How to contact Borland

Borland offers a variety of services to answer your questions about InterBase:

Customer support	In the United States and Canada, registered customers can call the InterBase customer support line at 1-800-437-7367 with their questions about InterBase. International customers can call 408-431-5400. Also, general classes are available for InterBase training and consultants are available for individual customer site needs.
Marketing information	If you are not already an InterBase customer and want information about this product, call the product information line at 1-800-245-7376.
Documentation feedback	We welcome your feedback about errors or missing topics in the InterBase documentation.

Table of Contents

Installation Overview

This document describes how to install, register, and license InterBase on the following UNIX systems:

- DECstations and DECservers
- DG-AViiON
- HP 9000 Series 300, Series 400, and Series 600 workstations
- HP 9000 Series 700 Precision Architecture machines
- HP 9000 Series 800 Precision Architecture machines
- IBM RS/ 6000
- Motorola Delta and Motorola IMP
- SCO Unix
- Silicon Graphics

It also describes the InterBase servers, the *lock_ header* file, platform-specific language use, and new InterBase features that are specific to UNIX.

The installation process consists of the following steps:

1. Backing up any databases created using a previous version of InterBase.
2. Installing the software.
3. Registering and licensing your installation with Borland.
4. Restoring backed-up databases.
5. Making system-specific adjustments, if necessary.
6. Modifying the *lock-header* file and starting a server, if desired.

Contents of the InterBase Software Package

In addition to this installation document, your InterBase software package includes:

- Installation media, distributed in **tar** format. SCO installation diskettes are shipped in compressed **tar** format.
- Version 3.3 Release Notes

If you have not previously received a Version 3.*n* shipment, your software package also includes:

- InterBase 3.0 to 3.2 Release Notes
- The InterBase documentation set, consisting of the following books:
  - — *Getting Started with* InterBase
  - — *Data Definition Guide*
  - — *DDL Reference*
  - — *Programmer's Guide*
  - — *Programmer's Reference*
  - — *DSQL Programmer's Guide*
  - — *Qli Guide*
  - — *Qli Reference*
  - — *Database Operations*
  - — *Forms Guide*
  - — *Sample Programs*
  - — *Master Index*

If you are missing any of these items from your software package, contact InterBase Customer Support at Borland International.

System Requirements

The following table lists the minimum system requirements for installing and running InterBase. All systems support the TCP/IP network protocol. DECstations and DECservers also support DECnet.

Table 1: System Requirements

System	OS	Approx Free KB	Other
DECstation, DECserver	Ultrix 4.2	12,000	DEC NFS File Locking patch installed [1]
DG-AViiON	DG-UX 5.4	19,500	
HP 9000/300, 400, 600	HP-UX 8	12,000	
HP 9000/700	HP-UX 8.0.5	8,000	
HP 9000/800	HP-UX 8.0.5	16,000	
IBM RS 6000	AIX 3.2	6,500	
386, 486 PCs	SCO UNIX 3.2.4	6,900	
Motorola Delta	Sys V/68 Rel 3	7,600	
Motorola IMP	UNIPLUS+ 3.1	7,500	
Silicon Graphics	IRIX 3.3.1 , 4.0.1	23,500	

[1] The DEC File Locking patch, available from Digital Equipment Corporation, is only necessary if you plan to run InterBase on a disked client where databases reside on the client's disk. The File Locking patch must be installed before you install InterBase.

Before You Install

This section describes the actions you should take *before* installing InterBase on your system. InterBase requires certain minimum UNIX options and configurations, described below in *Preparing Your System*. In addition, if you are upgrading to InterBase version 3.3 from an earlier version, you should save some existing InterBase files before installing the new version.

Preparing Your System

Before you install InterBase, you should verify that the following options exist and are properly configured on your system:

- Kernel facility requirements:
  - — IPCSHMEM, the shared memory facility
  - — IPCSEMAPHORE, the semaphore facility

  To add these options to the kernel, edit the kernel configuration file and rebuild the kernel. See your UNIX system administrator for information.
- Free semaphore requirements:

Table 2: Required Semaphores

Operating System	Free Semaphores Required
SCO UNIX 3.2.4	10
All others	49

To determine the total number of semaphores available in your kernel consult your system administrator.

To determine the number of semaphores currently in use, use the following command:

```
% ipcs -a
```

The total number of semaphores in use is the total of the NSEMS column. InterBase uses three sets of semaphore clusters:

Table 3: Semaphore Clusters

Platform	Clusters
DG AViiON	25, 16, 1
SCO UNIX	10
All others	32, 16, 1

If you do not see these sets of semaphores listed in NSEMS or you have other semaphores installed, you may not have enough semaphores to run InterBase. You can drop some of the existing semaphores using:

```
% ipcrm -s <semaphore ID from ipcs display>
```

You can also edit the kernel configuration file to specify more semaphores if the maximum number of semaphores for your system has not already been reached, but then you must rebuild the kernel before installing InterBase.

- Kilobyte requirements:

  Verify that you have the required number of kilobytes on the partition where you want to install the InterBase directory tree. See *System Requirements* for the required kilobytes.

Caution

The InterBase directory tree should be located on a disk local to the server on which you are installing InterBase. If the directory is located on a disk of a remote machine, you may not be able to use the InterBase help and example databases.

- Save previous InterBase version files:

  If you have a previous InterBase version installed, you should save certain files before installing a new version.

