

**Net.Commerce for OS/390  
Version 3 Release 1.2**

# **Frequently Asked Questions (FAQ)**

**December 3, 1999**  
(Second Edition)

Don Bagwell  
Washington Systems Center  
Gaithersburg, MD, USA  
1-301-240-3016  
dbagwell@us.ibm.com

Nick Carlin  
ATS PISC  
Hursley, England, UK  
+44 (0) 1962 816567  
nick\_carlin@uk.ibm.com

(This page intentionally left blank)

(It allows the Table of Contents to be on the right side of the book when duplexed)

---

# Table of Contents

<b>Table of Contents</b> .....	1
<b>Introduction</b> .....	4
Purpose of this document .....	4
Electronic location of this document .....	4
Revisions history .....	4
Submissions to this document .....	5
Acknowledgements .....	5
<b>What's New with OS/390 Net.Commerce V3.1.2</b> .....	6
Object Oriented command structure .....	6
Enhanced security features .....	6
Store Creator .....	6
Additional sample databases .....	6
Additional database tables .....	7
<b>Planning Considerations</b> .....	8
In simple words, what is Net.Commerce? .....	8
What is the difference between a "mall" configuration and a "store" configuration? .....	8
What's involved with implementing Net.Commerce? .....	8
Why is Net.Commerce sometimes referred to as a "toolkit"? .....	9
How much time does a typical implementation take? .....	9
Is it possible to migrate an NT or AIX Net.Commerce solution to OS/390? .....	9
How does Net.Commerce on OS/390 compare to Net.Commerce on NT/AIX? .....	10
What skills are required? .....	12
What aspect of Net.Commerce can be customized? .....	13
Is there any documentation that helps with customization of a store? .....	14
What documentation is available for Net.Commerce V3.1.2? .....	14
Does Net.Commerce need to be running to access the online documentation? .....	15
What education is available for Net.Commerce on OS/390? .....	15
What kind of security function comes with Net.Commerce? .....	15
What utilities come with Net.Commerce? .....	16
How much space will my online store take on my system? .....	16
When should I use JPG files versus GIF files? .....	17
Is DB2 the only database that can be used with Net.Commerce OS/390? .....	17
Can Net.Commerce run on a different system from the DB2 subsystem? .....	17
What browsers work with Net.Commerce? .....	18
I plan on using Product Advisor. What planning issues are involved? .....	18
Do I need to acquire a separate copy of Net.Data to use with Net.Commerce? .....	19
Is Net.Commerce WLM-enabled? .....	19
Can Net.Commerce operate in a Parallel Sysplex environment? .....	19
<b>Installation, System Preparation and Configuring Net.Commerce</b> .....	20
What FMIDs will be included in the product tape? .....	20
Is there any critical maintenance I should be aware of? .....	20
What are the programming minimum requirements for Net.Commerce? .....	20
What are all these different levels of Domino Go Webserver and WebSphere Application Server and Servlet Express? I'm confused. ....	20
What are the issues surrounding Domino Go Webserver 5 vs. Version 4.6.1? .....	21
How much disk space will I need to install Net.Commerce? .....	21
What would a high-level overview of installation and configuration process process look like? .....	21
Anything unusual about the SMP/E installation process for Net.Commerce? .....	21
What do the CMNISHFS, CMNISMKD and CMNMKDIR pre-APPLY jobs do? .....	22
Must I install Net.Commerce at the "/usr/lpp/NetCommerce" mount point? .....	22
What system preparation is required after the code is installed? .....	23
What do I do if I'm using the DSNTIJCL DB2 CLI BIND job? .....	24
What utility is used to configure Net.Commerce? .....	25
What exactly am I to do with the Net.Commerce environment variables file? .....	26
What attributes should the RACF ID for Net.Commerce have? .....	26

May I use Net.Commerce before my updates to SYS1.PARMLIB have taken effect? .....	27
What is an "instance" of Net.Commerce? .....	27
What is the input to the CMNCONF configuration utility? .....	27
What is the output from the CMNCONF configuration utility? .....	28
Is there any way to get all the configuration files into a single HFS directory? .....	30
What is the difference between the various sample malls provided? .....	34
Is there anything I need to do after I run CMNCONF? .....	34
May I manually change the values found in the ncommerce.conf file? .....	35
May I have different people running CMNCONF at the same time? .....	35
How many databases may I employ? .....	35
How many instances of Net.Commerce may I configure? .....	35
Can two instances of Net.Commerce share the same database? .....	35
Why would I ever want to configure more than one instance? .....	35
How can I stop a user from being able to browse the directory contents of the Net.Commerce root directory? .....	35
How do I protect other Net.Commerce files? .....	36
May I turn off SSL? .....	37
May I use a non-SSL port other than 80? .....	38
May I use an SSL port other than 443? .....	38
What's the difference between %%DB2%% and %%SAF%% for authentication? .....	38
If I use %%SAF%%, does that mean the shoppers don't need to be registered? .....	39
Why use %%SAF%%? .....	39
How do I use the %%SAF%% authentication function? .....	39
What is needed to run the Product Advisor function? .....	40
What is needed to enable shoppers to use the Product Advisor Function? .....	40
What is needed to administer and create Product Advisor Catalogs? .....	44
<b>Operating Net.Commerce</b> .....	<b>46</b>
How do I start Net.Commerce? .....	46
How do I stop Net.Commerce? .....	46
Is there any requirement to start the Webserver before I start Net.Commerce? .....	46
What console message will I see to indicate the Webserver is up and running? .....	46
What console message will I see to indicate Net.Commerce is up and running? .....	46
How should I test my Net.Commerce instance to make sure everything is working properly? .....	46
Is there some way to dynamically pick up changes to the Webserver's configuration file? .....	47
Is there some way to dynamically pick up changes to the Net.Commerce configuration files? .....	47
Is there a way to trace Net.Commerce activity? .....	48
Where does the output of the tracing function go? .....	50
Is the "-tr" parameter in the JCL proc used for tracing still there? .....	50
I need to change my Net.Commerce environment variable file. What do I need to do? .....	50
What is the Net.Commerce Administrator (NCADMIN) facility? .....	51
Must I use the NCADMIN function? .....	51
How do I access the NCADMIN function? .....	52
What is default userid and password for the NCADMIN facility? .....	52
How do I change the default password for the Net.Commerce administrator ID? .....	52
What do I do if I forgot the password for the default ncadmin administrator ID? .....	52
Can any registered shopper access the NCADMIN functions? .....	54
How do I create an administrator ID? .....	54
How can I prevent one merchant administrator from seeing the information for another merchant? .....	54
What kinds of Net.Data macros can the Template Designer create? .....	54
Why do Template Designer-created Net.Data macros have the SQL twice? .....	55
What is the Store Creator and when should I use it? .....	56
What changes to the Net.Commerce environment require a restart of the Webserver or Net.Commerce? .....	57
How do I tell Net.Commerce to look for my custom HTML files in a separate, custom directory? .....	58
If I want to create a separate directory for my custom Net.Data macros, how do I do that? .....	59
Can I reference macros located in sub-directories of the directories shown on MACRO_PATH? .....	60
If I have a macro from NC V3.1.1, may I use it with Net.Commerce V3.1.2? .....	60

May I use the directories under /usr/lpp/NetCommerce/instance for my custom HTML or Net.Data macros? .....	60
What is an "Overridable Function"? .....	62
How do I get Net.Commerce to recognize and use my Overridable Function? .....	62
May I keep my custom OFs in a separate directory? .....	63
If I have API Task DLLs left over from V3.1.1, may I use them with V3.1.2? .....	63
How does the Net.Commerce caching function work? .....	64
Is there a way to cache something other than Product or Category pages? .....	66
How do I clear pages from cache? .....	68
How does the Mass Import utility work? .....	69
How can I split my command lines across multiple lines of the input file? .....	71
May I insert comments into the input file of Mass Import? .....	72
How can I improve the performance of Mass Import? .....	72
How does the NCCLEAN utility work? .....	73
How does the Net.Commerce performance monitor work? .....	75
What is the USRTRAFFIC table, and for what is it used? .....	77
<b>Error Symptoms and Their Causes</b> .....	79
<b>Maintenance Information</b> .....	89
APAR/PTFs against Net.Commerce V3.1.2 (FMID H01B312) .....	89
Known bugs not yet included in a PTF .....	89
<b>Index</b> .....	91

---

## Introduction

### Purpose of this document

Very simple: to provide a single-source for the information that always seems so difficult to find were it not located in a single place like this.

It is important to note that this document *will not* serve as a complete "how to" manual for Net.Commerce. The online documentation that comes with the product serves that purpose.

### Electronic location of this document

This document is provided in PDF format on the web at the following location:

Internal IBM	<a href="http://w3.hursley.ibm.com/~carlinn">http://w3.hursley.ibm.com/~carlinn</a>
Customer Version	<a href="http://www.ibm.com/support/techdocs">http://www.ibm.com/support/techdocs</a>

### Revisions history

Date	Update
27-Sept-1999	Original document
03-Dec-1999	<ul style="list-style-type: none"><li>• Added important note concerning potential security hold in Net.Commerce. See "Known bugs not yet included in a PTF" on page 89.</li><li>• Added information on configuring Net.Commerce instance so that all configuration files are in one directory. See "Is there any way to get all the configuration files into a single HFS directory?" on page 30.</li><li>• Corrected "Can any registered shopper access the NCADMIN functions?" on page 54. It used to incorrectly list table ACCTRL as operative table for access control.</li><li>• Added "What do I do if I forgot the password for the default ncadmin administrator ID?" on page 52.</li><li>• Added "How can I split my command lines across multiple lines of the input file?" on page 71.</li><li>• Added a note about CMNCONF's creation of <code>/etc/services</code> on page 29.</li><li>• Added three known bugs to "Known bugs not yet included in a PTF" on page 89.</li><li>• Added "Message "nc3_common.so was not found" running CMNCONF" on page 79.</li><li>• Updated "May I use Net.Commerce before my updates to SYS1.PARMLIB have taken effect?" on page 27.</li><li>• Added "CEE3512S An HFS load of &lt;module name&gt; failed." on page 84.</li><li>• Added "What kinds of Net.Data macros can the Template Designer create?" on page 54.</li><li>• Added "Why do Template Designer-created Net.Data macros have the SQL twice?" on page 55.</li><li>• Added a note about a special consideration for storing Category and Product page macros in custom directories. The note falls under the heading "If I want to create a separate directory for my custom Net.Data macros, how do I do that?" on page 59.</li></ul>

## **Submissions to this document**

If you have a suggestion for improving this document, including ideas for new topics or correcting errors, please send those suggestions to:

dbagwell@us.ibm.com

## **Acknowledgements**

We would like to extend special acknowledgement to the following people who have been particularly instrumental in the development of this document:

- Keith Baillie, IBM US
- Ken Morgan, IBM US

As well as the following people whose input or support was also of great value:

- Dawn Hamilton, IBM US
- Kevin Curley, IBM US
- Stephen Wehr, IBM US
- Steve Matulevich, IBM US
- Bob Teichman, IBM US
- Jay Brodfuehrer, IBM US
- Scott Engleman, IBM US
- Anthony Ciccone, IBM US
- Jay Parmar, IBM US
- Michael Yeung, IBM US
- David Share, IBM US
- Alan Glickenhause, IBM US
- Nils Bergquist, IBM Sweden
- William Wandell, IBM US
- Jack Hoarau, IBM France

---

## What's New with OS/390 Net.Commerce V3.1.2

Net.Commerce V3.1.2 has a large number of new features, functions and details over what was present in the V3.1.1 product. The online help has a section that details all of "what's new" (see "What documentation is available for Net.Commerce V3.1.2?" on page 14 for information on how to access the online help). This document will only touch on some of the more significant changes, and leave the details to the online help.

### Object Oriented command structure

The command structure of OS/390 V3.1.2 mirrors that of the NT and AIX platform. That command structure, and more importantly, the programming structure with which you program Overridable Functions, is Object Oriented, and Net.Commerce comes with a Class Library which provides the framework for developing the OFs.

The advantage of this structure is:

- Increased flexibility
- Portability of code between the NT/AIX platform and the OS/390 platform (with some C++ programming caveats, but it's pretty much a straight port)

The number of commands has expanded significantly as well.

### Enhanced security features

- **Logon/Logoff** -- in V3.1.1, authentication was invoked through the use of the `msprotect` directive. This worked, but didn't provide a way to log off the system without shutting down the browser. With 3.1.2, there's a specific command to logon and logoff, which provides a greater degree of control.
- **Authentication and SSL control** -- in V3.1.1 the invocation of authentication was through the `msprotect` directive (as stated in the earlier paragraph), and SSL was controlled by command group specification at the time of configuration. With V3.1.2, authentication and SSL is controlled down at the command level.
- **Administrative Command Restrictions** -- in V3.1.1, the level of administrative restriction was limited to only the stores an administrator could access. With V3.1.2, you may limit down to the command what functions an administrator may use. It also supports the creation of "groups" that can have those restrictions defined, and administrative IDs can belong to those groups and assume the restrictions defined for the group.

### Store Creator

The Store Creator is a Java Applet used to easily create a standard-format store in Net.Commerce. It walks an administrator through a handful of panels that results in a store being created. It is very handy for new administrators whose needs for a highly customized store are low.

### Additional sample databases

Net.Commerce V3.1.2 now comes with four sample "malls":

- The Metropolitan Demomall
- The Business-to-Business mall
- The East-West Grocery Store (new)



- The East-West Grocery Tutorial
- The Base Mall

The East-West mall has a simplified shopping flow and illustrates some sophisticated techniques of presenting running subtotal of shopping cart. The East-West Tutorial is a modified version of the East-West mall, and is accompanied by a tutorial in the online help to walk you through the process of customizing the mall.

#### **Additional database tables**

Many new tables have been added to support this new function. Some tables from V3.1.1 have been expanded to include additional columns. The online help has details of all these changes.

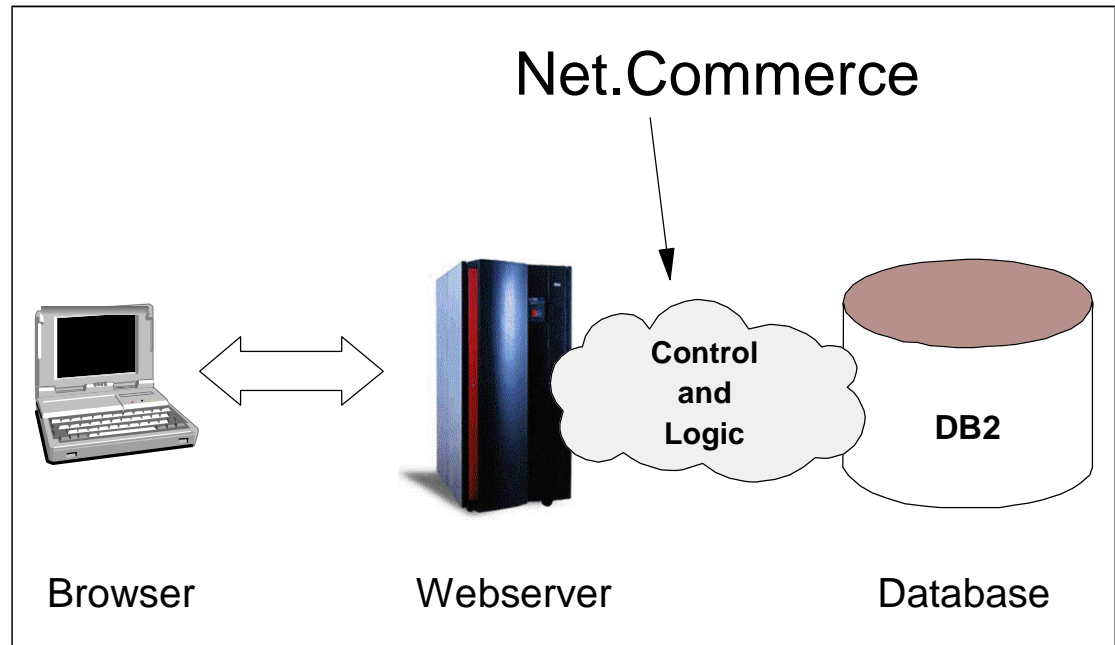
As stated earlier, there is a great deal of further detail that's not being mentioned here. Throughout this document some of those details will come out. It is recommended you consult the online help for specific details of any new function.

---

## Planning Considerations

### In simple words, what is Net.Commerce?

Net.Commerce is at its heart a database application. Access to the database is through the Internet with a browser. The "smarts" needed to control the flow and to generate the HTML pages sit between the Webserver (the access into the system) and the DB2 database (the data repository). Those "smarts" are what Net.Commerce is:



### What is the difference between a "mall" configuration and a "store" configuration?

Net.Commerce has the concept of a "mall" whereby multiple stores are contained within a single logical structure. Shoppers who access the mall are given the opportunity to choose from some number of stores in which to enter, and once in the store they see the typical shopping experience of browsing, selecting and ordering.

This topic causes some confusion because *most* implementations are for a single store, not multiple stores. Net.Commerce is capable of handling a single store configuration as easily as a multiple store configuration. A single store configuration is simply a mall setting with one store.

### What's involved with implementing Net.Commerce?

Implementing Net.Commerce is *much* more than simply installing the code and configuring Net.Commerce. There is a tremendous amount of planning and customization that is required to construct the sophisticated sites you're used to seeing on the web. For now, let's look at a high-level outline of what's involved:

- Determine what you want your Net.Commerce site to do. This would include the screen flow, the function you want invoked and where you want it invoked, and the source and location of the data that will be used by the function.

- Map your requirements against what is supplied by Net.Commerce "out of the box" and determine what aspects of Net.Commerce will need to be customized to meet your needs.
- Plan the customization of Net.Commerce. This would include things such as the HTML pages, Net.Data pages, shopping flow, Overridable Functions, and database changes.
- Perform the work necessary to make the customizations, and then test your system.

This is easier said than done: it requires a great deal of knowledge about Net.Commerce to know what needs changing. This is why involving IBM Global Services (or a qualified business partner) is so important.

### **Why is Net.Commerce sometimes referred to as a "toolkit"?**

Because it is intended to be a framework on which a solution is built. In other words, it is not a "solution" straight out of the box. It is true that several sample malls and stores are supplied with Net.Commerce, but virtually no customer will be satisfied with every aspect of the sample. Therefore, some customization will be required.

That's why the design of Net.Commerce is that of a "toolkit": to allow the product to be stretched and bent to meet the customer's needs.

### **How much time does a typical implementation take?**

That will depend a great deal on the complexity of the site. However, we believe it is safe to say that a typical site involving integration with backend systems will take *at least* 12 weeks. Some implementations have taken more than 6 months to fully bring to production.

This isn't meant to scare you; it's meant to give you a realistic understanding of what to expect.

### **Is it possible to migrate an NT or AIX Net.Commerce solution to OS/390?**

It is possible, though I'm not prepared to say what the process is. There is a redbook residency scheduled for the fall of 1999 to address that very issue. When that redbook comes out, this FAQ will be updated to have the book's order number available.

However, at a high level, we can say the following:

- HTML files should migrate easily. The one issue here is that files on the OS/390 HFS are, by default, in EBCDIC format. It is possible to store HTML files in HFS in ASCII format and have the Webserver pass them down to the browser untranslated. But if you don't wish to take that route, then you need to translate the files to EBCDIC during the file transfer process, and be aware of any issues that might arise because of special or weird characters not translating correctly.
- Net.Data files should also migrate fairly easily. Unlike HTML files which may be stored in the HFS directory in ASCII, Net.Data macros must be in EBCDIC. As you file transfer, be aware of any translation problems you might encounter. (In other words, test your macros.)

The degree of common function between the platforms on which Net.Data runs

is impressive. However, you should be aware that there are environment variables and functions that are specific to the NT or AIX platform that do not apply to OS/390, or are not part of Net.Data for OS/390:

Category	What's Not Supported for OS/390
Environment Variables	DATABASE DTW_EDIT_CODES DTW_PAD_PGM_PARMS LOGIN NULL_RPT_FIELD PASSWORD
Miscellaneous Variables	DTW_DEFAULT_MESSAGE DTW_LOG_LEVEL
Web Registry Functions	(there are no Net.Data web registry functions that apply to OS/390)

If your Net.Data macros on NT or AIX utilize any of these functions, you may need to modify the macros to work on OS/390. Please refer to the Net.Data *Reference* manual for more details:

<http://www.software.ibm.com/data/net.data/library.html>

- The database format is essentially the same between the two platforms, but we have not done a table-by-table analysis of any differences. The redbook team will no doubt take a closer look at this.

Some differences may exist between the use of DB2 on NT and OS/390. Specifically, the use of triggers, text extenders and some SQL syntax might require modification to work on Net.Commerce OS/390 running DB2 V5. With the introduction of DB2 V6, *some* of these will be reduced, but this is an area that will require some analysis.

- The C++ overridable functions will need to be carefully analyzed to see what the issues might be porting to OS/390. If the C++ code was coded to ANSI standards then it should be okay. If direct manipulation of ASCII characters is performed in your code you will have to scrub that for the EBCDIC environment on OS/390. The code will have to be compiled on 390, of course.

For porting code from UNIX/NT to 390 have a look at the UNIX Systems Services porting webpage at:

<http://www.s390.ibm.com/oe/bpxalpor.html>.

The porting guide itself is a PDF file which can be downloaded from:

<http://www.s390.ibm.com/ftp/os390/oe/docs/port.pdf>

and contains a full discussion of the ASCII/EBCDIC considerations, plus many more.

This topic will be explored further in the redbook.

### **How does Net.Commerce on OS/390 compare to Net.Commerce on NT/AIX?**

What follows is a functional comparison matrix of Net.Commerce Version 3.1.2 on both the OS/390 platform and the NT/AIX platform, as well as a column for the new V3.2 on NT and AIX.

**Note:** The functionality of NT/AIX is based on reading rather than hands-on experience.

### **Commerce Engine Functions**

Function	V3.1.2		V3.2	Note
	OS/390	NT/AIX	NT/AIX	
Multiple Store "Mall" Structure	X	X	X	
Shopping Carts, Multiple Shipto, etc.	X	X	X	1
Commands, Tasks and OFs	X	X	X	2
Command and OF Class Libraries	X	X	X	
Create New Commands	X	X	X	
Create New Tasks	X	X	X	
Create New Overridable Functions	X	X	X	
Customize Shopping Flow	X	X	X	
Net.Data	X	X	X	
Customize Backend Integration	X	X	X	3
Support MQSeries, EDI, CICS, IMS	X	X	X	
Backend Server	X	X	X	
Dynamic Page Caching	X	X	X	
Multiple Currency Display Support	X	X	X	
Euro Currency Support	X	X	X	
Translated Versions	X	X	X	4
Automatic Update of Cached Pages		X	X	5
Database Schema Provided	X	X	X	
Referential Integrity	X	X	X	
Customizable Database	X	X	X	
Multi-vendor Database Support		X	X	6
"Tier" Structure	2	2 or 3	2 or 3	7
Multi-vendor HTTP Server Support		X	X	8
Webserver/Net.Commerce Interace	API	API	API	
Multiple Server Configuration	X	X	X	9
Multi-server Load Balancing	X	X	X	10
Automatic Failover	X	X	X	
Use of Cookies	X	X	X	
Defer guest shopper creation until item placed in shopping cart			X	
Userid/PW Protection for Shoppers	X	X	X	
SET Payment for Customer Wallet	X	X	X	
Merchant Originated Set Payment	X	X	X	
Integration with Payment Server	X	X	X	11
Configurable SSL	X	X	X	
Configurable authentication	X	X	X	
Encrypted Password Storage	X	X	X	
Password Storage in Database	X	X	X	
Password Storage in RACF	X			
Logon/Logoff Capability	X	X	X	
ISV plugins available		X	X	12
Prevent userid information from flowing on non-SSL link	X			

## Utilities

Function	V3.1.2		V3.2	Note
	OS/390	NT/AIX	NT/AIX	
Configuration Utility	X	X	X	13
Store Administrator	X	X	X	
Site Administrator	X	X	X	
Template Designer	X	X	X	
Access Control Lists	X	X	X	
Product Advisor	X	X	X	
Catalog Architect		(NT only)	(NT only)	14
Store Creator	X	X	X	
Hosting Server		X	X	15
DB2 Text Extender		X	X	16
Mass Import (massimpt)	X	X	X	
Database Cleanup (ncclean)	X	X	X	
Performance Monitor	X		x	
Multi-level Tracing Facility	X		X	
Dynamic Start/Stop of Tracing and Performance Monitoring	X			

## Notes

Note	Detail
1	This level of functionality is identical between the platforms
2	The NC2 architecture is now common between the platforms. This is a key point, because it allows for a greater degree of portability between the platforms.
3	Slightly richer function in NT/AIX
4	Japanese only for OS/390
5	OS/390 Net.Commerce must first implement support for DB2 UDB triggers. Since DB2 UDB for OS/390 only very recently was released, this support is not yet available.
6	OS/390 Net.Commerce relies on DB2 exclusively
7	OS/390 Net.Commerce requires Webserver, NC and DB2 to reside on the same machine. This is for performance reasons.
8	OS/390 Net.Commerce relies on Websphere (IBM HTTP Server)
9	OS/390 Net.Commerce can achieve this with multiple instances running on the same OS/390 system, or running in separate systems and connected via Parallel Sysplex.
10	OS/390 relies on Workload Manager (WLM)
11	Net.Commerce on NT/AIX is very tightly coupled, on OS/390 it is more loosely coupled (providing greater flexibility to use latest version of Payment Server)
12	IBM is actively working with ISVs to port more of their applications to OS/390 to work with Net.Commerce on this platform
13	Different interfaces, but same result.
14	Output from NT version can be used with OS/390 Net.Commerce
15	This is a high-priority "to do" item for OS/390 Net.Commerce
16	OS/390 Net.Commerce waiting on DB2 UDB

## What skills are required?

A wide variety of skills will ultimately be required to make a Net.Commerce implementation successful:

- OS/390 system administration Skills
  - RACF
  - DB2 (particularly if you're planning on customizing the database)
  - UNIX Systems Services
  - HTTP Server
  - SMP/E
- Net.Commerce application skills
  - Knowledge of how the command structure works
  - Knowledge of how to use the NCADMIN utility
  - Knowledge of Net.Data
  - Knowledge of the Net.Commerce database schema
- Programming skills
  - C++
  - Other languages if that's what you use to access backend systems
- HTML, Javascript, graphics and design skills
  - Page layout
  - Image composition
- Business skills
  - Project Management (critical component of a Net.Commerce project)
  - Business process knowledge
  - Taxation
  - Logistics
  - Accounting and payment processing
  - Legal
  - Marketing
  - Advertising

This is why the involvement of IBM Global Services is recommended: a Net.Commerce implementation is a complex project spanning many disciplines. Professional project management is critical.

### **What aspect of Net.Commerce can be customized?**

Nearly everything:

- The static HTML pages used by the site
- The Net.Commerce commands that execute when a shopper clicks on a link or a button
- What happens when a Net.Commerce command is executed
- Whether a new command (added by you) is executed
- Net.Data macros, including the following customization within the macros:
  - The SQL queries issued against the database
  - The HTML that is wrapped around the returned information
  - The Net.Data logic employed to act upon the user's actions or the information returned from the database
  - The number of queries issued
- The information in the Net.Commerce database

- The layout of the Net.Commerce database (with some limitations)
- The error messages that result when errors are encountered

**Is there any documentation that helps with customization of a store?**

Yes. The East West Foodmart Tutorial provides a step-by-step process for customizing the sample East West store. The tutorial can be found in the online help, or by downloading the `ewtut.pdf` document located in the HFS directory at `/usr/lpp/NetCommerce/books/en_US`.

