Chapter 8. Configuring the DNS Server

The Domain Name System (DNS) server — commonly referred to as simply a *name server* — maps a host name to an internet address or an internet address to a host name. To configure the DNS server virtual machine, you must perform the following steps:

DNS Server Configuration Steps

1. Update the TCPIP server configuration file.
2. Update the DTCPARMS file for the DNS server.
3. Determine the type of DNS server to configure. The section titled "DNS Server Overview" on page 134 may help in this determination.
   - Configure a Caching-Only name server:
     a. Customize the DNS server configuration file to include a **CACHINGONLY** statement.
     b. In order for the name server to be able to contact the root name server for queries that it cannot resolve itself, specify on the CACHINGONLY statement the name of a root cache file containing information that is used for contacting the root name server.
   - Configure a Primary name server:
     a. Customize the DNS server configuration file to include **PRIMARY** and **SECONDARY** statements.
     b. Prepare the DB2 Server for VSE & VM database.
     c. Define the DB2 database.
     d. Define the appropriate resource records for your installation.
     e. Create a MASTER DATA file.
     f. Install the name server database.
     g. If you are not using a FORWARDERS statement, you must customize the DNS server configuration file to include a ROOTCACHE statement that specifies the name of a root cache file containing information used for contacting the root name server(s). This allows the name server to contact the root name server(s) for queries that it cannot resolve itself.
4. If you want to have all unresolved queries (those that this nameserver cannot answer) forwarded directly to another nameserver, then customize the DNS server configuration file to include a FORWARDERS statement that specifies the IP address(es) of the name server(s) to which unresolved queries are to be sent. If the FORWARDERS statement is specified, the VM name server will ignore any information that is specified in a root cache file.

Dynamic Server Operation

The DNS server provides a VM Special Message (SMSG) interface that allows you to perform server administration tasks through a set of privileged commands. For more information see "Dynamic Server Operation" on page 156.

DNS Server Overview

A Domain Name System (DNS) server maps a host name to an internet address or an internet address to a host name. The domain name server (or more simply, the *name server*) is like a telephone book that contains a person's name, address, and telephone number. For each host, the name server can maintain internet addresses, nicknames, mailing information, and available well-known services (for example, DNS, SMTP, FTP, or Telnet). This information is maintained through the use of resource records.

When a client needs to communicate with another host, the client uses either the internet address of the remote host or sends a query to the Domain Name System (DNS) for host name resolution.

However, before the name server can resolve a query, it must be supplied with resource records that either define a zone, or identify a different (remote) name server that can provide the information required for a response. For information about resource record format, see "Resource Records" on page 152.

Primary Versus Caching-Only Name Servers

If a name server maintains local data and has authority for the zone defined by this data, it is called a *primary* name server. If a name server zone transfers a zone from a remote primary name server, it is called a *secondary* name server. Once a secondary name server receives zone data, it has authority for that zone. The secondary name server then periodically refreshes this zone data based on values defined in the zone's authority record.

For more information about primary and secondary zones, see RFC 1034 and 1035.

A name server that does not have authority for any zone is called a *caching-only* name server. To respond to queries, a caching-only name server must communicate with a remote name server in the internet that has access to zone data.

The TCP/IP domain name server can be configured to run as a primary, secondary, or caching-only server. Both primary and secondary name servers store zone data in DB2 tables. To set up a name server that does *not* use DB2, configure the domain name server as a caching-only name server.

The manner in which the name server operates is controlled by statements defined in a DNS configuration file. This file is described in more detail in "DNS Configuration File Statements" on page 138.

Host Name Resolution

When a query is received at the VM name server, it will retrieve the necessary data to process that query from one of three main locations:

- Cache memory
- A DB2 database, or
- Other name servers

If the name server has authority for the query's zone, the name server consults its DB2 database for the query answer. This answer is then sent back to the resolver.

If the name server does not have authority for the query's zone, the name server examines its cache memory to see if it has an answer saved from processing the query previously. If it finds a saved answer, it uses that information to answer the current query. Records remain active in the cache based on the time-to-live (TTL) field for each of the records. If a cache becomes full, the least recently used entry is deleted, if it has been in the cache for the number of seconds specified using the LRUTIME statement. For more information, see "LRUTIME Statement" on page 141.

If the name server does not have the information necessary to respond to the query, it will forward the question to another name server that it believes can provide an answer.

The name server communicates with other name servers to query records outside its zone, to answer queries about zones for which it has authority, and to transfer zones both to and from other name servers. To contact other name servers, the name server must ask questions starting at the root node for the domain name space. The servers who service this root node are referred to as the root name servers. The addresses of these root name servers are made available in the root cache file. If the name server is configured to run as caching-only, the name of this file is supplied in the name server configuration file on the CACHINGONLY statement. If the name server is configured as a primary or secondary name server, the name of this file is supplied in the name configuration file on the ROOTCACHE statement. A sample root cache file is supplied and is named NSMAIN SCACHE. The data in this file is used only to obtain the real addresses of the root name servers and is not used to resolve any of the queries the name server is processing.

It may be desired that all unresolved queries be sent to another name server for resolution. For example, if the VM name server is behind a firewall, it may not be able to contact directly other name servers that it needs in order to process its queries. Instead, it must ask a firewall name server to obtain the answers to its queries. Another example where it makes sense to send all unresolved queries to another name server would be in an organization that has a service provider that is providing the domain name support for the organization. The service provider's name server is known to have good response characteristics and the domain name information that is most often used by the organization. In these and other cases, a FORWARDERS statement can be used to define to the VM name server the addresses of the other name servers to forward all unresolved queries. When a FORWARDERS statement has been specified, any information that is specified in a root cache file is ignored.

Figure 1 on page 136 illustrates the domain name resolution process.

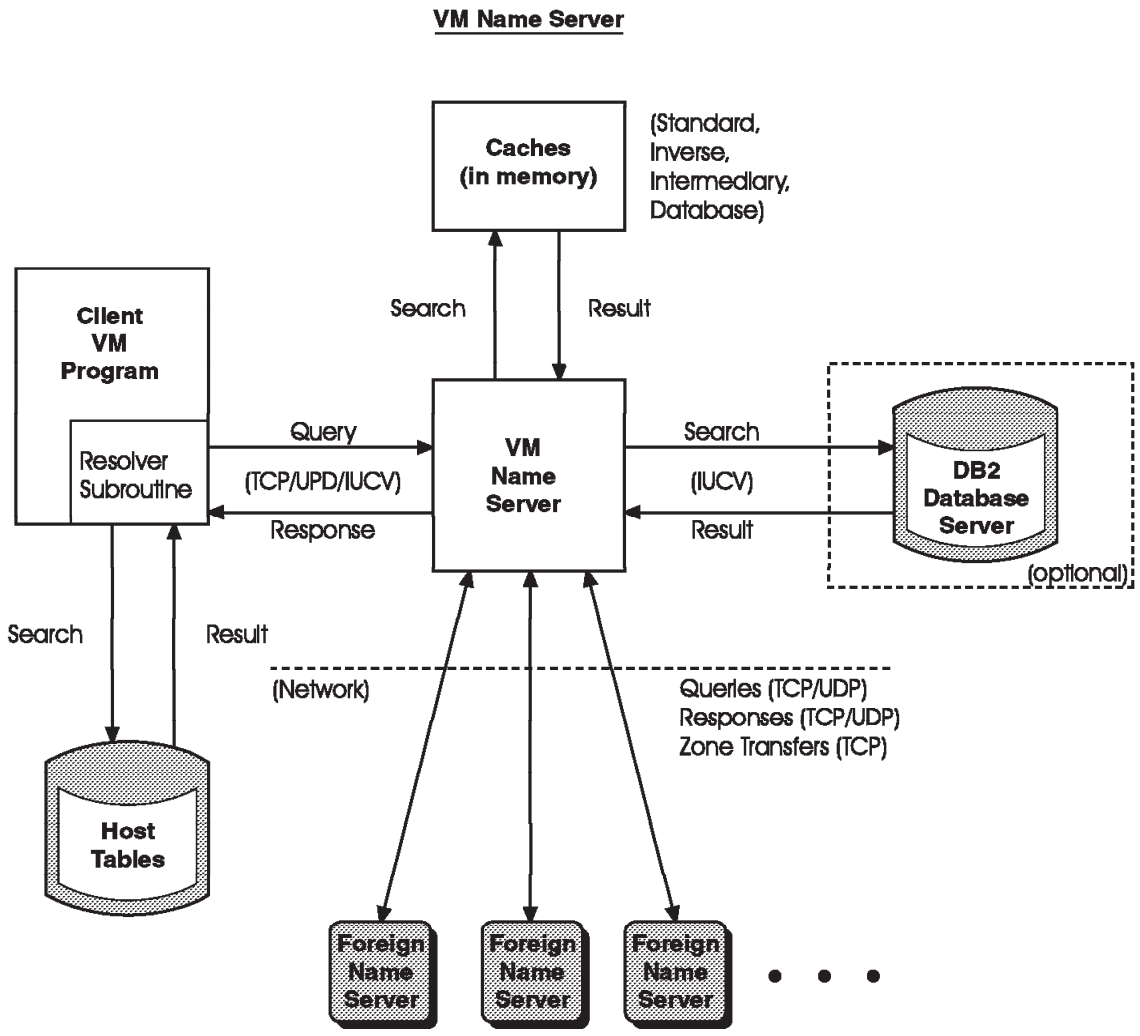


Figure 1. VM Domain Name Resolution

Step 1: Update PROFILE TCPIP

Include the DNS server virtual machine user ID in the AUTOLOG statement of the TCPIP server configuration file. The DNS server is then automatically started when TCPIP is initialized. The IBM default user ID for this server is **NAMESRV**. Verify that the following statement has been added to the PROFILE TCPIP file:

```
AUTOLOG
  NAMESRV  0
```