  — */usr/interbase/isc_license.dat* (*isc_lic.dat* on SCO)

  — */usr/interbase/lock_header* (if you modified the file)

  — */usr/interbase/bin/pictor**
    */usr/interbase/pictor_help.**
    */usr/interbase/help/pictor_local_help.** (if you installed Pictor)

- */tmp* directory recommendation:

  InterBase places files in the */tmp* directory during data processing (e.g., queries, building indexes, sorting data). If you plan on issuing queries that retrieve a large number of records or if you will be using indexes or sorting on large relations, you may want to increase the amount of temporary space available to InterBase. You can change the temporary working location from */tmp* to another directory by setting the environment variable, TMP, to point to the desired directory.

- On HP platforms running HP-UX, **unmask** must be set to zero on a machine before proceeding with InterBase installation:

```
unmask 0
```

Installing InterBase on UNIX

This section provides numbered steps for installing InterBase on your system. If you are installing InterBase under SCO, refer to *SCO Installation*, below. On all other systems, refer to *UNIX Installation*.

UNIX Installation

1. Log in as superuser.

   ```
   % su
   ```

2. Insert the distribution medium into the drive.

3. To install InterBase in */usr/interbase*, change your directory as follows:

   ```
   % cd /usr
   ```

   You can install InterBase in any other directory, but after the installation is complete, define a link to */usr/interbase* in another directory.

4. Use the **tar** utility to read the files from the cartridge, magnetic tape, or diskette. The tape contains the */interbase* directory, which has the files and subdirectories that are used during the installation procedure.

   If you are not using the default device, use the f option of **tar** and specify the name of the device you are using. The following example uses the default device:

   ```
   % tar -xvp
   ```

5. When the **tar** finishes restoring the files, run the installation script:

   ```
   % sh interbase/install
   ```

   The installation script moves the files from the distribution tape to directories in */interbase*. It also creates links to this directory from the */usr,/usr/lib,* and the */usr/include* directories. For a complete listing of the InterBase files, see *InterBase Files Installed on Your System*.

Notes

A. If your node is not already registered, a node registration message, as shown below, appears one or more times during the installation. The installation will continue beyond this message. See the *Registering InterBase* section below for node registration instructions.

```
        STOP -- STOP -- STOP

Your node is not registered to use InterBase.
The license file does not exist or could not be opened for
read.

If you are using this node to demonstrate software, or because
the registered node is temporarily unavailable, you may
continue by closing this window.

To register this node for InterBase, call (800) 437-7367
between 9 am and 8 pm Monday through Friday Eastern Standard
time. From outside the U.S. and Canada call (408) 431-5400.
Be prepared to provide the following information:

    Your version is <version number>
    Your node id is <node id>
```

B. If you do not have enough semaphores configured on your machine, you will see this message:

```
% semget failed
```

Refer to *Before You Install* section for information about reseting the number of semaphores. If you do not reinstall InterBase files after resetting the semaphores, you must manually install the sample databases and the help database using **gbak**:

```
% cd /usr/interbase/help
% gbak -r help.gbak help.gdb

% cd /usr/interbase/examples
% gbak -r atlas.gbak atlas.gdb
% gbak -r emp.gbak emp.gdb
```

6. When your system prompt is displayed, the installation is complete.
7. If you have a network connection and want to use the network immediately, kill the *inetd* process and restart it.
8. Logout as superuser.

SCO Installation

InterBase is now installed using the SCO *custom* utility. InterBase is shipped in compressed tar format on three numbered diskettes.

1. Log in as superuser.

   ```
   % su
   ```

2. To install InterBase in *usr/interbase*, change your directory as follows:

   ```
   % cd /usr
   ```

   You can install InterBase in any other directory, but after installation is complete, define a link to *usr/interbase* in another directory.

3. Run the *custom* utility:

   ```
   % custom
   ```

   By default the rfd0 drive is used. If you want to use another drive, type:

   ```
   % custom -m <drive_name>
   ```

   Select *Install a new product* from the *custom* menu, then follow the *custom* prompts to install InterBase.

Note

If you do not have enough semaphores configured on your machine, you will see this message:

```
% semget failed
```

Refer to the *Before You Install* section for information about resetting the number of semaphores. If you do not reinstall InterBase files after resetting the semaphores, you must manually install the sample databases and the help database using **gbak**:

```
% cd /usr/interbase/help
% gbak -r help.gbak help.gdb
% cd /usr/interbase/examples
% gbak -r atlas.gbak atlas.gdb
% gbak -r emp.gbak emp.gdb
```

4. When your system prompt is displayed, the installation is complete.
5. If you have a network connection and want to use the network immediately, kill the *inetd* process and restart it.
6. Logout as superuser.

Registering InterBase

Once you install InterBase, you must register and license it with InterBase Customer Support at Borland. Follow the registration and licensing instructions below which are pertinent to your site.

Upgrading From a Previous Version of InterBase

If you upgraded from a previous version of InterBase, then the license ID you were given for the previous version is still valid. Copy your backed up version of *isc_license.dat* to */usr/interbase/isc_license.dat*. InterBase is now licensed for use.

First Time Installation of InterBase

You must register and license your copy of InterBase with InterBase Customer Support. Follow the instructions below, in *Registering the Installation*, and *Licensing InterBase*.

Registering the Installation

To register your installation:

1. Start qli and ready the sample *atlas.gdb* database:

   ```
   % /usr/interbase/bin/qli
   QLI> ready /usr/interbase/examples/atlas.gdb
   ```

   InterBase displays a "STOP" message indicating your node is not registered. The "STOP" message contains the telephone number of Interbase Customer Support at Borland, the software version, and the machine node ID.