**What documentation is available for Net.Commerce V3.1.2?**

Here is a list of the documentation that is available for Net.Commerce V3.1.2 for OS/390:

<b>GI10-4657-02</b>	<i>Program Directory</i> , which covers how to install the product using SMPE
<b>GC24-5862-01</b>	<p><i>Configuring and Getting Started</i>, which covers how to prepare your system to run Net.Commerce, as well as providing a reference source on what the various configuration parameters are and what they mean.</p> <p>Download this from the web (in PDF format) at location:  <a href="http://www.s390.ibm.com/ecommerce">http://www.s390.ibm.com/ecommerce</a></p>
<b>SC24-5886-00</b>	<p><i>Messages and Codes</i>, which documents the messages and their meanings.</p> <p>Download this from the web (in PDF format) at location:  <a href="http://www.s390.ibm.com/ecommerce">http://www.s390.ibm.com/ecommerce</a></p>
(online)	<p>Much of the specific function "how-to" documentation is in the form of online help accessible with a browser. If you have Net.Commerce installed on a system, this help can be accessed with the following URL:</p> <p><a href="http://&lt;your host name&gt;/nchelp/index.htm">http://&lt;your host name&gt;/nchelp/index.htm</a></p>
<b>ewtut.pdf</b>	<p>East-West Foodmart Tutorial, which is a PDF-format document of the tutorial found in the online help. This tutorial walks you through some of the steps involved with customizing a sample store. This PDF file can be found in the HFS directory of an installed copy of Net.Commerce at:</p> <p><code>/usr/lpp/NetCommerce/books/en_US/ewtut.pdf</code></p>
<b>ncadmin.pdf</b>	<p>A PDF-format document that contains much of the information found in the online help pertaining to the Net.Commerce administrative (NCADMIN) function. This PDF file can be found in the HFS directory at:</p> <p><code>/usr/lpp/NetCommerce/books/en_US/ncadmin.pdf</code></p>



<b>nctd.pdf</b>	A PDF-format document that contains information on the Template Designer function of Net.Commerce. It can be found in the HFS directory at:  <code>/usr/lpp/NetCommerce/books/en_US/nctd.pdf</code>
<b>patut.pdf</b>	A PDF-format tutorial document that contains information on the Product Advisor function of Net.Commerce. It can be found in:  <code>/usr/lpp/NetCommerce/books/en_US/patut.pdf</code>

The `www.s390.ibm.com/ecommerce` web site also has some sample code used to integrate Net.Commerce with backend systems like CICS and MQSeries. If you're not a programmer, don't bother; it's very complex. If you are a programmer, then have fun.

### **Does Net.Commerce need to be running to access the online documentation?**

No. It's just static HTML that is accessible through the Webserver without requiring Net.Commerce to be running.

### **What education is available for Net.Commerce on OS/390?**

The Washington Systems Center offers a class on Net.Commerce for OS/390. The class is four days in length, has hands-on labs, and runs ever month through June of 2000. For enrollment and more information, contact Don Bagwell, `dbagwell@us.ibm.com`.

### **What kind of security function comes with Net.Commerce?**

Security is probably the number one issue in the minds of many people planning a Net.Commerce implementation. The security of the *system* (not just Net.Commerce) is provided from several sources:

- Access to the site is controlled by firewall technology that can be placed between the shoppers on the Internet and the network inside the firewall. The firewall can be configured to allow only certain protocols (HTTP) or certain ports (80 and 443) through. OS/390 itself has a firewall, shipped free of charge with the product since OS/390 2.5.
- For users accessing the Webserver with a browser, the RACF ID under which the user will operate is called PUBLIC. That ID is set up with extremely limited capabilities. The PUBLIC userid should be defined with "no access" to any restricted resources on your system.
- The ability to browse or otherwise see the contents of directories accessible by the Webserver is protected by directives in the Webserver's configuration file.
- Once a shopper is into the Net.Commerce system, access to the individual functions of Net.Commerce can be controlled using the Net.Commerce administrator (NCADMIN) function. Each command can be designated as required authentication or not. If so designated, the user must be pre-registered and log on to access the function.
- Users designated as administrators (able to use the NCADMIN utility) can be restricted to accessing information for certain stores only, and can be restricted from using any NCADMIN command. This is how you provide limited administrative authority to people so they can only do certain things.

- Transmission of information across the Internet can be encrypted using SSL. This is designated within Net.Commerce by command.

This is a very complex topic and somewhat beyond the scope of this document, with the exception of those tasks related to configuring the Net.Commerce security functions.

### What utilities come with Net.Commerce?

Here's a list of the utilities:

Utility	Description	Page
CMNCONF	An ISPF panel application used to configure the Net.Commerce "instances".	25
Tracing	A facility to trace the activity of Net.Commerce.	48
NCADMIN	A set of utilities used to create and administer the information contained in the database, as well as two Java applets to a) create Net.Data macros, and b) create stores.	51
Mass Import	An OMVS command-line utility used to bulk-load information into the database from a delimited file.	69
NCCLEAN	An OMVS command-line utility used to clean up specified tables in the database	73
Performance Monitor	A utility used to report on the time it has taken for Net.Commerce to execute the commands it has received from the shopper's browser.	75

The "page" column points to the page in this document where a more complete description of this utility can be found.

### How much space will my online store take on my system?

This is a very difficult question to answer precisely because so much depends on the makeup of your store. There are several components of Net.Commerce that will consume space (beyond the code itself):

- Any static HTML used by the store or mall
- The JPG and GIF graphics files
- The Net.Data macros used by the store or mall

The amount of space this takes for your site depends entirely on how many files your system has. To give you a feeling, the "Metropolitan Demomall" provided as a sample has about 350 macro, HTML, JPG and GIF files and consumes about 2.8MB of space.

- The information about the store, its products, etc., kept in the Net.Commerce database (in DB2).

The amount of space this takes depends on the makeup of the data in the database. However, we can give a rough idea of the space a given type of record takes up, and then extrapolate from there:

Record	Bytes per record	Note
Customer	1200 bytes	
Address	1100 bytes	1
Category	4000 bytes	2
Product	4000 bytes	2

Record	Bytes per record	Note
Product Attribute	600 bytes	3
Shopping Cart	350 bytes	
Shipto Assignment	4000 bytes	
Order	400 bytes	
<b>Notes:</b> <ol style="list-style-type: none"> <li>1. A shopper may have more than one address record</li> <li>2. There are two tables that define the category-to-category relationship and the product-to-category relationship. They are not accounted for here. But the space they consume is not significant.</li> <li>3. A product may possess 0 or more attribute records</li> </ol>		

Imagine a store with 4 categories, each with 4 products (16 products total), each product with 4 attribute records (64 attribute records total); 100 registered shoppers, each with 1 address record, 5 products in their shopping cart (500 records), and 1 order placed and confirmed.

The amount of space that scenario would consume is approximately **950KB**.

This is a *rough estimate only*. There are over a hundred tables in the database schema, and this calculation didn't touch on most of them.

- Customized commands or overridable functions, which will be in the form of executable C++ programs

This also depends heavily on the degree of customization you employ. An overridable function with a basic set of function will be fairly small; one that performs a great deal of function will necessarily be larger.

#### **When should I use JPG files versus GIF files?**

Generally speaking, Jpeg is best for complex, fully, rendered images, this includes the following:

- photographic pictures
- naturalistic and semi-realistic artwork
- grey-scaled images

On the other hand, GIF is better with images of few distinct colours of the following kind:

- line drawings
- practical lettering with borders etc.
- simple cartoon drawings with limited colors used

#### **Is DB2 the only database that can be used with Net.Commerce OS/390?**

At the present time, yes. You may, however, through the use of customized Overridable Functions (OFs) access other databases.

#### **Can Net.Commerce run on a different system from the DB2 subsystem?**

No. Net.Commerce must reside on the same system as the DB2 subsystem. This question is typically asked by people who have seen pictures of the NT or AIX platform and its support of ODBC, which allows Net.Commerce to sit on one NT or AIX box, and the database engine to sit on another. Net.Commerce on OS/390 looks to connect to a native DB2 subsystem, and that "on another box" picture doesn't apply to Net.Commerce on OS/390.

Even in a Sysplex environment, each physical system in the Sysplex would have a local copy of DB2 running. The Sysplex Coupling Facility would be the function that maintains the sharing of the data across the machines in the sysplex.

### What browsers work with Net.Commerce?

The minimum requirement for a *shopper's* browser is the ability to handle cookies, Javascript and URL redirects. Any standard browser (Netscape or IE) will do.

**Note:** You should apply PTF UQ34468 to prevent problems running Net.Commerce with Internet Explorer.

The Net.Commerce administrator function (NCADMIN) is only supported officially on Netscape. Use at least Netscape 4.5 or higher.

### I plan on using Product Advisor. What planning issues are involved?

Several things:

- Planning the installation of the code and making it all work. Refer to "What is needed to run the Product Advisor function?" on page 40 for more on this.
- Planning the product catalog so it'll work with Product Advisor and allow it to work to its fullest. This we'll discuss here.

Within Net.Commerce is the concept of a "product attribute", which is a way of describing a product without relying exclusively on long text fields in the DB2 tables. These attributes are in the form of "name/value pairs", and are housed in the Net.Commerce database. For example, consider the case of a simple sweater. You could describe it like this:

*"A warm accessory for the chilly months, this sweater comes in four different colors (blue, red, white and black), two styles (V-neck or crew-neck), and three fabrics (wool, cotton, or blended)."*

And this is fine, but it is difficult for a program to efficiently parse this description and do much with it. It would be better to somehow house this information in a database table, and to access the information via DB2 calls.

That's what Net.Commerce attributes are all about.

Now take a close look at the description provided above. Assuming the options come in all combinations, that description implies 24 different stock-keeping units, or things that can be sold (4 colors x 2 styles x 3 fabrics = 24).

Take as an example *one* of those 24 combinations: a red crew neck sweater in wool. Another way to describe that would be to use attribute name/value pairs:

Attribute Name	Attribute Value
Color=	Red
Style=	Crew
Fabric=	Wool

So now imagine this information kept in a relational database, and imagine a browser interface program that allowed a customer to select the color, style and fabric. With the attributes set up like this and housed in the database, Net.Commerce could then quickly scan through the database and find the combination of attributes that matched what the customer desired.

Product Advisor, at a very high level, is a browser-interface program that allows shoppers to select products based upon their attributes. As such, it requires that products have attributes defined. Defining attributes is not a requirement of Net.Commerce, so an implementer of Net.Commerce must plan and define the attributes for Product Advisor to work.

So, the message is this: *if you wish to use Product Advisor, begin thinking about how you'd define your products with attributes.*

**Do I need to acquire a separate copy of Net.Data to use with Net.Commerce?**

No. It comes with a copy of Net.Data under the covers.

**Is Net.Commerce WLM-enabled?**

Net.Commerce *itself* is not WLM enabled. The Webserver is.

**Can Net.Commerce operate in a Parallel Sysplex environment?**

Yes. The underlying database is DB2, and it may be shared across systems in a Sysplex. Net.Commerce would run in each machine in the Sysplex. HTML files, JPG/GIF files and Net.Data macros would be held in the HFS of each system. TCP/IP traffic would be spread across the systems using IBM's Net.Dispatcher program, working in conjunction with OS/390 Workload Manager (WLM).

---

## Installation, System Preparation and Configuring Net.Commerce

### What FMIDs will be included in the product tape?

There are two FMIDs delivered on the product tape:

- H01B312 (Net.Commerce V3.1.2)
- HCME121 (Payment Server)

It is *not required* that you install the Payment Server. Install that only if you plan on utilizing the SET (Secure Electronic Transactions) function of Net.Commerce.

### Is there any critical maintenance I should be aware of?

See "Maintenance Information" on page 89 for more information on this topic.

### What are the programming minimum requirements for Net.Commerce?

For Net.Commerce:

- OS/390 Version 2 Release 4. Recommendation is you be at V2R5 at a minimum to pick up significant enhancements in TCP/IP performance.
- DB2 Version 5.1. DB2 Version 6 (UDB) has been announced, but we have little testing experience with this. At this time, Net.Commerce has *not* been updated to take advantage of two key aspects of UDB: text extenders (for doing searches into long text fields), and triggers (to automatically update a portion of the database when a change occurs in another area of the database).
- Domino Go Webserver V4.6.1 or Domino Go Webserver V5.0 or V5.1, which has been renamed to "IBM HTTP Server".

There's a "gotcha" with respect to running with Domino Go Webserver V5.0 or V5.1, and you can read more about it by referring to the next item.

### What are all these different levels of Domino Go Webserver and WebSphere Application Server and Servlet Express? I'm confused.

You have a right to be confused. The Corporation could have made a better job of the migration from Domino Go WebServer to WebSphere Application Server. Here is a brief summary.

- With OS/390 2.4 you basically had a WebServer called Domino Go WebServer 4.6.1, which ran Java Servlets (such as the Product Advisor) in separate MVS address spaces to the WebServer.
- OS/390 2.5 introduced Domino Go 5.0 which had a product called "Servlet Express" available to it. Servlet Express is a way to run Java Servlets (such as the Product Advisor) inside the WebServer's address space, thus greatly improving performance.
- OS/390 2.6 and 2.7 introduced the WebSphere Application Server. This is a rebranding of both the Domino Go Web Server and the Servlet Express products into a unified "WebSphere Application Server". The WebSphere Application Server has two components, an HTTP Server and a Java Servlet environment. Unfortunately the Java Servlet environment is also called the WebSphere Application Server! Thus all the confusion.

Today, if you order the latest release of OS/390 you'll get these two components and you should use them for your Net.Commerce site.

## What are the issues surrounding Domino Go Webserver 5 vs. Version 4.6.1?

For a straightforward Net.Commerce store that just uses HTML pages and Net.Data macros there is little difference between version 4.6.1 and version 5.x of the WebServers. As a general rule of thumb, the newer the version of WebServer the better the performance is. We have run Net.Commerce with Domino Go 4.6.1, 5.0 and 5.1 and they all seem to work.

If your site will employ the Product Advisor function of Net.Commerce please refer to "What is needed to run the Product Advisor function?" on page 40.

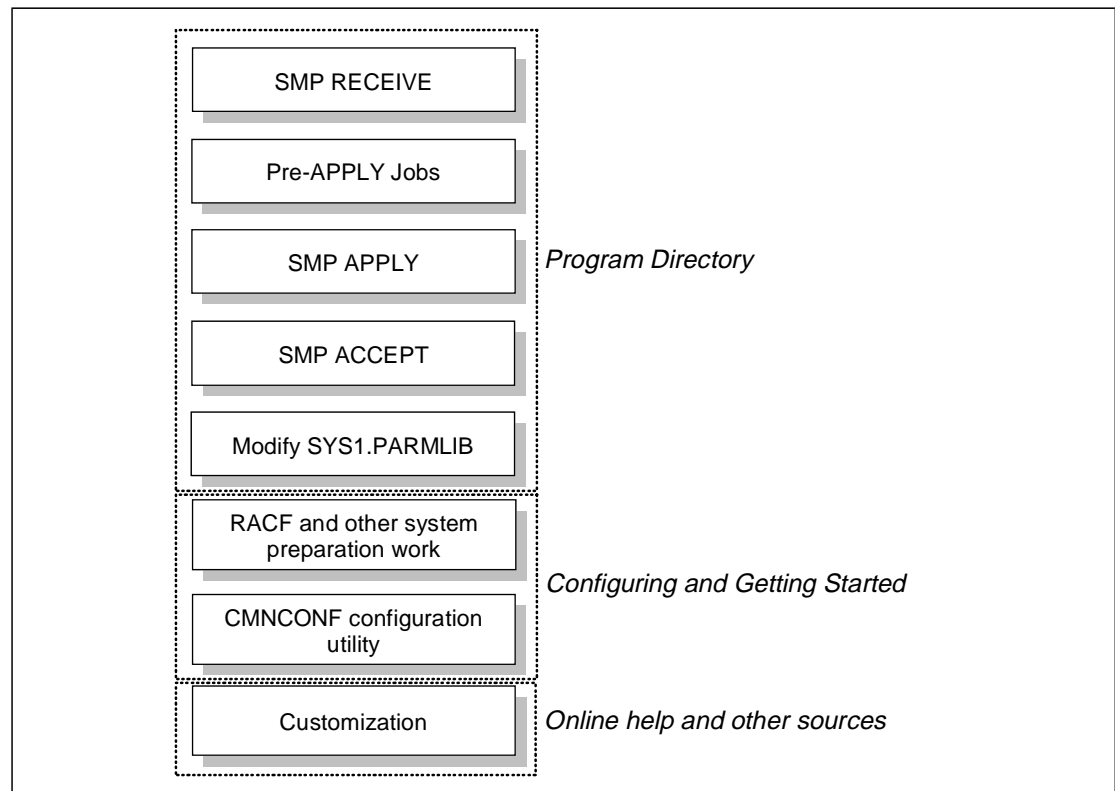
## How much disk space will I need to install Net.Commerce?

The *Program Directory* provides the following information:

- Target Library, 300 Tracks of 3390
- Distribution Library: 1255 Tracks of 3390
- HFS: 1500 Tracks of 3390

## What would a high-level overview of installation and configuration process process look like?

Something like this:



All of the "boxes" in the chart with the exception of "Customization" are well covered in the two documents shown. Customization is a big subject.

## Anything unusual about the SMP/E installation process for Net.Commerce?

No. It's pretty straight-forward. If you don't have any familiarity with OS/390 UNIX Systems Services, then the pre-APPLY jobs CMNISHFS, CMNISMKD and CMNMKDIR might look a little foreign to you.

## What do the CMNISHFS, CMNISMKD and CMNMKDIR pre-APPLY jobs do?

Here's a brief explanation:

- **CMNISHFS** does three things:
  1. Allocates the HFS dataset that will house the Net.Commerce HFS structure
  2. Creates a directory in the HFS to act as a mount point for the Net.Commerce HFS structure
  3. Mounts the Net.Commerce HFS structure at the mount point.
- **CMNISMKD** does two things:
  1. Calls the CMNMKDIR script to create all of the subdirectories in the Net.Commerce HFS structure
  2. Creates the symbolic links and external links within the HFS structure
- **CMNMKDIR** is simply a REXX exec that contains the commands used to do the work for step #1 in the CMNISMKD job

### Must I install Net.Commerce at the "/usr/lpp/NetCommerce" mount point?

The jobs don't really suggest you should change the `/usr/lpp/NetCommerce` part of this, though you technically could. What the jobs do provide is a way to change the path prefix of the mount point, so that you can make it unique.

If you are installing the `/usr/lpp/NetCommerce` off the root directory, then the value you provide for `-PathPrefix-` will be simply the forward slash symbol. So in the CMNISHFS job, the job will be changed to look like this:

#### Sample Job

```
//SYSTSIN DD *
PROF MSGID WTPMSG
MKDIR '-PathPrefix-usr/lpp/' MODE(7,5,5)
MKDIR '-PathPrefix-usr/lpp/NetCommerce/' MODE(7,5,5)

MOUNT FILESYSTEM('-cmnhfs-') /* and mount the HFS */ +
MOUNTPOINT('-PathPrefix-usr/lpp/NetCommerce') TYPE(HFS) MODE(RDWR)
/*
```

#### Updated to reflect installation off of root

```
//SYSTSIN DD *
PROF MSGID WTPMSG
MKDIR '/usr/lpp/' MODE(7,5,5)
MKDIR '/usr/lpp/NetCommerce/' MODE(7,5,5)

MOUNT FILESYSTEM('OEHFS.CMN.V312.HFS') /* */ +
MOUNTPOINT('/usr/lpp/NetCommerce') TYPE(HFS) MODE(RDWR)
/*
```

Bottom line: you can change the `-PathPrefix-` value, but it's probably best not to mess with the `/usr/lpp/NetCommerce` value.



## What system preparation is required after the code is installed?

This is covered in two places, the *Program Directory* and *Configuring and Getting Started* manual. The basic things to be done are:

- Modify SYS1.PARMLIB and add SCMNLMOD to the LNKLIST xx or PROGxx member

You can skip this step if you'd like, but then you must reference the library on the STEPLIB of the Webserver proc and the Net.Commerce proc.

- Change the Userid directive in the webserver's httpd.conf file to allow PUBLIC access

This is a simple one word change in the file.

- Enable program control of the Net.Commerce load module and DB2 load module

The *Configuring and Getting Started* manual doesn't mention the DB2 load module, but we've found this to be necessary.

- Create a RACF userid that will own the Net.Commerce database

See "What attributes should the RACF ID for Net.Commerce have?" on page 26 for details on this.

- Update the Webserver start procedure to point to SCMNLMOD and SCLBDLL in the STEPLIB DD statement

If these datasets are referenced in the SYS1.PARMLIB LNKLISTxx or PROGxx members, then you don't need to STEPLIB them.

- Update the Webserver's environment variable file to reflect the Net.Commerce directories

Here's what you'll be adding:

- /usr/lpp/NetCommerce/msg/en\_US/%N to the NLSPATH value
- /usr/lpp/NetCommerce/lib to the LIBPATH value

- Updating the Net.Commerce sample start procedure

The sample JCL start procedure is provided in the dataset hlq.SCMNSAMP, member CMNMSERV. Copy this member to your PROCLIB and make the updates indicated in the comments of the JCL.

You will be asked to provide a pointer to the instance envvars and configuration file, which you may not have yet created. You may either provide the name you plan to use, or skip this step until after you run CMNCONF.

- Binding the DBRM to the Net.Commerce plan

Which bind job you use depends on what database you intend to employ:

Database	Bind Job
Demo Mall	hlq.SCMNSAMP (CMNBIND)
Base Mall	hlq.SCMNSAMP (CMNBIND)
Business-to-Business Mall	hlq.SCMNSAMP (CMNBNDDB)
Euromall	hlq.SCMNSAMP (CMNBIND)
Grocery	hlq.SCMNSAMP (CMNBNDDEW)
Grocery Tutorial	hlq.SCMNSAMP (CMNBNDDEW)

Database	Bind Job
Custom	This depends on the sample database from which your custom database is derived. Use the bind job guideline provided for the sample database most closely related to your custom database.
Any of the above if you intend to use Websphere Application Server along with the mall. This implies the use of Product Advisor, or your own custom Websphere application.	hlq.SDSNSAMP(DSNTIJCL)  It is necessary to modify the default DSNTIJCL BIND make it work. See "What do I do if I'm using the DSNTIJCL DB2 CLI BIND job?" on page 24 for more information on this.

- Updating the sample Net.Commerce environment variable file  
See "What exactly am I to do with the Net.Commerce environment variables file?" on page 26 for details on this.
- Adding the Net.Commerce panel and message library datasets to the ISPF logon procedure concatenation list

The datasets are:

- hlq.SCMNPENU -- panel library; concatenate to ISPPLIB
- hlq.SCMNMENU -- message library; concatenate to ISPMLIB

The details of all of these are offered in the two documents referenced at the top of this entry.

### What do I do if I'm using the DSNTIJCL DB2 CLI BIND job?

On page 22 of the Configuring and Getting Started manual there is a write-up on the binding of the DBRM to the Net.Commerce plan. In that write up are some instructions on what to do if you use the CLI BIND job supplied by DB2, called DSNTIJCL. Boiled down, here it is:

- Add the Net.Commerce DBRM library to the DBRMLIB DD statement of the JCL:

```
//JOB LIB DD DISP=SHR,
//          DSN=DSN!!0.SDSNLOAD
//BINDCLI EXEC PGM=IKJEFT01,DYNAMNBR=20
//DBRMLIB DD DISP=SHR,
//          DSN=DSN!!0.SDSNDBRM
//          DD DISP=SHR,DSN=hlq.SCMNDBRM
//SYSTSPRT DD SYSOUT=*
```

**Note:** you will also have to modify the default DSNTIJCL job to meet your local DB2 environment.

- Modify the end of the PKLIST record to include the necessary Net.Commerce stuff. What you put on the end of it will depend on what database you are using. What you'll do is steal some of the information from the Net.Commerce sample bind jobs and put it into the DSNTIJCL job

Database	Take information from
Demomall Base Mall Euromall	hlq.SCMNSAMP(CMNBIND)
Business to Business Mall	hlq.SCMNSAMP(CMNBNDDB)

Database	Take information from
East West Mall East West Tutorial	hlq.SCMNSAMP (CMNBNDREW)

And the information you'd pull is from the BIND PLAN line. What you want is the members and the bind parameters. For example, the information you'd pull from CMNBIND would be:

```
DSN S(DSN1)
BIND PLAN(MSERVER) MEMBER(NETCDB2,DTWGA225) ACTION(REPLACE) +
ISOLATION(CS) CURRENTDATA(NO);
RUN PROGRAM(DSNTIAD) PLAN(DSNTIA51)
```

(Pull the information in **BOLD**)

So the resulting PKLIST section of the DSNTIJCL job would look like this:

```
BIND PLAN(DSNACLI) -
PKLIST(DSNAOCLI.DSNCLICS -
DSNAOCLI.DSNCLINC -
DSNAOCLI.DSNCLIRR -
DSNAOCLI.DSNCLIRS -
DSNAOCLI.DSNCLIUR -
DSNAOCLI.DSNCLIC1 -
DSNAOCLI.DSNCLIC2 -
DSNAOCLI.DSNCLIF4 -
DSNAOCLI.DSNCLIMS -
DSNAOCLI.DSNCLIQR ) -
MEMBER(NETCDB2,DTWGA225) -
ISOLATION(CS) CURRENTDATA(NO)
```

```
END
/*
```

<b>Note:</b>	you may need to place an ACTION(REPLACE) at the end of this if DSNACLI was previously bound.
<b>Warning:</b>	<p>The sample DSNTIJCL job supplied by DB2 does <b>not</b> contain the command:</p> <pre>GRANT EXECUTE ON PLAN MSERVER TO PUBLIC;</pre> <p>as do the sample bind jobs supplied by Net.Commerce. Failure to execute that command after binding the DSNTIJCL job will result in Net.Commerce failing to start, with a reason code of F30034.</p>

Consult page 23 and 24 of *Configuring and Getting Started* for examples of what information you'd pull for the Business-to-Business and East-West malls. For information on where to get that manual, see "What documentation is available for Net.Commerce V3.1.2?" on page 14.

### What utility is used to configure Net.Commerce?

The utility provided is called CMNCONF, and it is a ISPF panel based application used to configure the "instances" of Net.Commerce. The next several entries provide additional detail on what CMNCONF does.