The name server requires ports TCP 53 and UDP 53 to be reserved for it. Verify that the following statements have been added to your TCPIP server configuration file as well:

```
PORT
  53 TCP NAMESRV     ;   DNS Server
  53 UDP NAMESRV     ;   DNS Server
```

Step 2: Update the DTCPARMS File

When the DNS server is started, the TCP/IP server initialization program searches specific DTCPARMS files for configuration definitions that apply to this server. Tags that affect the name server are:

```
:nick.NAMESRV
    :PARMS.
    :DB2_DATABASE.
```

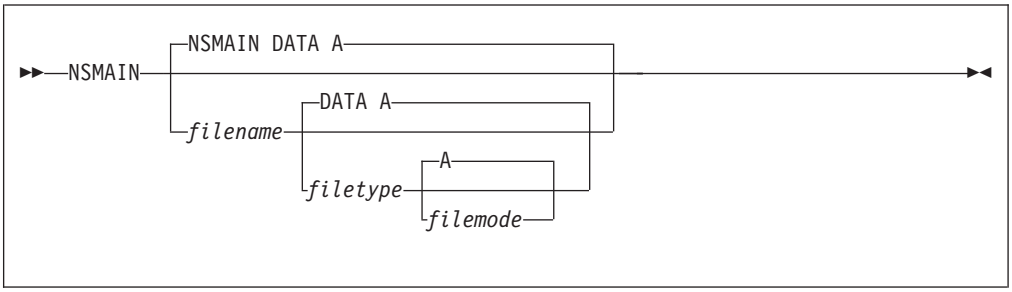
If more customizing is needed than what is available in the DTCPARMS file, a server profile exit can be used.

For more information about the DTCPARMS file, customizing servers, and server profile exits, see Chapter 5, “General TCP/IP Server Configuration”, on page 31.

Note: You should modify the DTCPARMS file for the name server if you:

- Use a configuration file other than NSMAIN DATA.
- Configure a primary name server and need to identify a DB2 database to be referenced.

NSMAIN Command



Specify NSMAIN command operands as **:Parms.** tag start-up parameters in your DTCPARMS file.

Purpose

DNS services are initiated using the NSMAIN command.

Operands

filename
The file name of the DNS server configuration file. The default file name is NSMAIN.

filetype
The file type of the configuration file. The default file type is DATA.

filemode
The file mode of the configuration file. The default file mode is A.

Step 3: Customize the NSMAIN DATA File

The NSMAIN DATA configuration file, NSMAIN DATA, defines how the DNS server is to operate. This file allows you to specify:

- Internet addresses
- Nicknames for fully qualified domain names
- Mailing information

- DB2 tables
- Trace options

See “DNS Configuration File Statements” for detailed information about how to specify entries within this file. A sample DNS configuration file is provided as NSMAIN DATA on the TCPMAINT 591 disk. Your customized DNS configuration file should be copied to the TCPMAINT 198 minidisk and renamed to match what is specified in the NSMAIN command (the default name is NSMAIN DATA).

DNS Configuration File Statements

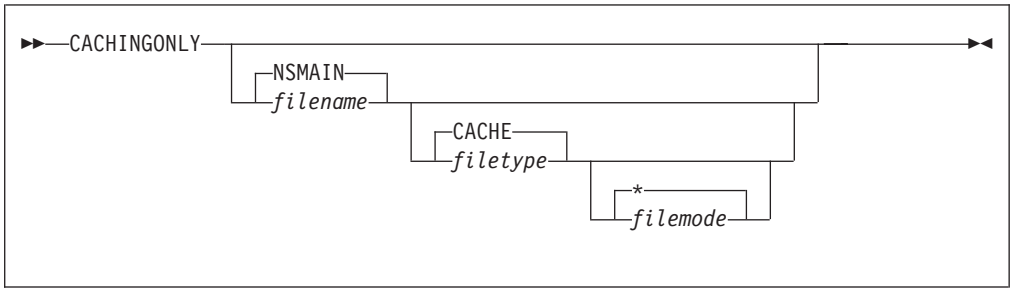
This section describes the statements used to configure the Domain Name Server.

CACHINGONLY Statement

Purpose

The CACHINGONLY statement specifies that the name server is to operate in caching-only mode. The file specified by this statement contains information (type NS and A resource records) necessary to obtain the true addresses of the root name servers.

The data in this file is used only to obtain the real addresses of the root name servers and is not used to resolve any of the queries the name server is processing. The purpose of the file is to provide hints to the name server about where the root name servers are located in the network. Since the network can change dynamically, these addresses are not used for any purpose other than finding the true addresses of the root name servers.



Operands

filename
Specifies the file name of the cache file that points to the root name servers.

filetype
Specifies the file type of the cache file.

filemode
Specifies the file mode of the cache file.

Examples

To use a cache file named **MYCACHE DNSINFO**, specify the following in the DNS server configuration file:

```
CACHINGONLY MYCACHE DNSINFO
```

The content of the MYCACHE DNSINFO file might be:

```
.                608400  IN NS C.NYSER.NET
C.NYSER.NET      608400  IN A  192.33.4.12
```

In this example, the name server will ask the C.NYSER.NET name server (IP address 192.33.4.12) for the addresses of all the root name servers. It will not use that address to answer any other queries. When it has the true addresses of the root name servers, it will commence using them to resolve its queries.

Note: A sample cache file is provided on the TCPMAINT 591 disk as NSMAIN SCACHE. If needed, copy this file to the TCPMAINT 198 minidisk, customize the file, and rename it to match the file name and type specified by the CACHINGONLY statement in the DNS server configuration file (the provided sample, NSMAIN SDATA, specifies NSMAIN CACHE). One possible reason to customize this file is that you are operating on a true intranet that has its own root name space and root name servers. In that case, you should include their names and address in the file in place of the internet root name server data that the sample file contains. You can also customize the file to change the number or selection of the root name servers you initially ask to obtain the true root name server addresses.

Usage Notes

1. You must create the cache file named by the CACHINGONLY statement, and add the information required for the caching-only name server to communicate with the root name server(s). The information in this file is a combination of name server (type NS) resource records and their corresponding (type A) IP address resource records.
2. Record types other than NS and A, such as MX or CNAME, should not be specified in the cache file. If such records are present, they will be ignored.
3. If your data resides in a DB2 table, this statement should not be used. Instead, use the ROOTCACHE statement to provide root name server data when the name server is operating as a PRIMARY or SECONDARY name server.
4. If you have either PRIMARY or SECONDARY statements in the configuration file, as well as a CACHINGONLY statement, the name server will not start.
5. The time-to-live (TTL) field of resource records specified in the cache file is ignored when these records are processed.
6. For more information about resource records, see the *TCP/IP User's Guide* or a suitable Domain Name System reference.
7. When the caching-only name server receives a query, it will look for a response in its cache. If found, that response is returned to the client requestor. If not found, the query will be forwarded, as needed, to a remote name server.

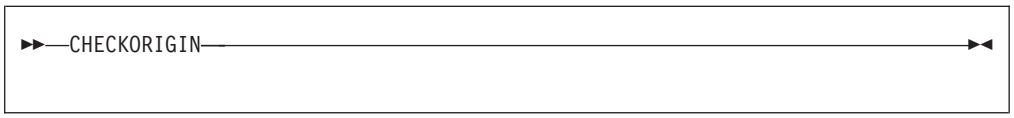
   When a query response arrives at the caching-only name server, it is cached and returned to the client. This response will remain in the cache for as long as specified by the time-to-live (TTL) field in the response.
8. If a FORWARDERS statement is encountered, the NS and A resource records in the cache file are ignored.

CHECKORIGIN Statement

Purpose

Some resolvers incorrectly implement search lists (see note below) causing a duplicate domain origin to be appended to the query name. The CHECKORIGIN statement causes the name server to check the query name for a duplicate of the last label. This option was created for the instances when the Name Server is being bombarded with these queries. The statement is supplied as a single token in the

name server configuration file that is processed by the NSMAIN command (default name: NSMAIN Data). The CHECKORIGIN statement has no parameters. No duplicate checking is performed if CHECKORIGIN does not appear in the file.



Note: A description of search lists can be found in RFC 1123, section 6 1.4.3.

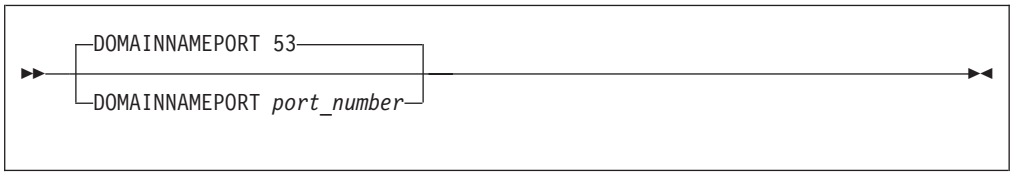
DATABASEQUERYCACHE Statement

The DATABASEQUERYCACHE statement is no longer used. If it is present in the configuration file, it is ignored.

DOMAINNAMEPORT Statement

Purpose

The DOMAINNAMEPORT statement changes the port number that the name server uses. This parameter should be changed only for debugging purposes.