2. Copy this information in the "STOP" message to the license worksheet at the end of this document. You will need this information later to answer the licensing utility prompts.

3. To register the installation, call InterBase Customer Support at one of the following numbers between 9 A.M. and 8 P.M. Eastern Standard time:

   1-800-437-7367 from anywhere in the United States and Canada

   408-431-5400 from outside the United States

When you call Customer Support have the following information ready:

— Your customer number.

— Your purchase order number.

— The license worksheet, complete with the node ID and software version.

4. Write the license ID and registration key information provided by Customer Support on the license worksheet.

Licensing InterBase

After you have registered InterBase, you must license your installation using the **iscinstall** utility.

To license your installation:

1. Log in as superuser.

    ```
    % su
    ```

2. Run the **iscinstall** utility:

    ```
    % /usr/interbase/bin/iscinstall
    ```

    Note

    To cancel the **iscinstall** utility, type `exit` or `quit` at any of the prompts.

3. Answer each prompt with the information you recorded on the license worksheet, or press the Return key to use the default value. The default value for each **iscinstall** prompt is in parentheses. The prompts are described below.

    - `Do you need instructions (N)?`

      Type Y if you want instructions for using the **iscinstall** utility. If you exit before completing the instructions by typing exit or quit, you must restart **iscinstall**. If you complete the instructions, you remain within the utility and the next prompt is displayed.

    - `License file (/usr/interbase/isc_license.dat):`

      Press the Return key to use the default pathname (*/usr/interbase/isc_license.dat*), or type the filename you want to use to store the licensing information. (If you do not use the default file, you must append the licensing information to the *isc_license.dat* file after the licensing procedure is complete.)

- `Check for duplicate licenses (N)?`

  Press the Return key if you do not want to check for duplicates, which may result in duplicate licenses being registered.

  Type `Y` to check for duplicates, which will assure that no duplicate licenses are registered.

- `Enter PRODUCT:`

  Type `interbase`

- `Enter VERSION:`

  Press the Return key unless otherwise instructed by Customer Support when you registered InterBase.

- `Enter OPTIONS:`

  Type the options that apply to your license, as instructed by Customer Support. Do not type commas or spaces between each option letter/number. The following letters/numbers activate the indicated license options:

Table 4: License Options

Character	License option
I	Internal relation access
E	External relation access
R	Remote interface
D	Data definition utilities
S	Remote server
Q	qli
L	gpre for all languages, except ADA
C	gpre for C
F	gpre for FORTRAN
A	gpre for ADA
3	gpre for C++

- `Enter ETHER:`
  `Enter NODE:`

  Type the machine Ethernet ID or node ID. For HP machines, type the node ID for your machine; for MIPS type the machine Ethernet ID.

  `Enter UNTIL:`

  Press the Return key unless you purchased a temporary license for product evaluation. If you have a temporary license, type the expiration date, provided by InterBase Support, using the format, DD-MMM-YYYY.

- `Enter COMMENT:`

  Type an optional comment of up to 80 characters, or press the Return key if you do not want to enter a comment.

- `Enter ID:`

  Type the license ID provided by InterBase Customer Support.

- `Enter KEY:`

  Type the license key provided by InterBase Customer Support.

4. When you finish entering the licensing information, **iscinstall** displays the information you entered.

   ```
   Entered: PRODUCT interbase, NODE 1700u98a, ID isc-76, KEY 93-
   1-2-2
   ```

   — If the record is not valid, **iscinstall** prints an error message and does not create the license file entry. Verify that the information you entered matches the above instructions and information provided by InterBase. If the problem still exists, contact the InterBase Support hotline at (800) 437-7367.

   — If the record is valid, it is placed in the *isc_license.dat* license file on your system. If you specified a different authorization file in which to place the license record, you must append the licensing information from that file to *isc_license.dat*.

5. **iscinstall** then begins another licensing routine and prompts you for the next InterBase product:

   ```
   Enter PRODUCT:
   ```

   If you are not installing another InterBase product, type `quit` or `exit`.

6. To exit from the **iscinstall utility**, type N at the following prompts:

```
License another node (Y)?: n
Another license file (Y)?: n
```

7. Verify that the *isc_license.dat* file has "read" permission for all InterBase users.

8. Log out as superuser.

iscinstall Messages

The following table lists iscinstall messages and the reasons for their occurrences. Where appropriate, corrective actions are suggested.

Table 5: iscinstall Messages

Message	Probable Cause
Cannot open help file.	The help file either does not exist or does not have read permission.
Duplicate licenses will be registered.	You typed N in response to the "Check for duplicate licenses" prompt.
Duplicate licenses will NOT be registered.	You typed Y in response to the "Check for duplicate licenses" prompt.
File <filename> cannot be read or created. Check the file's permissions.	The license file you want to use cannot be opened. (The most likely cause is that the file does not have read/write permission or you do not have write access to the directory.) After you correct the file permissions, rerun **iscinstall.**
File <filename> does not seem to exist. Create it (Y)?	The file in which license information will be stored does not exist. If you type Y, the file will be created.
File <filename> doesn't appear to be an InterBase License File....Continue (N)?	The file you specified to use as the license file already exists but cannot be used as an InterBase Authorization File. Type Y to continue and enter a different file name.
ID is mandatory, please re-enter.	You did not enter a required license ID at the "Enter ID" prompt.
Invalid character in key.	You entered one or more incorrect characters in the key value. Re-enter the key, correcting the invalid character(s) shown in the message.
Invalid character in node ID.	You entered one or more incorrect characters in the node value. Re-enter the node, correcting the invalid character(s) shown in the message.