## What exactly am I to do with the Net.Commerce environment variables file?

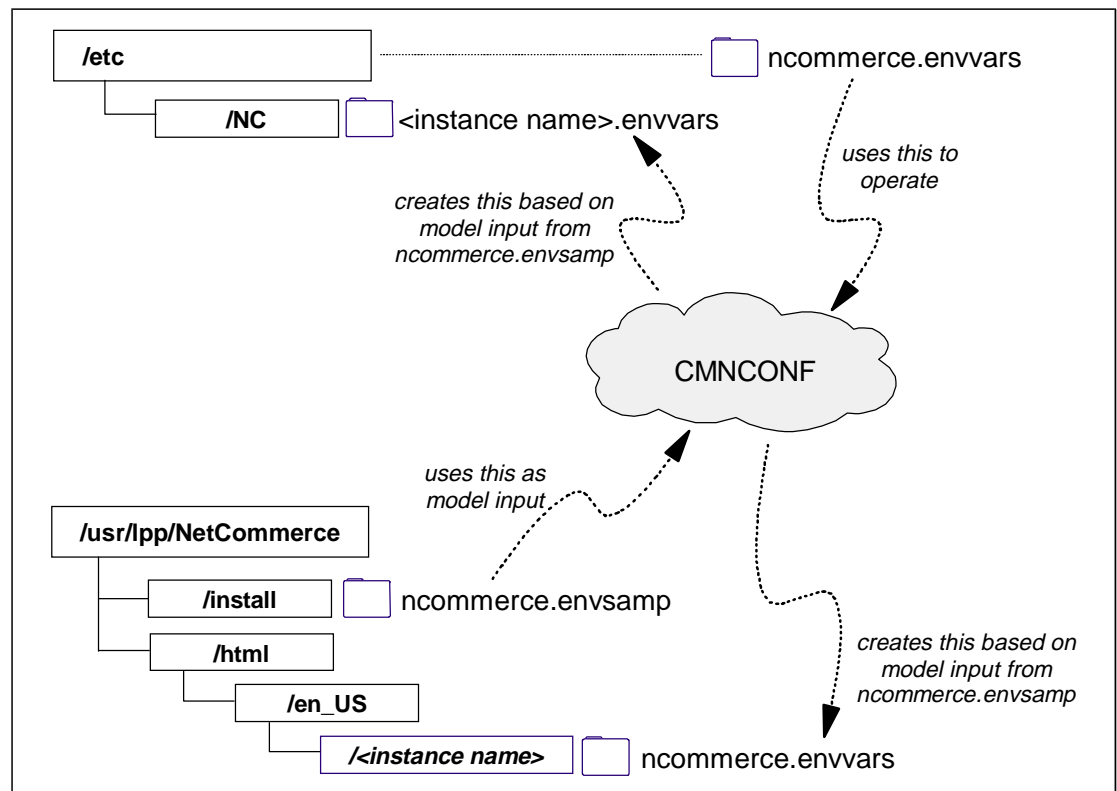
There's a bit of confusion about this. There are two uses for this environment variables file:

- To provide the environment with which the CMNCONF utility runs
- To provide a model for CMNCONF to use when creating the instance configuration.

So here's what you do:

1. Locate the file `/usr/lpp/NetCommerce/install/ncommerce.envsamp`. This is the sample environment variable file provided with Net.Commerce.
2. Make a backup of this file. You'll be making changes to the original, so to be safe you should back it up. `ncommerce.envsamp.original` is as good a name as any.
3. Now go back and edit the `ncommerce.envsamp` file and make any changes necessary to match your system. Refer to Table 1, page 25, in the manual, *Configuring and Getting Started*, for information on what each line in the environment variable is.
4. With the `ncommerce.envsamp` file updated properly, now copy the file to the `/etc` directory and rename it `ncommerce.envvars`. This file is what provides the CMNCONF utility with the environment variables it needs to operate.

Now you're ready to run the CMNCONF utility. Here's a picture of what will result when you run CMNCONF:



## What attributes should the RACF ID for Net.Commerce have?

The userid should have the following attributes:

- UID of 0 so that it always runs with superuser authority
- Read access, if you have defined the BPX.DAEMON facility class
- Update access, if you have defined the BPX.SERVER facility class
- Read access to the datasets defined in the startup procedure

To create the ID, first establish the group to which the ID will belong:

```
ADDGROUP CMNGRP SUPGROUP(SYS1) OMVS(GID(2))
```

Then create the ID

```
ADDUSER CMNSRV DFLTGRP(CMNGRP) OMVS(UID(0))
  HOME('/usr/lpp/NetCommerce') PROGRAM ('/bin/sh'))
PERMIT BPX.DAEMON CLASS(FACILITY) ID(CMNSRV) ACCESS(READ)
PERMIT BPX.SERVER CLASS(FACILITY) ID(CMNSRV) ACCESS(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

### May I use Net.Commerce before my updates to SYS1.PARMLIB have taken effect?

Yes. If you wish to use the CMNCONF utility you will need to issue the TSO command `tsolib act dsn('hlq.SCMNLMOD')`. Then, if you want to start Net.Commerce, you need to make sure that `hlq.SCMNLMOD` and `hlq.SCLBDLL` are referenced on STEPLIB in the Webserver's start procedure, as well as the Net.Commerce start procedure.

Note also that the Mass Import and NCCLEAN utilities (as well as CMNDB2), which operate out of the OMVS environment, need access to `hlq.SCMNLMOD`. If that dataset is not referenced on LINKLST, then you have to provide a STEPLIB environment variable for your OMVS environment that points to the `hlq.SCMNLMOD` dataset. Don't forget to export the environment variable:

```
STEPLIB='hlq.SCMNLMOD'
export STEPLIB
```

See "CEE3512S An HFS load of <module name> failed." on page 84 for the error symptom and resolution for this problem.

### What is an "instance" of Net.Commerce?

An "instance" is simply a running copy of the Net.Commerce program. You may run more than one "instance" of Net.Commerce on the same OS/390 system. Each unique instance will have its own configuration files and its own JCL start procedure. It may or may not have its own database, depending on how you configured it.

### What is the *input* to the CMNCONF configuration utility?

The key input is what you provide with your fingers on the keyboard. However, the CMNCONF utility goes out and grabs information from a number of different sources:

1. The `/etc/nconfig.dat` file is used to track all of the instances of Net.Commerce that have already been configured.
2. The `/etc/services` file is used by CMNCONF to see what port ranges have already been reserved.
3. The `/etc/ncommerce.envvars` file is used to provide a set of environment variables for the CMNCONF program itself to use.

4. The `/usr/lpp/NetCommerce/install/ncommerce.envsamp` file, which will serve as the model input for the environment variables that will be copied to the instance's HTML root directory and the `/etc/NC` directory when the CMNCONF is finished.
5. The `/usr/lpp/NetCommerce/html/en_US/db2www.ini` file, which is copied to the instance's document root by CMNCONF and then modified based on the input provided in the fields of the CMNCONF screens.
6. Any model configuration file that you point to on the first panel of the CMNCONF utility. Using this feature may save time if you have fields in the configuration that are common across all instances configured.
7. All other instance configuration files will be checked to make sure there's no overlap of information between instances. For example, the same owning RACF ID can not be used in two instances pointing to different database names; and the same database name can not be used in two different instances with different owning RACF IDs.
8. The information you supply in the fields of the CMNCONF application.
9. The MVS dataset `hlq.SCMNSAMP` holds the SQL script used to create the sample database (if that's the selection you make, as opposed to creating a custom database).
10. There's another file that contains SQL script to complete the creation of the database:

```
/usr/lpp/NetCommerce/html/en_US/demomall/data/democom.sql
```

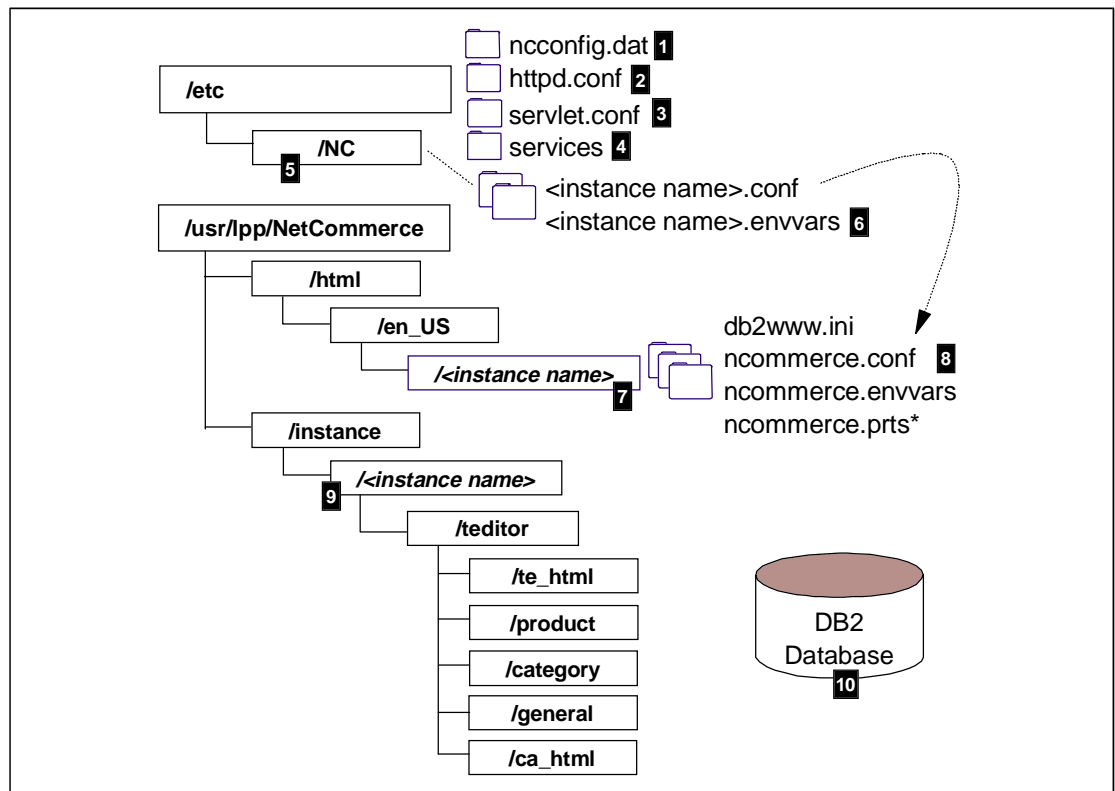
11. Finally, the data itself that is loaded into the tables is located in HFS files. Each table is represented by a separate file. The bulk of the data to be loaded is in

```
/usr/lpp/NetCommerce/html/en_US/demomall/data
```

The other sample malls (Grocery, Euromall, and Business-to-Business) also have `/data` directories that appear to offer the incremental differences those malls have over the Demomall.

### **What is the *output* from the CMNCONF configuration utility?**

Here's a picture of the *default* output from CMNCONF:



And here's a description of what each referenced item is for:

1. One line per instance configured is included in the `ncconfig.dat` file, and the line points to the configuration files included under the `/etc/NC` directory (by default; it may be something else if you name it differently) as referenced with numbers 5 and 6.
2. The Webserver's configuration file (called `httpd.conf` in this picture, but it can be something else if you wish) is updated by CMNCONF to include the appropriate directives to "connect" the Webserver to Net.Commerce.
3. The Webserver's servlet configuration file (called `servlet.conf`) is updated by CMNCONF to include information about Net.Commerce's Product Advisor function. If this file doesn't exist, CMNCONF will create it. There is a good deal more work that needs doing to make Product Advisor function. To read more about that, refer to "What is needed to run the Product Advisor function?" on page 40.
4. The `/etc/services` file is where CMNCONF provides the information about the ports across which the GWAPI code will communicate with the daemons.

**Note!** It appears that OS/390 will search the HFS structure for a "services" file first, and if it finds it then it won't search further into the MVS dataset `TCPIP.ETC.SERVICES`. If you rely on the contents of an MVS dataset to define your TCP services, be aware that CMNCONF will place a `/etc/services` file that appears to take precedence over the MVS dataset.

5. On the "System Configuration" panel of CMNCONF, you're asked the location of the "Server Controller Conf File". By default that value will specify the `/etc/NC` directory. If you wish, you may change it.

6. The files that are created are shown here. The high-level name of the files will be equal to the name of the instance configured. The extension will be `.conf` and `.envvars`. If you're modifying an existing configuration, you'll also see a `.confold` file. This is simply the previous version of the file backed up and named with a `.confold` extension.

These files are known as the "Server Controller" configuration files. Don't worry so much about what a "server controller" is; just be aware this directory and these files are here.

The contents of the `<inst_name>.conf` file point to the `ncommerce.conf` file that will be used (reference number 8 in the picture).

7. A directory will be created off of the `/usr/lpp/NetCommerce/html/en_US` directory with a name equal to the name of your configured instance. It's here that all the server configuration files are kept. This directory is also known as your "Instance HTML root" directory, or sometimes the Net.Commerce "document root".

**Note:** this is the second set of `*.conf` and `*.envvars` file for a Net.Commerce setup. The other is located in `/etc/NC`.

8. The files in this directory are always going to be named `db2www.*` and `ncommerce.*`. These are the primary configuration files for the instance.

**Note:** the `ncommerce.prt` file is not actually created by CMNCONF, but rather when the Net.Commerce server is started. We've shown it here just so you won't be surprised if you browse the directory later.

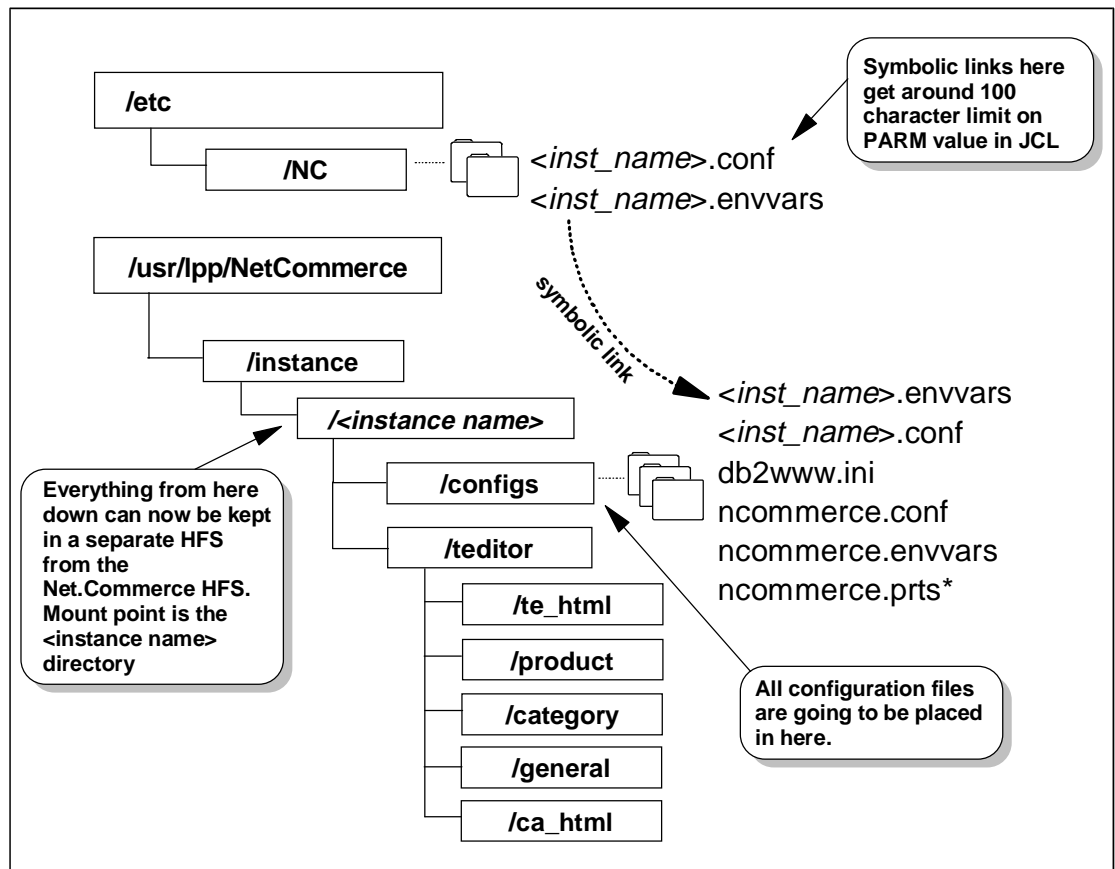
9. A directory will be created off of the `/usr/lpp/NetCommerce/instance` directory with a name equal to the name of your configured instance. There will be subdirectories under this directory, and it's in here that files created using the "Template Designer". In the act of creating the instance, a handful of example files will be copied into these lower subdirectories.
10. Finally, if you choose to create and load a database, the information is added to DB2.

### Is there any way to get all the configuration files into a single HFS directory?

Yes. And this concept can (and should) be taken one step further by placing all of your Net.Commerce customized files under a single directory and make that single directory a mount point for a separate HFS. This makes it easier to isolate your custom files from program fixes, and allows you to roll changes from your test system up to your production system by simply cloning the HFS and carrying it over.

Before we begin, recall what CMNCONF does by default. This was presented under "What is the *output* from the CMNCONF configuration utility?" on page 28. What we'll be doing is the following:





Here's the process:

1. Install Net.Commerce. The Net.Commerce HFS will be mounted at the mount point `/usr/lpp/NetCommerce` (by default) in the root HFS of your system.
2. Determine what name by which your instance will be known. We'll refer to that as `<inst_name>` here.
3. Using OMVS or ISHELL, create a directory called `<inst_name>` off of the `/usr/lpp/NetCommerce/instance` directory.

**Note:** this directory must have permissions of 755!

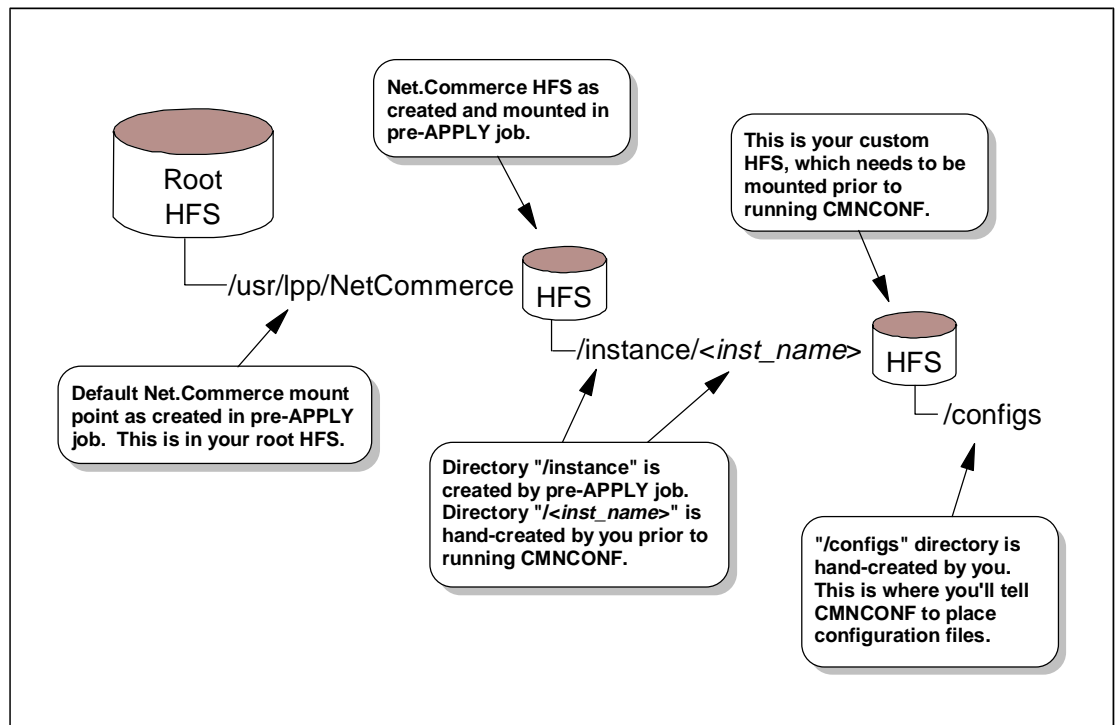
4. Allocate a new HFS file and mount it at the `/usr/lpp/NetCommerce/instance/<inst_name>` mount-point.

**Note:** the allocation of a separate HFS isn't a strict requirement of this process, but the point of this is to have all your custom files in a single HFS that can be easily un-mounted and copied.

5. Using OMVS or ISHELL, create a directory called `configs` off of the `/usr/lpp/NetCommerce/instance/<inst_name>` directory.

**Note:** this directory must have permissions of 755!

Here is what you have at this point:



- Run CMNCONF to configure your instance. When you get to the "System Configuration - Part 1 of 2" panel (under Option 1), type in the following for the "Server Controller Conf File" field:

```
/usr/lpp/NetCommerce/instance/<inst_name>/configs/<inst_name>.conf
```

The field is long enough to permit a directory and file of this length, but it will wrap:

```
CMNCF1----- SYSTEM CONFIGURATION - PART 1 OF 2 -----
COMMAND ==>

Instance Name           ==> <inst_name>

Web Server Instance Host Name
  ==> wsc.ericsson
Port Number             ==> 16490
Number of Processes     ==> 2

DB2 Subsystem Name      ==> DSN
DB2 Plan Name           ==> mserver
DB2 Database Owner      ==> cmnsrv
DB2 Database Name       ==> demomall

Password Storage        ==> %%DB2%% (%%DB2%% or %%SAF%%)
Server Controller Conf File ==> /usr/lpp/NetCommerce/instance/<inst_name>
/configs/<inst_name>.conf

Enter EN to return to the INSTANCE CONFIGURATION menu.
```

Note!

Note!

```
F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

- Continue in CMNCONF. On the “System Configuration - Part 2 of 2” page (still under Option 1), enter the following in the “HTML Path” field:

```
/usr/lpp/NetCommerce/instance/<inst_name>/configs
```

The panel will look like this:

```
CMNCONF2----- SYSTEM CONFIGURATION - PART 2 OF 2 -----
COMMAND ==>

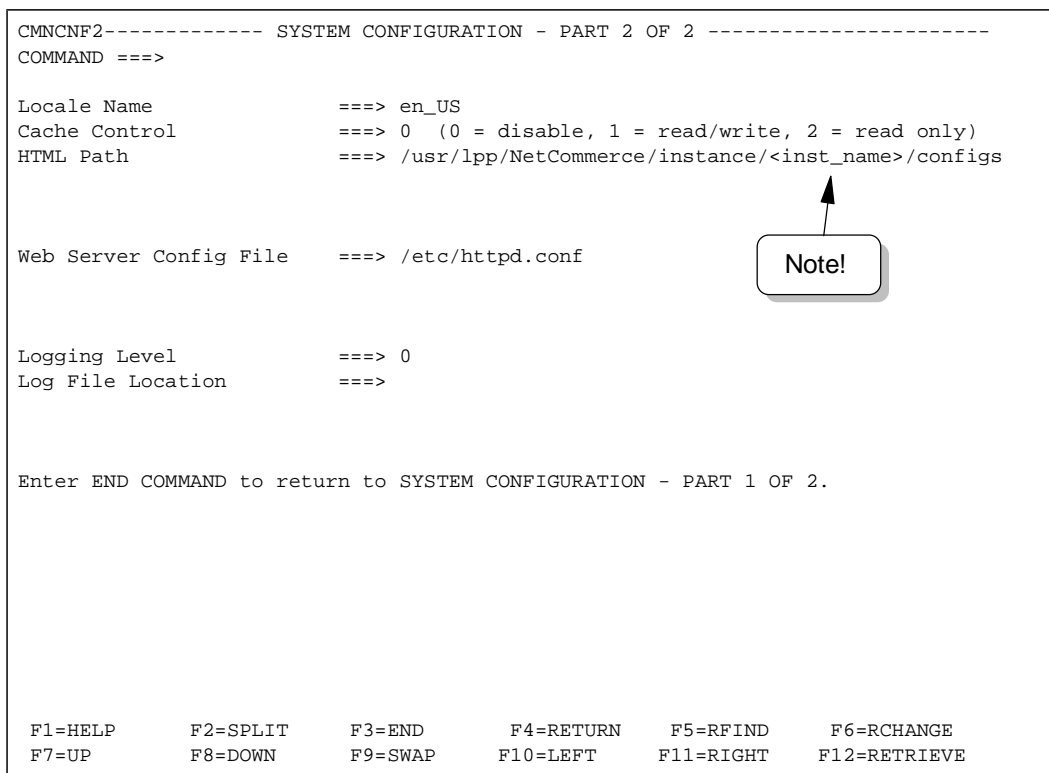
Locale Name           ==> en_US
Cache Control         ==> 0 (0 = disable, 1 = read/write, 2 = read only)
HTML Path             ==> /usr/lpp/NetCommerce/instance/<inst_name>/configs

Web Server Config File ==> /etc/httpd.conf

Logging Level         ==> 0
Log File Location     ==>

Enter END COMMAND to return to SYSTEM CONFIGURATION - PART 1 OF 2.

F1=HELP      F2=SPLIT    F3=END      F4=RETURN   F5=RFIND    F6=RCHANGE
F7=UP        F8=DOWN     F9=SWAP    F10=LEFT   F11=RIGHT   F12=RETRIEVE
```



- Complete the rest of the CMNCONF configuration, including database create and load.

One of the things CMNCONF will do is create a directory called `/teditor` under the `/instance/<inst_name>` directory. This is identical to the structure shown in the picture provided in “What is the *output* from the CMNCONF configuration utility?” on page 28.

- The sample JCL proc is located in the `hlq.SCMNSAMP` dataset. Its name is `CMNMSERV`. By default, the PROC statement looks like this:

```
//CMNMSERV PROC NETCPARM='',
// LEPARM='ENVAR("_CEE_ENVFILE=/etc/NC/-INSTNAME-.envvars")'
```

And a little lower in the proc the EXEC statement looks like this:

```
//CMNMSERV EXEC PGM=CMNSRVRC,
// PARM=('&LEPARM/&NETCPARM -i /etc/NC/-INSTNAME-.conf'),
// REGION=0M,TIME=(1440)
```

Note how in both sections LEPARM and PARM point to the default `/etc/NC` directory. Note also how the value for LEPARM (on PROC statement) gets placed into the PARM= on EXEC statement (with “&LEPARM”).

The problem here is that the PARM field has a limit of 100 characters. With a long value for LEPARM of:

```
/usr/lpp/NetCommerce/instance/<inst_name>/configs/<inst_name>.envvars
```

and one of equal length in the PARM field on the EXEC statement, the length quickly exceeds 100 character limit. You'll see an error that looks like this:

```
IEFC642I EXCESSIVE PARAMETER LENGTH ON THE EXEC STATEMENT
```

The way to get around this is to create a symbolic link lower in the HFS structure that points to the `<inst_name>.envvars` and `<inst_name>.conf` files deep in the directory structure for your instance.

10. Because the sample JCL proc uses `/etc/NC`, we will as well. Hand-create the directory `/etc/NC` and provide permission bits of 755. Then create the symbolic links in the `/etc/NC` directory:

```
ln -s /usr/lpp/NetCommerce/instance/<inst_name>/configs/<inst_name>.envvars
<inst_name>.envvars
ln -s /usr/lpp/NetCommerce/instance/<inst_name>/configs/<inst_name>.conf
<inst_name>.conf
```

11. Now modify the supplied JCL and point to `/etc/NC/<inst_name>.envvars` and `/etc/NC/<inst_name>.conf`.

### What is the difference between the various sample malls provided?

Four sample malls are provided with Net.Commerce. A brief description of each is provided here, along with what differentiates them from one another.

- **Metropolitan Mall** -- this is the classic mall implementation with seven stores defined in the database. The background color is ugly, and the shopping flow attempts to illustrate the use of every possible Net.Commerce command, which results in a very long and unwieldy process. But it's a favorite.

We recommend you start with this one when you first install Net.Commerce. It is the most simple to install and use.

- **Euro Mall** -- a variation on the Metropolitan, this mall illustrates the use of the Euro currency support, complete with a conversion rate table.
- **East West Food Mart** -- a frames-based grocery store with a simplified shopping flow as compared to the Metropolitan Mall. It uses quite a bit of Javascript, and if you plan to base your implementation off of this sample, you will need to be fluent in Javascript to understand what's going on.
- **Office Window** -- is a business-to-business scenario of an office supplies store. The trick with this mall is that shoppers are given access to only those stores and products that an administrator provides them. This would mimic a real environment where not everyone is authorized to purchase every product in an office supply catalog.

### Is there anything I need to do after I run CMNCONF?

Yes! You need to make sure that your Webserver's ID has DBADM authority over the database you just created. From SPUFI, issue the command:

```
grant dbadm on database <database> to <webserver id>
```

If you fail to do this, you will see the problem described in "Caching function not working and Performance Monitor not working; SQL -551" on page 88.

### **May I manually change the values found in the `ncommerce.conf` file?**

In some cases, yes. In some cases running CMNCONF is required. But not all parameters may be changed manually. To find out which parameters are okay to modify by hand (without running CMNCONF), refer to *Configuring and Getting Started*, Table 7 on page 59 of that manual.