Operands

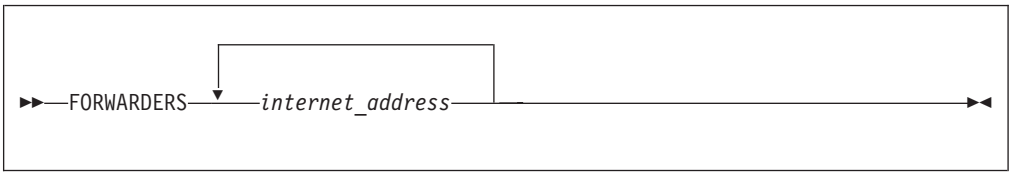
port_number
An integer in the range of 1 through 65 535 that specifies the port number that the name server uses for all TCP and UDP communications. The default port number is 53.

FORWARDERS Statement

Purpose

This statement directs the name server to send any question that it cannot answer to the name server(s) at the IP address(es) specified. These name servers should support recursion. A name server that supports recursion will work on a query until it has a complete answer, and then return that complete answer. Most name servers do support recursion, but some heavily-trafficked name servers, such as the root name servers, have recursion disabled.

Use the FORWARDERS statement to cause all unresolved queries to be sent to another name server for resolution. For example, if the VM name server is behind a firewall, it may not be able to contact directly other name servers that it needs in order to process its queries. Instead, it must ask a firewall name server to obtain the answers to its queries. Another example of when to use the FORWARDERS statement would be that an organization has a service provider that is providing the domain name support for the organization, and the service provider's name server is known to have good response characteristics and the domain name information that is most often used by the organization.



Operands

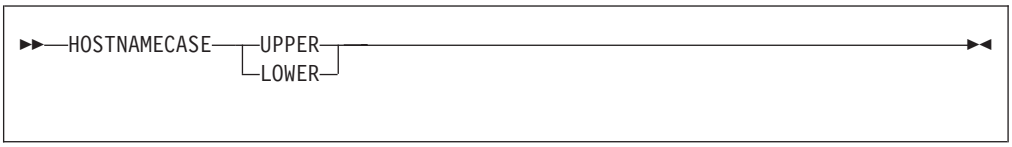
internet_address
Specifies the internet address of a remote name server that should handle questions when this name server doesn't know the answer. This could be a nearby recursive name server or a firewall name server. If multiple name servers can be specified, there will be less chance of failure should one of the name servers not be available.

HOSTNAMECASE Statement

Purpose

The HOSTNAMECASE statement defines the case to which the host names in all queries are translated in order to match the contents of the name server DB2 database. The contents of the DB2 Server for VSE & VM database are assumed to be in the case specified by this statement. If they are not, the name server will not operate correctly.

Note: This statement pertains only to DB2 database queries.



Operands

UPPER
Specifies that the data in the DB2 database is in UPPER case.

LOWER
Specifies that the data in the DB2 database is in LOWER case.

INTERMEDIARYQUERYCACHE Statement

The INTERMEDIARYQUERYCACHE statement is no longer used. If it is present in the configuration file, it is ignored.

INVERSEQUERYCACHE Statement

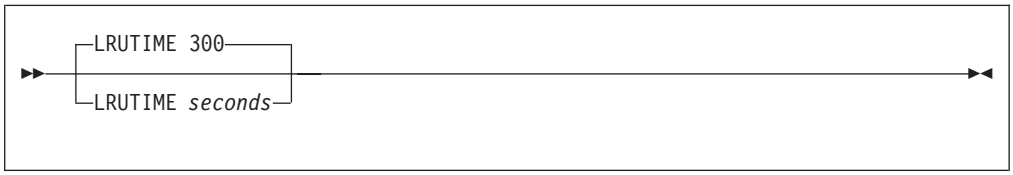
The INVERSEQUERYCACHE statement is no longer used. If it is present in the configuration file, it is ignored.

LRUTIME Statement

Purpose

The LRUTIME statement specifies the amount of time a record must be in the cache before it can be replaced by a new cache entry.

The name server uses a least recently used (LRU) algorithm to replace entries in a full cache.



Operands

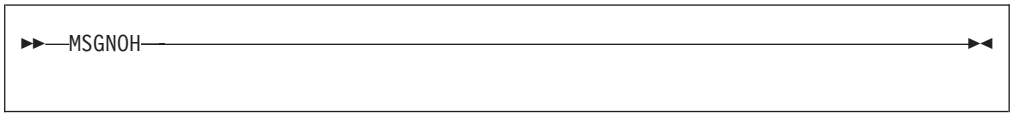
seconds
The number of seconds after which cache entries should be replaced. The default time is 300 seconds (5 minutes).

MSGNOH Statement

Purpose

The MSGNOH statement specifies that the name server use the CP MSGNOH command to reply to an SMSG command. Otherwise, the name server uses the CP MSG command.

Note: The CP MSGNOH command is a privileged command. Ensure that your name server is authorized to issue this command if you are using the MSGNOH statement.



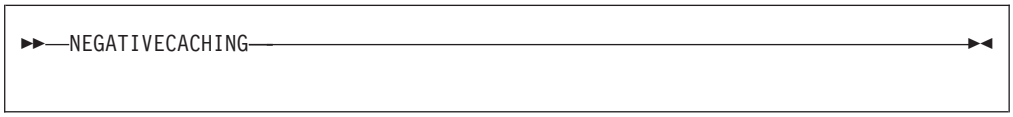
The MSGNOH statement has no operands.

NEGATIVECACHING Statement

Purpose

The NEGATIVECACHING statement specifies that the VM name server caches negative queries.

Negative caching prevents the name server from repeatedly searching for nonexistent resource records. The VM name server caches queries that do not contain answers, but have an SOA resource record in the additional section. The query response remains in the cache for the time specified in the minimum time-to-live (TTL) field of the SOA record. Subsequent queries for the same entry return the negative query response from the cache.



Operands

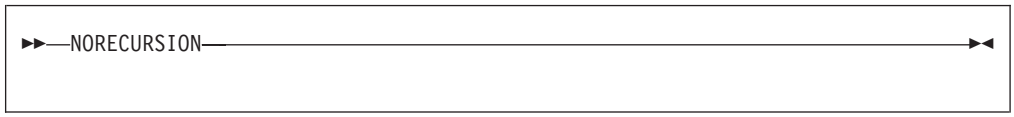
The NEGATIVECACHING statement has no operands.

NORECURSION Statement

Purpose

The NORECURSION statement forces the name server to respond with either the query answer (if available) or with the remote name server host name and the corresponding internet address.

Normally, the name server does recursion by querying remote name servers for information outside the local domain.



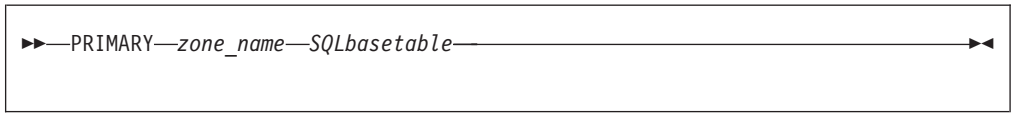
Operands

The NORECURSION statement has no operands.

PRIMARY Statement

Purpose

The PRIMARY statement identifies a local zone for which the name server is responsible, as well as the name of a DB2 table where data about this zone is maintained.



Operands

zone_name
The name of the local zone.

SQLbasetable
The name of the DB2 base table where zone data is maintained.

When declaring PRIMARY statements, you must establish separate SQL tables for a parent domain (for example, ibm.com) and its subdomains (for example, watson.ibm.com). Multiple subdomains can exist in the same table, as long as none of their own subdomains are contained within that table. Violating this rule may result in zone transfer errors.

To define authoritative tables for different local servers, each SQL base table must be unique. You would add lines to the name server initialization file as follows:

```
PRIMARY ibm.com             ibm
PRIMARY watson.ibm.com      watson
PRIMARY endicott.ibm.com    endicott
```

These statements create two different tables for each of the three SQL base tables (ibm0, ibm1, watson0, watson1, endicott0, and endicott1).

To define multiple domains in the same SQL base table, you would add the following lines to the name server initialization file:

```
PRIMARY ibm.com             ibm
PRIMARY watson.ibm.com      otheribm
PRIMARY endicott.ibm.com    otheribm
```

This establishes four tables (ibm0, ibm1, otheribm0, and otheribm1), with the latter two domains both defined in the second pair. The parent domain is placed in a separate SQL base table to prevent zone transfer errors.

Before starting a name server that uses the PRIMARY statement, you must execute the NSTABLE MODULE, with the file name and file type of the DNS server configuration file specified as parameters.

The NSTABLE MODULE creates two tables in the name server's dbspace using the *SQLbasetable* specified in the PRIMARY statement. The name server switches between the two tables. Queries are answered from one table until an SMSG FLIPTABLE command is received. For more information, see "SMSG FLIPTABLE Command" on page 157. At this point, the table name in the remark statement of the system.syscatalog is updated. The name server must be able to process queries from the old zone until the new zone is completely updated.

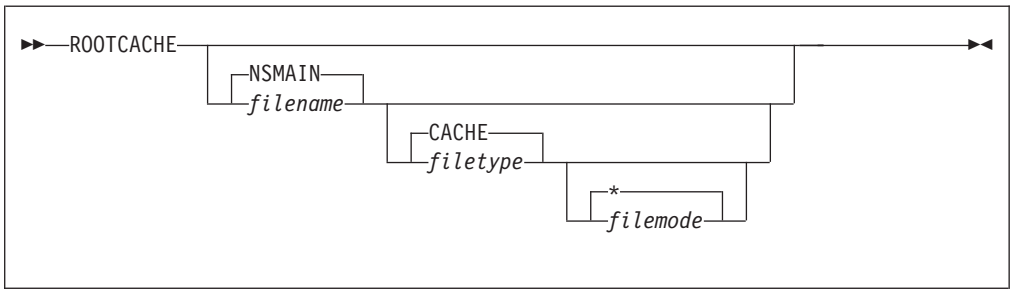
Also, before starting a name server that uses the PRIMARY statement, you must load the tables with data. The recommended way to do this is to use the NSDBLOAD MODULE.