Table 5: iscinstall Messages

Message	Probable Cause
Invalid date.	You entered an incorrect date or did not enter the date in a valid format. Use the date format, DD-MMM-YYYY. For example, January 21, 1991 should be entered as 21-JAN-1991.
Invalid format for key.	You entered the key value using an incorrect format. Re-enter the key using the format shown in the message.
Invalid key value, check entries for accuracy. Please retry.	You entered an incorrect key value. Rerun **iscinstall** and enter the correct key value. If you see this message again, contact InterBase Customer Support.
Invalid license option. Valid options are: <options list>	You entered a letter that does not correspond to an option. Re-enter using one or more of the options listed in the message.
Invalid PRODUCT ID.	You entered an incorrect product ID. Re-enter the product ID using one of the product IDs listed in the message.
Key is mandatory. Please re-enter.	You did not enter a key value. Enter the key value provided by InterBase Customer Support.
Node is limited to a maximum of 8 characters. Please re-enter.	You entered a node ID that is longer than 8 characters. Re-enter the correct node ID.
Product is mandatory! Valid entries are: <entry list>	You did not enter a product ID. Enter a product ID using one of the entries from the list in the message.
Re-try licensing node (Y)?	You exited **iscinstall** before completing the license by typing exit or quit. To rerun **iscinstall**, press the Return key. If you do not want to install InterBase at this time, type N.

Testing the Installation

You should test the InterBase installation as follows:

1. Add */usr/interbase/bin* to your path.

2. Go to your own demo directory and use **gbak** to restore the backup version of one of the sample databases. The following examples use the atlas database:

   ```
   % gbak -r /usr/interbase/examples/atlas.gbak atlas.gdb
   ```

3. Start **qli**, the interactive data manipulation facility, and open the sample database:

   ```
   % qli
   Welcome to QLI
   Query Language Interpreter
   QLI> ready atlas.gdb
   QLI> show relations
    Database "/usr/interbase/examples/atlas.gdb" readied as
   QLI_0
       BASEBALL_TEAMSCITIES
       CITY_TONCROSS_COUNTRY
       ⇓  ⇓
   QLI>
   ```

4. Test **qli** by issuing some print, store, modify, or erase record statements in the sample database. For more information about **qli** statements, see the *Qli Guide*.

5. Quit **qli** and type Y to rollback (i.e., not save) any changes you made:

   ```
   QLI> quit
   Do you want to rollback updates for QLI_0? y
   %
   ```

After You Install

After you install, register, and license InterBase, you may need to make additional modifications to your system to use InterBase most effectively. You should also test your installation to make sure InterBase runs correctly. This section describes general and system-specific changes you might need to make.

General Information for All Systems

- All users who intend to use InterBase must add */usr/interbase/bin* to their paths.
- Any user who accesses a database must have read/write access to that database. To lessen the security risk, you can associate a security class with the database. See the *Data Definition Guide* for information on security schemes.
- If you upgraded to this version of InterBase, you should restore your previous *lock_header* file to */usr/interbase*.
- InterBase uses a lock table managed by Unix System V shared memory directives to determine and keep track of lock and shared memory usage. The lock table file, *usr/interbase/gds.lock.<machine_name>*, contains a lock header block, process blocks, lock request blocks, lock blocks, and lock history records. You may need to change or update this file. For more information, see *Lock Manager*, below.

Additional Information for VAX Ultrix DECnet

1. If you are using VAX Ultrix and you want to use InterBase with DECnet remote database capability, you must use the **ncp** utility to define the DECnet object (**gds_db**) to the DECnet database:

   ```
   NCP> define object gds_db number 128
       /usr/interbase/bin/gds_dnet_server
   NCP> set object gds_db number 128 file
       /usr/interbase/bin/gds_dnet_server
   ```

2. Verify that users of the remote capability have the proper privileges and quotas on the machines where remote databases are stored. The InterBase remote database uses the DECnet default or a proxy account, unless an account is associated with the DECnet object. See the *DECnet-VAX System Manager's Guide* for information about defining DECnet objects.

Lock Manager

InterBase uses a lock table managed by Unix System V shared memory directives to determine and keep track of lock and shared memory usage. The lock table file, */usr/interbase/gds.lock.<machine name>*, contains a lock header block, process blocks, lock request blocks, lock blocks, and lock history records.

Shared memory and semaphores are limited system resources built into your kernel. All processes share from the pool of resources that is hard-coded into your operating system. When one application claims shared memory and/or semaphores, the pool size is decreased until that application returns the resource. The first InterBase process claims the default or *lock_header*-specified amount of shared memory and semaphores for all InterBase processes. These resources are not returned until you run **gds_drop** or the system is rebooted. The more semaphores you claim, the fewer semaphores are available to other applications.

The **gds_relay** process performs lock manager signal delivery. This process runs only when needed. If you stop all InterBase processes, you do not need to also stop **gds_relay**.

Lock Header File

The lock header file defines the maximum size of the lock table and contains other parameters for processes that use shared memory, semaphores, and signals.