### **May I have different people running CMNCONF at the same time?**

Theoretically, yes. Practically, no. It gets really tricky trying to keep the different people from stepping on each other if they're running CMNCONF at the same time. Yeah, it's possible; but as a general rule don't do it.

### **How many databases may I employ?**

Any single instance of Net.Commerce can only employ a single database. However, you can configure and start multiple instances of Net.Commerce, each one utilizing a separate database if you choose. In that scenario, you can employ as many databases as your resources (memory, disk, etc.) can support.

### **How many instances of Net.Commerce may I configure?**

As many as you'd like. But there are some things the CMNCONF program checks to make sure there isn't overlap, such as the port number specified and whether the RACF userid specified is used in a different instance with a different database name. So though there's really almost no limit, the CMNCONF program will enforce certain restrictions.

### **Can two instances of Net.Commerce share the same database?**

Yes. When you configure the instances using CMNCONF, simply point to the same database using the same owner and plan name. The instances will then share the same database.

### **Why would I ever want to configure more than one instance?**

Several reasons come to mind:

- To have a "test" and "production" Net.Commerce environment on the same system
- To provide different URLs for the individual stores in the same mall
- To provide different malls on the same system, each having its own running copy of Net.Commerce and its own database

Of course, you don't have to configure more than one instance. Many people will run with just one.

### **How can I stop a user from being able to browse the directory contents of the Net.Commerce root directory?**

By this, we mean someone putting a URL in their browser location window and getting a listing of all the files in the directory. Generally this is considered a security problem because users should not see the contents of the directories on your server.

There's a directive in the Webserver's `httpd.conf` file that controls whether users can browse directories. The directive is `DirAccess`, and you should make sure it is set to `Off`.

However, the error message a user will get if DirAccess is off and no other page is provided is pretty ugly, so you should automatically direct them to where you want them to go, which is your home page. Therefore, do this:

- Place an `ErrorPage` directive in your `httpd.conf` file. This will direct users who attempt specified actions to a named homepage:

```
ErrorPage    dirbrowse    /homepage.html
```

In this example, users who by mistake (or otherwise) actually pointed their browser to `http://<your_server_name>/` or `http://<your_server_name>/some_directory` would be directed to `homepage.html`.

### How do I protect other Net.Commerce files?

At the top of `httpd.conf` after running the CMNCONF configuration utility you will find directives that look something like this:

```
Protect /*.conf <your_host_name> {
ServerId      Private_Authorization
AuthType     Basic
GetMask      All@(*)
PutMask      All@(*)
PostMask     All@(*)
Mask         All@(*)
}
```

This causes the Web Server to "protect" all files with a type of `*.conf`. These statements cause the Web Server to ask the user to enter a valid Web Server administration and userid password if a request is made to directly access any file ending in `.conf`, such as `ncommerce.conf`. The Net.Commerce document root directory also contains other files users should not be able to access:

- Net.Data initialization file called `db2www.ini`.
- A backup copy of `db2www.ini` called `db2www.iniold` (if CMNCONF has been run to modify the instance)
- Net.Commerce environment variables file called `ncommerce.envvars`

We recommend that you protect these files as well.

Protection of all `*.ini` files (as an example) can be achieved in a similar fashion to protecting all the `*.conf` files by adding the following statements to `httpd.conf`:

```
Protect /*.ini <your_host_name> {
ServerId      Private_Authorization
AuthType     Basic
GetMask      All@(*)
PutMask      All@(*)
PostMask     All@(*)
Mask         All@(*)
}
```

You can protect any sensitive filetype in this fashion. You should protect `*.inidel`, `*.iniold`, `*.confold`, `*.confdel` and `*.envvars` as they are also sensitive configuration files.

You may also block access to these files with the `Fail` directive in the webserver's configuration file. The format of this directive is:

```
Fail /*.ini*
```

What this says is "fail any attempted direct access to files whose extension starts with ".ini". The user at the browser who attempts this direct access will get a webserver error message saying access is denied.

If you want to *really* lock down your site, add the following directives:

```
Fail /cgi-bin/msrvr/;execmacro/ncadmin/*
Fail /msprotect/msrvr/;execmacro/ncadmin/*
Fail /cgi-bin/ncommerce3/ExecMacro/ncadmin/*
Fail /msprotect/ncommerce3/ExecMacro/ncadmin/*
Fail /*.ini*
Fail /*.prts*
Fail /*.envvars*
Fail /*.confold*
```

**Note:** Be careful about where you place these new `Protect` directives in the `httpd.conf` file. If you re-run `CMNCONF`, what you added may be overwritten or removed. Your custom directives should fall *outside and after* the `Net.Commerce` block found at the top of the file.

### May I turn off SSL?

A qualified "yes." There are three main aspects of `Net.Commerce` to which this question applies:

- SSL for the shopping functions
- SSL for the NCADMIN administrative functions
- SSL for the Store Creator Utility

Let's look at each in turn:

#### **Turning off SSL for the shopping functions**

This is controlled by the settings for a command in the NCADMIN "Command Security" function. Each command can be set separately in this fashion. Then, when that command is seen by `Net.Commerce`, SSL will be invoked.

To access this feature, start NCADMIN (see "What is the `Net.Commerce` Administrator (NCADMIN) facility?" on page 51) and click on the "Site Manager" button in the navigation frame. Then click on the "Command Security" button.

What you'll find is a listing of each command with an indication for SSL as well as authentication. If you wish to turn off SSL, then go through the list and update those commands that by default have `SSL=Yes` to `SSL=No`.

#### **Turning off SSL for NCADMIN**

This is accomplished by editing two HTML files:

```
/usr/lpp/NetCommerce/html/en_US/ncadmin/index.htm
/usr/lpp/NetCommerce/html/en_US/ncadmin/index.html
```

**Note:** The two files are identical to one another in content, but their names are slightly different (notice the "l" at the end of one and not the other). The following instructions apply to both files

In those files a hard-coded `https` SSL invocation is written. You need to go in and change those to `http` (drop the "s"):

```
000019 <SCRIPT Language="JavaScript">
000020
000021   window.location = "https://" + window.location.host
000022
000023 </SCRIPT>
```

Change this to "http"

The text goes well beyond the right side of the screen, so be careful when you remove the "s" so you don't leave a "hole" in the text string.

Then when you invoke the NCADMIN utility again it'll start without invoking SSL.. However, you can't use the "Store Creator" function, which has a hard-coded SSL invocation.

### ***Turning off SSL for the Store Creator***

This can't be turned off. SSL invocation appears to be a hard-coded function of this utility.

### **May I use a non-SSL port other than 80?**

You may, but it's not recommended. The standard port for HTTP requests is 80, and for shoppers to use something other than 80 would require them to code the port on the URL they type into the location window.

That being said, for testing in your environment, here's what you'd do:

1. Change the Port directive in `httpd.conf` file to specify a port other than 80 (or you may, as an alternative, use the `-p` parameter on `ICSPARM=` of the JCL start procedure to point to a different port and thus override the value found in `httpd.conf`).
2. Restart your Webserver
3. In the location window of your browser, point to Net.Commerce with a different port number:

```
http://<host_name>:8080/demomall/basemall.htm
```

### **May I use an SSL port other than 443?**

No. Net.Commerce is looking for port 443.

### **What's the difference between %%DB2%% and %%SAF%% for authentication?**

When a user "logs in" to Net.Commerce, that userid/password combination provided is checked against a repository where the information is maintained. If the userid/password from the browser matches what's in the repository, then the person is allowed to access the page. If not, then they are rejected.

This parameter, specified in the CMNCONF configuration program, tells the Net.Commerce instance where password validation will be performed: in the DB2 table (the SHOPPER table to be specific), or through the Security Access Facility (SAF) interface (or RACF, but other programs equivalent in function will also work).



There's a very crucial difference between the two, and it has to do *with who does the authentication*:

	%%DB2%%	%%SAF%%
Who performs the authentication?	Net.Commerce	Webserver

The reason why this difference is highlighted is because the way you trigger initial authentication is different:

	%%DB2%%	%%SAF%%
Authentication triggered by:	msprotect in URL	By direct invocation of the Net.Commerce Logon command, or based on "Command Security" setting for the command invoked

### If I use %%SAF%%, does that mean the shoppers don't need to be registered?

The shoppers will still need to be registered. Remember, all the %%SAF%% does is have the Webserver go against SAF to authenticate who they are. From that point forward, Net.Commerce still needs to operate against the record in the SHOPPER table corresponding to that shopper. And if they authenticated with `userid=SMITH`, then Net.Commerce will for the life of that session consider that shopper "SMITH" and be looking for a record in SHOPPER that matches "SMITH".

Therefore, %%SAF%% does *not* eliminate the need for a registered user.

### Why use %%SAF%%?

Only if you want to restrict who can access your site to a known and limited set of users as defined in your SAF facility. In an environment where you have some unknown number of shoppers on the Internet accessing your site, you'll want to use %%DB2%%. %%SAF%% is more intended for the business-to-business model.

### How do I use the %%SAF%% authentication function?

Here's how you'd enable the %%SAF%% function:

- Apply the PTF UQ34468.
- Run CMNCONF and provide %%SAF%% in the "Password Storage" storage field of "System Configuration (1 of 2)". Don't forget to run Option 2 ("Access Control" so this change makes it into the appropriate configuration files.
- Place the following statement in the Webserver's `httpd.conf` configuration file, just below the Net.Commerce block at the top of the file:

```
Authentication BASIC
    /usr/lpp/NetCommerce/lib/nc_cgi_icapi.so:cpp_nc_auth
```

- Place the following in the Webserver's `httpd.conf` configuration file, *just below* the Net.Commerce block at the top of the file:

```
Protect /msprotect/* <hostname> {
ServerId      Private_Authorization
AuthType      Basic
GetMask       All@(*)
PutMask       All@(*)
```

```

PostMask      All@(*)
Mask          All@(*)
PasswdFile    %%SAF%%
}

```

Where <hostname> is the host name of your instance.

- Restart the Webserver
- At the point in the shopping flow where you want to authenticate the shopper, code an URL with the following:

```

https://<your_host>/msprotect/ncommerce3/<Net.Commerce command>

```

**Note:** it is very important that you directly invoke SSL with the "s" on the "http" portion of the URL. Failure to do that will result in a second login request the first time SSL is invoked after the %%SAF%% authentication. By tying the authentication with SSL like this, you avoid that "double login".

### What is needed to run the Product Advisor function?

Product Advisor is an intelligent catalog tool. It does three things, all designed to minimize the time it takes a shopper to find a product once they have arrived at the site:

- Product Advisor enables inexperienced users to find products they seek, through a Q & A type session (this is known as the "Sales Assistant" metaphor)
- Product Advisor enables experienced users to quickly find their product of choice (this is known as the "Product Exploration" metaphor)
- Product Advisor allows all users to compare products side by side in a "snapshot" fashion (this is known as the "Product Comparison" metaphor).

Most customers today are not using the Product Advisor function of Net.Commerce at all. The question of "What is needed to run the Product Advisor function?" can be divided into two questions:

Question	Go To
<p><i>"What is needed to enable shoppers to use the Product Advisor function?"</i></p> <p>Shoppers need only a browser to access an enabled Product Advisor function. The "Next Generation" store in the Demomall provides the PA (Product Advisor) functions, but there's some work you need to do to switch it on.</p>	Page 40
<p><i>"What is needed to administer and create Product Advisor Catalogs?"</i></p> <p>Creating your own catalog for Product Advisor is a more complex task than simply using the shopper's function.</p>	Page 44

Please refer to the appropriate section of the FAQ for further information.

### What is needed to enable shoppers to use the Product Advisor Function?

The Product Advisor function is basically provided by Java Servlets that need to access DB2. In addition to the base software requirements for Net.Commerce, to run Product Advisor for shoppers you must also have the following components installed and configured on your OS/390 server:

- Java Servlet Support
- DB2 Call Level Interface (CLI)
- Java Database Connection (JDBC)

### **Java Servlet Support**

First ensure that the WebSphere Application Server is listening on port 9090. This can be seen on the output of the TSO `netstat` command. You should see the WebServer address space listening on three ports by default:

- port 80 (for http requests)
- port 443 (for https requests)
- port 9090 (for WebSphere Application Server Administration).

An example output of the command (as run from ISPF option 6):

```
tso netstat
:
:
WEBSRV51    00123 0.0.0.0..80          0.0.0.0..0
  Listen
WEBSRV51    00124 0.0.0.0..443         0.0.0.0..0
  Listen
WEBSRV51    00127 0.0.0.0..9090        0.0.0.0..0
  Listen
:
:
```

If your WebServer is not listening on port 9090, make sure that you have installed the WebSphere Application Server correctly, and run the setup shell script `/usr/lpp/WebSphere/AppServer/config/postinstall.sh`.

Try running the sample SnoopServlet from the URL

`http://<your_server>/servlet/SnoopServlet`. If the servlet runs and returns "Request Information" then your base Java Servlet support is working.

There are basically two ways to configure the WebSphere Application Server to run the Product Advisor Java servlets. One is to edit the configuration files in the HFS, the other is to use the WebSphere Application Server Administration GUI. We look at both of those options here:

### **Editing the HFS files**

Edit the file:

```
/usr/lpp/WebSphere/AppServer/properties/server/servlet
/servletservice/jvm.properties
```

Add the following library to the `ncf.jvm.classpath` variable:

```
ncf.jvm.classpath= .....
  /usr/lpp/NetCommerce/www/Servlets/Public/
```

Edit the file:

```
/usr/lpp/WebSphere/AppServer/properties/server/servlet
/servletservice/servlets.properties
```

Add the following to the "Servlets added by user" section

```

servlet.icviewer.code=icviewer

servlet.icviewer.initArgs=configfile=ncommerce.conf,
  content=text/html,<net.commerce machine ip name>
  =/usr/lpp/NetCommerce/html/en_US/<net.commerce
  instance name>

servlet.icviewer.description=Product Advisor

```

**Note:** the `servlet.icviewer.initArgs` should all be on one line.

## Updating the WebSphere Application Server through the Administration GUI

We were going to paste in some bitmap images of the screens here, but it would have resulted in the PDF file being too large, so we'll rely on the descriptive power of words to walk you through this.

1. Point your browser at your webserver with the following URL:

```
http://<your_host>:9090
```

**Note:** 9090 is the default port on which Webserver's administrative plugin listens.

2. The login screen will appear. Supply the userid and password which, by default, is `admin` and `admin` respectively. Click on the "Login" button.
3. This will bring up a panel that lists all the services on your webserver and their status. Click on the "Manage" button.
4. Up will pop a sub-window with four colorful buttons across the top reading, in order, "Setup", "Monitor", "Security" and "Servlets." Click on the "Servlets" button.
5. The next window that will appear will have an expanded tree in the left frame of the window with "Servlets" at the top of the tree, and "Add" as the next item down the tree. Click on the "Add" item.
6. The right-hand frame of the window will now change and show a "Add a New Servlet" title. Two windows will also be on the window:
  - Servlet Name: *Product Advisor*
  - Servlet Class: *icviewer*

Provide the values as shown above. Then click on the "Add" button at the bottom of the screen.

7. The new "icviewer" servlet will appear in the expanded tree in the left frame. Highlight "icviewer" by clicking on it. The right-hand frame of the window will change to have a two-tab object. The two tabs are titled "Configuration" and "Properties". Click on the "Properties" tab.
8. The "Properties" tab will have an input section that looks like this.

Name	Value
Configfile	ncommerce.conf
Content	text/html
<your host IP address>	/usr/lpp/NetCommerce/html/en_US/<inst>

You supply the "values". The "Configfile" will be `ncommerce.conf`. The "Content" will be `text/html`. Supply the directory of your instance's "document root" in opposite the name that is your host IP address. Your instance's document root will be:

```
/usr/lpp/NetCommerce/html/en_US/<instance_name>
```

9. The "Save" button should now be un-greyed. Click on "Save" to save the values.
10. Close the Webserver and restart it to pick up the changes.

### ***Install Java Database Connectivity Support***

This support is a separate feature of DB2, usually installed by your DB2 administrator into directory `/usr/lpp/db2/db2510` (for DB2 version 5).

### **Create A CLI initialization (DSNAOINI) file and make it accessible to the Web Server**

This should be a sequential file or member of a PDS. Make sure that ISPF edit has "num off" when creating this file.

Our DSNAOINI file looks like this:

```
; This is a comment line...
; Example COMMON stanza
[COMMON]
MVSDEFAULTSSID=DDH3

; Example SUBSYSTEM stanza for DSN subsystem
[DDH3]
MVSATTACHTYPE=CAF
PLANNAME=DSNACLI

; Example DATA SOURCE stanza for sample data source
[OS390S27]          <<--- This is your DB2 location name
AUTOCOMMIT=1
CONNECTTYPE=1
```

Make this file available by adding it to startup JCL of the WebServer and by adding a reference to it in the `httpd.envvars` file.

Here is an example of the JCL change:

```
//DSNAOINI DD DSN=CMN.V312.SCMNCINI,DISP=SHR
```

Here is an example of the `httpd.envvars` update:

```
DSNAOINI=CMN.V312.SCMNCINI
```

### **How do I find out my DB2 location name?**

You can ask your DB2 administrator, but there are a number of ways to find it out:

- Probably the easiest is to use SPUFI to issue an SQL command against any table, like this:

```
select current server from <table>;
```

DB2 has a default table at install time, so issue the command:

```
select current server from sysibm.sysdummy1;
```

and the output will display your DB2 location name. This assumes your DB2 system is not connected to other DB2 systems.

- Look in syslog (using something like SDSF) at the time when DDF starts. When DDF starts a message appears giving the DB2 location name. The location name also appears in the sysout of the DB2 master address space.
- Run a DB2 "print log job". This job will print the contents of your DB2 BSDS which will include your DB2 location name. The program for your EXEC PGM card is DSNJU004, this Utility is documented in the DB2 Utilities Guide & Reference

### ***Install the DB2 Call Level Interface (CLI)***

This support is a separate feature of DB2, usually installed by your DB2 administrator and involves running some JCL jobs and SPUFI scripts.

### ***Net.Commerce Product Advisor Bind Job***

The Product Advisor is a case where the Websphere Application Server is being used, so the discussion in "What do I do if I'm using the DSNTIJCL DB2 CLI BIND job?" on page 24 applies. Please refer to that section for details.

### ***Grant SYSADM to the userid that the WebServer runs under (not the PUBLIC userid)***

Through SPUFI, issue:

```
GRANT SYSADM TO webserver_userid
```

This is the userid under which the webserver is running (in our case it was WEBSRV). This authorisation is required to allow the Product Advisor Java Servlet to change the current SQL userid to the userid who owns the Net.Commerce database (in our case this userid is CMNSRV).

### **What is needed to administer and create Product Advisor Catalogs?**

The "Next Generation" store in the Demomall is a good illustration of what the Product Advisor functions provides, but the product and product attribute information found in Next Generation won't match what your site will require. What is needed is a way to construct your own "catalog" for Product Advisor. The Product Advisor Catalog Builder is provided for that purpose.

The person who creates the Product Advisor Catalog is known as the Product Advisor Administrator. This administrator needs some extra software installed on their workstation. The tool that creates the Product Advisor catalogs (Catalog Builder) is actually a Java application that gets downloaded to the browser. This java application is used to create the input used by the three metaphors described earlier. The application needs to update DB2 tables.

With DB2 v5 for OS/390 there is a restriction that the Java application running on the workstation cannot directly update DB2 tables on the OS/390 system, so the workstation needs DB2 Connect installed to make the connection, and the host DB2 needs DDF to be running. DB2 Connect Personal (as well as the more

expensive Enterprise) Edition is acceptable. The requirement for DB2 Connect may go away with DB2 V6 (UDB), but that has yet to be confirmed.

The Administrator also needs to make some changes to their java environment. All of the information needed to configure this is located in the *Configuring and Getting Started* Guide manual in the "Product Advisor Administration Workstation Support" section.

---

## Operating Net.Commerce

### How do I start Net.Commerce?

A sample JCL start procedure is provided in the dataset `hlq.SCMNSAMP(CMNMSERV)`. Copy that member to your system proclib, make the modifications as indicated in the comments of the JCL, and start the task from the MVS console.

### How do I stop Net.Commerce?

From an MVS console issue the following command:

```
P CMNMSERV
```

It will generally come down gracefully. In the event that it's acting stubborn, kill it with:

```
C CMNMSERV
```

### Is there any requirement to start the Webserver before I start Net.Commerce?

It's recommended that Net.Commerce be started first, and then start the Webserver. The reason is that the ports file (`ncommerce.prt`s) is not recreated until the Net.Commerce server is started. If you start the Webserver first and there is a hit before the NC server is started, the old ports file will be used (or there will be no ports file at all, resulting in some error message).

DB2 must be up first, however.

### What console message will I see to indicate the Webserver is up and running?

```
IMW3536I SA 436207633 0.0.0.0:80 * * READY
```

### What console message will I see to indicate Net.Commerce is up and running?

```
CMN0411I THE NET.COMMERCE SERVER IS READY FOR CONNECTIONS 175
```

### How should I test my Net.Commerce instance to make sure everything is working properly?

It's important to differentiate the *initial* testing done to see if your installation and configuration is correct from the testing required to insure your *customization* is functioning properly. This item will only speak to testing the initial implementation.

There's five basic things you need to test:

- Access to Net.Commerce static HTML pages
- Operation of SSL
- Access to Net.Commerce database
- Invocation of SSL by Net.Commerce
- Operation of authentication routine

To test this, issue the following URLs from the browser after Net.Commerce has been started:

1. `http://<host>/demomall/basemall.htm`
2. `https://<host>/demomall/basemall.htm`
3. `http://<host>/cgi-bin/ncommerce3/ExecMacro/mall_dir.d2w/report`



4. `http://<host>/cgi-bin/ncommerce3/InterestItemDisplay`
5. `http://<host>/ncadmin`

Here's what each test accomplishes:

1. This URL will access the Net.Commerce Metropolitan Mall home page. It is a static HTML page. If you get it, it shows that the Webserver is running, and the changes put into the Webserver's configuration file by CMNCONF have taken effect (the PASS directives added by CMNCONF point to the file `basemall.htm`).
2. By placing the "s" on the `http://` portion, you're telling the Webserver to invoke SSL for this request. For a Netscape browser the indication you're in SSL mode is the little padlock symbol located at the bottom left of the screen closing. If you successfully go into SSL mode, then SSL is enabled properly on your Webserver.

**Note:** close the browser and reopen it after this step.

3. This URL will execute a Net.Commerce command, and will invoke Net.Data to display a page. If you get the "Mall Directory" page, then you've verified that a) Net.Commerce is running, b) the Webserver successfully passed the command up to Net.Commerce, and c) that Net.Commerce is able to access the DB2 system.
4. This URL will also execute a Net.Commerce command, and by default this command has SSL flagged as required for the command. That means that Net.Commerce will invoke SSL when this command is accessed. If you get a page that says something like "Shopping Cart for Unknown Shopper" *and* the padlock symbol is locked, then you've verified that Net.Commerce's ability to invoke SSL is working.
5. Finally, issue the command to access the Net.Commerce Administrative function. A login panel will be displayed. The default userid is `ncadmin` and the default password is `ncadmin`. If you get the main panel of this application, it means that a) the authentication routine worked, b) the administrative function worked, and c) the ability to get to the DB to verify the userid and password worked.

### **Is there some way to dynamically pick up changes to the Webserver's configuration file?**

Yes. You may issue the following MVS console command to restart the Webserver and pick up changes in the `httpd.conf` file:

```
F <task name>,APPL=-RESTART
```

Note the "dash" between "APPL" and "RESTART".

### **Is there some way to dynamically pick up changes to the Net.Commerce configuration files?**

No. You must stop and start again the Net.Commerce instance to pick up changes made to either the "Server Controller Configuration File" or the "Server Configuration File."

If you have the requirement for a true 24 x 7 x 365 site, then you will require multiple Net.Commerce instances running, sharing a common database. Traffic to

the site is balanced across the multiple instances of Net.Commerce using some IP traffic balancer (IBM Net.Dispatcher being one). That would allow you to make a change to one instance's configuration file and then stop and start again that instance. While that instance is down, the other instances can continue servicing the traffic (provided the resources for that remaining instances is sufficient to handle the load). When the modified instance comes back on line, it may then step in and continue accepting traffic.

This is an example of where a Sysplex configuration would be ideal.

### **Is there a way to trace Net.Commerce activity?**

There are two components of Net.Commerce that have separate tracing facilities:

1. The Net.Commerce Director trace, which traces the activity of the GWAPI API code that runs in the Webserver's address space.
2. The Net.Commerce Server trace, which traces the activity of the Server Controller and the Server processes.

For both of those tracing facilities there are four levels tracing detail that can be enabled:

- Level **0** -- errors only
- Level **1** -- errors, plus status information
- Level **2** -- errors, status, plus debug information
- Level **3** -- errors, status, debug, plus tracing information

Which you choose to use is really a matter of what information you're looking for. Levels 0 and 1 are probably all you'd need for typical problem determination stuff you do yourself. Levels 2 and 3 contain information of more value to developers. Level 3 in particular spits out enormous amounts of information, so you should never allow that run for any considerable length of time.

The tracing may be enabled either dynamically or statically. Here's how you do that:

#### ***Director Trace -- starting statically***

1. Edit the Webserver's environment variable file (`httpd.envvars`).
2. Insert the directive `CMN_TRACE=n` where "n" is the level of logging you desire.
3. Stop and start again the Webserver.

The tracing will begin at the time the Director is started.

**Note:** you may dynamically turn off tracing using the dynamic control facility under NCADMIN (see next item).

#### ***Director Trace -- starting dynamically***

1. The Webserver and Net.Commerce must be running
2. Connect to and logon to the NCADMIN utility. The URL to access this facility is `http://<your_host>/ncadmin` and the userid you should use is "ncadmin".
3. Click on the "Site Manager" button in the navigation frame

- Click on the "System Monitoring" button. The resulting screen will look like this:

### System Monitoring

Performance Monitor Output

Net.Commerce Server : <your\_host\_name>  
Time: (date and time)

No Net.Commerce Command Performance Data.

---

No Net.Commerce Error Data.

---

Trace, Cache and Performance Monitor Settings

Director Trace:  Enable  Disable

Director Debug:  Enable  Disable

Cache:  Enable R/W  Enable R/O  Disable

Purge Cache Data:  Yes  No

Performance Monitor  Enable  Disable

Purge Performance Data  Yes  No

Frequency of Saving Performance Data

- Click on the "enable" radio buttons for "Director Trace" and/or "Director Debug"
- Click on the "Submit" button.

The tracing will be dynamically started at that point in time.

**Note:** this facility turns on *just* that level you specify, and doesn't include the "lower levels" like the static method does. If you turn on tracing using the NCADMIN interface, and what you *really* want is tracing and debug, you would have to enable both radio buttons and click on the "submit" button.

#### **Server Trace -- starting statically**

- Run the CMNCONF utility
- Set the "Logging Level" option on "System Configuration" (Option 1, panel 1 of 2) to the level of logging you desire.
- Save the configuration by exiting CMNCONF
- Restart Net.Commerce

**Note:** manually changing the value of MS\_LOGLEVEL (along with MS\_LOGPATH) in the configuration files *is* permitted, even though the *Configuring and Getting Started* manual suggests otherwise. Generally you want to use the CMNCONF program for setting this value, but if you want the value in /etc/NC/<inst\_name>.conf to be different from that in /usr/lpp/NetCommerce/html/en\_US/<inst\_name>.conf, then you'll need to set them manually.

#### **Server Trace -- starting dynamically**

- Get to a system console

## 2. Issue the modify command:

```
F <NC started task name>,APPL=<option>
```

Where <option> is:

```
traceon -- turns tracing on
traceoff -- turns tracing off
debugon -- turns debugging on
debugoff -- turns debugging off
```

**Note:** this facility turns on *just* that level you specify, and doesn't include the "lower levels" like the static method does. If you turn on tracing using the modify command, and what you *really* want is tracing and debug, you would have to issue both commands.