ROOTCACHE Statement

Purpose

The ROOTCACHE statement specifies the name of the file that contains information (type NS and A resource records) necessary to obtain the true addresses of the root name servers. These remote name servers are usually internet or intranet root name servers.

The data in this file is used only to obtain the real addresses of the root name servers and is not used to resolve any of the queries the name server is processing. The purpose of the file is to provide hints to the name server about where the root name servers are located in the network. Since the network can change dynamically, these addresses are not used for any purpose other than finding the true addresses of the root name servers.



Operands

filename
Specifies the file name of the cache file that points to the root name servers.

filetype
Specifies the file type of the cache file.

filemode
Specifies the file mode of the cache file.

Examples

To use a cache file named **MYCACHE DNSINFO**, specify the following in the DNS server configuration file:

```
ROOTCACHE MYCACHE DNSINFO
```

The content of the MYCACHE DNSINFO file might be:

```
.               608400  IN NS C.NYSER.NET
C.NYSER.NET     608400  IN A  192.33.4.12
```

In this example, the name server will ask the C.NYSER.NET name server (IP address 192.33.4.12) for the addresses of all the root name servers. It will not use that address to answer any other queries. When it has the true addresses of the root name servers, it will commence using them to resolve its queries.

Note: A sample cache file is provided on the TCPMAINT 591 disk as NSMAIN SCACHE. If needed, copy this file to the TCPMAINT 198 minidisk, customized the file, and renamed it to match the file name and type specified by the ROOTCACHE statement in the DNS server configuration file (the provided sample, NSMAIN SDATA, specifies NSMAIN CACHE). One possible reason to customize this file is that you are operating on a true intranet that has its own root name space and root name servers. In that case, you should include their names and address in the file in place of the internet root name server data that the sample file contains. You can also customize the file to change the number or selection of the root name servers you initially ask to obtain the true root name server addresses.

Usage Notes

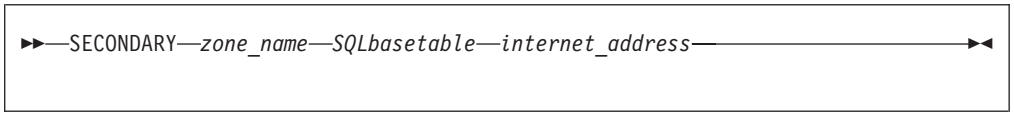
1. You must create the cache file named by the ROOTCACHE statement, and add the information required for the only name server to communicate with the root name server(s). The information in this file is a combination of name server (type NS) resource records and their corresponding IP (type A) address resource records.
2. Record types other than NS and A, such as MX or CNAME, should not be specified in the cache file. If such records are present, they will be ignored.
3. The time-to-live (TTL) field of resource records specified in the cache file is ignored when these records are processed.
4. For more information about resource records, see the *TCP/IP User's Guide* or a suitable Domain Name System reference.
5. When the name server receives a query, it will look for a response in its cache or in its DB2 database. If found, that response is returned to the client requestor. If not found, the query will be forwarded, as needed, to a remote name server.

   When a query response is returned to the name server, it is cached and returned to the client. This response will remain in the cache for as long as specified by the time-to-live (TTL) field in the response.
6. The ROOTCACHE statement is used when you have a PRIMARY or SECONDARY name server.
7. The ROOTCACHE file is ignored if a FORWARDERS statement is encountered.

SECONDARY Statement

Purpose

The SECONDARY statement identifies a zone for which information will be transferred from a remote location, as well as the name of a DB2 table where this zone data is maintained.



Operands

zone_name
The name of the zone to transfer.

SQLbasetable
The name of the DB2 base table where transferred zone data is maintained.

internet_address
The internet address of the remote name server from which zone data is transferred.

For example, to zone transfer the `raleigh.ibm.com` and `phoenix.ibm.com` zones, the following lines are added to the DNS server configuration file:

```
SECONDARY raleigh.ibm.com   raleigh  9.67.43.126
SECONDARY phoenix.ibm.com   phoenix  9.4.1.2
```

Before starting a name server that uses the SECONDARY statement, you must execute NSTABLE MODULE, with the file name and file type of the DNS server configuration file as parameters.

The NSTABLE MODULE creates two tables in the name server's dbspace using the *SQLbasetable* specified in the SECONDARY statement. The name server switches between the two tables. Queries are answered from one table until a new zone is completely received in the other table. At this point, the table name in the remark statement of the system.syscatalog is updated. The name server must be able to process queries from the old zone until the new zone is completely received.

The table specified in a SECONDARY statement is loaded by the name server, via the network, from the name server at the address specified by the *internet_address*.

SMSGUSERFILE Statement

Purpose

The SMSGUSERFILE statement specifies the name of an exec used to verify whether a user ID is authorized for a particular name server SMSG command. The authorization exec is passed the following arguments, in order:

- the node ID of the system from which the SMSG command originated
- the user ID that issued the SMSG command
- the SMSG command

Although the authorization exec has a node parameter, there is currently no support for SMSG command processing from remote nodes. The parameter exists to maintain interface compatibility across releases.

The authorization exec, if used, must return a value of **0** if the user is allowed to perform the SMSG command, and should return a **1** if the user is not to be allowed to perform the command. The HELP, LIST, and STATS commands are commonly accepted as the only SMSG commands that are not privileged. For information about SMSG commands, see “SMSG Interface to the DNS Server” on page 157.

Operands

filename
The file name of the verification exec. The first exec of this name found in the CMS search order is used. The default file name is VALIDUSR.

A sample verification file is shipped as VALIDUSR SEXEC on the TCPMAINT 591 disk. If needed, this file should be copied to the TCPMAINT 198 disk, customized, and renamed to match what is specified in the DNS server configuration file. (The provided NSMAIN SDATA sample specifies VALIDUSR).

Usage Notes

1. If the SMSGUSERFILE statement is omitted, SMSG authorization will still be attempted, using the VALIDUSR default.
2. The authorization exec can be changed while the name server is running. The name server refreshes its access to all file modes before it calls the authorization exec. It will have access to any updates made to the file.
3. If no SMSG authorization exec exists, all SMSG command requests will be rejected. In this case the following message will be displayed on the name server console for each SMSG request:

   ```
   DMSEXT072E Error in EXEC file VALIDUSR, line 0 - not found
   ```

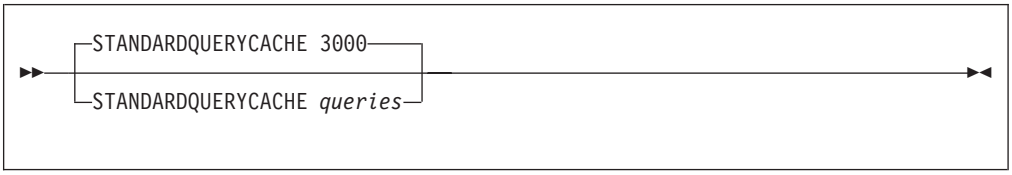
4. Prior to customizing the user verification exit described in this section, ensure that you have reviewed the exit limitations and customization recommendations presented in “Customizing Server-specific Exits” on page 43.

STANDARDQUERYCACHE Statement

Purpose

The STANDARDQUERYCACHE statement allows the specification of the number of standard queries and answers to be managed by the cache.

The query and the resulting answer are stored in a cache. Query answers that are cached do not access the DB2 database and are not sent to remote name servers. This results in faster responses to the client.



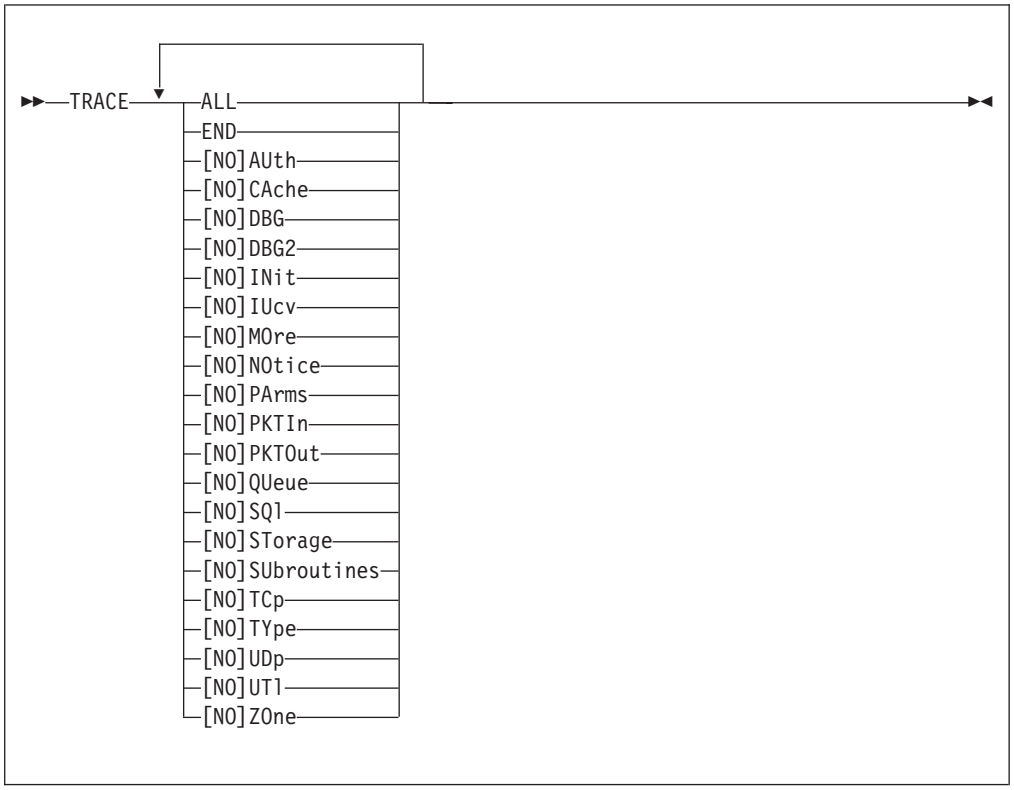
Operands

queries
An integer value that represents the number of standard queries and answers to be managed by the cache. The default is 3000. A cache size of 1000 to 5000 entries is recommended to improve performance. The cache size is dynamically reduced if the server runs low on memory.

