed to by the MSO Load Balancing algorithm.

MSO Configuration File	This file is a text file that contains the startup specifications for the MSO region.
Communication data sets	A pair of sequential files that are dedicated to an individual MSO region. These data sets provide the basis for access to MSO for TSO and CICS users.
Service Block	The MSO configuration file must have at least one Service Block defined that specifies PROGRAM=FOCUS. This is the basic definition of the MSO FOCUS server. In most MSO environments only one service group is defined per MSO region.
Service Name	Each Service Block definition is started by specifying SERVICE=service_name. This name is referred to as the service name.
Application name	A new Service Block keyword to support Load Balancing. It associates a meaningful name such as "PAYROLL" or "ACCT" with one or more MSO Load Balancing regions in an MSO Load Balancing group.

Sub-Group or Target  
Group

A subset of MSO regions in an MSO Load Balancing Group. This subset is mapped based upon the application name specified in the Logon request and is treated as the total MSO group of regions eligible for the user.



## NF576: MSO Dynamic VTAM Re-configuration

MSO installations can re-configure the VTAM LU2 applid used for MSO FOCUS access. This allows them to dynamically switch to a newly specified LU2 applid without recycling the MSO region.

### Usage

The SET LU2\_NAME command is issued at the MSO console or via the modify command at the MVS system console. This conditions the MSO region to use the new LU2 applid when the RESTART LU2\_NAME command is issued from the MSO console or the MVS system console. When the RESTART LU2\_NAME command is issued, current MSO FOCUS sessions are terminated. The newly specified LU2\_NAME is then activated for MSO FOCUS logons. All sessions connected to the MSO region from TSO or CICS are unaffected.

It is recommended that all MSO console commands be issued while viewing the MSO LOG. The command's effect is then immediately visible by pressing ENTER.

### Syntax      How to Use a new VTAM LU2 Applid

You can specify a new VTAM LU2 applid either from the MSO console or the MVS system console.

From the MSO console:

1. Issue the following command to condition MSO to use a new applid

```
SET LU2_NAME = applid_name
```

2. Issue the following command to switch to the new applid

```
RESTART LU2_NAME
```

From the MVS System Console

1. Issue the following command to condition MSO to use a new applid

```
F msojobname,SET LU2_NAME = applid_name
```

2. Issue the following command to switch to the new applid

```
F msojobname,RESTART LU2_NAME
```

### Note:

- The keyword LU2\_APPLID is synonymous with LU2\_NAME and the two may be used interchangeably.
- DYNAM re-configuration of the MSO Console LU2 applid has always been available using the RESTART CONSOLE command. The “local\_lu\_name” keyword in the MSO console configuration file must be changed to the new LU2 applid prior to issuing the RESTART CONSOLE command and the new applid will take effect for the MSO console.

### Examples

Not Applicable

## Reference Special Considerations

The VTAM LU2 applid name specified in the SET LU2\_NAME command must be an active LU2 applid that is defined with the following characteristics in SYS1.VTAMLST:

```
FOCUS APPL AUTH=(VPACE,PASS),EAS=10,PARSESS=YES
```

**Note:** The PASS specification is only required if the Load Balancing feature of MSO will be used.

## Reference Error Messages

MSO13360 INVALID KEYWORD IN SET COMMAND

The keyword in the SET command is unknown. Please check the spelling of the keyword and be sure our documentation matches the version of the program you are using.

MSO13361 INVALID OPERAND IN SET COMMAND

A valid SET command keyword has an invalid operand. Please check the spelling of the operand and be sure your documentation matches the version of the program you are using.

MSO13362 LU2\_APPLID IS NOW SET TO %1

The specified new value for LU2\_APPLID is now in effect.

## NF578: FOCUS Personal Agent - Interruptible Server for VM and MVS FOCUS

FOCUS Personal Agent allows users and application developers to use FOCUS For Windows as a graphical user interface for generating code and producing styled Reporting and Graph applications that run against character-based versions of FOCUS on a number of platforms, including VM/CMS and MVS/TSO. When a large report request is run against data on the host via FPA, the interruptible server feature allows users to monitor, terminate, or display partial data as the request is processed on the mainframe. It is also possible to interrupt a request as the data is being downloaded from the mainframe to the PC. This provides a powerful way to govern large data queries and prevent runaway requests. The feature is invoked via a window from the Personal Agent product on the PC.

The interruptible server is a new feature in FOCUS Personal Agent Release 6.11 and is available only for VM/CMS and MVS/TSO FOCUS Releases 7.0.6 and above, and only when connected via Personal Agent's TCP/IP option (see also NF543 and NF544).

## Reference Usage

Invoke Personal Agent's Interrupt Menu by selecting "Interrupt..." from the ACTIONS menu, or hitting CTRL-C when the FPA window is active. The Interrupt Menu offers four options: Terminate Request, Display Partial, Pause, and Statistics. These four options are only active when it is possible to interrupt a request. Their functions are described in the Syntax section below and are only available when running a request in Personal Agent's CLIENT\_AND\_SERVER mode.

Requests can be interrupted at two stages:

- During processing on the host (VM/CMS or MVS/TSO): In this case, the Personal Agent icon displays a Blue box in the upper left corner signifying that processing is occurring on the mainframe. At this point, a request may be interrupted as the data is being retrieved.
- While the answer set is being downloaded from the host to the PC: In this case, the down arrow on the Personal Agent icon is red indicating data is being downloaded. The FPA window displays the number of records being downloaded and the percent downloaded thus far (i.e., 25 of 100 records, 25%). This download process can be interrupted by Personal Agent.

## Syntax How to Interrupt a Request

During processing on the Host and during the downloading of data, the following options are activated on FPA's Interrupt menu:

- Terminate Request:

This option immediately aborts a request. During processing on the host or downloading, the request is terminated and control is returned to the client (FOCUS for Windows).

- Display Partial:

During processing on the host, this option aborts the report request and downloads data accessed up to that point. During downloading of data, the Display Partial option stops the download operation and displays only data retrieved up to that point. (**Note:** When Display Partial is selected, the other interrupt options are deactivated.)

- Pause:

This option temporarily stops a request during processing on the host or during the downloading of data. When the PAUSE button is selected, the request stops and the button turns into a CONTINUE button. While the request pauses, you can select one of the other three options, or select CONTINUE to resume processing.

- Statistics:

This displays numbers of Lines and Records in a window on the Interrupt menu during processing on the host. These numbers represent the Lines and Records processed up to that point. It is important to note that processing continues after the Statistics are displayed, so you may wish to hit PAUSE first and then select STATISTICS to get a precise number of records accessed so far. As data is downloaded, the FPA window displays the number of records being downloaded and the percent of data downloaded so far (i.e., 25 of 100 records, 25%). The STATISTICS button is not active during the download process.

## **Examples**

N/A

## **Special Considerations**

N/A

## **Error Messages**

N/A

## Project 2000 - Phase II

The second phase of the cross-century dates feature expands the sliding window technique to include settings at the file and field level of applications. Four new settings were added for Master File Descriptions: FDFCENT and FYRTHRESH at the file level, and DEFCENT and YRTHRESH at the field level.



## 7.0.5 New Features

### Year 2000

[NF497: Project 2000—Cross-Century Dates in FOCUS Applications](#)

### Performance Enhancements

[NF500: Keyed Retrievals from FOCUS Extract Files](#)

[NF509: MINIO - New FOCUS Database Access Method Available Under MVS](#)

### Reporting Enhancements

[NF499: Scrolling Report Headings in HotScreen](#)

[NF510: Date and Time Stamp in Reports](#)

### Maintain

[MAINTAIN Improved SU Performance](#)

[CDN Support](#)

[NF518: VSAM Support](#)

[NF519: Field Level PFKeys](#)

[NF520: Dynamically Changing Attributes of MAINTAIN Winforms](#)

[NF521: Enhancements to Objects in MAINTAIN](#)

## General Enhancements

[NF494: FOCUS Client: Remote Data Access via EDA/SQL](#)

## The Multi-Session Option

[NF493: Full Support for SDSF under MSO and TSO](#)

[NF501: Public and Private DDname for MSO](#)

## Interfaces

[NF513: Redefining Fields in Non-FOCUS Files](#)

## FOCSAM Interface

[NF518: VSAM Support](#)

## National Language Support

[National Language Support](#)

## NF493: Full Support for SDSF under MSO and TSO

FOCUS supports viewing of batch job output through SDSF. The new SDSF command is available for both MSO and TSO FOCUS users. For some time, you have been able to submit batch jobs from within TSO and MSO using the DYNAM Submit command. Now you can view the output of your batch job from the SDSF main panel.

From an MSO standpoint this new feature eliminates the need for TSO, ISPF and SDSF. This provides an easy transition to MSO for TSO users.

### **Syntax**     **Accessing the SDSF Main Panel**

To gain access to the SDSF main panel under MSO or FOCUS, enter the following command at the FOCUS prompt.

```
{MVS|TSO} SDSF
```

### **Procedure**   **How SDSF Establishes Job Control Values for MSO**

To SDSF, MSO is a batch job. SDSF establishes the following values for that job.

USERID	The userid SDSF uses comes from the (ACEE) built at logon time by the security package. If there is no security package installed, SDSF uses the jobname minus the last character.
--------	--

PROCNAME	The TSO logon procedure name (PROCNAME) is BATCH.
Terminal name	The terminal name is BATCH.
TSOAUTH	The TSOAUTH parameter in the ISFPARMS module is JCL.

This information is provided to help the person responsible for SDSF establish appropriate security levels for users.

## NF494: FOCUS Client: Remote Data Access via EDA/SQL

This bulletin discusses FOCUS client/server computing, including elements of Information Builders' Enterprise Data Access/SQL (EDA/SQL) family of middleware products. An understanding of the EDA/SQL components is essential for those planning to implement client/server applications.

### Client/Server Computing and Middleware

Client/server computing is the dominant architecture today because it enables sites to exploit the power of networked computing systems. Heterogeneous in nature, client/server architecture divides application components such as screen formatting, program logic, and data access between several networked processors to exploit unique characteristics in each environment:

- A client, such as mainframe FOCUS, typically builds a request and specifies a report layout.
- A remote server, or back-end, then performs data selection and handles data integration and aggregation.

By enabling numerous front-end tools to share centralized back-end services, client/server computing introduces the concept of *interoperability*. Seamless integration of the front- and back-end processes is accomplished through a new layer of systems software, called *middleware*, which provides interoperability by delivering transparent data transmission and translation services between physically linked processors.

EDA/SQL middleware insulates end users and application developers from dealing with the complexities and incompatibilities of networked proprietary computing environments. This is accomplished through three EDA/SQL components:

- **The Application Programming Interface (API).** This client component, built into mainframe FOCUS, requests remote services using SQL requests.
- **The Database Server or Gateway.** This back-end server or gateway translates remote requests into formats suitable for the specified target environment and executes them.
- **Network Communications.** Network communication services provide protocol translation services masking incompatibilities between proprietary networks and interconnected systems.

The communications system and protocol subsystem, which together make up Network Communications, shield the API and server from details of various communications protocol syntaxes. Each protocol subsystem is a platform-specific interface to a supported communications protocol. Together, these components enable applications to send requests to a variety of servers and to receive answers (data or messages) in return.

## Overview: Using FOCUS to Access Data on EDA/SQL Servers

This section describes processing alternatives when using FOCUS to access data on EDA/SQL servers:

### **Remote Execution**

This approach allows you to read, analyze, consolidate, and update remote data by sending the FOCUS stack to an EDA/SQL server on another platform.

You can also execute FOCEXECs stored on the server (called Remote Procedures), using a process known as a Remote Procedure Call (RPC).

In remote execution, the EDA/SQL server typically handles data processing, while request generation and data presentation are done on the client.

### **Distributed Execution**

In Distributed Execution (also referred to as the EDA Interface), data retrieved from the server is processed on the client. Here, the server is primarily involved with data access. Distributed Execution shields end users from needing to know where data actually resides via a mechanism known as *location transparency*.

## Establishing and Configuring the FOCUS User Environment

FOCUS users can employ startup FOCEXECs to logon to a remote server and establish environmental conditions for a session. PROFILE FOCEXECs can also establish environmental conditions or build menu shells of user options.

On the server, the PROFILE FOCEXEC can establish the working environment for the EDA/SQL Server for MVS—for example, setting WIDTH and PANEL parameters for reporting.

## **Prerequisites**

To use EDA/SQL with mainframe FOCUS, you must have the following products installed:

- EDA/Link, Release 2.2 or higher.
- An EDA/SQL MVS or VM Server, Release 2.0 or higher.
- As a client, you can use mainframe FOCUS Release 6.8/9404 or higher, or FOCUS Release 7.0.5 or higher. Depending on your method of remote execution (described in the following major sections), you also need an EDA configuration file.

## **The EDA Configuration File**

In FOCUS 6.8 you must have an EDA configuration file with ddname CONFIG to use Remote Execution, and a configuration file with ddname EDACFG, to use Distributed Execution (also called “SUFFIX=EDA”). These are mutually exclusive alternatives, and you cannot switch back and forth between Remote and Distributed Execution modes during a FOCUS 6.8 session.

In FOCUS 7.0, the EDACFG ddname applies for both the Remote Execution and Distributed Execution modes, and it is possible to switch back and forth between them within your session.

In VM, you issue a FILEDEF to name the EDA configuration file. In MVS, you allocate the ddname for the file containing EDA configuration information to either a sequential file or a member of a partitioned dataset.



## Remote Execution

Remote Execution allows users to transmit local FOCUS requests to a remote host for execution. In this operating mode, a request, built on the client, is shipped to the EDA/SQL server for execution. On completion, output is returned to the client and displayed in Hotscreen. All data processing is done on the server, while request generation and data presentation are handled by the client. Remote Execution also supports execution of server-based requests (Remote Procedures), through Remote Procedure Calls (RPCs).

Before sending a request to a remote server, you must connect to it, by either:

- Issuing the REMOTE DESTINATION, USERID and PASSWORD commands at the FOCUS prompt or
- Creating a FOCEXEC that issues the REMOTE DESTINATION, REMOTE USERID, and REMOTE PASSWORD commands for you.

**Note:** All files supported by EDA/SQL servers can be accessed using Remote Execution.

## Logging on with REMOTE Commands

To have FOCUS automatically log you on to an EDA/SQL Server, create a FOCEXEC containing the REMOTE DESTINATION, REMOTE USERID, and REMOTE PASSWORD commands.

## Syntax     Specifying a Target Server: REMOTE DESTINATION

The REMOTE DESTINATION command directs requests to a particular server. The syntax is

```
REMOTE DEST[INATION] = entity
```

where:

*entity*

Is the EDA/SQL server name and is taken from the client configuration file. It must match the ENTITY keyword in the EDA/SQL configuration file on the server.

## Syntax     Identifying the User: REMOTE USERID

The REMOTE USERID command sets the user ID. When FOCUS issues a request, it sends the user ID to the security system on the server (for example, RACF on MVS). The syntax is

```
REMOTE USERID = serveruserid
```

where:

*serveruserid*

Is your user ID.

This user ID remains in effect until you reissue the REMOTE USERID command.

## Syntax     Authenticating the User: REMOTE PASSWORD (USERPASS)

The REMOTE PASSWORD command sets the user password. When FOCUS issues a request, it sends the password to the security system on the server (for example, RACF). The syntax is

```
REMOTE PASSWORD = serverpassword
```

where:

*serverpassword*

Is your password.

This password remains in effect until you reissue REMOTE PASSWORD.

## Sending Requests to a Remote Server

You can send FOCUS requests to the server for execution as follows:

- Use the REMOTE EX command from the FOCUS Session prompt to name a FOCEXEC on the client that you want to execute on the server.
- Include the commands -REMOTE BEGIN and -REMOTE END around the code you are sending to the server. Then execute the procedure as you would any other FOCUS procedure. Here again, expansion of Dialogue Manager amper variables takes place *on the client*, before the request is forwarded to the server.
- Use TableTalk to create the request and select the 'Execute on Remote Server' option on the final TableTalk screen.

After executing a report request, FOCUS displays the report in Hotscreen.

## Syntax      How to Execute a Request Remotely Using the REMOTE EX Command

Issue the following command at the FOCUS Session prompt

```
REMOTE EX focexecname
```

where:

*focexecname*

Is the name of the FOCEXEC. Note that Dialogue Manager amper variables in the FOCEXEC will be expanded *on the client* before the request is shipped to the server.

**Important!** A request executed this way may not contain the commands -REMOTE BEGIN and -REMOTE END. Those commands are issued automatically by REMOTE EX.

### Example      Executing a Request Remotely Using the REMOTE EX Command

The following sample FOCEXEC (EDHOURS) is executed at the FOCUS Session prompt with the REMOTE EX command. For this example, the actual FOCEXEC is displayed below the EX command, by which it is invoked:

```
>> REMOTE EX EDHOURS
```

```
TABLE FILE EMPLOYEE
  HEADING CENTER
  "SUMMARY REPORT OF EMPLOYEE CLASSROOM HOURS"
  "  "
  SUM ED_HRS BY EMP_ID
END
```

## Using -REMOTE BEGIN and -REMOTE END to Execute Requests Remotely

Another way to execute report requests against remote data is to append -REMOTE BEGIN in front of the FOCEXEC that you want executed on the server and follow your END statement with -REMOTE END. Then execute the focexecname.

Using -REMOTE BEGIN and -REMOTE END provides three advantages:

- The remote procedure syntax is the same that you would use locally.
- By adding -REMOTE BEGIN and -REMOTE END, developers can segment FOCEXECs into separate components; each of which may be executed remotely or locally.
- You do not need to have a Master File Description for the target file on the client with this approach.

Keep the following restrictions in mind:

- When developing applications (that is, when coding FOCEXECs), you can use multiple -REMOTE BEGIN and -REMOTE END pairs within the FOCEXEC. You cannot nest -REMOTE BEGIN and -REMOTE END pairs.

- If a -RUN command falls between a -REMOTE BEGIN and -REMOTE END command pair, it's treated as if it were a -REMOTE END, followed immediately by a -REMOTE BEGIN. All FOCUS commands preceding the -RUN are sent up to, and executed on the server. Commands following the -RUN are placed on the FOCUS stack when control returns to the client and are executed on the server when the -REMOTE END is encountered.
- You cannot mix REMOTE commands within a FOCEXEC. For example, you cannot use the REMOTE EX command for a FOCEXEC that contains -REMOTE BEGIN and -REMOTE END commands. This restriction applies to other FOCEXECs executed from within your main FOCEXEC by -INCLUDE or EX commands.

### Example Executing Requests using -REMOTE BEGIN and -REMOTE END

The example below demonstrates the use of a Dialogue Manager amper variable (&1) with FOCUS commands in a FOCEXEC named MARGIN. The request executes a JOIN and report request on the server. The numbers on the left refer to the notes that follow.

1.        -REMOTE BEGIN
2.        JOIN PROD\_CODE IN &1 TO PROD\_CODE IN PROD AS AJOIN  
TABLE FILE &1  
PRINT UNIT\_SALES  
AND COMPUTE  
MARGIN/D8.2= RETAIL\_PRICE - UNIT\_COST;  
BY STORE\_CODE BY PROD\_CODE  
END
3.        -REMOTE END

1. -REMOTE BEGIN identifies the beginning of the command stream to be executed on the server. Any commands already sitting in FOCUS stack are executed when -REMOTE BEGIN is encountered.
2. The MARGIN procedure includes FOCUS commands and an amper variable, which is expanded on the client when used with -REMOTE.
3. -REMOTE END identifies the end of the command stream to be executed on the server.

You execute MARGIN and provide the name of the target database as an argument on the command line. For example, to obtain the margin report for the SALES file, issue the command

```
EX MARGIN SALES
```

at the FOCUS Session prompt. SALES is substituted for the Dialogue Manager variable &1 in the JOIN and TABLE commands, and the commands then execute on the server. The resulting report is returned by the server and displayed in Hotscreen on your terminal.

## Syntax      How to Execute Remote Procedures

In addition to making data available to the client, an EDA/SQL server can also hold *remote* or *stored procedures* that can be executed from FOCUS.

To call a procedure that resides on the server, use the following technique:

```
>> REMOTE EX focexec
```

where:

*focexec*

Is a FOCEXEC on the client that contains a command that executes a procedure on the server. For example::

`EX server_focexec` where `server_focexec` is a FOCUS procedure stored on the server.

Another way to do this is, is to issue the following commands:

```
-REMOTE BEGIN  
EX server_focexec  
-REMOTE END
```

You may use a `-INCLUDE` within a FOCEXEC to call another FOCEXEC stored on the client.

### **Example** Using `-INCLUDE` to Call a FOCEXEC Stored on the Client

This example downloads data from a host to a FOCUS client and updates the EMP file.

```
-REMOTE BEGIN  
TABLE FILE EMPLOYEE  
  PRINT EMP_ID SALARY START_DATE  
  ON TABLE HOLD AT CLIENT AS LD  
END  
-REMOTE END  
-INCLUDE FOCEXEC
```



The FOCEXEC contains the following commands:

```
MODIFY FILE EMP
  FIXFORM FROM LD
  DATA ON LD
END
```

**Note:** If you use Remote Execution to execute a FOCEXEC on the server by employing -INCLUDE, the FOCEXEC named must reside on the server and the -INCLUDE statement must precede the -REMOTE END statement.

## Viewing System and Error Messages

All FOCUS and system messages returned by the EDA/SQL server are displayed, as are error messages resulting from Remote Execution. These latter (FOCNET) messages consist of information specific to the interaction of FOCUS and EDA/SQL, and are of the form

```
FNTnnnn
```

where:

```
nnnn
```

Is a number in the thousands range.

For example,

```
FNT32042
FOCNET session establishment failed for client
```

FOCUS places the FOCNET error message numbers in the Dialogue Manager variable &FOCERRNUM, which can then be tested in FOCEXECs. You could also issue '? 32042' to see the error message.

## Terminating the Remote Session: REMOTE FIN

The REMOTE FIN command logically terminates a FOCUS session with an EDA/SQL server. It should be issued at the conclusion of remote data access. The server must be available when you issue this command. (Note: Entering a FIN command in native FOCUS will close all active sessions.)

### Syntax      How to Terminate a Remote Session With the REMOTE FIN Command

`REMOTE FIN entity`

where:

`entity`

Must match the server name in the CONFIG or EDACFG file

## Checking on Remote Session Parameter Settings: ? REMOTE

The ? REMOTE command prompts for a display of all remote session parameters in force. Issue it at any time in your FOCUS session.

### Syntax      How to Check Remote Session Parameter Settings With ? REMOTE

`? REMOTE`

The resulting screen shows:

```
Remote Destination - - > EDASERVE
Remote User Id     - - > EDARTW
Remote User Password - - > . . . . .
Remote Execution Mode - - > IMMEDIATE
Conversations exist with :
HOSTNAME = EDASERVE
```

## Distributed Execution: The EDA Interface (or “SUFFIX=EDA”)

With the EDA Interface, or Distributed Execution, you can access all data files and tables accessible to an EDA/SQL server. Your Master File Descriptions and Access File Descriptions tell FOCUS where to access the data.

The EDA Interface provides two advantages:

- Once you have set up your Master and Access File Descriptions on the client, you can use the FOCUS tools to access remote data just as if it were local data.
- You can also join files across platforms. For example, join a file on MVS to a file on a UNIX platform.

### Syntax      How to Implement Distributed Execution: Location Transparency

When you include the following attribute in a file description:

```
SUFFIX=EDA
```

it directs FOCUS to pass all requests for that file directly to the EDA Interface, which passes them on to an EDA/SQL server. You can establish the name of the target server in either of the following two ways:

- Store the name of the target server in the Access File Description. The syntax in the Access File is

```
SERVER = servername
```

where:

```
servername
```

Is the name of the target server.

The ddname of the Access File Description is "FOCSQL."

- Issue the following command:

```
SQL EDA SET SERVER servername
```

A server name in an Access File Description always overrides any name specified in a SQL EDA SET SERVER command. By removing the SERVER parameter from the Access File Description, you can dynamically control the server location with SQL EDA SET SERVER commands.

On VM, you specify an Access File name that matches your Master File name, with a file type of FOCSQL. For example:

```
      CAR MASTER A - Master File Description  
CAR FOCSQL A - Access File Description
```

On MVS, your Access File membernames must also match your Master File Description membernames and must reside in a partitioned dataset allocated to ddname FOCSQL. For Example

```
DYNAM ALLOC F1 MASTER DS userid.MASTER.DATA SHR REU
DYNAM ALLOC F1 FOCSQL DS userid.FOCSQL.DATA SHR REU
```

where:

```
userid.MASTER.DATA(CAR)
```

Is the Master File Description

```
userid.FOCSQL.DATA(CAR)
```

Is the Access File Description

### **Example** Storing a Server Name in an Access File Description

The following example shows how to store server name IBMSERVE in an Access File Description.

```
SEGNAME=ONE , TABLENAME=CAR , KEYS=1 , WRITE=YES , SERVER=IBMSERVE , $
```

### **How Location Transparency Works**

This feature is called *location transparency* because, once a Master File and Access File Description have been generated, end users can access data in the file without knowing where it resides. You can manipulate databases described with SUFFIX=EDA as if they were local. The data output by an EDA/SQL server can be converted to any format supported by FOCUS.

Generally, you need to create Master and Access File Descriptions only once, and it's only necessary to regenerate them if the structure of the database on the server changes.

Keep the following in mind:

- FOCUS can join databases accessible through EDA that reside on different platforms. The communication protocols used to connect to the EDA/SQL servers may be the same or different (for example, TCP/IP to an EDA/SQL Server for VMS, and LU0 to an EDA/SQL Server for MVS).
- The EDA Interface (SUFFIX=EDA) is a relational view. Like other relational database interfaces such as Oracle or SQL Server, it uses the ALIAS value from a file description when generating a report request to ship to the relational database. Consequently, make sure that you provide values for the ALIAS attributes in your descriptions.
- Master File Descriptions should also contain values for ACTUAL format attributes.

## Logging onto the Server

You can logon to an EDA/SQL server by issuing SQL EDA SET commands in a FOCEXEC or at the FOCUS Session prompt.

### Syntax      Setting the Server Destination

To set the server destination (an alternative to placing the server name in the Access File Description), use the following command

```
SQL EDA SET SERVER servername
```

where:

*servername*

Is the EDA/SQL server name. It must match the ENTITY keyword in the EDA/SQL configuration file on the server. (These two keywords should already match.)

## Syntax      Sending a User ID and Password to the Server

To send a user id and password to the server, issue

```
SQL EDA SET USER Servername/userid,password
```

where:

*servername*

Is the server with which you want to associate this User ID and password.

*userid*

Is the User ID.

*password*

Is the password.

This command does not affect the EDA/SQL server that requests will be directed to; it simply associates a user id and a password with a particular EDA/SQL server.

## Joining Files Across Platforms

You can use the EDA Interface to join files on different platforms. The presence of SUFFIX=EDA in the file descriptions causes FOCUS to use the EDA Interface. For example, you could join an EMPLOYEE file on an MVS system to a JOBFIL file on a UNIX system. The EDA Interface works faster if you join the smaller file to the larger file. This is because the EDA Interface will make one SQL call to the first file, and then make one SQL call to the second file for each value of the referenced field.

## Issuing SQL Commands to the EDA/SQL Server

You can issue FOCUS or SQL commands to the EDA/SQL server. Issue the SQL commands at the FOCUS Session prompt or place them in FOCUS procedures. For example, from the FOCUS Session prompt, you could issue the following commands (notice that the prompt changes from >> to SQL>):

```
>>SQL EDA
SQL>SELECT * FROM CAR;
SQL>END
```

## Executing Stored Procedures

In addition to making data available to the client, an EDA/SQL server can store procedures, which are called *remote procedures* or *stored procedures*.

### Syntax      How to Execute a Stored Procedure Using the EDA Interface

You can execute stored procedures from FOCUS, by issuing the command:

```
SQL EDA EX rpcname parm1, parm2, ...
END
```



where:

*rpcname*

Is a procedure on the server.

*parm1, parm2, ...*

Are character strings sent to the server (they are the same as parameters you can pass on the execution line of a FOCEXEC).

## Syntax      Reviewing EDA Interface Settings

To view EDA Interface parameter settings, issue the command:

SQL EDA ?

## Using SQL

The EDA Interface also provides an SQL Passthru mode, through which SQL requests can be sent directly to the EDA/SQL server and passed to a relational DBMS with no translation by FOCUS. To use SQL Passthru:

1. Invoke SQL Passthru mode on the EDA/SQL server through the use of a server profile or stored procedure.

For example, if you want to use SQL Passthru with DB2, you would execute the command:

```
EDASET ENGINE,DB2
```

For more information, see your EDA server manual.

2. Execute an SQL Passthru command either from the FOCUS Session prompt or in a FOCUS application using the format

```
SQL EDA sqlstatement
```

where:

*sqlstatement*

Is a valid SQL statement.

You do not need a Master File Description or Access File Description on the client when using SQL Passthru.

## NF497: Project 2000—Cross-Century Dates in FOCUS Applications

With the year 2000 fast approaching, application managers must cope with the fact that 19 is assumed as the first two digits of the year for most of their applications. Applications passed 00 as a transaction year will typically interpret it as 1900. As the turn of the century approaches, date-sensitive calculations for existing applications will be thrown off unless an apparatus is provided for determining the century in question. This is expected to impact almost every type of application, including those that process mortgages, insurance policies, anniversaries, bonds, inventory replenishment, contracts, leases, pensions, receivables and customer records.

FOCUS Release 7.0.5 delivers two new SET parameters that provide a means of interpreting the century if the first two digits of the year (YYYY) are not provided elsewhere. If the first two digits are provided, they are simply accepted and validated. Using the new parameters, you can have FOCUS provide the first two digits in the following cases:

- Conversion from old dates to smart dates when the first two digits are not provided.
- Conversion from ACTUAL to USAGE when the first two digits are not provided
- Data entry when the first two digits are not provided

Automatic validation will only be done for smart dates, ensuring that the date provided is a valid date.

## Using DEFCENT and YRTHRESH to Establish a Century Window

Century specification is implemented using the SET parameters, DEFCENT and YRTHRESH.

The combination of DEFCENT and YRTHRESH establishes a base year for a 100-year window. Any 2-digit year is assumed to fall within that window, and the first two digits are set accordingly. Years outside the declared window must be handled by user coding.

The default values for the 2 parameters are DEFCENT= 19, YRTHRESH= 00. When you provide a year threshold, years greater than or equal to that value assume the value assigned by DEFCENT. Years lower than that threshold become DEFCENT plus 1.

To see how DEFCENT and YRTHRESH are applied to interpret two-digit years, consider the following:

DEFCENT=19, YRTHRESH=80

These values describe a range from 1980 to 2079. If a two-digit year field contains 99 then the year is 1999. If it's 79 then the year is 2079. If it's 00 then the year is 2000:

0	< YRTHRESH=80 ≥	99
↑		↑
Century=DEFCENT+1 (20)		Century=DEFCENT (19)

### Syntax      How to Set the Century Window for Two-digit Year Values

The syntax of DEFCENT is

```
SET DEFCENT = {nn,19}
```

where:

*nn*

Sets lower value of the century window. The default is 19.

## **Syntax**      **How to Set the Threshold Year for Two-digit Year Values**

The syntax of YRTHRESH is:

```
SET YRTHRESH = {nn,0}
```

where:

*nn*

Is the lowest year value assumed for the century established by DEFCENT. Any lower value is assumed to apply to the century DEFCENT + 1. Zero is the default.

## **Support for Date Variables in Dialogue Manager**

Release 7.0.5 enables a 4-digit year display with dialogue manager. The variable name is &DATE. It can include all combinations of YMD formats.

## **Syntax**      **How to Display Four-digit Year Values in Dialogue Manager**

The syntax of &DATE is:

```
&DATEfmt =
```

where:

*fmt*

Is any combination of options described in the table in *Alternate Date Formats* of the chapter *Describing Non-FOCUS Data Structures* in the *Data Description and Reporting Users Manual Release 7.0 Volume 1*.

### **Example**    **Displaying Four-digit Year Values in Dialogue Manager**

For example:

```
TABLE FILE MYFILE
HEADING
"TODAY'S 4 DIGIT DATE IS: &DATEYYMD"
PRINT .. etc.
END
```

Delimiters are no longer allowed in Dialogue Manager. These delimiters are shown in the *Alternate Date Formats* section mentioned above.

## NF499: Scrolling Report Headings in HotScreen

You can make report headings and footers scroll along with the report contents in your HotScreen report by using the new SET command BYSCROLL. Previously, when viewing a HotScreen report, the headings and footers did not scroll along with the data. This could lead to confusion in matching the data with a corresponding header or footer.

### Syntax      How to Scroll Report Headings in HotScreen

The syntax is:

```
SET BYSCROLL = {ON|OFF}
```

where:

ON

Enables BYSCROLL.

OFF

Disables BYSCROLL. OFF is the default.

In order to use BYSCROLL, the text in the report must be longer than 80 characters and BYPANEL must be set ON. With BYPANEL OFF, headings and footings will not scroll. Note that fencing is not supported while BYPANEL is on. To determine the setting of BYSCROLL, enter ? SET ALL. (? SET does not display BYSCROLL).

## NF500: Keyed Retrievals from FOCUS Extract Files

Keyed retrieval is supported with FOCUS HOLD files. This can greatly reduce the IOs incurred in reading extract files. The performance gains are accomplished by using the SEGTYPE= parameter in the MFD as a logical key for sequential files. With FIXRETRIEVE=ON, FOCUS stops the retrieval process when an equality test on this field holds true. This changes former behavior, as the interface previously read all of the records from the QSAM file and then passed them to FOCUS to apply the screening conditions when creating the final report.

### **Syntax**      How to Control Keyed Retrieval for an Extract File

The syntax is

```
SET FIXRETRIEVE = {OFF|ON}
```

where ON is the default.

You can abbreviate this as:

```
SET FIXRET = {OFF|ON}
```



## Uses for Keyed Retrieval From an Extract File

It is common for FOCUS users to read one of the many databases we support and create extract files using ON TABLE HOLD. They then join these FIX files to other databases to narrow their view of the data. The concept of a logical key in a FIX file permits qualified keyed searches for all records that match IF/ WHERE tests for the first n KEY fields coded in SEGTYPE=. Retrieval stops when the screening test detects values greater than those specified in the IF/ WHERE test.

### Example Master File for Keyed Retrieval From an Extract File

The following Master File describes a FIX file with sorted values of MKEY in ascending order from 1 to 100,000:

```
FILE=SORTED, SUFFIX=FIX, $
SEGNAME=ONE, SEGTYPE=S1, $
  FIELD=MYKEY, MK, I8, I8, $
  FIELD=MFIELD, MF, A10, A10, $
TABLE FILE SORTED
  PRINT MFIELD
  WHERE MYKEY EQ 100
END
```

In this instance, with FIXRETRIEVE=ON, retrieval stops when MYKEY reaches 101, vastly reducing the potential number of IOs, as we have read only 101 records out of a possible 100K records.

When a Master File for the extract file is created, a SEGTYPE of either Sn or SHn is inserted based on the BY fields in the request. For example, PRINT field BY field creates a HOLDMAST with SEGTYPE=S1. Using BY HIGHEST field creates a Master with SEGTYPE=SH1.

## Example Selection Criteria for Keyed Retrieval From an Extract File

Selection criteria may include lists of equality values. For example,

```
{IF|WHERE} MYKEY EQ x OR y OR z
```

IF/WHERE tests may also include range tests. For example,

```
{IF|WHERE} MYKEY IS-FROM x TO y
```

The maximum number of BY fields remains 32.

Keep in mind, in using this feature, that when adding unsorted records to a sorted HOLD file, the duplicate records will not be retrieved as in earlier FOCUS Releases. For example, if a sorted file contains the following 3 records:

Key

1 1200

2 2340

3 4875

and you add the following record at the bottom of the file:

1 1620

the new (duplicate) record with a key value equal to 1 would be displayed with earlier Releases of FOCUS. With Release 7.0.5 and FIXRETRIEVE=ON, the new record with a key value of 1 would be omitted, as retrieval would stop as soon as a key value of 2 was encountered.

## NF501: Public and Private DDname for MSO

MSO may be used in different ways. Some users simply run production applications and the global allocations for the MSO region as a whole are adequate. Other users need to allocate their own MASTER or FOCEXEC PDSs. These users can now use PRIVATEDD to perform allocations of PDSs for their exclusive use.

PRIVATEDD allows users to put their allocations in front of, in back of, or override the allocations for the MSO region. Applications can now be tested, or new ones put into production without affecting the region as a whole.



This feature extends the flexibility of MSO allocation. Users are able to allocate a copy of a DDname that has already been allocated in the MSO start up JCL. Allocations that are accomplished using the MSO JCL are referred to as Global allocations and those allocations that are done by MSO FOCUS users with the DYNAM command are referred to as Private allocations.

### Syntax How to Activate PRIVATEDD

This feature is activated by specifying, in the configuration file;

```
PRIVATEDD=[OFF|FIRST|LAST|REPLACE]:
```

where:

*setting*

Is the PRIVATEDD setting. Possible values are:

- |                    |  |
|--------------------|--|
| <code>FIRST</code> | When “FIRST” is specified, the Private allocations are logically “first” in the concatenation. MSO logic searches the user's allocation before it searches the Global allocation. When the file is sequential the Private DDname allocation replaces the Global.   |
| <code>LAST</code>  | When “LAST” is specified, the Private allocation is logically after the Global allocation. MSO logic searches the Global allocation before it searches the Private allocation. When the file is sequential, the GlobalDDname allocation masks the Private one . The Private allocation is not processed. |

<code>REPLACE</code>	When “REPLACE” is specified, and a Private DDname is allocated it completely replaces the Global DDname.
<code>OFF</code>	OFF When “OFF” is specified the MSO allocation and search characteristics are identical to prior releases. This is the default setting if PRIVATEDD is not specified in the configuration file.

Prior to this feature, if a DDname was globally allocated, it was the only occurrence that was allowed. If an MSO user attempted to allocate the same named DDname via the DYNAM command, the allocation would fail with either a FOC881 or a FOC398 error.

## **How PRIVATEDD Works**

This new MSO facility allows the installation to give FOCUS users the ability to allocate their own copy of the same file name. The installation, through the PRIVATEDD keyword in the configuration file, controls whether these Private allocations are logically before the global allocations, after them, or completely replace them.

The implementation of Private DDname is transparent to any application running in MSO. When PRIVATEDD is enabled, DYNAM processing ignores the existence of the Global DDname. The DYNAM error messages are only based upon whether there is a Private allocation of the DDname. Therefore, a FOCUS user can have a Private allocation of FOCEXEC. However, after the FOCEXEC DDname is allocated, a subsequent allocation of FOCEXEC that does not specify REUSE causes the following message to display:

`(FOC881) FOLLOWING DDNAME IS NOT AVAILABLE: FOCEXEC, #099:04-0410`

FOCUS functions that dynamically search concatenated partitioned data sets (PDSs) to find a particular member may search Private then Global; Global then Private, or Private only depending upon the particular parameter that was supplied with the PRIVATEDD keyword in the MSO configuration file. FOCUS functions that process sequential files; files that are not PDSs or that reference just one member of a PDS will access either the Private OR the Global allocation.

The ? TSO DDNAME, ? TSO DDNAME DDname, and ? TSO DDNAME ddna\* commands have been updated to show the effective allocation of global and private DDnames based upon the current setting of PRIVATEDD. See Section ? TSO DDNAME, ? TSO DDNAME DDname\*

When TED searches for a particular member based upon the DDname (not a specific data set name) it searches based upon the current setting for PRIVATEDD. However, TED **always** saves the member into the Private allocation.

When PRIVATEDD=LAST is set this might mean that TED initially fetches the member, if it exists, from one of the PDSs concatenated to the Global allocation. When the member is saved it is saved into the first concatenated data set of the Private allocation. Then when the FOCEXEC is executed, the original, unmodified version is executed. This is apparent to the user who edited the FOCEXEC because the modification did not cause the expected result. In these cases the ? TSO DDNAME command should be executed to display the current allocations. The specific data set allocated to the Global file name is apparent. Then the TED command could be issued specifying the data set name. In that case, the initial fetch and final file will both be against the Global allocation.

## **Console File Allocation Displays**

The MSO console continues to display both the Global and Private allocations but the logical relationship is not apparent. The Global allocations are displayed by selecting the REGION entry on the display user panel with a W. The Private allocations are displayed by selecting the individual user id on the display user's panel with a W.

**Figure 1: Console Display of Region's Allocations**

MSO1 JOB03161--- .CONSOLE DISPLAY USERS PANEL. ---Line:001(002) Col:001  
 COMMAND ===>

```

+-----DATA SET ALLOCATIONS-----+
| //STEPLIB      DD DSN=MYUSR1.MYBAT.LOAD,DISP=SHR
| //TASKLIB      DD DSN=MYUSR2.NULL.LOAD,DISP=SHR
| //            DD DSN=PROD.FOCLIB.LOAD,DISP=SHR
| //            DD DSN=PROD.FUSELIB.LOAD,DISP=SHR
| //ERRORS      DD DSN=PROD.ERRORS.DATA,DISP=SHR
| //FOCEXEC      DD DSN=PROD.FOCEXEC.DATA,DISP=SHR
| //MASTER      DD DSN=PROD.MASTER.DATA,DISP=SHR
| //MSOPRINT     DD SYSOUT=*
| //MSGET       DD DSN=MYUSR2.MSGET.DATA,DISP=SHR
| //MSPUT       DD DSN=MYUSR2.MSPUT.DATA,DISP=SHR
| //MSOPROF     DD DSN=MYUSR2.PROFILE.DATA,DISP=SHR
| //FOCCONS     DD DSN=MYUSR2.CNTL(FOCCONS2),DISP=SHR
| //FOCMSO      DD DSN=MYUSR2.CNTL(FOCMSO2),DISP=SHR
|
|
+-----+
  
```

Global allocation of file, FOCEXEC



## Figure 2: Console Display of Region's Allocations

MSO1 JOB03161--- .CONSOLE DISPLAY USERS PANEL. ---Line:001(002) Col:001

COMMAND ===>

```

+-----DATA SET ALLOCATIONS-----+
| //STEPLIB      DD DSN=MYUSR1.MYBAT.LOAD,DISP=SHR
| //TASKLIB      DD DSN=MYUSR2.NULL.LOAD,DISP=SHR
| //             DD DSN=PROD.FOCLIB.LOAD,DISP=SHR
| //             DD DSN=PROD.FUSELIB.LOAD,DISP=SHR
| //ERRORS       DD DSN=PROD.ERRORS.DATA,DISP=SHR
| //FOCEXEC      DD DSN=PROD.FOCEXEC.DATA,DISP=SHR
| //MASTER       DD DSN=PROD.MASTER.DATA,DISP=SHR
| //MSOPRINT     DD SYSOUT=*
| //MSGET        DD DSN=MYUSR2.MSGET.DATA,DISP=SHR
| //MSPUT        DD DSN=MYUSR2.MSPUT.DATA,DISP=SHR
| //MSOPROF      DD DSN=MYUSR2.PROFILE.DATA,DISP=SHR
| //FOCCONS      DD DSN=MYUSR2.CNTL(FOCCONS2),DISP=SHR
| //FOCMSO       DD DSN=MYUSR2.CNTL(FOCMSO2),DISP=SHR
|
|
+-----+

```

Global allocation of file, FOCEXEC

MYUSR22 JOB05667----- .CONSOLE DISPLAY USERS PANEL. -----Line:001(002)  
Col:001

COMMAND ===>

```
+-----DATA SET ALLOCATIONS-----+  
| //STDOUT DD DUMMY /*TR000004 |  
| //OFFLINE DD SYSOUT=A /*OF001001 |  
| //FOCSORT DD SPACE=(TRK,(5,5)),DISP=NEW /*FO001001 |  
| //FOCEXEC DD DSN=MYUSR2.FOCEXEC.DATA,DISP=SHR /*FO002001 |  
| //MASTER DD DSN=MYUSR2.MASTER.DATA,DISP=SHR /*MA001001 |  
+-----+  
  
|  
  
|  
  
|  
  
|  
  
|  
  
+-----+  
  
|  
  
+-----+  
  
|  
  
+-----+  
  
|  
  
+-----+  
  
|  
  
+-----+
```

Private allocation of a file, FOCEXEC

## ? TSO DDNAME? DDname\*

These two commands have been updated to properly display the current logical concatenation order of the Global and Private files. Below are sample TOE screens showing the output of these commands with both Global and Private allocations and PRIVATEDD=FIRST.

```
+ Output----->+
| FOCUS 7.0.5 11/07/95 15.00.03 GEN28.01
|>? TSO DDNAME
|DDNAME OCCURRENCES DSNAME
|
|ERRORS      1 PROD.ERRORS.DATA
|FOCCONS     1 MYUSR2.MYMSO.CNTL
|FOCEXEC     1 MYUSR2.FOCEXEC.DATA
|FOCEXEC     1 PROD.FOCEXEC.DATA
|FOCMSO      1 MYUSR2.MYMSO.CNTL
|FOCSORT     1 SYS95311.T150003.RA000.MYUSR2.R0171001
|MASTER      1 MYUSR2.MASTER.DATA
|MASTER      1 PROD.MASTER.DATA
|MSGET       1 MYUSR2.GRP.MSGET.DATA
|MSOPRINT    1 MYUSR2.MYUSR22.JOB03161.D0000101.?
+-----+
+-----+ (MORE) ---+
+ History--- (MORE) -->+
+ FOCUS Command-----+
+-----+
+-----+
```

**Figure 3: PRIVATEDD=FIRST, first screen displayed of ? TSO DDNAME**

Filenames are sorted. The private allocation is displayed first because PRIVATEDD is set to FIRST

```
+ Output------(MORE)-->+
|MSOPROF 1 MYUSR2.MYMSO.PROFILE.DATA
|MSPUT 1 MYUSR2.GRP.MSPUT.DATA
|OFFLINE 1 MYUSR2.MYUSR22.JOB03161.D0000104.?
|STDOUT 1 NULLFILE
|STEPLIB 1 MYUSR1.MYBAT.LOAD
|TASKLIB 3 MYUSR2.NULL.LOAD
|          PROD.FOCLIB.LOAD
|          PROD.FUSELIB.LOAD
|>? TSO DDNAME FOCEXEC*
|DDNAME OCCURRENCES DSNAME
|
|FOCEXEC 1 MYUSR2.FOCEXEC.DATA,DISP=SHR *PRIV*
|FOCEXEC 1 PROD.FOCEXEC.DATA,DISP=SHR
+-----+
+ History-----(MORE)-->+
| >? TSO DDNAME FOCXE
+ FOCUS Command-----+
+-----+
```

**Figure 4: PRIVATEDD=FIRST, ? TSO DDNAME FOCEXEC\***

Filenames appear in logical order

Wild card form of query lists abbreviated version of output. The private allocation is flagged with \*PRIV\*.

When PRIVATEDD=LAST is set, the console and ? TSO DDNAME results are the same except that the Global allocation appears first in the output. When PRIVATEDD=REPLACE is the setting, only the Private allocation appears in the displays to the user. The MSO Console still displays all of the Global allocations.

## **PRIVATEDD Restrictions for User Written Subroutines**

The PRIVATEDD function has been added to improve usability of MSO files. It was not implemented as an aid for User Written Subroutines and provides limited benefit to user applications. Installation written MSO applications that use the MSODDX API to obtain the translated DDname for their processing continue to work.

If PRIVATEDD=FIRST has been specified, then MSODDX will return the translated DDname of the **PRIVATE** allocation.

If PRIVATEDD has been set to LAST or REPLACE then MSODDX will return the DDname of the **GLOBAL** allocation.

User Written Subroutines do not automatically get to “see” the logical concatenation of files created with PRIVATEDD. They have to be written to use MSO services and to do multiple DDname searches if they are to support PRIVATEDD for their own files.

## **Planning for the Implementation of PRIVATEDD**

Planning is required to determine if PRIVATEDD should be used in any particular MSO address space. It provides the flexibility of allowing users to override the installation provided FOCEXECs and Master File Descriptions or to develop ones for their own, unique purposes. This can lead to confusion if existing applications do not behave as expected. This could easily happen if PRIVATEDD=FIRST is active; the user saves a “new” FOCEXEC into their private FOCEXEC library; and that new FOCEXEC happens to have the same name as one in an existing application.

The setting for PRIVATEDD is active for all users in the MSO address space in which it is set. It cannot be turned off for selected users, groups of users, or service blocks. So, for a particular MSO region, the decision has to be made as to whether FIRST or LAST should be specified. With FIRST, all private modifications to Master File Description and FOCEXEC libraries override the Globally provided libraries. When using LAST, users can have their own FOCEXECs and Master Files available but that they want to be sure that the Globally provided libraries will be in effect. REPLACE guarantees that the Globally provided libraries are not referenced at all when a private allocation is made.

Although the term “user” has been stated it could also mean “application”. Once PRIVATEDD has been used in a particular installation, it could be the basis for the design of a particular application. Then when an application has been implemented that depends upon the setting of FIRST, LAST, or REPLACE, it becomes extremely important that it is executed in an MSO region with that setting active for PRIVATEDD.

The setting of PRIVATEDD has no effect at all if a private copy of a DDname is not made by the user, their profile, or an application that they execute. Although every account has a different profile for a “typical” user, the majority of users execute production applications and do not develop their own FOCEXECs or their own Master Files. For these users, PRIVATEDD can be set to any value.