The number of bytes of shared memory you need depends on how many buffers, relations, databases, transactions, and processes you expect to be running concurrently. To determine approximately how large the SHMSIZE parameter in the *lock_header* file should be, use the following information:

- one lock is required for each buffer, relation, database, and transaction
- each lock uses 90 bytes
- each process uses 36 bytes
- 100 bytes are required for the lock header block
- 288 bytes are required for the lock history records

The default lock header parameters, which InterBase uses if you do not have a *lock_header* file, are described below. These parameters are listed in the */usr/interbase/lock_header_template* file. (SHMFACTOR, TIMEOUT, and SEMKEY are no longer used by InterBase for locking. If you have them in your *lock_header* file, they will not affect locking.)

Table 6: Default Lock Header Parameters

Parameter	Default	Description
SHMSIZE	32,768	Specifies the size in bytes of the shared memory lock table. SHMSIZE can be expanded up to the maximum size allowed for shared memory in your machine kernel (SHMMAX). Refer to your operating system documentation for kernel size limits.
SEMCOUNT	10, 25, or 32*	Specifies the number of semaphores reserved for InterBase locking. In V33 the maximum number of semaphores that can be reserved for Interbase locking has been increased from 32 to 128. For HP9000/400, SEMCOUNT must be at least 17 less than the maximum number of semaphores allowed on your machine because 16 are required for central server access and 1 is required for events. For all others, SEMCOUNT must be at least 1 less than the maximum because 1 is required for events. **Note:** Do *not* modify SEMCOUNT without reading the following sections.
BLKSIG	SIGUSR1	Specifies the signal number to use to signal InterBase processes for lock requests. The default for BLKSIG is taken from the */usr/include/sys/signal.h* file. BLKSIG must be a valid signal number.

* DG-AViiON DG-UX, and Silicon Graphics IRIX, default to 25; SCO defaults to 10; all others default to 32.

On UNIX systems, the maximum number of semaphores that may be reserved for InterBase locking has been increased from 32 to 128. InterBase default settings are considerably lower and vary according to your system. To increase the number of reserved semaphores, you can modify the SEMCOUNT parameter in */usr/interbase/lock_header*.

Semaphores

Semaphores are used for locks, central servers, and events. A group of SEMCOUNT semaphores is allocated for locking. When lock conflicts occur (e.g., two or more processes attempting to access the same lock), the processes communicate using one semaphore per conflicting lock request. The number of semaphores required depends primarily on the number of concurrent InterBase processes sharing access to one or more databases, the complexity of the transactions, the page size, and the number of relations.

InterBase allocates a group of 16 semaphores for central server access, so no more than 16 processes can use central server access at any one time. One semaphore is used for all events. You cannot change the number of semaphores allocated for central servers or events.

Lock Manager Semaphores

The total number of semaphores available for *all* processes on your system is hard-coded in the operating system kernel, and may be less than 128. To increase the number of semaphores available for InterBase locking *and* all other processes on your system, you may have to reconfigure and rebuild your kernel. The following kernel configuration options are related to the SEMCOUNT parameter and may need to be modified. Some systems do not use all of these options, and some systems may not permit the total number of semaphores to be set as high as 128. Consult your UNIX system documentation before making any kernel changes.

Table 7. Kernel Configuration Options

Option	Purpose
SEMMNS	Sets the maximum total number of semaphores for your system.
SEMMNI	Specifies the maximum number of semaphore clusters (groups of semaphores).
SEMMSL	Indicates the maximum number of semaphores per cluster.
SEMMNU	Sets the number of undo structures. Undo structures are used to free objects that remain closed to other users when, for example, a process dies leaving that object closed. You should have one undo structure for each InterBase process you expect to run.

InterBase 3.2 Considerations

Applications linked with the *gds_b.a* back end shipped with InterBase V3.2 cannot take advantage of an increase to the number of lock semaphores. These applications should either be relinked with the *gds_b.a* back end shipped with V3.3, or, if possible, should be relinked to use shared libraries instead.

In an environment running both InterBase V3.2 and V3.3 applications, if the lock table is accessed first from a V3.3 database and the number of semaphores is greater than 32, then subsequent V3.2 database users will not be able to access the V3.3 database. If the lock table is accessed first from a V3.2 database, then any subsequent V3.3 database users will be limited to a lock table of 32 semaphores.

Note

Do not attempt to change any *lock_header* file parameters until you have read the *Modifying the Default Lock Header File* section below.

Error Messages

If you see any of the messages below, you should either create a *lock_header* file so the default values are not used, or change the parameters in your existing *lock_header* file. (See the *Modifying the Default Lock Header Values* section below.)

Table 8: Lock Manager Error Messages

Error Message	Description
Fatal lock manager error: lock manager out of room	The lock table is full and the process that requested the lock is terminated. The lock table will still be usable and other processing can continue, but if heavy contention occurs, it will cause processes to abort if their lock requests overflow the lock table. You should increase the SHMSIZE parameter in the *lock_header* file.
Fatal lock manager error: semaphores exhausted	No more semaphores are available. You should increase the SEMCOUNT parameter in the *lock_header* file.

For the BLKSIG parameter, no error messages are displayed. If another application is using BLKSIG to communicate, that application may capture signals intended for InterBase processes, which will cause InterBase processes to hang.

Modifying the Default Lock Header Values

If you want to modify any parameters in the *lock_header* file, follow the procedure shown below. Changing any lock header parameter is not a dynamic process; to avoid having to frequently change the values, you should set them to the maximum possible values you will need. The examples use */usr/interbase* as the directory in which InterBase was installed. If you installed InterBase in a different directory, substitute your directory name for */usr/interbase*.