TRACE Statement

Purpose

The TRACE statement establishes the set of trace categories for internal name server tracing. Name server trace output is displayed on the name server console. Most of the trace categories are for special diagnostic situations. The **[NO]** form of the trace category turns that specific trace category off. Thus, specifying TRACE ALL NOSUB NOSTOR NOMORE would enable all trace categories except storage and subroutine tracing, and would disable the trace points that require the MORE trace category.



Operands

ALL
All trace categories are enabled. This results in extensive console tracing.

END
All trace categories are disabled.

AUth
NOAUth
Enables or disables console tracing of authority determination.

CAche

NOCAche
Enables or disables console tracing of caching activities.

DBG
NODBG
Enables or disables console tracing in special diagnostic circumstances.

DBG2
NODBG2
Enables or disables console tracing in special diagnostic circumstances.

INit
NOINit
Enables or disables console tracing of initialization activities.

IUcv
NOIUcv
Enables or disables console tracing of IUCV activity.

MOre
NOMOre
Enables or disables additional trace points that are normally skipped because of the large amount of trace data displayed.

NOtice
NONOtice
Enables or disables console tracing of TCP/IP message notifications.

PArms
NOPArms
Enables or disables console tracing of input parameters to most subroutines.

PKTIn
NOPKTIn
Enables or disables console tracing of most inbound DNS messages received.

PKTOut
NOPKTOut
Enables or disables console tracing of most inbound DNS messages sent to other hosts.

QUeue
NOQUeue
Enables or disables console tracing of questions asked of this name server and questions and answers related to those questions.

SQl
NOSQl
Enables or disables console tracing of various activities involving the DB2 database.

STorage
NOSTorage
Enables or disables console tracing of storage usage.

SUbroutines
NOSUbroutines
Enables or disables console tracing of each subroutine executed. Some utility subroutines are instead traced with the UTI trace.

STorage
NOSTorage
Enables or disables console tracing of storage usage.

TCp
NOTCp
Enables or disables console tracing of activities using the TCP protocol.

UDp
NOUDp
Enables or disables console tracing of activities using the UDP protocol.

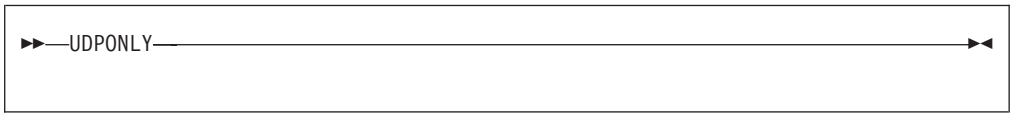
UTI
NOUTI
Enables or disables console tracing of input parameters to and results from many utility subroutines.

ZOne
NOZOne
Enables or disables console tracing of zone transfer requests, and zone transferring of local zones.

UDPONLY Statement

Purpose

The UDPONLY statement instructs the name server to perform recursive name resolution to remote name servers using only the UDP protocol. Without this statement, the name server attempts to communicate with remote name servers using UDP, and if that fails, it tries again using the TCP protocol. The use of UDPONLY is not recommended.



Operands

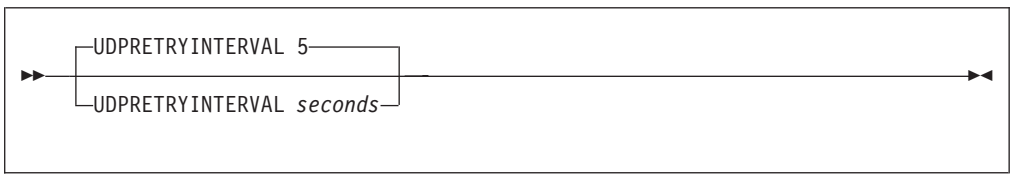
The UDPONLY statement has no operands.

UDPRETRYINTERVAL Statement

Purpose

The UDPRETRYINTERVAL statement instructs the name server to wait a specified amount of time before attempting to communicate with another authoritative name server.

The name server attempts to communicate with remote name servers using UDP. If a remote name server fails to respond, the name server attempts to communicate with another remote name server.



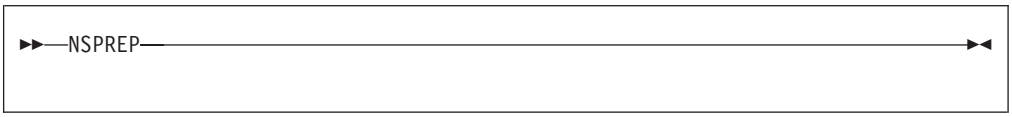
Operands

seconds
The number of seconds to wait before sending the query to the next remote name server. The default is 5 seconds.

Prepare the DB2 Database

Purpose

The NSPREP EXEC has been included to prepare your DB2 database for the new name server. This EXEC allows access to the DB2 database by the name server programs, which have been written using the SQL/DS Version 3 Release 5, or DB2 Server for VM Version 5 Release 1 C language interface.



NSPREP preprocesses the following source files (with file types CSQL or ASMSQL) into your DB2 database.

File	Description
NSACQ	Acquires a database space (dbspace). The default number of pages is 5120.
NSTABLE	Creates two tables for each primary and secondary domain defined in the initialization file.
NSDBLOAD	Creates resource records defining a domain from a master file. The resource records are then inserted into a DB2 table.
NSDBSQL	Contains a subroutine of the NSMAIN MODULE. It uses the assembler DB2 interface. (This file type is ASMSQL.)

NSPREP also pre-processes the NSAXSUB1, NSINSERT, and NSSQL files.

Invoke NSPREP within the DNS server virtual machine.

Define the DB2 Database

Purpose

The name server's DB2 database should be defined with the help of the DB2 database administrator. The DB2 Server for VSE & VM database administrator must acquire a dbspace for the name server by executing the NSACQ MODULE.

Note: DB2 Version 3 Release 5, or DB2 Server for VM Version 5 Release 1 or later, is required if you are using a DB2 database.

The types of SQL commands that the name server generates for each query type are:

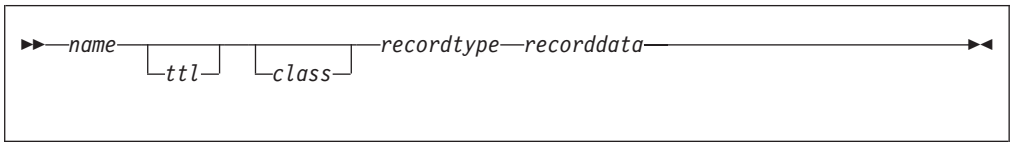
Query Type	Command
Standard	SELECT * FROM *sqltable* WHERE TYPE=? AND CLASS=? - AND NAME=?
Inverse	SELECT NAME, TTL FROM *sqltable* WHERE TYPE=? AND - CLASS=? AND RDATA=?
Database	SELECT RDATA FROM *sqltable* WHERE TYPE=? AND - NAME=?
Database Update	UPDATE *sqltable* SET RDATA=? WHERE TYPE=? AND - NAME=?

Note: Completion queries are no longer supported.

Resource Records

Purpose

After setting up the database as described in “Define the DB2 Database” on page 151, you must load the data into the SQL table. These data entries are called resource records. NSDBLOAD MODULE can assist you in inserting the data into the table. The following is the format of a resource record that is used as input for the NSDBLOAD MODULE.



Operands

name
The location in the dbspace for the resource record, such as the owner. If a resource record line starts with a blank, it is loaded into the location specified by the most recent location specifier. The location specifier is relative to an origin provided on the $ORIGIN record.

ttl The time-to-live (TTL) field, which is optional. This field is expressed as a decimal number. If it is omitted, the *ttl* default value is specified in the Start of Authority (SOA) record. TTL specifies, in seconds, how long the data remains in the cache.

class
The class of the data.

recordtype
The type of the data. This field describes the kind of data that exists in the next field, *recorddata*.

The following record types are recognized:

A	Address
NS	Name Server
CNAME	Canonical Name (nickname, alias)
SOA	Start of Authority
WKS	Well Known Services
PRT	Domain Name Pointer
HINFO	Host Information
MINFO	Mailbox Information
MX	Mail Exchanger
TYPE97	Database Update Authority
TXT	Text String

recorddata
Depends on the values of the resource record class and type. The fields that make up the record data, are usually expressed as decimal numbers or as domain names.

Create a MASTER DATA File

Purpose

MASTER DATA files are text files that contain resource records in text form. For more information about resource records, see “Resource Records”. MASTER DATA files, stored on TCPMAINT 198, are used to define a zone because the contents of

a zone can be expressed as a list of resource records. MASTER DATA files can also be used to list the contents of a cache.

The format of these files is a sequence of entries. Entries are line-oriented, but you can use parentheses to continue a list across a line boundary, and text literals can contain Carriage Return Line Feeds (CRLF) within the text. You can use any combination of tabs and spaces as a delimiter between the items that make up an entry. You can end any line with a comment. The comment starts with a semicolon (;). You can use blank lines, with or without comments, anywhere in the file.