In an environment that is primarily a production one, the more conservative use would be to specify LAST. In this way, an individual that developed their own private library of FOCEXECs and Master Files would be in no danger of inadvertently modifying the behavior of production applications. PRIVATEDD is not intended as a security mechanism but this is one way in which an installation can allow users to do their “own FOCUS thing” and yet not be able to tamper with the installation provided applications.

A test environment is often necessary for those individuals that are maintaining or enhancing existing applications. Up till now they would normally do their development and testing under TSO. Now, if PRIVATEDD=FIRST is set on in an MSO region they can test their modifications dynamically.

It is apparent that an existing MSO installation might wish to have multiple MSO regions active. Each region might have PRIVATEDD set to a different value depending upon the activities going on in each. Another new 7.0.5 feature is Load Balancing. This feature allows the MSO user to specify an application name when they Logon to MSO. Considering this application name, i.e., “test”, Logon is routed to an MSO region that has been specified as a destination for that application. This allows a user to readily get to the MSO region that is set up for production, private development, or test. See NF503 for details on implementing Load Balancing.



## NF509: MINIO - New FOCUS Database Access Method Available Under MVS

MINIO is a new I/O buffering technique that improves performance by reducing I/O operations when accessing FOCUS databases under MVS. With MINIO set on, no block is ever read more than once, and therefore the number of reads performed will be the same as the number of tracks present. This results in an overall reduction in elapsed times when reading and writing.

### Syntax      How to Set the Access Method

SET MINIO = {ON|OFF}

where

ON

Enables MINIO. ON is the default.

OFF

Disables MINIO.

With FOCUS databases that are not disorganized, MINIO can greatly reduce the number of database I/O operations for TABLE and MODIFY commands. I/O reductions of up to fifty percent are achievable with MINIO. The actual reduction will vary depending on database structure and average numbers of children segments per parent segment. By reducing I/O operations, elapsed times for TABLE and MODIFY commands also drop.

## **How MINIO Works**

MINIO reduces CPU time slightly while slightly raising memory utilization. MINIO requires one track I/O buffer per referenced segment type. Between 40K and 48K of above-the-line virtual memory is needed per referenced segment, depending on database device type.

When MINIO is enabled, FOCUS decides for each command whether or not to employ it, and which databases to use it with. It is possible in executing a single command referencing several databases that MINIO might be used for some but not for others. Databases accessed via indexes, or physically disordered through online updates, are not candidates for MINIO buffering. Physical disorganization, in this case, means that the sequence of selected records jumps all over the database, as opposed to progressing steadily forward. When disorganization occurs, MINIO abandons its buffering techniques and resorts to the standard I/O methodology.

When reading databases, MINIO is used with TABLE, TABLEF, GRAPH, MATCH and during the DUMP phase of the REBUILD command, provided the target database is not accessed via an index or is physically disorganized.

When writing to databases, MINIO is used with MODIFY but never with MAINTAIN, provided there is no CRTFORM or COMMIT subcommand. CRTFORMs indicate online transaction processing, which requires that completed transactions be written out to the database. COMMITs are explicit orders to do so. These events are incompatible with MINIO minimization logic and therefore rule out its use.

As with reads, using MINIO with MODIFY also requires that a database be accessed sequentially. Attempts to access an index, or update physically disorganized databases both cause MINIO to be disabled. In addition, frequent repositioning to previously accessed records, even within well-organized databases will cause MINIO to be disabled.

The ? STAT command is used to determine whether the previous database access command employed MINIO. Typing ? STAT generates a screen similar to the following:

```
STATISTICS OF LAST COMMAND

RECORDS      =          0      SEGS CHNGD    =          0
LINES        =          0      SEGS DELTD   =          0
BASEIO       =          87     NOMATCH      =          0
TRACKIO      =          16     DUPLICATES  =          0
SORTIO       =          0      FORMAT ERRORS =          0
SORT PAGES   =          0      INVALID CONDTS =          0
READS        =          1      OTHER REJECTS =          0
TRANSACTIONS =          1500    CACHE READS  =          0
ACCEPTED     =          1500    MERGES       =          0
SEGS INPUT   =          1500    SORT STRINGS =          0

INTERNAL MATRIX CREATED: YES      AUTOINDEX USED:      NO
SORT USED:           FOCUS        AUTOPATH USED:      NO
MINIO USED:           YES
```

In the preceding example MINIO USED is displayed as YES. It may also display NO or DISABLED.

- YES means that MINIO buffering has taken place reducing the number of tracks read/written to the FOCUS database.

- NO, means that MINIO buffering has not taken place.
- DISABLED means that MINIO buffering was started but terminated as no performance gains could be made. This does not mean that the command did not complete successfully. It only indicates that MINIO buffering began and ended during the database read/write.

## Reference Restrictions

- When MINIO is used with MODIFY, all CHECK subcommands are ignored. If a MODIFY command terminates abnormally, the condition of the database is unpredictable, and it should be restored from a backup copy and the update repeated. Since MINIO is designed to minimize I/O during large database loads and updates, it has no checkpoint or restart facility. If this is unacceptable, set MINIO off.
- MINIO is not used to access databases through sink machines or HLI programs.
- MINIO requires the presence of the TRACKIO feature. Meaning, TRACKIO must be set to ON which is the default setting. If TRACKIO is set to OFF, then MINIO is deactivated.
- MINIO buffering starts when the FOCUS database exceeds 64 pages in size. If this size is never reached, MINIO is never activated.
- If the file being modified UPDATES, INCLUDES or DELETES a field that is indexed, MINIO is disabled. In other words, FIELDTYPE=I or INDEX=I is coded in the Master File Description for this field.
- CRTFORM and COMMIT commands disable MINIO.
- MAINTAIN procedures will not use MINIO buffering techniques.

- MINIO is not enabled if the database is physically disorganized by transaction processing.

## NF510: Date and Time Stamp in Reports

Time and date can be set in reports. This is useful for determining exactly when your report has been run. You can do this statically or dynamically using the new SET command, DATETIME. In the past, the time shown in your report represented the time your report had been run with no other options available. DATETIME allows you to choose different time settings.

### Syntax      How to Choose Date and Time Options

The syntax is:

```
SET DATETIME = {STARTUP | CURRENT | NOW | RESET }
```

where:

STARTUP

Is the time and date when you began your FOCUS session. STARTUP is the default.

CURRENT/NOW

Changes each time it is interrogated.

RESET

Freezes the date and time of the current run for the rest of the session or until another SET DATETIME is issued.

You can display the DATETIME value using any FOCUS date variable, for example, YMD, MDY, TOD, etc. If DATETIME is not set, the behavior of the FOCUS date variables remains the same.

**Note:** CURRENT and NOW are interchangeable.

## NF513: Redefining Fields in Non-FOCUS Files

Support is provided for redefining record fields in non-FOCUS files. This allows a field to be described with an alternate layout.

Within the Master File Description, the redefined field(s) must be described in a separate unique segment (SEGTYPE=U) using the POSITION=fieldname and OCCURS=1 attributes.

### **Syntax**      **How to Redefine a Field**

The syntax for a redefinition segment is:

```
SEGNAME=xxx,SEGTYPE=U,PARENT=name of parent segment,  
OCCURS=1,POSITION=fieldname of field being redefined,$
```

A one-to-one relationship is established between the parent record and the redefined segment.

### **Example**      **Redefining a VSAM Structure**

The following example illustrates redefinition of the VSAM structure described in the COBOL file description where the COBOL FD is:

```
01 ALLFIELDS
02 FLD1  PIC X(4)      - this describes alpha/numeric data
02 FLD2  PIC X(4)      - this describes numeric data
02 RFLD1 PIC 9(5)V99 COMP-3 REDEFINES FLD2
02 FLD3  PIC X(8)      - this describes alpha/numeric data

FILE=REDEF, SUFFIX=VSAM, $
SEGNAME=ONE, SEGTYPE=S0, $
  GROUP=RKEY, ALIAS=KEY, USAGE=A4      ,ACTUAL=A4      , $
  FIELDNAME=FLD1,        USAGE=A4      ,ACTUAL=A4      , $
  FIELDNAME=FLD2,        USAGE=A4      ,ACTUAL=A4      , $
  FIELDNAME=FLD3,        USAGE=A8      ,ACTUAL=A8      , $
SEGNAME=TWO, SEGTYPE=U, POSITION=FLD2, OCCURS=1, PARENT=ONE , $
  FIELDNAME=RFLD1,      USAGE=P8.2    ,ACTUAL=Z4      , $
```

The redefined fields may have any user-defined name. ALIAS names for redefined field are not required.

Use of the unique segment with redefined fields helps avoid problems with multipath reporting.

### Reference Special Considerations for Redefining Fields

- Redefinition is a read-only feature and is used only for presenting an alternate view of the data. It is not used for changing the format of the data.
- A field that is being redefined must be equal in length to the field that it is redefining (same actual length).
- For integer and packed fields, you must know your data. Attempts to print numeric fields that contain alpha data will produce data exceptions or errors converting values.
- More than one field can be redefined in a segment.



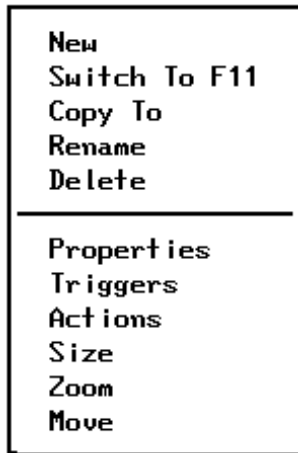
- Redefines are only supported for IDMS, IMS, VSAM, DB2 and FIX files.

## NF518: VSAM Support

MAINTAIN supports read/write functionality for VSAM databases. When using MATCH logic, do not MATCH on the GROUP but on the individual fields that make up the GROUP KEY.

## NF519: Field Level PFKeys

Enhancements have been made to the Forms menu. The Forms menu now includes two new selections, Triggers and Actions. A trigger is a specified event that invokes—"triggers"—a procedure. Each time that the event occurs, the procedure is invoked. In MAINTAIN the invoked procedure is a case or special function, and the event is something the user does in a Winform. You may also edit an object by pressing PF12 and choosing Triggers. An action refers to a system action. System actions are actions created by your system for the purpose of manipulating your Winform. For example, system actions enable you to scroll your Winform. When you select Forms the following is generated:



If you select Triggers a screen similar to the following is generated:

Triggers					
Key	System Action	Trigger	Key	System Action	Trigger
Enter					
PF1	accept		PF13		
PF2			PF14		
PF3	exit		PF15		
PF4	close		PF16		
PF5			PF17		
PF6			PF18		
PF7	backward		PF19		
PF8	forward		PF20		
PF9			PF21		
PF10	left		PF22	fieldleft	
PF11	right		PF23	fieldright	
PF12			PF24	ontop	

Ok F4      Cases F5      Cancel F3

Type the name of the case you wish to execute next to the function key you wish to use as a trigger. Whenever you press this assigned function key, the corresponding case will be executed. Position the cursor next to any PFKey and press PF5 to view the cases available to you. Press PF4 to assign the case to the function key and return to the previous screen. Press PF3 to cancel the transaction and return to the previous screen.

If you select Actions a screen similar to the following is generated:

System Actions					
Key	Description	Action	Key	Description	Action
Enter					
PF1	Accept List	accept	PF13		
PF2			PF14		
PF3	Exit Focexec	exit	PF15		
PF4	Close Winform	close	PF16		
PF5			PF17		
PF6			PF18		
PF7	Scroll Grid Up	backward	PF19		
PF8	Scroll Grid Down	forward	PF20		
PF9			PF21		
PF10	Scroll Grid Left	left	PF22	Scroll Field Left	fieldleft
PF11	Scroll Grid Right	right	PF23	Scroll Field Right	fieldright
PF12			PF24	Move Form to Top	ontop

Ok F4      Actions F5      Cancel F3

Type the name of the action you wish to execute next to the function key you wish to use as an action. The corresponding description will appear to the left of your selected action. Whenever you press this assigned function key, the corresponding action will be executed. Position the cursor next to any PFKey and press PF5 to view the system actions available to you. Press PF4 to assign the action to the function key and return to the previous screen. Press PF3 to cancel the transaction and return to the previous screen.

## Reference Notes for Using PFKeys

- If a trigger and system action are assigned to the same PFKey, the trigger will execute first.
- An object level trigger will execute before a form level trigger.

- The order of execution of triggers and actions is dependent upon cursor position. If the cursor is not on an object but within a form, the form level trigger takes place followed by the system level action. System actions will occur regardless of the position of the cursor.

## NF520: Dynamically Changing Attributes of MAINTAIN Winforms

You can dynamically change the following attributes on MAINTAIN Winforms:

- Foreground and background colors
- Hide objects or make hidden objects visible
- Protect/ unprotect objects
- Make objects blink or restore to unblinking status
- Turn on/off bold emphasis
- Underline/remove underlines from objects
- Move the cursor to an object

All Winform Set/Get commands should precede the Winform Show command or reside in a case triggered from a form. SET FOCUS can only be set from a trigger.

### **Syntax**      **How to Change and Display Foreground and Background Colors**

To change the color of the text assigned to an object, enter:

```
WINFORM SET formname.objectname.FOREGROUND_COLOR to Color value number or  
word of the color
```

To display an object's foreground color, enter:

```
WINFORM GET formname.objectname.FOREGROUND_COLOR INTO variable
```

To display an object's background color, enter:

```
WINFORM GET formname.objectname.BACKGROUND_COLOR INTO variable
```

To change an object's background color, enter:

```
WINFORM SET formname.objectname.BACKGROUND_COLOR to Color value number or  
word of the color
```

Valid foreground and background colors are:

Color	Value
Black	1
Blue	2
Green	3
Turq	4
Red	5
Pink	6
Yellow	7
White	8

Note that you get an error message if you give a variable the same name as any of the above colors. Note also that on the mainframe you can set either the foreground color or the background color - not both. This restriction does not apply to MAINTAIN Winforms on FOCUS for Windows.



## **Syntax**      **How to Hide Objects or Make Objects Visible**

To make an object visible, enter:

```
WINFORM SET formname.objectname.VISIBLE TO YES
```

To hide an object, enter:

```
WINFORM SET formname.objectname.VISIBLE TO NO
```

To determine an object's visibility, enter:

```
WINFORM GET formname.objectname.VISIBLE INTO variable
```

Visible objects return a value of 1. Hidden objects return a value of 0.

## **Syntax**      **How to Protect or Unprotect an Object**

To protect an object, enter:

```
WINFORM SET formname.objectname.PROTECTED TO YES
```

To remove protection, enter:

```
WINFORM SET formname.objectname.PROTECTED TO NO
```

To learn whether an object is protected or not, enter

```
WINFORM GET formname.objectname.PROTECTED INTO variable
```

Protected objects return a value of 1. Unprotected objects return a value of 0.

## **Syntax**      **How to Move the Cursor to an Object**

To move the cursor to an object, enter:

```
WINFORM SET formname.objectname.FOCUS TO HERE
```

## **Syntax**      **How to Make an Object's Fonts Bold**

To make a font bold, enter:

```
WINFORM SET formname.objectname.BOLD_FONT TO YES
```

To return a bold font to normal, enter:

```
WINFORM SET formname.objectname.BOLD_FONT TO NO
```

To determine how an object is displayed, enter:

```
WINFORM GET formname.objectname.BOLD_FONT INTO variable
```

If the object specification indicates a bold font, the variable will contain 1. If the font is normal, the variable will contain zero.

## **Syntax**      **How to Make Objects Blink**

To make an object blink, enter:

```
WINFORM SET formname.objectname.BLINK_FONT TO YES
```

To remove the blinking, enter:

```
WINFORM SET formname.objectname.BLINK_FONT TO NO
```

You determine whether or not an object is blinking, enter:

```
WINFORM GET formname.objectname.BLINK_FONT INTO variable
```

If the object blinks the variable will contain 1. If it does not the variable will contain zero.

## **Syntax**      **How to Underline Objects or Remove Underlines from Objects**

To underline an object, enter:

```
WINFORM SET formname.objectname.UNDERLINE_FONT TO YES
```

To turn off underlining, enter:

```
WINFORM SET formname.objectname.UNDERLINE_FONT TO NO
```

To determine whether or not an object is underlined, enter:

```
WINFORM GET formname.objectname.UNDERLINE_FONT INTO variable
```

If the object is underlined the variable will contain 1. If not, it will contain a zero.

## NF521: Enhancements to Objects in MAINTAIN

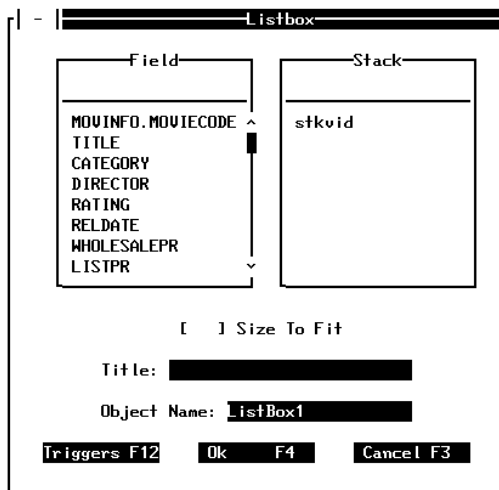
Enhancements have been introduced for objects created with the MAINTAIN Painter. List boxes, combo boxes and radio groups (formerly called radio buttons) can now include triggers to extend their functionality. Triggers are specified events that invoke (or "trigger") a procedure each time that event occurs. In MAINTAIN applications, these invoked procedures may either be cases or special functions, and they are coded in a Winform.

### List Boxes

Use List boxes to display lists of application alternatives.

#### **Procedure** How to Create a List box

When you have more choices than fit in the box, you provide scroll bars to permit the user to move through the list. Selecting Listbox from the Objects Menu prompts you for the dimensions of the list box you are creating. After choosing the upper left-hand and lower right-hand points of the box to be created, a screen similar to the following is generated:



You must select a stack in order to create a list box. Check SizeTo Fit if you wish to size your object to fit the data inside the object. The Title and Object Name are optional. ListBox1 is generated as the default name the first time you create a list box within an application, followed by ListBox2 and so forth. You can change Object Names to whatever you wish.

To assign a trigger press PF12. A screen similar to the following is generated when you select Triggers:

Triggers					
Key	System Action	Trigger	Key	System Action	Trigger
Enter					
PF1	accept		PF13		
PF2			PF14		
PF3	exit		PF15		
PF4	close		PF16		
PF5			PF17		
PF6			PF18		
PF7	backward		PF19		
PF8	forward		PF20		
PF9			PF21		
PF10	left		PF22	fieldleft	
PF11	right		PF23	fieldright	
PF12			PF24	ontop	

Ok F4      Cases F5      Cancel F3

Type the name of the case you wish to have executed next to the function key you are assigning as a trigger. Whenever this function key is pressed, the corresponding case will be executed. Position the cursor next to any PFKey and press PF5 to view the available cases. Press PF4 to assign the case you select to the function key and return to the previous screen. Press PF3 to cancel the transaction and return to the previous screen.

### Combo Boxes

When an entry field appears together with a list box, the combination is called a combo box. You can place values in entry fields by selecting them from the list box contents or by typing them directly into the entry field.

## Procedure How to Create a Combo Box

Selecting Combobox from the Objects Menu prompts you for the dimensions of the box. After you select the upper left-hand and lower right-hand points of the box you are creating, a screen similar to the following is generated:

Combobox

Field

MOVINFO, MOVIECODE  
TITLE  
CATEGORY  
DIRECTOR  
RATING  
RELEDATE  
WHOLESALEPR  
LISTPR

Stack

stkvid

Length:  [ ] Accepts Combo

Height:

Object Name:

Triggers F12    Ok F4    Cancel F3

You must also select a stack in order to create a combo box. Check “Accepts Combo” if there are Accepts defined in the Master file Description that you wish to use. If you do not check Accepts Combo, Accepts in the Master are ignored. If you check Accepts Combo but there are no Accepts in the Master, your request is ignored. Enter the dimensions of the combo box by assigning a length and height.

To assign a trigger press PF12 and follow the directions as stated in the previous section.

## **Radio Groups**

Radio groups represent mutually exclusive options. You can select only one button within a radio group at a time. Radio groups use parentheses to indicate button groups and the selected buttons display an asterisk.

### **Procedure** How to Create a Radio Group

Selecting Radio Groups on the Objects Menu prompts you for the dimensions of the Radio Group box you are creating. After selecting the upper left-hand and lower right-hand points of the box you are creating a screen similar to the following is generated:



Radio Group																	
<table border="1"> <thead> <tr> <th>Field</th> </tr> </thead> <tbody> <tr><td>MOVINFO.MOVIECODE</td></tr> <tr><td>TITLE</td></tr> <tr><td>CATEGORY</td></tr> <tr><td>DIRECTOR</td></tr> <tr><td>RATING</td></tr> <tr><td>RELDATE</td></tr> <tr><td>WHOLESALEPR</td></tr> <tr><td>LISTPR</td></tr> </tbody> </table>	Field	MOVINFO.MOVIECODE	TITLE	CATEGORY	DIRECTOR	RATING	RELDATE	WHOLESALEPR	LISTPR	<table border="1"> <thead> <tr> <th>Stack</th> </tr> </thead> <tbody> <tr><td>stkvid</td></tr> </tbody> </table>	Stack	stkvid	<table border="1"> <thead> <tr> <th>Justification</th> </tr> </thead> <tbody> <tr><td><input type="checkbox"/> Left</td></tr> <tr><td><input checked="" type="checkbox"/> Center</td></tr> <tr><td><input type="checkbox"/> Right</td></tr> </tbody> </table>	Justification	<input type="checkbox"/> Left	<input checked="" type="checkbox"/> Center	<input type="checkbox"/> Right
Field																	
MOVINFO.MOVIECODE																	
TITLE																	
CATEGORY																	
DIRECTOR																	
RATING																	
RELDATE																	
WHOLESALEPR																	
LISTPR																	
Stack																	
stkvid																	
Justification																	
<input type="checkbox"/> Left																	
<input checked="" type="checkbox"/> Center																	
<input type="checkbox"/> Right																	
Columns: <input type="text" value="1"/>																	
Column Width: <input type="text" value="13"/>																	
<input type="checkbox"/> Border																	
Title: <input type="text"/>																	
Object Name: <input type="text" value="RadioGroup1"/>																	
<input type="button" value="Triggers F12"/> <input type="button" value="Ok F4"/> <input type="button" value="Cancel F3"/>																	

You must select a stack in order to create a radio group. Select where in the radio group you wish text placed by checking Left, Center or Right. Enter the number of columns you would like displayed. The default is 1. The column width is determined by the length of the selected field, as defined in the Master File Description. If you shorten this length the displayed information will be truncated. Checking Border places a border around the radio group.

## CDN Support

MAINTAIN now supports comma decimal notation for International users.

## MAINTAIN Improved SU Performance

MAINTAIN SU now uses data compression with very large blocks of data to maximize throughput for the client applications. This feature can be exploited by using set-based MAINTAIN requests in the form of FOR ALL NEXT...WHERE...INTO.

## National Language Support

The 'SET LANG=' command is now fully supported for all languages. Japanese, Dutch, German, French, Swedish, and Spanish language files are all available. If you require additional language support, please contact the International Division for more information.

## 7.0.3 New Features

### Reporting Enhancements

**NF491: Distinct Operator**

### MAINTAIN

**Missing Values**

### ADABAS Interface

**NF495:ADABAS Interface: Multifetch/Prefetch Options**

### AUTODATACOM

**NF498: Enabling the CA-DATACOM/DB CBS Trace**

## NF491: Distinct Operator

The Distinct verb object operator (DST.) may be used to aggregate and list unique values of any database field. Similar in function to the SQL COUNT, SUM, and AVG(DISTINCT col) column functions, it permits you to determine the total number of distinct values in a single pass of the database.

The DST. operator can be used with the SUM, PRINT or COUNT verbs, and also in conjunction with the aggregate verb operators SUM., CNT., and AVE.

### Syntax      How to Use the Distinct Operator

The syntax is:

```
[SUM, PRINT or COUNT] DST.fieldname
```

or

```
SUM [operator].DST.fieldname
```

where:

DST.

The Distinct verb object operator

fieldname

Indicates the verb object or fieldname.

operator

Indicates SUM., CNT., or AVE.

### Example      Using the Distinct Operator

The FOCEXEC requesting a count of unique ED\_HRS values is either:

```
TABLE FILE EMPLOYEE
SUM CNT.DST.ED_HRS
END
```

or

```
TABLE FILE EMPLOYEE
COUNT DST.ED_HRS
END
```

The output is:

```
COUNT
DISTINCT
ED_HRS
-----
```

9

Notice that the count excludes the second records for values 50.00, 25.00, and .00 resulting in nine unique ED\_HRS values.

When used with PRINT, DST acts in the same manner as a BY statement. It can be attached to several PRINT verb objects, but not more than 32 (the current limit for sort fields in FOCUS). DST. verb objects must precede all non-distinct verb objects named by the PRINT verb. If not, you will get a (FOC1855) error (DISTINCT FIELDS MUST PRECEDE THE NONDISTINCT ONES).

## Reference Restrictions

- The prefix operators CNT., SUM., and AVE. may only be used with the SUM verb. The error (FOC1853) CNT/SUM/AVE.DST CAN ONLY BE USED WITH AGGREGATION VERBS occurs if used with any other verb.

- DST. may not be used in a MATCH or TABLEF command. A (FOC1854) THE DST OPERATOR IS ONLY SUPPORTED IN TABLE REQUESTS error will occur.
- Only one DST. verb object operator may be coded for the SUM verb. If more than one is coded a (FOC1856) ONLY ONE DISTINCT FIELD IS ALLOWED IN AGGREGATION error occurs.
- When used with the PRINT verb, the DST.fieldname becomes a BY field. Therefore reformatting of a BY field generates a (FOC1862) REFORMAT DST.FIELD IS NOT SUPPORTED WITH PRINT error.
- The DST. verb object operator cannot be used in an ACROSS or FOR statement. A (FOC1864) THE DST OPERATOR IS NOT SUPPORTED FOR ACROSS OR FOR error will be generated.
- The DST. verb object operator used with the SUM verb must be at the lowest level of aggregation. A multi-verb request, SUM DST.fieldname BY field PRINT fld BY fld generates a (FOC1867) DST OPERATOR MUST BE AT THE LOWEST LEVEL OF AGGREGATION error.
- The DST. operator may not be used as part of a HEADING or a FOOTING.
- TABLE requests that contain the DST. operator are not converted to TABLEF.



## NF495:ADABAS Interface: Multifetch/Prefetch Options

The ADABAS Multifetch and Prefetch options reduce execution time and allow ADABAS data to be retrieved with a high degree of efficiency. The ADABAS Interface included in FOCUS and EDA/SQL now supports these ADABAS features.

The Multifetch feature reduces the communication overhead between the ADABAS Interface and the ADABAS nucleus by buffering multiple record results from a single call. ADABAS Release 4.0 is the first to support Prefetch and Release 5.3.2 is the first to support Multifetch. The Interface uses the Multifetch feature, if available, otherwise it uses the Prefetch feature, if available.

### INVOKING MULTIFETCH/PREFETCH

This feature requires that OPEN=YES (which is the default value for OPEN) be specified in the FOCADBS Access File Description. You can activate or deactivate the Multifetch/Prefetch feature as long as OPEN=YES is specified.

There are two ways to activate (or deactivate) this feature. One way is to issue SET commands before running the request. The other way is through syntax in the FOCADBS Access File Description. The SET commands override the Access File Description settings.

#### **Syntax**      **How to Activate the Multifetch/Prefetch Option With SET Commands**

The syntax for the new SET commands is

```
CMS/MVS ADBSINX SET FETCH ON
```

CMS/MVS ADBSINX SET FETCH OFF

CMS/MVS ADBSINX SET FETCHSIZE n

where:

n

Is any number up to 3 digits

CMS/MVS ADBSINX SET FETCHSIZE MAXimum

where:

MAXimum

Is automatically calculated by the Interface to allow maximum records within a 32K buffer.

## **Syntax**      **How to Activate Multifetch/Prefetch in an Access File**

The access file (FOCADBS) now contains the following new keywords:

FETCH = ON/OFF

FETCHSIZE = n/MAX

where

n

Is any number up to 3 digits. The default is 10 per buffer.

MAXimum

Is automatically calculated by the Interface to allow maximum records within a 32K buffer.

FETCH and FETCHSIZE are applicable only to segments described as 'ACCESS=ADBS' in the FOCADBS Access File.

## **ADABAS Commands Supporting the Multifetch/Prefetch Feature**

The following ADABAS commands support the Multifetch/Prefetch feature:

- L2            read physical
- L3            read logical (sequential, random and join)
- L9            histogram
- L1            read with option get next after FIND (S2) command

## **Tracing the FETCH Feature**

The Interface trace file 'FSTRACE4' will contain the following new statistical information for the FETCH feature:

1. Using FETCH is allowed or disallowed,
2. Which FETCH technique is being used (Prefetch or Multifetch),
3. Number of records per FETCH buffer,
4. Fetch buffer (ISN buffer) size.

For information on using the FSTRACE4 facility, refer to Tech Memo 7906.1.

## NF498: Enabling the CA-DATACOM/DB CBS Trace

CA-DATACOM/DB's Compound Boolean Selection (CBS) Trace can be invoked through the FOCUS CA-DATACOM/DB Interface beginning with FOCUS Release 6.8 9410 and 7.0.3. The CBS Trace is invoked whenever a FOCUS or EDA/SQL request contains selection criteria and the CA-DATACOM/DB interface passes a SELFR/SELNR command combination to CA-DATACOM/DB.

The CBS Trace is invoked by adding the following syntax to the Access File Description (AFD):

### Syntax      How to Activate the CBS Trace

By including the TRACE=CBS parameter in the Access File, the first three characters of the jobname are replaced with \$\$\$ which instructs CA-DATACOM/DB to write the CBS Trace to the CA-DATACOM/DB log.

### Example      Enabling the CBS Trace in an Access File

```
TRACE=CBS , USERTABLE=CUSFURT , $  
SEGNAME=CUSTOMER , DBID=030 , TABLENAME=CUS , KEYNAME=CUSKY , $
```

## Missing Values

Missing values are supported in MAINTAIN. The MISSING attribute may be placed in the Master File in the declaration of the field missing the values, after the format. For example, this field declaration specifies the MISSING attribute for the RETURNS field:

```
FIELDNAME=RETURNS, ALIAS=RTN, FORMAT=I4, MISSING=ON, $
```

**Note:** You may add MISSING field attributes to the Master File Description at any time. However, MISSING attributes only affect the data entered into the database after the attributes were added.

You may use the MISSING ON feature in a COMPUTE command in the MAINTAIN request. This section describes the COMPUTE commands whose expressions include fields that have missing values. You can control this by using the following syntax:

```
field[/format]MISSING {ON|OFF} [NEEDS] {SOME|ALL} [DATA]=expression;
```

where:

`field`

Is the COMPUTE field.

`/format`

Is the format of the field. The default is D12.2.

`MISSING ON|OFF`

Enables FOCUS to declare the value of the computed field to be missing. The default is OFF.

### NEEDS

Is optional. It helps to clarify the meaning of the command.

### SOME

Is the default. Indicates that if at least one field in the expression has a value, then the COMPUTE field has a value (the fields missing values are evaluated as 0 or blank). If all of the fields in the expression are missing values, then the COMPUTE field is missing its value.

### ALL

Indicates that if all the fields in the expression have values, the computed field has a value. If at least one field in the expression is missing a value, then the computed field is missing its value.

### DATA

Is optional. It helps to clarify the meaning of the command.ar field Is

## 7.0.1 New Features

### Performance Enhancements

[NF425: External Indices for FOCUS Databases](#)

[NF426: AUTOPATH](#)

[NF440: Improved Page Handling - SET TRACKIO](#)

[NF441: External Sort](#)

[NF457: The IBI MVS Subsystem](#)

[NF467: Automatic Indexed Retrieval \(AUTOINDEX\)](#)

[NF470: Loading Access File Descriptions](#)

### Raised Limits

[NF423: Specifying up to 256 Verb Objects](#)

[NF427: Longer Length for HOLD FORMAT LOTUS Files](#)

[NF428: Increased Size of FOCSORT](#)

[NF434: Larger FOCUS Databases](#)

[NF435: Increased Number of Literal Values in a File](#)

[NF443: Large Packed Fields](#)

[NF449: Increased Number of Indices](#)

## Reporting Enhancements

[NF431: Universal Concatenation](#)

[NF432: Renaming/Rejustifying Row and Column Total Labels](#)

[NF446: Using StyleSheets](#)

## Dialogue Manager Enhancements

[NF421: Determining Which FOCEXEC is Running](#)

[NF444: ASIS Function](#)

[NF452: Capturing SET Parameter Values](#)

## MAINTAIN

[NF414: The MAINTAIN Facility](#)

## General Enhancements

[NF415: SET SAVEMATRIX](#)

[NF429: Generalized Listings of DDNAMEs](#)

[NF433: Improved Handling of Text Fields In TED](#)

[NF448: FOCPARM Enhancements](#)

[NF455: FOCUS File Date and Time Stamp](#)

[NF465: Changes to the Catalog Search in FOCUS](#)

[NF471: Enhanced ? SET Command](#)

[NF472: Enhancement to the TED Command in MVS](#)

[NF490: Online Release Information](#)



## The Multi-Session Option

[NF474: APF Internal Authorization](#)

[NF477: FASTPDS](#)

[NF478: MSO CONSOLE Browser](#)

[NF479: The New MSO/CICS Interface](#)

[NF480: Fast Logon Enhancements](#)

[NF481: DYNAM Utilities Menu](#)

[NF483: The MSO Resource Manager \(MRM\)](#)

[NF484: Allowing VSAM File Allocation in MSO JCL](#)

## HiperFOCUS

[NF468: HiperBudget](#)

## Relational Interfaces

[NF469: DRDA Support Enhancements to the DB2  
Interface](#)

[NF476: Usability Enhancements](#)

[NF496: Static SQL for TABLE Requests](#)

[NF504: Installation Enhancement for the SQL/DS  
Interface](#)

[NF505: Optimization Enhancements](#)

## **FOCSAM Interface**

[NF456: VSAM Data and Index Buffers](#)

[NF485: Dynamic GETPRV Exit](#)

[NF486: Dynamically Setting the Addressing Mode](#)

[NF487: Enhancement to the ZCOMP1 User Exit](#)

[NF508: DYNAM Support for BUFNI and BUFND](#)

## **ADABAS Interface**

[NF437: AUTOADBS](#)

[NF460: Changing the Default CALLTYPE](#)

[NF473: The New ADABAS Interface](#)

## **CA-DATACOM Interface**

[NF488: AUTODATACOM](#)

[NF507: FSTRACE for the CA-DATACOM/DB Interface](#)

## **Model 204 Interface**

[NF438: MODEL 204 Interface Enhancements](#)

## **IMS Interface**

[NF466: Controlling IMS Access via DBCTL](#)

[NF482: IMS Enhancements via SET IMS=NEW](#)

## National Language Support

[NF420: Double-Byte Character Set K Format and G Prefix](#)

[NF436: Checking Current Language Settings](#)

## NF414: The MAINTAIN Facility

### Introduction to MAINTAIN

MODIFY developers will recognize some of the basic commands, and will be pleased to note the extensions and changes that put increased power at their fingertips.

To meet the needs of developers who wish to continue maintaining some of their existing MODIFY applications, the MODIFY facility will remain available at its current release level.

### Using MAINTAIN to Manage Data

MAINTAIN combines power and simplicity in a single data management facility. It incorporates the following features:

- Set-based processing.  
You can manipulate a group of database records at the same time. You can define the group as a sequential range of records, as all records which satisfy selection criteria, or a combination of the two. For example, you can select and retrieve all of the records for the first 100 employees who have the job code A25—using just a single MAINTAIN command!
- Record-at-a-time processing.  
In addition to set-based processing, you can also identify and work with one record at a time, as in traditional MODIFY requests.
- Sophisticated Graphical User Interfaces (GUIs).

You can use the Winform Painter to create sophisticated interactive windows for entering data, displaying information, and selecting options. You can design these Winforms to include user-friendly features such as dialog boxes for requesting special information, check boxes for making choices, buttons for invoking actions, and entry fields with automatic data validation for entering valid values.

- Triggers and event-driven processing.

The flow of control in conventional processing is mostly pre-determined—that is, the programmer determines the few paths which the user will be able to take through the procedure.

To make your application more responsive to the user—via the graphical user interface—MAINTAIN introduces triggers and event-driven processing. A trigger is a specified event that invokes—"triggers"—a procedure. Each time that the event occurs, the procedure is invoked. In MAINTAIN, the invoked procedure is a case or special function, and the event is something the user does in a Winform. For example, you might create a trigger for retrieving data: it would notice whenever a user clicks a certain button on the Winform, and react by retrieving the specified data from the database and displaying it in the Winform.

- Event-driven development.

Developing a request by writing out sequential lines of code may be sufficient for conventional linear processing, but event-driven processing demands event-driven development. Developing an application in this way enables you to build much of the application's logic around the user interface. For example, you could start by developing part of the user interface (a Winform), then assign a trigger to a particular Winform event, specify the action (that is, the case) associated with the trigger, and finally code the case—all from within the Winform Painter!

- Improved flow-of-control.

MAINTAIN provides many different ways of controlling the flow of a procedure, using enhanced versions of commands found in MODIFY as well as entirely new commands and functions. For example, you can transfer control via PERFORM, GOTO, and CALL; perform conditional actions using IF; specify blocks of code with CASE and BEGIN; and loop via REPEAT.

- Transaction Integrity.

MAINTAIN enables you to define a transaction in ways that are meaningful to your application: one transaction can include multiple INCLUDE, UPDATE, and DELETE operations. MAINTAIN respects your DBMS's transaction integrity strategy, and lets it ensure that the entire transaction is written to the database only if all of its component operations were successful. (This is not supported for concurrent database access in Release 7.0.)

- Modular processing.  
You can create several MAINTAIN requests which work together, one request calling another.
- Client-server processing.  
A MAINTAIN request can call other requests residing on different nodes of a network. In this way, you can locate different parts of your application on different platforms to leverage the strengths of each platform. This also enables cooperative processing, so that one MAINTAIN transaction can process related data on many different nodes.

These are just some of the features you can use to develop powerful, flexible, and robust data management requests.

The MAINTAIN facility is fully documented in a separate volume with a complete language reference and examples of syntax (DN1000990.1294).

## NF415: SET SAVEMATRIX

You can now preserve the internal matrix and keep it available for subsequent RETYPE, HOLD, SAVE, SAVB, and REPLOT commands. In Release 6.8, this feature was added for HiperFOCUS only, and was activated automatically. In Release 7.0, this feature has also been made available outside of HiperFOCUS and it has been enhanced to operate via a SET command. The syntax is

### Syntax      How to Preserve the Internal Matrix

```
SET SAVEMATRIX = {OFF|ON}
```

where:

OFF

Does not guarantee that the internal matrix will be available. This is the default setting.

ON

Turns on the SAVEMATRIX facility and causes FOCUS to save the last internal matrix generated.

Before SAVEMATRIX was added, if you generated a Hot Screen report and then issued a FOCEXEC containing a Dialogue Manager statement it was possible to overwrite the matrix. Now, you can retype your Hot Screen report later in the session. For example, if you execute the TABLE request:

```
TABLE FILE CAR  
PRINT MODEL  
BY COUNTRY  
END  
-RUN
```



and then execute the following program:

```
-IF &RECORDS NE 0 GOTO MESSAGE;  
-MESSAGE  
-TYPE ...  
.  
.  
.
```

the Hot Screen report will still be available for RETYPE.

**Note:** If AUTOTABLEF is in effect there may not be a matrix to retype.

## NF420: Double-Byte Character Set K Format and G Prefix

National Language Support (NLS) enables software products to work in many countries and languages. In order to support languages such as Kanji, which requires two bytes of storage to uniquely represent each Kanji character, it is necessary to support Double Byte Character Sets (DBCS). FOCUS Release 7.0 provides this support through the introduction of K format and G prefix for representing pure DBCS strings or DB2 graphic data types.

On IBM mainframes, DBCS processing is enabled via the use of special delimiters. HEX value 0E, or the shift-out (S/O) character, indicates the beginning of a DBCS string to FOCUS, while HEX value 0F, or the shift-in (S/I) character, indicates the end of a DBCS string. Mixed DBCS strings may contain combinations of DBCS and Single Byte Character Set (SBCS) characters, which are separated by S/O and S/I delimiters. Pure DBCS strings that contain only double-byte characters are stored without the S/O and S/I delimiters (e.g. DB2 GRAPHIC data type).

### The K Format

The ACTUAL keyword is used to specify pure DBCS strings, where ACTUAL=Knn defines the number of Kanji characters.

The USAGE keyword indicates the display format and includes shift character lengths. ACTUAL=Knn maps to USAGE=Amm, where "mm" is the number of bytes. USAGE=Amm is calculated with the following formula:

$$mm = nn * 2 + SHIFT\_LENGTH * 2$$

On an IBM mainframe,  $mm = nn * 2 + 2$ , as illustrated in the following example:

```
USAGE = A22, ACTUAL = K10, $
```

Do not specify K format for USAGE (USAGE = Knn), or the following error message will be generated:

```
(FOC851) ERROR IN DEFINING FIELDNAME:Format K is not supported with  
USAGE.
```

K format is valid in the following FOCUS environments:

- TABLE against external files
- MODIFY FIXFORM, PROMPT and CRTFORM

## The G Prefix

The G prefix was introduced for representing GRAPHIC type literal strings, and can be applied to any literal string surrounded by single quotation marks. This includes GRAPHIC data, including VARGRAPHIC and LONG VARGRAPHIC, that does not include a shift-out and shift-in. In SQL statements, DBCS must be specified with shift-out and shift-in. When the G prefix is applied to literals, SQL assumes that the literal strings do not contain shift codes. The G prefix should be used to test for K format value.

### Example Using the K Format and G Prefix

Master File Description:

```
FILE = DBCS1, SUFFIX = FIX, $  
SEGNAME = SEG1, SEGTYPE = S1, $  
FIELD=FLD1, USAGE = A12, ACTUAL = A12, $  
FIELD=FLD2, USAGE = A12, ACTUAL = K5, $
```

### Field Values:

FLD1 <D1D2DsDsDs>

FLD2 D1D2DsDsDs

where:

<,>                shift-out and shift-in

D1,D2             DBCS

Ds                 DBCS space

### Example 1:

Trailing spaces can be omitted for K format.

IF FLD1 EQ <D1D2DsDsDs>    retrieves the record.

IF FLD1 EQ <D1D2>        does not retrieve the record.

IF FLD2 EQ <D1D2DsDsDs>    retrieves the record.

IF FLD2 EQ <D1D2>        retrieves the record.

### Example 2:

WHERE FLD2 = G'<D1D2>'    retrieves the record.

WHERE FLD2 = '<D1D2>'    retrieves the record.

In both cases, the SQL interface generates the following WHERE clause:

WHERE FLD2 = G'<D1D2>' .

However, in the following cases, the SQL Interface does not generate the G prefix.

WHERE FLD2 = G'<D1D2>' OR FLD5 = 'XXXX'    retrieves the record.

WHERE FLD2 = '<D1D2>' OR FLD5 = 'XXXX'    does not retrieve it.

## NF421: Determining Which FOCEXEC is Running

Two new FOCUS variables enable you to easily determine which procedure is running. These are particularly useful for managing reporting operations involving many similarly named requests that are either included using -INCLUDE or executed using EX. These amper variables, &FOCFOCEXEC and &FOCINCLUDE, can be specified within a request or in a Dialogue Manager statement to display the name of the currently running FOCEXEC, and the current INCLUDEd FOCEXEC, respectively.

The variables can be used in a TABLE request as shown below:

```
TABLE FILE EMPLOYEE
"REPORT: &FOCFOCEXEC -- EMPLOYEE SALARIES"
PRINT CURR_SAL BY EMP_ID
END
```

If the above request is stored as SALPRINT FOCEXEC, it would produce the following heading when executed:

```
REPORT: SALPRINT      -- EMPLOYEE SALARIES
EMP_ID                CURR_SAL
-----
071382660             $11,000.00
112847612             $13,200.00
117593129             $18,480.00
.
.
.
```

The variables can also be used in a -TYPE statement. Suppose you have a FOCEXEC named EMPNAME, which contains the following:

```
-TYPE &|FOCFOCEXEC IS:  &FOCFOCEXEC
```

When EMPNAME is executed, the following output is produced:

```
&FOCFECEXEC IS: EMPNAME
```

## NF423: Specifying up to 256 Verb Objects

Up to 256 verb objects may now be used in a FOCUS TABLE request. Earlier releases supported only 95 verb objects. The 256 verb object limit includes all named fields, whether displayed or not, (e.g., COMPUTE fields) and some internal fields such as TABPAGENO.

## NF425: External Indices for FOCUS Databases

FOCUS Release 7.0 introduces the External Index feature. FOCUS users with READ access to a local FOCUS database may now create an index database that facilitates indexed retrieval when joining or locating records. External indices offer equivalent performance to permanent indices for retrieval and analysis operations. Unlike permanent indices, they allow indexing on concatenated FOCUS databases, indexing on real and DEFINEd fields, and indexing selected records from WHERE/IF tests. External indices are created as temporary datasets unless pre-allocated to a permanent dataset. They are not updated as the indexed data changes.

With timestamping of the FOCUS database in Release 7.0, it becomes very important to ensure that the timestamp on the index database is not out of date. The index database becomes outdated when any of the databases that comprise the index are changed via MODIFY, MAINTAIN, SCAN or FSCAN. If records which comprise the index are changed, you must recreate the index database. You can accomplish this by using the EXTERNAL INDEX option of the REBUILD utility. MODIFY and MAINTAIN commands may not be used to update, include or delete the contents of this database. If the changes do not affect records that were used to create the external index database, you can use the new TIMESTAMP option of the REBUILD utility to make the timestamps match. See the document titled *FOCUS File Date and Time Stamp* in the chapter *General Enhancements* for additional information. If the timestamp on the index database is out of date, the following error message will be generated:

```
(FOC976) EXTERNAL INDEX EXPIRED
```



## Creating an External Index

The creation of an external index is accomplished via the enhanced REBUILD command. Internally, REBUILD begins a process which reads the databases that comprise the index, gathers the index information, and creates an index database containing all field, format, segment, and location information. If your index database has already been created, the add feature may be used to append additional index records to the index database without completely recreating the index database. This is described in the section below, *Concatenating Index Databases*.

The REBUILD process requires you to provide information about:

- Whether or not you want to add new records from a concatenated database to the index database.
- The name of the external index database that you want to build.
- The name of the database from which the index information is obtained.
- The name of the field from which the index is to be created.
- Whether or not you want to position the index field within a particular segment.
- Any valid WHERE or IF record selection tests.

## Concatenating Index Databases

The external index feature allows indexed retrieval from concatenated FOCUS databases, whereas prior releases of FOCUS did not allow one index to span multiple concatenated files. If you wish to concatenate databases that comprise the index, you must issue the appropriate USE command prior to the REBUILD. The USE must include all cross-referenced and LOCATION files. REBUILD EXTERNAL INDEX contains an add function that allows you to append only new index records from a concatenated database to the index database, eliminating the need to recreate the index database.

The original database from which the index was built may not be in the USE list when adding additional index records. If it is, REBUILD EXTERNAL INDEX will generate the following error message:

```
(FOC999) WARNING. EXTERNAL INDEX COMPONENT REUSED: ddname
```

## Positioning Indexed Fields

This feature enables you to create relationships between source and target segments, where the source segment is the segment which contains the indexed field, and the target segment is any segment above or below the source segment within its path. You can position retrieval of indexed values within a particular segment in order to enhance retrieval performance by entering the hierarchy at a segment other than the root.

Positioning requires that the indexed field be a DEFINEd field, not a real field. Since the DEFINE can contain any valid FOCUS expression, you can use a real field for positioning by assigning it to a temporary fieldname. The temporary fieldname is then used for REBUILD EXTERNAL INDEX.

If the target segment is not in the same path as the source, the following error message will be generated:

**(FOC974) EXTERNAL INDEX ERROR. INVALID TARGET SEGMENT**

While a source segment can be a cross-referenced or location segment, target segments cannot be cross-referenced segments. If an attempt is made to place the target on a cross-referenced segment the following error message is generated:

**(FOC1000) INVALID USE OF CROSS REFERENCE FIELD**

If you do not choose to associate your index with a particular field, the source and target segments will be the same.

## Activating an External Index

After building the external index database, you must associate it with the databases from which it was created. This is accomplished with the USE command. The syntax is the same as when USE is issued prior to building the external index database, except the WITH option is now available. The syntax is

```
USE [ADD      ]
    [REPLACE]
database_name [AS mastername]
index_database_name {WITH|INDEX} mastername
.
.
.
END
```

where:

**ADD**

Appends one or more new databases to the present USE list. Without the ADD option, the existing USE list is cleared and then replaced by the current list of USE databases.

**REPLACE**

Replaces an existing database\_name in the USE list.

**database\_name**

Is the name of the database. This is the ddname allocated for the database in MVS, and the filename, filetype and filemode in CMS.

**AS**

Is used with a Master File Description name to concatenate databases.

**mastername**

Specifies the FOCUS Master File Description name.

**index\_database\_name**

Is the name of the external index database. In MVS, this is the ddname allocated to the index database. In CMS, this is the fileid (filename, filetype, filemode) of the index database.

**WITH | INDEX**

Are new keywords that create the relationship between the component databases and the index database. They can be used interchangeably.