1. Verify that all users stop accessing any database served by the node and that no users access the node until you finish editing the *lock_header* file.

2. Log in as superuser.

3. Remove the current lock table, deleting its reserved shared memory and semaphores, using the **gds_drop** utility with the **-l** (lock header) option:

   ```
   % gds_drop -l
   ```

4. If this is the first time you are changing the lock header parameters, copy the *lock_header_template* file to a *lock_header* file.

   ```
   % cp /usr/interbase/lock_header_template
   /usr/interbase/lock_header
   ```

5. Edit the *lock_header* file to change any of the lock header parameters.

   — If you change the SEMCOUNT parameter, do not increase it so that your kernel becomes too large; each semaphore requires space for data structures to support it.

   — To change the BLKSIG parameter, uncomment the line and change the value to another valid signal number.

   — SHMFACTOR, TIMEOUT, and SEMKEY are no longer used by InterBase for locking. If you have them in your *lock_header* file, they will not affect locking.

When any process attaches a database, the new lock parameters are used.

Note: You do not have to reboot the node after modifying the *lock_header* file.

InterBase Servers

InterBase includes the following servers on UNIX:

- **gds_inet_server**, a single-client TCP/IP server. This server runs as a separate background process.
- **gds_dnet_server**, a single-client DECnet server.

gds_inet_server

When you attach a database from remote node, a single-client **gds_inet_server** automatically starts because the installation script added the following TCP/IP configuration line to the */etc/inetd.conf* file:

```
gds_db stream tcp   nowait root
 /usr/interbase/bin/gds_inet_server gds_inet_server
```

gds_dnet_server

If you are using the DECnet server, you must use the **ncp** utility to add the **gds_dnet_server** object (object 128) to the DECnet database. Refer to the *Guide to DECnet-VAX Networking* for information about **ncp**.

Using pyxis

Depending upon the platform you use, **pyxis** may or may not be available. In some cases you may need to set up your system so that **pyxis** works correctly. The chart below indicates the availability of **pyxis** by platform. On some platforms you may be required change your system setup to make **pyxis** work correctly on your terminal.

Table 9: forms (pyxis) Availability

Platform	Available y/n	Comment
IBM (AIX)	yes	xterm only
SGI	yes	Requires enhancement of vt100 and xterm descriptions. If **pyxis** is used in a program, you must add code to return a terminal to line buffered mode. See below.
IBM (SCO)	yes	
Motorola IMP Motorola Delta	no	
Data General	yes	Requires enhancement of vt100 and xterm descriptions and setting keyboard mappings. If **pyxis** is used in a program, you must add code to return a terminal to line buffered mode. See below.
HP 9000/ 300, 400 HP 9000 / 600, 800 HP 9000/ 700	yes	Requires setting up function keys. See below.

Using pyxis on DG and SGI

For InterBase **pyxis** to work correctly, the terminfo descriptions for vt100 and xterm terminals must be enhanced with the correct key sequences for the home key.

1. Log in as root and go to the */tmp* directory:

```
cd /tmp
```

2. Use the infocmp program to extract the terminal descriptions to files:

```
infocmp -I vt100 > vt100.info
infocmp -I xterm > xterm.info
```

3. Edit the *vt100.info* file by adding the following line immediately after the second line of the file, using tabs to align the columns. Note that you must include the final comma and a space after that comma.

```
khome=\EOH,
```

4. Edit the *xterm.info* file by adding the following line immediately after the second line of the file, using tabs to align the columns. Note that you must include the final comma and a space after that comma.

```
khome=\E[H,
```

5. Test the changes you made:

```
tic -c vt100.info
tic -c xterm.info
```

   If any error messages are displayed, repeat the above instructions for the file(s) with the error, beginning with step 2.

6. Rename the old terminfo files:

```
mv /usr/lib/terminfo/v/vt100 /usr/lib/terminfo/v/vt100.sav
mv /usr/lib/terminfo/x/xterm /usr/lib/terminfo/x/xterm.sav
```

7. Compile your changes:

```
tic vt100.info
tic xterm.info
```

8. For each user running InterBase on a Silicon Graphics machine, the following line must be added to their *.cshrc* or *.profile* file so the new home key definition will be used each time a shell is created:

```
bindkey -l home
```

If you are using **pyxis** in your program, the curses function leaves STDOUT in a nonbuffered mode when exiting a form. To return the terminal to line buffer mode, you must insert the following lines into your routine:

```
#include <stdio.h>
char buf[BUFSIZ];
setvbuf(stdout, buf, _IOLBF, BUFSIZ);
```

For example, the following routine is used to return a terminal to line buffer mode:

```
#include <stdio.h>
#include <curses.h>
main ()
{
char buf[BUFSIZ];
int ans;

initscr ();

clear ();

wrefresh (stdscr);

endwin ();

setvbuf(stdout, buf, _IOLBF, BUFSIZ);
printf ("Do you want to commit the updates (Y/N): ");
ans = getchar();
}
```

If you are using **pyxis** with **qli**, you may get a blank screen after you accept or reject a form. To correct this, exit **qli**. When you re-enter Forms through **qli**, your updates will be visible.