### Where does the output of the tracing function go?

Where the output goes is summarized in the following chart:

Trace	Output Goes ...
Director Trace	SYSPRINT of the Webserver's started task
Server Trace	The "Server Controller" (the "main daemon") sends its output to SYSPRINT of the Net.Commerce started task
	The "Server Processes" (the "child daemons") send their output to SRVOUT of the Net.Commerce started task
<b>Note:</b> you may also direct the output of the server trace to an HFS file by specifying a "Log File Location" parameter on the "System Configuration" option of CMNCONF (option 1, panel 2 of 2). The parameter in the configuration file that directs the output is MS_LOGPATH. If MS_LOGPATH exists, it will take precedence over any DD_NAME specification for both the Server Controller as well as the Server Processes. Each server process will have its own file created, and the Server Controller will have its own file.	

As stated, without a MS\_LOGPATH value specified, the Server Processes output will go to whatever dataset is specified on the SRVOUT DD card in the JCL start procedure for Net.Commerce.

You may direct the output to a different DD card by specifying a DD\_NAME parameter in the ncommerce.conf file, and then providing that DD card in your JCL start proc. See *Configuring and Getting Started* for the format of the DD\_NAME parameter of the configuration file.

The Server Controller output is unaffected by the DD\_NAME parameter. It will always go to SYSPRINT, unless MS\_LOGPATH is specified.

### Is the "-tr" parameter in the JCL proc used for tracing still there?

No. It's been removed. Same goes for the "-d" parameter.

### I need to change my Net.Commerce environment variable file. What do I need to do?

This gets tricky because the file is in several different places:

- The ncommerce.envsamp file, located in /usr/lpp/NetCommerce/install, is the one used by CMNCONF as its model for all configurations.

- The `ncommerce.envvars` file, located in `/etc`, is what CMNCONF uses so it may run properly (in other words, the CMNCONF program needs the environment variables, and it looks for them in `/etc/ncommerce.envvars`).
- The `<instance_name>.envvars`, located in `/etc/NC` (which is a configurable directory location and may be different based on what you provided to CMNCONF). This is the environment variable file used by the "Server Controller" of Net.Commerce.
- The `ncommerce.envvars` file, located in `/usr/lpp/NetCommerce/html/en_US/<instance_name>`, is the one used by the Net.Commerce Server instance named in `<instance_name>`.

If you haven't yet configured any instances of Net.Commerce, and you haven't yet copied `/usr/lpp/NetCommerce/install/ncommerce.envsamp` to the `/etc` directory, then all you need to do is make the modification in `/usr/lpp/NetCommerce/install/ncommerce.envsamp`.

If you've run CMNCONF and you have some instances already configured, then you'll have to make sure whatever change you wish to make is propagated to all those different locations.

### What is the Net.Commerce Administrator (NCADMIN) facility?

The Net.Commerce Administrative function is a set of functions and utilities intended to assist you in managing the Net.Commerce site. It contains three major functions:

- **Site Manager/Store Manager** -- a set of Javascript forms that display information from the database and let you modify or add information without having to know the database structure that well. For example, the act of adding products to a store can be accomplished using Store Manager. A series of panels will provide fields for input that, when saved, will place information into the appropriate database tables to create the products and tie them to the appropriate category in the right store.
- **Template Designer** -- a Java applet that provides an object-oriented tool for the creation of Net.Data macros. The key between this tool and any other HTML design tool is the Template Designer is well aware of the database structure of Net.Commerce, and allows you to create SQL queries that will pull information from the database and dynamically create the HTML page.
- **Store Creator** -- another Java applet that aids the user in creating a simple store. It uses a set of model store formats and "walks" the user through several panels to create the store. Once done, the information that comprises the store (Net.Data macros, HTML files, information in the database) is added to the site to actually create the store.

### Must I use the NCADMIN function?

Not necessarily. Much of the function provided by NCADMIN can be accomplished through other means. For example, the creation of a Net.Data macro can be done by hand if you're skilled in Net.Data. Adding information to the database can be done with SPUFI if you know the database structure and data requirements.

Many people start out with NCADMIN, and once they become familiar with the workings of Net.Commerce they find other ways of accomplishing certain tasks more quickly (for example, by using the Mass Import Utility).

## How do I access the NCADMIN function?

The URL you access is:

```
http://<host_name>/ncadmin/
```

**Note:** by default you must have SSL enabled to use the NCADMIN function. Turning off SSL for the NCADMIN function is possible. See "May I turn off SSL?" on page 37 for information on disabling SSL for your Net.Commerce system.

## What is default userid and password for the NCADMIN facility?

The userid is `ncadmin` and the password is `ncadmin`.

## How do I change the default password for the Net.Commerce administrator ID?

Here's the process you'd use:

1. Log onto the Net.Commerce Administrator function using the default ID of `ncadmin` and default password of `ncadmin`.
2. Click on the **Site Manager** button in the navigation frame.
3. Click on the **Shopper Information** button.
4. Click on the **Search** button at the bottom of the screen. That will result in all of the shoppers in the DB2 database being presented in a list format in the "feedback" frame at the bottom of the screen. It's a small frame, so you will probably need to resize it.
5. Locate the `ncadmin` ID in the list and click on the name. This will populate the fields of the form with the information for the `ncadmin` ID.
6. Fill in the **Password** and **Password Confirmation** fields with the new password of your choice.
7. Scroll down the form and fill in all the fields whose titles are shown in bold type. The fields that are *required* are:
  - Address
  - City
  - State
  - Country
8. Click on the **Save** button at the bottom of the screen.
9. A window will pop up that says, "An existing shopper record will be updated. Do you want to continue?" Click on **OK**. If the update was successful, you will receive a message in the "feedback frame" that, "The record has been successfully updated in the database."

## What do I do if I forgot the password for the default ncadmin administrator ID?

If you have another ID defined that also has "Site Administrator" authority, you can use that to reset the password by following the instructions provided for "How do I change the default password for the Net.Commerce administrator ID?" on page 52.

However, if "ncadmin" is the only administrator ID you have and you've forgotten the password, you have to go through a few gyrations to get the password reset:

1. Go to the directory `/usr/lpp/internet/sbin`. That's where the Webserver's `htadm` utility resides.

2. Create a password file using the following command:

```
htadm -create /<your directory>/<file name>
```

This file is a temporary file, so you don't need to worry about putting it someplace "official", nor do you need to name it anything special. So put it in your home directory and call it "pwfile.txt".

3. Generate the password with the following command (command split across lines in this document for illustration purposes; type on one line in OMVS):

```
htadm -adduser /<your directory>/<file name>  
ncadmin ncadmin default administrator
```

Where /<your directory>/<file name> is the password file created in the previous step, the first instance of "ncadmin" is the userid, *the second instance of "ncadmin" is the password*, and "default administrator" is the real name of the person. The password is the only meaningful value you supply, though htadm requires all three.

In this example, the password is being set to ncadmin. You can set it to whatever value you want.

4. Edit the password file. The format will look like this:

```
ncadmin:n3sKcTL0l1h0bB:default administrator
```

Notice the three fields, separated by the colons? The first field is the ID, the second (seemingly garbled string) is the password, and the third field is the "real name" of the person.

5. Mark out just the password field (the encrypted string -- but *do not* include the colon field separators) and copy it to the clipboard.
6. Using SPUFI, determine the SHRFNBR value for the ncadmin ID. You can use this SQL:

```
select SHRFNBR, SHLOGID  
from <owner id>.SHOPPER  
where SHLOGID='ncadmin';
```

By default it should be 0, but do this step just to be sure.

7. Now, using the password field that you copied to the clipboard a few steps ago, create the following SQL to update the SHLPSWD field of SHOPPER:

```
update <owner id>.SHOPPER  
set SHLPSWD='<new password>'  
where SHRFNBR=<shrfnbr value>;
```

That'll update the password field and allow you to again access the "ncadmin" ID. At this point you can erase the password file you created using the htadm utility ... it's no longer used for anything.

## Can any registered shopper access the NCADMIN functions?

No. Shoppers registered into the Net.Commerce database are granted authority to be administrators by an entry in the ACC\_USRGRP table. Entries in that table tie a user to a "access group" (as defined in the ACC\_GROUP) table. If no record exists in the ACC\_USRGRP table for a given record in the SHOPPER table, that shopper is *not* an administrator and can not use the NCADMIN functions.

## How do I create an administrator ID?

Here's the process you'd follow:

- Log onto the NCADMIN function with an ID that presently has access to the Site Manager function (the default ID of `ncadmin` has this authority)
- Click on the **Site Manager** button in the navigation frame
- Click on **Access Control**
- Fill in all the fields with a title in bold letter ("Administrator's ID", "Password", "Password Confirmation", "Last Name" and "First Name")
- Click on the **Save** button at the bottom of the screen.
- You'll get an informational window that says, "A new administrator record will be created. Do you want to continue?" Click on **OK**.
- You'll get a message in the feedback frame that says, "A record has been successfully added to the database."
- Click on the Access Assignment button. This is where you'll tell Net.Commerce what kind of authority the new administrator's ID will have.
- Select the group authority you wish for this ID. The one labeled "Site Administrators(Mall)" will give the ID the same authority as `ncadmin`.
- Click on **Add Groups**
- The group will now appear in the "Groups Assigned to Administrator" window. Your new ID has the authority assigned to that group.

## How can I prevent one merchant administrator from seeing the information for another merchant?

The process involves setting the Access Assignment for the ID in the Site Manager function of NCADMIN. To restrict an ID to a particular store, you should make sure the ID only has for its "Groups Assigned to Administrator" the store(s) over which you wish that administrator to have control.

## What kinds of Net.Data macros can the Template Designer create?

The Template Designer is limited to product pages and category pages, and simple macros that draw individual pieces of information out of the database.

You could not develop, for example, a shopping cart page using Template Designer because it doesn't have the built in functions to develop the SQL necessary to extract the shopping cart records for a given shopper. But it does have the SQL built in to product a product table and a category table. And it does give you pulldowns to access nearly every field in the database.

The Template Designer may also be used to create HTML pages and Product Advisor pages.



## Why do Template Designer-created Net.Data macros have the SQL twice?

When viewing a Template Designer-created macro, you need to be aware that it places several comment blocks at the top of the macro that look like they contain SQL. It's real SQL, but it won't be executed because it's in a comment block. Other information is also stored in the comment blocks, such as the merchant reference number of the store that "owns" the macro.

Here's an example of a *very simple* category page macro with a single very simple category table:

```
%{
2067
TEMPLATE V3.01
Title
EditPanel 1280 1024
Grid 40 20
bgColor 255 255 255
textColor 0 0 0
linkColor 0 0 255
vlinkColor 255 0 255
alinkColor 255 0 0
Boxes 1
%}
%{
Box 19 Category table
0 40 1008 20

Description

border
1
Category Name
$(V_CATEGORY.CGNAME)
linkSub
sql_start
    select CATEGORY.CGRFNBR, CATEGORY.CGMENBR,
    CATEGORY.CGNAME, CGRYREL.CRSEQNBR
    from CATEGORY, CGRYREL
    where crpcgnbr=$(cgrfnbr) and crmenbr=$(cgmenbr) and cgpub=1

sql_end
%}

%{**** DB2WWW MACRO starts here ****%}
%function(dtw_sql) SQL0(){
    select CATEGORY.CGRFNBR, CATEGORY.CGMENBR, CATEGORY.CGNAME,
    CGRYREL.CRSEQNBR
    from CATEGORY, CGRYREL
    where crpcgnbr=$(cgrfnbr) and crmenbr=$(cgmenbr) and cgpub=1

    order by crseqnbr
    %REPORT{
<TABLE BORDER=1>
<TR><TH>Category Name</TH></TR>
    %ROW{
<TR><TD><A
        HREF="/cgi-bin/ncommerce3/CategoryDisplay?cgrfnbr=$(V_CGRFNBR)&
        cgmenbr=$(V_CGMENBR)">$(V.CGNAME)</A></TD></TR>
    %}
</TABLE>
%}
```

```

    %MESSAGE{100:{ %} :continue %}
    %}
    %HTML_REPORT{
    <HTML><head>
    </HEAD>
    <BODY BGCOLOR=#ffffff TEXT=#000000 LINK=#0000ff VLINK=#ff00ff
    ALINK=#ff0000>
    <CENTER>
    <TABLE CELLSPACING=0 CELLPADDING=0 WIDTH=40 BORDER=0>
    <TR> <TD WIDTH=1 HEIGHT=1><TD WIDTH=40></TR>
    <TR> <TD HEIGHT=20><TD> </TR>
    <TR> <TD HEIGHT=20><TD> </TR>
    <TR> <TD HEIGHT=20></TR>
    </TABLE>

    @SQL0( )
    <TABLE CELLSPACING=0 CELLPADDING=0 WIDTH=40 BORDER=0>
    <TR> <TD WIDTH=1 HEIGHT=1><TD WIDTH=40> </TR>
    </TABLE>
    </CENTER>
    </BODY>
    </HTML>
    %}

```

Notice the **%{\*\*\*\* DB2WWW MACRO starts here \*\*\*\*%}** comment block we have made bold in this example? Everything above that are simply comments. Everything below it is the portion of the macro that's actually executed.

### What is the Store Creator and when should I use it?

Store Creator provides an easy-to-use graphical application that steps a user through the creation of a store in nine easy steps. It is designed to be used to quickly (in 20 minutes!) create your online store. To use Store Creator you do NOT need any knowledge of HTML or programming. Store Creator is a good tool to use and we recommend using it to create any news stores. However, you should be aware that the store will have a standardized "look and feel" (though you have a few different "looks and feels" to choose from) and no "backend integration" will exist.

You access the Store Creator from the Net.Commerce administrative pages (NCADMIN), found at

```
http://<your_host_name>/ncadmin
```

Store Creator is one of the main buttons in the navigation frame on the left side of the page.

Store Creator is a Java applet that gets downloaded to your browser. It works well with Netscape Communicator 4.04 or above.

**Note:** Store Creator presently does not work with Internet Explorer 5.0. For more information on browser support, see "What browsers work with Net.Commerce?" on page 18.

Store Creator supports three store models:

- One Stop Shop
 

A simple business-to-consumer store model that does not require shoppers to register.
- Personal Delivery

A business-to-consumer store model that offers more shopper functions including registration and address book.

- **Business to Business**

A business-to-business store model that includes an approvals process for purchases.

Store Creator is only accessible by an administrator in the Site Administrators or Store Creators access group.

Once you have used the Store Creator to create your new store, you will see new files and directories in the HFS at location:

```
/usr/lpp/NetCommerce/html/en_US/<instance name>/<store name>  
/usr/lpp/NetCommerce/macro/en_US/product/<store name>  
/usr/lpp/NetCommerce/macro/en_US/<store_name>
```

By default the new store can be accessed from the URL

```
http://<your_host_name>/store_name
```

Where `store_name` is what you called your new store, but without any blank characters between the words. You now use the Net.Commerce administration dialog to add products to the store, and to delete the sample products if necessary.

### **What changes to the Net.Commerce environment require a restart of the Webserver or Net.Commerce?**

Here's a table that summarizes when a "restart" is necessary. "Restart" means stopping and starting again the process. A *modify restart* is applicable in some cases, and is noted where applicable.

<b>Activity</b>	<b>Restart Websrv?</b>	<b>Restart NC?</b>
Changes to Net.Commerce configuration files or envvars file.	No <sup>1</sup>	Yes <sup>2</sup>
Changes to Webserver configuration file or envvars file	Yes <sup>3</sup>	No
Add new macro directory	No	Yes
Add new HTML directory	Yes <sup>4</sup>	No
Change existing macro	No	No <sup>4</sup>
Change existing HTML	No	No
Add new macro to existing directory	No	No
Add new HTML to existing directory	No	No
Add new OF DLL directory	No	Yes
Add new OF to existing DLL directory	No	Yes
Modify existing DLL in existing DLL directory	No	Yes
Update information in database through NCADMIN utility	No	No <sup>5</sup>

Activity	Restart Websrv?	Restart NC?
<p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>The one exception we have found is the caching facility controlled by <code>NC_DMN_CACHE</code> and <code>NC_CACHE_LIST</code>. If you modify those parameters, it appears you need to stop and start again both the Webserver and Net.Commerce.</li> <li>To pick up these changes you must stop and start again Net.Commerce. The only aspect of Net.Commerce that is affected by an MVS "modify" command is the starting and stopping of traces.</li> <li>Changes to the Webserver's <code>httpd.conf</code> file can be picked up by doing a modify restart on the started task (<code>F &lt;task name&gt;,appl=-restart ...</code> note the dash between the "appl" and "restart"). Changes to the Webserver's <code>envvars</code> file requires a stop and restart of the task.</li> <li>If you have enabled the Net.Commerce dynamic HTML page caching facility, then changes to an existing macro might not appear on the browser screen if that page has already been cached. You would need to clear that page from cache to pick up the new changes. See "How do I clear pages from cache?" on page 68 for information on clearing the cache.</li> <li>Most updates of information in the database do not require a restart of Net.Commerce. An exception is any change to the tables that affect commands (CMDs) or the tables affecting commands (<code>ACC_MODE</code>, <code>POOL_CMD</code>, <code>ACC_CMDGRP</code>), or the use of the NCADMIN "Task Management" function to change API tasks, the use of the "Access Groups" function, or the use of the "Command Security" function.</li> </ol>		

### How do I tell Net.Commerce to look for my custom HTML files in a separate, custom directory?

HTML files are referenced via the `Pass` directives in the Webserver's `httpd.conf` file. For example, to access the Metropolitan Demomall's front page, you put the following in the browser's location window:

```
http://<your_host>/demomall/basemall.htm
```

By default, there is no directory called `/demomall` off the root directory. So how does the Webserver resolve that correctly? With a `Pass` directive that's added by the CMNCONF utility that looks like this:

```
Pass /demomall/* /usr/lpp/NetCommerce/html/en_US/demomall/*
```

What this says is this: whenever the Webserver receives a URL request with `/demomall` immediately following the host name, then resolve the location to the full path as specified on the `Pass` directive.

So, if you want to create your own directory for custom HTML files, you would do the following:

- Create the directory
- Add a `Pass` directive in the Webserver's `httpd.conf` file

**Note:** do not put this `Pass` directive in the directives added by CMNCONF that are located at the top of the `httpd.conf` file. CMNCONF will overwrite that.

- Restart the Webserver

You may, if you desire, put custom HTML files in a directory that's created by CMNCONF. See "May I use the directories under `/usr/lpp/NetCommerce/instance` for my custom HTML or Net.Data macros?" on page 60 for more information on this.

## If I want to create a separate directory for my custom Net.Data macros, how do I do that?

The file you want to modify to include the new directory is `db2www.ini` in the instance's "html root" and add the directory to the `MACRO_PATH` directive. The "html root" is located at:

```
/usr/lpp/NetCommerce/html/en_US/<instance_name>
```

Where `<instance_name>` is the name of the instance you wish to modify. The `db2www.ini` file's layout looks like this:

```
MACRO_PATH /usr/lpp/NetCommerce/instance/<inst_name>/teditor;  
           /usr/lpp/NetCommerce/macro/en_US/ncsample;  
           /usr/lpp/NetCommerce/macro/en_US/demomall;  
           /usr/lpp/NetCommerce/macro/en_US  
INCLUDE_PATH /usr/lpp/NetCommerce/macro/en_US/ncsample;  
            /usr/lpp/NetCommerce/macro/en_US;  
            /usr/lpp/NetCommerce/instance/<inst_name>/teditor;  
            /usr/lpp/NetCommerce/html/en_US;  
ENVIRONMENT (DTW_SQL) dtwsql ()  
ENVIRONMENT (DTW_SYS) sysdll ()  
ENVIRONMENT (DTW_SYSTEM) sysdll ()  
ENVIRONMENT (DTW_PERL) perl.dll ()  
ENVIRONMENT (DTW_REXX) rexx.dll ()  
ENVIRONMENT (DTW_FILE) filedll ()  
DB2MSGS NONE  
DTW_REMOVE_WS NO
```

**Note:** the `MACRO_PATH` and `INCLUDE_PATH` lines actually run on one line, but we broke them across multiple lines here to illustrate it better.

The `MACRO_PATH` is what Net.Commerce uses (or, more precisely, what Net.Data uses) to search through for macros called. It operates just like any other "path" like directive: the directories are searched in the order of their appearance on the `MACRO_PATH` directive. So if you want to add a custom directory, you simply add it to the search order.

```
MACRO_PATH /my_custom_directory;  
           /usr/lpp/NetCommerce/instance/<inst_name>/teditor;  
           /usr/lpp/NetCommerce/macro/en_US/ncsample;  
           /usr/lpp/NetCommerce/macro/en_US/demomall;  
           /usr/lpp/NetCommerce/macro/en_US
```

Once added, you need to *restart Net.Commerce* for the change to take effect.

**Note:** There's a twist to this for category and product macros you wish to invoke using the `CategoryDisplay` and `ProductDisplay` commands. *For those two command only*, Net.Commerce assumes the macro will be found in a sub-directory offset of `/category` and `/product` off the directories listed on the `MACRO_PATH` directive in `db2www.ini`.

Therefore, your custom directory for category and product pages (that will be invoked with `CategoryDisplay` and `ProductDisplay`) should be *sub-directories* of `/category` and `/product` off of the custom directory named on `MACRO_PATH`.

For example, the directory shown on `MACRO_PATH` above is `/my_custom_directory`. Create two sub-directories off that directory, one

named `/product` and one named `/category`. Leave the `MACRO_PATH` designation `/my_custom_directory`. Place your category and product macros in their respective sub-directories. Now when `CategoryDisplay` or `ProductDisplay` is run, it'll assume the `/category` and `/product` offsets and will find the macros in your custom directory.

You may also place custom `Net.Data` macros in the directories created by `CMNCONF`. See "May I use the directories under `/usr/lpp/NetCommerce/instance` for my custom HTML or `Net.Data` macros?" on page 60 for more information on this.

**Warning!** Information you add to `db2www.ini` will be overwritten whenever you run the "Access Control" function of `CMNCONF` (Option 2). You should keep a backup of the file, and re-apply any changes to `db2www.ini` any time you run `CMNCONF` to modify a configuration.

### **Can I reference macros located in sub-directories of the directories shown on `MACRO_PATH`?**

Yes. You simply provide the subdirectory reference. For example, assume you have a macro called `xyz.d2w`, and it's located in the subdirectory `/subdir` off of the directory `/my_custom_directory` that's referenced in your `MACRO_PATH`. On the `ExecMacro` command (which may be used to directly invoke a macro, and is used here only as an obvious example of what we're trying to illustrate) would look like this:

```
ExecMacro/subdir/xyz.d2w/report
```

The inclusion of the `/subdir` reference tells `Net.Data` to look for a directory called `/subdir` off each directory named in `MACRO_PATH`, and if it finds it, then look for the macro called `xyz.d2w`.

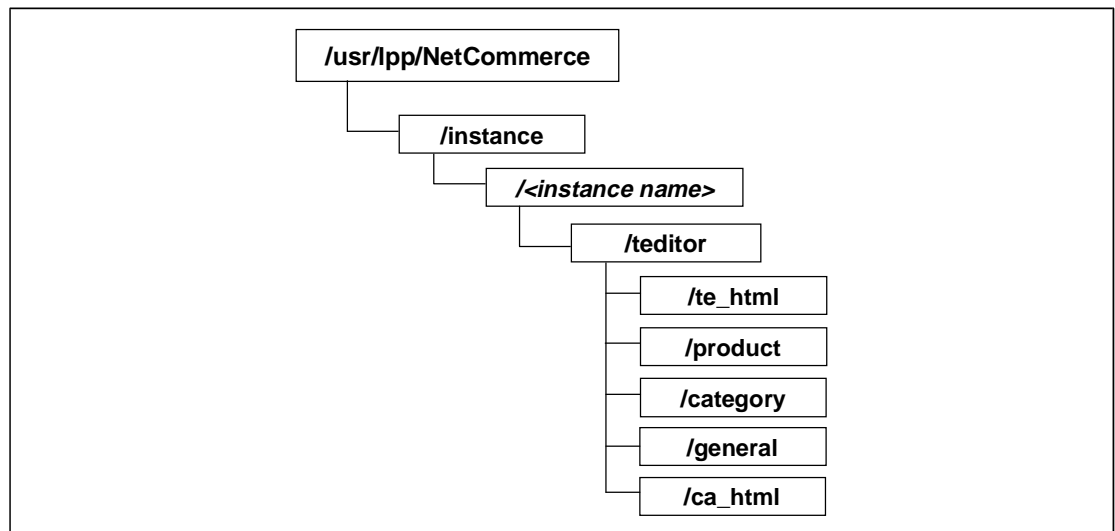
### **If I have a macro from `NC V3.1.1`, may I use it with `Net.Commerce V3.1.2`?**

A qualified "yes". The most obvious difference between a `V3.1.1` macro and a `V3.1.2` macro is the command structure (`V3.1.1` used the format, `;shopcart/add`, while `V3.1.2` is `InterestItemAdd`). Thankfully, `Net.Commerce V3.1.2` will automatically recognize the format of the `V3.1.1` commands and execute the equivalent `V3.1.2` command. So you do not need to crawl through each macro and change the commands.

The other potential issue is the SQL employed in the macro. The table structure of `V3.1.2` is very similar to that of `V3.1.1`. But you should test the macro to make sure the SQL still works against the `V3.1.2` database structure.

### **May I use the directories under `/usr/lpp/NetCommerce/instance` for my custom HTML or `Net.Data` macros?**

Yes. The directory structure under `/usr/lpp/NetCommerce/instance` looks like this:



You can place your HTML files in `/te_html` or your Net.Data macros in `/product`, `/category`, or `/general`. There is a couple of things to know:

- Any references to HTML files in `/te_html` should use the directory name of `/te_html` as part of the request. For example, if you tell Net.Commerce that the home page for a store is `mystore.html`, you should register the home page as `/te_html/mystore.html`.
- Any references to Net.Data macros in `/product` or `/category` or `/general` should use those directories as prefixes to the Net.Data name. For example, if you want to use the `ExecMacro` command to run a macro, you'd point to the macro `mymacro.d2w` with `ExecMacro/product/mymacro.d2w/report`.

**Note:** there is one exception to this requirement to prefix the macro name with the directory. When associating a "template" with a category or product in the Store Administrator function, you do not need to prefix the directory name. Why not? Because Net.Commerce, when executing the `DisplayItem` or `DisplayCategory` commands to display a product or category page, it's smart enough to assume the `/product` or `/category` prefix to the macro name.

There are two potential downsides to putting your customer HTML or Net.Data macros in this directory structure:

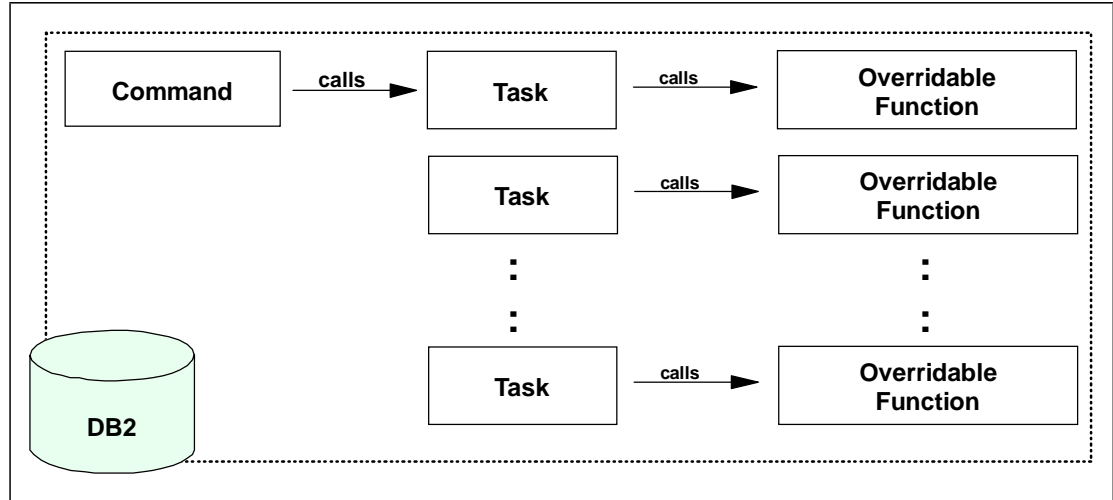
1. If you delete an instance and forget that these files are in the directory structure, you might delete the `<instance_name>` directory and all its subdirectories as part of cleaning up the instance.
2. If you use the Template Designer and name a macro or HTML the same name as one you've manually added to these directories, you might accidentally click "Okay" to the "Replace?" question, thus overlaying a file you really didn't wish to overlay.