Two control entries are defined:

Entry	Definition
$ORIGIN	Followed by a domain name, and resets the current origin for relative domain names to the stated name.
$INCLUDE	Inserts the named file into the current file, and can specify a domain name that sets the relative domain name origin for the included file. The use of $INCLUDE is optional.

For a complete description of master files, see RFC 1034 and RFC 1035.

Some characters have special meanings:

Character	Description
@	A free-standing @ denotes the current origin.
•	One free-standing dot represents the null domain name of the root.
\X	*X* is any character other than a digit. Use \X to quote this character so that its special meaning does not apply, as in a mail box specification, or an SOA record. For example, you can use a backslash followed by a dot (\.) to place a dot character in a label.
( )	Parentheses group the data that crosses a line. Line terminations are not recognized within parentheses.
;	A semicolon begins a comment. The remainder of the line is ignored.

Each zone must contain an SOA and NS resource record. The following is an example of a master domain file:

```
;The following $ORIGIN record will be (IBM.COM.) appended to any
;name that does not end in a dot until the next $ORIGIN record.
;***************************************************************************
;             Authoritative for Domain IBM.COM
;***************************************************************************
$origin IBM.COM.         ; THE ORIGIN FOR ALL NAMES
ibm.com.  IN  SOA  yktvmx.watson.ibm.com.  zohar.yktvmx.watson.ibm.com.  (
                        091690    ;serial number for data
                        21600     ;refresh value for secondary NS (in secs)
                        3600      ;retry value for secondary NS (in secs)
                        360000    ;expire data when refresh not available (secs)
                        86400 )   ;minimum time to live value (secs)
ibm.com.  IN   NS   yktvmx.watson.ibm.com.
ibm.com.  IN   NS   aides.watson.ibm.com.
ibm.com.  IN   NS   vmn.almaden.ibm.com.
$include ibm data a              ;File contain hosts addresses
```

The following is an example of an IBM data file:

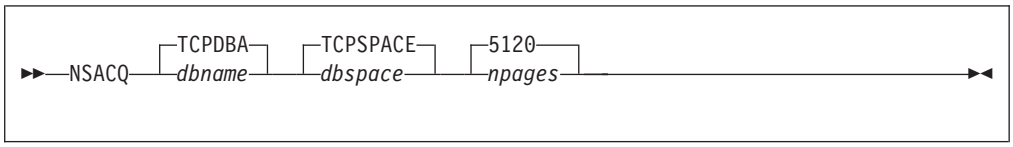
```
raleigh.ibm.com.          IN    NS  ralvmm.raleigh.ibm.com.
ralvmm.raleigh.ibm.com.   IN    A   9.67.43.126
;
watson.ibm.com.           IN    NS  yktvmx.watson.ibm.com.
yktvmx.watson.ibm.com.    IN    A   129.34.128.246
```

The DNS defines a special domain called in-addr.arpa to translate internet addresses to domain names. An in-addr.arpa name is composed of the reverse octet order of an IP address concatenated with the in-addr.arpa string.

Install the Name Server Database

Purpose

At this point, your DB2 database virtual machine has been created. To acquire a dbspace in the database, log on to the name server virtual machine, and execute the NSACQ MODULE. The NSACQ MODULE issues the SQL ACQUIRE DBSPACE command. The name server must have the authority granted to execute the NSACQ command. The authority can be acquired by granting the name server Database Administrator (DBA) authority.



Operands

dbname
The name of the DB2 database set up for the name server by the database administrator. The default is TCPDBA.

dbspace
The name of the database space set up for the name server. The default is TCPSPACE.

npages
The size of the database machine. For example, you can choose 128, 256, 512, 1024, 2048, 5120, or 12 800. The default is 5120.

Insert Data in the Database

Purpose

You must insert the resource records that define a zone into separate DB2 tables. To update a zone without shutting down the name server, you need two tables for each zone. Each of these tables contains the following fields:

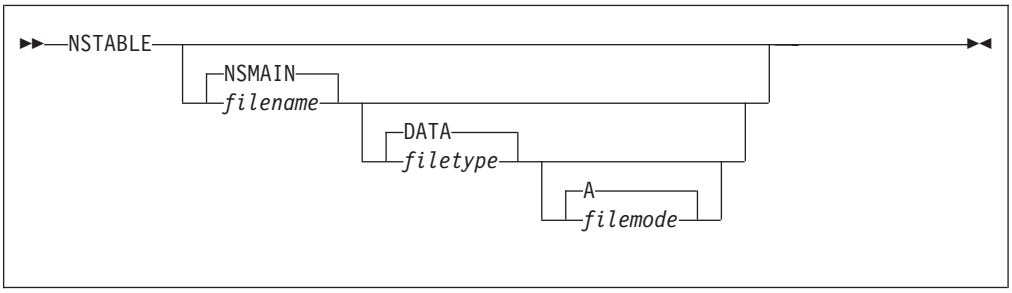
```
NAME      varchar (150)
CLASS     smallint
TYPE      smallint
TTL       integer
RDATA     varchar (250)
```

For each primary zone, add a line to the name server configuration file containing the zone origin and base name (17 characters maximum) of the DB2 table. For each secondary zone, add a line to the name server configuration file containing the zone origin, the base name of the DB2 table to use, and the internet address of the remote name server. For example:

```
primary     watson.ibm.com.          watson
secondary   raleigh.ibm.com.         raleigh   9.67.43.100
```

After defining the primary and secondary zones in the configuration file, executing NSTABLE creates two DB2 tables by appending a `0` and `1` to the base name for each primary and secondary zone defined. The following tables are defined for the previous example:

```
watson0 watson1 raleigh0 raleigh1
```



Operands

filename
The file name of the DNS server configuration file that includes the PRIMARY and SECONDARY statements which identify the DB2 base table names that NSTABLE should construct. The default file name is NSMAIN.

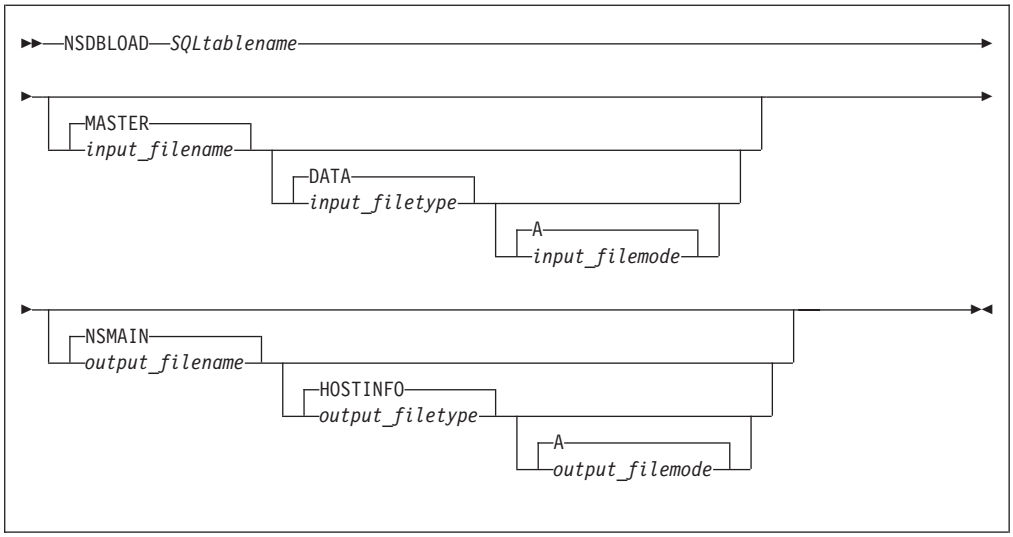
filetype
The file type of the name server configuration file. The default file type is DATA.

filemode
The file mode of the name server configuration file. The default file mode is A.

When your database table is constructed, use the NSDBLOAD MODULE to load your database. NSDBLOAD reads the master data file you specify and creates a CMS file. The CMS file is read and the entries are loaded into the database.

After the name server is configured and running, you can use NSDBLOAD from another user to update the database. This user should have DB2 authority and be able to access the name server tables.

Once the tables are loaded from this user, an SMSG FLIPTABLE command can be issued to make the name server switch to the updated table. This eliminates the need to shutdown the name server for updates.



Operands

SQLtablename
The name of the DB2 table. If you use the base name for the table, NSDBLOAD will attempt to determine which table the name server is currently using and update the other table. If it cannot determine which table is in use, it will update the table that ends with a zero (`0`)— (for example, `watson0` or `raleigh0`). You can use the full table name (for example, `watson0`, `raleigh1`) and that table will be updated.

input_filename
The file name of the master data file written in resource record format. The default file name is MASTER.

input_filetype
The file type of the master data file. The default file type is DATA.

input_filemode
The file mode of the master data file. The default file mode is A.

output_filename
The file name of the master data file. The default file name is NSMAIN.

output_filetype
The file type of the master data file. The default file type is HOSTINFO.

output_filemode
The file mode of the master data file. The default file mode is A.

Dynamic Server Operation

The VM Special Message Facility (SMSG) command provides an interactive interface to the DNS server to:

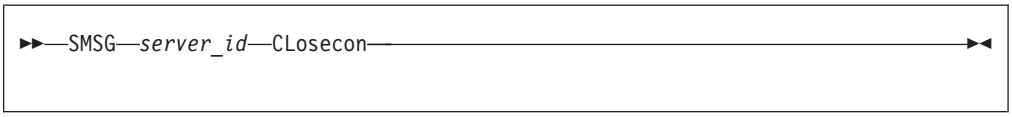
- obtain information about the name server
- diagnose name server problems using trace facilities
- purge cache entries
- change DB2 for VM tables
- close the name server console log.