Additional procedures on DG only

To enable **pyxis** on Data General, you will need to perform the following:

- From within an **mterm** window, type the following command appropriate to the shell:

Command	Shell
setenv TERM vt100	cshell
TERM=vt100 export term	Bourne or kshell

- Select the OPTIONS panel within the window and set the mode to vt100.
- Use the keyboard mappings as displayed below or display them from the HELP menu:

vt100/vt52 key	AViiON key
PF1	ALT-Insert
PF2	ALT-PgUp
PF3	ALT-Delete
PF4	ALT-PgDn

Using pyxis on HP-UX

X Windows

If you run **pyxis** under X Windows with HPterm on HP-UX Version 8 and want to use function keys, you must create, if necessary, and add the lines listed below to the files, */usr/lib/X11/sys.Xdefaults* and */usr/lib/X11/app-defaults/HPterm*.

(Note that xterm emulation does not work with HP-UX Version 8 and **pyxis**.)

```
softkeyInitialize16:TRUE
HPterm*f1Attribute:0
HPterm*f2Attribute:0
HPterm*f1Attribute:0
HPterm*f3Attribute:0
HPterm*f4Attribute:0
HPterm*f5Attribute:0
HPterm*f6Attribute:0
HPterm*f7Attribute:0
HPterm*f8Attribute:0
HPterm*f9Attribute:0
HPterm*f10Attribute:0
HPterm*f11Attribute:0
HPterm*f12Attribute:0
HPterm*f13Attribute:0
HPterm*f14Attribute:0
HPterm*f15Attribute:0
HPterm*f16Attribute:0
```

HP VUE

1. If you run **pyxis** under HP VUE, you must create the file, */usr/lib/X11/vue/Vuelogin/Xterms* and add the lines listed above for X Windows to it.

2. You must add the following line to the file, */usr/lib/X11/vue/Vuelogin/Xconfig* with no space before, after, or within it:

```
Vuelogin:environment:XENVIRONMENT=/usr/lib/X11/vue/Vuelogin/Xterms
```

Linking Applications

The information that follows supplements the general information provided in the *Programmer's Guide*. Information on linking application programs with pyxis is provided later in this installation guide.

Table 10: Linking Application Programs

Platform/OS	To use	Link with
IBM RISC 6000 AIX 3.2	shared library	`-lgdsshr`
	back end	`-lgds_b`
	pipe server	not supported
	Comment: On IBM (AIX) linking your application to the shared library can yield smaller images that have the execution speed of the full library. The shared library is not statically bound into the application so InterBase versions are upgraded automatically.	
SGI IRIX 3.3 or 4.0	shared library	`-lgds_s -lgdsf_s -lsun`
	back end	`-lgds_b`
	pipe server	`-lgds`
	Comment: Silicon Graphics warns against mixing System V and BSD signals in a process. The InterBase access method uses BSD signals. If your application must use System V signals, you should link to the InterBase library, *gds.a*. You should note that linking to *gds.a* will result in degraded performance because the access method will be executing in a sub-process instead of in the same process as your application. **Note** If you intend to use the shared library on SGI, refer to the section in this document called *Using Shared Libraries on SGI*.	

Table 10: Linking Application Programs

Platform/OS	To use	Link with
SCO SCO UNIX 2.2.4	shared library	not supported
	back end	`-lgds_b -lmalloc -ls_s -lrpc -lsocket`
	pipe server	`-lgds -lcrypt_i -lmalloc -lc_s -lrcp -lsocket`
	Comment: On SCO, before you can link a C application to InterBase, the files *libsocket.a* and *librpc.a* must exist in the */lib* or */usr/lib* directory. To use events, you must link using the *-lgds_b* option (back end). If you intend to link with the InterBase back end (*gds_b.a*), you must use the SCO encryption library. This library is available directly from SCO.	
Motorola IMP UNIPLUS V3.1 Motorola Delta Sys V/68 Rel 3	shared library	`-lgds_s -lgdsf_s` (IMP only, unsupported on Delta)
	back end	`-lgds_b`
	pipe server	`-lgds`
	Comment: Linking syntax for Motorola may change in an upcoming release.	
Data General DG-UX 5.4	shared library	`-lgds -lgdsf`
	back end	`-lgds_b`
	pipe server	`-Bstatic -lgds -Bdynamic`
	Use the following options in the compiler command to use the pipe server rather than the shared library: C `-Bstatic -lgds -Bdynamic` FORTRAN `/usr/interbase/lib/gds.a` C++ `/usr/interbase/lib/gds.a`	

Table 10: Linking Application Programs

Platform/OS	To use	Link with
HP 9000/ 300, 400 HP-UX 8.0 HP 9000/ 600, 800 HPUX 8.0.5 HP 9000/ 700 HP-UX 8.0.5	shared library	`-lgds -ldld`
	back end	`-lgds_b`
	pipe server	`-Wl,-a,archive -lgds -Wl,-a,default -ldld`
	Comment: If you use the shared library on the HP 9000/800, you must relink your applications.	