But there's a significant up-side to this, and it has to do with keeping all of your customized files under a single directory structure (including configuration files). This permits you to make that single directory a mount point and mount a separate HFS at that location. See "Is there any way to get all the configuration files into a single HFS directory?" on page 30 for more information on how to do that.

## What is an "Overridable Function"?

When Net.Commerce receives a command, it carries out the function of that command by executing some number of "tasks", each representing a piece of the overall function of the command. Each task has, in turn, what's called an "overridable function" (OF) associated with it. The OF is the program that does the actual work of the task, such as calculating the shipping charges, or updating some information in the database.

So the picture looks something like this:



The reason why DB2 is referenced here is because information about the commands, tasks, and OFs are all kept in the Net.Commerce DB2 database.

The reason why it's called an Overridable Function is because you may code your own C++ program to replace the default function. You may wish to do this to provide custom processing of, for example, the checking for inventory, or the calculation of taxes.

## How do I get Net.Commerce to recognize and use my Overridable Function?

Assuming your OF is properly coded according to the the specifications for Net.Commerce OFs, and it compiled clean, here's what you do:

- Register the OF into the database using SPUFI or some other DB2 utility
- Assign the DLL to the appropriate task using NCADMIN

The registration of the OF information into the database is done with the following SQL. It is a manual process (in other words, there is no NCADMIN process to do this step):

```
insert into ofs (refnum , dll_name, vendor, product,
name, version, description)
values ((select max(refnum) from ofs) + 1, '<DLL_Name>',
'<IBM>' , '<NC>' , '<Overridable_Function_Name>',
<1.0> , <description> )
```

There is a "nested select" statement in that which does not work on OS/390 DB2 V5.1. Essentially, the `(select max(refnum) from ofs)+1` portion of this is looking for the maximum value of `refnum` from the OFS table, and then setting the `refnum` value for the insert to that value + 1.



Unfortunately, this can't be done automatically like this on DB2 V5.1. So simply do a SPUFI query of the refnum column of OFS table, find the maximum value, and then hardcode that number plus one in place of `(select max(refnum) from ofs)+1` in the provided SQL.

Once that's done, then use the NCADMIN function "Task Management" (found under "Site Manager") to assign the OF DLL to the task.

There is a very good write-up on this in the online help under "overridable functions (concepts)" in the index.

### May I keep my custom OFs in a separate directory?

Yes. In fact, you should. The directory in which Net.Commerce will look for OF DLLs is specified on the LIBPATH directive of this file:

```
/usr/lpp/NetCommerce/html/en_US/<inst_name>/ncommerce.envvars
```

By default, the LIBPATH directive points to:

```
LIBPATH=/usr/lpp/internet/bin:/usr/lpp/NetCommerce/lib
```

Here's what you would do:

- Create the directory in which you will store you OF DLLs. Don't hang it off the `<inst_name>` directory. If you do and you delete the instance, you may forget about the DLLs being out there and delete the `<inst_name>` directory (and all its subdirectories) as part of cleaning up your deleted instance.
- Add your new directory to the LIBPATH directive. Separate the different directories with a colon.

**Note:** You should update `ncommerce.envvars` in several places:

- In the `/usr/lpp/NetCommerce/html/en_US/<inst_name>` directory corresponding to your instance.
  - In the `/usr/lpp/NetCommerce/html/en_US/<inst_name>` directory corresponding to any other instances you are running and for which you wish that instance to get at the DLLs
  - In the `/usr/lpp/NetCommerce/install/ncommerce.envsampp` file so that future CMNCONF configurations pick up the change
  - In the `/etc/NC/<inst_name>.envvars` file so that it's consistent with the server envvars file in the `/usr/lpp/NetCommerce/html/en_US/<inst_name>.envvars` file.
  - In the `/etc/ncommerce.envvars` file so it is consistent as well.
- Stop and start again Net.Commerce to pick up the change.

### If I have API Task DLLs left over from V3.1.1, may I use them with V3.1.2?

Yes. Net.Commerce V3.1.2 has a "proxy API" function which will permit API tasks written for V3.1.1 to be used without modification in V3.1.2.

You need to run an (undocumented) utility program called `migproxy` to get this to work. This updates the `TASK_MER_OF` table so the OF being pointed to for a task is the proxy OF instead of the default OF. It reads the `APIS` table and for each row in it, gets the corresponding row in the proxy table. It then updates the `TASK_MER_OF` table so the `TASK_RN` and `MERCHANT_RN` fields match the info in the `APIS` table and the `OF_RN` matches the `PROXYOFS` in the `PROXY` table.

You run the utility from OE. The syntax is:

```
migproxy [-verbose] dbowner dbplan dbssid
```

There are two notes to this:

- You need to move the information contained in your V3.1.1 APIS table over to the V3.1.2 APIS table. Use whatever DB2 utility you wish to do this export/import.
- This utility, like "Mass Import" and "NCCLEAN", has the requirement that the NLSPATH and LIBPATH variables have references to the appropriate Net.Commerce information:

```
NLSPATH      /usr/lpp/NetCommerce/msg/en_US/%N
LIBPATH      /usr/lpp/NetCommerce/lib
```

### How does the Net.Commerce caching function work?

The dynamic HTML generation engine of Net.Commerce is Net.Data. As a performance enhancement to Net.Commerce, a function was added to the product that will cache in DB2 the resulting HTML generated by Net.Data. The next time the page is called, the HTML is pulled from DB2, which is much faster and involves less overhead than invoking Net.Data.

This feature is enabled two different ways:

1. With the "Cache Control" setting found on Option 1 of CMNCONF (System Configuration), panel 2 of 2. A restart of the Webserver and Net.Commerce is then necessary. This sets the value for NC\_DMN\_CACHE in the `ncommerce.conf` file.
2. With the "System Monitoring" function of the NCADMIN and enabling the "Cache" function. This will dynamically start the caching facility.

**Note:** This "Cache Control" feature of CMNCONF, by itself, only involves the caching of Category pages and Product pages. To cache other pages, please see "Is there a way to cache something other than Product or Category pages?" on page 66.

There are three levels of caching for this feature:

"Cache Control" Setting	"System Monitoring" Setting	Function
<b>0</b>	<b>Disable</b>	Disables the function. No caching is performed. No cached pages are pulled from DB2. All requests for dynamic pages result in the calling of Net.Data and the dynamic generation of the page.
<b>1</b>	<b>Enable R/W</b>	Enables the "Read/Write" caching. What that means is this: if the page requested has already been cached in DB2, then pull the HTML for that page from DB2. If the page has <i>not</i> been cached, then generate the page with Net.Data, serve the page to the browser, <i>and</i> cache the page in DB2.

"Cache Control" Setting	"System Monitoring" Setting	Function
2	Enable R/O	<p>Enables the "Read Only" caching. What that means is this: if the page requested has already been cached in DB2, then pull the HTML for that page from DB2. If the page has not been cached, then generate the page with Net.Data, serve the page to the browser, and <b>do not</b> cache the page in DB2.</p> <p>This is different from "Read/Write" in that new pages will not be added to cache. Only those already found in DB2 will be served from DB2. Why this feature? Because it allows you to "seed" the DB2 cache table (NC_CACHE) with only those pages you wish to take up space in your DB2 table space. Pages infrequently touched by shoppers serve little purpose in being cached. Only those pages frequently requested should be cached.</p>
<p><b>Note:</b> the parameter in the <code>nccommerce.conf</code> configuration file related to this setting is <code>NC_DMN_CACHE</code>. You may set this value manually, even though the <i>Configuring and Getting Started</i> book says it much be done via the CMNCONF program.</p>		

This function raises many questions:

- What happens if I change the Net.Data macro so it has a different HTML look and feel? Answer: what's in cache will be served out until that record is purged from cache.
- What happens if I change the information in the database that will display on a Category or Product page? Answer: what's in cache will be served out until that record is purged from cache, or the setting for "Cache Control" is set to 0.
- If I change the information in the database, will the cache record be automatically cleared? Answer: nice idea, but no. What's needed is a function of DB2 called "triggers", which is available in V6. What's also needed is function added to Net.Commerce to take advantage of the trigger support to provide automatic update of cached records upon the change of another record.
- How do I purge the cache? Answer: refer to "How do I clear pages from cache?" on page 68 for information on purging cache records.
- Let's say I wish to use the "Read Only" feature. How do I populate the cache page with only those records I wish it to hold? Answer: there are several ways you can achieve this:
  - Turn on caching to "Read/Write" and then manually hit the pages you wished cached. Then change the caching to "Read Only" using the "System Monitoring" and dynamically reset the caching of the server.
  - Create two instances of Net.Commerce, both sharing the same database. One is your production instance, and for it you have caching set at "Read Only". The other instance is a test instance and has caching set at "Read/Write". It is only accessible within your company and, preferably, only by you. With a browser hit those pages you wish cached. The pages will be cached in the database, and because the database is being shared, the

other instance (the production one with "Read Only") will start seeing those records in cache and will start feeding them from DB2.

- Set caching at "Read/Write". Every morning (or however often you choose) purge all the records in cache. Allow your customers to "vote" on which pages are the most actively hit pages by allowing all pages to be cached. Sure, some infrequently touched pages will be cached, but on the whole the pages that will be cached will be what your customers are most frequently touching.

### Is there a way to cache something other than Product or Category pages?

Yes. This caching function picks up where the previous one left off (see "How does the Net.Commerce caching function work?" on page 64); that is, it'll allow you to specify pretty much anything to be cached.

The starting point for this caching function is a new parameter that resides in the `ncommerce.conf` configuration file. This parameter specifies what string will be used to match against the incoming URLs to see if the page should be cached.

**Note:** This is a key point: in the database table that holds the cached HTML pages the *entire URL* is kept. *The entire URL found in the database table must match the incoming URL for the cached page to be drawn from cache.*

The specification on this parameter is used to determine if something should be put *into* the cache table; the entire URL is used to determine if something should be pulled out of the table.<sup>1</sup>

<sup>1</sup>Truth is, it's a bit more complicated than this, but what we just said is essentially correct. The page will be pulled from cache if the URL matches *and* the `NC_CACHE_LIST` parameter is active.

#### Format of parameter

Here's the format of the new parameter:

```
NC_CACHE_LIST <string>
```

Where `<string>` is the character string of the URL you wish to cache. It must be a contiguous string (no spaces).

You may have more than one `NC_CACHE_LIST` parameter in the `ncommerce.conf` file. This is permitted because you may wish to cache different pages based on multiple strings.

#### How it works

Imagine you have a parameter of:

```
NC_CACHE_LIST mall
```

With that enabled, any URL that comes in with the string "mall" will be checked against what's in the cache table. That would mean all of the following URLs would be checked:

```
.../cgi-bin/ncommerce/ExecMacro/mall_dir.d2w/report  
.../cgi-bin/ncommerce/ExecMacro/search_mall.d2w/report  
.../cgi-bin/ncommerce/ExecMacro/macro_name.d2w/mall
```

You get the point. If the page is already cached (*based on the entire URL matching, not just "mall" matching!*), Net.Commerce will pull the HTML from cache and send it to the browser. If it's not in cache, then it runs the command, stuffs the HTML into the cache, and then sends the HTML to the browser.

### **Behaviors of NC\_CACHE\_LIST facility**

- The string set on the NC\_CACHE\_LIST parameter determines what will be placed into the cache table. The entire URL must match to have the cached HTML pulled out of the table and sent to the browser.
- The "cache control" feature of Net.Commerce (as specified by the NC\_DMN\_CACHE parameter in `ncommerce.conf` -- which is set by the "cache control" fields of the CMNCONF configuration utility) controls the behavior of this new caching facility:
  - If NC\_DMN\_CACHE=0 ("None"), then no caching into or out of the cache table occurs.
  - If NC\_DMN\_CACHE=1 ("Read/Write"), then HTML pages are written into the cache table if the string on NC\_CACHE\_LIST matches anywhere within the received URL. If the inbound URL matches what's already in the cache table, the record is pulled from cache and issued to the browser.
  - If NC\_DMN\_CACHE=2 ("Read Only"), then only if the entire URL is found in the cache table will the cached HTML be issued to the browser.
- Static HTML pages can't be cached with this facility because requests for those pages never get passed up to Net.Commerce -- they are handled by the Web Server.
- In theory, all that you should need to do is stop and start again the Webserver to have the NC\_CACHE\_LIST setting take effect. When this function was added to V3.1.1, we found that both the Webserver and Net.Commerce had to be "bounced". We have not yet tested this on V3.1.2 to see if both still need to be "bounced."

### **Be very careful with this function!**

Unlike the "cache control" mechanism that only caches category or product pages, this facility can result in customer-specific information being cached and shown to other shoppers.

Many of the Net.Commerce commands utilize the shopper's ID when invoking their function, though the shopper's ID doesn't appear anywhere in the URL itself. For example, the command to display the shopping cart is:

```
InterestItemDisplay
```

Under the covers of Net.Commerce the shopper's ID (`$SESSION_ID`) is used to query the database to show the items in the shopping cart for that particular shopper. But that variable is not on the URL.

So, imagine you have `NC_CACHE_LIST InterestItemDisplay` specified in your `ncommerce.conf` file. The first shopper to come along and display their shopping cart will have the HTML for their shopping cart cached into the table. All subsequent shoppers will see that first shopper's cart because all shoppers will use the simple `InterestItemDisplay` command to display their shopping cart!

You should avoid caching any pages related to the following functions:

- Shopping Cart
- Shipto information
- Order information
- Registration forms and Address Forms

You might wonder what you may use as a string for this function. The original demand for this function came from a customer with a great many "execmacro" functions developed. That's really what this function is good for. However, you need to be careful to cache based on a string that uniquely defines the HTML page that will result. As mentioned earlier, specifying simply:

```
NC_CACHE_LIST ExecMacro
```

is probably a bad idea. However, specifying a more complete string will work just fine:

```
NC_CACHE_LIST ExecMacro/search_front_page.d2w/input
NC_CACHE_LIST ExecMacro/todays_specials.d2w/report
```

Recall that you may specify multiple lines in the `ncommerce.conf` file, so don't try to be so generic with your specified string that you end up caching everything.

### How do I clear pages from cache?

There are three ways you can clear the cached pages from the table `NC_CACHE`:

- Use SPUFI (or other DB2 utility) to clear the records from the table, for example:

```
delete from NC_CACHE;
```

The SQL shown here clears the whole table. If you're looking to clear just a certain record from cache, you may do that as well, but you need to be careful. The HTML page might be split across multiple records in the database.

Here's an example of what the records in `NC_CACHE` look like. This is just showing the `HTMLNAME` field of the record. You'll see that for the first category page, *two* records in the database exist:

```
-----+-----+-----+-----+
HTMLNAME
-----+-----+-----+-----+
CategoryDisplay?cgmenbr=2067&cgrfnbr=24
CategoryDisplay?cgmenbr=2067&cgrfnbr=24
CategoryDisplay?cgmenbr=2067&cgrfnbr=27
CategoryDisplay?cgmenbr=2067&cgrfnbr=27
ProductDisplay?prmenbr=2067&prrfnbr=46
ProductDisplay?prmenbr=2067&prrfnbr=46
```

What this means is you'd need to make sure you deleted both of those records are deleted. You could use this SQL:

```
delete from cmnsrv.nc_cache
where htmlname='CategoryDisplay?cgmenbr=2067&cgrfnbr=24' ;
```

- Use the `NCCLEAN` utility (see "How does the `NCCLEAN` utility work?" on page 73 for more on this). This method has some granularity to it: you may purge those records for a specified merchant; purge just category or product pages (also by merchant if desired); and you can specify that records older than a certain period of days be purged.

- Use the "Purge Cache Data" option of "System Monitoring" function under the NCADMIN "Site Manager" button:

**System Monitoring**

Performance Monitor Output

Net.Commerce Server : <your\_host\_name>  
Time: (date and time)

No Net.Commerce Command Performance Data.

---

No Net.Commerce Error Data.

---

Trace, Cache and Performance Monitor Settings

Director Trace:  Enable  Disable

Director Debug:  Enable  Disable

Cache:  Enable R/W  Enable R/O  Disable

**Purge Cache Data:**  Yes  No

Performance Monitor  Enable  Disable

Purge Performance Data  Yes  No

Frequency of Saving Performance Data

Click on the "Yes" radio button, then click on the "Submit" button. This wipes out the entire table.

If you're curious about what this "Performance Monitor" thing is, please see "How does the Net.Commerce performance monitor work?" on page 75 for more information.

### How does the Mass Import utility work?

At the heart of Net.Commerce is a DB2 database. Typically information about products is placed into the database with the NCADMIN function, but that function can prove to be cumbersome when you have hundreds or thousands of products to load into the database.

The Mass Import Utility is a UNIX Systems Services program that takes records from a delimited HFS file and loads them directly into the database. It is an ideal way of bulk loading information into the database.

**Note:** Using Mass Import requires a fair bit of knowledge about the layout of the Net.Commerce database tables and what each is used for.

Here's an example of an input file for Mass Import that will create another category of "Tools" in the sample store "Basics" of the Demomall. The new "Tools" category is going to be a sub-category of the "Tops" category (which from a marketing point of view is a silly thing to do, but bear with us).

```
#STORE;Basics
#CATEGORY;Tools;;;/demomall/cat_hat2.gif;/demomall/cat_hat2.gif;1;;Tops
#CATESGP;Tools;cat_bas2.d2w
#PRODUCT;SKU-001;;Socket Set;;;/demomall/hardth1.gif;/demomall/hardth1.gif;1
#CGPRREL;Tools;SKU-001;1
#PRODPRCS;SKU-001;;29.95;CAD
#PRODSGP;SKU-001;;tempbas1.d2w
```

For complete information on what each "#" record command is for, and what each field of an input records does, refer to the online help. Each Mass Import command will have its own listing in the index.

Briefly, here's what each record in this input file will do:

- #STORE; identifies the store against which all subsequent records in the input file will be processed. In this case it's "Basics".
- #CATEGORY; creates a new category, provides the thumbnail and fullsize image files for the category, and then identifies the parent category to which it will belong
- #CATESGP; associates a category macro with the new category
- #PRODUCT; creates a new product, and provides information about the product
- #CGPRREL; associates the product with a category, in this case the one we just created with the #CATEGORY record
- #PRODPRCS; provides a price for the product
- #PRODSGP; associates a product macro with the new category

Now look back at the example input file and note a few things:

- There is an order to the records found in the file. The #STORE; record must come first. After that, records that make reference to a category or product being created by mass import must come after the record that creates the category or product
- The order of the parameters on the record is very strictly controlled by the syntax of the record. Many parameters are not required, and if you wish to omit them you may, but you must "hold the place" of the parameter with the semi-colons. Parameters at the end of the record that will be null do not need to be "place held."

If you're looking at this and thinking that the commands for Mass Import look very similar to the names of the database tables for Net.Commerce, you'd be correct in your observation. What Mass Import is doing is loading records into the tables, and the Mass Import command relates directly to the table into which the information will be store.

To invoke Mass Import, here's what you need to do:



**Note:** The environment of the OMVS session from which you invoke this utility must have references to Net.Commerce on the LIBPATH and NLSPATH variables:

```
LIBPATH must reference /usr/lpp/NetCommerce/lib
NLSPATH must reference /usr/lpp/NetCommerce/msg/en_US/%N
```

Failure to provide this will result in errors, as detailed in ""CEE3501S The module nc3\_dbc\_db2.so was not found"" on page 85 and ""ERROR : Message file 'nc3\_util.cat' could not be opened!"" on page 86.

- Make sure DB2 is up and running
- Go to the OMVS shell
- Change into the /usr/lpp/NetCommerce/bin directory
- Issue the following command:

```
massimpt -infile <note 1> -f <note2> -log <note 3>
```

**Note 1** the fully qualified directory and name of the input file. For example, /u/homedir/input.file

**Note 2** the fully qualified path and name of the NC configuration file. This is the ncommerce.conf file, and it is located in this directory:

```
/usr/lpp/NetCommerce/html/en_US/<instance_name>
```

(What this "-f" pointer does is provides access to the configuration file, which has all the information about the DB2 subsystem name, the database name, the plan name, etc. You can provide all that info with separate parameters on mass import, but why bother when you can simply point to the conf file?)

**Note 3** the fully qualified path and name of the log file you wish to create. For example, /u/homedir/log.file

There are other parameters for the massimpt command, so you are strongly encouraged to consult the online help for the syntax and usage of the other commands. This is particularly true if you read the question "How can I improve the performance of Mass Import?" on page 72. The write-up there refers to many of the parameters this question has not made mention of.

In summary, the Mass Import utility is a bulk data loading facility that takes as its input a flat delimited file. The records in that flat delimited file must abide by the syntax rules for the record, and the best place to find the syntax rules is in the online help.

### How can I split my command lines across multiple lines of the input file?

The command lines for mass import can grow quite long, particularly if you're supplying lengthy description fields. This can get cumbersome with editors because the lines will stretch well beyond the right side of the screen. There is a way to get around this. It involves using the #ROWDELIMITER command.

The mass import utility "sees" the end of a command record by the presence of a "newline" character (hexadecimal x'15' in EBCDIC). If you break the lines without

providing a #ROWDELIMITER command, mass import will see the x'15' at the end of each line as meaning a new command is coming next. When it doesn't see a new command, it'll flag an error.

The #ROWDELIMITER command allows you to specify a new value that mass import will use to detect the end of a command record. The format of the command is this:

```
#ROWDELIMITER;<new value>
```

You place this command at that point in the input file where you wish the new value to take effect. It'll stay in effect until another #ROWDELIMITER command is seen, or the end of the input file is reached.

Here's an example of what this would look like. Imagine adding a new category to the "Basics" store of the Demomall. This is what the input file would look like without being able to split input records across lines:

```
#STORE;Basics
#CATEGORY;New Category;Short Description;Long Description
```

Using the #ROWDELIMITER command and changing the value to an ampersand (&), the input file may now look like this:

```
#STORE;Basics
#ROWDELIMITER;&
#CATEGORY;New Category
;Short Description
;Long Description&
```

### May I insert comments into the input file of Mass Import?

Sadly, no.

### How can I improve the performance of Mass Import?

We've identified several things you can do:

- Use higher -commitcount parameter, for example -commitcount 300.

**Note:** the marginal benefits derived from commitcount tend to diminish with higher and higher values. The default value is 1. Increasing that value brings immediate results. Beyond around 500 or so the incremental benefits get small.

- Use -loadnew product|category parameter for Mass Import.
- Make sure every table has proper indexes defined such as:
  - **CATEGORY** table  
create index i\_category on category (cgmenbr);
  - **CGRYREL** table  
create unique index ui\_cgryrel on cgryrel (crmenbr, crpcgnbr, crccgnbr);  
create index i\_cgryrel1 on cgryrel (crpcgnbr);  
create index i\_cgryrel2 on cgryrel (crccgnbr);
  - **PRODUCT** table  
create unique index ui\_product on product (prmenbr, prnbr);  
create index i\_product1 on product (prmenbr);  
create index i\_product2 on product (prprfnbr);
  - **CGPRREL** table  
create unique index ui\_cgprrel on cgprrel (cpmenbr, cpcgnbr, cprnbr);  
create index i\_cgprrel1 on cgprrel (cpmenbr);

- ```

create index i_cgprrel2 on cgprrel (cpcgnbr);
create index i_cgprrel3 on cgprrel (cpprnbr);

```
- **PRODPRCS table**

```

create unique index ui_prodprcs on prodprcs (ppmenbr,
      ppprnbr, ppsgnbr, pppre, ppcur);
create index i_prodprcs1 on prodprcs (ppmenbr);
create index i_prodprcs2 on prodprcs (ppprnbr);
create index i_prodprcs3 on prodprcs (ppsgnbr);

```
  - **PRODATR table**

```

create unique index ui_prodatr on prodatr (paprnbr, paname);
create index i_prodatr1 on prodatr (pamenbr);
create index i_prodatr2 on prodatr (paprnbr);

```
  - **PRODDSTATR table**

```

create unique index ui_proddstatr on proddstatr (pdprnbr,
      pdname);
create index i_proddstatr1 on proddstatr (pdmenbr);
create index i_proddstatr2 on proddstatr (pdprnbr);

```
  - **PRODSGP table**

```

create unique index ui_prodsgrp on prodsgrp (psmenbr, psprnbr,
      pssgnbr);

```
  - **CATESGP table**

```

create unique index ui_catesgp on catesgp (csmenbr, cscgnbr,
      cssgnbr);

```
- To benefit from the performance advantages that caching provides, you need to order your records as follows:
    - Enter the records pertaining to categories in the order given in Create an Import File. The first record in the category group must be #CATEGORY. The category group consists of the tables CATEGORY and CATESGP.
    - Enter the records pertaining to products as a group in the order described in Create an Import File. The first record in the product group must be #PRODUCT. The product group consists of the tables PRODUCT, PRODDSTATR, PRODATR, PRODPRCS and PRODSGP.
    - Enter child products immediately after the parent product.
    - Enter child categories immediately after the parent category.
    - Enter category product relationship records (using the #CGPRREL command) after the corresponding product and category groups.

In addition to the specific Mass Import suggestions provided above, there are some general DB2 Database tuning things that can achieve results:

- Large bufferpool size.
- Use the Prepared Statement Cache.
- Perform a DB2 reorganize table regularly on tables that are updated frequently.
- Perform a DB2 runstats on table regularly on tables that are updated frequently, and after you reorganize any table.

### **How does the NCCLEAN utility work?**

The NCCLEAN utility is intended to assist in the cleanup of some (not all) of the database tables. The information that can be removed by NCCLEAN is:

- Records in the PRODUCT table with PRPUB field set to "2" (marked for deletion).
- Records in the SHOPPER table with SHSHTYP field set to "G" (guest shopper)

- Records in the SHADDR table with SAADRFLG field set to "T" (temporary address, which are created whenever an address is updated -- the old address is copied over to a separate record and flagged with a "T")
- Records in the ORDERS table
- Records in the NC\_CACHE table, which are the result of the caching function being enabled (see "How does the Net.Commerce caching function work?" on page 64 for Product and Category page caching, and "Is there a way to cache something other than Product or Category pages?" on page 66 for other types of page caching).
- Records in the USRTRAFFIC table, which are the result of the "User Traffic Logging" function being enabled (see "What is the USRTRAFFIC table, and for what is it used?" on page 77 for more information on this function).

Many people ask, "Could I clean up that information with other tools?" And that answer to that is yes, you could. NCCLEAN simply provides a tool for use by those who might not be as SQL-savvy.