## **Example** Creating an External Index From a Concatenated Database

Assume you have the following USE in effect:

```
USE CLEAR *
USE
EMPLOYEE
EMP2 AS EMPLOYEE
JOBFILE
EDUCFILE
END
```

Note that EMPLOYEE and EMP2 are concatenated and can be described by the EMPLOYEE Master File Description. To initiate REBUILD, enter the following at the FOCUS prompt:

```
REBUILD
```

You are prompted to select an option:

```
ENTER OPTION (REBUILD,REORG,INDEX,EXTERNAL INDEX,CHECK OR TIMESTAMP)
```

Select the external index option by entering:

```
EXTERNAL INDEX
```

REBUILD responds with:

```
NEW, OR ADD TO EXISTING INDEX DATABASE? (NEW/ADD)
```

For this example, assume you are creating a new index database, and respond NEW. REBUILD then prompts:

```
ENTER FILENAME OF EXTERNAL INDEX
```

Enter the name of the external index database, for example:

```
EMPIDX
```

You are prompted for the name of the database from which the index information is obtained:

```
ENTER NAME OF FOCUS FILE TO INDEX
```

Enter the name of the file from which the index records are obtained:

EMPLOYEE

You are then prompted for the name of the field to index:

ENTER NAME OF FIELD TO INDEX

Enter the name of the field:

CURR\_JOBCODE

You are prompted for a field to associate the index with. This allows for positioning of the indexed field outside the source segment:

ASSOCIATE INDEX WITH A PARTICULAR FIELD? (YES/NO)

Reply NO for this example.

Lastly, you are prompted for any record selection tests. Again, reply NO to the prompt:

ANY RECORD SELECTION TESTS? (YES/NO)

At this point the REBUILD process will continue and display the following when it has completed:

EXTERNAL INDEX FILE: EMPIDX  
DATE/TIME OF LAST CHANGE: 10/12/94 10.59.55

EXTERNAL INDEX DATABASE PAGES: 00000001  
DATABASE INDEXED: EMPLOYEE  
FIELD NAME: CURR\_JOBCODE  
FIELD FORMAT: A3  
SEGMENT NAME: EMPINFO  
SEGMENT LOCATION: EMPLOYEE

EXTERNAL INDEX DATA COMPONENTS:

EMPLOYEE  
EMP2

This output is from the ? FDT command, which is automatically performed at the end of the REBUILD EXTERNAL INDEX process. This is done in order to validate the contents of the index database. Note that the ? FDT command has been updated in Release 7.0 to include a new display for the index database.

Record selection may also be used by responding YES to the prompt:

```
ANY RECORD SELECTION TESTS? (YES/NO)
```

You are then prompted:

```
ENTER IF OR WHERE TESTS (TERMINATE WITH 'END' ON A SEPARATE LINE)
```

Your record selection tests may span more than one line and must be terminated by the END command. For example, if you want to select particular instances of CURR\_JOBCODE, you could respond:

```
IF CURR_JOBCODE EQ 'B02' OR 'B03' OR 'B04'  
OR 'B14'  
END
```

## **Example**    **Activating the External Index Database**

You must activate the external index database prior to referencing it in TABLE. This is done by placing it in the USE list as follows:

```
USE CLEAR
USE
EMPLOYEE
EMP2 AS EMPLOYEE
JOBFILE
EDUCFILE
EMPIDX WITH EMPLOYEE
END
TABLE FILE EMPLOYEE.CURR_JOBCODE
PRINT FIRST_NAME LAST_NAME CURR_JOBCODE
WHERE CURR_JOBCODE EQ 'A07'
END
```

The TABLE request illustrates the use of an external index database, EMPIDX, for the indexed field CURR\_JOBCODE. This field has two records in EMPLOYEE and two records in EMP2 with the value 'A07'. Therefore, indexed retrieval on EMPLOYEE retrieves four records due to concatenation with EMP2.

The output from TABLE is:

FIRST_NAME	LAST_NAME	CURR_JOBCODE
-----	-----	-----
MARY	GREENSPAN	A07
ALFRED	STEVENS	A07
ALBERT	EINSTEIN	A07
ANN	RICE	A07

The first two records are from EMPLOYEE; the last two are from EMP2.



**Example** Adding a New Database to an Existing Index Database

EMP3 can be described by the EMPLOYEE Master File Description. You should clear the original USE list, then reissue the USE command concatenating EMP3, but omit the references to the EMPLOYEE and EMP2 databases. USE must also point to EMPIDX, the index database, in order for EMPIDX to be updated with information from EMP3. You can run REBUILD from a FOCEXEC, rather than interactively as follows. The FOCEXEC contains:

```
USE CLEAR
USE
EMP3 AS EMPLOYEE
JOBFILE
EDUCFILE
EMPIDX
END

REBUILD
EXTERNAL INDEX
ADD
EMPIDX
EMPLOYEE
CURR_JOBCODE
NO
NO
```

Now that the index database has been updated with records from EMP3, you can issue the USE to activate the external index database, and then execute your TABLE request. Note that the USE list references EMPLOYEE, EMP2, and EMP3, and EMPIDX uses the WITH option:

```
USE CLEAR
USE
EMPLOYEE
EMP2 AS EMPLOYEE
EMP3 AS EMPLOYEE
JOBFILE
EDUCFILE
EMPIDX WITH EMPLOYEE
END
TABLE FILE EMPLOYEE.CURR_JOBCODE
PRINT CURR_JOBCODE
IF CURR_JOBCODE EQ 'A07'
END
```

### **Example** Positioning an Indexed Field

The Master File Description for the file AUTOINS contains:

```
FILE=AUTOINS , SUFFIX=FOC , $
SEGNAME=STATE , SEGTYPE=S1 , $
  FIELD=STATE_CODE , STATC , A2 , $
  FIELD=STATE_COMPANY , STATCOMP , A10 , $
SEGNAME=POLICY , PARENT=STATE , SEGTYPE=S1 , $
  FIELD=AGENCY_NO , AGNO , A2 , $
  FIELD=EXPIRE_DATE , EXPDTE , MDY , $
  FIELD=EFFECT_DATE , EFFDTE , MDY , $
  FIELD=HOMEOWNER , OWNER , A3 , $
SEGNAME=VEHICLE , PARENT=POLICY , SEGTYPE=S1 , $
  FIELD=VIN , VIN , A9 , $
  FIELD=MODEL , MODEL , A15 , $
  FIELD=MAKE , MAKE , A15 , $
  FIELD=YEAR , YEAR , YY , $
SEGNAME=PRODUCTS , PARENT=VEHICLE , SEGTYPE=S1 , $
  FIELD=PROD_CODE , PCODE , A2 , $
  FIELD=COVERAGE , COVER , I8 , $
  FIELD=CLASS_CODE , CCODE , A4 , $
  FIELD=RATING , RATE , A3 , $
```

The output from CHECK FILE AUTOINS PICT is:

```
STATE
01      S1
*****
*STATE_CODE **
*STATE_COMPA>**
*          **
*          **
*          **
*****
*****
      I
      I
      I POLICY
02      I S1
*****
*AGENCY_NO **
*EXPIRE_DATE **
*EFFECT_DATE **
*HOMEOWNER **
*          **
*****
*****
      I
      I
      I VEHICLE
03      I S1
```

```

*****
*VIN          **<---- Target
*MODEL        **
*MAKE         **
*YEAR         **
*             **
*****
*****
          I
          I
          I PRODUCTS
04         I S1
*****
*PROD_CODE    **<---- Source
*COVERAGE    **
*CLASS_CODE   **
*RATING       **
*             **
*****
*****

```

The PRODUCTS segment is the source segment, because the external index is built from a DEFINEd field, SPCODE, whose value is based on PROD\_CODE. By associating the index with the VEHICLE segment by choosing the VIN field in response to the prompt, "ASSOCIATE INDEX WITH A PARTICULAR FIELD", the VEHICLE segment now becomes the target segment. The following FOCEXEC creates this relationship:

```
DEFINE FILE AUTOINS
SPCODE/A2=PROD_CODE;
END
REBUILD
EXTERNAL INDEX
NEW
AUTOIDX
AUTOINS
SPCODE
YES
VIN
YES
WHERE SPCODE EQ 'A3'
END
```

The following is displayed by REBUILD when the process is complete:

```
EXTERNAL INDEX FILE:          AUTOIDX
DATE/TIME OF LAST CHANGE:    11/16/94 10.33.58
```

```
EXTERNAL INDEX FILE PAGES:  00000001
DATABASE INDEXED:          AUTOINS
FIELD NAME:                 SPCODE
FIELD FORMAT:               A2
SEGMENT NAME:               VEHICLE
SEGMENT LOCATION:          AUTOINS
```

EXTERNAL INDEX DATA COMPONENTS:

```
AUTOINS
```

## Reference Special Considerations

- Up to eight indices can be activated at a time in a USE list via the WITH statement. More than eight indices may be activated in a FOCUS session by USE CLEAR and issuing new USE statements.
- MODIFY may only make use of the external index via the FIND or LOOKUP functions. The external index cannot be used as an entry point, such as with MODIFY FILE filename.index.
- Indices may not be created on fieldnames longer than twelve characters.
- Text fields may not be used as indexed fields.
- The USE options NEW, READ, ON, LOCAL, and AS master on userid READ are not supported for the external index database.
- The external index database need not be allocated as CREATE FILE automatically does a temporary allocation. If a permanent database is required then an allocation for the index database ddname must be in place prior to the REBUILD EXTERNAL INDEX command.
- SORTOUT, the workfile that the REBUILD EXTERNAL INDEX process creates, must be allocated with adequate space. The FOCUS default size is SPACE=(TRK,(5,5)), which may not be large enough to accommodate all retrieved index records. In order to estimate the space needed for SORTOUT, the following formula may be used:
- bytes = (field\_length + 20) \* number\_of\_occurrences

## Reference Error Messages

(FOC970)      EXTERNAL INDEX SYNTAX ERROR: CONFLICTING OPTIONS

External index syntax does not support following USE command options: NEW, READ, ON, LOCAL, AS master ON userid READ.

(FOC971)      **EXTERNAL INDEX SYNTAX ERROR: ILLEGAL DDNAME**

Key word WITH must be followed by a valid DDNAME, whose length is greater than 0 and less than or equal to 8 characters.

(FOC972)      **EXTERNAL INDEX ERROR: MORE THAN 8 INDICES IN USE**

There can be maximum of 8 external indices in USE.

(FOC973)      **EXTERNAL INDEX INVALID CONFIGURATION:**

Not all components used to build concatenated external index are in use at this time, it is not safe to use this index.

(FOC974)      **EXTERNAL INDEX ERROR: INVALID TARGET SEGMENT**

Target segment for external index is invalid. Real database fields cannot have a target segment different from the location segment.

(FOC975)      **EXTERNAL INDEX ERROR: SEGMENT NOT IN PATH**

(FOC976)      **EXTERNAL INDEX EXPIRED:**

One of the component databases was updated after the index was generated. It is not safe to use index under these conditions.

(FOC977)      **EXTERNAL INDEXES CANNOT BE CREATED FOR NON-FOCUS DATABASES**

External indexes may only be created for databases with SUFFIX=FOC.

(FOC999)      **WARNING.    EXTERNAL INDEX COMPONENT REUSED: ddname**



Attempt is made to add data to an external index from a database that was previously used as a component.

(FOC1000)    **INVALID USE OF CROSS REFERENCE FIELD**

Cross reference fields may only be used as an external index if they are associated with a real database field. An external index cannot be associated with a cross reference field.

## NF426: AUTOPATH

The AUTOPATH feature introduced in Release 7.0 dynamically selects an optimal retrieval path for accessing a FOCUS file. This is done by analyzing the file's structure and the fields referenced, and choosing the lowest possible segment as the entry point. The feature enables users to optimize requests without first having to analyze both the structure of the FOCUS file, and any DEFINE statements in effect (implicitly or explicitly). AUTOPATH is the automation of TABLE FILE ddname.fldname syntax, where fldname is not indexed and physical retrieval starts at the fldname segment.

### Initiating AUTOPATH

Activate AUTOPATH by issuing the SET command

```
SET AUTOPATH = {ON|OFF}
```

where:

**ON**

Turns AUTOPATH on. ON is the default.

**OFF**

Turns AUTOPATH off.

### Reference Special Conditions

- The use of AUTOPATH is entirely invisible and automatic, although choosing certain FOCUS features (such as LAST and self-referencing DEFINES) will restrict use of AUTOPATH.

- AUTOPATH affects the order of displayed data. If no BY field is specified, AUTOPATH displays the detail records in the order in which they are physically stored in the database. If sorted output is required, a BY field should be specified.

## NF427: Longer Length for HOLD FORMAT LOTUS Files

You can now generate LOTUS-formatted HOLD files with record lengths (LRECLs) up to 512 bytes. Releases prior to 7.0 supported a maximum LRECL of 240 for HOLD FORMAT LOTUS files.

## NF428: Increased Size of FOCSORT

The maximum size of FOCSORT has been increased to 1G, or 256K pages.

## NF429: Generalized Listings of DDNAMEs

Release 7.0 supports generalized listings of ddnames beginning with any prefix you specify, using the ? TSO DDNAME command. The syntax is

```
? TSO DDNAME abc*
```

where:

abc

Represents the first n (at least 3, or up to 8) characters of the ddnames you wish to list. This example would list all ddnames beginning with abc, regardless of what followed the first three characters.

You may also use a ? as a single character wildcard. For example, ? TSO DDNAME FOC???? could be used to find the ddnames FOCEXEC and FOCSORT.

**Note:** ? MVS DDNAME is synonymous with ? TSO DDNAME.

## NF431: Universal Concatenation

All data, regardless of source can now be concatenated to appear as if from a single file. Universal Concatenation is designed to support the retrieval of data from unlike database types in a single retrieval request. MORE, the new FOCUS keyword is used to concatenate any type of database (FOCUS, DB2, IMS, VSAM, etc.) provided they share corresponding fields with the same format. Screening of data (IF and WHERE) can be used with MORE.

Retrieval using MORE is divided into a main request and subrequests. The main request defines the data fields, the sorting criteria, and the output format for all data. The subrequests simply define the files and the data fields to be concatenated to the data of the main request. During retrieval, FOCUS gathers data from the first database, and then gathers data from the succeeding databases. All data is then sorted together and output as described in the main request.

### The MORE Subcommand

Data from files with dissimilar Master File Descriptions can be retrieved as a single file using the MORE subcommand of TABLE, GRAPH, or MATCH.

For example, the EMPLOYEE file and the EXPERSON file have Employee information. With MORE, you can concatenate their common data into a single file.

Employee	
Emp_id	Curr_sal
123456789	50.00

Experson	
SSN	Wage
987654321	100.00

```
DEFINE FILE EXPERSON
EMP_ID/A9 =SSN;
CURR_SAL/D12.2 = WAGE;
END

TABLE FILE EMPLOYEE
PRINT CURR_SAL
BY EMP_ID
MORE
FILE EXPERSON
END
```

Report Output	
<u>Emp_id</u>	<u>Curr_sal</u>
123456789	50.00
987654321	100.00



## Syntax How to Concatenate Data Sources With MORE

The syntax is

```
{TABLE|GRAPH|MATCH} FILE file1
    main request
MORE
FILE file2
    subrequest
MORE
FILE file3
    subrequest
MORE
.
.
.
{END|RUN}
```

where:

`TABLE|GRAPH|MATCH`

All MORE subrequests must be issued from within TABLE, GRAPH, or MATCH commands.

`file1`

Is the name of the first file.

`main request`

Is a request without END or RUN. Sorting, formatting, aggregation, COMPUTEs, etc. for all data must be described within this request. IF and WHERE commands used within the main request only screen data from this database.

**MORE**

Begins a subrequest. There is no limit on the number of subrequests (other than available memory).

**FILE file2**

The next file for concatenation follows the keyword FILE.

**subrequest**

Subrequests may only include the WHERE and IF subcommands.

**RUN|END**

Ends a request.

## **Fieldname and Format Matching**

All fields referenced in the main request must be DEFINEd to the file(s) named in the subrequest(s). Referenced fields include those used in COMPUTEs, HEADINGS, aggregation, and sorting as well as those used after the PRINT, LIST, SUM, COUNT, WRITE, or ADD commands.

Field names in the main request must correspond with those in the subrequests in both name and format. A DEFINE FILE statement must be used where a name or format needs to be remapped or changed to the corresponding field in the main request.

A successful format match means:

**Format Type**

**Correspondence**

A	Format type and length must be equal.
I,F,D	Format type must be the same.
P	Format type and scale must be equal.
DATE (new)	Always correspond.
DATE (old)	Edit options must be the same.

**Note:** TEXT fields cannot be concatenated.

### **Example** Using MORE to Concatenate Data

The following three examples show how data is concatenated during a MORE request. The first TABLE request shows the data from EMPDATA. The second TABLE request shows the data from the PAYHIST file. The third TABLE request shows how the data is concatenated using MORE in a single request.

```
TABLE request 1
DEFINE FILE EMPDATA
NEWID/A11 = EDIT
(ID,'999-99-9999');
END
```

```
TABLE FILE EMPDATA
HEADING
"CURRENT EMPLOYEE SALARIES"
" "
PRINT CSAL BY NEWID AS
'EMPLOYEE ID'
WHERE CSAL GT 65000
END
-RUN
```

```
TABLE request 2
DEFINE FILE PAYHIST
NEWID/A11 = EDIT(SSN,'999-99-9999');
CSAL/D12.2M = NEW_SAL;
END
```

```
TABLE FILE PAYHIST
HEADING
"HISTORICAL EMPLOYEE SALARIES"
" "
PRINT CSAL BY NEWID
WHERE NEW_SAL GT 500
END
```

PAGE 1

CURRENT EMPLOYEE SALARIES

EMPLOYEE ID	SALARY
-----	-----
000-00-0030	\$70,000.00
000-00-0070	\$83,000.00
000-00-0200	\$115,000.00
000-00-0230	\$80,500.00
000-00-0300	\$79,000.00

PAGE 1

HISTORICAL EMPLOYEE SALARIES

NEWID	CSAL
-----	-----
100-10-1689	\$842.90
	\$982.90
100-11-9950	\$508.75
100-14-2166	\$876.45
100-15-5843	\$508.75
100-16-2791	\$567.89
100-16-4984	\$1,236.78
100-17-5025	\$734.56
100-18-9299	\$567.89

TABLE request 3

```
DEFINE FILE EMPDATA
NEWID/A11 = EDIT (ID, '999-99-9999');
END
DEFINE FILE PAYHIST
NEWID/A11 = EDIT (SSN, '999-99-9999');
CSAL/12.2M = NEW_SAL;
END
TABLE FILE EMPDATA
HEADING
"EMPLOYEE SALARIES"
" "
PRINT CSAL BY NEWID AS 'EMPLOYEE ID'
WHERE CSAL GT 65000
MORE
FILE PAYHIST
WHERE NEW_SAL GT 500
END
```

PAGE 1

## EMPLOYEE SALARIES

EMPLOYEE ID	SALARY
-----	-----
000-00-0030	\$70,000.00
000-00-0070	\$83,000.00
000-00-0200	\$115,000.00
000-00-0230	\$80,500.00
000-00-0300	\$79,000.00
100-10-1689	\$1,685.80
	\$982.90
100-11-9950	\$508.75
100-14-2166	\$876.45
100-15-5843	\$508.75
100-16-2791	\$567.89
100-16-4984	\$1,236.78
100-17-5025	\$734.56
100-18-9299	\$567.89

## MATCH FILE Considerations

By using MATCH with the MORE subcommand you can concatenate unlimited numbers of files together and merge the concatenated file with up to six sets of concatenated files.

All of the requirements of the MATCH command must be met in the main request. Refer to the *FOCUS for IBM Mainframe Users Manual* for a full discussion of the MATCH command.

## Syntax How to Use MORE With MATCH FILE

The syntax is:

```
MATCH FILE file1
```

```
    main request
```

```
MORE
```

```
FILE file2
```

```
    subrequest
```

```
MORE
```

```
FILE file3
```

```
    subrequest
```

```
RUN
```

```
FILE file4
```

```
    main request
```

```
[AFTER MATCH merge_phrase]
```

```
MORE
```

```
FILE file5
```

```
    sub request
```

```
MORE
```

```
FILE file6
```

```
    subrequest
```

```
RUN
```

```
    FILE file7
```

```
        main request
```

```
[AFTER MATCH merge_phrase]
```

```
MORE
```

```
FILE file8
```

```
    subrequest
```

```
MORE
```

```
FILE file9
```

```
    subrequest
```

```
END
```

First answer set

Second answer set

Third answer set



All of the data concatenated in the first answer set is merged together with the data concatenated in the second answer set using the AFTER MATCH merge\_phrase (e.g., OLD-OR-NEW) in the second answer set. All the merged data from the first and second answer sets, now a HOLD file, is then merged with the data concatenated in the third answer set using the AFTER MATCH merge\_phrase in the third answer set. This final set of merged data is contained in a HOLD file. Following are two examples showing MATCH with MORE:

**Example** Using MORE With MATCH FILE

Example 1:

```
DEFINE FILE EMPDATA
CURR_SAL/D12.2M = CSAL;
FIRST_NAME/A10 = FN;
EID/A9 = PIN;
END
-*
MATCH FILE EMPLOYEE
SUM CURR_SAL AS 'CURRENT'
      FIRST_NAME AS 'FIRST'
BY EID AS 'SSN'
-*
MORE
FILE EMPDATA
RUN
-*
FILE TRAINING
PRINT EXPENSES
BY PIN AS 'SSN'
AFTER MATCH HOLD OLD-OR-NEW
END
-*
TABLE FILE HOLD
PRINT *
END
```

SSN	CURRENT	FIRST	EXPENSES
---	-----	-----	-----
000000010	\$55,500.00	DANIEL	2,300.00
000000020	\$62,500.00	MICHAEL	.
000000030	\$70,000.00	LOIS	2,600.00
000000030	\$70,000.00	LOIS	2,300.00
000000040	\$62,500.00	RUTH	3,400.00
000000050	\$54,100.00	PETER	3,300.00
000000060	\$55,500.00	DORINA	.
000000070	\$83,000.00	EVELYN	.
000000080	\$43,400.00	PAMELA	3,200.00
000000080	\$43,400.00	PAMELA	3,350.00
000000090	\$33,000.00	MARIANNE	.
000000100	\$32,400.00	TIM	3,100.00
000000110	\$19,300.00	ANTHONY	1,800.00
000000110	\$19,300.00	ANTHONY	2,500.00
000000110	\$19,300.00	ANTHONY	2,400.00
000000120	\$49,500.00	KATE	2,200.00
000000130	\$62,500.00	MARCUS	.

Example 2

```
DEFINE FILE EMPDATA
CURR_SAL/D12.2M = CSAL;
FIRST_NAME/A10 = FN;
EID/A9 = PIN;
END
-*
MATCH FILE EMPLOYEE
SUM CURR_SAL AS 'CURRENT'
    FIRST_NAME AS 'FIRST'
BY EID AS 'SSN'
-*
MORE
FILE EMPDATA
RUN
-*
FILE TRAINING
PRINT EXPENSES
BY PIN AS 'EID'
AFTER MATCH HOLD OLD-OR-NEW
END
-*
TABLE FILE HOLD
PRINT *
END
```

Notice how the use of the AS phrase can change the answer set.

SSN	CURRENT	FIRST	EID	EXPENSES
---	-----	-----	---	-----
000000010	\$55,500.00	DANIEL	000000010	2,300.00
000000020	\$62,500.00	MICHAEL	000000030	2,600.00
000000030	\$70,000.00	LOIS	000000030	2,300.00
000000040	\$62,500.00	RUTH	000000040	3,400.00
000000050	\$54,100.00	PETER	000000050	3,300.00
000000060	\$55,500.00	DORINA	000000080	3,200.00
000000070	\$83,000.00	EVELYN	000000080	3,350.00
000000080	\$43,400.00	PAMELA	000000100	3,100.00
000000090	\$33,000.00	MARIANNE	000000110	1,800.00
000000100	\$32,400.00	TIM	000000110	2,500.00
000000110	\$19,300.00	ANTHONY	000000110	2,400.00
000000120	\$49,500.00	KATE	000000120	2,200.00
000000130	\$62,500.00	MARCUS	000000140	3,600.00
000000140	\$62,500.00	VERONICA	000000150	3,400.00
000000150	\$40,900.00	KARL	000000160	1,000.00
000000160	\$62,500.00	ROSE	000000180	1,250.00
000000170	\$30,800.00	WILLIAM	000000190	3,150.00

## NF432: Renaming/Rejustifying Row and Column Total Labels

This FOCUS Release 7.0 feature enables you to rename COLUMN-TOTAL and ROW-TOTAL labels and rejustify them (center, right, or left), as well as override total field formats (for ROW-TOTALs only) at runtime. The new capabilities permit you to improve the aesthetic appearance of reports and easily adjust ROW-TOTALs to accommodate overflow conditions.

### Syntax      How to Rename and Rejustify ROW-TOTAL

The syntax used to rename and rejustify ROW-TOTAL labels and reformat ROW-TOTALs is

```
ROW-TOTAL[/R,L,C] [/format] [AS 'new name']
```

where:

```
/R
```

Right justifies the ROW-TOTAL label.

```
/L
```

Left justifies the ROW-TOTAL label.

```
/C
```

Centers the ROW-TOTAL label.

```
/format
```

Is the new format (i.e., D10.4).

```
new name
```

Is your replacement name for the ROW-TOTAL label.

## Syntax How to Rename and Rejustify COLUMN-TOTAL

The syntax used to rename and rejustify COLUMN-TOTAL labels is

```
COLUMN-TOTAL[/R,L,C] [AS 'new name']
```

where:

```
/R
```

Right justifies the COLUMN-TOTAL label.

```
/L
```

Left justifies the COLUMN-TOTAL label.

```
/C
```

Centers the COLUMN-TOTAL label.

```
new name
```

Is your replacement name for the COLUMN-TOTAL label.

## Example Renaming and Rejustifying COLUMN-TOTAL and ROW-TOTAL

Example 1. Rename and center ROW-TOTAL and COLUMN-TOTAL.

```
TABLE FILE CAR
SUM DCOST
RCOST ROW-TOTAL/C/D12 AS 'TOTAL_COST'
BY COUNTRY
ON TABLE COLUMN-TOTAL/C AS 'FINAL_TOTAL'
END
```

COUNTRY	DEALER_COST	RETAIL_COST	TOTAL_COST
ENGLAND	37,853	45,319	83,172
FRANCE	4,631	5,610	10,241
ITALY	41,235	51,065	92,300
JAPAN	5,512	6,478	11,990
W GERMANY	54,563	64,732	119,295
FINAL_TOTAL	143,794	173,204	316,998

Example 2. Rename and center justify ROW-TOTAL and COLUMN-TOTAL when using ACROSS.

```
TABLE FILE CAR
SUM DCOST
RCOST ROW-TOTAL/C/D12 AS 'TOTAL_COST'
ACROSS COUNTRY
IF COUNTRY EQ 'ENGLAND'
ON TABLE COLUMN-TOTAL/C AS 'FINAL_TOTAL'
END

COUNTRY
ENGLAND
DEALER_COST RETAIL_COST DEALER_COST RETAIL_COST
-----
          37,853      45,319          37,853      45,319

FINAL_TOTAL
          37,853      45,319          37,853      45,319
```

Example 3. Rename and left justify ROW-TOTAL and COLUMN-TOTAL used with COMPUTE.



```
TABLE FILE CAR
SUM DCOST RCOST
COMPUTE PROFIT/D12=RCOST-DCOST;
ROW-TOTAL/L/D12 AS 'TOTAL_COST'
BY COUNTRY
ON TABLE COLUMN-TOTAL/L AS 'FINAL_TOTAL'
END
```

COUNTRY	DEALER_COST	RETAIL_COST	PROFIT	TOTAL_COST
-----	-----	-----	-----	-----
ENGLAND	37,853	45,319	7,466	90,638
FRANCE	4,631	5,610	979	11,220
ITALY	41,235	51,065	9,830	102,130
JAPAN	5,512	6,478	966	12,956
W GERMANY	54,563	64,732	10,169	129,464
FINAL_TOTAL	143,794	173,204	29,410	346,408

## NF433: Improved Handling of Text Fields In TED

FOCUS text fields have been enhanced significantly with Release 7.0, and a new SET command has been introduced to preserve downward compatibility with prior FOCUS releases. The syntax is

```
SET TEXTFIELD = {OLD|NEW}
```

where:

**OLD**

Allows you to use text field data in prior releases of FOCUS when that data has been created or modified in Release 7.0. OLD is the default.

**NEW**

Disables the ability to use text field data in prior FOCUS releases when that data has been created or modified in Release 7.0.

In addition, FOCUS Release 7.0 can now preserve text fields exactly as entered into the database via ON MATCH/NOMATCH TED.

The syntax for defining a text field in a Master File Description is:

```
FIELD=fieldname, ALIAS=aliasname, FORMAT=TXnn,$
```

or

```
FIELD=fieldname, ALIAS=aliasname,FORMAT=TXnnF,$
```

where:

**fieldname**

Is the name you assign the text field.

**aliasname**

Is an alternate name for the fieldname.

nn

Is the output display length in TABLE for the text field.

F

Is used to format the text field for redisplay when TED is called via ON MATCH or ON NOMATCH. When F is specified, the text field is formatted as TX80 and is displayed. When F is not specified, the field is redisplayed exactly as entered.

## Example Handling Text Fields in TED

In the following Master File Description

```
FILE=TEXT,SUFFIX=FOC
  SEGNAME=SEGA,SEGTYPE=S1
  FIELD=KEYFLD, ,A1, $
  FIELD=TXTFLD, ,TX20, $
```

the text field does not include the F option.

The data entered into the field TXTFLD in TED is:

```
THIS IS A TEST OF THE NEW TED OPTION 'F'.  REMEMBER THAT TED DISPLAYS 80
CHARACTERS ON THE SCREEN.  THREE LEADING BLANKS ARE USED TO INDICATE A NEW
PARAGRAPH.TEXT FIELD DATA IS ALWAYS STORED EXACTLY AS ENTERED.  WHEN F IS
INCLUDED IN THE FORMAT AND THE TEXT FIELD IS REDISPLAYED, BLANKS ARE
OMITTED AND THE FIELD IS CONDENSED.
WHEN F IS NOT INCLUDED, THE FIELD IS REDISPLAYED AS ENTERED.
```

When the data is redisplayed in TED (ON MATCH TED TXTFLD), it appears as:

THIS IS A TEST OF THE NEW TED OPTION 'F'. REMEMBER THAT TED DISPLAYS 80 CHARACTERS ON THE SCREEN. THREE LEADING BLANKS ARE USED TO INDICATE A NEW PARAGRAPH.TEXT FIELD DATA IS ALWAYS STORED EXACTLY AS ENTERED. WHEN F IS INCLUDED IN THE FORMAT AND THE TEXT FIELD IS REDISPLAYED, BLANKS ARE OMITTED AND THE FIELD IS CONDENSED.  
WHEN F IS NOT INCLUDED, THE FIELD IS REDISPLAYED AS ENTERED.

**In the next master file description, the text field includes the F option:**

```
FILE=TEXT,SUFFIX=FOC  
  SEGNAME=SEGA,SEGTYPE=S1  
  FIELD=KEYFLD,,A1,$  
  FIELD=TXTFLD,,TX20F,$
```

**The same data is entered as in the previous example:**

THIS IS A TEST OF THE NEW TED OPTION 'F'. REMEMBER THAT TED DISPLAYS 80 CHARACTERS ON THE SCREEN. THREE LEADING BLANKS ARE USED TO INDICATE A NEW PARAGRAPH. TEXT FIELD DATA IS ALWAYS STORED EXACTLY AS ENTERED. WHEN F IS INCLUDED IN THE FORMAT AND THE TEXT FIELD IS REDISPLAYED, BLANKS ARE OMITTED AND THE FIELD IS CONDENSED.  
WHEN F IS NOT INCLUDED, THE FIELD IS REDISPLAYED AS ENTERED.

**However, when the field is redisplayed, blanks are omitted and the field is condensed:**

THIS IS A TEST OF THE NEW TED OPTION 'F'. REMEMBER THAT TED DISPLAYS 80 CHARACTERS ON THE SCREEN. THREE LEADING BLANKS ARE USED TO INDICATE A NEW PARAGRAPH. TEXT FIELD DATA IS ALWAYS STORED EXACTLY AS ENTERED. WHEN F IS INCLUDED IN THE FORMAT AND THE TEXT FIELD IS REDISPLAYED, BLANKS ARE OMITTED AND THE FIELD IS CONDENSED. WHEN F IS NOT INCLUDED, THE FIELD IS REDISPLAYED AS ENTERED.

## Displaying or Printing a Text Field in TABLE

FOCUS prints text fields according to the FORMAT defined in the Master File Description. This default may be overridden by using the /TXnn option in a request. The syntax is

```
PRINT fieldname/TXnn
```

where:

*fieldname*

Is the name assigned in the MFD.

*nn*

Is the output display length of the text on the report. If omitted, the text will be displayed according to the format in the Master File Description.

See the *FOCUS for IBM Mainframe Users Manual* for more information on printing text fields.

## NF434: Larger FOCUS Databases

The maximum size of a FOCUS database has been increased to 1G, or 256K pages. This is a fourfold improvement over previous releases. This change in size excludes files that were created prior to Release 7.0 with SHADOW=ON. The 64 segment limit per physical FOCUS database has not been changed.

### Syntax      How to Set the SET SHADOW Command

A new setting has been introduced in Release 7.0 for the SET SHADOW command. The syntax is:

```
SET SHADOW = OLD
```

where:

**OLD**      Denotes the use of the old (pre-7.0) shadow technology. This means fewer pages are shadowed as compared to SHADOW=ON for large databases in Release 7.0. If your FOCUS file was created with SHADOW=OLD, the maximum number of pages is 63,551.

If the 63,551 limit is exceeded, the following error message will be issued:

```
(FOC198)      FATAL ERROR IN DATABASE I/O. FOCUS TERMINATING CODE=ABCABCAB
```

A severe error was encountered in reading from or writing to a FOCUS database. FOCUS was unable to recover from this error, and will return to the host environment.

## NF435: Increased Number of Literal Values in a File

The previous FOCUS limit of 3200 bytes in a file used for selection criteria has been changed to 32,767 literals.

### **Syntax**      **How to Increase the Number of Literal Values in a File**

This change applies to IF tests that use the following syntax:

```
IF fieldname operator (ddname) [OR (ddname)...]
```

The limit for the WHERE phrase remains unchanged. See the *FOCUS for IBM Mainframe Users Manual* for more information on IF and WHERE syntax.

#### Limitations

- The file against which the report request is run must be a FOCUS file or a relational table (SUFFIX=SQLDS). All other non-FOCUS files continue to have the limit of 3200 bytes.
- When more than one ddname is mentioned in a single IF test, the combined maximum number of literals for all ddnames is 32,767.
- When there is more than one IF test, the 32,767 limit applies to each IF test individually.

## NF436: Checking Current Language Settings

The ? LANG command can be used to determine the current language setting and parameters pertaining to language, such as continental decimal notation.

### **Syntax**      **How to Determine Current Language Settings**

The syntax is:

```
? LANG
```



## NF437: AUTOADBS

The AUTOADBS facility is now available to automatically generate Master File Descriptions (MFD) and Access File Descriptions (AFD) for ADABAS files and userviews based on information stored in the Predict Data Dictionary and user selections. The facility requires FOCUS Release 6.8 or greater, ADABAS Release 5.0 or greater, and Predict Release 2.3.2 or greater. For installation instructions, see Technical Memo 7907 "Installing AUTOADBS for MVS".

AUTOADBS may be executed interactively or in the background in a batch job. Multiple files can be related in one MFD in interactive mode; background mode is limited to one file per execution. One or more versions of the Predict dictionary can be used concurrently to describe several ADABAS files (Production and Test dictionaries, for example). The utility can be executed with both the new and old ADABAS interfaces and can create MFDs for either version. MFDs created for the old interface can be used in the new interface while MFDs created for the new interface can only be used with ADABAS=NEW, but will take advantage of the new features available.

The utility will accurately describe the ADABAS file or userview within the constraints of the interface. This includes superdescriptors, subdescriptors, periodic elements (PE groups) and multi-value fields (MU fields). Those fields that are not supported are described as comments in the generated MFD.

Several options are provided in AUTOADBS that allow the user to customize the output. The NATURAL column heading stored in Predict can be included in the MFD to be used as the default column heading in FOCUS reports. The starting position and length of the Predict fieldname can be specified. This feature allows the user to strip off common prefixes from the fieldname or truncate the fieldname to 12 characters (for example, when downloading data using ON TABLE PCHOLD). Note: The maximum fieldname length with the old ADABAS interface is limited to 12 characters. For NATURAL date fields, the user can specify the FOCUS Smart Date format desired. In addition to creating the MFD and AFD, help files for TableTalk (FOCDEFs) can be created as a user option. The first three lines of the Predict REMARKS field will be used to provide online TableTalk help.

## Documentation in the Master and Access File Descriptions

The descriptions generated by AUTOADBS will contain several commented entries. These are provided as a convenience to the user and are not used by FOCUS. The userid, date, and time of creation are included in both the MFD and AFD. The ADABAS filename is indicated for each file described. Fields not supported by the interface are included in the MFD but are commented out. The complete Predict fieldname and ADABAS format and length are included as comments. This is useful when the fieldname length is truncated by the user. These comments may optionally be excluded from the MFD with a menu option. For PE and MU repeating fields, the maximum number of occurrences is included with the SEGMENT attribute in the MFD and the fieldname is included in the AFD. All duplicate fieldnames are included at the bottom of the MFD with the number of times that the field is encountered and the segment names where they are found. Superdescriptors that are composed of partial fields are described with the NOP suffix and their component fields are included as comments. If the user chooses to include fieldname comments (a menu option), then the start and end position of the partial field is also included as a comment.

### Ease of Use Features

- Default menu parameters can be logged to disk and will be recalled in subsequent interactive AUTOADBS executions.
- An extensive on-line help facility is provided for all menus and screens; in fact, much of the text of this bulletin is also available via this function.

- A list of ADABAS Master files and Userviews described in Predict is available online. The wild card character asterisk (\*) can be used to select a subset of files.
- The Master File Description and the Access File Description can be edited without exiting the AUTOADBS facility.
- The capability to produce a PICTURE from the Main Menu for the named MASTER is provided. (It must be allocated to ddname MASTER in MVS.)
- Many files can be described in one AUTOADBS session.
- AUTOADBS has several parameters that can be customized for site specific situations. Refer to the AUTOADBS Installation instructions for details.

## How to Use the AUTOADBS Facility

Sufficient disk space is required for the facility to write the new file descriptions and parameter log file, if applicable, to the datasets specified on the initial input screen (in MVS) or to the disk accessed as filemode A (in VM/CMS). In addition, sufficient disk space must be available for temporary work files. The amount of temporary space depends on the size of the files being described. In the VM/CMS environment, a minimum of 3 available cylinders is recommended. The AUTOADBS facility will use the temp disk for all temporary files that are created.

For MVS execution, the Master File Descriptions AUTOADBS, PREDDB, and PREDEL must be members in a library allocated to the ddname MASTER. The Access File Descriptions PREDDB and PREDEL must be members in a library allocated to the ddname FOCADBS. The AUTOADBS FOCEXEC must be a member in a library allocated to the ddname FOCEXEC. These members should be located in your site's production FOCUS libraries as a result of the Interface installation process.

For VM/CMS execution, the files AUTOADBS FOCEXEC, AUTOADBS MASTER, PREDDB MASTER, PREDDB FOCADBS, PREDEL MASTER, and PREDEL FOCADBS should all be on a minidisk available to the user. This is usually the FOCUS production disk, and is a result of the Interface installation process.

Before you begin, you should decide which files are required for your Master File. Determine which Predict dictionary describes the file (if your site installation supports multiple dictionary access from AUTOADBS), the file name, database number, and file number. (If the database number is not included in Predict, you may optionally supply a value in AUTOADBS.) For each file described, you should know which field should be used as the SEQFIELD, the desired CALLTYPE, and the ADABAS security password (if one is assigned). Consider which file should be the root segment and, if necessary, which files will be included as descendants. When describing multiple files in one view, determine which field in each file will be used as the cross-reference key (KEYFLD and IXFLD). Contact your site's ADABAS DBA for specific details of the files you are working with. Since you can create several Master File Descriptions in the same AUTOADBS session you may want to prepare several file views before proceeding.

## Starting AUTOADBS

To start AUTOADBS, enter the FOCUS environment and type at the FOCUS command level:

```
EX AUTOADBS
```

Press the ENTER key.

### Example The Initial AUTOADBS Screen in MVS

The following is an example of the initial AUTOADBS screen in MVS. Subsequent user entries are in lower case.

```
Main Menu      Master File Generation Facility for ADABAS

Master Filename =====> sample

Describe ADABAS Files:
File Name =====> sysdic*
  DBnr =====> *      Fnr =====> *
PREDICT Dictionary=>    ( )

Description will be a member of:
Master Target PDS =====> USER1.MASTER.DATA
Access Target PDS =====> USER1.FOCADBS.DATA
FOCDEF Target PDS                                     =====>
USER1.FOCDEF.DATA
```

```
Replace Existing Description?=>N Start with Fieldname Position=> 1(1-32)
Include Fieldname Comments? ==>Y           for a Total Length of =>36(5-36)
Use NATURAL Column Headings? =>N Date Display Format===== YY  D
Use COMMENTS for FOCDEF? ===== N Display all Userviews? =====N
Use NEW ADABAS Interface? =====Y
Parm File => USER1.FOCADBS.DATA
```

```
PF1=Help PF2=Restart PF3=Exit PF4=Log PF5=MFD PF6=AFD PF9=Picture
PF10=List
```

## **Example** The Initial AUTOADBS Screen in VM/CMS

The following is an example of the initial AUTOADBS screen in VM/CMS. Subsequent user entries are in lower case.

```
Main Menu   Master File Generation Facility for ADABAS
```

```
Master Filename =====> sample
```

```
Describe ADABAS Files:
```

```
File Name =====> sysdic*
```

```
DBnr =====> *      Fnr =====> *
```

```
PREDICT Dictionary=>      ( )
```

Replace Existing Description?=>N Start with Fieldname Position=> 1(1-32)  
Include Fieldname Comments? ==>Y for a Total Length of =====>36(5-36)  
Use NATURAL Column Headings? => N Date Display Format =====> YYM D  
Use COMMENTS for FOCDEF? =====> N Display all Userviews? =====> N  
Use NEW ADABAS Interface? =====> Y  
Parm File => ADBS\$PRM FOCADBS A

PF1=Help PF2=Restart PF3=Exit PF4=Log PF5=MFD PF6=AFD PF9=Picture  
PF10=List

**Note:** With the exception of the Main Menu, AUTOADBS screens are identical in MVS and VM/CMS.

On the Main Menu, provide information about the files to be described, a filename to use in report requests, and target datasets for the descriptions generated (MVS only). Select the appropriate options desired. Specify the following information on the AUTOADBS Main Menu:

MASTER FILENAME

Enter a 1 to 8 character name for the file description. (In MVS, this name must be a valid member name. In CMS, this name must be a valid filename.)

DESCRIBE ADABAS FILES

You may enter any combination of File Name, DBnr, or Fnr.



FILE NAME

Enter the 1 to 32 character ADABAS file name as registered in the Predict dictionary. The wild card character '\*' may be used anywhere within the file name to create a list that matches the provided pattern. The '\*' is a multi-character mask. Entering only an '\*' will select all files.

DBNR

Enter the 1 to 3 digit ADABAS Database Identifier of the files that you wish to describe, or enter an '\*' to select all database numbers.

FNR

Enter the 1 to 3 digit ADABAS file number of the files that you wish to describe, or enter an '\*' to select all file numbers.

PREDICT DICTIONARY

Enter the one character dictionary suffix for the PREDICT dictionary where the desired files are described. The available values are listed on the menu in parentheses, and are determined by the ADABAS DBA when AUTOADBS is installed. If only one dictionary is available (the suffix is blank), you will not be able to enter this field.

MASTER TARGET PDS

(MVS only)

Enter the fully qualified dataset name of the Master File PDS where the MFD will be stored. Do not use quotes in the dataset name.

ACCESS TARGET PDS

(MVS only)

Enter the fully qualified dataset name of the Access File PDS where the AFD will be stored. Do not use quotes in the dataset name.

FOCDEF TARGET PDS

(MVS only)

Enter the fully qualified dataset name of the FOCDEF File PDS where the TableTalk Help file will be stored. Do not use quotes in the dataset name. This entry is only required if 'Use COMMENTS for FOCDEF?' is 'Y'.

REPLACE EXISTING  
DESCRIPTION?

(Y/N)

Enter 'N' if you do not want to overwrite an existing MFD/AFD. Enter 'Y' if you wish to replace an MFD/AFD that already exists on disk.

INCLUDE FIELDNAME  
COMMENTS?

(Y/N)

Enter 'N' if you wish to exclude field comments in the MFD. Enter 'Y' if you wish to include them. Comments contain the full Predict fieldname, '\*TRUNC\*' if the name is truncated, the ADABAS format and length, and start/end characters for superdescriptor elements.

USE NATURAL COLUMN  
HEADINGS?

(Y/N)

Enter 'Y' if you wish to use the NATURAL heading as the column heading in reports. Enter 'N' if you wish to use the fieldname for headings.

USE COMMENTS FOR  
FOCDEF?

(Y/N)

Enter 'Y' to use Predict comments as HELP for TableTalk. Enter 'N' if help is not desired. FOCDEF Target PDS is required in MVS with 'Y'.

USE NEW ADABAS  
INTERFACE?

(Y/N)

Enter 'Y' if you wish to take advantage of features in the new ADABAS interface. You must SET ADABAS=NEW when using this MFD. Enter 'N' if you wish to use the MFD with SET ADABAS=OLD or NEW.

START WITH FIELDNAME  
POSITION

(1-32)

Enter the starting position of the element name to be used as the first character in the fieldname. By increasing the starting position, you may strip off common prefixes from the element name.

FOR A TOTAL LENGTH OF

(5-36 with ADABAS=NEW) (5-12 with ADABAS=OLD). Enter the maximum length of fieldnames described with AUTOADBS. Predict fieldnames can be up to 32 characters and AUTOADBS can add 4 character suffixes to them. The maximum length with ADABAS=OLD is 12 characters.

DATE DISPLAY FORMAT

Enter a valid smart date format for fields described in Predict as dates. Formats may include any valid combination of YY, Y, M, or D and can use month translation (T or TR) and day translation (W or WR). You may also select to display the day of week (W or WR), the quarter (Q, YQ, or YYQ), or Julian dates (JUL). See the manual for a list of other formats.

DISPLAY ALL USERVIEWS?

(Y/N)

Enter 'Y' if you wish to display all master files and userviews for the selected file(s). This option is useful when selecting particular filenames; all userviews are displayed when selecting only DBNRs or FNRs. You may also toggle the display from the file list menu.

The following functions are available from the main menu via the PF keys:

## Help (PF1)

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

## **Refreshing the List of Files (PF2)**

Press PF2 to clear the existing list of files maintained by AUTOADBS for this session. When you select files from the Main Menu, information is gathered from the PREDICT dictionary. Each time you change the selection criteria, the new files are appended to the existing list. Pressing PF2 will purge this list without exiting AUTOADBS. You will receive the message 'Enter new file selection criteria' when the list is successfully purged. No message will be displayed when PF2 is pressed and you have not yet created a list of files.

## **Exit (PF3)**

Press PF3 to exit AUTOADBS.

## **Logging Default Menu Parameters (PF4)**

Press PF4 to save your customized default values for the Main Menu to disk. In MVS, these will be saved in member ADBS\$PRM in the parm dataset shown on the menu. This dataset will be: the PDS pre-allocated to ddname ADBS\$PRM, prefix.FOCADBS.DATA, or the first dataset allocated to ddname FOCADBS (see the section Search Order for Parameter Log file for further details). Parameter logging assumes that the user has write access to the target dataset. An attempt to log parameters to a dataset without write access will result in a security abend. In CMS, these will be saved to filename ADBS\$PRM FOCADBS A.

The following information will be logged:

- \* File name
- \* DBnr

- \* Fnr
- \* Predict dictionary
- \* MFD Partitioned Dataset Name (MVS only)
- \* AFD Partitioned Dataset Name (MVS only)
- \* FOCDEF Partitioned Dataset Name (MVS only)
- \* Replace existing description
- \* Include fieldname comments
- \* Use NATURAL column headings
- \* Create FOCDEF file
- \* Use NEW ADABAS Interface
- \* Start with Fieldname Position
- \* For a total length of
- \* Date display format
- \* Display all userviews

**Note:** All validations must pass before the default values are logged.

### **TED MFD (PF5)/TED AFD (PF6)**

Press PF5 or PF6 to edit (using TED) the master or access file description entered in MASTER FILENAME. In MVS, this will select the member in the dataset named in either MASTER TARGET PDS or ACCESS TARGET PDS. In CMS, this will select mastername MASTER A or mastername FOCADBS A.

**Note:** These files did not have to be created with AUTOADBS to edit them.

## Picture of MFD (PF9)

Press PF9 to generate a diagram of the structure of the file entered in MASTER FILENAME. After the picture is displayed, type in any character and press the ENTER key. In MVS, the mastername must be a member of a dataset allocated to ddname MASTER to generate the picture (the MASTER TARGET PDS is not used).

## File List (PF10)

Press PF10 to display a list of all files that meet the screening criteria provided in file name, DBnr, and Fnr. The Display All Userviews prompt is not used for this report. The report is sorted depending on the selection criteria. The precedence is: DBnr, Fnr, and file name.