See the "SMSGUSERFILE Statement" on page 146 for information on controlling authorization to issue SMSG commands to the name server.

SMSG Interface to the DNS Server

The various SMSG commands and operands supported by the name server are presented throughout the remainder of this section.

SMSG CLOSECON Command



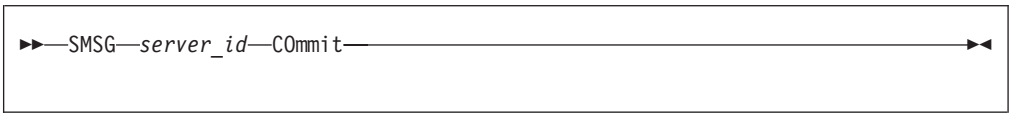
Purpose

The SMSG CLOSECON command causes the name server virtual machine to issue the command `CP SPOOL CONSOLE CLOSE`. The name server console is spooled to the owner identified in the DTCPARMS file.

Operands

server_id
The user ID of the name server virtual machine.

SMSG COMMIT Command



Purpose

The SMSG COMMIT command forces a commit of all DB2 transactions and releases the link to SQL.

Operands

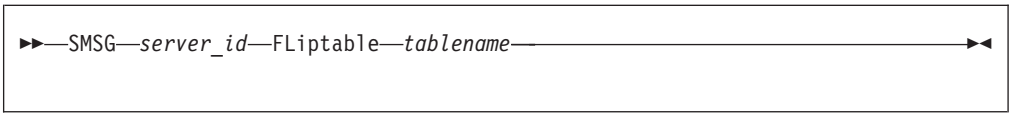
server_id
The user ID of the name server virtual machine.

SMSG DUMP Command

Purpose

The SMSG DUMP command has been replaced by the SMSG STORAGE command. The SMSG DUMP command may be removed in a future release. It will accept the parameters of the SMSG STORAGE command and behave like the SMSG STORAGE command. This behavior is different than the behavior of the old SMSG DUMP command.

SMSG FLIPTABLE Command

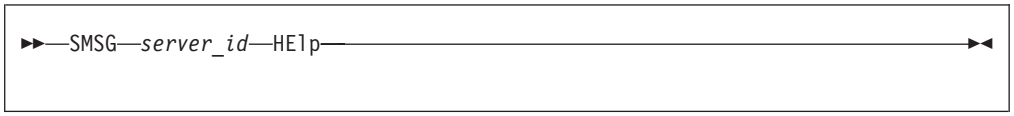


Purpose
The SMSG FLIPTABLE command informs the name server that new zone data appears in a DB2 table. Two DB2 tables are defined for each primary zone. NSDBLOAD can change the zone data in the DB2 table that the name server is not using. After the new data is in the DB2 table, you can issue an SMSG FLIPTABLE command to force the name server to use the updated DB2 table.

Operands
server_id
The user ID of the name server virtual machine.

tablename
The name of the table to flip. This is the base table name, and *not* the DB2 table name for example, (`watson`, not `watson0` or `watson1`).

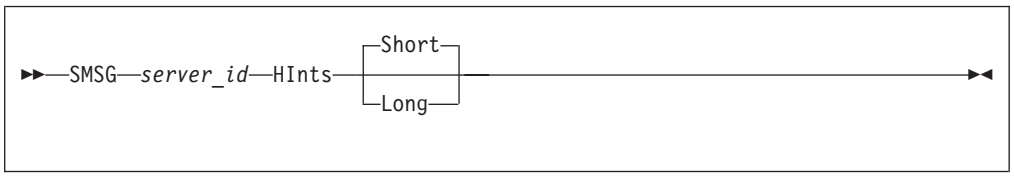
SMSG HELP Command



Purpose
The SMSG HELP command lists the SMSG commands supported by the name server.

Operands
server_id
The user ID of the name server virtual machine.

SMSG HINTS Command



Purpose
The SMSG HINTS command allows you to display the addresses of the name servers that this name server will use when it does not know the answer to a question. The data comes from the FORWARDERS statement, if one is present in the configuration file. If a FORWARDERS statement is not present, the data comes from the contents of the file specified in the CACHINGONLY or ROOTCACHE statement.

Operands
server_id
The user ID of the name server virtual machine.

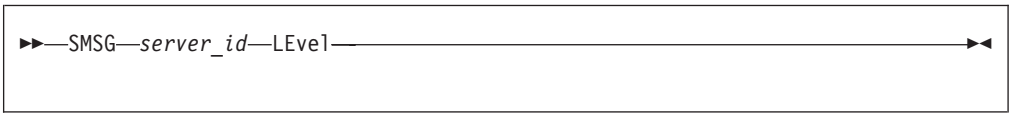
Short
Indicates that only the IP addresses of the remote name servers are to be displayed.

Long
Indicates the IP addresses and the names of the remote name servers are to be displayed.

Usage Notes

- If a FORWARDERS statement from the configuration file is in effect, all parameters on the SMSG HINTS command are ignored and only the IP addresses of the remote name servers will be displayed.

SMSG LEVEL Command



Purpose

The SMSG LEVEL command allows you to display the service level of the name server and its component parts. The overall service level is provided in the response. The component part service information is directed to the name server console, and it can be obtained using the SMSG CLOSECON command. Output from a LISTFILE NSMAIN * * command is also included. This can be used to obtain the date and time of the NSMAIN MODULE in use and to ensure you are executing the intend level. If default configuration file names (NSMAIN DATA, NSMAIN CACHE) are in use, these will also be displayed by the LISTFILE command.

Operands

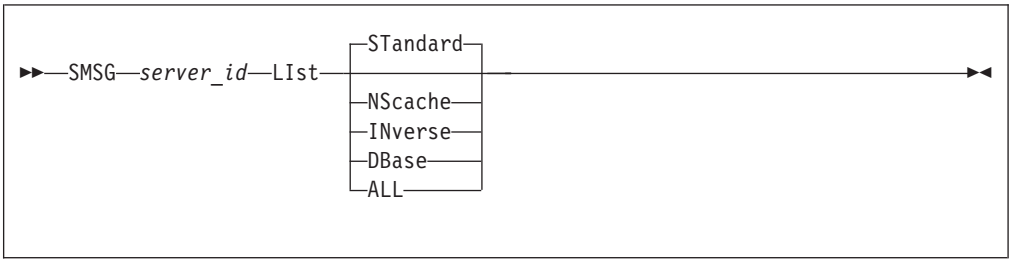
server_id
The user ID of the name server virtual machine.

For example, if you enter `SMSG NAMESRV LEVEL`, the following information is displayed.

```
SMSG NAMESRV LEVEL

VM TCP/IP Name Server Level nnn,
service level PQ12345
```

SMSG LIST Command



Purpose

The SMSG LIST command allows you to display the contents of the current name server caches.

Operands

server_id
The user ID of the name server virtual machine.

NScache
This cache is no longer used. This operand is now ignored, but is accepted to maintain compatibility with prior levels of TCP/IP for VM.

STandard
Lists the contents of the standard cache. This is the default. This operand is now ignored, but is accepted to maintain compatibility with prior levels of TCP/IP for VM.

INverse
This cache is no longer used. This operand is now ignored, but is accepted to maintain compatibility with prior levels of TCP/IP for VM.

DBase
This cache is no longer used. This operand is now ignored, but is accepted to maintain compatibility with prior levels of TCP/IP for VM.

ALL
Lists the contents of all caches. Because the standard query cache is the only cache used, the parameter is equivalent to specifying STANDARD.

For example, if you enter `SMSG NAMESRV LIST ALL`, the following information is displayed:

```
-----------------------------------------------------
Type Class Used Orig.TTL  Rem.TTL  Entry
Using standard query cache
A    IN       0   136347   130364  image.eimg.com
A    IN       0    86400    86235 -open
A    IN       0    86400    85244 -LOBO
A    IN       0    43200    42089 -ROGMO2@ENDICOTT.IBM.COM
PTR  IN       3    85802    57746  3.9.244.9.in-addr.arpa
PTR  IN       6    58298    56641  3.240.139.9.in-addr.arpa
A    IN       0    13056     6760  cnbcads.cnbc.com
A    IN       0    53561    43990  c.realtor.com
A    IN       6  9999999  9756054  dukhat.torolab.ibm.com
A    IN       1    21600    13120 -DS.INTERNIC.NET.ENDICOTT.IBM.COM
```

The following is a description of the fields that appear in the LIST command response:

Type
The resource record type.

Class
The resource record class.

Used
The number of times the record was returned from the cache.

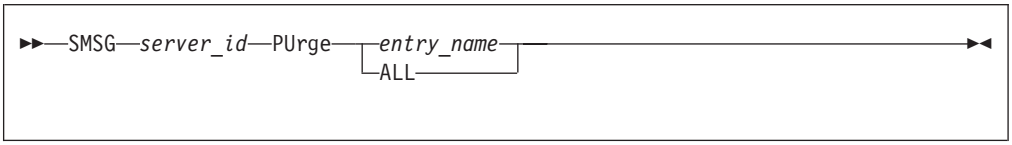
Orig.TTL
The original value of the time-to-live field.

Rem.TTL
The remaining time that the record remains in the queue

Entry
The resource record name. Entries that are preceded by a negative sign (-) are

cached negative queries. Negative entries indicate that the entity does not exist. If a query comes in for that entity, we will not forward the question to the name server for that domain, but will respond indicating that the entity does not exist. Negative entries are generated only when the NEGATIVECACHING statement has been specified in the configuration file.

SMSG PURGE Command



Purpose

The SMSG PURGE command purges one or all entries from all maintained caches.