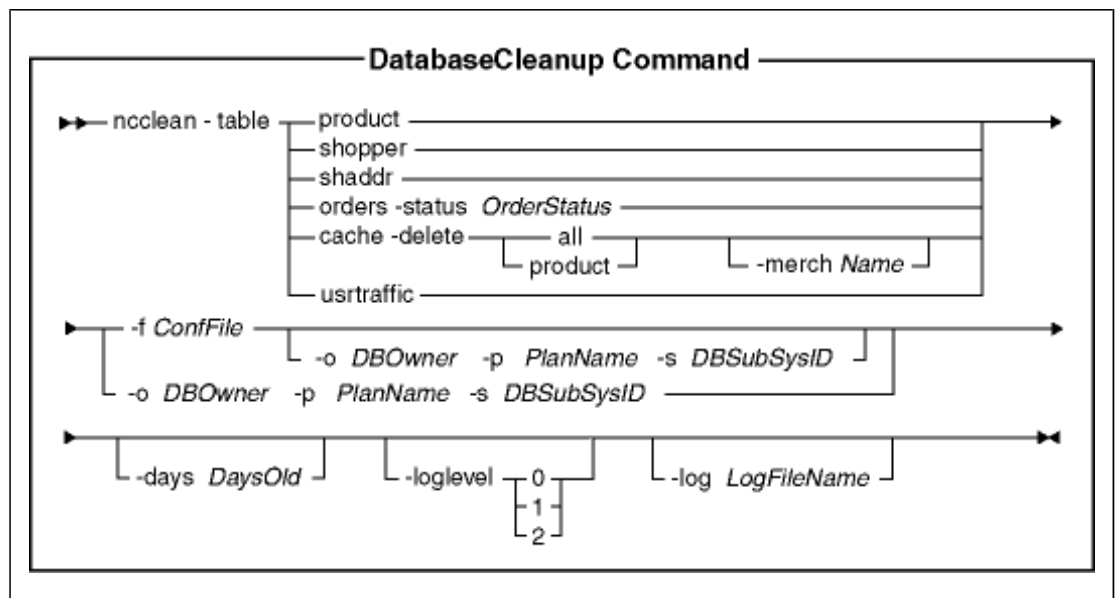
The utility is run by issuing a command from the OMVS command line. The program resides in the `/usr/lpp/NetCommerce/bin` directory, so that's where you want to run the command, unless you have that directory on your PATH environment variable.

**Note:** Just like for the Mass Import utility, the environment of the OMVS session from which you invoke this utility must have references to Net.Commerce on the LIBPATH and NLSPATH variables:

```
LIBPATH must reference /usr/lpp/NetCommerce/lib
NLSPATH must reference /usr/lpp/NetCommerce/msg/en_US/%N
```

Failure to provide this results in errors as described in "'CEE3501S The module nc3\_containers.so was not found.'" on page 85 and "'ERROR : Message file 'nc3\_util.cat' could not be opened!'" on page 86.

The command syntax for the NCCLEAN utility is shown in this diagram:



A couple of notes about this syntax:

- Only one "table" reference can be made per execution of NCCLEAN
- The `-f confile` parameter is used to point to the server configuration file, which is the one that resides at:

```
/usr/lpp/NetCommerce/html/en_US/<inst_name>/ncommerce.conf
```

In pointing to that file you provide NCCLEAN with all the information it needs about the database, such as the name, the owner, and the plan. You could choose to supply that information separately with the other commands shown, but we've found it much easier just to point to the `confile`.

- The online help has greater detail on all of this. It can be accessed at:

```
<your_host>/nchelp/index.htm
```

**Note:** look under "Cleanup Utility" in the index, *not* "ncclean"

### How does the Net.Commerce performance monitor work?

When Net.Commerce is running, and performance monitor is enabled, Net.Commerce continually captures information relating to statistics on the each of the commands (as well as errors) it has seen:

- A timestamp of the first time it saw the command
- A timestamp of the most recent time it saw the command
- A count of the number of times it saw the command
- The shortest time it took to execute the command
- The longest time it took to execute the command
- The average time it has taken to execute the command

These statistics are kept in memory until a specified number of commands are seen, and then the statistics are written out to the PERFLOG table of the database.

To enable performance monitoring, or to view the contents of this table, a function called "System Monitoring" has been added to the navigation bar of SiteManager, under the NCADMIN utility. This function, when accessed, will query the contents of the table and display the results in an HTML page on your browser.

Here's what the "System Monitoring" screen looks like when you first access it:

## System Monitoring

Performance Monitor Output

Net.Commerce Server : <your\_host\_name>  
Time: (date and time)

No Net.Commerce Command Performance Data.

---

No Net.Commerce Error Data.

---

Trace, Cache and Performance Monitor Settings

**Director Trace:**  Enable  Disable  
**Director Debug:**  Enable  Disable

**Cache:**  Enable R/W  Enable R/O  Disable  
**Purge Cache Data:**  Yes  No

**Performance Monitor**  Enable  Disable  
**Purge Performance Data**  Yes  No  
**Frequency of Saving Performance Data**

By default performance monitor will be disabled when you start Net.Commerce

To enable it, click on the "Enable" radio button next to "Performance Monitor" and then click on "Submit." Net.Commerce will start collecting the statistics. If you go back to the System Monitor page, you'll see a page that looks *something* like this:

## System Monitoring

Performance Monitor Output

Net.Commerce Server : <your\_host\_name>  
Time: (date and time)

Net.Command Command Performance Data

| Command             | Last Time | Minimum Time | Maximum Time | Average Time | Number Trans | First Occurrence | Last Occurrence |
|---------------------|-----------|--------------|--------------|--------------|--------------|------------------|-----------------|
| InterestItemDisplay | 0.02      | 0.01         | 0.18         | 0.09         | 79           | timestamp        | timestamp       |
| ExecMacro/Search    | 0.04      | 0.12         | 0.49         | 0.11         | 134          | timestamp        | timestamp       |
| RegisterForm        | 0.01      | 0.01         | 1.44         | 0.06         | 256          | timestamp        | timestamp       |
| OrderItemDisplay    | 0.12      | 0.05         | 0.79         | 0.14         | 23           | timestamp        | timestamp       |

There will be as many commands as seen by Net.Commerce up to that point.

### ***Refreshing data***

Just click on the "Submit" button. The page will be refreshed with the most recent data.

**Note:** Don't hit the "reload" button. NCADMIN will go back to the home page if you do.

### ***Purging data and starting fresh***

Click on the "Yes" radio button for "Purge Performance Data" and then click on "Submit". This will clear out the PERFLOG table.

### ***Changing the frequency of saves to DB2***

The default value is 100, which means that for every 100 commands seen by Net.Commerce, it'll write whatever statistics it has into the PERFLOG table. You may change this value to whatever you'd like, keeping in mind that a too-frequent update schedule will be an extra burden on your system.

**Note:** You should be aware that the contents of the memory cache are written to the database *every time you access this "System Monitoring" page, or every time you hit the "Submit" button*. Even if only 20 commands have been seen since the last save to PERFLOG, accessing this page will re-write the contents of the memory cache to this table.

Therefore, you should think long and hard before you set this frequency value too low.

### ***Keeping a time-sliced history of performance***

A common misunderstanding about this table is that it keeps a historical record of performance by time period. It doesn't. The statistics kept in the memory cache are a cumulative value, and every time they're written out to the database the previous values in the database are overwritten.

Therefore, if you're interested in keeping track of the performance statistics by hour of the day, for example, what you'd need to do is have some automated routine (not supplied by Net.Commerce) to copy the contents of the table to another table every so often in a manner that will allow you to keep historical data like that.

### **What is the USRTRAFFIC table, and for what is it used?**

The USRTRAFFIC table is designed to hold information on the activity of your shoppers. With this function turned on (see below), the activities of your shoppers will be logged into this table. You may then use this information as a raw source of information for analyzing browsing and buying patterns.

The fields in this table are described in the online help under the index heading of "USRTRAFFIC Table".

The function is enabled by placing the following line:

```
USERTRAFFIC_LOG 1
```

in the server configuration file:

```
/usr/lpp/NetCommerce/html/en_US/<inst_name>/ncommerce.conf
```

**Note:** you must stop and start again Net.Commerce to make this change take effect.



---

## Error Symptoms and Their Causes

**Note!** This section is intended to be a source for information on commonly encountered problems only. It is *not* the authoritative source on all error messages. The Net.Commerce *Messages and Codes* manual serves that purpose. Please refer to "What documentation is available for Net.Commerce V3.1.2?" on page 14 for more information on that manual.

### Message "nc3\_common.so was not found" running CMNCONF

#### **Symptom Detail:**

When trying to create a new instance using CMNCONF, you get the following message:

```
CEE3501S The module nc3_common.so was not found.  
From entry point encrypt_passwd(const char*,char*) at  
compile unit offset +000000CE at address 09487FB6.
```

#### **Cause:**

The CMNCONF utility can't find a copy of `ncommerce.envvars` in the `/etc` directory.

#### **Resolution:**

Provide the `/etc` directory a copy of your updated `ncommerce.envvars` file. Refer to "What exactly am I to do with the Net.Commerce environment variables file?" on page 26 for instructions on this process.

### Message "Cannot create directory" running CMNCONF

#### **Symptom Detail:**

Pressing the "enter" key on the second panel of the "System Configuration" option of CMNCONF (Option 1), you get a "Cannot create directory" message in the upper right part of the screen.

PF1 shows more detail of the error:

```
CMN0910E The mkdir command for  
/usr/lpp/NetCommerce/html/en_US/<inst_name> with  
permissions 755 failed.
```

#### **Cause:**

The TSO ID under which you are running CMNCONF must have the authority to create directories in the HFS. Typically, this will require superuser authority because directories are made in a number of different places.

#### **Resolution:**

Make sure your ISPF shell has superuser authority at the time of running CMNCONF.

**CMNCONF Message: CMN7270E Could not connect to database, caf return code = -1006.**

#### **Symptom Detail:**

When you tried to create the database using Option 3 of CMNCONF, you get the following error message back on your ISPF screen:

```
db2_load_caf failed return_code 8 , reason_code f30002
CMN7270E Could not connect to database, caf return code = -1006.
Possible reason is an incorrect plan name or subsystem id.
```

***Cause:***

DB2 was not active when you attempted to create the database.

***Resolution:***

Start DB2.

**CMNCONF database create fails, CMN7270E appears on screen.**

***Symptom Detail:***

When you exercise Option 3 of CMNCONF (Create Database), you get the following error message on your ISPF screen:

```
CMN0885I Processing started - this operation can take several
minutes.
db2_load_caf failed return_code c , reason_code f30040
CMN7270E Could not connect to database, caf return code = -1006.
Possible reason is an incorrect plan name or subsystem id.
***
```

***Cause:***

The plan name you specified for the Net.Commerce configuration was not bound prior to trying to create the database.

***Resolution:***

Run the appropriate BIND job and bind the database plan prior to attempting to create the database. See the bullet "Binding the DBRM to the Net.Commerce plan" on page 23 for information on what sample bind jobs are provided.

**Net.Commerce fails to start; db2\_load\_caf failed return\_code 8 , reason\_code f30002 in SYSPRINT of Net.Commerce**

***Symptom Detail:***

When attempting to start Net.Commerce, it fails immediately. The message shown in this item's title is found in the SYSPRINT of the Net.Commerce task.

***Cause:***

DB2 was not up and active when you tried to start Net.Commerce

***Resolution:***

Start DB2.

**Net.Commerce fails to start; SYSPRINT shows CMN0540E -1019 and SQLCODE = -204 messages.**

***Symptom Detail:***

SYSPRINT shows the following errors:

```
CMN0540E - Reporting Error. Internal Return Code = -1019
DSNT408I - SQLCODE -204, ERROR: <db_owner_id>.POOLS IS AN
UNDEFINED NAME
```

**Cause:**

The database has not been created.

Upon startup, Net.Commerce attempts to read the content of the POOLS table of the database to retrieve critical startup information. If the database has not been created, the query to the DB2 system for <db\_owner\_id>.POOLS will fail. Net.Commerce will then be shut down.

**Resolution:**

Run CMNCONF and create (option 3) and load (option 4) the database.

**Net.Commerce fails to start; S222 ABEND****Symptom Detail:**

When starting Net.Commerce a S222 ABEND occurs.

**Cause:**

The PUBLIC ID has not been granted EXECUTE on the plan. This is one of the last steps performed in the sample BIND jobs (CMNBIND). However, if you use the sample BIND job supplied with DB2 (DSNTIJCL) for a Net.Commerce database that uses Websphere Application Server, the GRANT statement is not included. It's very easy to forget to do this.

**Resolution:**

Execute the following SQL from SPUFI:

```
grant execute on plan mserver to cmnsrv;
```

And then restart Net.Commerce.

**Net.Commerce starts; CMN0950E on browser; SQLCODE = 100 in SYSPRINT****Symptom Detail:**

SYSPRINT shows the following errors:

```
CMN0540E - Reporting Error. Internal Return Code = -1013
DSNT404I - SQLCODE = 100 NOT FOUND: ROW NOT FOUND FOR FETCH,
          UPDATE, OR DELETE, OR THE RESULT OF A QUERY IS AN EMPTY TABLE
          :
          :
ERR:CMN0102E The KeyManager could not initialize properly.
ERR:CMN0001E The database returned the error code -1013.
ERR:CMN0101E The Command Manager could not initialize properly.
          Terminate.
```

**Cause:**

The database has been created, but not loaded. The tables are empty.

**Resolution:**

Run CMNCONF and load the database (option 4).

**Error 404 on browser; CMN0402E in Webserver SYSPRINT with Director trace on****Symptom Detail:**

Access to Net.Commerce HTML pages works. Access attempt on any Net.Commerce function that executes a Net.Commerce command results in an

"Error 404" on the browser screen, and the following in the SYSPRINT of the Webserver when the Director Trace is enabled:

```
CMN0402E Failed to open file:  
  '/usr/lpp/internet/server_root/pub/ncommerce.conf
```

**Cause:**

You specified an IP address rather than host name in browser URL location window.

**Resolution:**

Use a host name (www.your\_host.com) rather than an IP address.

**Error 404 on browser, ICH408I on system console**

**Symptom Detail:**

After Net.Commerce has been started, any access attempt on a Net.Commerce function results in an "Error 404" on the browser screen, and the following message appears on the system console:

```
ICH408I USER(PUBLIC ) GROUP(EXTERNAL) NAME(#####)  
  /usr/lpp/NetCommerce/html/en_US/<inst_name>/ncommerce.conf  
  CL(DIRSRCH ) FID(01E6E2D3E2D4E200010F000000000003)  
  INSUFFICIENT AUTHORITY TO LOOKUP  
  ACCESS INTENT(--X) ACCESS ALLOWED(OTHER ---)
```

**Cause:**

The permission bits on all directories leading down to the `ncommerce.conf` file, and the `ncommerce.conf` file itself, need to provide execute. By default Net.Commerce sets the permission bits at 755. If for whatever reason those permissions are set so the "execute" bit is off, then this error will occur.

Once again, this problem will surface if *any* of the directories leading down to the file grant insufficient access. It doesn't have to be just the lowest directory.

**Resolution:**

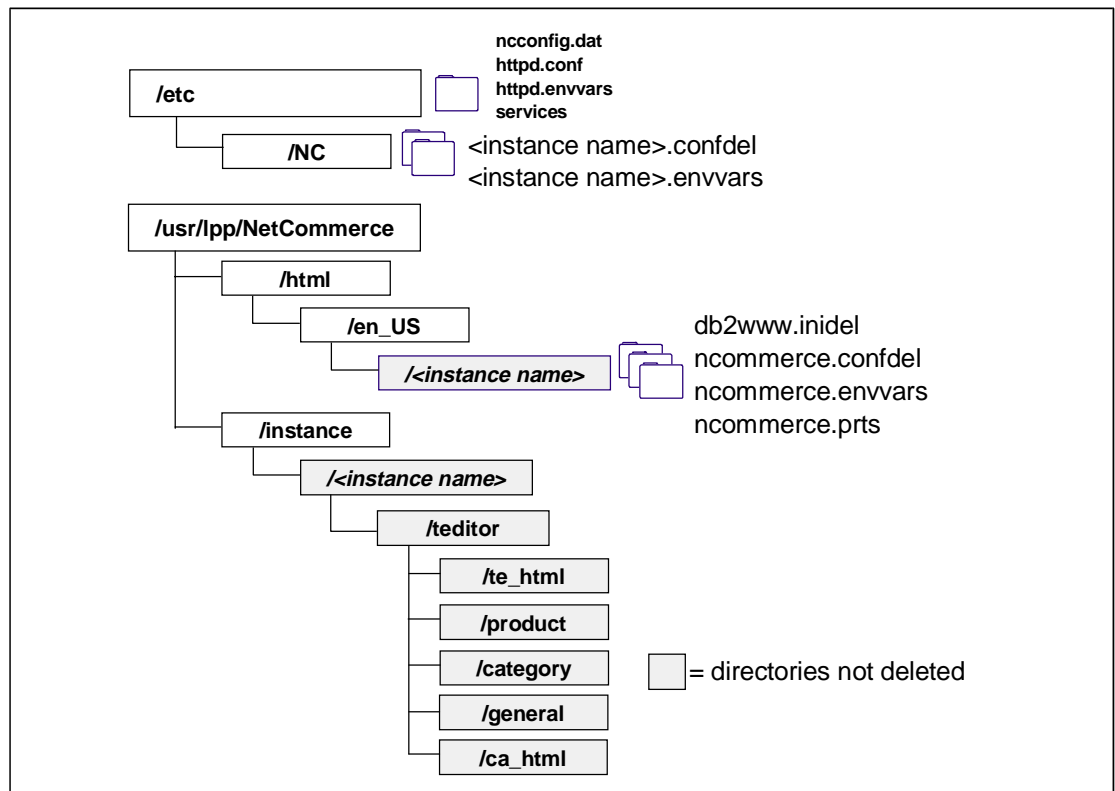
Check the mode fields on all the directories leading down to the HTML document root of Net.Commerce (where `ncommerce.conf` resides) and check the mode field of the `ncommerce` file itself. All must have the execute bit on. If any of them do not, change it so it does.

**Instance directories and files not deleted when instance deleted in CMNCONF.**

That's Net.Commerce working as designed. When you run the delete option of CMNCONF to remove an instance, all it does is remove the directives for that instance from the Web Server's configuration file (`/etc/httpd.conf`), and remove the information about the instance from the `/etc/nconfig.dat` file.

**Note:** The information in `/etc/httpd.envvars` and `/etc/services` will remain after the instance is deleted.

That leaves the following things on your system:



**Note:** The configuration and "ini" files are renamed for any instance that's been deleted.

What this is saying is the just because you delete an instance using CMNCONF, a lot of stuff is still left on your system. If you wish to really delete an instance, you must go in and manually delete these items.

### Custom "Pass" directives being overwritten in Webserver's configuration file.

#### **Symptom Detail:**

Hand-edited "Pass" directives added to the Webserver's httpd.conf file between the Net.Commerce section delimiters are being overwritten whenever the CMNCONF utility is used to modify the configuration.

#### **Cause:**

This is Net.Commerce working as designed. The block of static HTML "pass" directives will be overwritten each time the "enter" key is pressed on the panel of the "Access Control" option of CMNCONF (Option 2).

#### **Resolution:**

Move your custom "Pass" directives immediately after the closing Net.Commerce block delimiter.

### Customization of db2www.ini file lost when CMNCONF run

#### **Symptom Detail:**

Custom changes to the db2www.ini file located in the /usr/lpp/NetCommerce/html/en\_US/<instance\_name> directory are lost when CMNCONF is run.

**Cause:**

The CMNCONF "Access Control" step overwrites information in the db2www.ini file.

**Resolution:**

Make a backup of the db2www.ini file and replace the custom information after the CMNCONF run.

**Changes to Net.Data macros not showing up on browser****Symptom Detail:**

Modifications are made to a Net.Data macro, and yet when you run the Net.Commerce function that displays the macro, the changes do not appear. Clearing the browser cache does not resolve the problem..

**Cause:**

There are two known causes for this:

1. The Net.Commerce caching function is enabled, and the page represented by the Net.Data macro you're changing is in cache.
2. A Net.Data macro with the same name as the one you're modifying exists in a directory located closer to the front of the MACRO\_PATH directive in db2www.ini. This is particularly common when you're modifying Net.Data macros supplied with the sample malls. As part of creating the configuration, some macros are copied into the /usr/lpp/NetCommerce/instance/teditor directories, and those are located closer to the front of the MACRO\_PATH directive than is /usr/lpp/NetCommerce/macro/en\_US.

**Note:** it's possible that *both* of these causes is at work ... check both.

**Resolution:**

The resolution for each of the above two causes is shown below:

1. Clear the cache (see "How do I clear pages from cache?" on page 68), or turn the caching function off and restart Net.Commerce (see "How does the Net.Commerce caching function work?" on page 64).
2. Determine if a macro by the same name is located in a directory found earlier on the MACRO\_PATH. If so, modify that macro, or rename it so it is not found before the one you are working on is found.

**"CEE3512S An HFS load of <module name> failed."****Symptom Detail:**

When running mass import, NCCLEAN, or any other Net.Commerce utility from the OMVS shell, you receive messages that looks like this:

```
CEE3512S An HFS load of module nc3_dbc_db2.so failed.
The system return code was 000000157;
the reason code was 0BDF03B2.
From entry point __dllstaticinit at compile
unit offset +000008C6 at address 0941293E.
[1] + Done(137) massimpt
16777240      Killed ./massimpt
```

```
CEE3512S An HFS load of module nc3_containers.so failed.
The system return code was 0000000157;
the reason code was 0BDF03B2.
From entry point __dllstaticinit at compile
unit offset +0000060C at address 0940E92C.
[1] + Done(137) ncclean
33554456      Killed  ./ncclean
```

**Cause:**

This is due to the Net.Commerce *hlq.SCMNLMOD* dataset not being accessible from the OMVS shell environment. Either *hlq.SCMNLMOD* is not in LNKLIST, or you have not provided a STEPLIB environment variable pointing to the dataset.

**Resolution:**

Place your Net.Commerce load dataset (*hlq.SCMNLMOD*) in your LNKLIST and activate it, or provide a STEPLIB variable in your */etc/profile* and export the variable:

```
STEPLIB='hlq.SCMNLMOD'
export STEPLIB
```

**"CEE3501S The module nc3\_dbc\_db2.so was not found"**

**Symptom Detail:**

On OMVS screen while attempting to run the *massimpt* (or one of the other utilities such as *cmndb2* or *loadtbls*) utility you get:

```
CEE3501S The module nc3_dbc_db2.so was not found.
From entry point __dllstaticinit at compile unit offset
+000008C6 at address 0811293E.
[1] + Done(137) massimpt
100663321      Killed  ./massimpt
```

**Cause:**

The environment variables for the OMVS session under which you are running Mass Import are not set up properly. Specifically, the LIBPATH variable does not include a reference to:

```
/usr/lpp/NetCommerce/lib
```

or the variable has not been "exported" to the environment.

**Resolution:**

Update the environment variables file to include the reference to this directory. Verify that the environment variable is there by issuing the SET command from the OMVS shell to list out the environment variables for your session. Look for the LIBPATH entry and make sure that */usr/lpp/NetCommerce/lib* is listed.

**"CEE3501S The module nc3\_containers.so was not found."**

**Symptom Detail:**

On OMVS screen while attempting to run the *ncclean* utility you get:

```
CEE3501S The module nc3_containers.so was not found.
From entry point __dllstaticinit at compile unit offset
```

```
+0000060C at address 0810E92C.  
[1] + Done(137) ncclean  
218103832 Killed ./ncclean
```

**Cause:**

The environment variables for the OMVS session under which you are running Mass Import are not set up properly. Specifically, the LIBPATH variable does not include a reference to:

```
/usr/lpp/NetCommerce/lib
```

or the variable has not been "exported" to the environment.

**Resolution:**

Update the environment variables file to include the reference to this directory. Verify that the environment variable is there by issuing the SET command from the OMVS shell to list out the environment variables for your session. Look for the LIBPATH entry and make sure that /usr/lpp/NetCommerce/lib is listed.

**"ERROR : Message file 'nc3\_util.cat' could not be opened!"**

**Symptom Detail:**

On OMVS screen while attempting to run the massimpt utility or the ncclean (or cmndb2 or loadtbls) utility you get:

```
ERROR : Message file 'nc3_util.cat' could not be opened!
```

```
ERROR : Message file 'nc3_util.cat' could not be opened!
```

```
ERROR : Message file 'nc3_util.cat' could not be opened!
```

**Cause:**

The environment variables for the OMVS session under which you are running Mass Import or NCCLEAN are not set up properly. Specifically, the NLSPATH variable does not include a reference to:

```
/usr/lpp/NetCommerce/msg/en_US/%N
```

or the variable has not been "exported" to the environment.

**Resolution:**

Update the environment variables file to include the reference to this directory. Verify that the environment variable is there by issuing the SET command from the OMVS shell to list out the environment variables for your session. Look for the NLSPATH entry and make sure that

```
/usr/lpp/NetCommerce/msg/en_US/%N is listed.
```

**"CMN7009E Selecting the store name <store name> from the table MERCHANT failed, (SQL class) return code = -1012."**

**Symptom Detail:**

When you run the massimpt utility and look in the Mass Import log, you see the following messages:

```
CMN7304I Started reading input file. Processing began.
```



```
CMN7303I Connected successfully to the database.
CMN7009E Selecting the store name <store name> from the table
    MERCHANT failed, (SQL class) return code = -1012.
SELECT merfnbr FROM <ID>.merchant WHERE mestname='<store name>'
CMN7010E Program terminated because of error in identifying the
    store.
CMN7302I The following transaction failed...
[00001]#STORE;<store name>
```

```
Number of Successful Transactions : 0
Number of Failed Transactions : 1
```

```
CMN7311W Processing terminated before reaching the end of the
    input file.
```

Where <ID> is the RACF ID that owns the database, and <store name> is the name of the merchant that is required on the first card of the Mass Import source deck.

**Cause:**

The TSO ID under which you run the massimpt utility must have *at least* DBADM authority on the Net.Commerce database. Absent that authority, you receive these messages when TSO ID attempts a SELECT on a database for which it has no DBADM authority.

**Resolution:**

From a TSO ID with the authority to grant DBADM authority to another ID, issue the following SQL:

```
grant dbadm on database <database_name> to <ID>;
```

Where <database\_name> is the name of the Net.Commerce database, and <ID> is the name of the TSO ID under which you will run massimpt. Then re-run the Mass Import job.

**NCCLEAN: "db2\_load\_caf failed return\_code 8 , reason\_code f30034"**

**Symptom Detail:**

While attempting to run the ncclean utility you get the error shown above on the OMVS screen, and the following in the log file created:

```
CMN7501I Database Cleanup Utility started.
CMN7269E Could not connect to database, caf return code = -1006.
Possible reason is an incorrect plan name or subsystem id.
CMN7202E Database Cleanup Utility encountered an unexpected error.
```

**Cause:**

The TSO ID under which you run the ncclean utility must have SYSADM authority to perform its work.

**Resolution:**

From a TSO ID with the authority to grant SYSADM authority to another ID, issue the following SQL:

```
grant sysadm to <ID>;
```

Where <ID> is the TSO ID under which you will run ncclean. Then re-run the NCCLEAN job.

**Note:** the `massimpt` utility requires only DBADM. We have not yet tested `cmndb2` or `loadtbls` to see what authority they require. We suspect you'd see the same message for those two utilities with only DBADM. Certainly, having the authority SYSADM will cover all your bases.

**"CMN7267E There is no value for DBOWNER, MS\_DBSSID, or MS\_DBPLAN in the configuration file."**

***Symptom Detail:***

When running either the `massimpt` or `ncclean` utilities, you get the message shown in above on the OMVS screen, and no log file is created.

***Cause:***

The `-f` parameter of both `massimpt` and `ncclean` is to provide an easy way to provide the database information, rather than typing each value for DBOWNER, MS\_DBSSID or MS\_DBPLAN individually. This parameter simply points to the Net.Commerce configuration file, where all the information is kept.