**Note:** All validations must pass for these PFkey options.

Enter the appropriate values on the initial screen, and press the ENTER key. AUTOADBS will inform you that it is accessing the Dictionary by issuing the following message:

```
*****  
** Retrieving FILE information from dictionary      **  
** Please wait...                                  **  
*****
```

This message is only displayed during the first retrieval per AUTOADBS session, or when the file selection criteria has changed since the previous retrieval (within the session). AUTOADBS will not access the dictionary a second time for the same immediate selections in a single session.



## The File Selection Screen

When file information retrieval has been completed, the File Selection Screen will be displayed. The following example is for all of the Predict dictionary files (SYSDIC\* was entered as the file name on the Main Menu). Subsequent user entries are in lower case.

### Example Entering SYSDIC\* in the File Selection Screen Main Menu

```
Master:          Master File Generation Facility for AUTOADBS  Show
userviews: N
SAMPLE          ==File Selection==          Sort: FILENAME
Select the ROOT file with 'R', children files with 'C'.
When a file exists in several databases, select the one entry that
describes
the file/database combination desired.
```

R/C	File	A	Fnr	DBnr	Database	D
	SYSDIC	A	012	000	SYSDIC	
r	SYSDIC-DB	U	012	000	SYSDIC	
	SYSDIC-DDM	U	012	000	SYSDIC	
	SYSDIC-DEF	U	012	000	SYSDIC	
	SYSDIC-DESC	U	012	000	SYSDIC	
	SYSDIC-EL	U	012	000	SYSDIC	
c	SYSDIC-FI	U	012	000	SYSDIC	
	SYSDIC-FI-ADA	U	012	000	SYSDIC	
	SYSDIC-FI-GEN	U	012	000	SYSDIC	
	SYSDIC-IMS	U	012	000	SYSDIC	
	SYSDIC-KY	U	012	000	SYSDIC	
	SYSDIC-MO	U	012	000	SYSDIC	

PF1=Help PF2=Sort PF3=End PF4=Add Files PF5=Views PF7=Up  
PF8=Down

The File Selection Screen is used to choose the files to be described. This screen displays all files that match the selection criteria provided on the main menu. Userviews for selected file names will be displayed if you selected 'Display all Userviews' on the Main Menu. You can toggle between including the userviews and excluding them by pressing PF5. The list is originally sorted by file name. You may toggle the sort order by pressing PF2.

Select the root file by placing an 'R' in the column labeled 'R/C'. Select all other files to be used in the view by placing a 'C' in the 'R/C' column. After selecting the files, AUTOADBS will retrieve field information for the files, if they have not yet been retrieved. Then the Access File Attribute screen will be presented for each of the selected files.

The File Selection Screen displays the following information:

Master	Master file description name provided on the Main Menu.
Userviews	'Y' indicates that Userviews are included on the list. 'N' indicates that Userviews are not included.
Sort	Sort order of the file list: FILENAME, FNR, or DBNR.
R/C	Select the root File with 'R', children Files with 'C'.
File	File name of selected files.
A	File type: A=ADABAS Master file, U=Userview.
Fnr	File Number.
DBnr	Database Number.
Database	Database name (only the first 28 characters are displayed).
D	Predict dictionary suffix selected on Main Menu.

The following functions are available from the File Selection Screen via the PF keys:

## Help (PF1)

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

## Sort (PF2)

Toggle the sort order: By File Name, By Fnr, By DBnr.

## Exit (PF3)

Press PF3 to exit the file description generator and return to the main menu.

## Add Files (PF4)

Return to the Main Menu to add additional files to the list.

## Views (PF5)

Toggle between including userviews and excluding them.

## Scroll Backward (PF7)/Scroll Forward (PF8)

Press PF7 to scroll the record list backward; press PF8 to scroll the record list forward.

After a file or files have been selected, AUTOADBS will inform you that it is accessing the Dictionary for field information by issuing the following message:

```
**=====**
** Retrieving FIELD information from dictionary      **
** Please wait...                                  **
**=====**
```

This message is displayed once for each Predict dictionary that must be accessed, based on the location of the files selected. This message does not appear if all selected files (or any of their associated userviews) have already been selected during this AUTOADBS session.

## The Access File Attribute Screen

When the field information retrieval is completed, the Access File Attribute screen will be presented for each of the selected files. Subsequent user entries are in lower case.

```
Master:          Master File Generation Facility for AUTOADBS
SAMPLE          ==Access File Attributes==
```

```
File:   SYSDIC-DB          Dictionary:
Enter:  CALLTYPE  RL      PASSWORD          DBNR  000
```

```
Select SEQFIELD with 'S' (Natural Default:
```

```
)
```

S	Fieldname	Alias	Usage	Actual	D	U	S	Ty	L
s	DATA_BASE_REC	S1	A34	A34	D	N	SP	1	
	FILE_TYPE_EL	S7	A33	A33	D	N	SP	1	
	DATA_BASE_TYPE	AT	A1	A1	D	N		1	
	L_DBNR	AU	P3	Z3	D	N		1	
	P_DBNR	DP	P5	Z5	D	N		2	
	IMS_DB_NAME	AS	A8	A8	D	N		2	
	DB2_DB_NAME	A1	A8	A8	D	N		2	

```
PF1=Help  PF2=Restart  PF3=End
```

```
PF7=Up    PF8=Down
```

The Access File Attributes screen is used to describe attributes that allow the interface to translate report requests into direct ADABAS calls. This screen will be displayed once for each file selected, in file name order. The cursor is initially placed at the first field to be selected as a SEQFIELD. You may tab to the CALLTYPE, PASSWORD, and DBNR fields.

Select the SEQFIELD by placing an 'S' in the column labeled 'S'. The site installation may require that you select a SEQFIELD. The recommended SEQFIELD (as supplied by Predict) is displayed at the top of the screen. AUTOADBS displays only those fields that can be used as a SEQFIELD on this screen. The SEQFIELD must be a descriptor, a superdescriptor, or a subdescriptor. It cannot be a PE group, an element of a PE group, an MU field, or a reformatted field.

Enter either RL or FIND as the CALLTYPE. The default is RL. If the site installation requires a CALLTYPE of RL, you cannot change this value.

Enter the password if this file is password protected.

Enter the DBNR if you have access to multiple databases in your ADABAS environment. If this value is 0, the DBNR will be determined from the DBID attribute in the ADABAS DDCARD available to your session. You can only change this value if it is 0 in the Predict dictionary. Otherwise, the Predict value will be used.

The Access File Attribute Screen displays the following information:

Master	Master file description name provided on the Main Menu.
File	File name of selected file.

Dictio nary	Predict dictionary suffix selected on Main Menu.
CALLTY PE	Enter RL or FIND. If the site installation requires a CALLTYPE of RL, you will not be able to change this value. <b>Note:</b> With ADABAS=OLD, CALLTYPE will be changed to RL for the child file in short-to-long JOINS and CALLTYPE will be changed to FIND when the IXFLD is a '.NOP' descriptor.
PASSWO RD	Enter the password for password protected files.
DBNR	Enter the DBNR for files whose Predict DBNR is 0.
S	Enter 'S' for the selected SEQFIELD.
Fieldn ame	Predict fieldname.
Alias	ALIAS used in the MFD. This is the ADABAS name.
Usage	Usage format of the field in the MFD.
Actual	Actual format of the field as described in the MFD.
D	Descriptor flag. Value is 'D' for descriptors.
U	Unique flag. Value is 'U' if descriptor is unique.

S	Null suppression option. Value is 'N' for null suppression, 'F' for fixed format, or blank.
Ty	Field type. Values are: GR-Group, PE-Periodic Group, MU-Multi-value field, SP-Superdescriptor, SB-Subdescriptor.
L	Level number. Values are from 1 to 7.

The following functions are available from the Access File Attribute Screen via the PF keys:

### **Help (PF1)**

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

### **Restart (PF2)**

Press PF2 to cancel all previous selections and restart the master file generation process at the File Selection Screen.

### **Exit (PF3)**

Press PF3 to exit the file description generator and return to the Main Menu.

### **Scroll Backward (PF7)/Scroll Forward (PF8)**

Press PF7 to scroll the record list backward; press PF8 to scroll the record list forward.



## The Child Selection Screen

If only one file was selected on the File Selection Screen, the description will be generated and you will be returned to the Main Menu. Otherwise, you will be asked to select the children of the root file with the following screen. Subsequent user entries are in lower case.

```
Master:      Master File Generation Facility for AUTOADBS
SAMPLE      ==Child Selection==
```

```
File:   SYSDIC-DB                      Dictionary:
```

```
Select children files with 'S'
```

```
S  File                                Fnr Dictionary
-  -----
s  SYSDIC-FI                          012
```

```
PF1=Help  PF2=Restart  PF3=End  PF4=None  PF5=Picture  PF7=Up  PF8=Down
```

The Child Selection Screen is used to choose descendants of the file shown on the top of the screen. When files are selected as children, the IXFLD Selection screen will be displayed, once for each selected file.

Place an 'S' at each file desired and press ENTER. Press PF4 if none of the files are to be assigned as children of the parent segment.

The file relationships will be described in top down, left to right order. Initially, this screen will display the root file and all files selected as children. After returning from the IXFLD and KEYFLD Selection Screen for the root, the first child of the root will be displayed at the top of the screen. Descendants can then be assigned to this file. Child selection continues down this first path until the user selects no descendants (by pressing PF4). Only then will the second child of the previous parent be displayed at the top of the screen. This continues until all files have been assigned children or all possible descendants have been exhausted. At this point, the description will be generated and the Main Menu will be redisplayed.

The Child Selection Screen displays the following information:

Master	Master file description name provided on the Main Menu.
File	The current file being assigned children.
Dictionary	The Predict dictionary where this file is described.
S	Select the children files by placing an 'S' in this column.
File	Files to be assigned as children for the shown parent.
Fnr	File number.
Dictionary	Predict dictionary where this file is described.

The following functions are available from the Child Selection Screen via the PF keys:

### **Help (PF1)**

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

### **Restart (PF2)**

Press PF2 to cancel all previous selections and restart the master file generation process at the File Selection Screen.

### **Exit (PF3)**

Press PF3 to exit the file description generator and return to the main menu.

### **None (PF4)**

Press PF4 if you do not wish to describe any of the displayed files as descendants of the current parent file.

### **Creating a Picture of the Description (PF5)**

Press PF5 at any time to generate a diagram of the structure of the file being described. Note that the description will be created on disk (and will remain there even if the program is ended with PF3). After the picture is displayed, type in any character and press the ENTER key. In MVS, the target master PDS must be allocated to ddname MASTER to generate the picture. If a member exists with the master name selected in a dataset concatenated in front of the target dataset, the picture will be generated from that member.

## **Scroll Backward (PF7)/Scroll Forward (PF8)**

Press PF7 to scroll the record list backward; press PF8 to scroll the record list forward.

## **The IXFLD Selection Screen**

Once all children of the root have been selected and the ENTER key has been pressed, the following screen is displayed. Subsequent user entries are in lower case.

Master: Master File Generation Facility for AUTOADBS  
 SAMPLE ==IXFLD Selection==

File: SYSDIC-FI Dictionary:  
 Parent: SYSDIC-DB Dictionary:

Select the common field (IXFLD) from the descendant file with 'S'

S	Fieldname	Alias	Usage	Actual	D	U	S	Ty	L
	HOLD_USER	AF	A8	A8	D	N			1
s	FILE_REC	S6	A33	A33	D	N	SP		1
	L_FNR	AR	P3	Z3	D	N			1
	ADANET_FILE_TYPE	D8	A1	A1	D	N			2
	ADABAS_AVB_REC	S4	A34	A34	D	N	SP		1
	IMS_SEGMENT_NAME	AX	A8	A8	D	N			2
	IMS_PARENT_FILE	AY	A32	A32	D	N			2
	IMS_SOURCE_FILE1	A2	A32	A32	D	N			2
	IMS_SOURCE_FILE2	A3	A32	A32	D	N			2
	DB2_EDIT_PROGRAM	D9	A8	A8	D	N			2
	DB2_VALID_PROGRAM	EA	A8	A8	D	N			2
	DSNAME	DL	A44	A44	D	N			2

PF1=Help PF2=Restart PF3=End PF4=Skip PF7=Up PF8=Down

The IXFLD Selection Screen is used to identify the Foreign key in the child file. Select the common field in the cross-reference file by entering an 'S' in the column labeled 'S'. Press PF4 to skip this file as a descendant if it was selected in error. This will not affect previous or subsequent selections. AUTOADBS displays only those fields that can be used as an IXFLD on this screen. The IXFLD must be a descriptor, a superdescriptor, or a subdescriptor. It cannot be a PE group, an element of a PE group, an MU field, or a reformatted field.

The IXFLD Selection Screen displays the following information:

Master	Master file description name provided on the Main Menu.
File	The child file from which the IXFLD is being selected.
Dictionary	Predict dictionary where this file is described.
Parent	The parent file name.
Dictionary	Predict dictionary where this file is described.
DBNR	Enter the DBNR for files whose Predict DBNR is 0.
S	Enter 'S' for the selected IXFLD.
Fieldname	Predict fieldname.
Alias	ALIAS used in the MFD. This is the ADABAS name.
Usage	Usage format of the field in the MFD.
Actual	Actual format of the field as described in the MFD.
D	Descriptor flag. Value is 'D' for descriptors.
U	Unique flag. Value is 'U' if descriptor is unique.

S	Null suppression option. Value is 'N' for null suppression, 'F' for fixed format, or blank.,
Ty	Field type. Values are: GR-Group, PE-Periodic Group, MU-Multi-value field, SP-Superdescriptor, SB-Subdescriptor.
L	Level number. Values are from 1 to 7.

The following functions are available from the IXFLD Selection Screen via the PF keys:

### **Help (PF1)**

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

### **Restart (PF2)**

Press PF2 to cancel all previous selections and restart the master file generation process at the File Selection Screen.

### **Exit (PF3)**

Press PF3 to exit the file description generator and return to the main menu.

## **Skip (PF4)**

Press PF4 to skip this file as a descendant if it was selected in error. This does not affect previous or subsequent selections; however, the "skipped" segments appear in subsequent lists when the user is returned to the Child Selection Screen to define any additional paths in the hierarchy.

## **Scroll Backward (PF7)/Scroll Forward (PF8)**

Press PF7 to scroll the record list backward; press PF8 to scroll the record list forward.

## **The KEYFLD Selection Screen**

Once the IXFLD has been selected and the ENTER key has been pressed, the following screen is displayed. Subsequent user entries are in lower case.



Master: Master File Generation Facility for AUTOABDS  
 SAMPLE ==KEYFLD Selection==

File: SYSDIC-DB Dictionary:  
 Child: SYSDIC-FI Dictionary:  
 IXFLD: FILE\_REC Actual: A33

Select the common field (KEYFLD) from the parent file with 'S'

S	Fieldname	Alias	Usage	Actual	D	U	S	Ty	L
-	-----	-----	-----	-----	-----	-----	-----	-----	-----
	RECORD_TYPE	AA	A2	A2				N	1
	COMMENTS	BD	A30	A30				N MU	1
	DESC	AB	A1	A1				N	1
	MAINTENANCE_ACTION	BG	A1	A1				N	2
	KEY_WORD	KY	A32	A32	D			N MU	1
	OWNER_ID	II	A32	A32	D			N MU	1
	DATA_BASE_NAME	JJ	A32	A32				N	1
	SPMRF_TYPE	DA	A1	A1				N	1
s	FILE_ELEMENT	DC	A32	A32				N MU	1
	FILE_TYPE_EL	S7	A33	A33	D			N SP	1
	DATA_BASE_TYPE	AT	A1	A1	D			N	1
	GEN_FLAG	CI	A1	A1				N MU	2

PF1=Help PF2=Restart PF3=End PF4=Skip

PF7=Up PF8=Down

The KEYFLD Selection Screen is used to identify the primary key in the parent file. Select the common field in the host file by entering an 'S' in the column labeled 'S'. Press PF4 to skip this parent/child relationship if it was selected in error. This will not affect previous or subsequent selections. AUTOADBS displays only those fields that can be used as a KEYFLD on this screen. The KEYFLD must have the same format type as the IXFLD and field length may be the same or shorter than that of the IXFLD. It cannot be a PE group, a subdescriptor, a superdescriptor composed of partial fields, or a reformatted field. When using the old ADABAS interface, the KEYFLD cannot be an element of a PE group or an MU field if the actual format is zoned or packed. With the old interface, subdescriptors and superdescriptors composed of partial fields must have the same length as the IXFLD.

The KEYFLD Selection Screen displays the following information:

Master	Master file description name provided on the Main Menu.
File	The parent file from which the KEYFLD is being selected.
Dictionary	Predict dictionary where the parent file is described.
Child	The child file from which the IXFLD was selected.
Dictionary	Predict dictionary where this child file is described.
IXFLD	The selected IXFLD.

Actual	The ACTUAL format of the selected IXFLD.
S	Enter 'S' for the selected KEYFLD.
Fieldname	Predict fieldname.
Alias	ALIAS used in the MFD. This is the ADABAS name.
Usage	Usage format of the field in the MFD.
Actual	Actual format of the field as described in the MFD.
D	Descriptor flag. Value is 'D' for descriptors.
U	Unique flag. Value is 'U' if descriptor is unique.
S	Null suppression option. Value is 'N' for null suppression, 'F' for fixed format, or blank.
Ty	Field type. Values are: GR-Group, PE-Periodic Group, MU-Multi-value field, SP-Superdescriptor, SB-Subdescriptor.
L	Level number. Values are from 1 to 7.

The following functions are available from the KEYFLD Selection Screen via the PF keys:

## **Help (PF1)**

Press PF1 to display an extensive online help facility; much of the text of this document is available via this function.

## **Restart (PF2)**

Press PF2 to cancel all previous selections and restart the master file generation process at the File Selection Screen.

## **Exit (PF3)**

Press PF3 to exit the file description generator and return to the main menu.

## **Skip (PF4)**

Press PF4 to skip this file as a descendant if it was selected in error. This does not affect previous or subsequent selections; however, the "skipped" segments appear in subsequent lists when the user is returned to the Child Selection Screen to define any additional paths in the hierarchy.

## **Scroll Backward (PF7)/Scroll Forward (PF8)**

Press PF7 to scroll the record list backward; press PF8 to scroll the record list forward.

## **Background Execution**

The AUTOADBS FOCEXEC can be executed in the background to create a single file MFD by passing values normally provided on the menus as arguments.

## Syntax How to Create a Single File MFD in Background Execution

The syntax is:

```
EX AUTOADBS BATCH=Y,USER=userid,MASTER=master,FILE_NAME=filename,
              SEQFIELD=seqfield (,options)
```

The USER parameter is the default high level qualifier for output datasets (MVS only).

Options:

FNR=filenumber	File number	(default *)
CALLTYPE=r1/find (RL,FIND)	Calltype	(default RL)
PASSWORD=password	File Password	(default ' ')
DBID=nnn	Database number	(default 0)
DICT=suffix (custom)	Use PREDICT Dictionary	(custom)
REPLACE=yn (Y=Yes,N=No)	Replace existing description?	(default N)
COMMENTS=yn (Y=Yes,N=No)	Include Fieldname Comments?	(default Y)
HEADINGS=yn (Y=Yes,N=No)	Use NATURAL Column Headings?	(default N)
FOCDEF=yn (Y=Yes,N=No)	Create FOCDEF File?	(default N)
NEW=yn (Y=Yes,N=No)	Use NEW ADABAS Interface?	(default Y)
START=nn (1-32)	Start with Fieldname Position	(default 1)
LENGTH=nn (5-36)	For a Total Length of	(default 36)
DATEDISP=format	Date Display Format	(default YYMD)
REQ_SEQFIELD=NO	Use only when SEQFIELD is not known	
USER=userid (MVS only)	High level qualifier of output datasets (required if target dataset names not provided)	
MASTERDATA=dsn (MVS only)	Master Target PDS (default &USERID.MASTER.DATA)	
FOCADBSDATA=dsn (MVS only)	Access Target PDS (default &USERID.FOCADBS.DATA)	
FOCDEFDATA=dsn (MVS only)	FOCDEF Target PDS (default &USERID.FOCDEF.DATA)	

Options are provided on the command by line by entering the name-value pair and separating them with commas. Options may extend over several lines. User provided values and default values are echoed after the MFD is generated.

## Reference Usage Notes

- FNR may be used in place of FILE\_NAME. If FNR is used, FILE\_NAME is ignored. Whether FILE\_NAME or FNR is used, the selection must be unique within the Predict dictionary.
- DBID is only used to populate the DBID attribute in the Access File Description. It is not used for file selection.
- CALLTYPE will be set to RL if it is invalid, or if FIND is supplied and the installation requires CALLTYPE=RL.
- PASSWORD will be ignored if its length exceeds 8 characters.
- SEQFIELD will be ignored if the named field cannot be found in the dictionary for the selected file, or the field cannot be used as a SEQFIELD, or the installation does not require a SEQFIELD. Otherwise, the description will not be generated.

## The Generated Descriptions

The AUTOADBS facility creates complete Master File Descriptions and Access File Descriptions for the files selected by the user. The following describes the attributes and values assigned to them by AUTOADBS.

---

## File and Segment Attributes

FILENAME=	Master file name specified on the Main Menu.
SUFFIX=	Always ADBSIN.
SEGNAME=	<p>The segment names in the description generated by AUTOADBS follow a logical format to provide uniqueness within the file. For the root segment of the ADABAS file (ACCESS=ADBS), the segment name is 'Snn', where 'nn' is a two digit number that indicates the order that the file was selected. The first file (selected as the root from the file selection menu) has SEGNAME=S01. Subsequent files used in the same description (selected as children from the file selection menu) will have SEGNAME=S02, SEGNAME=S03, etc.</p> <p>The segment names for PE groups and MU fields have the format 'aammnn', where 'aa' is the ADABAS name of the field, 'mm' is a two digit number that indicates the order that the PE group or MU field appears in the PREDICT description of the file, and 'nn' is the order number used for the root segment. For example, SEGNAME=BE0201 would describe the segment for the field BE, which is the second (02) PE group or MU field described in the first segment (01).</p>

<b>SEGTYPE=</b>	For the root segment and cross-reference segments with non-unique IXFLDs, the segtype is 'S'; for cross-reference segments with unique IXFLDs, the segtype is 'U'. The segtype for all PE and MU fields is 'S'.
<b>PARENT=</b>	For dependent segments, this is the value of SEGNAME of the parent record.
<b>OCCURS=</b>	For PE group and MU field segments only, this will contain the ADABAS fieldname with the suffix 'C', which is the ALIAS of the counter field in the parent segment.
<b>MAX=</b>	For PE group and MU field segments only, this will indicate the maximum number of occurrences of the PE group MU field as described in the Predict dictionary. This attribute is a comment only and is not used by the interface.

## **Access File Attributes**

<b>RELEASE=</b>	Indicates the ADABAS Release and is always 5.
<b>OPEN=</b>	Determines if the interface should issue ADABAS OPEN and CLOSE calls for each report request. This is always YES.



SEGNAM=	The segment name as described in the MFD.
ACCESS=	Specifies the access method for the segment. ADBS indicates segments that contain non-repeating data. PE indicates segments that describe periodic groups. MU indicates segments that describe multi-value fields.
DBNO=	ADABAS database number. This attribute will not be included in the AFD if the Predict dictionary value is 0 and a value was not provided during the AUTOADBS execution.
FILENO=	ADABAS file number.
PASS=	ADABAS password for the file. This attribute is only included if a value was provided during the AUTOADBS execution.
CALLTYPE=	Determines whether FIND or RL calls are made to retrieve the data. Note: With ADABAS=OLD, CALLTYPE will be changed to RL for the child file in short-to-long JOINS and CALLTYPE will be changed to FIND when the IXFLD is a '.NOP' descriptor.
SEQFIELD=	The SEQFIELD selected by the user. This attribute will not be included if a SEQFIELD was not selected.
KEYFLD=	For child segments only, this is the common field in the parent segment used to relate two files.

IXFLD=	For child segments only, this is the common field in the child segment used to relate two files.
FIELD=	Describes a descriptor, superdescriptor, subdescriptor or a component of a superdescriptor. Used only with ADABAS=NEW.
TYPE=	Indicates the type of descriptor for FIELD. Values are SUP for super descriptors, NOP for subdescriptors or superdescriptors composed of partial fields, or DSC for a simple descriptor. Used only with ADABAS=NEW.
NU=	Indicates if the field uses null suppression. Values are YES or NO. Used only with ADABAS=NEW for elements of superdescriptors.

## Field Attributes

FIELD=	Fieldname from the Predict dictionary.
ALIAS=	Actual ADABAS name.
USAGE=	The FOCUS usage format of the field. This determines how the value is displayed in reports. Note: USAGE= is not included in the MFD but is inferred by its location in the FIELD description.

ACTUAL=	The field format as stored in the ADABAS file. Note: ACTUAL= is not included in the MFD but is inferred by its location in the FIELD description.
INDEX=I	Indicates that the field is a superdescriptor, subdescriptor or simple descriptor.
TITLE=	The column heading used in reports. This attribute is included only if the field has a column heading value in the Predict dictionary and the 'Include NATURAL Column Headings' was selected.
GROUP=	This attribute identifies fields described as simple groups or PE groups.
\$GROUP=	The field is a group field that is itself part of another group. Imbedded groups are not supported by the interface.
\$GRMU =	The field described is a group that contains a PE group or an MU field.
\$2LONG=	The field is a group field whose total length exceeds the maximum of 256 characters supported by FOCUS.
\$PEMU=	The field described is a PE group that contains an MU field or is a group field that contains a PE group or an MU field. In both cases, the field cannot be used to retrieve data using the interface.

\$PEDIF=	The field is a PE group described in a Userview where all of the component elements that are described in the master file are not included in the userview. As a result, the length of the PE group cannot be determined accurately.
\$NOMAS=	The field described in a Predict userview does not exist in the Master ADABAS file. Typically this will be a group field that combines several real fields from the master file in the userview. The ADABAS name associated with this field does not exist in the physical ADABAS file's FDT.
\$REDEF=	The field is a redefinition of an actual ADABAS field. The field itself does not exist in the ADABAS file and does not have an ADABAS name in the Predict dictionary.
\$PH=	The field is a phonetic descriptor. These are not supported by the interface.
\$HY=	The field is a hyper descriptor. These are not supported by the interface.
\$HM=	The field is a hyper descriptor used as an MU field. Hyper descriptors are not supported by the interface.
\$HP=	The field is a hyper descriptor used as a PE group. Hyper descriptors are not supported by the interface.

`$fieldname`

These entries, found at the end of the master, identify duplicate fields described in the master. It lists the fieldname as found in the MFD, the number of duplicate occurrences, the segments where the duplicate fields are located, and the full Predict fieldname.

## Fieldnames

Fieldnames are derived from the Predict dictionary and may be truncated based on user selections on the main menu. Hyphens in the Predict fieldname are converted to underscores. AUTOADBS will also create several fields in the generated description, both for ease of use and interface requirements. These include counter fields, order fields, superdescriptor elements, DEFINEd fields, and cross-reference fields.

Counter fields are generated for each PE group and MU field in the ADABAS file. This field is used to determine the number of occurrences of the PE group or MU field. It is located in the repeating group's parent segment and is referenced in the OCCURS attribute of the repeating group's segment. This is accomplished by appending a 'C' to the ADABAS name in the ALIAS attribute of the file description. You may refer to this field in report requests to print the number of occurrences of the repeating group. The FIELDNAME of the counter field has the format 'aammnn\_CNT', where 'aammnn' is the same as the segment name where the field is described. This field is located in the repeating group's parent segment.

Order fields are generated for each repeating group and are added at the end of the repeating group's segment. This field has an ALIAS of order and can be used to determine the sequence of a particular repeating field. The FIELDNAME of the order field has the format 'aammnn\_OCC', where 'aammnn' is the same as the segment name where the field is described.

AUTOADBS may also generate a field with the format 'aa\_NOP', where 'aa' is the ADABAS name of the field. This field is generated only when an imbedded group field is selected as a SEQFIELD, KEYFLD, or IXFLD, or, with ADABAS=OLD, the selected field has a zoned or packed actual format. This field is also included in the AFD for the appropriate attribute. Normally, you should not refer to this field in your report requests.

For superdescriptors, AUTOADBS will generate a unique fieldname for the element fields. A suffix in the format '\_Snn' will be appended to the element fieldname, where 'nn' is incremented by 1 for each superdescriptor in the ADABAS file. This allows AUTOADBS to list the element fields with minimal risk of creating duplicate fieldnames, particularly in cases where the element fields are used in several superdescriptors. Note that the element fields are also described in the MFD. If the element field appears in only one superdescriptor, you may edit the MFD to remove the original occurrence of the field and rename the superdescriptor entry. In the new ADABAS interface, perform any selection tests on the element field (with the '\_Snn' suffix) rather than the original field to take advantage of implicit use of the superdescriptor inverted list.

AUTOADBS generates an MFD based DEFINE for Natural fields described as dates. This allows FOCUS to convert the internal ADABAS date value to a FOCUS smart date. This DEFINEd fieldname has the format 'fieldname\_DAT', where fieldname is the name from the Predict dictionary.

## Aliases

The ALIAS generated by AUTOADBS will typically be the ADABAS fieldname. In certain situations, however, the ALIAS will be altered. This includes fields generated as counters, superdescriptor elements that are PE groups or MU fields, fields with ADABAS formats not directly supported in FOCUS, and fields that have different formats in several userviews. In the simplest case, counter fields for PE groups and MU fields will be generated where the alias is the ADABAS name with the letter 'C' appended.

For superdescriptor elements that are PE groups or MU fields, the alias will consist of the ADABAS name with the digit '1' appended. This ADABAS syntax indicates that the first field of the repeating group be retrieved, rather than all values in the group. Normally, you should not reference these fields in a report request. Instead, use the original field.

## **Syntax**      **How to Select Against a Superdescriptor that has both Alphanumeric and Numeric Elements**

This syntax is provided solely to allow record selection against a superdescriptor that has both alphanumeric and numeric elements. The syntax for selecting against these fields is

```
WHERE super EQ aaa/nnn/aaa
```

where 'aaa' represents the alphanumeric portion of the group and 'nnn' represents the numeric portion. You should also not print the superdescriptor, as the interface assembles the value from the element fields (superdescriptor values are not retrievable from ADABAS).

Fields with ADABAS formats not directly supported in FOCUS and fields that have different formats in several userviews will be reformatted by ADABAS before being presented to FOCUS. This is accomplished by adding ADABAS formatting options to the alias in the format 'aa,l,t', where 'aa' is the ADABAS name of the field, 'l' is the length to be used, and 't' is the ADABAS format.

## Usage and Actual Formats

The following table illustrates how the ADABAS format and length is converted to the FOCUS USAGE and ACTUAL formats by AUTOADBS. In cases where the ADABAS field must be reformatted to a supported FOCUS format, the ALIAS is provided to show how this is accomplished.



Predict Format	Predict Length	FOCUS USAGE	FOCUS ACTUAL	FOCUS ALIAS	Comments
A	n	An	An	aa	
B	n	An	An	aa	(Super and subdescriptors)
B	1	I4	I1	aa	
B	2	I6	I2	aa	
B	3	I8	I4	aa,4,B	
B	4	I9	I4	aa	
B	5	P12	P7	aa,7,P	
B	6	P15	P8	aa,8,P	
B	>6	An	An	aa,n,A	where n=(length * 2) + 2 defined date field created
D		P7	P4	aa	
L		I1	I1	aa	
T		P13	P7	aa	
F	4	F9.2E	F4	aa	
F	8	D15.2E	D8	aa	
I	1	I4	I1	aa	
I	2	I6	I2	aa	
I	4	I9	I4	aa	
I	8	A18	A18	aa,20,A	
N	n	Pn	Zn	aa	
U	n	Pn	Zn	aa	
P	n	Pn	Pk	aa	k=(n+2)/2
N	n.m	Pj.m	Zk	aa	j=n+m+1, k=n+m
U	n.m	Pj.m	Zk	aa	j=n+m+1, k=n+m
P	n.m	Pj.m	Pk	aa	j=n+m+1, k=(n+m+2)/2
NS	n	Pj	Zk	aa	j=n+m+1, k=n+m
US	n	Pj	Zk	aa	j=n+m+1, k=n+m
PS	n	Pj	Pk	aa	j=n+m+1, k=(n+m+2)/2
NS	n.m	Pj.m	Zk	aa	j=n+m+2, k=n+m
US	n.m	Pj.m	Zk	aa	j=n+m+2, k=n+m
PS	n.m	Pj.m	Pk	aa	j=n+m+2, k=(n+m+2)/2

## Differences Between ADABAS=OLD and NEW

When the Main Menu option 'Use NEW ADABAS Interface?' is selected (to be used with ADABAS=NEW), superdescriptors and subdescriptors will be identified with the TYPE= attribute in the AFD, rather than with the '.SPR' or '.NOP' required in the old interface. Also, all superdescriptor elements will be described in the AFD with the NU= attribute. When the Main Menu option 'Use New ADABAS Interface?' is not selected, several restrictions are imposed. Fieldnames generated in the MFD are limited to a maximum of 12 characters. A field that has a zoned or packed format and is an element of a PE group or is an MU field cannot be used as a KEYFLD. A KEYFLD or IXFLD that has a zoned or packed format will also be described as alphanumeric with a fieldname that has a suffix of '\_NOP'. A KEYFLD with the .NOP suffix must be the same length as the IXFLD; it cannot be shorter. CALLTYPE will be changed to RL for the child file in short-to-long JOINS and CALLTYPE will be changed to FIND when the IXFLD is a '.NOP' descriptor, regardless of what the user provided in AUTOADBS.

## Changes to the Generated Descriptions

In general, you will not need to change the descriptions generated by the AUTOADBS facility. However, you may need to edit the generated MFD and/or its corresponding AFD if:

- Duplicate fieldnames exist.

- The DBID or FILENO included in the Predict dictionary is different than the actual ADABAS file.
- You prefer to use different fieldnames than those in the Predict dictionary.
- You wish to add edit display options to fields.
- You wish to increase the display length of numeric fields.
- You wish to change OPEN=YES to OPEN=NO in the AFD.
- You wish to replace AUTOADBS generated superdescriptor elements that occur only once in the file with the original field.
- You wish to add MFD based DEFINES to describe redefined fields described in the Predict dictionary.
- You must remove segments because the 64 segment FOCUS limitation has been exceeded. This step will be necessary when AUTOADBS generates the completion message: DESCRIPTION GENERATED - nnn SEGMENTS DESCRIBED. Typically, you will remove PE groups or MU field segments that are not required. Alternatively, you may remove the segment and add the repeating field to the non-repeating segment, once for each anticipated occurrence of the field. For example, if you anticipate 3 occurrences, you will add three fields to the MFD (with unique fieldnames). The ALIAS will have the format 'aan', where 'aa' is the ADABAS name of the field and 'n' is the relative entry number from 1 to 3. MU fields that are part of a PE group are described in the non-repeating segment with an ALIAS in the format 'aan(m)', where 'aa' is the ADABAS name of the field, 'n' is the relative entry number of the PE group and 'm' is the relative entry number of the MU field within the PE group.

## Creating a Record of Masters Generated

The AUTOADBS facility maintains a temporary list of the Master File Descriptions generated during any one session. This list is refreshed at the beginning of each session and is normally erased at the end of the session. You may retain this list by executing AUTOADBS with the following syntax:

### Syntax      How to Generate a List of Master File Descriptions

EX AUTOADBS MFDLIST=Y

In MVS, the list will reside in a temporary dataset allocated to ddname AUTOADBL, with a disposition of MOD and record length of 114. In CMS, the list will reside in the file AUTOADBL FOCTEMP with a record length of 26. It is the user's responsibility to free or erase this file when no longer required. The file layout is:

Length	Columns	Description
8	1 - 8	Master Filename
8	9 - 16	Number of duplicate fieldnames
5	17 - 21	Number of segments described
5	22 - 26	Blank
44	27 - 70	Master Target PDS name (MVS only)

## Search Order for Parameter Log File (MVS only)

The AUTOADBS parameter log file in MVS will be located in one of three ways, in the following order:

### 1. DDNAME ADBS\$PRM

If this ddname is allocated to a PDS prior to execution of AUTOADBS, it will be used. The member ADBS\$PRM will be used or created for the user.

If this ddname is allocated to a sequential dataset, it will be freed, a message generated, and parameter logging will be disabled.

AUTOADBS does not free this ddname upon exiting, assuming that it may be used again, even if it was allocated by AUTOADBS with one of the following two methods.

### 2. Dataset name userid.FOCADBS.DATA

This is the default name provided in the code as &DSNP0. If that default has been changed during the interface installation, then that dataset name will be used.

This dataset must be a PDS. If not, a message is displayed and parameter logging is disabled.

### 3. DDNAME FOCADBS

The first dataset allocated to ddname FOCADBS will be used as the parameter file if the first two methods fail.

If ddname FOCADBS is not allocated, a message is displayed and AUTOADBS exits. AUTOADBS cannot be executed if there is no FOCADBS that describes the Predict dictionary files.

Method number two assumes standard IBI naming conventions. Method number three assumes that a user's dataset is allocated first in the concatenation of datasets to ddname FOCADBS. The first method allows the user to identify the profile dataset prior to execution of AUTOADBS. This option is recommended for sites that have non-standard dataset naming conventions.

## NF438: MODEL 204 Interface Enhancements

The new *FOCUS for IBM Mainframe MODEL 204 Interface Users Manual and Installation Guide (DN1000945.0594)* is available for FOCUS Release 6.8.

The major new features in MODEL 204 are available beginning in FOCUS Release 6.8 Put Level 9404, and are as follows:

- Improved COUNT performance.  
The Interface has been enhanced to more efficiently process COUNT requests against MODEL 204 fields.
- The ACCDATA attribute.  
This attribute is available for an Access File field declaration, and instructs the Interface to generate either character string or numeric HLI selection tests for MODEL 204 fields that do not have a MODEL 204 key specification.
- New Master and Access File Descriptions.  
Existing Master and Access File Descriptions have been cleaned up, and new files have been added. These are available on the tape that is supplied with the Interface.

Consult the new manual for in-depth documentation of these new features.

## NF440: Improved Page Handling - SET TRACKIO

The latest FOCUS MVS releases offer improved physical page handling, resulting in significant reductions in I/O requirements and in elapsed time for FOCUS files. In essence, MVS FOCUS now gathers more pages to fill a track before reading or writing the pages to disk.

A new command is provided for turning this feature on and off

```
SET TRACKIO = {OFF / ON}
```

where:

**ON** Causes FOCUS to fill a track before reading or writing to disk.  
ON is the default.

**OFF** Uses the old method for track I/O.

You can determine if TRACKIO is ON or OFF by issuing ? SET at the FOCUS prompt.

Performance enhancements can be measured by comparing requests with TRACKIO ON versus OFF. The savings realized are in the form of reduced EXCP counts.



## NF441: External Sort

Impressive performance achieved with the HiperFOCUS External Sort feature led us to make external sort products available to all FOCUS users in Release 7.0. When generating a large report you can now invoke an external sort, such as DFSORT, SyncSort, etc., to improve processing efficiency. The actual impact varies from case to case, but the feature has significantly decreased CPU time in situations where there were many lines of report output. External sorts can be called for any TABLE, EMR, MATCH or GRAPH request.

### Syntax      How to Enable External Sorts with the SET EXTSORT Command

External sorts are enabled by the SET EXTSORT command. The syntax is

```
SET EXTSORT = {ON/OFF}
```

where:

- ON**              Enables FOCUS to exploit an external product for sorting reports. This is the default.
- OFF**             Selects the FOCUS internal sort procedure for sorting reports.

To determine the sort product used, issue ? STAT after the report request and examine the value returned in the SORT USED parameter:

- FOCUS**              The internal FOCUS sort was used.

SQL	RDBMS sort facilities (accessed by FOCUS) were used to sort the report.
EXTERNAL	An external product was used to sort the report. To confirm the extent of external sorting used, issue ? STAT. The INTERNAL MATRIX CREATED parameter will display NO if an external sort handled the report and YES if FOCUS handled the first group of records and an external sort handled the remainder.
NONE	The report did not require sorting.

## Requirements

The following sort products are supported:

- MVS - DFSORT, SyncSort, PLSORT
- CMS - DFSORT, SyncSort, VMSORT (must be 31-bit addressable).

### CMS External Sort Requirements

In CMS, issue a GLOBAL TXTLIB command to identify the location of the sort software. If FOCUS cannot locate the sort, it will issue a GLOBAL TXTLIB SORTLIB command. If FOCUS still cannot locate the sort software, it terminates with the following error message:

```
(FOC909) CRITICAL ERROR IN EXTERNAL SORT. RETURN CODE IS: 16
```

You will also receive an error message in CMS if your system sort is DFSORT/CMS and your temp disk is not large enough for your sort requirements. With DFSORT/CMS, the maximum temp disk that can be obtained for a 3380 device is 16 cylinders. If this is not enough for your larger sort requests, modification to the DUFMAC MACRO may be required to allow more space for sort work files. Consult the *IBM DFSORT/CMS* manual for your release of DFSORT. FOCUS always respects your sort product's installation parameters.

## Displaying External Sort Messages

By default, FOCUS does not display messages created by external sort products. However, it is possible to display them for diagnostic purposes. See the instructions below for displaying DFSORT and SyncSort messages under MVS.

To display DFSORT messages:

1. Create a sequential file with these attributes:

```
DCB=(LRECL=80,RECFM=F,BLKSIZE=80)
```

2. Create the following record beginning in column 2:

```
OPTION MSGPRT=ALL, MSGDDN=trace_name
```

where trace\_name is any user-defined ddname.

3. Allocate the file to ddname DFSPARM.
4. Allocate the ddnames trace\_name and SORTDIAG to a single file or to SYSOUT. This ensures that all trace and diagnostic messages are written to the same file.

To display SyncSort messages:

1. Create a file with these attributes:

```
DCB=(LRECL=80,RECFM=F,BLKSIZE=80)
```

2. Create the following record beginning in column 2:

```
MSG=AB,BMSG,MSGDD=trace_name,LIST
```

where trace\_name is any user-defined ddname.

3. Allocate the file to ddname \$ORTPARM.
4. Allocate the ddname trace\_name to a file or to SYSOUT.

## AUTOTABLEF

The FOCUS reporting facility derives much of its power and flexibility from different stages of internal processing. However, if you have a simple report request, it may not require the full complement of processing capabilities. You can optimize such requests, granting FOCUS permission to avoid unnecessary work and save resources, by using the SET AUTOTABLEF command in conjunction with the AUTOTABLEF facility.

When AUTOTABLEF is set ON and you issue a report request, FOCUS determines if your report requires the temporary FOCUS work area called the internal matrix. If not, FOCUS does not create this matrix, potentially saving significant time and system resources.

AUTOTABLEF can optimize all TABLE, EMR, and GRAPH requests except those that include the following syntax:

- ACROSS
- FOR

- PCT.
- PCT.CNT.
- TOT.
- Within
- Multiple display commands (PRINT, LIST, SUM, COUNT)
- Multiple display paths (display fields from multiple paths)

## Syntax      How to Turn Optimization On and Off using the SET AUTOTABLEF Command

You can turn optimization on and off using the SET AUTOTABLEF command.

```
SET AUTOTABLEF = {ON / OFF}
```

where:

- |     |  |
|-----|--|
| ON  | Switches report optimization on, creating no internal matrix unless required.  |
| OFF | Switches report optimization off. This is the default. Disabling optimization guarantees the creation of an internal matrix. This ensures your ability to redisplay or extract a report later in your FOCUS session. |

You can confirm whether or not FOCUS created an internal matrix for your most recent report by issuing the ? STAT command. The command's INTERNAL MATRIX CREATED parameter will be set to YES or NO respectively.

## NF443: Large Packed Fields

Users who need to use either very large numbers or numbers with a great deal of precision may declare packed numbers to a maximum of 31 digits.

The Large Packed Fields feature provides support for 16-byte packed data. Previous FOCUS releases only supported packed fields of up to 8-bytes in length. You declare packed fields in Master File Descriptions, DEFINES, and COMPUTEs as follows:

Actual	$P_n$	Packed decimal where n is a number from 1 to 16 bytes.
Format or Usage	$P_{m.n}$	Where m is up to 33 positions (which includes a position for the sign and decimal place), and n can be up to 31 digits.

This increase in size allows for zoned format fields to be described with an ACTUAL of up to Z31.

The behavior of packed fields in numeric computations does not change from prior releases. If the result of an expression cannot fit into an 8 or 16 byte packed field, a 0 is stored. Conversely, if the result fits into 8 or 16 bytes but the display usage or format is not large enough, asterisks are displayed.

### Reference Special Considerations

- Date formats can be used with packed numbers declared with an ACTUAL of less than 8.

- The ASQ. prefix is not valid for a packed field of any length.

## Reference Error Messages

(FOC1847)            **DECIMAL OVERFLOW EXCEPTION**

An attempt to perform a decimal arithmetic operation has occurred, and has surpassed the hardware magnitude limit. Error cause is indeterminate, and is possibly due to input of binary data which bypasses the normal character checking operation

(FOC1848)            **DECIMAL DIVIDE EXCEPTION**

An attempt to perform a decimal division has occurred, and has diverged from the requirements of such division. Error cause is indeterminate, and is possibly due to input of binary data which bypasses the normal character checking operation.

(FOC1852)            **OPTION ASQ. NOT ALLOWED FOR PACKED FIELDS:**

Squaring a packed field may often exceed hardware limitations. Therefore, the ASQ option may not be applied to packed fields.

FOCUS will terminate when the FOC1847 and FOC1848 errors occur more than 10 times in a FOCUS session.

## NF444: ASIS Function

Dialogue Manager now distinguishes between a blank and a zero. Using the ASIS function, numeric string constants and variables defined as numeric strings (numerics within single quotes) can be differentiated from fields defined simply as numbers. The ASIS function forces FOCUS to evaluate a variable as entered rather than converting it into a number. The ASIS function is used in equality expressions only.

The examples below show how the ASIS function effects the way FOCUS recognizes values. In the first example, with no ASIS function, FOCUS does not distinguish between variables defined as '123' and 123. The second example shows the use of the ASIS function to distinguish between the two variables.

### Example Distinguishing Variables Without the ASIS Function

The FOCEXEC is:

```
-SET &VAR1 = '123';  
-SET &VAR2 = 123;  
-IF &VAR2 EQ &VAR1 GOTO ONE;  
-TYPE VAR1 &VAR1 EQ VAR2 &VAR2 NOT TRUE  
-QUIT  
-ONE  
-TYPE VAR1 &VAR1 EQ VAR2 &VAR2 TRUE
```

The output is:

```
VAR1 123 EQ VAR2 123 TRUE
```



## Example Distinguishing Variables With the ASIS Function

The FOCEXEC is:

```
-SET &VAR1 = '123';  
-SET &VAR2 = 123;  
-IF &VAR2 EQ ASIS(&VAR1) GOTO ONE;  
-TYPE VAR1 &VAR1 EQ VAR2 &VAR2 NOT TRUE  
-QUIT  
-ONE  
-TYPE VAR1 &VAR1 EQ VAR2 &VAR2 TRUE
```

The output is:

```
VAR1 123 EQ VAR2 123 NOT TRUE
```

## NF446: Using StyleSheets

Many of the standard features for producing high-quality, printed reports are available directly through FOCUS StyleSheets. These features include multiple fonts, sizes, styles, and colors, and control over page orientation, column placement, column widths, margins and paper size. With these features, you will easily generate visually interesting reports that accentuate key information.

StyleSheets enable you to produce attractive reports that highlight key information. With StyleSheets, you can style report components individually. For example, you can make a column title 18 point Times, make data values in the same column 16 point Helvetica and use different colors for values that are above or below specified conditions, and italicize the column total. You can also move that column anywhere in the report. In addition, you can control page parameters, such as margins, page size, and orientation. StyleSheets are fully documented in the:

FOCUS for MAINFRAME Users Manual Release 7.0

## NF448: FOCPARM Enhancements

The FOCPARM file is used to implement site-wide configuration parameters and is required for FOCUS Release 7.0. FOCPARM is executed prior to the PROFILE FOCEXEC and can only include valid FOCUS SET commands.

In MVS, it is a member of the ERRORS.DATA PDS. In VM it is FOCPARM ERRORS \*. If the FOCPARM file is renamed or cannot be found by FOCUS, the following error message is generated:

**ERROR READING FOCPARM INSTALL FILE.**

The sample FOCPARM file that is provided on the FOCUS tape is shown below:

```
-----  
-*  
-* THIS SECTION CONSISTS SOLELY OF SET COMMANDS, USED TO CUSTOMIZE  
-* THE BEHAVIOR OF FOCUS AT YOUR SITE. PLEASE NOTE THAT ONLY SET  
-* COMMANDS ARE SUPPORTED, AND THAT ANY OTHER FOCUS COMMANDS ARE  
-* NOT PERMITTED AND WILL FORCE YOU OUT OF FOCUS. THIS MEMBER MAY  
-* NOT BE USED AS A PROFILE EXCEPT FOR SET COMMANDS.  
-----  
SET EMPTYREPORT=OFF  
SET BLKCALC=NEW  
SET FIELDNAME=NEW  
SET QUALCH= .  
SET QUALTITL=OFF  
SET HOLDSTAT=OFF  
SET HOTMENU=OFF  
SET IMMEDIATE=OFF  
SET AUTOPATH=ON  
SET AUTOINDEX=ON
```

There are two other FOCPARM files: FOCPARMA and FOCPARMC, where FOCPARMA is for aggressive and FOCPARMC is for conservative settings. The aggressive settings are those that exploit new features, such as BLKCALC, AUTOPATH and AUTOINDEX. FOCPARMC does not exploit new features and behavior, and is generally more conservative than both FOCPARM and FOCPARMA. These files are members of the ERRORS.DATA PDS in MVS, and have a filetype of ERRORS on VM. You can use either member by changing FOCPARM to a different name, then renaming FOCPARMA or FOCPARMC to FOCPARM.