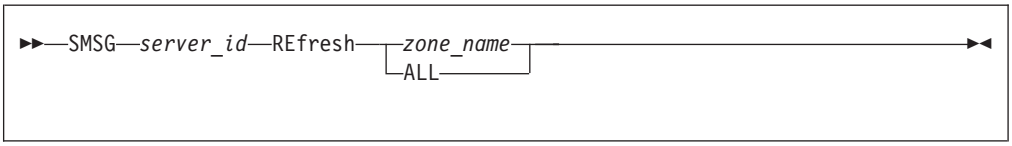
Operands

server_id
The user ID of the name server virtual machine.

entry_name
A specific cache entry for which all occurrences should be purged from all caches.

ALL
Purges the contents of all caches.

SMSG REFRESH Command



Purpose

The SMSG REFRESH command instructs the name server to refresh immediately the zone data contained in a DB2 table. This is used for a table for which the name server is acting as a secondary name server. It will cause the name server to attempt to contact the name server at the IP address specified on its associated SECONDARY configuration statement and reload the table with the zone data from that name server. It is useful when it is discovered that the primary name server has been updated and the data at our name server is obsolete.

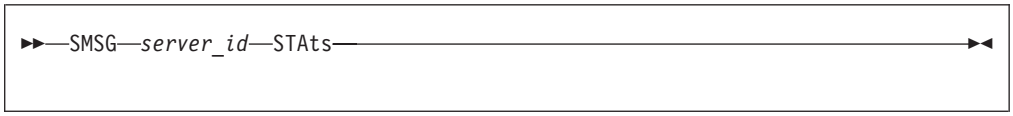
Operands

server_id
The user ID of the name server virtual machine.

zone_name
The name of the zone (`ibm.com`, to take an example from the sample configuration file) that should be refreshed. The *zone_name* must match the *zone_name* from a SECONDARY statement to have any effect. If the *zone_name* is not valid, no error message will be issued, but the command will not have any effect.

ALL
Causes the name server to refresh all zones for which it is acting as a secondary name server.

SMSG STATS Command



Purpose
The SMSG STATS command returns useful information about the name server, such as start time, number of queries received, and size of cache.

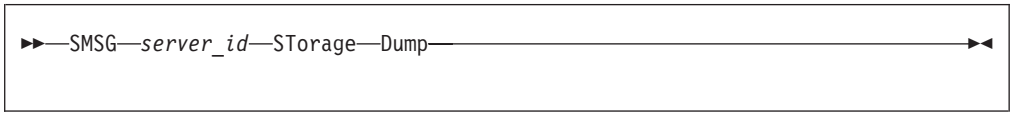
Operands
server_id
The user ID of the name server virtual machine.

For example, if you enter `SMSG NAMESRV STATS`, the following information is displayed.

```
SMSG NAMESRV STATS

NS start time:            Mon Oct  1 06:30:16
--------------------------------------------------
Total number of queries:  1632
Answers from cache:       435    (27%)
Size of cache:            150  used: 6
```

SMSG STORAGE Command

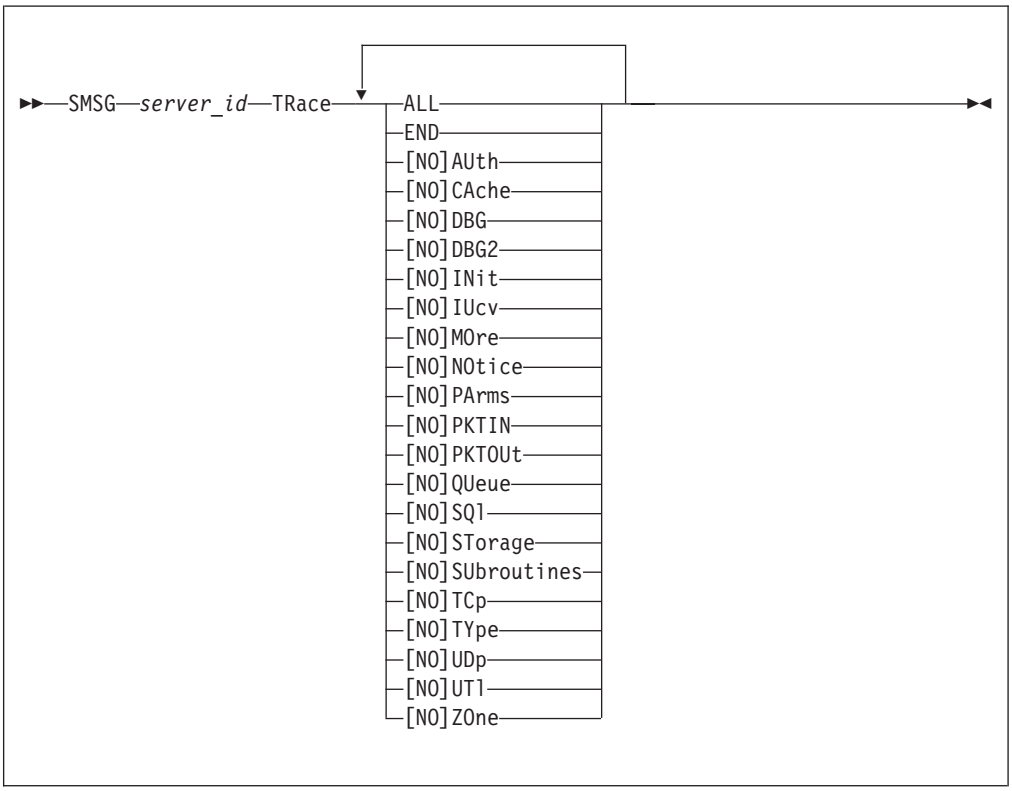


Purpose
The SMSG STORAGE command dumps the internal storage management chain to the name server console. The data can then be recovered using the SMSG CLOSECON command.

Operands
server_id
The user ID of the name server virtual machine.

Dump
Causes the internal storage management chain to be displayed on the name server console. If the Dump keyword is missing, no action is taken.

SMSG TRACE Command



Purpose

The SMSG TRACE command starts or stops server tracing. The **[NO]** form of a trace category turns that specific trace category off.

Tracing is used primarily for problem diagnosis. The output from QUEUE tracing can also be useful in monitoring what hosts are using your name server and what other hosts they are attempting to access.

Operands

server_id
The user ID of the name server virtual machine.

ALL
All trace categories are enabled. This results in extensive console tracing.

END
All trace categories are disabled.

AUth
NOAUth
Enables or disables console tracing of authority determination.

CAche
NOCAche
Enables or disables console tracing of caching activities.

DBG
NODBG
Enables or disables console tracing in special diagnostic circumstances.

DBG2
NODBG2
Enables or disables console tracing in special diagnostic circumstances.

INit
NOINit
Enables or disables console tracing of initialization activities.

IUcv
NOIUcv
Enables or disables console tracing of IUCV activity.

MOre
NOMOre
Enables or disables additional trace points that are normally skipped because of the large amount of trace data displayed.

NOtice
NONOtice
Enables or disables console tracing of TCP/IP message notifications.

PArms
NOPArms
Enables or disables console tracing of input parameters to most subroutines.

PKTIn
NOPKTIn
Enables or disables console tracing of most inbound DNS messages received.

PKTOut
NOPKTOut
Enables or disables console tracing of most inbound DNS messages sent to other hosts.

QUeue
NOQUeue
Enables or disables console tracing of questions asked of this name server and questions and answers related to those questions.

SQl
NOSQl
Enables or disables console tracing of various activities involving the DB2 database.

STorage
NOSTorage
Enables or disables console tracing of storage usage.

SUbroutines
NOSUbroutines
Enables or disables console tracing of each subroutine executed. Some utility subroutines are instead traced with the UTI trace.

STorage
NOSTorage
Enables or disables console tracing of storage usage.

TCp
NOTCp
Enables or disables console tracing of activities using the TCP protocol.

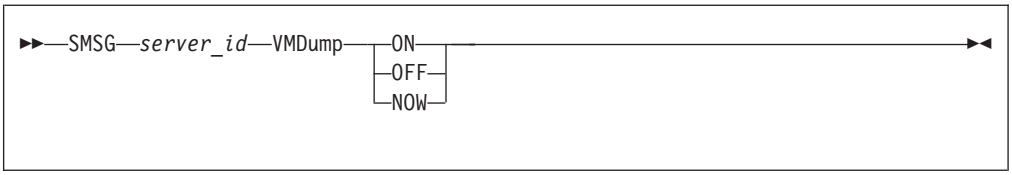
UDp

NOUDp
Enables or disables console tracing of activities using the UDP protocol.

UTI
NOUTI
Enables or disables console tracing of input parameters to and results from many utility subroutines.

ZOne
NOZOne
Enables or disables console tracing of zone transfer requests, and zone transferring of local zones.

SMSG VMDUMP Command



Purpose
The SMSG VMDUMP command allows you to cause the name server to take a CMS DUMP if an abend occurs. It also allows you to take a DUMP immediately.

Operands
server_id
The user ID of the name server virtual machine.

ON
Specifies that the name server should take a dump if the abend handler is entered.

OFF
Specifies that the name server should NOT take a dump if the abend handler is entered.

NOW
Specifies that the name server should take a dump right now. Execution continues after the dump is taken.

Rebuilding the Name Server Modules
To compile the name server ASMSQL and CSQL files, you must have a Database Administrator issue the following command:

```
grant connect to server_id identified by sqldbapw
```

where *server_id* is the user ID of the name server virtual machine.

If you do not want to use the password SQLDBAPW, you must issue the GRANT command with a password of your choice and modify the TCPOBJCT EXEC to reflect the new password.