The Net.Commerce configuration file is located at:

```
/usr/lpp/NetCommerce/html/en_US/<inst_name>/ncommerce.conf
```

Where `<inst_name>` is the name of your instance.

However, if you mistype this string -- *even one letter off from correct* -- the utility will not be able to open the file and find the information it requires. It will act as if the location you provided simply doesn't have the information it needs, rather than saying that the location you provided is wrong.

***Resolution:***

Double (and triple) check the exact syntax of that parameter to make sure the utility has access to the correct file.

**Caching function not working and Performance Monitor not working; SQL -551**

***Symptom Detail:***

When attempting to use the functions shown above, you find that no data is ever written into the NC\_CACHE or PERFLOG tables. A look in the SYSPRINT of the Webserver (not Net.Commerce) shows SQL -551 errors.

***Cause:***

The ID under which the Webserver (not Net.Commerce) runs must have DBADM authority on the Net.Commerce database. Without that authority, the Webserver's ability (actually, the Net.Commerce API code, but it runs in the Webserver's address space, so it operates under the Webserver's ID) to update those tables is lost.

***Resolution:***

From a TSO ID with the authority to grant DBADM authority to another ID, issue the following SQL:

```
grant dbadm on database <database_name> to <ID>;
```

Where `<database_name>` is the name of the Net.Commerce database, and `<ID>` is the name of the Webserver's ID.

## Maintenance Information

### APAR/PTFs against Net.Commerce V3.1.2 (FMID H01B312)

| PTF     | APAR(s)                       | Prereq  | Coreq | Avail.     | Desc.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------|-------------------------------|---------|-------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UQ34468 | PQ27986<br>PQ29113<br>PQ29135 | Base NC | none  | 20-Sept-99 | <ul style="list-style-type: none"><li>• Allow Internet Explorer to function seamlessly with the NetCommerce flow. This was done by working around an IE known defect.</li><li>• The error message CMN0301E was re-coded to be a trace message instead of an error message. It is more of an informational message than an error message.</li><li>• The RACF authentication code was fixed so that a valid RACF user was allowed access into the NetCommerce site.</li><li>• The latest NetData dlls and DBRM were packaged in this PTF, and they fixed the erroneous HTML text problem as well as added some performance enhancements.</li><li>• Some NCAdmin javascript was changed so that dates entered HTML textbox field were rejected if those dates had already passed.</li></ul> |

#### Known bugs not yet included in a PTF

- **Potential security hole in Net.Commerce**

The NCADMIN utility is used to administer the Net.Commerce site. The utility uses a good many Net.Data macros that at present can be directly accessed by someone who knows about the ExecMacro command and knows the directory structure and file structure of Net.Commerce.

You can block direct access to these Net.Data macros by placing the following commands in your webserver's httpd.conf file. They would appear *immediately after* the Net.Commerce block found at the top of the file:

```
Fail /cgi-bin/msrvr/;execmacro/ncadmin/*  
Fail /msprotect/msrvr/;execmacro/ncadmin/*  
Fail /cgi-bin/ncommerce3/ExecMacro/ncadmin/*  
Fail /msprotect/ncommerce3/ExecMacro/ncadmin/*
```

This serves to block any attempted access directly to the Net.Data macros kept in the /ncadmin directory.

- **Incorrect SQL created by Template Designer for category table and product table**

The Template Designer fails to insert a critical "where" condition in the SQL it creates a category table or product table. Without this condition, all the categories in the mall will appear in the category table, or all the products in the mall will appear in the product table.

The SQL created by the Template Designer for a category table will look something like this:

```
select CATEGORY.CGRFNBR, CATEGORY.CGMENBR,
       CATEGORY.CGNAME, CGRYREL.CRSEQNBR
from CATEGORY, CGRYREL
where crpcgnbr=$(cgrfnbr) and crmenbr=$(cgmenbr) and cgpud=1
```

*What's missing from this is an **and CRCCGNBR=CGRFNBR** condition on the where clause.*

Similarly, the product table SQL will look something like this:

```
select PRODUCT.PRRFNBR, PRODUCT.PRMENBR, PRODUCT.PRSDESC,
       CGPRREL.CPSEQNBR
from PRODUCT, CGPRREL
where cpcgnbr=$(cgrfnbr) and cpmenbr=$(cgmenbr) and prpub=1
```

*What's missing from this is an **and CPPRNBR=PRRFNBR** condition on the where clause.*

You may correct this by either hitting the "Edit SQL" button on the object dialog box for the category or product table and then typing in the missing "where" condition, or you may edit the macro that's created and add it to the SQL in the actual file.

- ***Currency conversion rate limited to two decimal places***

When a Net.Commerce instance starts up, it populates a table called CURRCCACHE by reading values from a table called CURRCONV and stuffing them into CURRCCACHE. Values appear okay in CURRCONV (for example, 3.141592653) but appear truncated CURRCCACHE (3.140000000 as an example).

- ***Logoff Command doesn't clear both cookies***

Net.Commerce sets two cookies when a user logs onto Net.Commerce: ENDCT\_ID= and SESSION\_ID=. The Logoff command is designed to clear those cookies. That has the effect of returning the browser to a state where Net.Commerce sees it as a new shopper connecting to the site for the first time.

Unfortunately, the Logoff command only resets the ENDCT\_ID= cookie. The SESSION\_ID= cookie remains. Therefore, Net.Commerce continues to recognize the browser as belonging to the previously logged on user.

---

## Index

### 0

#### #ROWDELIMITER

- command of Mass Import, 71
- /etc/nconfig.dat
  - as it relates to CMNCONF, 27
- /etc/ncommerce.envvars
  - modifying during installation, 26
- /etc/services
  - as it relates to CMNCONF, 27
  - updated during configuration, 29
  - warning about HFS vs. MVS copies, 29

### 4

443

- using other than port 443 for SSL, 38

### 8

80

- using other than port 80, 38

## A

#### ABEND

- S222, 81

#### Access Control

- function of NCADMIN, 54

#### Administrative Function, 51

- access control, 54
- accessing, 52
- creating new userid, 54
- default userid and password, 52
- requirement to use, 51
- Site Manager, 51
- Store Creator, 51
- Store Manager, 51
- Template Designer, 51

#### API

- using V3.1.1 APIs, 63

#### Attributes, 18

#### Authentication

- BASIC directive for SAF, 39
- differences between DB2 and SAF, 38
- Protect directive update in httpd.conf for SAF, 39
- reason for using SAF, 39
- using SAF authentication, 39

## B

#### BIND

- how to use DSNTIJCL job, 24
- of the Net.Commerce DBRM, 23

#### Browsers

- what works with Net.Commerce, 18

## C

### C++

- location of OFs, 63
- migrating OFs from NT/AIX to OS/390, 10
- overview of Overridable Function, 62

### Caching Function

- caching other than Product/Category pages, 66
- clearing pages from cache, 68
- confusion that can result when enabled, 84
- getting NC\_CACHE\_LIST to take effect, 67
- how it works, 64
- not working, 88
- populating pages, 65
- settings for NC\_DMN\_CACHE, 64

### Call Level Interface (CLI)

- initialization file for Product Advisor function, 43

### Cannot create directory

- error message running CMNCONF, 79

### Category Pages

- caching, 64
- caching other than, 66
- created using Template Designer, 54
- special consideration regarding custom directories, 59

### Category Table

- bad SQL from Template Designer, 89

### CategoryDisplay

- where it looks for Net.Data macros, 59

### CEE3501S

- error running mass import, 85
- error running NCCLEAN, 85

### CEE3512S

- error running utilities, 84

### Changes

- to configuration files and restarting the servers, 57

### CMN0402E

- in SYSPRINT when connecting to Net.Commerce, 81

### CMN0540E

- while starting Net.Commerce, 80

### CMN0910E, 79

### CMN0950E

- error on browser, 81

### CMN7009E

- when running mass import, 86

### CMN7267E

- when running mass import or NCCLEAN, 88

### CMN7270E, 79

### CMNBIND, 23

### CMNBNDBB, 23

### CMNBNDEW, 23

### CMNCONF

- /etc/services taking precedence over MVS services dataset, 29
- and definition of instance, 27
- brief definition of configuraton utility, 25

- can't create new instance, 79
- does not delete directories, 82
- getting all configuration files into single directory, 30
- input to, 27
- losing custom changes to db2www.ini, 83
- multiple people using at same time, 35
- number of instances you may configure, 35
- output from, 28
- overwriting custom Pass directives, 83
- post-configuration activity, 34
- reason for configuring multiple instances, 35
- sharing databases, 35
- using directories created by, 60
- warning about overwriting db2www.ini changes, 60
- warning about overwriting httpd.conf updates, 37
- CMNISHFS, 22
- CMNISMKD, 22
- CMNMKDIR, 22
- CMNSRV
  - RACF userid for Net.Commerce, 26
- Comments
  - in macros created by Template Designer, 55
  - in Mass Import input file, 72
- Configuration
  - definition of instance, 27
  - input to CMNCONF utility, 27
  - making modifications to in 24 x 7 site, 47
  - multiple people using CMNCONF at same time, 35
  - number of databases that may be used, 35
  - number of instances you may configure, 35
  - output from CMNCONF utility, 28
  - protecting configuration files, 36
  - reason for configuring multiple instances, 35
  - Server Controller files, 30
  - Server files, 30
  - sharing databases, 35
  - updating ncommerce.conf file, 35
  - utility to use, 25
  - when changes to configuration require restart, 57
- Console Message
  - to indicate Net.Commerce has started successfully, 46
  - to indicate Webserver has started successfully, 46
- Could not connect to database
  - error message running CMNCONF, 79
- Customization
  - changes to Net.Data macros not showing on browser, 84
  - creating custom HTML file directory, 58
  - creating Net.Data macro directory, 59
  - custom Pass directives lost when CMNCONF run, 83
  - getting Net.Commerce to recognize OFs, 62
  - JPG vs. GIF images, 17
  - overview of Overridable Function (OF), 62
  - referencing subdirectories off MACRO\_PATH, 60
  - storing your custom OFs, 63
  - to db2www.ini lost when CMNCONF run, 83
  - turning off SSL, 37

- using directories created by CMNCONF, 60
- what can be customized, 13
- why Net.Commerce is a toolkit, 9

## D

### Database

- authority for Webserver ID and Product Advisor, 44
- authority needed to run NCCLEAN, 87
- BIND, 23
- Call Level Interface (CLI) installation, 44
- cleaning records with other tools, 74
- DBADM on Webserver ID, 34
- error when created but not loaded, 81
- failing to give webserver ID DBADM, 88
- migrating from NT/AIX to OS/390, 10
- NCCLEAN cleanup utility, 73
- number used with Net.Commerce, 35
- plan to use with Java Servlet support, 44
- sharing between instances, 35
- space used by store records, 16
- using DSNTIJCL BIND job, 24
- database create fails
  - error running CMNCONF, 80
- DB2
  - authority and webserver ID with Product Advisor, 44
  - authority needed to run mass import utility, 87
  - CLI initialization file, 43
  - CLI installation for Product Advisor, 44
  - differences with SAF authentication, 38
  - error when PUBLIC ID not granted execute on plan, 81
  - loading records using Mass Import, 69
  - location name, 43
  - other DBMS that can be used, 17
  - running with Net.Commerce, 17
  - sample BIND jobs for databases, 23
  - using DSNTIJCL BIND job, 24
- db2\_load\_caf failed return\_code 8
  - when running NCCLEAN utility, 87
- db2www.ini
  - custom changes lost when CMNCONF run, 83
  - protecting Net.Data file, 36
  - referencing subdirectories off MACRO\_PATH, 60
  - updating to point to custom directory, 59
  - warning about overwriting of changes, 60
- DD\_NAME parameter, 50
- Default
  - userid and password of NCADMIN, 52
- Description
  - of Net.Commerce, 8
- DirAccess
  - directive in httpd.conf file, 35
- Director
  - dynamic start of trace, 48
  - static start of trace, 48
- Directory
  - custom for HTML files, 58

Disk Space

- occupied by online store, 16
- required to install Net.Commerce, 21

Document Root

- location of Net.Commerce document root, 30

Documentation

- available for V3.1.2, 14
- examples of backend integration, 15
- most recent version of this document, 4
- online and whether Net.Commerce needs to be running, 15

Domino Go Webserver

- related to Websphere, 20
- V4.6.1 vs. V5, 21

Double Login Problem

- with %%SAF%% authentication, 40

DSNTIJCL

- DB2 BIND during installation, 23
- does not contain GRANT for plan to PUBLIC ID, 25
- how to use with Net.Commerce, 24

Dynamic Restart

- of Net.Commerce, 47
- of the Webserver, 47

**E**

Education

- available for Net.Commerce, 15

Environment Variable

- modifying Net.Commerce's during installation, 23
- ncommerce.envvars in /etc directory, 27
- problems when not set for mass import or NCCLEAN, 86
- problems when not set for mass import utility, 85
- procedure for modifying Net.Commerce envvars, 26, 50
- sample envvars file supplied with Net.Commerce, 28

Error 404 on browser

- when connecting to Net.Commerce, 81, 82

ErrorPage

- directive in httpd.conf, 36

Errors

- monitoring frequency of errors, 75

**F**

F30034

- SQL reason code upon Net.Commerce failing to start, 25

Fail Directive

- of Webserver, used to block access to files, 36

FAQ

- electronic location, 4
- most recent version of this FAQ, 4

Features

- of OS/390 NC vs. NT or AIX, 10

Firewall

- part of security plan, 15

shipped free with OS/390, 15

FMID

- for Net.Commerce, 20
- for Payment Server, 20

Function

- comparison between platforms, 10

**G**

GIF

- comparison to JPG, 17

Groups

- used to provide access control, 54

**H**

HFS

- disk space for installation, 21
- getting all Net.Commerce configuration files into single directory, 30
- mount-point at installation, 22

htadm

- using to generate password, 52

HTML files

- creating using Template Designer, 54
- custom file location, 58
- migrating from NT/AIX to OS/390, 9

HTML Root

- location of Net.Commerce HTML root, 30

httpd.conf

- custom Pass directive in, 58
- picking up changes with dynamic restart, 47
- preventing directory browsing, 35
- protecting other Net.Commerce files, 36
- updated during configuration, 29
- updates for using SAF, 39
- warning about CMNCONF overwriting updates, 37

**I**

ICH408I

- on console when permission bits not correct, 82

Implementation

- mall vs. store, 8
- overview, 8
- skills required, 13
- time involved, 9

Installation

- disk space required, 21
- minimum programming requirements, 20
- mount-point, 22
- overview of process, 21
- procedure for modifying envvars file, 26
- RACF ID definition, 26
- system preparation, 23

Instance

- definition of, 27
- directories not deleted when instance deleted, 82
- employing multiple to allow 24 x 7, 47

- number of instances you may configure, 35
- reason for configuring multiple, 35
- sharing database, 35
- using the directories off instance subdirectory, 60

**ISPF**

- concatenating Net.Commerce libraries, 24

## **J**

### **Java**

- DB plan to use with Servlet support, 44
- servlet support and Product Advisor, 41

### **JCL**

- sample start procedure, 23

### **JPG**

- comparison to GIF, 17

## **K**

### **Known Bugs**

- Template Designer incorrect SQL, 89

## **L**

### **LIBPATH**

- updating in ncommerce.envvars for custom OFs, 63

### **load failure**

- when running OMVS utilities, 84

### **Login**

- being prompted twice, 40

## **M**

### **MACRO\_PATH**

- custom path for category or product macros, 59
- referencing subdirectories, 60
- updating to point to custom directory, 59

### **Mall**

- differences in sample malls provided, 34
- vs. store configuration, 8

### **Mass Import**

- comments in input file, 72
- example of parameters, 71
- how it works, 69
- improving performance, 72
- invoking, 70
- problems when ID has insufficient DB authority, 86
- SCMNLMOD not in LINKLST, 27
- SCMNLMOD not in LNKLST, 84
- splitting lines in the input file, 71
- using the #ROWDELIMITER command, 71

### **Migration**

- database from NT/AIX to OS/390, 10
- from NT/AIX to OS/390, 9
- HTML files from NT/AIX to OS/390, 9
- Net.Data files from NT/AIX to OS/390, 9
- Overridable Functions from NT/AIX to OS/390, 10
- porting code from NT/AIX to OS/390, 10
- redbook, 9

### **Modifications**

- to configuration files and restarting, 57

### **Modify**

- MVS modify of Net.Commerce to enable trace, 50

### **Monitoring**

- system performance, 75

### **MS\_CACHE\_LIST**

- clearing pages from cache, 68
- note of caution, 67

### **MS\_LOGLEVEL**

- modifying by hand, 49

### **msprotect**

- when using SAF, 40

## **N**

### **N3390**

- IBM education course for Net.Commerce, 15

### **NC\_CACHE**

- clearing pages from cache, 68
- database table and how caching function works, 64

### **NC\_CACHE\_LIST**

- behavior of, 67
- caching function for other than Product/Category pages, 66
- caching parameter format, 66
- how it works, 66
- what is needed to have setting take effect, 67

### **nc3\_common.so**

- not found running CMNCONF, 79

### **nc3\_containers.so**

- when running NCCLEAN, 85

### **nc3\_util.cat could not be opened**

- error when running mass import or NCCLEAN, 86

### **NCADMIN**

- access control function, 54
- accessing, 52
- authority to access, 54
- changing default password, 52
- creating a new userid, 54
- default userid and password, 52
- forgotten password, 52
- requirement to use, 51
- types of macros created by Template Designer, 54
- utility of Net.Commerce, 51

### **NCCLEAN**

- database cleanup utility, 73
- parameter syntax chart, 74
- SCMNLMOD not in LNKLST, 27, 84
- using other tools to delete records, 74
- using to clear cache table, 68

### **ncommerce.conf**

- dynamic restart to pick up changes, 47
- mode fields leading to do not have execute bit on, 82
- protecting, 36
- updating Server configuration file, 35
- used as data input to mass import or NCCLEAN, 88



## ncommerce.envvars

- CMNCONF error if missing from /etc, 79
- modifying, 50
- updating LIBPATH for custom OFs, 63

## Net.Commerce

- and Parallel Sysplex, 19
- and WLM, 19
- authentication of shoppers, 38
- clearing pages from cache, 68
- CMNCONF utility, 25
- comparison of OS/390 to other platforms, 10
- console message to indicate start successful, 46
- definition of instance, 27
- differences between sample malls, 34
- document root directory, 30
- dynamic restart, 47
- dynamic start of director trace, 48
- dynamic start of server trace, 49
- education available, 15
- Failing to start with F30034 SQL reason code, 25
- fails to start, 80, 81
- how caching function works, 64
- how Mass Import works, 69
- how NCCLEAN works, 73
- HTML root directory, 30
- initial testing, 46
- input to CMNCONF utility, 27
- mall vs. store concept, 8
- message and panel libraries, 24
- modifying environment variables during installation, 23
- MVS modify of Net.Commerce, 50
- NCADMIN utility, 51
- number of databases that may be used, 35
- online documentation and whether Net.Commerce needs to be running, 15
- order of start with Webserver, 46
- output of tracing function, 50
- overview of implementing Net.Commerce, 8
- overview of Overridable Functions (OF), 62
- preventing browsing of root directory, 35
- procedure for modifying envvars file, 26, 50
- Product Advisor function, 40
- protecting other Net.Commerce files, 36
- RACF ID definition, 26
- security functions, 15
- Server configuration files, 30
- simple definition, 8
- starting, 46
- static start of director trace, 48
- static start of server trace, 49
- stopping, 46
- toolkit aspect of, 9
- tracing overview, 48
- turning off SSL, 37
- updating ncommerce.conf file, 35
- utilities supplied, 16
- V3.1.1, using macros from, 60
- when a restart is necessary, 57

## Net.Data

- caching generated HTML, 64
- changes to macros not showing on browser, 84
- custom macro directory, 59
- location of online library, 10
- macros created by Template Designer, 54
- referencing subdirectories off MACRO\_PATH, 60
- supplied with Net.Commerce, 19
- using macros from V3.1.1, 60

## Net.Data files

- changes to not showing on browser, 84
- creating custom directory for, 59
- duplicates in directory structure, 84
- migrating from NT/AIX to OS/390, 9
- special consideration when storing category or product pages, 59
- what can be customized, 13

## netstat

- using to check port usage, 41

## no value for DBOWNER

- error when running mass import or NCCLEAN, 88

## NT/AIX

- functional comparison with OS/390, 10
- migrating from Net.Commerce from NT/AIX, 9

## O

### OF

- getting Net.Commerce to recognize, 62
- nested select used to register OF in database, 62
- overview of Overridable Function, 62
- where to store your custom OFs, 63

### OS/390

- migrating Net.Commerce from NT/AIX, 9

### Output

- of tracing function, 50

### Overridable Functions

- getting Net.Commerce to recognize, 62
- migrating NT/AIX to OS/390, 10
- overview, 62
- web location of samples, 15
- where to store your custom OFs, 63

## P

### Parallel Sysplex

- example of where it might apply, 47
- Net.Commerce and DB2, 18
- running Net.Commerce in Sysplex environment, 19

### Pass Directive

- custom in httpd.conf, 58
- overwritten when CMNCONF run, 83

### Password

- changing default NCADMIN, 52
- default of NCADMIN, 52
- resetting forgotten NCADMIN password, 52

### PathPrefix

- in SMP/E pre-APPLY jobs, 22

- recommendation, 22
- PERFLOG**
  - frequency of updates to, 77
- Performance**
  - how caching function works, 64
  - improving Mass Import, 72
  - monitoring, 75
  - tracking across time, 77
- Performance Monitor**
  - enabling, 76
  - frequency of updates to DB2 table, 77
  - function of System Monitoring, 75
  - not working, 88
  - providing time-slice history, 77
  - purging information in database table, 77
  - refreshing data on the screen, 77
- Port 443**
  - using other than for SSL, 38
- Port 80**
  - using other than, 38
- Porting**
  - code from NT/AIX to OS/390, 10
  - guide in PDF format on web, 10
- PQ27986, 89**
  - applying for SAF fix, 39
- Pre-requisites**
  - on system for running Net.Commerce, 20
- Product Advisor**
  - administering and creating catalogs, 44
  - creating pages using Template Designer, 54
  - DB authority needed by webserver, 44
  - enabling so shoppers can use PA, 40
  - overview, 40
  - planning considerations, 18
  - Product Comparison metaphor, 40
  - Product Exploration metaphor, 40
  - Sales Assistant metaphor, 40
- Product Comparison**
  - Product Advisor metaphor, 40
- Product Exploration**
  - Product Advisor metaphor, 40
- Product Pages**
  - caching, 64
  - caching other than, 66
  - created using Template Designer, 54
  - special consideration regarding custom directories, 59
- Product Table**
  - bad SQL from Template Designer, 89
- ProductDisplay**
  - where it looks for Net.Data macros, 59
- Proxy**
  - API function, 63
- Purging**
  - cache table using NCCLEAN, 68
  - pages from NC\_CACHE table, 68

## R

- RACF**
  - definition for PUBLIC id, 15, 23
  - ID for Net.Commerce, 26
  - id used for browser access, 15
- Read Only Caching, 65**
- Read/Write Caching, 64**
- reason code f30002, 79**
- reason\_code f30002**
  - while starting Net.Commerce, 80
- reason\_code f30034**
  - error when running NCCLEAN, 87
- Redbook**
  - migration, 9
- Registration**
  - requirement with SAF, 39
- Restart**
  - when restarting Net.Commerce is necessary, 57

## S

- SAF**
  - and shopper registration, 39
  - BASIC directive update in httpd.conf, 39
  - differences with DB2 authentication, 38
  - double login problem, 40
  - Protect directive update in httpd.conf, 39
  - reason for using, 39
  - using msprotect in URL, 40
  - using SAF authentication, 39
- Sales Assistant**
  - Product Advisor metaphor, 40
- Sample**
  - database BIND jobs, 23
  - differences in sample malls provided, 34
  - JCL start procedure, 23
  - JCL start procedure for Net.Commerce, 46
- Security**
  - DB2 vs. SAF authentication, 38
  - firewall, 15
  - functions available with Net.Commerce, 15
  - preventing browsing of Net.Commerce root directory, 35
  - Protect directive update in httpd.conf for SAF, 39
  - protecting Net.Commerce files, 36
  - RACF ID definition for Net.Commerce, 26
  - reason for using SAF authentication, 39
  - turning off SSL, 37
  - using msprotect in URL with SAF, 40
  - using SAF authentication, 39
  - using the Fail directive, 36
- Server**
  - configuration files, 30
  - dynamic start of trace, 49
  - static start of trace, 49
  - updating server configuration file, 35
- Server Controller**

- configuration files, 30
- Servlet
  - support and Product Advisor, 41
- Site Manager
  - overview, 51
- Skills
  - required during implementation, 13
- SMP/E
  - CMNISHFS pre-APPLY job, 22
  - CMNISMKD pre-APPLY job, 22
  - CMNMKDIR pre-APPLY job, 22
  - disk space for target and distribution libraries, 21
  - HFS mount-point at installation, 22
  - overview of installation process, 21
- Space
  - disk space used by online store, 16
- SPUFI
  - using rather than NCADMIN, 51
  - using to clear cache, 68
- SQL -551
  - in SYSPRINT, 88
- SQLCODE= -100
  - error in SYSPRINT, 81
- SQLCODE= -204
  - while starting Net.Commerce, 80
- SSL
  - turning off, 37
  - using other than port 443, 38
- Starting
  - Net.Commerce, 46
  - Net.Commerce fails to start, 80
  - order of Net.Commerce and Webserver, 46
- Stopping
  - Net.Commerce, 46
- Store
  - vs. mall configuration, 8
- Store Creator
  - overview, 51
  - reason to use it, 56
- Store Manager
  - overview, 51
- SYS1.PARMLIB
  - effect on OMVS utilities, 27
  - update during installation, 23
  - using Net.Commerce before changes take effect, 27
- SYSADM
  - authority for webserver ID and Product Advisor, 44
- System Monitoring
  - enabling director trace, 49
  - performance monitoring function, 75
  - using to clear cache pages, 69
  - using to enable caching function, 64
- T**
- TCPIP.ETC.SERVICES
  - /etc/services taking precedence, 29

- Template Designer
  - example of macro created, 55
  - overview, 51
  - types of macros it can create, 54
- Testing
  - initial testing of Net.Commerce, 46
  - tracing Net.Commerce, 48
- Tracing
  - dynamic start of director trace, 48
  - dynamic start of server trace, 49
  - output of tracing functions, 50
  - overview, 48
  - removal of -tr parameter from V3.1.1 code, 50
  - static start of director trace, 48
  - static start of server trace, 49
- Tracking
  - shopper activity, 77
- U**
- URL
  - of NCADMIN function, 52
- Userid
  - changing default password for NCADMIN, 52
  - creating new NCADMIN ID, 54
  - default of NCADMIN, 52
- USERTRAFFIC
  - table used to track shopper traffic, 77
- USERTRAFFIC\_LOG
  - setting in ncommerce.conf, 77
- Utilities
  - CMNCONF, 25
  - how Mass Import works, 69
  - how NCCLEAN works, 73
  - improving mass import performance, 72
  - NCADMIN, 51
  - supplied with Net.Commerce, 16
- W**
- Webserver
  - and WLM, 19
  - configuration file updated during configuration, 29
  - console message to indicate start successful, 46
  - creating custom HTML file directory, 58
  - custom Pass directive, 58
  - DB authority needed for Product Advisor function, 44
  - DBADM for ID, 34
  - different flavors of, 20
  - dynamic restart, 47
  - Fail directive, 36
  - failing to give ID DBADM, 88
  - order of start with Net.Commerce, 46
  - preventing directory access, 35
  - when a restart is required, 57
- Website
  - where this document is kept, 4
- Websphere

how it relates to DGW, 20  
WLM  
in Sysplex environment, 19  
what's enabled, 19