FOCPARMA contains:

```
-----*
-* THIS SECTION CONSISTS SOLELY OF SET COMMANDS, USED TO CUSTOMIZE *
-* THE BEHAVIOR OF FOCUS AT YOUR SITE. PLEASE NOTE THAT ONLY SET *
-* COMMANDS ARE SUPPORTED, AND THAT ANY OTHER FOCUS COMMANDS ARE *
-* NOT PERMITTED AND WILL FORCE YOU OUT OF FOCUS. THIS MEMBER MAY *
-* NOT BE USED AS A PROFILE EXCEPT FOR SET COMMANDS. *
-----*

SET EMPTYREPORT=OFF
SET BLKCALC=NEW
SET FIELDNAME=NEW
SET QUALCH=.
SET QUALTITL=ON
SET HOLDSTAT=ON
SET HOTMENU=ON
SET IMMEDIATE=ON
SET CACHE=256
SET AUTOTABLEF=ON
SET AUTOPATH=ON
SET AUTOINDEX=ON
```

FOCPARMC contains:

```
-----*
-* THIS SECTION CONSISTS SOLELY OF SET COMMANDS, USED TO CUSTOMIZE *
-* THE BEHAVIOR OF FOCUS AT YOUR SITE. PLEASE NOTE THAT ONLY SET *
-* COMMANDS ARE SUPPORTED, AND THAT ANY OTHER FOCUS COMMANDS ARE *
-* NOT PERMITTED AND WILL FORCE YOU OUT OF FOCUS. THIS MEMBER MAY *
-* NOT BE USED AS A PROFILE EXCEPT FOR SET COMMANDS. *
-----*
```

```
SET EMPTYREPORT=ON
SET BLKCALC=OLD
SET FIELDNAME=OLD
SET HOLDSTAT=OFF
SET HOTMENU=OFF
SET IMMEDIATE=OFF
SET AUTOPATH=OFF
SET AUTOINDEX=OFF
```

An assembly language file called FOCPARM has also been provided in MVS, which enables your site to change startup default options, including some which may not be specified in SET commands. Instructions are provided below for changing the following FOCUS startup options:

- Displaying the FOCUS banner (BANNER).
- Clearing the screen (CLRSCRN).
- National Language Support Defaults (LANG).
- Selecting an alternative PROFILE or no PROFILE (PROFOPT).
- Continental Decimal Notation (CDN).

You can change the options by replacing old options with new ones, and then assembling and linking the options into FOCUS. The default values in the sample FOCPARM provided with FOCUS are shown below. A value of '1' enables an option; a value of '0' disables an option.

```

*****
* This file is used to change startup options
*
FOCPARM  CSECT
BANNER   DC      F'1'          1. Indicates display banner
CLRSCRN  DC      F'1'          2. Indicates clear screen at startup
*
* NATIONAL LANGUAGE SUPPORT DEFAULTS
* DEFEBE and DEFASC are used to fix the interpretation of client
* codepage indicators coming from OLDSTYLE (EDALINK 1.1) connections
*
LANG      DC      F'1'          3. Indicates Language Number (INTLCM)
JTERM    DC      CL8'IBM3270'  4,5.Indicates J-Terminal (INTLCM)
CODEPG   DC      F'00037'     6. Mainframe CodePage (INTLCM)
DEFEBE   DC      F'00037'     7. Default EBCDIC CodePage (INTLCM)
DEFASC   DC      F'00437'     8. Default ASCII CodePage (INTLCM)
*
PROFOPT  DC      F'0'          9. NOPROF/PROFILE 0=not allowed,1=allow*
CDN      DC      F'0'          10. CDN SETTING 0=OFF 1=ON
XDBSC    DC      CL4'@'       11. CHARACTER FOR EXT. DB SECUR. (7975)
*-----*
* A value of '1' enables an option, a value of '0' disables an option
* Other values are defaults for the option.
*
* DO NOT ADD, DELETE, OR CHANGE ORDER OF THE OPTIONS ABOVE!
*
* DO NOT ALTER THE CODE BELOW IN ANY WAY!
*****
.
.
.

```

**Note:** Since each installation of FOCUS generates its own version of the FOCPARM options, the procedures outlined below must be performed each time a new version or PUT Level of FOCUS is installed.

- Edit member FOCPARM of FOCCTL.DATA to revise the options as needed, and save the member back to dataset FOCCTL.DATA.
- Assemble the member FOCPARM from FOCCTL.DATA. The following is sample JCL for this:

```
//ASSEM      EXEC  PGM=IFOX00 ,PARM=' LIST,DECK,OBJ,NOALIGN'
//SYSLIB    DD   DISP=SHR,DSN=SYS1.MACLIB
//SYSUT1    DD   UNIT=SYSDA,SPACE=(TRK,(30,10))
//SYSUT2    DD   UNIT=SYSDA,SPACE=(TRK,(30,10))
//SYSUT3    DD   UNIT=SYSDA,SPACE=(TRK,(30,10))
//SYSPRINT  DD   SYSOUT=*
//SYSPUNCH  DD   DUMMY
//SYSGO     DD   DISP=SHR,DSN=prefix.OBJ(FOCPARM)
//SYSIN     DD   DISP=SHR,DSN=prefix.FOCCTL.DATA(FOCPARM)
```

- Link this new program into FOCUS. The following is sample JCL for this:

```
//LINKFOC   EXEC  PGM=IEWL,PARM='LET,NCAL,SIZE=1024K'
//SYSPRINT  DD   SYSOUT=*
//SYSUT1    DD   UNIT=SYSDA,SPACE=(CYL,(10,1))
//SYSLMOD   DD   DISP=OLD,DSN=prefix.FOCLIB.LOAD    <- FOCUS
//MAINTAIN  DD   DISP=SHR,DSN=prefix.FOCCTL.DATA    <- FOCCTL
//OBJECT    DD   DISP=SHR,DSN=prefix.OBJ           <- Assembled code
//SYSLIN    DD   *
              INCLUDE OBJECT(FOCPARM) <----- Assembled FOCPARM
              INCLUDE SYSLMOD(FOCINI)  <----- Module to be Changed
              INCLUDE MAINTAIN(FOCINI) <----- Linkedit Control Statements
              NAME      FOCINI(R)       <----- New Module
```

/\*

## NF449: Increased Number of Indices

In FOCUS Release 7.0 the combined number of real segments, plus indices, plus text segments has been increased to 189. The number of real segments alone can be up to 64.



## NF452: Capturing SET Parameter Values

In FOCUS Release 7.0, you can capture parameter values in amper variables.

### **Syntax**      How to Capture SET Parameter Values in Amper Variables

The syntax is

```
-? SET parameter ampervar
```

where:

`parameter`            Is any setting that may be queried with the ? SET ALL command.

`ampervar`              Is the name of the variable where the value is to be stored..

### **Example**      Capturing Parameter Values in Amper Variables

For example, if you enter

```
-? SET ASNAMES &abc
```

the value of &abc becomes the value of ASNAMES. If you omit &abc from the syntax, a variable called &ASNAMES is created containing the value of ASNAMES.

## NF455: FOCUS File Date and Time Stamp

FOCUS applies the date and time stamp to a FOCUS file each time it changes. In prior releases of FOCUS this information was updated on a per page basis.

### Date/Time Stamp

FOCUS updates the DATE/TIME stamp whenever the FOCUS file is changed via SCAN, FSCAN, CREATE, REBUILD, HLI, MAINTAIN, or MODIFY commands. The ? FDT command shows the current date and time stamp.

### Example Showing the Current Date and Time Stamp Using the ? FDT Command

Following is an example of the output from the ? FDT command:

```
> ? FDT CAR
DIRECTORY:CAR      FOCUS      A ON 05/09/94 AT 15.34.12
DATE/TIME OF LAST CHANGE: 04/06/94 11.04.46

  SEGNAME    LENGTH    PARENT    START    END    PAGES    LINKS    TYPE
1  ORIGIN      5             1         1         1         2
2  COMP        9             1         2         1         5
3  CARREC      9             2         3         1         3
4  BODY       12            3         4         1         3
5  SPECS     21             4         5         1         2
6  WARRANT    12            2         6         1         2
7  EQUIP     12             2         7         1         2
   COUNTRY      8             8         8         1         128 NEW
```

## Changes to the REBUILD command

All FOCUS files not updated within Release 7.0 may have a DATE/TIME stamp that does not reflect the last time the file changed but only the last time the first page changed. Use the REBUILD command to update the DATE/TIME stamp. .

### Example Using the REBUILD Command to Update the Date and Time Stamp

In the following example, the new REBUILD prompts are:

```
rebuild
```

```
ENTER OPTION (REBUILD,REORG,INDEX,EXTERNAL INDEX,CHECK OR TIMESTAMP)
```

```
timestamp
```

```
ENTER NAME OF FOCUS FILE (FN FT FM)=
```

```
car focus a
```

```
ENTER OPTION: TODAY'S DATE(T), SEARCH FILE FOR DATE(D) OR MMDDYY HHMMSS
```

```
123194 093025
```

where:

TIMESTAMP

This is the keyword to update the DATE/TIME stamp in the file.

TODAY'S DATE (T)

Updates the DATE/TIME stamp with the current date and time.

SEARCH FILE FOR  
DATE(D)

Updates the DATE/TIME stamp with the last time the file actually changed. FOCUS scans each page of the file. The most recent date and time recorded for a page is applied to the file. This option is synonymous with the ? FILE query, and can be time consuming when the file is very large.

MMDDYY HHMMSS

Enter the DATE/TIME stamp you wish to apply to the file. MMDDYY must be in the format mmddy and HHMMSS must be in the format hhmss. There must be a space between these two entries.

**Note:** The following error message is displayed when you respond with an invalid date or time:

(FOC961) INVALID DATE INPUT IN REBUILD TIME:

## NF456: VSAM Data and Index Buffers

Two new SET commands make it possible to establish DATA and INDEX buffers for processing VSAM files online.

The AMP sub-parameters BUFND and BUFNI allow MVS BATCH users to enhance the I/O efficiency of TABLE, TABLEF, MODIFY, and JOIN against VSAM files by holding frequently used VSAM Control Intervals in memory, rather than on physical DASD. By reducing the number of physical Input/Output operations, job throughput is improved. The new SET commands allow FOCUS users (in CMS, MVS/TSO, and MSO) to realize similar performance gains in interactive sessions. In general, BUFND (data buffers) increase the efficiency of physical sequential reads, whereas BUFNI (index buffers) are most beneficial in JOIN or KEYED access operations.

### **Syntax**      How to Set DATA Buffers for Processing VSAM Files Online

The syntax is

```
{MVS|CMS} VSAM SET BUFND n
```

### **Syntax**      How to Set INDEX Buffers for Processing VSAM Files Online

The syntax is

```
{MVS|CMS} VSAM SET BUFNI n
```

where:

`n` Is the number of data or index buffers.

The default values are BUFND=8, BUFNI=1 (eight data buffers and one index buffer).

To determine how many buffers are in effect at any time, issue the query

```
{MVS|CMS} VSAM SET ?
```

which returns the following output:

```
(FOC1177) SET OPTIONS - : BUFND = n / BUFNI = n / AMODE = n
```

VSAM data buffers can be set online in earlier releases of FOCUS (beginning with FOCUS 6.8/9304) by issuing the command:

```
{MVS|VSAM} SET BUFNO n
```

This syntax is still supported in Release 7.0 and is the equivalent of:

```
{MVS|VSAM} SET BUFND n
```

## NF457: The IBI MVS Subsystem

The IBI Subsystem is a state-of-the-art facility which provides communication between and coordination among address spaces running Information Builders products on MVS systems.

The IBI Subsystem can replace the IBI SVC, providing additional functionality, easier installation and maintenance, and enhanced ease of use.

By using the IBI Subsystem, installation of FOCUS and MSO will be faster and simplified. Performance and capacity have been improved, and error recovery capability has been enhanced. The Multi-Session Option (MSO), Simultaneous Usage (SU), SmartMode, Application Control Environment (ACE), and IMS BMP all use the IBI Subsystem for communications purposes. The HiperBudget feature of HiperFOCUS utilizes the IBI Subsystem to regulate use of expanded storage on a system-wide basis.

For additional information about the Subsystem, see the *FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0*.

## NF460: Changing the Default CALLTYPE

The new ADABAS Interface provides improved optimization by generating ADABAS FIND (S1) calls even when non-descriptor fields are used as search criteria. This occurs whenever CALLTYPE= FIND is specified in the FOCADBS Access File and you include an IF or WHERE test, referencing a non-descriptor field in your TABLE or TABLEF request.

In most cases, this will prove more efficient than generating a READ PHYSICAL (L2) ADABAS CALL, although in cases where large amounts of data exist, it may prove more efficient to alter your retrieval strategy and perform a READ PHYSICAL call. A SET command is provided for changing the default ADABAS call when selecting non-descriptor fields.

### Syntax      How to Change the Default CALLTYPE

```
{MVS|CMS} ADBSINX SET NDFIND [ON|OFF]
```

where:

- ON** Causes a FIND to be issued when selection is performed against a non-descriptor field. ON is the default.
- OFF** A READ PHYSICAL is generated when selection is performed against a non-descriptor field.

**Note:** This command only applies if CALLTYPE= FIND is specified in the FOCADBS Access File. If CALLTYPE= RL is specified, then a READ PHYSICAL occurs when selection is performed on a non-descriptor field.



## NF465: Changes to the Catalog Search in FOCUS

FOCUS no longer performs a catalog search in order to dynamically allocate FOCUS files with the naming convention 'prefix.mfd\_name.FOCUS', where mfd\_name is the name of the FOCUS Master File Description. You must therefore explicitly allocate all FOCUS databases. This behavior change affects FOCUS files only, not Master File Descriptions and FOCEXECs.

## NF466: Controlling IMS Access via DBCTL

IMS versions 3.1 and above have a new DBCTL (Database Control) option for controlling access to IMS databases in MVS. DBCTL provides a number of security, efficiency, and application-control enhancements and is easily implemented. Information Builders IMS Interface supports this new IBM subsystem and detailed instructions for implementing DBCTL and examples of its use appear in Technical Memo 7909.

The following describes the advantages of using DBCTL and tells what it means in terms of added security.

### Advantages of DBCTL

The advantages of accessing IMS through DBCTL include:

- Improved efficiency by directly linking user applications and IMS. Each task establishes its own connection to IMS by loading and calling the DBCTL subsystem when needed; DBCTL then issues DLI calls directly to IMS. Since this eliminates the BMP extension, no additional address space or communications dataset is required.
- Enhanced security via access to standard security systems through the standard SAF interface. The SAF interface is supported by security products such as RACF, CA-TOP SECRET, and CA-ACF2. Before allowing access to a particular PSB, the system verifies that the user is authorized to read the PSB.

- Support of dynamic PSB selection and switching within an application. After allocating ddname FOCPSB to a partitioned dataset, you can issue a SET command within an application to select any member PSB.
- Transparent PSB selection. This option is accomplished by including PSB names in Access File Descriptions.
- Support for sharing PCBs between applications. This minimizes overhead when supporting multiple concurrent users.
- The ability to view current settings (PSB, IMSSEC, IMSCLASS, IMSMODE, IMSPZP) by issuing the IMS SET ? query command.
- Access to most IMS databases (HDAM, HIDAM, Fast Path DEDB) except for Fast Path MSDB.

**Note:** DBCTL is not available when using the Cross-Machine Interface (XMI).

## NF467: Automatic Indexed Retrieval (AUTOINDEX)

FOCUS now retrieves data faster by automatically taking advantage of indexed fields in most cases where TABLE requests contain equality or range tests on those fields.

### Syntax      How to Set AUTOINDEX

The syntax is

```
SET AUTOINDEX = {ON / OFF}
```

where:

- ON**            Uses indexed retrieval when possible. ON is the default.
- OFF**           Sets AUTOINDEX off. When AUTOINDEX is OFF, indexed retrieval is only performed when explicitly specified via an indexed view.

In order for indexed retrieval to be performed when AUTOINDEX is set to ON, the request must contain an equality or range test against an indexed field, and that indexed field must exist in the highest referenced segment. Equality and range tests are allowed for all data types; however, if you use a range test with packed data, FOCUS will perform sequential retrieval.

## Example Setting AUTOINDEX

Consider the following Master File Description:

```
FILENAME=INVENT, SUFFIX=FOC,
  SEGNAME=STOR_SEG, SEGTYPE=S1,
    FIELDNAME=AREA, ALIAS=LOC, FORMAT=A1, $
  SEGNAME=DATE_SEG, PARENT=STOR_SEG, SEGTYPE=SH1,
    FIELDNAME=DATE, ALIAS=DTE, FORMAT=A4MD, $
  SEGNAME=DEPT, PARENT=DATE_SEG, SEGTYPE=S1,
    FIELDNAME=DEPARTMENT, ALIAS=DEPT, FORMAT=A5, FIELDTYPE=I, $
    FIELDNAME=DEPT_CODE, ALIAS=DCODE, FORMAT=A3, FIELDTYPE=I, $
    FIELDNAME=PROD_TYPE, ALIAS=PTYPE, FORMAT=A10, FIELDTYPE=I, $
  SEGNAME=INVENTORY, PARENT=DEPT, SEGTYPE=S1, $
    FIELDNAME=PROD_CODE, ALIAS=PCODE, FORMAT=A3, FIELDTYPE=I, $
    FIELDNAME=UNIT_SOLD, ALIAS=SOLD, FORMAT=I5, $
    FIELDNAME=RETAIL_PRICE, ALIAS=RP, FORMAT=D5.2M, $
    FIELDNAME=DELIVER_AMT, ALIAS=SHIP, FORMAT=I5, $
```

The following FOCEXEC contains an equality test on DEPT\_CODE and PROD\_CODE. DEPT\_CODE is used for indexed retrieval since it is in the higher referenced segment.

```
SET AUTOINDEX=ON
TABLE FILE INVENT
SUM UNIT_SOLD RETAIL_PRICE
IF DEPT_CODE EQ 'H01'
IF PROD_CODE EQ 'B10'
END
```

If your TABLE request contains an equality or range test against more than one indexed field in the same segment, AUTOINDEX uses the first index referenced in that segment for retrieval. The following FOCEXEC contains an equality test against two indexed fields. Since DEPT\_CODE occurs before PROD\_TYPE in the Master File Description, DEPT\_CODE is used for retrieval:

```
SET AUTOINDEX=ON
TABLE FILE INVENT
SUM UNIT_SOLD AND RETAIL_PRICE
IF PROD_TYPE EQ 'STEREO'
IF DEPT_CODE EQ 'H01'
END
```

If the equality or range test is against an indexed field that does not reside in the highest referenced segment, indexed retrieval does not occur. In the following example, indexed retrieval is not performed because the request contains a reference to AREA, a field in the STOR\_SEG segment:

```
SET AUTOINDEX=ON
TABLE FILE INVENT
SUM UNIT_SOLD AND RETAIL_PRICE
BY AREA
IF PROD_CODE EQ 'B10'
IF PROD_TYPE EQ 'STEREO'
END
```

## **Implications When a Request Contains an Indexed View**

When indexed retrieval is explicitly specified via an indexed view (as in TABLE FILE filename.indexed\_fieldname) indexed retrieval is performed:

- When AUTOINDEX is set to OFF and the request contains an equality test on the indexed field.

- When AUTOINDEX is set to ON and request contains either an equality or a range (FROM ... TO) test against the indexed field.

## Reference Special Considerations

- AUTOINDEX is never performed when the TABLE request contains an alternate file view (e.g., TABLE FILE filename.fieldname).
- Indexed retrieval is not performed when the TABLE request contains BY HIGHEST or BY LOWEST phrases and AUTOINDEX is ON. They are only performed when the request uses an indexed view.
- When using External Indices, you must explicitly code for indexed retrieval by using the syntax TABLE FILE filename.indexed\_fieldname. AUTOINDEX does not automatically utilize External Indices.

## NF468: HiperBudget

The HiperBudget feature of HiperFOCUS regulates the use of expanded storage on a system-wide basis. This feature requires that the IBI MVS Subsystem be used. For information on the Subsystem, consult the *FOCUS for IBM Mainframe MVS/TSO Installation Guide, Release 7.0*.



## NF469: DRDA Support Enhancements to the DB2 Interface

Starting with FOCUS Release 7.0, the DB2 Interface supports IBM's Distributed Relational Database Architecture (DRDA) Level 2 commands included in DB2 Version 3, and the SET CURRENT PACKAGESET command included in DB2 Version 2 Release 3 and Version 3. The Level 1 DRDA CONNECT command has been supported since FOCUS Release 6.8.

### Syntax      How to Invoke Level 2 DRDA Commands

The syntax for invoking the Level 2 DRDA commands is

```
SQL DB2 SET CONNECTION server-name
SQL DB2 RELEASE {server-name}
                {CURRENT   }
                {ALL [SQL] }
                {ALL PRIVATE}
```

where server-name is the location name of any valid DRDA database server.

The SET CURRENT PACKAGESET command may be used to switch application packages during a session. The effect of this is similar to the Interface SET PLAN command, but does not require that the DB2 thread be closed and reopened (for example, when switching Isolation Levels).

The syntax for issuing the command is

```
SQL DB2 SET CURRENT PACKAGESET {packageset-name}
                                {USER           }
```

where packageset-name is the name of any packageset previously bound into the current DB2 plan.

Use of DRDA and the SET CURRENT PACKAGESET command with the Interface requires that the Interface be installed using the DB2 BIND PACKAGE command. For more information, consult with the DBA at your site.

For a complete description of these commands, refer to the *IBM DB2 SQL Reference for DB2 Version 3*. In the case of the SET CURRENT PACKAGESET command, the *Version 2 Release 3* manual may be used.

## NF470: Loading Access File Descriptions

You can now load Access File Descriptions into memory. This is similar to loading Master File Descriptions and FOCEXECs, and eliminates I/Os required to read the files each time they are referenced.

### Syntax    How to Load Access File Descriptions into Memory

The syntax is

```
LOAD access_type access_name [access_name... ]
```

where:

<code>access_type</code>	Specifies the type of Access File Description to be loaded.
<code>access_name</code>	Specifies the name of the file or files to be loaded. Separate the access_type and access_name(s) with a space.

### Example    Loading the DB2 Access File Description into Memory

The following command loads the DB2 Access File Description named DB2CAT into memory:

```
LOAD FOCSQL DB2CAT
```

The following are the access\_types for the various Access File Descriptions:

ADABAS	FOCADBS
DATACOM	FOCDTCM
DB2	FOCSQL
IDMS	FOCIDMS
IMS (IMS=NEW only)	ACCESS
INFOMAN	ACCESS
MODEL 204	FOCM204
ORACLE	FOCSQL
SQLDS	FOCSQL
S2K	FOCS2K
SUPRA	ACCESS
TERADATA	FOCSQL
TOTAL	FOCTOTAL

## NF471: Enhanced ? SET Command

The ? SET command has been enhanced so that all Release 7.0 parameters are displayed when the command is issued. In addition, you can now display all FOCUS SET parameters by adding the ALL option to the ? SET command.

### **Syntax**      How to Display all FOCUS SET Parameters

The syntax is:

```
? SET ALL
```

## NF472: Enhancement to the TED Command in MVS

The TED command syntax has been enhanced in Release 7.0 to allow you to edit a member of the FOCEXEC PDS without mentioning ddname FOCEXEC. This eliminates extra typing when the membername is not one of the currently allocated ddnames.

### Syntax      How to Edit the FOCEXEC PDS without FOCEXEC

The syntax is

TED name

where:

name

First, the ddname of an allocated sequential dataset or

Second, the member of a dataset allocated to the ddname FOCEXEC.

To force FOCUS to look for a PDS rather than a sequential file, you must use the syntax:

TED FOCEXEC(member)

**Note:** This is only available from the FOCUS command line.

## NF473: The New ADABAS Interface

A new ADABAS interface is available, and can be accessed via a SET command.

### Syntax      How to Access the New ADABAS Interface

The syntax is:

```
SET ADABAS = { NEW|OLD}
```

where:

OLD              Uses the old ADABAS interface.

NEW              Activates the new ADABAS interface. NEW is the default.

**Note:** The term descriptor is used in this document to apply to ADABAS descriptors, subdescriptors, and superdescriptors, unless otherwise noted.

The following enhancements are included in the new ADABAS Interface:

- Field suffixes may now be specified in the FOCADBS file. The field SUFFIX, previously required in the Master File Description to identify superdescriptors and subdescriptors may now be placed in the FOCADBS Access file. The new syntax is:

```
FIELD=fieldname, TYPE={DSC,SPR,NOP,blank},$
```

where:

<code>fieldname</code>	is the fieldname or group in the Master File Description.
<code>TYPE</code>	is the field suffix (DSC, SPR, NOP)
<code>DSC</code>	is for a descriptor.
<code>SPR</code>	is for a superdescriptor comprised of whole fields.
<code>NOP</code>	is for a superdescriptor or subdescriptor comprised of partial fields.
<code>blank</code>	is for non-descriptor fields.

- This feature allows for fieldname consistency between FOCUS and Natural/Predict. For upward compatibility, the old syntax is still supported. If you attempt to include the suffix in both the MFD and Access file, an error message is generated. If you specify a TYPE in the ACCESS file, then `FIELDTYPE = I` is optional in the Master File Description. For more information about field suffixes, please see the *FOCUS Interface to ADABAS User's Manual* (DN 1000017).
- Support for the ADABAS L9 direct call (HISTOGRAM). The HISTOGRAM call is activated by issuing the FOCUS COUNT verb or SUM CNT.field within a TABLE request. L9 calls are, by definition, limited to FIELDS or GROUPs defined to FOCUS with `FIELDTYPE=I` in the MFD, and/or with a field suffix of NOP, DSC, or SPR, and defined to ADABAS as a descriptor, subdescriptor, or superdescriptor.



L9 calls allow an aggregate ISN count to be returned for each unique value on the inverted list at a cost of only one call per unique value. This allows a dramatic efficiency gain over passing each record to FOCUS, and having FOCUS process the count. Selection criteria (e.g. WHERE descriptor EQ '1234') can be passed along with the L9 to further limit the number of calls. If any non-descriptor fields are mentioned in the TABLE request, a HISTOGRAM call is not possible, and FOCUS reverts to its own internal count processing. If a BY field is used, it must be the same field or GROUP name used as the object of the COUNT verb.

- Multi-field JOIN and Short to Long JOIN capability. Multi-field and short to long JOINS are fully supported in the new interface. For Multi-field JOIN, the number of fields used in the JOIN must be the same for both the HOST and the cross reference file. The USAGE of these fields must be identical. Furthermore, the JOIN-TO fields must describe the left-most portion of a superdescriptor defined to FOCUS using the GROUP attribute. For example:

```
JOIN FLDA AND FLDB IN ADBS1 TO KEYPART1 AND KEYPART2 IN ADBS2 AS J1
```

For short to long JOIN, the JOIN-TO field must still be either a descriptor, subdescriptor or superdescriptor, or a field which describes the left-most portion of a GROUP superdescriptor.

IF/WHERE selection on all types of descriptors residing on OCCURS segments (PE and MU) may now be passed to ADABAS in order to take advantage of the appropriate inverted list. Previously, all selection tests on fields residing on MU and PE segments were processed by FOCUS after record retrieval.

- The Format Buffer passed to the ADABAS nucleus from the ADABAS interface is now in compressed form eliminating all spaces and separators. This saves space and gains efficiency.
- Find (S1) call now retrieves the first record. When an ADABAS FIND call on a descriptor is issued, it is issued with the "get first" option to return the first record in the inverted list. This reduces I/O by eliminating unnecessary calls. For example, if a FIND (S1) call returns an answer set containing only one record, no READ ISN (L1) call is issued. If the FIND returns more than one ISN in a list, the GET NEXT option of L1 is used to begin reading the ISN list from the SECOND entry. For information on when FOCUS issues the ADABAS FIND call please see the *FOCUS Interface to ADABAS User's Manual*, DN1000017.
- Up to 66 character fieldnames, including any qualifiers, are now supported by the interface.
- Elimination of extra calls for numeric fields. Previously, IF or WHERE tests against numeric fields passed multiple values in order to test for different SIGNS. In the new interface, only one value per numeric field is passed, based on the preferred sign values in ADABAS. For positive numbers, a sign value of 'F' is passed. For negative numbers, a sign value of 'D' is passed. This reduces the number of calls sent to ADABAS and also eliminates the need for ADABAS to perform any logical sign translation.

- Ability to specify NULL-SUPPRESSION of fields in ACCESS file. The data retrieval strategy of the new interface can produce very efficient ADABAS calls. In certain cases, this strategy depends on whether NULL-SUPPRESSION was used in ADABAS when defining the components of a superdescriptor. To allow the new interface to create the most efficient calls, while maintaining integrity of the answer set, any null-suppressed fields which are components of a superdescriptor should be defined in the Access file using the following syntax:

```
FIELD=fieldname, TYPE={DSC,NOP,SPR,blank}, NU={YES/NO},$
```

where:

**YES** Means that the field is described in the ADABAS FDT with null-suppression (null values are not stored in the file).

**NO** Means that null-suppression is not used. NO is the DEFAULT.

- superdescriptors, where appropriate. If a TABLE request contains IF or WHERE selection criteria against one or more fields which describe the left-most portion of a GROUP superdescriptor, the interface combines this into a test which uses the superdescriptor for greater efficiency. If any of the component (or parent) fields of the superdescriptor are defined to ADABAS with null-suppression, be sure to note this in the FOCADBS Access file to ensure accuracy of reads.

For example, consider the following MFD extract:

```
GROUP=UPERD,ALIAS=SD,USAGE=A8,ACTUAL=A8,FIELDTYPE=I,$  
  FIELD=PART1,ALIAS=P1,USAGE=A4,ACTUAL=A4,$  
  FIELD=PART2,ALIAS=P2,USAGE=A4,ACTUAL=A4,$
```

If, in a TABLE request, you include the following two tests:

```
WHERE PART1 EQ 'ABCD'  
WHERE PART2 EQ 'EFGH'
```

These tests are now combined into:

```
WHERE SUPERD EQ 'ABCD/EFGH'
```

This allows use of the superdescriptor's inverted list and optimizes the ADABAS call. Note that this optimization will only be performed if all null-suppressed (NU=YES) superdescriptor components are explicitly referenced in IF or WHERE tests.

- The new interface (module name ADBSINX) is now fully reentrant and runs above the 16-meg line to take advantage of newer operating system architecture.
- The following ADABAS statistical information is now available.

Number of I/O operations :

Number of commands issued :

Amount of CPU time in sec :

This information is returned after the ADABAS CLOSE call is issued. In order to generate this information, you must specify OPEN=YES in the FOCADBS Access file. You must ALLOCATE in MVS, or FILEDEF in VM, the ddname FSTRACE4 to a sequential dataset, or to the terminal or to a sequential dataset with a logical record length (LRECL) of 80 bytes and a record format (RECFM) of F, FB, or V. This feature helps the DBA and user to more carefully track ADABAS resource usage from within FOCUS.

FSTRACE4 also contains summary information about the types of calls passed to ADABAS, as well as the contents of the ADABAS buffers.

## NF474: APF Internal Authorization

The APF Internal Authorization Feature (APFAUTH) allows you to reduce APF authorization requirements of the MSO server. Modules that do not perform authorized functions no longer have to be contained in an authorized library. APF authorization is required.

### Syntax      How to Invoke the APF Internal Authorization Feature

The keyword APFAUTH should be included in the MSO configuration file. The syntax is

```
APFAUTH = { EXTERNAL | INTERNAL }
```

where:

EXTERNAL      Allows MVS to control authorization screening, preventing unauthorized libraries from executing modules. External is the default.

INTERNAL      Allows MSO to control authorization, allowing modules which do not perform authorized functions to run out of non-authorized libraries.

## Implementation

### **Example** Allocating Libraries which Require APF Authorization

Allocate FOCLIB.LOAD, which must be authorized, to the ddname STEPLIB in the MSO JCL. FOCLIB.LOAD should not appear in any other allocation. For example:

```
//STEPLIB DD DSN=prefix.FOCLIB.LOAD,DISP=SHR
```

### **Example** Allocating Libraries which do not Require APF Authorization

Allocate libraries which do not require APF authorization to the ddname USERLIB in the MSO JCL. For example:

```
//USERLIB DD DSN=prefix.FUSELIB.LOAD,DISP=SHR
```

Insert the Configuration file parameter APFAUTH=INTERNAL into the global service block of FOCMSO.

**Note:** Users will no longer be able to perform authorized functions such as COMPRESS or DYNAM COPY of an entire PDS.

## NF476: Usability Enhancements

The SQL Interface has been enhanced to include several new features and improve functionality. Among the enhancements are synonyms for the KEYORDER attribute, sample runtime CLISTs and JCL, support for the long DECIMAL datatype, improved default segment activation logic, enhancements to the EXPLAIN facilities, the ability to specify index space, parameters, and a collection-id parameter for the GENUSQL EXEC.

### **ASC/DESC Synonyms for the KEYORDER Attribute**

Starting with FOCUS Release 7.0, the values ASC and DESC may be used as synonyms for the values of LOW and HIGH, respectively, in the Access File Description. This is to provide enhanced compatibility with relational database terminology. The functionality of the KEYORDER attribute remains unchanged.

### **Sample Runtime CLISTs and JCL for the MVS Relational Interfaces**

Starting with FOCUS Release 7.0 sample CLISTs and JCL, previously provided only in the Interface Users Manuals, are also provided as members of the FOCSQL.DATA Interface dataset. For the CLISTs, the member name for DB2 is DB2EXCLI, for Teradata it is DBCEXCLI, and for Oracle it is ORAEXCLI. For the JCL, the member names for DB2 are DB2EXJCA (Call Attach Facility) and DB2EXJCT (TSO Attach Facility); for Teradata it is DBCEXJCL, and for Oracle it is ORAEXJCL. As always, these samples require modification in order to conform to site-specific requirements.



## Full Support for the Long DECIMAL Datatype

Starting with FOCUS Release 7.0, full support is provided for the DB2 and SQL/DS DECIMAL datatype with precision of between 1 and 31 significant digits, inclusive. This support encompasses increased lengths for both the USAGE and ACTUAL attributes in the Master File Description (including automatic generation of these lengths via the AUTODB2 facility), as well as support for these lengths in the Direct SQL Passthru facility. Full support is provided for static and dynamic TABLE and MODIFY, and for MAINTAIN.

The following chart shows original source file USAGE formats, conditions and the resulting ACTUAL formats when using ON TABLE HOLD FORMAT DB2/SQL. The field length is represented by n; m is 1 when the original source file USAGE format contains a decimal point, and 0 when it does not.

Source USAGE	Conditions	HOLD USAGE	HOLD ACTUAL
Pn	n LT 16	Pn	P(trunc((n + 2)/2))
	n GE 16	Pn	P(trunc((n - m + 2)/2))
	MISSING = ON	Pn	P8

## SET INCLUDE SUBTREE Segment Activation Behavior is the Default

Starting with FOCUS Release 7.0, the default segment activation logic employed by the FOCUS MODIFY facility has been altered to execute the logic previously invoked by the use of the SQL SET INCLUDE SUBTREE subcommand. This change affects all external Interfaces which support MODIFY write access. This behavior is fully described in both Technical Memo 7898, "Modification of External Interface Segment Activation Logic for MODIFY Write Processing Via the SQL SET INCLUDE SUBTREE Subcommand", and in the *DB2 and SQL/DS Read/Write Interface Users Manual* for FOCUS Release 6.8How to Modify the SET INCLUDE SUBTREE.

### Syntax      How to Modify the SET INCLUDE SUBTREE

You can issue the following MODIFY subcommand to invoke the behavior that was previously the default:

```
SQL SET INCLUDE LATERAL
```

This subcommand must be placed on the line immediately following the MODIFY command.

## **Enhancements to the Interface EXPLAIN Facilities for DB2 and SQL/DS**

Starting with FOCUS Release 6.8 PUT Level 9410 and continuing through FOCUS Release 7.0, two new FOCEXECs have been included to accommodate enhancements in the EXPLAIN facilities of both DB2 Version 3 and SQL/DS Version 3 Release 4. These FOCEXECs are EXPDB231 for DB2, and EXPSQL34 for SQL/DS. Member EXPDB231 is located in the FOCEXEC.DATA dataset on the MVS distribution tape. EXPSQL34 FOCEXEC is located on the FOCUS maintenance disk, and is copied to the production disk during the Interface installation procedure. Provided you are only running the above specified releases of DB2 and/or SQL/DS, your site may wish to rename EXPDB231 and EXPSQL34 to EXPDB2 and EXPSQL, respectively, as these names are generally more familiar to the FOCUS user community.

The Master and Access File Descriptions utilized by these facilities have been altered for FOCUS 7.0; however, they retain the same names as in prior releases of FOCUS, and they may still be used with currently supported versions of EXPDB2 and EXPSQL.

### **User Specification of DB2 Index Space Parameters**

Starting with FOCUS Release 7.0, you can override the default parameters for the DB2 index space implicitly created by FOCUS CREATE FILE and HOLD FORMAT DB2. Use the short form of the syntax for commands that fit on one line.

## **Syntax**      **How to Override Default Parameters for the DB2 Index**

The syntax is:

```
SQL DB2 SET IXSPACE index-spec
```

Use the long form for commands that exceed one line. The syntax is

```
SQL DB2  
SET IXSPACE index-spec  
END
```

where index-spec is the portion of the SQL CREATE INDEX statement beginning with the USING-BLOCK as specified in the IBM DB2 SQL Reference CREATE INDEX syntax diagram. Up to 94 bytes of specifications may be used. When the long form of the syntax is used, the SQL DB2 and END commands must be on separate lines. For example, to specify the USING-BLOCK, FREE-BLOCK, and CLUSTER portions of the CREATE INDEX statement, you would enter the following:

```
SQL DB2  
SET IXSPACE USING STOGROUP SYSDEFLT PRIQTY 100  
SECQTY 20 FREEPAGE 16 PCTFREE 5 CLUSTER  
END
```

The current setting for IXSPACE may be determined by the Interface query command:

```
SQL DB2 ?
```

If the current setting is the default, no setting for IXSPACE will display in the SQL DB2 ? output.

To reset to the DB2 default index space parameters, simply issue the SET IXSPACE command with no operands.

**Note:** This feature is not upwardly compatible with the similar feature introduced in FOCUS Release 6.8 Put Level 9410. The FOCUS 7.0 version requires that you specify the USING keyword (if the using-block is specified), and allows the options to be specified selectively and independently of each other. The 6.8 version requires that the using-block be specified.

## Enable Collection-Id Parameter for SQL/DS Interface GENUSQL EXEC

Starting with FOCUS Release 7.0, an additional parameter has been added to the GENUSQL EXEC, the EXEC which is used to generate static SQL application packages for FOCUS compiled MODIFY requests. This parameter, called COLLID, is used to pass a value to the collection-id parameter of the SQL/DS SQLPREP EXEC, allowing you to specify a value that will be used as the qualifier for the created application package. The value can be different from the VM userid or the value specified for the SQLUID parameter, if one exists. This capability proves useful in environments which make use of IBM's Distributed Relational Database Architecture (DRDA), as the passed value is used for the collection-id at any remote locations..

### Syntax      How to Invoke the COLLID Parameter

The syntax for specifying the value for COLLID is

```
EX GENUSQL MOD1 (SQLUID=USER1 SQLPSWD=PASS1 COLLID=USER2
```

where:

MOD1	Is the name of the MODIFY FOCEXEC for which the static SQL module is being created.
SQLUID	Is (as in prior releases of FOCUS) the SQL/DS authorization id for the SQLPREP EXEC. If COLLID is not specified, this is also the qualifier for the application package that is produced. If SQLUID and COLLID are not specified, the authorization id and package qualifier default to the current VM userid.
SQLPSWD	Is (as in prior releases of FOCUS) the SQL/DS authorization password for the SQLPREP EXEC. If SQLUID is specified, the SQLPSWD must also be specified.
COLLID	Is the collection-id, which will be used as the qualifier for the SQL/DS application package that is produced by the SQLPREP EXEC. If COLLID is not specified, it defaults to the value of SQLUID (if specified), or the current VM userid (if SQLUID is not specified).

For more information on the SQLPREP EXEC in general, and the collection-id parameter in particular, consult the *IBM SQL/DS Application Programming for VM Systems* manual.

## Relaxing of Join Field Data Length Requirement for Alphanumeric Fields

Beginning in FOCUS release 7.0, the requirement (published in Technical Memo 7915, "Update to the DB2 and SQL/DS Interface Users Manual") that field lengths be equal for join fields has been lifted for alphanumeric fields. Alphanumeric fields may now have different lengths, and the same output will be displayed regardless of Interface optimization behavior. The restrictions for other data types remain, as do the restrictions described in the document titled *Static SQL for TABLE Requests*, later in this chapter.

## NF477: FASTPDS

Activating FASTPDS causes MSO to read all partitioned datasets in the server's startup JCL and copy their directory information into memory. These tables of members are used for all future references to those libraries. The I/O to find the members is only done once, at initialization, which results in faster access to data in the globally allocated libraries. When activated, these global files become read-only, and may not be updated by MSO users.

### Syntax      How to Activate FASTPDS

The keyword FASTPDS should be included in the MSO configuration file. The syntax is

```
FASTPDS = { ON | OFF }
```

where:

- |     |  |
|-----|--|
| ON  | Turns FASTPDS on.                      |
| OFF | Turns FASTPDS off. OFF is the default. |



## NF478: MSO CONSOLE Browser

CONSOLE users are now allowed to browse datasets which are allocated to the MSO server. By placing a 'W' to the left of the user (or region) in Display Users, a list of all allocations to that user will be displayed. It is then possible to tab down to the desired dataset and select it by entering 'S' to the left of the dataset name. If the dataset is a PDS, a list of PDS members will be displayed. Selecting the desired member by placing an 'S' to the left of it will enable you to see the contents of the member. If the dataset is sequential, the contents of the dataset will be displayed. If the file has not been opened, the browser will not be able to display the contents of the dataset and will receive the message 'UNABLE TO OPEN DATASET'.

## NF479: The New MSO/CICS Interface

The CICS component of the FOCUS Multi-Session Option (MSO) has been substantially rewritten. This rewrite provides keyboard lock, attention key support, simplified installation, enhanced diagnostic facilities, a cleaner user interface, and enhanced error messages. Installation, samples, zaps, and user exits are new, as are error messages. Refer to Technical Memo TM7913 "The MSO/CICS Interface" for further information on installing and using the MSO/CICS Interface.

## NF480: Fast Logon Enhancements

Fast Logon allows you to bypass the logon screen and enter logon information from the VTAM screen. This feature has been enhanced to include the account, newpass, and group fields available from the VTAM MSO logon screen. Any DATA field may be omitted.

### Syntax      How to Use Fast Logon

The syntax is:

```
LOGON APPLID(msoappl) DATA(userid/password/service/account/newpass/  
group)
```

### Example      Entering Logon Information from the VTAM Screen

For example, in order to specify userid, password, and account, you would enter:

```
LOGON APPLID(msoappl) DATA(userid/password//account)
```

This would pass nothing for the service, newpass and group fields. Spaces are not accepted within the contents of the DATA fields.

## NF481: DYNAM Utilities Menu

A new menuing system is available to enable users to manipulate and submit files. By utilizing DYNAM functions, this menu is able to run in both TSO FOCUS and MSO. Available functions include ALLOCATE, RENAME, DELETE, COPY, SUBMIT, and COMPRESS, with the added ability to include a private application on the primary menu. No additional allocations are needed in order to access the DYNAM UTILITIES MENU.

Issue the following command to enter the menu:

`EX DYMENU`

**Note:** The COMPRESS option is not available in TSO FOCUS via DYNAM.

## NF482: IMS Enhancements via SET IMS=NEW

FOCUS Release 6.8/9312 introduces a new IMS Interface. This Interface contains many new features, including:

- The XMI server as a replacement for the FOCBMP server.
- A comma-delimited FOCPSB file.
- A new trace facility.
- New SET commands.
- The IMDTEST verification program.
- Improved security.
- An IMS Access File.

### Accessing the New IMS Interface

There are two ways to access the new IMS Interface. The first is to issue the following command from the FOCUS prompt:

```
SET IMS=NEW
```

The second method is to use the keyword SUFFIX=IMSX in your Master File Description. If you do not use either of these methods to access the new IMS Interface, the old Interface will be used by default.

### The XMI Server

When using the new Interface, you must follow PARM with the DFSRRC00 program.

## Syntax      How to Use the XMI Server

```
PARM=- ' [ DLI | BMP | DBB ] , XMI , psbname '
```

**Note:** The second parameter is XMI. XMI is the new server, and it replaces the FOCBMP server.

## Comma-Delimited FOCPSB

The FOCPSB now uses keywords and is comma-delimited. For example:

```
FOCPSB=EXTENDED, $
PCBNAME=<master>,PCBTYPE=[DB|SKIP|TERM], LOW=xxx,
HIGH=xxx, $
```

where:

FOCPSB=EXTENDED  
, \$

This is required and must be the first record of the FOCPSB.

PCBNAME

Is used to associate the Master File with the corresponding IMS PCB to be used for data retrieval. If this keyword is blank, the PCBTYPE keyword must specify SKIP or TERM.

PCBTYPE

Is used to specify the type of PCB defined in the IMS PCB definition. If the IMS PCB is an I/O PCB, you must specify TERM for PCBTYPE. If the PCB is a BD PCB, specify SKIP if you wish to ignore it; otherwise, specify DB.

LOWVALUE	Is used when setting up the FOCPSB for database concatenation and/or partitioning. Use LOWVALUE to specify the starting range of the key field to be used for concatenation. The value must be alphanumeric.
HIGHVALUE	Is used when setting up the FOCPSB for database concatenation and/or partitioning. Use HIGHVALUE to specify the ending range of the key field to be used for concatenation. The value must be alphanumeric.

Each record in the FOCPSB must end with a comma and a dollar sign (,\$).  
Records may span multiple lines.

## The New Trace Facility

The IMS interface now has a trace facility, which allows you to turn any of three traces by allocating the ddname associated with the trace. The traces are:

FSTRACE	Shows the initial setup and all DL/I commands issued to IMS for processing.
FSTRACE4	Shows the setup of the SSA to be used for key retrieval.
FSTRACE5	Shows the flow of the Interface as it progresses through the different modules that are associated with it.

## **New SET Commands**

The new IMS Interface has five new SET commands:

- 1. SET IMS=NEW**  
This command is used to activate the new IMS Interface.
- 2. TSO/MVS IMS SET IMPZB xx (default 00)**  
This allows you to select the DRA table used at DBCTL initialization. This command should be issued prior to the DBCTL command.
- 3. TSO/MVS IMS SET DBCTL ON**  
This command is used to change the access mode from the default to the DBCTL access mode.
- 4. TSO/MVS IMS SET PSB <psbname>**  
When DBCTL is on, you must issue this command to select the PSB used in your report request.
- 5. TSO/MVS IMS SET ?**  
This command is used to display the status of all IMS SET commands.

## **The IMDTEST Verification Program**

This program should be executed prior to the installation of the IMS/DBCTL Interface. It verifies that your site has properly installed the DBCTL option of IMS/ESA.

## **Improved Security**

The new IMS Interface has improved security which allows you to use security classes to control access to the IMS PSBs on a per user basis.



## The IMS Access File

The new IMS Interface introduces an IMS Access File. This file is allocated to the ddname ACCESS and contains:

PSB=<psbname>

This is useful in the EDA environment because it eliminates the need for issuing a SET command.

## General Enhancements

The new IMS Interface also includes:

- Better optimization of IF and WHERE screening.
- Short-to-long JOINS.
- IMS-to-IMS JOIN with selection criteria.
- Cross-format JOINS.
- Partial key selection.

## NF483: The MSO Resource Manager (MRM)

The MSO Resource Manager is a facility which monitors the activity of all users in each priority group. By doing so, it is able to prevent individual users from taking all CPU resources within their given priority group. Every 10 seconds MRM will query the users based on their SMF accounting statistics. If a user consumes more than 50% of the CPU available to MSO, their internal priority will be reduced within their priority group, allowing other users to obtain CPU time. APF authorization is required.

### Syntax    How to Use the MRM Facility

The keyword MRM should be included in the MSO configuration file. The syntax is:

```
MRM = { ON | OFF }
```

where:

- ON**            Activates MRM. ON is the default provided you are running with APF authorization and SMFNUM has been specified or allowed to default to 0. If these conditions are not met, MRM is set to OFF.
  
- OFF**           Deactivates MRM.

## NF484: Allowing VSAM File Allocation in MSO JCL

Specifying SZERO=YES in the configuration file allows subpool 0 to be shared across tasks. This is required when VSAM files are allocated in the MSO JCL. It is not needed when VSAM files are not accessed, or when they are allocated to individual users with the DYNAM command. APF authorization is not required.

### Syntax      How to Allocate VSAM Files in MSO JCL

The keyword SZERO should be included in the MSO configuration file. The syntax is

```
SZERO = { YES | NO }
```

where:

YES                      Allows subpool 0 to be shared.

NO                        Prevents subpool 0 from being shared. NO is the default.

## NF485: Dynamic GETPRV Exit

The FOCSAM Interface contains a user exit that can be invoked as an alternative to the lowest-level READ routines that are part of the Interface. This exit makes it possible to combine simple user-written code devoid of any dependence on internal FOCUS structures, with the logical retrieval functions of the FOCSAM Interface, such as record selection logic, treatment of multiple record types within a single physical file, JOINS between various types of files, and so forth.

This exit has been developed as an alternative to the original GETPRV Private Exit. The major enhancements are as follows:

1. By providing a new parameter, HANDLE, the exit now supports reentrancy, which reduces storage and enhances invocation performance.
2. Support for multiple concurrent exit processors is provided through an ACCESS file where a user exit module can be named on a per Master File basis.
3. The user exit is dynamically called at execution time, unlike the original GETPRV which required a link to either VVSET or FOCSAM, thus avoiding the need to run complex link-edits for every FOCUS upgrade or maintenance application.
4. An initialization call has been added to the exit to allow for the exit module to perform initial housekeeping.

5. The QUALIF parameter has been enhanced to support the additional options of (O) OPEN file, (R) OPEN request (position), (C) CLOSE, and (F) FIN. These new control options are in addition to the S, G and E read options that have always been available in the original GETPRV exit and eliminate the need for the separate PRVCLS call of the original exit.
6. There is support for multiple positions on the same file.

Functionally, the private code is a substitute for READ calls typically used against but not limited to key-sequenced VSAM files, and can be used against any data source that can be represented as such a file. Such files include compressed or encoded files. It need not deal with any intra-record structures represented by OCCURS clauses, or with the translation of FOCUS IF conditions into lower-level criteria. Both of these functions are performed by the driving logic in the FOCSAM Interface.

For complete Technical and Implementation information, please refer to Technical Memo 7916, *The Dynamic GETPRV Exit*.

## NF486: Dynamically Setting the Addressing Mode

A new SET command is available to switch the AMODE of the FOCSAM Interface (which reads VSAM and flat files) to 24-bit addressing. The Interface runs in 31-bit mode by default, in order to take advantage of modern operating system architecture. By extension, the Interface also builds 31-bit addresses for VSAM buffers and ACBs. However, some external VSAM buffering packages run in 24-bit mode, and do not recognize 31-bit addresses. The new SET command allows the Interface to be run with these 24-bit programs.

### Syntax      How to Set the Addressing Mode

The syntax is:

```
{MVS|CMS} VSAM SET AMODE n
```

where:

`n` is 24 or 31      With AMODE 31, FOCSAM builds ACBs and buffers in 31-bit addresses. 31 is the default. With AMODE 24, FOCSAM builds ACBs and buffers in 24-bit addresses.

To determine the addressing mode that is in effect at any time, you can issue the query

```
{MVS | CMS} VSAM SET ?
```

which returns the following output:

```
(FOC1177) SET OPTIONS - : BUFND = n / BUFNI = n / AMODE = n
```

If you are not using any external programs or buffering packages which require 24-bit addresses for the ACB or buffers, you will not need to change the default.

## NF487: Enhancement to the ZCOMP1 User Exit

The ZCOMP1 decompression user exit has been enhanced. It now provides an initialization entry point for housekeeping functions, and it supports re-entrant code. Existing ZCOMP1 programs will be unaffected by the enhancement and will remain upwardly compatible. Please refer to TM7917 *Installing the ZCOMP1 User Exit* for information on how to implement the enhancements.



## NF488: AUTODATACOM

FOCUS Release 7.0 introduces AUTODATACOM, which you can use to automatically create Master and Access File Descriptions for your CA-DATACOM/DB data using information stored in the CA-DATACOM/DB Data Dictionary.

To start AUTODATACOM, issue the following command from the FOCUS prompt:

```
EX AUTODTCM
```

The following input screen will be displayed:

```
Auto Datacom      Automatic Master and Access Facility      Main Menu

                                Datacom Names
                                (Enter URT and Information for A, B, C or D)

User Requirements Table (URT) :
Data Table
  A. TBLNAM/DBID           :      /
  B. TBLNAM/DB Occurrence  :      /
  C. Table Occurrence      :
  D. Dataview Occurrence   :

                                Master and Access File Names
Member      :                               Replace Member?: NO
Master PDS: userid.MASTER.DATA
Access PDS: userid.ACCESS.DATA
PF1-Help           ENTER>Create Member      PF2-List
PF3-Exit Auto Datacom      PF4-Log Menu Choices
PF5-Edit Master Member PF6-Edit Access Member PF9-Draw Master Member
```

where:

User  
Requirements  
Table (URT)

The name of the User Requirements Table that will be used in the Access member. It can be from 1 to 8 characters in length.

TBLNAM/DBID

TBLNAM is the 3 character table name to be described in the Master and Access File Descriptions. DBID is the database ID in which the table resides. If a TBLNAM is supplied, DBID must also be supplied.

TBLNAM/DB  
Occurrence

TBLNAM is the 3 character table name to be described in the Master and Access File Descriptions. DB occurrence is the 1 to 32 character name of the database occurrence in which the table resides. If a value is supplied for TBLNAM, you must also supply a value for DB occurrence.

Table  
Occurrence

The name of the table to be described in the Master and Access File Descriptions. If a value is supplied for table occurrence, that value overrides TBLNAM/DBID and TBLNAM/DB. The value can be from 1 to 32 characters in length.

Dataview  
Occurrence

The name of the CA-DATACOM/DB dataview that contains all or a subset of table elements to be described in the Master and Access File Descriptions. Dataview occurrence overrides all previous CA-DATACOM/DB table specifications. The value can be from 1 to 32 characters in length.

Member

The name you select for referring to the data in requests. This name can be from 1 to 8 characters in length, and must refer to a member of a PDS.

Master PDS

Is the fully qualified dataset name of the PDS in which the Master File Descriptions created by AUTODATACOM will be stored. You do not need to surround the dataset name with quotation marks. The default is userid.MASTER.DATA. The dataset name can be up to 44 characters in length. If the PDS you specify does not exist, you will be prompted to create it.

Access PDS

Is the fully qualified dataset name of the Master in which the Access File Descriptions created by AUTODATACOM will be stored. You do not need to surround the dataset name with quotation marks. The default is userid.ACCESS.DATA. The dataset name can be up to 44 characters in length. If the PDS you specify does not exist, you will be prompted to create it.

Replace Member?

Allows you to specify whether or not you want to overwrite existing Master and Access File Descriptions. NO or N is the default.

## NF490: Online Release Information

Release statistics, installation and operational changes, program temporary fix (PTF) information, and release notes are now available online. This information can be found in member README of ERRORS.DATA in MVS, or in README ERRORS in CMS. These files contain all of the information that was previously published in the Maintenance Log for each FOCUS PUT level, and replace the Maintenance Log. To view the online release information, issue the following command from the FOCUS prompt:

```
HELP README
```

## NF496: Static SQL for TABLE Requests

Starting with FOCUS Release 7.0, a facility is available which allows users to generate DB2 and SQL/DS static SQL modules for virtually any FOCUS TABLE report requests contained in any FOCEXEC. The FOCEXEC may contain any other FOCUS commands (such as MODIFY or SET commands) as well as any Dialogue Manager logic. Only the TABLE requests will be translated into static SQL.

In prior releases of the Interface, FOCUS TABLE access to the RDBMS was entirely through dynamic SQL requests. While dynamic SQL is ideal for those applications (such as ad hoc reporting) in which you do not know in advance the SQL statements you will execute at run time, it is less desirable for applications that do not require such flexibility.

In contrast, static SQL is ideal for situations in which you know beforehand all the SQL statements a procedure may execute. You register the procedure itself, including the SQL statements, with the RDBMS through a process known as binding (for DB2) or preprocessing (for SQL/DS). The result is a database object, a DB2 application plan (or package) or SQL/DS application package, that is stored in the database and retrieved whenever you run the procedure.

An important part of this process is that the RDBMS optimizes each SQL statement in the program and chooses an access path for it. A data access path consists of low-level data access requests that the RDBMS formulates and stores in internal format. When you execute the procedure, the RDBMS retrieves these requests and executes them immediately.

The net effect is that the SQL statements in the static request are not reinterpreted at run time. The access path for each statement has already been chosen and is reused each time you run the procedure.

Therefore, with static SQL for TABLE, you can quickly and easily create application plans or packages (in DB2) or packages (in SQL/DS) for FOCUS procedures involving TABLE requests.

When you run these procedures, you take advantage of the following security and performance benefits of static SQL:

- Procedure level security

Privileges (for instance, SELECT) for operations on a table are available only to authorized users of the application plan (or package) associated with the FOCEXEC. No access to the underlying tables is permitted outside of the FOCEXEC. Under dynamic SQL, privileges must be granted on the actual tables. Many sites do not allow users dynamic access to tables, because users of these programs are then able to access the same tables for operations using other programs, such as QMF. Static SQL specifically addresses this security concern.

- Performance improvements

While the type of reporting operations performed by FOCUS requests do not usually exhibit substantial performance improvement when converted to static processing, the SQL syntax checking, object authorization verification, and query optimization are performed when the procedure's static SQL module is bound, as opposed to at run time for a dynamic SQL request. This generally results in a slight performance improvement, as well as decreased contention for the system catalog tables needed by the RDBMS to perform these operations. In general, the more SQL statements a procedure executes, the greater the performance improvement.

The actual speed of data retrieval is not enhanced by static SQL. In a static SQL procedure, the time required to retrieve a given set of rows is identical to that of the same procedure executing dynamically, minus the cost of access path selection, syntax checking, and authorization verification for the SQL statements.

**Note:** If RDBMS table statistics change, or if table or index structures are altered after the static procedure is bound, the procedure should be rebound to ensure optimum performance. Refer to step 6 "Optionally BIND the Plan for the FOCEXEC" in the section below, *Static FOCEXEC Creation for DB2* for a discussion of BIND in DB2. See step 4 "Execute the GENUSQL EXEC" in the section below, *Static FOCEXEC Creation for SQL/DS* for the corresponding information in SQL/DS.



- Ease of use

The FOCUS implementation of static SQL for TABLE offers a far less complex and time-consuming development cycle than is normally the case with static SQL applications. No third-generation language or SQL skills are required. In fact, you can write a FOCUS application destined to use static SQL without coding a single SQL statement. Additionally, you may develop, test, and revise your FOCEXEC using dynamic SQL without going through the more cumbersome static SQL preparation process every time you make a change. You do not have to create the application plan or package until the end of the development cycle.

## Requirements for Static SQL for TABLE

The vast majority of existing dynamic applications can be implemented as static procedures with little preparatory work. No special coding (for instance, embedded SQL) is required for a FOCEXEC to use static SQL. Most current FOCEXEC procedures that access DB2 or SQL/DS tables using TABLE FILE or MATCH FILE commands will not need to be altered in any way to be registered as a static SQL procedure.

Your FOCEXEC may also contain SQL commands (for instance, SQL COMMIT WORK, INSERT, or UPDATE), MODIFY procedures, Direct SQL Passthru commands, and Dialogue Manager commands, although none of these will be converted into static SQL. Your FOCEXEC may also invoke other FOCEXECs, although these would need to be converted separately if you wish to run them statically (see the section below, *Usage Restrictions*).

You must satisfy the following requirements to create a static SQL procedure:

- For DB2:

TSO ATTACH is not supported.

The DB2 Interface must have been installed to use the Call Attachment Facility (CAF). You can verify this by using the SQL DB2 ? command to display Interface settings (Call Attachment Facility should be ON). There is no similar requirement for SQL/DS.

- For SQL/DS:

The SQL/DS product does not support double-byte character set (DBCS) datatypes in static SQL procedures of the type generated by the Interface.

**Note:** Certain resource restrictions exist for any procedure that uses static SQL; under some circumstances they may make it impossible to COMPILE a FOCEXEC procedure to use static SQL. The section below entitled *Resource Restrictions* explains these restrictions and how you can determine if your procedure may be affected.

## Static TABLE Module Creation

Creating a static SQL module from TABLE requests in a FOCEXEC is accomplished by processing the FOCEXEC through the Interface SQL COMPILE facility. This facility does not compile FOCUS code in the same way as the FOCUS COMPILE facility; its sole function is to create static SQL code for TABLE requests.

The SQL COMPILE facility has several compilation options; the combination of these options that you will use to generate your static SQL modules depends largely on the functions each FOCEXEC performs. The choice of the proper combination of options will require some thought for each FOCEXEC being compiled, and the combinations will probably vary with each FOCEXEC.

## Syntax      How to Create a Static SQL Module from TABLE Requests in a FOCEXEC

The syntax for invoking the SQL COMPILE facility is

```
SQL [target_db] COMPILE focexec [-OPTION(options)] [parameters]  
END
```

where:

<i>target_db</i>	Indicates the target RDBMS. Acceptable values are DB2 or SQLDS. Omit if you previously issued the SET SQLENGINE command.
<i>focexec</i>	Is the name of the FOCEXEC from which the static SQL module will be generated.
<i>options</i>	Indicates the execution options to be used, each separated by a single space. No spaces are allowed between the word -OPTION and the opening parenthesis, and any text between unrecognized option names is ignored. Any combination of the following options may be used:

- INTERPRET

This option causes the FOCEXEC to be executed simultaneously with the generation of the static SQL module. It is useful when compiling a request that contains TABLE requests that are conditionally executed depending on output from other TABLE requests (for example, by testing the value of &LINES). If this option is not used, TABLE requests will not produce any output; therefore, some conditionally executed logic may not be processed by the facility.

**Note:** The INTERPRET option is mandatory if the FOCEXEC being compiled contains MATCH FILE commands.

- FROM(fxname)

fxname is the name of a FOCEXEC that calls the FOCEXEC being compiled. No spaces are allowed between the word FROM and the opening parenthesis. This option is useful when compiling a FOCEXEC that is usually called from another FOCEXEC (for example, a FOCEXEC that displays a menu and sets ampersand variables that are used by the FOCEXEC being compiled).

- REPEAT[(n)]

n is the number of times you wish to process the FOCEXEC in order to generate SQL. For each iteration, the generated SQL will be appended to the static module being generated. No spaces are allowed between the word REPEAT and the opening parenthesis, if n is specified. If n is not specified, you will be prompted after each execution as to whether you wish to end or execute the FOCEXEC again. This option is useful when compiling a FOCEXEC where values assigned to ampersand variables can affect the SQL that is generated, and you wish to generate all possible versions of SQL that can result from the FOCEXEC. Values for variables are assigned by either prompting for them with Dialogue Manager, or by using the TED option described below.
- TED

This option is used to edit the FOCEXEC with the TED editor prior to compiling it. This can be particularly useful in conjunction with the REPEAT option. In that case, TED would be invoked prior to each execution of the FOCEXEC, allowing you to make incremental changes at each iteration.

*parameters*

Are pairs of FOCUS ampersand variables and their corresponding values, in the form variable = value. Multiple pairs are separated from each other by commas. Any variables not specified in this manner will be prompted for by FOCUS during the SQL COMPILE. Depending on where these variables appear in any TABLE or MATCH FILE request(s) being compiled, their values will either be hard-coded into the generated SQL (in which case the same values must be specified for them at run time), or SQL host variables will be generated as place-holders for them (in which case new values may be specified for them at run time). Any variables that are not part of a TABLE or MATCH FILE request will have their values substituted only for the duration of the SQL COMPILE; new values may be specified for them at run time.

For more information on the SQL generation process at both SQL COMPILE and SQL RUN time, see the section below, *SQL COMPILE and SQL RUN Processing*.

## Static TABLE Module Execution

The syntax for executing a FOCEXEC for which a static SQL module has been created using the SQL COMPILE facility is

## Syntax      How to Execute a Static TABLE Module

```
SQL [target_db] RUN focexec [parameters]  
END
```

where:

- |                  |  |
|------------------|--|
| <i>target_db</i> | Indicates the target RDBMS. Acceptable values are DB2 or SQLDS. Omit if you previously issued the SET SQLENGINE command. |
| <i>focexec</i>   | Is the name of the FOCEXEC for which the static SQL module was previously generated by the SQL COMPILE facility.         |

### *parameters*

Are pairs of FOCUS ampersand variables and their corresponding values, in the form variable = value. Multiple pairs are separated from each other by commas. Any variables not specified in this manner will be prompted for by FOCUS during the SQL RUN execution. Depending on where these variables appear in any TABLE or MATCH FILE request(s) being executed by the FOCEXEC, their values have either been hard-coded into the generated SQL by the SQL COMPILE facility (in which case the same values must be specified for them at run time), or SQL host variables have been generated as place-holders for them (in which case new values may be specified for them at run time). Any variables that are not part of a TABLE or MATCH FILE request are processed in the usual way by FOCUS; new values may be specified for them at run time.

For more information on the SQL generation process at both SQL COMPILE and SQL RUN time, see the section below, *SQL COMPILE and SQL RUN Processing*.

## SQL COMPILE and SQL RUN Processing

In order to make optimal use of the Static SQL for TABLE facility, it is helpful to understand the SQL generation process that takes place during the execution of both the SQL COMPILE and SQL RUN commands.



When the SQL COMPILE facility is invoked, any TABLE or MATCH FILE commands that access DB2 or SQL/DS tables will generate input into two modules: a Data Base Request Module (DBRM) for DB2, or ASMSQL file for SQL/DS, containing the static SQL statements to be executed at SQL RUN time, and a STUBLIB load module containing an Assembler program. The DBRM or ASMSQL file will be used as input to the BIND command (DB2) or GENUSQL EXEC (SQL/DS), respectively, that must be executed subsequent to the SQL COMPILE process to generate an application package in the RDBMS. The STUBLIB load module will be used to invoke the static SQL statements from the FOCEXEC at SQL RUN time. The names of both the DBRM or ASMSQL file, and the STUBLIB load module will be identical to the FOCEXEC name. During the SQL COMPILE, multiple RDBMS requests will cause additional SQL requests to be appended to the DBRM or ASMSQL file until all applicable requests have been executed in the facility and all the resultant SQL statements have been stored.

It is important to be aware of what FOCUS TABLE and MATCH FILE request syntax will cause SQL host variables to be generated in the SQL statements that are created. This is because this can affect the ability to substitute new variable values at SQL RUN time, as well as cause different RDBMS access paths to be selected than if the same statements were executed with hard-coded values rather than SQL host variables. Different access paths may be selected by the RDBMS because when host variables are used, the optimizer must make assumptions regarding the proportion of records in the table that will pass the screening condition(s).

The general rule is that the SQL COMPILE facility will substitute an SQL host variable for any literal that appears to the right of a comparison operator in an IF or WHERE screening condition, regardless of whether or not it is an ampersand variable in the FOCEXEC. This is done because the Interface is unable to discern the difference between hard-coded values and values substituted for ampersand variables, since FOCUS performs the variable substitution prior to invoking the Interface. Therefore, to support variable substitution for literals in screening conditions at run time, the SQL COMPILE facility assumes that all literals following comparison operators are variable. In addition, if there are ampersand variables in a request that could change the generated SQL depending on their values, then all possible variations of SQL must be generated in a single execution of the SQL COMPILE facility. This is done using the REPEAT option (see the section above, *Static TABLE Module Creation*), specifying a different value for the ampersand variable with each iteration until all possible values have been specified. Examples of such variables include variable filename(s) or fieldname(s) in a FOCEXEC, or variables that would cause the Interface's OPTIMIZATION setting to change, thereby altering the generated SQL.

When the SQL RUN facility is invoked, the FOCEXEC is executed as usual by FOCUS. In addition, the Interface will generate SQL statements in memory from any TABLE or MATCH FILE requests which reference DB2 or SQL/DS tables. When a TABLE or MATCH FILE request is encountered that accesses DB2 or SQL/DS tables, SQL is generated; however, with the SQL RUN facility, it is not transmitted to the RDBMS via the dynamic Interface plan. Instead, the STUBLIB load module is invoked, and the RDBMS application plan or package is loaded. The static SQL statements in the plan or package are then searched until a statement is found that exactly matches the SQL that has been generated. If an appropriate statement is located, it is then executed. If no such statement can be found in the plan or package, a FOC1526 error is issued, and execution terminates.

The reason for the regeneration of SQL statements at SQL RUN time for comparison to the previously generated static SQL is to provide increased flexibility. This type of processing allows you to make changes to the source FOCEXEC, and as long as the generated SQL is not affected, the FOCEXEC does not have to be reprocessed by the SQL COMPILE facility in order to run in static mode. A FOCEXEC only need be recompiled if changes are made that would cause the generated static SQL statements to be different than those generated by the most recent SQL COMPILE.

## SQL Host Variable Length Considerations

In order for an SQL predicate containing host variables to be considered for index access, the DB2 and SQL/DS query optimizers require that the lengths of those host variables must be compatible with the lengths of the corresponding columns in the RDBMS. For this reason, the ACTUAL attribute in the Master File Description for any fields which will be used in screening conditions or as join fields in cross reference files (in FOCUS-managed joins) must be equal to the length of the column in the RDBMS, as this attribute is used in creating the host variable description for these fields in the generated Assembler program.

This requirement is sufficient for all data types except packed decimal fields. This is because the ACTUAL attribute alone can not be used to determine the exact precision of the RDBMS DECIMAL column. For example, an ACTUAL attribute of P6 could correspond to a DECIMAL column of precision either 10 or 11. For this reason, a new field attribute, PRECISION, has been added to the Access File Description for packed decimal fields. The syntax for specifying this attribute is

```
FIELD = fieldname, PRECISION = precision-type, $
```

where:

*fieldname*

Is the name of the indexed packed decimal field which will be used in screening conditions, or as a join field in a cross reference file in a FOCUS-managed join.

*precision-type*

Is either ODD or EVEN. ODD is the default, and causes the DECIMAL column precision for the host variable to be calculated using the formula:

$$2n - 1$$

where n is the length specified in the ACTUAL attribute in the Master File Description. EVEN will cause the following formula to be used:

$$2n - 2$$

For example, with an ACTUAL of P6 and the default precision-type of ODD, the host variable length would be declared as 11, meaning that index access will only be considered (provided the column is indexed) if the precision of the RDBMS column is 11. If the precision-type is EVEN, then the host variable length will be 10, which would enable index access for DECIMAL columns of precision 10.

**Note:** The SQL/DS optimizer will not allow index access from static SQL for packed decimal columns of even precision, no matter which setting for precision-type is used.

## Static FOCEXEC Creation for DB2

In order to use Static SQL for TABLE in the DB2 environment, you need only allocate some additional DDNAMEs and execute the Interface SQL COMPILE command. In response, the Interface automatically creates an Assembler program with the embedded SQL required by the FOCEXEC. It then precompiles, assembles, link-edits, and, optionally, binds the program. It accomplishes all steps, including the automatic BIND, from within FOCUS.

You must give some thought to the issue of plan management, discussed in the section below, *Plan Management in DB2*, as it will affect your choice of BIND option.

This section outlines the steps required to create a static FOCEXEC procedure, and provides a description of FOCUS processing for each step. An annotated example is included at the end of the discussion.

Creating a static FOCEXEC procedure for DB2 consists of the following steps:

1. Write the FOCEXEC Procedure

Most existing FOCEXECs containing TABLE FILE or MATCH FILE requests against DB2 tables may be used to generate static SQL. For restrictions see the section below entitled *Usage Restrictions*.

2. Allocate the Required DDNAMEs

The following chart lists the DDNAMEs you must allocate before compiling a FOCEXEC to use static SQL. You may add these allocations to your FOCUS CLIST, batch JCL, or PROFILE FOCEXEC. Note that these allocations are in addition to those normally required to run FOCUS with DB2:

DDNAME	DCB Parameters	Description
ASMSQL	DSORG(PO) RECFM(FB) LRECL(80)	Target dataset for the assembler source code generated by the compilation. The size of each member will vary depending on the number and size of the TABLE requests in the FOCEXEC.
DBRMLIB	DSORG(PO) RECFM(FB) LRECL(80)	Target dataset for the database request module generated by the compilation.
DB2LOAD	n/a	DB2 load library. The name for this library is site-specific, but usually follows the form 'DSNxy0.SDSNLOAD' (where 'xy0' is DB2 Version x Release y). The DB2 load library should be the same one used by the DB2 subsystem where the BIND will take place. NOTE: The STEPLIB allocation of the DB2 load library is still necessary.
STUBLIB	DSORG(PO) RECFM(U)	Contains the load module created for the procedure. This dataset is also required at run time.

SQLERR1	DSORG(PS) RECFM(FB) LRECL(130)	Contains the output of IBM's precompiler. DCB parameters are set by the precompiler. Output may be routed to the terminal using DA(*).
SQLERR2	DSORG(PS) RECFM(FM) LRECL(121)	Contains the output of IBM's assemble operation. DCB parameters are set by the assembler. Output may be routed to the terminal using DA(*).
SQLERR3	DSORG(PS) RECFM(FA) LRECL(81)	Contains the output of IBM's linkage editor. DCB parameters are set by the linkage editor. Output may be written to the terminal using DA(*).

**3. Optionally issue the SET STATIC Command**

The SET STATIC command need only be issued if you require the NOBIND option.

The syntax for the static SET command for DB2 is

```
SQL DB2 SET STATIC OFF/ON/NOBIND
```



where:

OFF/ON

Invokes static TABLE processing with automatic BIND when the SQL COMPILE command is issued. This option does not support extended plan management (see the section below, *Plan Management in DB2*), or modification of any of the default BIND parameters. OFF is the default. If you require more flexibility, use the NOBIND option.

**Notes:**

Before requesting an automatic BIND, make sure the DB2 subsystem is operational. In addition, the BIND operation itself requires DB2 privileges that you may not possess. Your DB2 database administrator can tell you if you are authorized to use BIND.

The Isolation Level is always CS if FOCUS automatically performs the BIND. To have an Isolation Level of RR, you must issue the BIND outside of FOCUS as described in the following section.

NOBIND

Invokes static processing without BIND when the SQL COMPILE command is issued. Use this option if you need extended plan management, to change the default BIND parameters, or to BIND your programs at another time for instance, during an off peak period).

## Notes:

- Omit the DB2 qualifier if you previously issued the SET SQLENGINE command for DB2.
- Batch considerations

You may create or run static FOCEXEC programs in batch. The DSN command processor cannot be invoked in batch; therefore, you must compile your programs using the NOBIND option and BIND the programs separately.

- FOCUS Multi-Session Option (MSO) considerations

You may create or run static FOCEXEC programs in MSO. The DSN command processor cannot be invoked from MSO; therefore, you must compile your programs using the NOBIND option and BIND the programs separately.

### 4. Optionally Issue the SET SSID Command

You must issue the SET SSID command (see Chapter 11 of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual*, DN1000048.1093) if you request the automatic BIND option and the DB2 subsystem in which you want the BIND to occur is different from your site's installation default. You can obtain the current setting for the DB2 subsystem by issuing the SQL DB2 ? command.

**Note:** The DB2 subsystem referenced by the SET SSID command should use the DB2 load library allocated to DDNAME DB2LOAD.

### 5. Compile the FOCEXEC

See the previous section entitled *Static TABLE Module Creation*.

FOCEXECs compiled with the Static SQL facility do not support the use of the LOAD command; to execute them you must use the SQL RUN command.

6. Optionally BIND the Plan for the FOCEXEC

This step is only necessary if the NOBIND option of the SET STATIC command was issued in step 3.

You may choose to BIND one application plan for each FOCEXEC (basic plan management) or to combine several FOCEXECs into one application plan (extended plan management). Plan management is discussed in the section below, *Plan Management in DB2*. Also use this option to supply BIND parameters (such as Isolation Level) other than the default.

Issue the BIND command outside the FOCUS environment, using one of the methods supplied by IBM. Your BIND may include one or more FOCEXEC DBRMs.

For example:

```
BIND PLAN (BIGPLAN) MEM(FEX1 FEX2 FEX3 FEX4) ACTION(ADD)
ISOLATION(CS)
```

**Note:** If the static module created is intended to function with IBM's Distributed Relational Database Architecture (DRDA), then a combination of the DB2 BIND PACKAGE and BIND PLAN commands must be used. For a full explanation of BIND methods and parameters, consult the *IBM DB2 Command and Utility Reference*.

7. Authorize Users to Run the Plan

Issue the SQL GRANT EXECUTE command to authorize users to execute the application plan created with the previous steps. This example shows how to issue the command from within a FOCUS session:

```
SQL DB2 GRANT EXECUTE ON PLAN BIGPLAN TO USER1, USER2
```

For more information on GRANT, consult the *IBM DB2 SQL Reference*.

## Run-Time Requirements

In addition to the allocations required for FOCUS and the Interface, outlined in Chapter 2 of the *FOCUS for IBM Mainframe DB2 and SQL/DS Users Manual*, DN1000048.1093, you must include allocations for the STUBLIB library, allocated to DDNAME STUBLIB.

If you use extended plan management (see the section below, *Plan Management in DB2*), you must issue the Interface SET PLAN command before running any of the procedures included in the DB2 application plan (see Section 12.9.1 of the above-mentioned manual):

```
SQL DB2 SET PLAN BIGPLAN
```

This setting overrides the plan for the FOCUS dynamic Interface. The section below, *Plan Management in DB2* explains options for resetting the plan at the conclusion of your FOCEXEC procedures.

You can use the SQL DB2 ? command to obtain all current Interface plan settings, for example:

SQL DB2 SET PLAN BIGPLAN

SQL DB2 ?

>

.  
.  
.

(FOC1448) ACTIVE PLAN FOR CALL ATTACH IS - :

(FOC1459) USER SET PLAN FOR CALL ATTACH IS - : BIGPLAN

(FOC1460) INSTALLATION DEFAULT PLAN IS - : P7009411

.  
.  
.

The active plan is set by the most recently issued request to DB2. If the thread is closed or marked inactive, there is no value for active plan. The "user set plan" is determined by your SET PLAN command.

## Processing and Security Overview

When you run a compiled FOCEXEC procedure, the STUBLIB load library is searched for a load module of the same name, which is then loaded. FOCUS then connects to DB2 and opens a thread to the application plan of the same name as the FOCEXEC procedure (you may use the SET PLAN command to override this). DB2 compares the owner, program name and timestamps in the STUBLIB member and the DB2 application plan for consistency. This process verifies that the program load module (STUBLIB member) and the DB2 application plan are valid and that no substitutions have been made.

## DB2 Static FOCEXEC Example

The following annotated example illustrates the creation and automatic bind of a static FOCEXEC procedure named FEX1. The numbers 1 through 6 refer to steps in the process that are described following the example. The Master and Access File Descriptions are identical to those used in Appendix A of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual, Release 6.8, DN1000048.1093*

This example uses a multi-table Master and Access File Description, EMPLOYEE, that relates the two tables PAYROLL and COURSES.

The Master File Description is:

```
FILENAME=EMPLOYEE, SUFFIX=SQLDS,$
SEGNAME=PAYROLL, SEGTYPE=S0,$
FIELD=EMP_ID, EMP_ID, A5, A5, MISSING=OFF,$
FIELD=SALARY, SALARY, P13.2 , P6, MISSING=OFF,$
SEGNAME=COURSES, SEGTYPE=S0,$
FIELD=EMP_NUM, EMP_NUM, A5, A5, MISSING=OFF,$
FIELD=COURSE_NUM, COURSE_NUM, A5, A5, MISSING=OFF,$
FIELD=POINTS, POINTS, I3, I2, MISSING=OFF,$
```

The Access File Description is:

```
SEGNAME=PAYROLL, TABLENAME="USER1"."PAYROLL",
KEYS=1, WRITE=YES, KEYORDER=LOW,$
SEGNAME=COURSES, TABLENAME="USER1"."COURSES",
KEYS=2, WRITE=YES, KEYORDER=LOW,
KEYFLD=EMP_ID, IXFLD=EMP_NUM,$
```

The FEX1 FOCEXEC reads as follows:

```
SQL DB2 SET OPTIMIZATION ON
TABLE FILE EMPLOYEE
PRINT EMP_ID SALARY POINTS
WHERE SALARY GE &SALARY
END
```

Also included for your information are the options FOCUS uses for each step (for instance, Assembler options). Please refer to the appropriate IBM manual for an explanation of these parameters and their possible settings.

```
SQL DB2 SET SSID DB2P
```

```
1. SQL DB2 COMPILE FEX1 SALARY=50000
   END
2. (FOC1472) STATIC SQL PROGRAM CREATED SUCCESSFULLY
3. (FOC1473) STATIC SQL PROGRAM PREPROCESSED. RETURN CODE IS: 4
4. (FOC1474) STATIC SQL PROGRAM ASSEMBLED. RETURN CODE IS : 0
5. (FOC1476) STATIC SQL PROGRAM LINKED. RETURN CODE IS : 0
   DSN
6. BIND PLAN(FEX1) MEM(FEX1) ACTION(REPLACE) ISOLATION(CS)
   DSNT252I - BIND OPTIONS FOR PLAN FEX1
           ACTION      ADD
           OWNER       USER1
           VALIDATE    RUN
           ISOLATION   CS
           ACQUIRE    USE
           RELEASE     COMMIT
           EXPLAIN     NO
   DSNT253I - BIND OPTIONS FOR PLAN FEX1
           NODEFER     PREPARE
           CACHESIZE   1024
           QUALIFIER   USER1
           CURRENTSERVER
           CURRENTDATA NO
           DEGREE      1
           SQLRULES    DB2
           DISCONNECT  EXPLICIT
   DSNT200I - BIND FOR PLAN FEX1      SUCCESSFUL
   DSN
   END
```

The steps in the process are:

1. Issuance of the SQL COMPILE command.



This command invokes the Static SQL for TABLE facility. Because the &SALARY variable in the FOCEXEC is located to the right of a comparison operator in a screening condition, the facility will substitute an SQL host variable in that location if a literal is specified for the variable value. This means that the specification of the value 50000 for the &SALARY variable is merely a placeholder for the duration of the SQL COMPILE. You will need to specify a value at SQL RUN time.

**2. Creation of the Assembler program.**

In this step, FOCUS reads the FOCEXEC procedure and creates an Assembler program with embedded SQL statements representing all SQL statements the FOCEXEC will execute. This program is the input file for the next step.

**3. Precompilation of the Assembler program.**

FOCUS invokes the IBM precompiler for the Assembler program created in Step 1. The precompiler comments out the embedded SQL statements and replaces them with Assembler calls to DB2. It places the SQL statements in a Database Request Module (DBRM) created as a member in the dataset allocated to DBRMLIB. This member is part of the input to the BIND process. FOCUS places the modified source program into a temporary dataset. The IBM precompiler listing is placed in the dataset allocated to DDNAME SQLERR1 or written to the terminal.

This step uses the Assembler precompiler included in the DB2 load library allocated to DDNAME STEPLIB. The Interface SET SSID command has no influence on the precompiler used. If you have more than one DB2 subsystem, be sure that the DB2 load library used in the precompilation step is the same one used by the DB2 subsystem in which the program will run. The DB2 subsystem does not have to be active during this step.

The precompiler options for this step are HOST(ASM),DATE(ISO).

**Note:** The precompilation process generates the statement "WARNINGS HAVE BEEN SUPPRESSED DUE TO LACK OF TABLE DECLARATIONS" in the Assembler program. This message is normal and may be ignored. The precompilation should complete with a return code of four (4).

**4.** Assembly of the program.

FOCUS assembles the modified source program using IBM's Assembler H and places the output into a temporary dataset. The resulting listing is written to the dataset allocated to DDNAME SQLERR2 or to the terminal.

The assembler options are DECK, NOOBJECT, NOALIGN and TERM. This step should complete with a return code of zero (0).

**5.** Linkedit of the program.

FOCUS links the assembled source program using IBM's linkage editor, placing the run-time module into the dataset allocated to DDNAME STUBLIB. Linkage editor output is written to the dataset allocated to DDNAME SQLERR3 or to the terminal.

The linkage editor options are TERM, RENT, AMODE(31), RMODE(ANY), and LIST. This step should complete with a return code of zero (0).

**6.** Optional BIND.

Since the SET STATIC OFF (the default) or ON command requests an automatic BIND, FOCUS invokes the DB2 DSN command processor and submits the following bind request:

```
DSN SYSTEM(DB2P)
BIND PLAN (FEX1) MEM(FEX1) ACTION(REPLACE) ISOLATION(CS)
```

**Notes:**

- The SSID of the DB2 subsystem in which this plan will be bound is DB2P. The target subsystem for the BIND was established by the SET SSID command.
- The plan name will be the same as that of the FOCEXEC (FEX1). The MEM parameter means that application plan FEX1 will include member FEX1 from the dataset allocated to DDNAME DBRMLIB. Member FEX1 was created during the precompilation step. The plan owner is the DB2 authorization id in effect at the time of the bind.
- FOCEXECs compiled with the Static SQL for TABLE facility do not support the use of the LOAD command; to execute them you must use the SQL DB2 RUN command.

## **Plan Management in DB2**

Through extended plan management, users can create application plans that contain multiple FOCEXEC procedures. This section discusses both the basic and extended plan management options.

## Basic Plan Management

With basic plan management, there is a separate DB2 application plan for every FOCEXEC procedure. You do not have to issue the SET PLAN command, and the Interface automatically establishes and terminates the necessary DB2 threads for each procedure invoked.

The following example assumes that FEX1 and FEX2 are static FOCEXEC procedures and that a corresponding DB2 application plan exists for each of them:

```
SQL DB2 RUN FEX1
END
SQL DB2 RUN FEX2
END
SQL DB2 RUN FEX2
END
EX RPT1
```

The Interface takes the following actions:

1. First, it closes the thread to a prior application plan, if one exists. It deallocates that plan with a Call Attachment Facility (CAF) CLOSE. Then it opens a thread to plan FEX1. If no prior connection existed, it first issues a CAF CONNECT. It establishes the thread to FEX1 with a CAF OPEN.
2. It opens a thread to plan FEX2. First it deallocates the thread to plan FEX1 with a CAF CLOSE, then issues a CAF OPEN to establish the thread to FEX2.

3. No action. The Interface recalls that it already has a thread to plan FEX2 and takes no action.
4. It opens a thread to the FOCUS dynamic plan. First it deallocates the thread to plan FEX2 with a CAF CLOSE. Then it issues a CAF OPEN to establish the thread to the installation default application plan for the dynamic SQL Interface.

**Note:** The settings for AUTOCLOSE and AUTODISCONNECT affect whether a thread is retained across invocations of the same FOCEXEC, as in items 2 and 3. Some settings require that a thread and/or connection to DB2 be severed, either at the conclusion of any FOCEXEC procedure, or within the procedure itself. More information is available in Chapter 10 of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual, Release 6.8*, DN1000048.1093.

## Extended Plan Management

With extended plan management, you can create a DB2 application plan that contains more than one FOCEXEC procedure. You must issue the SET PLAN command to establish an umbrella plan that includes each FOCEXEC to be invoked.

The primary purpose of extended plan management is to limit the amount of overhead involved in plan switching; therefore, use of this method is not consistent with the AUTOCLOSE or AUTODISCONNECT options, both of which terminate threads and/or connections to DB2.

The following example assumes that FEX1 and FEX2 have been bound into a plan called BIGPLAN:

```
SQL DB2 SET PLAN BIGPLAN
SQL DB2 RUN FEX1
END
SQL DB2 RUN FEX2
END
SQL DB2 RUN FEX2
END
SQL DB2 SET PLAN
EX RPT1
```

The Interface takes the following actions:

1. It sets the application plan. The SET PLAN command tells the Interface that all required SQL statements for subsequent FOCEXECs are contained in BIGPLAN.
2. It opens a thread to plan BIGPLAN. It establishes the thread to BIGPLAN with a CAF OPEN. If there is no prior connection to DB2, it executes a CONNECT before the OPEN.
3. No action. The Interface recalls that the user-set plan is BIGPLAN and that the thread has already been established.
4. No action. The Interface recalls that the user-set plan is BIGPLAN and that the thread has already been established.
5. It returns control to the dynamic Interface. Setting the plan to blank informs the Interface that you wish to return control to the installation default application plan. If you wish to return control to a plan other than the default plan, specify that plan name.

Another option would be to include the dynamic Interface in BIGPLAN by including the DBRM for RSQL as part of the member list when binding BIGPLAN. This would eliminate the need to reset the plan for dynamic access to DB2. If using this option, skip steps 5 and 6.

1. It opens a thread to the FOCUS dynamic plan. It executes a CAF OPEN to establish the thread to the installation default application plan.

## Static FOCEXEC Creation for SQL/DS

In order to use Static SQL for TABLE in the SQL/DS environment, you need only execute the Interface SQL COMPILE command. In response, the Interface automatically creates an Assembler program with the embedded SQL required by the FOCEXEC procedure. Complete the process by exiting FOCUS and running an Information Builders-supplied EXEC that preprocesses, assembles, and links the program and creates an application package in the SQL/DS database.

This section outlines the steps required to create a static FOCEXEC procedure, and provides a description of FOCUS processing for each step. An annotated example is included at the end of the discussion.

Execute all steps in this procedure after logging on to a userid authorized to connect to SQL/DS and after meeting the run-time requirements documented in Chapter 2 of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual*, DN1000048.1093.

Creating a static FOCEXEC procedure for SQL/DS consists of the following steps:

1. Write the FOCEXEC Procedure

Virtually any FOCEXEC containing TABLE FILE or MATCH FILE requests against DB2 tables may be used to generate static SQL. For restrictions, see the section below, *Usage Restrictions*. The Isolation Level is Cursor Stability (CS) and cannot be changed.

2. Compile the FOCEXEC

See the previous section entitled *Static TABLE Module Creation*. FOCEXECS compiled with the Static SQL facility do not support the use of the LOAD command; to execute them you must use the SQL RUN command.

3. Exit FOCUS

Issue the FIN command to exit from FOCUS before running the GENUSQL EXEC.

4. Execute the GENUSQL EXEC

The GENUSQL EXEC preprocesses, assembles, links, and creates an application package for your FOCEXEC procedure. GENUSQL requires input parameters for execution; they are set to default values within the EXEC. To change the defaults at run time, provide overriding values on the command line in the form PARM=value. Use blanks as delimiters between parameter values; do not insert spaces on either side of the assignment symbol (=).

The syntax is:

```
EX GENUSQL program (parm=value parm=value DBLIST (db1 db2...dbn)
```

where:



`program` Is the name of the program to be prepared. It will have the same name as the FOCEXEC.

`parm` Is one of the following:

#### SQLUID

Is the SQL/DS userid to be the creator of the package. Your VM logon id is the default.

#### SQLPSWD

Is the SQL/DS password associated with SQLUID. This is not the VM logon password. No password is required for the VM userid default.

#### COLLID

Is the collection-id, which will be used as the qualifier for the SQL/DS application package that is produced by the SQLPREP EXEC. If COLLID is not specified, it defaults to the value of SQLUID (if specified), or the current VM userid (if SQLUID is not specified).

#### GRANT

Automatically grants run privilege on the package to PUBLIC. Possible values are YES or NO. NO is the default.

### EXPLAIN

Causes the user's EXPLAIN tables to be updated by the SQL/DS optimizer with access path information.

Possible values are NO, the default, or YES (to update the EXPLAIN tables).

### INDVAR

Removes null indicator variables in SQL predicates in the ASMSQL program used as input to GENUSQL. The existence of these variables may cause a performance degradation in applications involving large SQL/DS tables. Possible values are YES, the default, or NO (to remove the variables).

`db1 . . . dbn`

Is a list of databases. You can specify multiple databases as targets for storage of the application package. The default is the database specified in the most recently issued SQLINIT command.

### Notes:

- The Isolation Level is Cursor Stability (CS) and cannot be changed.
- SQL/DS must be running when GENUSQL is invoked.

### 5. Authorize Users to Run the FOCEXEC

If you ran GENUSQL with GRANT=NO (the default), issue the SQL GRANT EXECUTE command to authorize users to execute the packages created with the previous steps. The following example shows how to issue the command from within a FOCUS session:

```
SQL SQLDS GRANT EXECUTE ON PLAN FEX1 TO USER1, USER2
```

For more information on GRANT, consult the *IBM SQL/DS SQL Reference*.

## Run-Time Requirements

The STUBLIB LOADLIB containing the run-time modules must reside on a minidisk the user has accessed. It is not necessary to GLOBAL the STUBLIB LOADLIB, as the FOCUS EXEC automatically does this. However, the user must access the minidisk before invoking FOCUS.

## Processing and Security Overview

When you run a compiled FOCEXEC procedure, FOCUS searches for a module of the same name in the STUBLIB LOADLIB file. FOCUS then invokes the static SQL procedure. The user is connected to SQL/DS and the package (which always has the same name as the FOCEXEC) is loaded. SQL/DS verifies that the STUBLIB load module for the program and the SQL/DS package have the same owner, program name, and consistency token.

This process verifies that the program load module (STUBLIB LOADLIB member), and the SQL/DS application package are valid and that no substitutions have been made.

**Note:** There is no equivalent in SQL/DS for the plan management techniques available for DB2. There must be a separate package for every static procedure, and one for the dynamic SQL/DS Interface. SQL/DS "switches" packages automatically in response to program execution.

## SQL/DS Static FOCEXEC Example

The following annotated example illustrates the creation of a static FOCEXEC procedure named FEX1. The numbers 1 through 9 refer to steps in the process that are described following the example.

The Master and Access File Descriptions are identical to those used in Appendix A of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual, Release 6.8, DN1000048.1093*)

This example uses a multi-table Master and Access File Description, EMPLOYEE, that relates the two tables PAYROLL and COURSES.

The Master File Description is:

```
FILENAME=EMPLOYEE, SUFFIX=SQLDS,$
SEGNAME=PAYROLL, SEGTYPE=S0,$
FIELD=EMP_ID, EMP_ID, A5, A5, MISSING=OFF,$
FIELD=SALARY, SALARY, P13.2 , P6, MISSING=OFF,$
SEGNAME=COURSES, SEGTYPE=S0,$
FIELD=EMP_NUM, EMP_NUM, A5, A5, MISSING=OFF,$
FIELD=COURSE_NUM, COURSE_NUM, A5, A5, MISSING=OFF,$
FIELD=POINTS, POINTS, I3, I2, MISSING=OFF,$
```

The Access File Description is:

```
SEGNAME=PAYROLL, TABLENAME="USER1"."PAYROLL",  
KEYS=1, WRITE=YES, KEYORDER=LOW,$  
SEGNAME=COURSES, TABLENAME="USER1"."COURSES",  
KEYS=2, WRITE=YES, KEYORDER=LOW,  
KEYFLD=EMP_ID, IXFLD=EMP_NUM,$
```

The FEX1 FOCEXEC reads as follows:

```
SQL SQLDS SET OPTIMIZATION ON  
TABLE FILE EMPLOYEE  
PRINT EMP_ID SALARY POINTS  
WHERE SALARY GE &SALARY  
END
```

Also included for your information are the options FOCUS uses for each step (for instance, assembler options). Please refer to the appropriate IBM manual for an explanation of these options and their possible settings.

```
1.SQL SQLDS COMPILE FEX1 SALARY=50000
  END
2.(FOC1472) STATIC SQL PROGRAM CREATED SUCCESSFULLY
3.FIN
  Ready;
4.genusql fex1 (sqluid=user1 sqlpswd=myword grant=yes
  Following installation parameters assumed:
      Pgm      = FEX1
      Sqluid   = USER1
      Sqlpswd  = MYWORD
      Grant    = YES
      Indvar   = YES
      Collid   =
5.ARI0717I START SQLPREP EXEC: 11/07/94 15:22:25 EST
  ARI0320I THE DEFAULT DATABASE NAME IS SQLDBA.
  ARI0663I FILEDEFS IN EFFECT ARE:
  SYSIN   DISK   FEX1   ASMSQL   *
  SYSPRINT DISK   FEX1   LISTPREP A1
  SYSPUNCH DISK   FEX1   ASSEMBLE A1
  ARISQLLD DISK   ARISQLLD LOADLIB G1
  ARI0713I PREPROCESSOR ARIPRPA CALLED WITH THE FOLLOWING PARAMETERS:
  .....PREP=FEX1,USER=USER1/*****,,ISOL(USER),BLK,NOPR,DATE(ISO)
  ARI0708I ALL SQLPREP EXEC PROCESSING COMPLETED SUCCESSFULLY.
  ARI0796I END SQLPREP EXEC: 11/07/94 15:22:28 EST
6.Assembling of FEX1 completed successfully.
7.ARI0717I START SQLDBSU EXEC: 11/07/94 16:21:00 EST
  ARI0659I LINE-EDIT SYMBOLS RESET:
      LINEND=# LINEDEL=OFF CHARDEL=OFF ESCAPE=OFF TABCHAR=OFF
  ARI0656I MESSAGE FILE (SYSPRINT): TERMINAL
  ARI0655I INPUT FILE (SYSIN): TERMINAL
  ARI0320I THE DEFAULT DATABASE NAME IS SQLDBA.
  ARI0663I FILEDEFS IN EFFECT ARE:
  ARISQLLD DISK   ARISQLLD LOADLIB G1
```

```
SYSIN      TERMINAL
SYSPRINT  TERMINAL
ARI0801I  DBS Utility started: 11/07/94 16:21:05.
          AUTOCOMMIT = OFF ERRORMODE = OFF
          ISOLATION LEVEL = REPEATABLE READ
ARI0828I  ...LINEWIDTH reset to 80.
ARI0827I  ...Begin command execution: ERRORMODE = CONTINUE
ARI0870I  Enter the command terminated by semicolon or enter
          EXIT to end.
----->
-----> CONNECT "USER1      " IDENTIFIED BY *****;
ARI8004I  User USER1 connected to server SQLDBA.
ARI0500I  SQL processing was successful.
ARI0505I  SQLCODE = 0 SQLSTATE = 00000 ROWCOUNT = 0

ARI0870I  Enter the command terminated by semicolon or enter
          EXIT to end.
-----> GRANT RUN ON USER1.FEX1 TO PUBLIC;
ARI0500I  SQL processing was successful.
```

```
ARI0505I  SQLCODE = 0  SQLSTATE = 00000  ROWCOUNT = 0

ARI0870I  Enter the command terminated by semicolon or enter
          EXIT to end.
-----> COMMIT WORK;
ARI0500I  SQL processing was successful.
ARI0505I  SQLCODE = 0  SQLSTATE = 00000  ROWCOUNT = 0

ARI0870I  Enter the command terminated by semicolon or enter
          EXIT to end.
-----> EXIT
ARI8997I  ...Begin COMMIT processing.
ARI0811I  ...COMMIT of any database changes successful.
ARI0809I  ...No errors occurred during command processing.
ARI0808I  DBS processing completed: 11/07/94 16:23:39.
ARI0660I  LINE-EDIT SYMBOLS RESTORED:
          LINEND=#  LINEDEL=¢  CHARDEL=@  ESCAPE="  TABCHAR=ON
ARI0796I  END SQLDBSU EXEC: 11/07/94 16:23:39 EST
8.DMSCPY721I Copy ARIRVSTC TEXT G1 APPEND FEX1 TEXT A1 (old file)
  IEW0000      INCLUDE OBJ
  IEW0000      ENTRY ADDRESS
9.IEW0000      NAME FEX1(R)
```

Ready;

The steps in the process are:

1. Issuance of the SQL COMPILE command.



This command invokes the Static SQL for TABLE facility. Because the &SALARY variable in the FOCEXEC is located to the right of a comparison operator in a screening condition, the facility will substitute an SQL host variable in that location if a literal is specified for the variable value. This means that the specification of the value 50000 for the &SALARY variable is merely a placeholder for the duration of the SQL COMPILE. You will need to specify a value at SQL RUN time.

**2. FOCUS creates the Assembler program.**

In this step, FOCUS reads the FOCEXEC program and creates an Assembler program with all the embedded SQL statements required by the FOCEXEC.

The Assembler program has the same filename as the FOCEXEC and a filetype of ASMSQL. It is placed on the minidisk accessed as filemode A. In this example, the Assembler program is called FEX1 ASMSQL A.

**3. The user exits FOCUS.**

**4. The user invokes the GENUSQL EXEC.**

**5. SQL/DS prepares the Assembler program.**

The GENUSQL EXEC invokes the SQLPREP EXEC supplied by IBM on the SQL/DS production disk. SQLPREP precompiles the Assembler program (FEX1 ASMSQL A) and creates a package for it in the SQL/DS database. The package naming convention is 'owner.program\_name', where the owner is the COLLID (if one was provided), SQLUID (if one was provided and COLLID was not provided), or the VM userid invoking GENUSQL (if neither COLLID nor SQLUID was provided). In this example, the package would be named USER1.FEX1.

The SQLPREP EXEC invokes the IBM preprocessor for Assembler programs (ARIPRPA, residing on the SQL/DS production disk) with the following parameters, provided by GENUSQL: ISOL(USER), BLK, NOPR, DATE(ISO). If the user provides the SQLUID and SQLPSWD parameters, USER=SQLUID/SQLPSWD. Output from the preprocessor is FEX1 LISTPREP A (the preprocessor output listing) and FEX1 ASSEMBLE A (the modified source program). The LISTPREP file contains a summary count of errors and warnings generated by the preprocessor, while the actual text of errors and warnings is placed in the modified source program. Successful completion of this step is indicated by the ARI0708I and ARI0796I messages generated by SQL/DS.

**6.** GENUSQL assembles the program.

The GENUSQL EXEC assembles the modified source program (FEX1 ASSEMBLE A) using IBM's Assembler H, if available. If Assembler H is not found, the installed version is used. This results in object FEX1 TEXT A (a temporary file) and FEX1 LISTING A (the output from the assembly). The Assembler options are LIST and NOALIGN.

Successful completion of this step is illustrated by the message displayed at item 6 in the example.

**7.** SQL/DS grants the RUN privilege on the package.

Since GENUSQL was invoked with GRANT=YES, the SQL/DS Database Services Utility program (DBSU) is started. It grants run privileges on USER1.FEX1 to PUBLIC (all users) and issues a COMMIT WORK.

Successful completion of this step is indicated by the ARI0811I and ARI0809I messages generated by SQL/DS.

8. GENUSQL adds the Resource Manager stub.

The GENUSQL EXEC appends the Resource Manager Stub (ARIRVSTC TEXT, residing on the SQL/DS production disk) to FEX1 TEXT A.

A non-zero return code for this step results in an error message.

9. GENUSQL link-edits the program.

The GENUSQL EXEC links FEX1 TEXT A into STUBLIB LOADLIB A. FEX1 TEXT A is then erased. The linkage editor options are LIST, MAP, XREF, and NCAL.

A non-zero return code for this step results in an error message.

## Verification and Control of Host Variable Placement

As discussed in the section above, *SQL COMPILE and SQL RUN Processing*, the Static SQL for TABLE facility follows certain rules to determine where SQL host variables should be inserted in the generated SQL to be executed at SQL RUN time. From time to time, you may wish to verify where these variables have been placed in the SQL. This may be accomplished by activating FSTRACE4 (see Appendix A of the *FOCUS for IBM Mainframe DB2 and SQL/DS Users Manual*, DN1000048.1093 for information on activating Interface traces).

For example, assume that the FEX1 FOCEXEC used in the previous illustrations was executed as follows:

```
SQL DB2 RUN FEX1 SALARY = 50000  
END
```

This would produce the following FSTRACE4 output:

```
SELECT T1.EMP_ID,T1.SALARY,T2.POINTS FROM "USER1"."PAYROLL" T1,  
"USER1"."COURSES" T2 WHERE (T2.EMP_NUM = T1.EMP_ID) AND  
(T1.SALARY >= :H) ORDER BY T1.EMP_ID;
```

Note the :H to the right of the comparison operator >= in the screening condition on T1.SALARY. This is an SQL host variable for which the value 50000 was substituted at run time. The presence of this variable indicates that a value may be substituted at run time, provided a FOCUS ampersand variable is present in the corresponding location in the TABLE request (for example, &SALARY in FEX1).

At SQL COMPILE time, SQL host variables will be substituted for all literals to the right of logical operators in FOCUS IF and WHERE screening conditions, in order to allow for the substitution of ampersand variable values at SQL RUN time. This substitution will take place even if the operand is not an ampersand variable. In other words, even if the literal 50000 had been explicitly coded in the example FOCEXEC above, the :H host variable would still have appeared in the generated SQL. This may have RDBMS optimization implications, especially in the case of SQL range predicates (for example, < or >), because the RDBMS optimizer will have to make certain assumptions concerning data distribution when choosing the access path to the data. If you are concerned about RDBMS optimization in the DB2 environment, execute the SQL COMPILE command with the STATIC option set to NOBIND, and execute a separate BIND using the EXPLAIN option. For more information about BIND options, consult the *IBM DB2 Command and Utility Reference*. In the SQL/DS environment, the EXPLAIN option of the GENUSQL EXEC may be used. For more information about this option, consult the explanation of the SQLPREP command in the *IBM SQL/DS Application Programming for VM Systems* manual.

In many cases, it is possible to use the FOCUS DEFINE facility to cause literal values to be explicitly coded into the generated SQL. For example, suppose the above example had the value 50000 explicitly coded in the request, but, since the 50000 value was unknown to the DB2 optimizer at BIND time (due to the substitution of the host variable :H), a table space scan access path was selected, even though the SALARY column was indexed, and only a few rows would be retrieved when the value 50000 was substituted for :H at SQL RUN time. Since it is known by the user at SQL COMPILE time that the screening value for this query will always be 50000, the request could be altered as follows:

```
SQL DB2 SET OPTIMIZATION ON
DEFINE FILE EMPLOYEE
SALVALUE/P13.2 = 50000;
TABLE FILE EMPLOYEE
PRINT EMP_ID SALARY POINTS
WHERE SALARY GE SALVALUE
END
```

The generated SQL would now read as follows:

```
SELECT T1.EMP_ID,T1.SALARY,T2.POINTS FROM "USER1"."PAYROLL" T1,
"USER1"."COURSES" T2 WHERE (T2.EMP_NUM = T1.EMP_ID) AND
(T1.SALARY >= 50000) ORDER BY T1.EMP_ID;
```

The DB2 optimizer would now be able to choose a direct index access path to the data because the literal screening value would be known at BIND time.

**Note:** This type of solution will not be possible for literals in LIKE predicates, as FOCUS does not allow field names to be used to the right of the LIKE operator in TABLE requests. Also, this technique will cause dual range predicates to be generated for FOCUS WHERE/FROM-TO clauses, as opposed to an SQL BETWEEN. The use of dual range predicates may have optimization implications.

## Usage Restrictions

- If the FOCEXEC being compiled contains MATCH FILE commands, then the INTERPRET option of the SQL COMPILE command must be used.
- Because of the SQL generation and comparison processing that takes place at SQL RUN time, it is extremely important that the Interface OPTIMIZATION setting be the same at SQL RUN time as at SQL COMPILE time. For this reason, it is strongly suggested that the appropriate SET OPTIMIZATION command(s) be explicitly specified in any FOCEXECs being processed, even if the default setting of ON is desired. This will ensure that the necessary settings are in effect, even if the overall session setting has somehow been altered.

- In TABLE requests that specify the MISSING value for screening conditions, the generated SQL will contain the NULL value, not a host variable. This will also occur if you type in the word MISSING for an ampersand variable at SQL COMPILE time. For this reason, if it is possible that either MISSING or literal values may be specified for this variable at SQL RUN time, then SQL COMPILE should be invoked with the REPEAT option, and both MISSING and another literal placeholder should be specified, in separate iterations. This will cause two SQL statements to be generated: one with the NULL value, and one with a host variable to accept any literals specified at SQL RUN time. If this is not done, a FOC1526 may result at SQL RUN time due to the lack of a matching static SQL statement.
- If the FOCEXEC being compiled calls other FOCEXECs, these other FOCEXECs will not be compiled. If you wish to run them statically, they must be compiled separately, and invoked with an SQL RUN command from the calling FOCEXEC at SQL RUN time.
- If any TABLE requests in the FOCEXEC contain IF or WHERE screening conditions that use the dname option, then the number of entries in the file being used must be the same at SQL RUN time as at SQL COMPILE time. If this is not the case, a FOC1526 message will be issued, and execution will terminate. This restriction may be bypassed, where practical, by the use of the REPEAT option, changing the size of the file with each iteration. Again, however, the number of entries at SQL RUN time must match one of the previously generated SQL statements exactly. This restriction may be lifted in a future release.

**Note:** Due to this restriction, it is strongly recommended that the AUTODB2 and AUTOSQL FOCXECs not be compiled for static SQL. Both of these facilities make extensive use of the IF/WHERE EQ (ddname) feature of FOCUS.

- When accessing multi-segment structures (whether created via the JOIN command or a multi-segment Master and Access File Description), the data types of any join fields must be equal. Also, scales of any packed decimal (P) fields being used as join fields must be equal. The scale is the length of the portion of the value to the right of the decimal point, specified in the USAGE attribute in the Master File Description. In addition, the USAGE lengths of any alphanumeric (A) join fields must be equal.
- In TABLE requests containing HOLD FORMAT DB2 or HOLD FORMAT SQL subcommands, no static SQL statements will be generated by these subcommands. They will continue to run dynamically, even when the FOCXEC is executed in the SQL RUN environment. The data query (SELECT) portion of the request will, however, generate static SQL statements, if applicable. If necessary, the application may be altered to generate a flat file and invoke a separate MODIFY that uses static SQL; however the CREATE FILE command would still need to be issued separately, and it must always be dynamic.



- If any TABLE requests contain screening conditions which use the LIKE comparison operator, and an ampersand variable is used in place of the literal to the right of the LIKE, bear in mind that different literals used with the same LIKE syntax may produce different SQL. In this case, you must ensure that any SQL which could be generated dynamically at SQL RUN time has matching statements in the STUBLIB module (see the section above entitled *SQL COMPILE and SQL RUN Processing*). This may be done using the REPEAT option of the SQL COMPILE facility, substituting a different LIKE literal format for each execution. For more information on the SQL generated by the LIKE operator, see section 7.2.1 of the *FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual*, 6.8, DN1000048.1093. If, at SQL RUN time, a static SQL statement cannot be located that matches the generated dynamic SQL, a FOC1526 message will be issued, and execution will terminate.

## Resource Restrictions

Static compilation creates an Assembler routine that includes SQL statements. Each SQL statement uses a certain number of Assembler resource units (base registers). There is a finite limit to the number of base registers that may be used. This might, in some cases, prohibit a large FOCEXEC from compiling successfully. These cases, however, should be extremely rare, and should only involve extremely large generated SQL SELECT statements. The vast majority of applications will be unaffected. These limitations are not unique to FOCUS; they apply to any Assembler program with embedded SQL.

The Interface uses an optimization algorithm to allocate resources from the available pool of 16 registers. If it determines that a limit has been exceeded, it issues an error message during static compilation. Therefore, if your FOCEXEC already exists, the easiest way to determine if it exceeds any limits is to statically compile it.

If the Interface determines that a base register limit has been exceeded during the creation of the Assembler program, the FOC1355 and FOC1359 error messages will be issued, and compilation will terminate. There are cases, however, when it can not be determined that any limits have been exceeded until the program is actually assembled. These errors will be flagged as addressability errors during the assembly, and may be found in the Assembler listing file that is generated.

## NF504: Installation Enhancement for the SQL/DS Interface

Starting with FOCUS Release 7.0, an additional parameter has been added to the GENFSQL EXEC, the EXEC which is used to generate the application package for the FOCUS-SQL/DS Interface. This parameter, called COLLID, is used to pass a value to the collection-id parameter of the SQL/DS SQLPREP EXEC, allowing you to specify the value that will be used as the qualifier for the application package. The value can be different from the VM userid, or the value specified for the SQLUID parameter, if one exists. This capability often proves useful in environments which make use of IBM's Distributed Relational Database Architecture (DRDA), as the passed value is used for the collection-id at any remote locations.

### Syntax      How to Specify the Value for COLLID

The syntax for specifying the value for COLLID is

```
EX GENFSQL (SQLUID=USER1 SQLPSWD=PASS1 COLLID=USER2
```

where:

`SQLUID`

Is the SQL/DS authorization id for the SQLPREP EXEC. This is the same as in prior releases of FOCUS. If COLLID is not specified, this is also the qualifier for the application package that is produced. If SQLUID and COLLID are not specified, the authorization id and package qualifier default to the current VM userid.

SQLPSWD	Is the SQL/DS authorization password for the SQLPREP EXEC. This is the same as in prior releases of FOCUS. If SQLUID is specified, SQLPSWD must also be specified.
COLLID	Is the collection-id, which is used as the qualifier for the SQL/DS application package that is produced by the SQLPREP EXEC. If COLLID is not specified, the default is the value of SQLUID, if specified, or the current VM userid if SQLUID is not specified.

For more information on the SQLPREP EXEC in general, and the collection-id parameter in particular, consult the *IBM SQL/DS Application Programming for VM Systems* manual.

## NF505: Optimization Enhancements

FOCUS Release 7.0 introduces a number of optimization enhancements to the Relational Interfaces. The Interfaces can now pass:

- Most joins to the RDBMS's that have been processed by FOCUS in prior releases due to the presence of the multiplicative effect or certain occurrences of the FOCUS FST. and LST. operators
- Certain occurrences of FOCUS FST. and LST. aggregation operators (both implicitly and explicitly specified) to the RDBMS as MIN and MAX, respectively.
- DEFINE-based screening conditions that pertain to single segments to the RDBMS, even in the event that a join specified in the request can not be passed.
- To the RDBMS a FOCUS WHERE clause which uses the MISSING operator. It is passed as an SQL WHERE clause containing the NULL keyword in the same manner that IF conditions containing the same operator have been passed in previous releases.
- A FOCUS WHERE clause containing the FROM/TO operator to the RDBMS as an SQL WHERE clause containing the BETWEEN operator. This functionality is available beginning with FOCUS Release 6.8 Put Level 9306 and continuing with Release 7.0.

In addition, FOCUS Release 7.0 includes optimization enhancements which apply solely to the Oracle Interface:

- The FOCUS IF or WHERE READLIMIT EQ n clause is passed to the Oracle RDBMS as an SQL WHERE ROWNUM = n clause.

- Oracle array blocking for improved performance is enabled through the Interface for any SELECT and INSERT operations, whether they are issued by FOCUS (such as from a TABLE request) or directly by the user (such as from a Direct SQL Passthru request). Oracle array blocking for SELECT statements is available starting with FOCUS Release 6.8 Put Level 9312. For INSERT statements, it is enabled starting with FOCUS Release 7.0.

## Interface-Managed Native Join Optimization

The Relational Interfaces are able to pass most joins to the RDBMS's that were managed by FOCUS in prior releases, due either to the detection of the multiplicative effect by the Interfaces, or because of multiple FST. or LST. operators on fields in a segment which can possibly have multiple records per sort break in the request. For more information on the multiplicative effect, see *Appendix A of the FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual, Release 6.8., DN1000048.1093*. For more information on FST. and LST. and multiple segment instances, see the section below on *Selective Translation of FST. and LST. Aggregation Operators*.

Prior to Release 7.0, if the multiplicative effect was detected for a request, you had two options. First, in environments created by issuing SET OPTIMIZATION FOCUS or SET OPTIMIZATION ON (ON is the default), the Interface temporarily disabled optimization, passing individual SQL statements to the RDBMS for each table (segment, in FOCUS terminology) in the join structure, and FOCUS corrected the segment multiplication prior to displaying the report. The second option was used if the situation demanded that the join be passed to the RDBMS regardless of the multiplicative effect, such as for performance reasons. A SET OPTIMIZATION SQL command was issued; however, any segment multiplication was not subsequently corrected by FOCUS, and resulting reports may have contained deceptive or incorrect output. There was no corresponding method to pass the join in the case of the FST. and LST. situation mentioned above, and FOCUS performed the processing regardless of the OPTIMIZATION setting.

Starting with Release 7.0, new behavior is invoked when the multiplicative effect is detected and OPTIMIZATION is set to ON or FOCUS. The new behavior is also invoked regardless of the OPTIMIZATION setting when there are multiple FST. or LST. operators on fields in a segment which can possibly have multiple records per sort break in the request, provided two conditions are met. First, the active joined structure being referenced must represent a single retrieval path, as shown by the output of the CHECK FILE PICTURE RETRIEVE command. Second, all segments in the active structure except the lowest level active segment must have a KEYS parameter value greater than zero in their Access File Descriptions. If either of these conditions is not met, the Interfaces process the request as in prior releases. You are not required to take action to make use of this enhancement unless you are currently using SET OPTIMIZATION SQL in the case of the multiplicative effect. If so, you need to switch to SET OPTIMIZATION ON or FOCUS to invoke the new behavior.

If the above conditions are met by the request, and the Interface detects either of the previously described situations, the join is passed to the RDBMS, with an SQL ORDER BY clause referencing all of the columns contained in each segment's primary key (as identified by the KEYS parameter in the Access File Description), in active segment order from top to bottom, not including the lowest level active segment. This allows the Interfaces to eliminate rows with duplicate primary key values prior to passing the data to FOCUS for output. Once FOCUS receives the answer set from the Interfaces, any necessary additional processing (such as sorting for BY fields) is performed, and the data is output.



In the example that follows, the Master and Access File Descriptions are identical to those used in *Appendix A of the FOCUS for IBM Mainframe DB2 and SQL/DS Read/Write Interface Users Manual, Release 6.8., DN100048.1093*. A multi-table Master and Access File Description named EMPLOYEE is used, that relates the two tables PAYROLL and COURSES. The concepts discussed also apply to structures created with the FOCUS dynamic join command.

The Master File Description is:

```
FILENAME=EMPLOYEE, SUFFIX=SQLDS,$
SEGNAME=PAYROLL, SEGTYPE=S0,$
FIELD=EMP_ID,EMP_ID,A5,A5,MISSING=OFF,$
FIELD=SALARY,SALARY,P13.2,P6,MISSING=OFF,$
SEGNAME=COURSES,SEGTYPE=S0,$
FIELD=EMP_NUM,EMP_NUM,A5,A5,MISSING=OFF,$
FIELD=COURSE_NUM,COURSE_NUM,A5,A5,MISSING=OFF,$
FIELD=POINTS,POINTS,I3,I2,MISSING=OFF,$
```

The Access File Description is:

```
SEGNAME=PAYROLL, TABLENAME="USER1"."PAYROLL",
KEYS=1,WRITE=YES,KEYORDER=LOW,$
SEGNAME=COURSES, TABLENAME="USER1"."COURSES",
KEYS=2,WRITE=YES,KEYORDER=LOW,
KEYFLD=EMP_ID,IXFLD=EMP_NUM,$
```

The KEYS attributes indicate that these tables participate in a one-to-many relationship. The RDBMS tables contain the following rows:

PAYROLL:	EMP_ID	SALARY	COURSES:	EMP_NUM	COURSE_NUM	POINTS
	11111	10000.00		11111	10674	95
	22222	15000.00		11111	10680	97
				22222	10674	93
				22222	10676	89
				22222	10682	71

Consider the following report request:

```
TABLE FILE EMPLOYEE
SUM SALARY POINTS
END
```

If OPTIMIZATION was set to SQL, the resulting SQL query would be:

```
SELECT SUM(T1.SALARY), SUM(T2.POINTS) FROM "USER1"."PAYROLL" T1,
"USER1"."COURSES" T2 WHERE (T1.EMP_ID = T2.EMP_NUM);
```

The following report would be generated. Note the multiplied value for SALARY:

```
SALARY      POINTS
-----
65000.00    445
```

In prior releases, with OPTIMIZATION set to ON or FOCUS, the Interface temporarily disabled optimization, issued simple SELECT requests for each table, and constructed the answer set with the "correct" information, as follows:

```
SELECT EMP_ID, SALARY FROM "USER1"."PAYROLL";
SELECT POINTS FROM "USER1"."COURSES" WHERE (EMP_NUM = ?);
```

SALARY	POINTS
25000.00	445

In Release 7.0, with OPTIMIZATION set to ON or FOCUS, the following single SQL statement results:

```
SELECT T1.EMP_ID, T1.SALARY, T2.EMP_NUM, T2.COURSE_NUM, T2.POINTS
FROM "USER1"."PAYROLL" T1, "USER1"."COURSES" T2 WHERE
(T2.EMP_NUM = T1.EMP_ID) ORDER BY T1.EMP_ID;
```

Note the ORDER BY T1.EMP\_ID clause. This allows the Interface to discard PAYROLL segment occurrences which contain duplicate values of EMP\_ID from the answer set, prior to passing the data to FOCUS for output. The resulting report is the same as in prior releases when the join was not passed to the RDBMS:

SALARY	POINTS
25000.00	445

**Notes:**

- In some cases, the new behavior in FOCUS Release 7.0 may not perform as efficiently as that of prior releases, when FOCUS managed the join instead of the RDBMS. For requests where this is the case, the prior behavior may be invoked by issuing the SET OPTIMIZATION OFF command.
- In cases where the Interface-managed native join logic is invoked, no other Interface optimization is done. That is, aggregation and sorting on BY fields are not passed to the RDBMS. This behavior is consistent with that of prior releases, when the join would not have been passed to the RDBMS. Expressions, however, will now be passed.

## **Selective Optimization of FST. and LST. Aggregation Operators**

Starting with FOCUS Release 7.0, the Relational Interfaces are able to pass certain occurrences of the FOCUS FST. and LST. operators to the RDBMS as SQL MIN and MAX operators, respectively. This includes situations in which these operators are explicitly specified in a report request, and those in which they are implicitly specified, such as when referencing a non-numeric field in a report heading or footing. This allows the Interfaces to pass aggregation to the RDBMS in many more cases than in previous releases, thereby decreasing both the size of the answer set returned to FOCUS and the amount of additional processing FOCUS must perform to generate report output. This enhancement is automatic and does not require action on your part in order to make use of it.

In prior releases, the implicit or explicit presence of even one FST. or LST. operator in a report request was sufficient to disable the passing of any aggregation operators to the RDBMS. This was due to the fact that the relational model in general, and the SQL language in particular, have no concept of the "first" or "last" occurrence in a table (segment, in FOCUS terminology).

In FOCUS Release 7.0, the Interfaces are able to detect certain situations where the conversion of FST. and LST. operators to SQL MIN and MAX does not violate the FOCUS concept of "first" and "last" occurrences of data. These situations have to do with how many FST. and LST. operators are present in a request, how they relate to any BY and join fields present, and the KEYS parameter values in the Access File Description(s) of the structure being referenced.

In general, FST. and LST. may be translated to MIN and MAX except in cases where there are multiple occurrences of these operators on a segment table in the structure being referenced, which could possibly return multiple instances or rows for a request. This concept can be made clearer by example: FST. operators on multiple fields in a segment that could return multiple instances, dictate that each field value displayed in the report must come from the same instance. Translating all of the FST. operators to MIN would not necessarily produce the same result, since MIN does not dictate that all returned field values come from the same instance. Even one FST. or LST. each in this situation would not allow translation, as this would dictate that the values in the two referenced fields come from two different instances. Again, MIN and MAX do not guarantee that this will be the case.

The number of segment instances which can be returned by a particular request is determined by Interface analysis of the BY fields in the request, the primary key of each segment in the structure as identified by the KEYS parameter in the Access File Description(s), and any join fields which are also part of the primary keys. This analysis is illustrated in the following example which uses the EMPLOYEE Master and Access File Descriptions from the section *Interface-Managed Native Join Optimization*. Consider the following report request:

```
TABLE FILE EMPLOYEE
SUM FST.SALARY FST.POINTS BY EMP_ID BY COURSE_NUM
END
```

In releases prior to 7.0, the following SQL was generated:

```
SELECT T1.EMP_ID,T1.SALARY,T2.EMP_NUM,T2.COURSE_NUM,T2.POINTS
FROM "USER1"."PAYROLL" T1,"USER1"."COURSES" T2 WHERE
(T2.EMP_NUM = T1.EMP_ID) ORDER BY T1.EMP_ID,T2.EMP_NUM,
T2.COURSE_NUM;
```

Note that no aggregation is passed to the RDBMS because there is no operation equivalent to FST. in the SQL language. This means that the entire answer set must be retrieved so that FOCUS may select the first record. Also, a sort is passed on the composite of the primary keys for both segments in the structure, which allows FOCUS to select the logical "first" record in the file. FOCUS must still sort the answer set by the BY fields in the request, adding still more processing overhead.

In FOCUS Release 7.0, the following SQL results from the same request:

```
SELECT T1.EMP_ID,T2.COURSE_NUM, MIN(T1.SALARY), MIN(T2.POINTS)
FROM "USER1"."PAYROLL" T1,"USER1"."COURSES" T2 WHERE
(T2.EMP_NUM = T1.EMP_ID) GROUP BY T1.EMP_ID,T2.COURSE_NUM ORDER
BY T1.EMP_ID,T2.COURSE_NUM;
```

Because aggregation is now passed to the RDBMS as MIN operators, the resulting answer set will most likely contain fewer rows, which results in less communication between the RDBMS and FOCUS, and less processing by FOCUS prior to producing report output.

Note that the Interface is able to pass the FST.POINTS field to the RDBMS as MIN(T2.POINTS) in the SQL request, even though the BY field for that segment COURSE\_NUM does not cover the segment's entire primary key. In this example, this is because this was the only FST. or LST. operator for that segment. However, due to the Interfaces' analysis capabilities, even if there had been more non-key fields from that segment with these operators, aggregation would still have been passed. This is because the other BY field in the request, EMP\_ID, is also a join field to the remaining component field of the primary key, EMP\_NUM. The Interface is able to determine that at most one data instance of the second segment would occur per sort break (EMP\_ID, COURSE\_NUM), so there is no limit on the number of FST. or LST. operators for that segment. Of course FST.SALARY is able to be passed simply because it is the only occurrence of FST. or LST. for the first segment. However, its entire primary key is covered by a BY field in the request (EMP\_ID), so additional occurrences would have been allowed here as well.

If join optimization for a report request is disabled, or if Interface-managed native join optimization is invoked (see the section on Interface-Managed Native Join Optimization above), then no aggregation optimization is allowed. This behavior is consistent with prior releases.

## Optimization of DEFINE-Based Screening Conditions on Single Segments

Starting with FOCUS Release 7.0, the Interfaces are able to pass screening conditions to the RDBMS based on DEFINE fields that derive their value from a single segment in the structure being accessed, even in the event that Interface join optimization is disabled. Optimization of screening conditions generally results in fewer rows returned from the RDBMS to FOCUS, and decreases the amount of processing FOCUS must perform prior to producing report output. No action on your part is required to make use of this feature.

In releases prior to 7.0, when Interface join optimization was disabled, any DEFINE field expressions (even those which are normally optimizable by the Interface) were not included in screening conditions in the SQL. In FOCUS 7.0, provided the DEFINE field(s) involve only fields from one segment in the structure per DEFINE field expression, and provided optimization was not explicitly disabled via the SET OPTIMIZATION OFF command, the DEFINE expression is included in an SQL WHERE clause and passed to the RDBMS.

The following example uses the EMPLOYEE Master and Access File Descriptions from the section *Interface-Managed Native Join Optimization*. Consider the following report request:



```
SET ALL=ON
```

```
DEFINE FILE EMPLOYEE  
NEWSALARY/P14.2 = SALARY * 1.1;  
NEWPOINTS/I3 = POINTS *.8;  
END
```

```
TABLE FILE EMPLOYEE  
PRINT *  
WHERE NEWSALARY EQ 100000 AND NEWPOINTS EQ 80  
END
```

The following SQL was generated in releases prior to 7.0, because SET ALL=ON disables optimization:

```
SELECT T1.EMP_ID,T1.SALARY FROM "USER1"."PAYROLL" T1;  
SELECT T2.EMP_NUM,T2.COURSE_NUM,T2.POINTS FROM  
"USER1"."COURSES" T2 WHERE (T2.EMP_NUM = ?);
```

In FOCUS Release 7.0, the DEFINE expressions now appear in SQL screening conditions:

```
SELECT T1.EMP_ID,T1.SALARY FROM "USER1"."PAYROLL" T1 WHERE  
((T1.SALARY * 1.1) = 100000.);  
SELECT T2.EMP_NUM,T2.COURSE_NUM,T2.POINTS FROM  
"USER1"."COURSES" T2 WHERE (T2.EMP_NUM = ?) AND  
((T2.POINTS * .8) = 80);
```

Because NEWSALARY and NEWPOINTS each derive their value from only one segment, PAYROLL and COURSES respectively, the DEFINE-based screening conditions for each segment are passed to the RDBMS.

**Note:** Passing of DEFINE fields may still be disabled by issuing a SET OPTIMIZATION OFF command. This behavior is consistent with prior releases.

## Optimization of WHERE/MISSING Clauses as SQL WHERE/NULL Clauses

Starting with FOCUS Release 7.0, the Relational Interfaces are able to pass FOCUS WHERE/MISSING screening conditions to the RDBMS as SQL WHERE/NULL clauses. Optimization of screening conditions generally results in fewer rows being returned from the RDBMS to FOCUS, and decreases the amount of processing FOCUS must perform prior to producing report output. This enhancement also improves the chances for index-based data access through the RDBMS optimizer when a WHERE/MISSING screening condition is used in a request. No action on your part is required to make use of this enhancement.

In releases prior to 7.0, the Interfaces were able to pass FOCUS IF/MISSING screening conditions to the RDBMS as SQL WHERE/NULL clauses. If the FOCUS WHERE/MISSING syntax was used, they were unable to pass that screening condition to the RDBMS as an SQL WHERE clause.

The following example uses the EMPLOYEE Master and Access File Descriptions from the section *Interface-Managed Native Join Optimization*, with the exception that the SALARY field is described as MISSING = ON, and the ACTUAL attribute is specified as P8. Consider the following report request:

```
TABLE FILE EMPLOYEE
PRINT EMP_ID SALARY
WHERE SALARY EQ MISSING
END
```

In releases prior to 7.0, the following SQL was generated:

```
SELECT T1.EMP_ID,T1.SALARY FROM "USER1"."PAYROLL" T1;
```

Note that the screening condition on SALARY was not passed to the RDBMS. This means that all of the data rows in USER1.PAYROLL were retrieved, and that FOCUS performed the screening condition prior to displaying the report output. This type of processing can cause significant overhead to be incurred. In FOCUS Release 7.0, the following SQL results from the same request:

```
SELECT T1.EMP_ID,T1.SALARY FROM "USER1"."PAYROLL" T1 WHERE  
T1.SALARY IS NULL;
```

Because the screening condition is passed to the RDBMS, only those rows where SALARY contains the NULL value are returned to FOCUS. This most likely results in less communication between the RDBMS and FOCUS, and less processing performed by FOCUS prior to displaying the report output.

## Optimization of WHERE/FROM-TO Clauses as SQL BETWEEN Clauses

Starting with FOCUS Release 6.8 Put Level 9306, and continuing with FOCUS Release 7.0, the Relational Interfaces are able to pass FOCUS WHERE/FROM-TO screening conditions to the RDBMS as SQL WHERE/BETWEEN clauses. This improves the chances for index-based data access through the RDBMS optimizer. No action on your part is required to make use of this enhancement.

In releases prior to 7.0, while the Interfaces were able to pass FOCUS IF/FROM-TO screening conditions to the RDBMS as SQL WHERE/BETWEEN clauses, if the FOCUS WHERE/FROM-TO syntax was used, the Interfaces translated that screening condition as dual range predicates in the generated SQL syntax. Depending on the particular screening condition in the request, this syntax sometimes resulted in full tablespace scans of the data, even though the field being screened was indexed in the RDBMS. With this enhancement, FOCUS WHERE/FROM-TO screening conditions are passed to the RDBMS exactly as IF/FROM-TO screening conditions were in prior releases, thereby improving the chances for index access by the RDBMS.

The following example uses the EMPLOYEE Master and Access File Descriptions from the section *Interface-Managed Native Join Optimization*. Consider the following report request:

```
TABLE FILE EMPLOYEE
PRINT EMP_ID SALARY
WHERE SALARY FROM 50000 TO 100000
END
```

In releases prior to 7.0, the following SQL was generated:

```
SELECT T1.EMP_ID, T1.SALARY FROM "USER1"."PAYROLL" T1 WHERE
(T1.SALARY <= 100000.00) AND (T1.SALARY >= 50000.00);
```

In FOCUS Release 6.8 Put Level 9306 and higher, and in FOCUS Release 7.0, the following SQL results from the same request:

```
SELECT T1.EMP_ID, T1.SALARY FROM "USER1"."PAYROLL" T1 WHERE
(T1.SALARY BETWEEN 50000. AND 100000.);
```

Provided the SALARY column has been indexed in the RDBMS, it is far more likely that the latter type of SQL syntax will cause the index to be used for data access, which is usually far more efficient than a full tablespace scan of the data.

## Oracle Interface Optimization Enhancements

### Optimization of IF/WHERE READLIMIT as WHERE ROWNUM

Starting with FOCUS Release 7.0, the Oracle Interface passes FOCUS screening conditions of the form IF or WHERE READLIMIT EQ n to the Oracle RDBMS as SQL WHERE ROWNUM <= n. ROWNUM is a special pseudocolumn that allows Oracle to limit the number of rows retrieved by the RDBMS and returned to the application (in this case, FOCUS). This facility proves useful for testing report requests against small subsets of large data tables before migrating them into production.

The following example uses the EMPLOYEE Master and Access File Descriptions from the section *Interface-Managed Native Join Optimization*, with the exception that the SUFFIX value is SQLORA. Consider the following report request:

```
TABLE FILE EMPLOYEE
PRINT EMP_ID
WHERE READLIMIT EQ 5
END
```

In releases prior to 7.0, the following SQL was generated:

```
SELECT T1.EMP_ID FROM "USER1"."PAYROLL" T1;
```

Note that the screening condition was not passed to the RDBMS. This means that all of the data rows in USER1.PAYROLL was retrieved, and that FOCUS performed the screening condition prior to displaying the report output. This type of processing can cause significant overhead to be incurred.

In FOCUS Release 7.0, the following SQL results from the same request:

```
SELECT T1.EMP_ID FROM "USER1"."PAYROLL" T1 WHERE ROWNUM <= 5 ;
```

Because the screening condition is passed to the RDBMS, only the first five rows retrieved by Oracle are returned to FOCUS. This results in less communication between the RDBMS and FOCUS, and less processing performed by FOCUS prior to displaying the report output.

**Note:** IF/WHERE RECORDLIMIT processing remains the same as in prior releases; it will not invoke the WHERE ROWNUM syntax.

## Enabling of Array Blocking for SELECT and INSERT Requests

Starting with FOCUS Release 6.8 Put Level 9312 (for SELECT requests only) and continuing with FOCUS Release 7.0 (for both SELECT and INSERT requests), the Oracle Interface supports array blocking of SELECT and INSERT requests. Array blocking is a method that Oracle utilizes to buffer repeated executions of the same SQL command (in this case FETCH or INSERT), and then execute them in bulk when the buffer is full. This feature can substantially increase efficiency for certain requests.

The default array block size for both SELECT and INSERT requests is 20. The block size for SELECT requests applies to TABLE FILE requests, MATCH file requests, MODIFY MATCH requests, and Direct SQL Passthru SELECT statements. The block size for INSERT requests applies to MODIFY INCLUDE requests and parameterized Direct SQL Passthru INSERT statements. For additional information on parameterized Direct SQL Passthru for Oracle, see the *FOCUS for IBM Mainframe Summary of New Features, Release 6.8, DN1000933.0593*.

### **Syntax**      **How to Enable Array Blocking for SELECT Requests**

To enable array blocking for SELECT requests, enter the following command

```
SQL SQLORA SET FETCHSIZE n
```

where n is the number of rows to be buffered and retrieved using blocked FETCH commands. Acceptable values are 1 to 5000. The default is 20.

### **Syntax**      **How to Enable Array Blocking for INSERT Requests**

To enable array blocking for INSERT requests, enter the following command

```
SQL SQLORA SET INSERTSIZE n
```

where n is the number of INSERT rows to be buffered before the block of rows is actually transmitted to the RDBMS. Acceptable values are 1 to 5000. The default is 20.

#### **Notes:**

- The sole purpose of the FETCHSIZE and INSERTSIZE settings is to influence performance. High values increase the efficiency of requests that involve many rows, at the cost of a higher virtual storage requirement. The best way to find the optimum value for your particular application is to experiment with different values. However, once values exceed 100 the increased efficiency they provide is generally negligible.
- The INSERTSIZE parameter is only functional for consecutive executions of INSERT statements that are identical to each other (except for the values to be inserted). No other intervening SQL statements are allowed, including COMMIT WORK. If a statement is issued that in any way (other than the inserted values) differs from the current blocked INSERT statement in effect, the block is immediately transmitted to the RDBMS, even if it is only partially full. This restriction has several ramifications:

For MODIFY requests, INSERTSIZE may only be invoked by using the SQL SET LOADONLY command within the MODIFY as follows:

```
MODIFY FILE filename
SQL SET LOADONLY
.
.
.
```



SQL SET LOADONLY suppresses MATCH-generated SELECT statements, as it causes the application to assume that the rows do not currently exist in the RDBMS table. Such statements would interrupt the INSERT statement block as described above, effectively making blocking impossible. For INSERT statements entered through Direct SQL Passthru, this restriction dictates that the INSERT statements must be entered using the parameterized Direct SQL Passthru facility. For additional information on parameterized Direct SQL Passthru for Oracle, see the *FOCUS for IBM Mainframe Summary of New Features, Release 6.8, DN1000933.0593*. This means that the INSERT must be issued using the PREPARE, BIND and EXECUTE command set, using parameter markers for the input values. In this case, the BIND command is not optional, as it is elsewhere. In addition, the PREPARE/BIND/EXECUTE command set must be specified within a BEGIN SESSION/END SESSION command pair. Once the first EXECUTE command is issued, issuing any command other than another EXECUTE of the same PREPARED statement causes the block to be inserted, even if it is not yet full.

Sample session:

```
SET SQLENGINE = SQLORA
SQL SET USER SCOTT/TIGER
SQL SET AUTOCOMMIT ON COMMAND
SQL SET INSERTSIZE 3
SQL BEGIN SESSION
SQL PREPARE ABC FOR INSERT INTO USER1.PAYROLL
VALUES (:001,:002);
END
```

```
SQL BIND ABC USING CHAR(5), DECIMAL(11,2);
END
SQL EXECUTE ABC USING '11111', 50000;      (Row is buffered.)
END
SQL EXECUTE ABC USING '22222', 65000;      (Row is buffered.)
END
SQL EXECUTE ABC USING '33333', 30000;      (Row is buffered and
END                                           block is transmitted.)
SQL END SESSION                             (LUW is committed.)
```

## NF507: FSTRACE for the CA-DATACOM/DB Interface

FSTRACE can be used with all FOCUS report requests to display information that the CA-DATACOM/DB Interface uses to communicate with the DATACOM/DB database. This information includes descriptions and parameters of the DATACOM/DB control blocks and the commands that are passed. You can store trace information in a file (it must be a sequential file in MVS), or you can display it online. To capture trace data in a file, issue the appropriate command from the FOCUS prompt.

### Syntax How to Capture Trace Data in a File

In MVS, the syntax is:

```
{TSO | MVS} ALLOC F(FSTRACE) DA('userid.FSTRACE') -  
    SHR REU LRECL(80) RECFM(F)
```

or

```
DYNAM ALLOC FILE FSTRACE DATASET userid.FSTRACE -  
    SHR REU LRECL 80 RECFM F
```

In CMS, the syntax is:

```
CMS FILEDEF FSTRACE DISK FSTRACE DATA A (LRECL 80 RECFM F
```

To view the trace information, use the system editor or the FOCUS TED editor. To display trace data at the terminal, issue the appropriate command from the FOCUS prompt. In MVS, the syntax is:

### Syntax How to View Trace Information

```
{TSO | MVS} ALLOC F(FSTRACE) DA(*) SHR REU
```

or

```
DYNAM ALLOC FILE FSTRACE DA *
```

In CMS, the syntax is:

```
CMS FILEDEF FSTRACE TERMINAL
```

## NF508: DYNAM Support for BUFNI and BUFND

In addition to the SET BUFNI and SET BUFND commands documented previously, one other method exists for activating the VSAM DATA and INDEX Buffers. The FOCUS DYNAM command now supports the parms BUFND and BUFNI for DATA and INDEX buffers. The syntax is

### **Syntax**      **How to Activate VSAM DATA and INDEX Buffers**

The syntax is:

```
DYNAM ALLOC FI ddname DS dataset-name BUFND m BUFNI n SHR REU
```

where:

- m**                      Is the number of DATA buffers.
- n**                      Is the number of INDEX buffers.

## **Index**

### **Symbols**

**&DATE**

**&DMYY**

**&FOCFOCEXEC**

**&FOCINCLUDE**

**&MDYY**

**&YYMD**

**? FDT Command**

**? FILTER**

**? PTF**

**? SET ALL**

**? SET FOR**

**? SET NOT**

**? TSO DDNAME with wildcard**

### **Numerics**

**31 Bit I/O**

### **A**

**ACCDATA**

**Access File Descriptions**

**ACCTNAME**

**ACCTPASS**

**ADABAS Interface**

**FOCUS 7.0.1**

**NF437 AUTOADBS**

**NF460 Changing the Default CALLTYPE**

**NF473 The New ADABAS Interface**

**FOCUS 7.0.3**

**NF495 Multifetch/Prefetch Options**

**FOCUS 7.0.6**

**NF517 ADABAS Dynamic Security**

**NF538 ADABAS Dynamic Database Number**

**SET PASSWORD**

**Aggregation**

**By External Sort**

**Controlling retrieval order**

**ALL**

**CHECK FILE HOLD**

**ALLOC**

**For multiple volumes**

**Allocation**

**Default space in TABLA**

**Multiple units**

**Multi-volume**

**Of FOCUS files automatically**

**ALLOWCVTERR**

And **DATEDISPLAY**

And **MISSING**

**APF** Internal Authorization

**APFAUTH**

**APPLICATION** in load balancing group

**Applid**

Dynamically changing for **MSO FOCUS**

**ASIS** Function

**AUTOADBS**

**AUTODATACOM**

**AUTOINDEX**

Automatic Indexed Retrieval

**AUTOPATH**

AutoSelect of index in **IMS** Interface

**AUTOTABLEF**

## **B**

**BACKGROUND COLOR** in a Winform

Base dates in **FOCUS** Reports

**Batch**

Allocating multi-volume data sources

Pooled Tables

**BLINK** in a Winform

**Bold** in a Winform

Boundary conditions for Pooled Tables



**Buffers in VSAM**

**BUFND and BUFNI**

**BUSDAYS (Business day units)**

**BY fields**

**Optimizing DEFINE field aggregation**

**BYSCROLL**

**Byte precision for COUNT and LIST**

## **C**

**CA-ACF2 and load balancing**

**CA-DATACOM Interface**

**FOCUS 7.0.1**

**NF488 AUTODATACOM**

**NF507 SET FSTRACE for the CA DTACOM/DB Interface**

**FOCUS 7.0.3**

**NF498 Enabling the CA-DATACOM/DB CBS Trace**

**CA-IDMS Interface**

**FOCUS 7.0.9**

**NF584 Dynamically Setting the IDMS DBNAME and DICTNAME**

**CALLTYPE=FIN**

**Carriage control**

**In FORMAT WP files**

**Catalog Search in FOCUS**

**CBS Trace**

**CC**

**CDN Support in Maintain**

**CEEPIPI**

**Changes to the REBUILD Prompt**

**Changing Retrieval Order with Aggregation**

**CHECK FILE HOLD ALL**

**CICS**

**MSO cooperative processing**

**CICSCOMM**

**CMS Extended Plists**

**CMS FOCUS**

**FOCUS Personal Agent TCP/IP Connectivity Option for**

**CMS Sink**

**Validating userids**

**CMSOAPI functions**

**COLUMN-TOTAL renaming and rejustifying**

**Combo box**

**Commands and subpool boundaries**

**Compiler**

**LE-supported**

**COMPUTE**

**And cross-century dates**

**And currency conversion**

**Concatenation, Universal**

**Console, MSO**

**Display of IMS PSB**

**Routing messages to**

**Controlling REBUILD Messages**

**COUNT**, expanding precision for

**COUNTWIDTH**

**Cross-Century Dates in FOCUS Applications**

**Phase II**

**CS**

**CURR**

**Currency**

**Conversion**

**Processing**

**Conversion calculations**

**Database for conversions**

**Euro support**

**Field in a data source**

**Sample codes**

**CURRENCY\_ID**

**Cursor, moving in a Winform**

## **D**

**Danish external sort**

**Database number, ADABAS, setting dynamically**

**DATA COM, enabling the CBS Trace**

**Datatypes, joining different**

**Date**

- Adding date units**
- And Time Stamp in Reports**
- Base date in FOCUS Reports**
- Business day units**
- Converting formats**
- Converting legacy**
- Cross-century**
- Difference between two**
- Displaying invalid smart dates in reports**
- Functions for the Year 2000**
- Handling for the Year 2000 in FOCUS**
- Holidays**
- Literal interpretation**
- MAINTAIN functions**
- Moving**
- Subtracting date units**
- System, altering for testing**
- Variable, displaying without separators**
- Weekday units**

**Date literal interpretation table**

**DATEADD**

- Using in MAINTAIN**

**DATECVT**

**DATEDIF**

- Using in MAINTAIN**

**DATEDISPLAY**

And ALLOWCVTERR

**DATEMOV**

Using in MAINTAIN

**DATETIME**

**DB2 Interface**

IF-THEN-ELSE Optimization

Left outer join

Optimization of joins between heterogeneous file types

Passing aggregations on DEFINE Fields in BY Clauses

SET ISOLATION

SET OPTIFTHENELSE

**DBCS**

Character support in Teradata

K format and G prefix

**DBCTL**

**DBNAME**

**DBNO**

**DDNAME**

Default space

Listing using wildcard

**DEFCENT**

And CHECK FILE HOLD ALL

In DEFINE and COMPUTE

**DEFINE**

And cross-century dates

And currency conversion

Increased limit

Passing aggregation on to RDBMS

**DEFINE Based Screening Conditions**

**DFC**

In **DEFINE** and **COMPUTE**

**DFSORT**

**Dialogue Manager**

Displaying four-digit year values

**Dialogue Manager Enhancements**

**FOCUS 7.0.1**

NF421 Determining Which FOCEXEC is Running

NF444 ASIS Function

NF452 Capturing SET Parameter Values

**DICTNAME**

**DIF format in Web Interface**

**Difference between two dates**

**Display commands**

Number in a report request

**Distinct operator**

**DNS Names Support**

**DRDA Support Enhancements to the DB2 Interface**

**DST.**

**DU**

## **DYNAM**

- Allocating multiple volumes**
- Support for BUFNI and BUFND**
- Support for existing relative GDG numbers**
- Support for unit count**
- Utilities Menu**

## **Dynamic GETPRV Exit**

## **Dynamic Language Environment support**

## **Dynamically Changing Attributes of MAINTAIN Winforms**

## **Dynamically Setting the Addressing Mode**

## **E**

## **EDA**

- IMS PSB display in console**
- MSO Bridge**

## **EDACFG**

- Enhanced ? SET Command (ALL)**
- Enhancement to ? SET (FOR, NOT)**
- Enhancement to the TED Command in MVS**
- Enhancement to the ZCOMP1 User Exit**
- Enhancements to JOIN**
- Enhancements to Objects in MAINTAIN**

## **Error**

- Messages for Pooled Tables**
- Processing for Pooled Tables**

## **ERRORS, IBITABLA default space allocation table**

**Escape character for LIKE**

**ESTLINES**

**ESTRECORDS**

**Euro support**

**Calculations**

**Currency conversion**

**Processing**

**Currency database**

**Currency-denominated field**

**Sample currency codes**

**EUROFILE**

**Excel (HOLD format)**

**EXCEL format in Web Interface**

**EXTAGGR**

**External index**

**Activating**

**Creating from concatenated database**

**For FOCUS database**

**Positioning indexed fields**

**External Sort**

**Aggregation by**

**Controlling Operations**

**Controlling retrieval order**

**Displaying Messages**

**Estimating SORTWORK sizes**

**HOLD from**



**EXTFOC**  
**EXTHOLD**  
**EXTSEC**

## **F**

**Fast Logon Enhancements**

**FASTPDS**

**FDEFCENT**

**And CHECK FILE HOLD ALL**

**FDFC**

**FETCH**

**FETCHSIZE**

**Field Level PFKeys**

**Actions**

**Forms**

**Triggers**

**Fieldnames**

**For universal concatenation**

**Fields**

**Redefining in Non-FOCUS Files**

**Filter**

**And joins**

**Assigning to a File for Reporting**

**Query command**

**SET command**

**Finish external sort**

**FIXRETRIEVE**

**FOCALLOC**

**FOCCTL.DATA** default space allocation table

**FOCEXEC**

Determining which is running

**FOCPARM**

Configuring Pooled Tables in  
Enhancements

**FOCPOOLT**

**FOCPROF** - The System Wide Profile

**FOCSAM** Interface

**FOCUS 7.0.1**

NF456 VSAM Data and Index Buffers

NF485 Dynamic GETPRV Exit

NF486 Dynamically Setting the Addressing Mode

NF487 Enhancement to the ZCOMP1 User Exit

NF508 DYNAM Support for BUFNI nad BUFND

**FOCSORT**

Allocating multiple units

Allocating multiple volumes

Allocation

Increased size

**FOCSUACC**

**FOCUS**

MSO access from EDA

Web Interface

**FOCUS 7.0.1**

**NF414 The MAINTAIN Facility**

**NF415 SET SAVEMATRIX**

**NF420 Double-Byte Character Set K Format and G Prefix**

**NF421 Determining Which FOCEXEC is Running**

**NF423 Specifying up to 256 Verb Objects**

**NF425 External Indices for FOCUS Databases**

**NF426 AUTOPATH**

**NF427 Longer Length for HOLD FORMAT LOTUS Files**

**NF428 Increased Size of FOCSORT**

**NF429 Generalized Listings of DDNAMEs**

**NF431 Universal Concatenation**

**NF432 Renaming/Rejustifying Row and Column Total Labels**

**NF433 Improved Handling of Text Fields In TED**

**NF434 Larger FOCUS Databases**

**NF435 Increased Number of Literal Values in a File**

**NF436 Checking Language Settings**

**NF437 AUTOADBS**

**NF438 MODEL 204 Interface Enhancements**

**NF440 Improved Page Handling - SET TRACKIO**

**NF441 External Sort**

**NF443 Large Packed Fields**

**NF444 ASIS Function**

**NF446 Using StyleSheets**

**NF448 FOCPARM Enhancements**

**NF449 Increased Number of Indices**

**FOCUS 7.0.1 (continued)**

- NF452 Capturing SET Parameter Values
- NF455 FOCUS File Date and Time Stamp
- NF456 VSAM Data and Index Buffers
- NF457 The IBI MVS Subsystem
- NF460 Changing the Default CALLTYPE
- NF465 Changes to the Catalog Search in FOCUS
- NF466 Controlling IMS Access via DBCTL
- NF467 Automatic Indexed Retrieval (AUTOINDEX)
- NF468 HiperBudget
- NF469 DRDA Support Enhancements to the DB2 Interface
- NF470 Loading Access File Descriptions
- NF471 Enhanced ? SET Command
- NF472 Enhancement to the TED Command in MVS
- NF473 The New ADABAS Interface
- NF474 APF Internal Authorization
- NF476 Usability Enhancements
- NF477 FASTPDS
- NF478 MSO CONSOLE Browser
- NF479 The New MSO/CICS Interface
- NF480 Fast Logon Enhancements
- NF481 DYNAM Utilities Menu
- NF482 IMS Enhancements via SET IMS=NEW
- NF483 The MSO Resource Manager (MRM)
- NF484 Allowing VSAM File Allocation in MSO JCL
- NF485 Dynamic GETPRV Exit

**FOCUS 7.0.1** *(continued)*

- NF486 Dynamically Setting the Addressing Mode
- NF487 Enhancement to the ZCOMP1 User Exit
- NF488 AUTODATACOM
- NF490 Online Release Information
- NF496 Static SQL for TABLE Requests
- NF504 Installation Enhancement for the SQL/DS Interface
- NF505 Optimization Enhancements
- NF507 FSTRACE for the CA-DATACOM/DB Interface
- NF508 DYNAM Support for BUFNI and BUFND

**FOCUS 7.0.3**

- NF491 Distinct Operator
- NF495 ADABAS Interface Multifetch/Prefetch Options
- NF498 Enabling the CA-DATACOM/DB CBS Trace

**FOCUS 7.0.5**

- NF493 Full Support for SDSF under MSO and TSO
- NF494 FOCUS Client - Remote Data Access via EDA/SQL
- NF497 Project 2000 - Cross-Century Dates in FOCUS Applications
- NF499 Scrolling Report Headings in HotScreen
- NF500 Keyed Retrievals from FOCUS Extract Files
- NF501 Public and Private DDname for MSO
- NF509 MINIO
- NF510 Date and Time Stamp in Reports
- NF513 Redefining Fields in Non-FOCUS Files
- NF518 VSAM Support in Maintain
- NF519 Field Level PFKeys

**FOCUS 7.0.5 (continued)**

**NF520 Dynamically Changing Attributes of MAINTAIN Winforms**

**NF521 Enhancements to Objects in MAINTAIN**

**FOCUS 7.0.6**

**NF502 MSO VTAM Logon Time-out**

**NF512 IMS Interface - Automatic Index Selection Using AutoSelect**

**NF517 ADABAS Dynamic Security**

**NF523 Cross-Century Dates in FOCUS Applications - Phase II**

**NF526 PRINTPLUS**

**NF530 Language Environment (LE) Support**

**NF531 31 Bit I/O**

**NF532 MSO Monitoring and Statistics**

**NF536 Multi-image FOCSORT**

**NF538 ADABAS Dynamic Database Number**

**NF539 Outer Join Optimization**

**NF540 Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS**

**NF542 Optimization of Joins Between Heterogeneous File Types**

**NF543 FOCUS Personal Agent - TCP/IP Connectivity Option for VM/CMS FOCUS**

**NF544 FOCUS Personal Agent - TCP/IP Connectivity Option for TSO FOCUS**

**NF546 Enhancements to JOIN**

**NF547 EDA to MSO Bridge**

**NF549 Interpreting Quotation Marks Within Quote-Delimited Literal Strings**

**NF552 Estimating SORTWORK Sizes for an External Sort**

**NF553 Enhanced Message Routing**

**FOCUS 7.0.6 (continued)**

**NF554 Enhanced Load Balancing for MSO**

**NF576 MSO Dynamic VTAM Re-configuration**

**NF578 FOCUS Personal Agent - Interruptible Server for VM and MVS FOCUS**

**FOCUS 7.0.7**

**NF580 Web Interface for FOCUS**

**FOCUS 7.0.8**

**NF550 EDA/MSO Console Display for IMS PSB**

**NF564 Pooled Tables**

**NF566 MSO/CICS Cooperative Processing**

**NF568 DB2 Interface IF-THEN-ELSE Optimization**

**NF571 DB2 Interface SET ISOLATION Command**

**NF574 System 2000 Interface Trace Facility**

**NF579 Assigning Screening Conditions to a File for Reporting Purposes**

**NF583 Teradata Outer Join Optimization**

**NF586 Expanding Byte Precision for COUNT and LIST**

**NF594 JAVA Report Assist**

**NF605 Date Handling for the Year 2000 in FOCUS**

**NF607 Default Space Allocation Table for Work Files**

**NF607 TABLA Enhancements**

**NF609 Sink Validation of Userids in CMS**

**NF617 Automatic Allocation of FOCUS Files**

**NF619 -HTMLFORM SAVE**

**NF620 Year 2000 Subroutines**

**NF623 Increasing the Number of Verbs in a Report Request**

**NF626 JAVA Graph Wizard**

**FOCUS 7.0.8 (continued)**

**NF628 Automatic Activation of Web Interface**

**NF630 Querying Which PTFs Have Been Applied for a Specific Release**

**NF631 Extended Plists**

**NF640 Dynamic Language Environment (LE) Support**

**NF642 Increased DEFINE Limitation**

**NF645 WEBHOME**

**NF647 Extended Support for Scandinavian External Sort**

**FOCUS 7.0.8R**

**NF557 REBUILD Enhancement - Legacy Date Conversion**

**NF653 Displaying Base Dates in FOCUS Reports**

**NF659 CHECK FILE HOLD ALL**

**NF700 New Date Math Functions for Year 2000**

**NF703 Displaying Invalid Smart Dates in Reports**

**NF705 Enhancement to YRTHRESH Command**

**NF708 Enhancement to the TODAY subroutine**

**NF709 Displaying a Date Variable Without Separators**

**NF710 Field FORMAT=YYJUL**

**NF711 Altering Your System Date for Testing**

**NF713 MSO Log Changes**

**NF714 LE Support**



**FOCUS 7.0.9**

**NF575 Fusion**

**NF584 Dynamically Setting the IDMS DBNAME and DICTNAME**

**NF597 Aggregation by External Sort**

**NF652 Teradata Interface Kanji Support**

**NF654 HOLD From External Sort**

**NF655 FOCPROF - The System Wide Profile**

**NF656 Controlling REBUILD Messages**

**NF660 Multi-volume Support in MVS FOCUS**

**NF670 DYNAM Support for Unit Count**

**NF673 Model 204 Interface Account Split**

**NF683 Web Interface support for Maintain Winforms**

**NF684 PCHOLD for Non-Html Files**

**NF691 Escape Character for LIKE**

**NF716 Euro Currency Support**

**NF718 DYNAM Support for Existing Relative GDG Numbers**

**NF720 SQLJOIN OUTER Setting for Relational Interfaces**

**NF722 FOCUS Client DNS Names Support**

**NF728 Changing Retrieval Order with Aggregation**

**NF730 Hold Format PDF**

**NF735 Enhancement to ? SET**

**NF740 Changes to the REBUILD PROMPT**

**NF744 HOLD FORMAT EXCEL**

**NF745 ? PTF Enhancements**

**NF746 Leading Zeros**

**NF748 HOLD FORMAT WP With Carriage Control**

**FOCUS Client**

**FOCUS 7.0.5**

**NF494 Remote Data Access via EDA/SQL**

**FOCUS 7.0.9**

**NF722 DNS Names Support**

**FOCUS database**

**Allocating Multiple units**

**Allocating multiple volumes**

**FOCUS File Date and Time Stamp**

**FOCUS MSO CICS Component**

**FOCUS Personal Agent**

**FOCUS 7.0.6**

**NF543 TCP/IP Connectivity Option for VM/CMS FOCUS**

**NF544 TCP/IP Connectivity Option for TSO FOCUS**

**NF578 Interruptible server for VM and MVS FOCUS**

**FOREGROUND COLOR in a Winform**

**Format**

**For universal concatenation**

**TX in TED**

**YYJUL**

**Forms**

**FST and LST Aggregation Operators**

**FSTRACE**

**For System 2000 Interface**

**For the CA DATACOM/DB Interface**

**Functional Area**

**ADABAS Interface**

- NF437 AUTOADBS
- NF460 Changing the Default CALLTYPE
- NF473 The New ADABAS Interface
- NF495 Multifetch/Prefetch Options
- NF517 ADABAS Dynamic Security
- NF538 ADABAS Dynamic Database Number

**CA-DATACOM Interface**

- NF488 AUTODATACOM
- NF498 Enabling the CA-DATACOM/DB CBS Trace
- NF507 FSTRACE for the CA DATACOM Interface

**CA-IDMS Interface**

- NF584 Dynamically Setting the IDMS DBNAME and DICTNAME

**Dialogue Manager Enhancements**

- NF421 Determining Which FOCEXEC is Running
- NF444 ASIS Function
- NF452 Capturing SET Parameter Values

**FOCSAM Interface**

- NF456 VSAM Data and Index Buffers
- NF485 Dynamic GETPRV Exit
- NF486 Dynamically Setting the Addressing Mode
- NF487 Enhancement to the ZCOMP1 User Exit
- NF508 DYNAM Support for BUFNI and BUFND

**Functional Area** *(continued)*

**FOCUS Client**

- NF494 Remote Data Access via EDA/SQL
- NF722 FOCUS Client DNS Names Support

**FOCUS Personal Agent**

- NF543 TCP/IP Connectivity Option for VM/CMS FOCUS
- NF544 TCP/IP Connectivity Option for TSO FOCUS
- NF578 Interruptible Server for VM and MVS FOCUS

**Fusion**

- NF575 Fusion

**General Enhancements**

- NF415 SET SAVEMATRIX
- NF429 Generalized Listings of DDNAMEs
- NF433 Improved Handling of Text Fields In TED
- NF448 FOCPARM Enhancements
- NF455 FOCUS File Date and Time Stamp
- NF465 Changes to the Catalog Search in FOCUS
- NF471 Enhanced ? SET Command
- NF472 Enhancement to the TED Command in MVS
- NF490 Online Release Information
- NF530 Language Environment (LE) Support
- NF547 Estimating SORTWORK Sizes for an External Sort
- NF549 Interpreting Quotation Marks Within Quote-Delimited Literal Strings
- NF607 TABLA Enhancements (Default Space Allocation Table for Work Files)
- NF609 Sink Validation of Userids in CMS
- NF630 Querying Which PTFs Have Been Applied for a Specific Release

**Functional Area**

**General Enhancements (*continued*)**

**NF631 Extended Plists**

**NF640 Dynamic Language Environment (LE) Support**

**NF655 FOCPROF - The System Wide Profile**

**NF656 Controlling REBUILD Messages**

**NF670 DYNAM Support for Unit Count**

**NF714 LE Support**

**NF718 DYNAM Support for Existing Relative GDG Numbers**

**NF735 Enhancement to ? SET**

**NF740 Changes to the REBUILD Prompt**

**NF745 ? PTF Enhancements**

**NF746 Leading Zeros**

**HiperFOCUS**

**NF468 HiperBudget**

**IMS Interface**

**NF466 Controlling IMS Access via DBCTL**

**NF482 IMS Enhancements via SET IMS=NEW**

**NF512 Automatic Index Selection Using AutoSelect**

**NF550 EDA/MSO Console Display for IMS PSB**

**Interfaces**

**NF513 Redefining Fields in Non-FOCUS Files**

**Functional Area** *(continued)*

**MAINTAIN**

NF414 The MAINTAIN Facility

NF519 Field Level PFKeys

NF520 Dynamically Changing Attributes of MAINTAIN Winforms

NF521 Enhancements to Objects in MAINTAIN

**Model 204 Interface**

NF438 MODEL 204 Interface Enhancements

NF572 Invisible Ordered Character and Ordered Numeric Data Type Key Support

NF673 Model 204 Interface Account Split

**Multi-Session Option**

NF466 MSO/CICS Cooperative Processing

NF474 APF Internal Authorization

NF477 FASTPDS

NF478 MSO CONSOLE Browser

NF479 The New MSO/CICS Interface

NF480 Fast Logon Enhancements

NF481 DYNAM Utilities Menu

NF483 The MSO Resource Manager (MRM)

NF484 Allowing VSAM File Allocation in MSO JCL

NF493 Full Support for SDSF under MSO and TSO

NF501 Public and Private DDname for MSO

NF502 MSO VTAM Logon Time-out

NF531 31 Bit I/O

NF532 MSO Monitoring and Statistics

**Functional Area**

**Multi-Session Option (*continued*)**

NF547 EDA to MSO Bridge

NF553 Enhanced Message Routing

NF554 Load Balancing for the Multi-Session Option (MSO)

NF576 MSO Dynamic VTAM Re-configuration

**National Language Support**

NF420 Double-Byte Character Set K Format and G Prefix

NF436 Checking Language Settings

NF647 Extended Support for Scandinavian External Sort

**Performance Enhancements**

NF425 External Indices for FOCUS Databases

NF426 AUTOPATH

NF440 SET TRACKIO

NF441 External Sort

NF457 The IBI MVS Subsystem

NF467 Automatic Indexed Retrieval (AUTOINDEX)

NF470 Loading Access File Descriptions

NF500 Keyed Retrievals from FOCUS Extract Files

NF509 MINIO - New FOCUS Database Access Method Available Under MVS

NF564 Pooled Tables

NF597 Aggregation by External Sort

NF617 Automatic Allocation of FOCUS Files

NF654 HOLD From External Sort

NF728 Changing Retrieval Order with Aggregation

**Functional Area** *(continued)*

**Raised Limits**

- NF423** Specifying up to 256 Verb Objects
- NF427** Longer Length for HOLD FORMAT LOTUS Files
- NF428** Increased Size of FOCSORT
- NF434** Larger FOCUS Databases
- NF435** Increased Number of Literal Values in a File
- NF443** Large Packed Fields
- NF449** Increased Number of Indices
- NF536** Multi-image FOCSORT
- NF642** Increased DEFINE Limitation

**Relational Interfaces**

- NF469** DRDA Support Enhancements to the DB2 Interface
- NF476** Usability Enhancements
- NF496** Static SQL for TABLE Requests
- NF504** Installation Enhancement for the SQL/DS Interface
- NF505** Optimization Enhancements
- NF539** Outer Join Optimization
- NF540** Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS
- NF542** Optimization of Joins Between Heterogeneous File Types
- NF568** DB2 Interface IF-THEN-ELSE Optimization
- NF571** DB2 Interface SET ISOLATION Command
- NF583** Teradata Outer Join Optimization
- NF720** SQLJOIN OUTER Setting



**Functional Area** (*continued*)

**Reporting Enhancements**

**NF431 Universal Concatenation**

**NF432 Renaming/Rejustifying Row and Column Total Labels**

**NF446 Using StyleSheets**

**NF491 Distinct Operator**

**NF499 Scrolling Report Headings in HotScreen**

**NF510 Date and Time Stamp in Reports**

**NF526 PRINTPLUS**

**NF546 Enhancements to JOIN**

**NF579 Assigning Screening Conditions to a File for Reporting Purposes**

**NF586 Expanding Byte Precision for COUNT and LIST**

**NF623 Increasing the Number of Verbs in a Report Request**

**NF691 Escape Character for LIKE**

**NF744 HOLD FORMAT EXCEL**

**NF748 HOLD FORMAT WP with Carriage Control**

**System 2000 Interface**

**NF574 System 2000 Interface Trace Facility**

**Teradata Interface**

**NF583 Teradata Outer Join Optimization**

**NF652 Kanji support**

**Functional Area** (*continued*)

**Web Interface for FOCUS**

- NF580 Web Interface for FOCUS
- NF594 JAVA Report Assist
- NF619 -HTMLFORM SAVE
- NF626 JAVA Graph Wizard
- NF628 Automatic Activation of Web Interface
- NF645 WEBHOME
- NF683 Web Interface support for Maintain Winforms
- NF684 PCHOLD for Non-Html Files
- NF730 Hold Format PDF

**Year 2000 Enhancements**

- NF497 Project 2000 - Cross-Century Dates in FOCUS Applications
- NF523 Cross-Century Dates in FOCUS Applications - Phase II
- NF557 REBUILD Enhancement - Legacy Date Conversion
- NF605 Date Handling for the Year 2000 in FOCUS
- NF620 Year 2000 Subroutines
- NF653 Displaying Base Dates in FOCUS Reports
- NF659 CHECK FILE HOLD ALL
- NF700 New Date Math Functions for the Year 2000
- NF703 Displaying Invalid Smart Dates in Reports
- NF705 YRTHRESH As an offset from current year
- NF708 Enhancement to the TODAY Subroutine
- NF709 Displaying a Date Variable Without Separators
- NF710 Field FORMAT=YYJUL
- NF711 Altering Your System Date For Testing
- NF713 MSO Log Changes

**Functions**

**For the Year 2000**

**Fusion**

**FYRT**

**FYRTHRESH**

**And CHECK FILE HOLD ALL**

**G**

**G prefix**

**GDG**

**DYNAM support for existing relative numbers**

**General Enhancements**

**FOCUS 7.0.1**

**NF415 SET SAVEMATRIX**

**NF429 Generalized Listings of DDNAMEs**

**NF433 Improved Handling of Text Fields In TED**

**NF448 FOCPARM Enhancements**

**NF455 FOCUS File Date and Time Stamp**

**NF465 Changes to the Catalog Search in FOCUS**

**NF471 Enhanced ? SET Command**

**NF472 Enhancement to the TED Command in MVS**

**NF490 Online Release Information**

**General Enhancements** (*continued*)

**FOCUS 7.0.6**

NF530 Language Environment (LE) Support

NF547 Estimating SORTWORK Sizes for an External Sort

NF549 Interpreting Quotation Marks Within Quote-Delimited Literal Strings

**FOCUS 7.0.8**

NF607 TABLA Enhancements (Default Space Allocation Table for Work Files)

NF609 Sink Validation of Userids in CMS

NF630 Querying Which PTFs Have Been Applied for a Specific Release

NF631 Extended Plists

NF640 Dynamic Language Environment (LE) Support

**FOCUS 7.0.8R**

NF714 LE Support

**FOCUS 7.0.9**

NF655 FOCPROF - The System Wide Profile

NF656 Controlling REBUILD Messages

NF670 DYNAM Support for Unit Count

NF718 DYNAM Support for Existing Relative GDG Numbers

NF735 Enhancement to ? SET

NF740 Changes to the REBUILD Prompt

NF745 ? PTF Enhancements

NF746 Leading Zeros

**GETPRV** Exit

**GNTINT**

**Graph Wizard, JAVA**

## H

### HDAY

Heterogeneous file types, join optimization

Hiding objects in a Winform

HiperBudget

HiperFOCUS

FOCUS 7.0.1

NF468 HiperBudget

HLIMAIN

HOLD

ALL

And DEFCENT, YRTHRESH

FORMAT EXCEL

Format LOTUS length

Format PDF

FORMAT WP

With carriage control

From external sort

Holidays

HTML and the Web Interface for FOCUS

-HTMLFORM SAVE

HTMLMODE

## I

IBI MVS Subsystem

**IBIS DISPLAY**

**IBIS SET**

**APPLGROUP**

**APPLNAME**

**IBITABLA**

**Default space allocation**

**IBMLE**

**Recommended settings**

**IF**

**Optimization in DB2 Interface**

**IMS Interface**

**FOCUS 7.0.1**

**NF466 Controlling IMS Access via DBCTL**

**NF482 IMS Enhancements via SET IMS=NEW**

**FOCUS 7.0.6**

**NF512 Automatic Index Selection Using AutoSelect**

**FOCUS 7.0.8**

**NF550 EDA/MSO Console Display for IMS PSB**

**Index**

**External for FOCUS database**

**Increased number**

**Selecting secondary in IMS Interface**

**Installation**

**Pooled Tables**

**All systems**

**MVS**

**VM/CMS**

**SQL/DS Interface**

**Interface Managed Native Join Optimization**

**Interfaces**

**FOCUS 7.0.5**

**NF513 Redefining Fields in Non-FOCUS Files**

**Interval for monitoring MSO execution**

**Invisible Ordered Character and Ordered Numeric Data Type Key Support**

**Isolation, setting level in DB2 Interface**

**J**

**JAVA Graph Wizard**

**JAVA Report Assist**

**Join**

**And filters**

**Between heterogeneous file types**

**Left outer**

**Controlling**

**Outer in Teradata Interface**

**JOINOPT**

**Justify row and column total labels**

## **K**

**K format**

**Kanji character set**

**Key, invisible Ordered Character and Ordered Numeric support**

**Keyed Retrievals From FOCUS Extract Files**

**KEYORDER ASC/DESC**

## **L**

**Language Environment (LE) Support**

**In FOCUS 7.0.6**

**In FOCUS 7.0.8**

**In FOCUS 7.0.8R**

**Large Packed Fields**

**Larger FOCUS Databases**

**LE support**

**Languages**

**Recommended settings**

**LEADZERO**

**LIKE with escape character**

**Limits**

**DEFINE**

**LIST**

**Expanding precision for**

**List box**



**Literal strings**

Quote-delimited, interpreting Quotation marks in

**Load balancing**

And CA-ACF2

And RACF

And SAF

APPLICATION parameter

Benefits

CICS problem determination

Configuration parameters

For MSO, enhanced

Logon procedures

MSO configuration

Setting defaults

Trouble shooting

Viewing defaults

**Loading Access File Descriptions**

Logs, MSO, support for four-digit years

Long DECIMAL Datatype

LOTUS format in Web Interface

LOTUS HOLD file length

**LU2 Applid**

Dynamically changing for MSO FOCUS

## **M**

### **MAINTAIN**

**CDN support**

**Creating a combo box**

**Creating a list box**

**Creating a radio group**

**Date math functions**

#### **FOCUS 7.0.1**

**NF414 The MAINTAIN Facility**

#### **FOCUS 7.0.5**

**NF518 VSAM Support**

**NF519 Field Level PFKeys**

**NF520 Dynamically Changing Attributes of MAINTAIN Winforms**

**NF521 Enhancements to Objects in MAINTAIN**

**Missing values**

**SU Performance**

**Winforms in Web Interface**

### **Master File**

**And cross-century dates**

### **MATCH FILE**

**With MORE**

### **MAXEXTSRTS**

### **Memory**

**Management with Pooled Tables**

### **Message routing in MSO**

**Messages**

**Pooled Tables**

**MINIO - New FOCUS Database Access Method Under MVS**

**MISSING**

**And ALLOWCVTERR**

**In Maintain**

**Model 204 Interface**

**FOCUS 7.0.1**

**NF438 MODEL 204 Interface Enhancements**

**FOCUS 7.0.8**

**NF572 Invisible Ordered Character and Ordered Numeric key support**

**FOCUS 7.0.9**

**NF673 Model 204 Interface Account Split**

**IOA and ION**

**Monitor**

**MSO execution**

**By userid**

**Region statistics**

**Storage statistics**

**Time interval**

**MORE**

**With MATCH FILE**

## **MSO**

**Access from EDA**

**Allocating multiple units**

**Allocating multiple volumes**

**CICS cooperative processing**

### **FOCUS 7.0.1**

**NF474 APF Internal Authorization**

**NF477 FASTPDS**

**NF478 MSO CONSOLE Browser**

**NF479 The New MSO/CICS Interface**

**NF480 Fast Logon Enhancements**

**NF481 DYNAM Utilities Menu**

**NF483 The MSO Resource Manager (MRM)**

**NF484 Allowing VSAM File Allocation in MSO JCL**

### **FOCUS 7.0.5**

**NF493 Full Support for SDSF under MSO and TSO**

**NF501 Public and Private DDname for MSO**

### **FOCUS 7.0.6**

**NF502 MSO VTAM Logon Time-out**

**NF531 31 Bit I/O**

**NF532 MSO Monitoring and Statistics**

**NF547 EDA to MSO Bridge**

**NF553 Enhanced Message Routing**

**NF554 Load Balancing for the Multi-Session Option (MSO)**

**NF576 MSO Dynamic VTAM Re-configuration**

**MSO** (*continued*)

**FOCUS 7.0.8**

**NF466 MSO/CICS Cooperative Processing**

**Glossary**

**IMS PSB display in console**

**Load balancing, enhanced**

**Logs with four-digit years**

**MSOPRINT**

**Routing messages to**

**Multifetch**

**Multi-image FOCSORT**

**Multiple**

**Units in MVS FOCUS**

**Volumes**

**Default allocations**

**Multi-Session Option**

*See MSO*

**Multi-verb request**

**Number of verbs in**

**MVS**

**Interruptible server via FOCUS Personal Agent**

**LE support**

**MVS batch**

**Allocating multi-volume data sources**

## **N**

### **National Language Support**

#### **FOCUS 7.0.1**

**NF420 Double-Byte Character Set K Format and G Prefix**

**NF436 Checking Current Language Settings**

#### **FOCUS 7.0.8**

**NF647 Extended Support for Scandinavian External Sort**

**Scandinavian external sort**

## **NC**

### **New date**

**Displaying invalid dates in reports**

### **New Features**

#### **FOCUS 7.0.1**

**Allowing VSAM File Allocation in MSO JCL**

**APF Internal Authorization**

**ASIS Function**

**AUTOADBS**

**AUTODATACOM**

**Automatic Index Retrieval (AUTOINDEX)**

**AUTOPATH**

**Capturing SET Parameter Values**

**Changes to the Catalog Search in FOCUS**

**Changing the Default CALLTYPE**

**Checking Current Language Settings**

**Controlling IMS Access via DBCTL**

**Determining Which FOCEXEC is Running**

**New Features**

**FOCUS 7.0.1 (continued)**

**Double-Byte Character Set K Format and G Prefix**  
**DRDA Support Enhancements to the DB2 Interface**  
**DYNAM Support for BUFNI and BUFND**  
**DYNAM Utilities Menu**  
**Dynamic GETPRV Exit**  
**Dynamically Setting the Addressing Mode**  
**Enhancement to the TED Command in MVS**  
**Enhancement to the ZCOMP1 User Exit**  
**External Indices for FOCUS Databases**  
**External Sort**  
**Fast Logon Enhancements**  
**FASTPDS**  
**FOCPARM Enhancements**  
**FOCUS File Date and Time Stamp**  
**FSTRACE for the CA-DATACOM/DB Interface**  
**Generalized Listings of DDNAMEs**  
**HiperBudget**  
**Improved Handling of Text Fields In TED**  
**Improved Page Handling - SET TRACKIO**  
**IMS Enhancements via SET IMS=NEW**  
**Increased Number of Indices**  
**Increased Number of Literal Values in a File**  
**Increased Size of FOCSORT**  
**Installation Enhancement for the SQL/DS Interface**

**New Features**

**FOCUS 7.0.1 (continued)**

**Large Packed Fields**

**Larger FOCUS Databases**

**Loading Access File Descriptions**

**Longer Length for HOLD FORMAT LOTUS Files**

**MODEL 204 Interface Enhancements**

**MSO CONSOLE Browser**

**Online Release Information**

**Optimization Enhancements**

**Renaming/Rejustifying Row and Column Total Labels**

**SET Enhanced ? SET Command**

**SET SAVEMATRIX**

**Specifying up to 256 Verb Objects**

**Static SQL for TABLE Requests**

**The IBI MVS Subsystem**

**The MAINTAIN Facility**

**The MSO Resource Manager (MRM)**

**The New ADABAS Interface**

**The New MSO/CICS Interface**

**Universal Concatenation**

**Usability Enhancements**

**Using StyleSheets**

**VSAM Data and Index Buffers**



**New Features** *(continued)*

**FOCUS 7.0.3**

**ADABAS Interface Multifetch/Prefetch Options**

**Distinct Operator**

**Enabling the CA-DATACOM/DB CBS Trace**

**FOCUS 7.0.5**

**Date and Time Stamp in Reports**

**Dynamically Changing Attributes of MAINTAIN Winforms**

**Enhancements to Objects in MAINTAIN**

**Field Level PFKeys**

**FOCUS Client - Remote Data Access via EDA/SQL**

**Full Support for SDSF under MSO and TSO**

**Keyed Retrievals From FOCUS Extract Files**

**MINIO - New FOCUS Database Access Method Under MVS**

**Project 2000 - Cross-Century Dates in FOCUS Applications**

**Redefining Fields in Non-FOCUS Files**

**Scrolling report headings in HotScreen**

**VSAM Support in Maintain**

**FOCUS 7.0.6**

**31 Bit I/O**

**ADABAS Dynamic Database Number**

**ADABAS Dynamic Security**

**Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS**

**Cross-Century Dates in FOCUS Applications - Phase II**

**EDA to MSO Bridge**

**Enhanced Load Balancing for MSO**

## **New Features**

### **FOCUS 7.0.6** *(continued)*

**Enhanced Message Routing**

**Enhancements to JOIN**

**Estimating SORTWORK Sizes for an External Sort**

**FOCUS Personal Agent - Interruptible Server for VM and MVS FOCUS**

**FOCUS Personal Agent - TCP/IP Connectivity Option for TSO FOCUS**

**FOCUS Personal Agent - TCP/IP Connectivity Option for VM/CMS FOCUS**

**IMS Interface - Automatic Index Selection Using AutoSelect**

**Interpreting Quotation Marks Within Quote-Delimited Literal Strings**

**Language Environment (LE) Support**

**MSO Dynamic VTAM Re-configuration**

**MSO Monitoring and Statistics**

**MSO VTAM Logon Time-out**

**Multi-image FOCSORT**

**Optimization of Joins Between Heterogeneous File Types**

**Outer Join Optimization**

**PRINTPLUS**

### **FOCUS 7.0.7**

**Web Interface for FOCUS**

### **FOCUS 7.0.8**

**Assigning Screening Conditions to a File for Reporting Purposes**

**Automatic Activation of Web Interface**

**Automatic Allocation of FOCUS Files**

**Date Handling for the Year 2000 in FOCUS**

**DB2 Interface IF-THEN-ELSE Optimization**

**New Features**

**FOCUS 7.0.8 (continued)**

**DB2 Interface SET ISOLATION Command**

**Default Space Allocation Table for Work Files**

**Dynamic Language Environment (LE) Support**

**EDA/MSO Console Display for IMS PSB**

**Expanding Byte Precision for COUNT and LIST**

**Extended Plists**

**Extended Support for Scandinavian External Sort**

**-HTMLFORM SAVE**

**Increased DEFINE Limitation**

**Increasing the Number of Verbs in a Report Request**

**Invisible Ordered Character and Ordered Numeric Data Type Key Support**

**JAVA Graph Wizard**

**JAVA Report Assist**

**MSO/CICS Cooperative Processing**

**Pooled Tables**

**Querying Which PTFs Have Been Applied for a Specific Release**

**Sink Validation of Userids in CMS**

**System 2000 Interface Trace Facility**

**TABLA Enhancements**

**Teradata Outer Join Optimization**

**WEBHOME**

**Year 2000 Subroutines**

**New Features** *(continued)*

**FOCUS 7.0.8R**

**Altering Your System Date for Testing**

**CHECK FILE HOLD ALL**

**Displaying a Date Variable Without Separators**

**Displaying Base Dates in FOCUS Reports**

**Displaying Invalid Smart Dates in Reports**

**Enhancement to the TODAY Subroutine**

**Enhancement to YRTHRESH Command**

**Field FORMAT=YYJUL**

**LE Support**

**MSO Log Changes**

**New Date Math Functions for Year 2000**

**REBUILD Enhancement - Legacy Date Conversion**

**FOCUS 7.0.9**

**? PTF Enhancements**

**Aggregation by External Sort**

**Changes to the REBUILD PROMPT**

**Changing Retrieval Order with Aggregation**

**Controlling REBUILD Messages**

**DYNAM Support for Existing Relative GDG Numbers**

**DYNAM Support for Unit Count**

**Dynamically Setting the IDMS DBNAME and DICTNAME**

**Enhancement to ? SET**

**Escape Character for LIKE**

**Euro Currency Support**

**New Features**

**FOCUS 7.0.9 (continued)**

**FOCPROF - The System Wide Profile**

**FOCUS Client DNS Names Support**

**Fusion**

**HOLD FORMAT EXCEL**

**Hold Format PDF**

**HOLD FORMAT WP With Carriage Control**

**HOLD From External Sort**

**Leading Zeros**

**Model 204 Interface Account Split**

**Multi-volume Support in MVS FOCUS**

**PCHOLD for Non-Html Files**

**SQLJOIN OUTER Setting for Relational Interfaces**

**Teradata Interface Kanji Support**

**Web Interface Support for Maintain Winforms**

**NLS**

**Scandinavian external sort**

**NOCC**

**Norwegian external sort**

**O**

**OCCURS**

**Online Release Information**

**ONLINE-FMT**

**OPTIFTHENELSE**

**Optimization**

**DB2 Interface IF-THEN-ELSE**

**Enhancements**

**Of aggregation on DEFINE fields**

**Of joins between heterogeneous file types**

**Of outer joins**

**Controlling**

**Teradata Interface outer join**

**Oracle Interface**

**Left outer join**

**Passing aggregations on DEFINE Fields in BY Clauses**

**Outer Join**

**Optimization**

**Controlling**

**P**

**Packed Fields**

**PCHOLD for Non-Html Files**

**PDF format**

**Creating**

**In Web Interface**

**Performance Enhancements**

**FOCUS 7.0.1**

- NF425 External Indices for FOCUS Databases
- NF426 AUTOPATH
- NF440 SET TRACKIO
- NF441 External Sort
- NF457 The IBI MVS Subsystem
- NF467 Automatic Indexed Retrieval (AUTOINDEX)
- NF470 Loading Access File Descriptions

**FOCUS 7.0.5**

- NF500 Keyed Retrievals from FOCUS Extract Files
- NF509 MINIO - New FOCUS Database Access Method Available Under MVS

**FOCUS 7.0.8**

- NF564 Pooled Tables
- NF617 Automatic Allocation of FOCUS Files

**FOCUS 7.0.9**

- NF597 Aggregation by External Sort
- NF654 HOLD From External Sort
- NF728 Changing Retrieval Order with Aggregation

**Personal Agent**

- Interruptible server for VM and MVS FOCUS
- TCP/IP Connectivity Option for TSO FOCUS
- TCP/IP Connectivity Option for VM/CMS FOCUS

**Plan Management in DB2**

**Plists, extended**

**Pooled Tables**

- Additional information**
- And common selection criteria**
- And non-relational databases**
- And relational databases**
- Batch mode**
- Commands and subpool boundaries**
- Configuring in FOCPARM**
- Error processing**
- Example**
- FOCPARM file**
- FOCPOOLT**
- Installing**
  - All systems**
  - MVS**
  - VM/CMS**
- Memory management**
- Memory needs**
- Messages**
- Questions about**
- Single TABLE clusters**
- Size estimates**
- Sort selection**
- Statistics**
- Subpool boundaries**
- Subroutines for use with**



**Pooled Tables** (*continued*)

Temporary work file

Trace facility

Tuning techniques

**POSITION**

Precision, expanding for COUNT and LIST

Prefetch

Prefix operators

**DST.**

**PRINT**

Text fields in TABLE

**PRINTPLUS**

Private DDname for MSO

**PRIVATEDD**

**PROTECTED** in a Winform

PSB, display in EDA/MSO console

PTF query command

PTF, querying which have been applied

Public and Private DDname for MSO

**Q**

Quote-Delimited Literal Strings

Interpreting quotation marks in

## **R**

**RACF and load balancing**

**Radio group**

**Raised Limits**

### **FOCUS 7.0.1**

**NF423 Specifying up to 256 Verb Objects**

**NF427 Longer Length for HOLD FORMAT LOTUS Files**

**NF428 Increased Size of FOCSORT**

**NF434 Larger FOCUS Databases**

**NF435 Increased Number of Literal Values in a File**

**NF443 Large Packed Fields**

**NF449 Increased Number of Indices**

### **FOCUS 7.0.6**

**NF536 Multi-image FOCSORT**

### **FOCUS 7.0.8**

**NF642 Increased DEFINE Limitation**

**REBUILD**

**Converting legacy dates**

**Date and Time Stamp**

**DATE NEW**

**REBUILDMSG**

**RECFM VBA**

**Redefine fields in non-FOCUS data sources**

**Relational Interfaces**

**FOCUS 7.0.1**

NF469 DRDA Support Enhancements to the DB2 Interface

NF476 Usability Enhancements

NF496 Static SQL for TABLE Requests

NF504 Installation Enhancement for the SQL/DS Interface

NF505 Optimization Enhancements

**FOCUS 7.0.6**

NF539 Outer Join Optimization

NF540 Aggregations on DEFINE Fields Referenced in BY Clauses Passed to RDBMS

NF540 Passing aggregations on DEFINE Fields in BY Clauses

NF542 Optimization of Joins Between Heterogeneous File Types

**FOCUS 7.0.8**

NF568 DB2 Interface IF-THEN-ELSE Optimization

NF571 DB2 Interface SET ISOLATION Command

NF583 Teradata Outer Join Optimization

**FOCUS 7.0.9**

NF720 SQLJOIN OUTER Setting

**Release**

**Querying which PTFs have been applied for**

**Remote Data Access via EDA/SQL**

**Rename**

**Row and column total labels**

**Report**

**Displaying invalid smart dates in  
Number of verbs in  
Pooling requests**

**Report Assist, JAVA**

**Reporting Enhancements**

**FOCUS 7.0.1**

**NF431 Universal Concatenation  
NF432 Renaming/Rejustifying Row and Column Total Labels  
NF446 Using StyleSheets**

**FOCUS 7.0.3**

**NF491 Distinct Operator**

**FOCUS 7.0.5**

**NF499 Scrolling Report Headings in HotScreen  
NF510 Date and Time Stamp in Reports**

**FOCUS 7.0.6**

**NF526 PRINTPLUS  
NF546 Enhancements to JOIN**

**FOCUS 7.0.8**

**NF579 Assigning Screening Conditions to a File for Reporting Purposes  
NF586 Expanding Byte Precision for COUNT and LIST  
NF623 Increasing the Number of Verbs in a Report Request**

**FOCUS 7.0.9**

**NF691 Escape Character for LIKE  
NF744 HOLD FORMAT EXCEL  
NF748 HOLD FORMAT WP With Carriage Control**

**Request**

Number of verbs in  
Pooling multiple reports

**Routing MSO messages**

**RR**

**S**

**SAF and load balancing**

**SAVEMATRIX**

**Scandinavian National Language character set**

And external sort

**Screening conditions**

Assigning to a File for Reporting

**SDSF**

**Security**

ADABAS Dynamic

**SERVICE**

**SET**

ACCTNAME

ACCTPASS

ALL

ALLOWCVTERR

APPLGROUP

APPLNAME

AUTOINDEX

AUTOPATH

**SET** (*continued*)

**AUTOTABLEF**

**BUSDAYS**

**BYSCROLL**

**COUNTWIDTH**

**CURRENT PACKAGESET**

**DATEDISPLAY**

**DATETIME**

**DBNAME**

**DBNO**

**DEFCENT**

**DICTNAME**

**ESTLINES**

**ESTRECORDS**

**EUROFILE**

**EXTAGGR**

**EXTFOC**

**EXTHOLD**

**EXTSORT**

**FETCH**

**FETCHSIZE**

**FILTER**

**FIXRETRIEVE**

**FOCALLOC**

**HDAY**

**HTMLMODE**

**SET** (*continued*)

**IBMLE**

**IMS**

**INCLUDE SUBTREE**

**ISOLATION** in DB2 Interface

**JOINOPT**

**LEADZERO**

**LU2\_NAME**

**MAXEXTSRTS**

**MINIO**

**NDFIND**

**ONLINE-FMT**

**OPTIFTHENELSE**

**Parameter Values**

**PASSWORD**

**POOL**

**POOLBATCH**

**POOLFEATURE**

**POOLMEMORY**

**POOLRESERVE**

**PRINTPLUS**

**SAVEMATRIX**

**SORTLIB**

**SQLJOIN OUTER**

**SUMPREFIX**

**TESTDATE**

**SET** (*continued*)

**TEXTFIELD**

**TRACEOFF**

**TRACEON**

**TRACKIO**

**WEBHOME**

**YRTHRESH**

**Sink**

**Validating userids in CMS**

**Size**

**Estimating SORTWORK for external sort**

**Smart date**

**Displaying invalid dates in reports**

**Sort**

**Controlling retrieval order**

**Estimating external sort SORTWORK sizes**

**Scandinavian National Language character set**

**Selection with Pooled Tables**

**SORTWORK, estimating size for external sort**

**Space**

**Default allocation table in TABLA**

**SQL COMPILE and SQL RUN Processing**

**SQLJOIN OUTER**

**Static SQL for TABLE Requests**

**Statistics for MSO**

**StyleSheets**



**SU performance in Maintain**

**Subpool**

**Boundaries and commands**

**Boundaries for Pooled Tables**

**Subroutines**

**And Pooled Tables**

**And Year 2000**

**Date literal interpretation table**

**LE support**

**Year 2000 support**

**DATEADD**

**DATECVT**

**DATEDIFF**

**DATEMOV**

**In MAINTAIN**

**TODAY**

**SUMPREFIX**

**Swedish external sort**

**SYNCSORT**

**System 2000 Interface**

**FOCUS 7.0.8**

**NF574 System 2000 Trace Facility**

**T**

**TABLA**

**Default space allocation table**

**TABLE**

Pooling requests

**TCP/IP**

FOCUS Personal Agent Connectivity Option for TSO FOCUS

FOCUS Personal Agent Connectivity Option for VM FOCUS

**TED**

Command enhancements in MVS

Handling of text fields

**Teradata Interface**

**FOCUS 7.0.8**

NF583 Teradata Outer Join Optimization

**FOCUS 7.0.9**

NF652 Kanji Support

Outer Join Optimization

Passing aggregations on DEFINE Fields in BY Clauses

**TESTDATE**

**Text field**

Handling in TED

Printing in TABLE

**Time and date stamp in reports**

**Time interval for monitoring MSO execution**

**Time-out for MSO VTAM Logon**

**TODAY**

And Year 2000

**Trace**

For Pooled Tables

For System 2000 Interface

**Triggers**

**TSO**

Allocating multiple units

Allocating multiple volumes

**TSO FOCUS**

FOCUS Personal Agent TCP/IP Connectivity Option for

**TX**

Handling in TED

**U**

**UCOUNT**

Allocating multiple units

**UNDERLINE in a Winform**

**Units**

Allocating multiple

Business day

Weekday

**Universal concatenation**

Fieldname and format matching

With MATCH FILE

**UR**

Usability Enhancements for SQL Interfaces

**USE**

Activating an external index

**Userid**

Monitoring MSO execution by

Validating by Sink in CMS

**User-written subroutines**

LE support

**V**

**Validation**

Of userids by Sink in CMS

**VARGRAPHIC**

**Variable**

Date without separators

**VBA**

**Verb objects**

Specifying 256

**Verbs**

Number in a report request

**VISIBLE in a Winform**

**VM**

CMS extended Plists

FOCUS Personal Agent TCP/IP Connectivity Option for

Interruptible server via FOCUS Personal Agent

LE support

**VMSORT**

**VOLSER**

**Multiple**

**Volume**

**Allocating multiple**

**VSAM**

**Allocation in MSO JCL**

**Data and Index Buffers**

**Support in Maintain**

**VTAM**

**Dynamically changing the MSO FOCUS applid**

**Logon Time-out for MSO**

**MSO Logon Enhancements**

**W**

**Warning messages**

**Pooled Tables**

**Web Interface for FOCUS**

**FOCUS 7.0.7**

**NF580 Web Interface for FOCUS**

**FOCUS 7.0.8**

**NF594 JAVA Report Assist**

**NF619 -HTMLFORM SAVE**

**NF626 JAVA Graph Wizard**

**NF628 Automatic Activation of Web Interface**

**NF645 WEBHOME**

**Web Interface for FOCUS** (*continued*)

**FOCUS 7.0.9**

**NF683 Support for Maintain Winforms**

**NF684 PCHOLD for Non-Html Files**

**NF730 Hold Format PDF**

**WEBHOME**

**Weekday units**

**WHERE/FROM-TO Clauses as SQL BETWEEN Clauses**

**WHERE/MISSING Clauses as SQL WHERE/NULL Clauses**

**WINFORM**

**GET**

**BACKGROUND COLOR**

**BLINK**

**BOLD\_FONT**

**FOREGROUND COLOR**

**PROTECTED**

**UNDERLINE**

**VISIBLE**

**SET**

**BACKGROUND COLOR**

**BLINK**

**BOLD\_FONT**

**FOCUS**

**FOREGROUND COLOR**

**PROTECTED**

**UNDERLINE**

**VISIBLE**

**Winforms**

Changing attributes of  
In Web Interface

**Work files**

Default Space Allocation Table

**WP format**

In Web Interface  
With carriage control

**Y**

**Year 2000 Enhancements**

**FOCUS 7.0.5**

NF497 Project 2000 - Cross-Century Dates in FOCUS Applications

**FOCUS 7.0.6**

NF523 Cross-Century Dates in FOCUS Applications - Phase II

**FOCUS 7.0.8**

NF605 Date handling for the Year 2000 in FOCUS

NF620 Year 2000 Subroutines

**FOCUS 7.0.8R**

NF557 Converting legacy dates

NF653 Displaying Base Dates in FOCUS Reports

NF659 CHECK FILE HOLD ALL

NF700 New Date Math Functions for the Year 2000

NF703 Displaying invalid smart dates in reports

NF705 YRTHRESH As an offset from current year

NF708 Enhancement to the TODAY Subroutine

**Year 2000 Enhancements**

**FOCUS 7.0.8R** *(continued)*

**NF709** Displaying a Date Variable Without Separators

**NF710** Field **FORMAT=YYJUL**

**NF711** Altering Your System Date For Testing

**NF713** MSO Log Changes

**YRT**

In **DEFINE** and **COMPUTE**

**YRTHRESH**

And **CHECK FILE HOLD ALL**

As an offset from the current year

In **DEFINE** and **COMPUTE**

**YYJUL**

**Z**

**ZCOMP1** User Exit