Linking Application Programs with pyxis

Table 11: Linking Application Programs with pyxis

Platform/OS	To use	Link with
IBM RISC 6000 AIX 3.2	shared library	`-lgdsshr -lcurses`
	back end	`-lgds_b -lcurses`
	pipe server	not supported
SGI IRIX 3.3 or 4.0	shared library	`-lgds_s -lgdsf_s -lsun -lgds_pyxis -lcurses`
	back end	`-lgds_b  -lgds_pyxis -lcurses`
	pipe server	`-lgds -lgds_pyxis -lcurses`
SCO SCO UNIX 2.2.4	shared library	not supported
	back end	`-lgds_b -lmalloc -lc_s -lrpc -lsocket -lcurses`
	pipe server	`-lgds -lcrypt_i -lmalloc -lc_s -lrcp -lsocket -lcurses`

Table 11: Linking Application Programs with pyxis

Platform/OS	To use	Link with
Data General DG-UX 5.4	shared library	`-lgds -lgdsf -lgds_pyxis -lcurses`
	back end	`-lgds_b -lgds_pyxis -lcurses`
	pipe server	`-Bstatic -lgds -Bdynamic -lgds_pyxis -lcurses`
HP 9000/ 300, 400 HP-UX 8.0 HP 9000/ 600, 800 HPUX 8.0.5 HP 9000/ 700 HP-UX 8.0.5	shared library	`-lgds_pyxis -lcurses -lgds -ldld`
	back end	`-lgds_pyxis -lcurses -lgds_b -ldld`
	pipe server	`-lgds_pyxis -lcurses -Wl,-a,archive -lgds -Wl,-a,default -ldld`
	Comment: If you intend to use **pyxis** in your application, you must link with a separate library, *gds_pyxis.a* from the InterBase shared library. If you use the shared library and **pyxis** on the HP 9000/700, you must relink your applications. *-ldgds_pyxis* should be the first library to be linked. All other libraries should follow.	

InterBase Language Support

If you intend to use any of the InterBase supported languages, refer to the information provided in the *Programmer's Guide* and the information that follows.

FORTRAN

InterBase provides support for HP FORTRAN 77 Version 2.00, DG-UX FORTRAN 77 Version 1.8.5.5, Silicon Graphics FORTRAN 77 Version 1.3, and IBM (AIX) FORTRAN Version. The following information supplements the information currently in the *Programmer's Guide.*

- For all HP9000s, Data General, IBM (AIX), and Silicon Graphics machines, FORTRAN files with embedded GDML or SQL should have the file extension *.ef*. The output file produced by **gpre** when you preprocess your file will have an *.f* extension.
- For all HP9000s, Data General, IBM (AIX), and Silicon Graphics machines, any GDML call with a dollar sign ($) in it must have an underscore substituted for the dollar sign. GDML calls produced by **gpre** will be in the correct format. For example:

  ```
  gds_$<xxx>    ---->    gds_ _<xxx>
  ```

- For the HP9000/300, IBM (AIX), and Data General, database declarations have global, static, or extern scope. Static scope has no meaning for FORTRAN and is interpreted as global. For a given database in a multifile application, global scope or static scope can occur in only one file. In all other files, declarations of that same database must contain the keyword, EXTERN. The format of a declaration using global scope is:

  ```
  DATABASE <alias name> = '<database name>.gdb'
  ```

  The format of a declaration using either static or extern scope is:

  ```
  DATABASE <alias name> = [EXTERN/STATIC] '<database name>.gdb'
  ```

- When you link your HP9000/300 FORTRAN application to the InterBase libraries, you may see the following error message:

  ```
  ld: multiply defined symbol_environ in file /lib/libc.a
  ```

  This message is caused by a known bug in HP 9000/300 FORTRAN, but it has no known effect on FORTRAN applications with embedded GDML. This item does not apply to FORTRAN applications.
- **blob_edit** calls in DG-UX FORTRAN Versions 1.8.5.2 and earlier fail without an error message. This bug is corrected in Versions 1.8.5.3 and later of the compiler.

- When you use DSQL or dynamic DDL, you set up a control structure which includes assigning an address to a variable. Since HP9000/300 FORTRAN does not use pointers, the InterBase library includes a C function, **isc_baddress**. This function takes a variable as its argument and returns, as a long, the address of that variable. In your application program on HP9000/300, call **isc_baddress** as follows:

  ```
  object_address = ISC_BADDRESS (object)
  ```

  This item does not apply to FORTRAN.
- For Silicon Graphics FORTRAN only, the length and name_lengths arguments in **blob_$edit**, **blob_$dump**, and **blob_$display** must be of type long.

COBOL

InterBase now supports Micro Focus COBOL on Data General, IBM (AIX), and HP9000 700/800 platforms.

Compiling and Linking COBOL programs

The following information supplements the information currently in the *Programmer's Guide*.

Using Microfocus COBOL requires the following:

- Compiling your programs using the following file extensions: *.cbl* or *.ecbl*
- Using the **-ansi** switch with **gpre**
- Using the semicolon (;) as a delimiter in your DSQL statements, since there is no command to determine delimiters.

To link your programs:

Table 12: Linking COBOL Programs

Platform	Link with:
Data General	Refer to the Micro Focus notes regarding linking and compiling for the ELF environment
IBM (AIX)	(link with the shared library) e.g. , cob -x <foo>.cbl -o<foo> -lgdsshr
HP	(link with the pipe) e.g. , cob -x <foo>.cbl -o<foo> -lgds

The chart below displays InterBase data types and the corresponding COBOL code:

Table 13: COBOL Supported Data types

InterBase Data type	**Platforms**		
	IBM (AIX)	**HP 700/800**	**DG**
short	PIC S9 (4) COMP	PIC S9 (4) COMP	PIC S9 (4) COMP
long	PIC S9 (9) COMP	PIC S9 (9) COMP	PIC S9 (9) COMP
float*	PIC X(4)	PIC X(4)	PIC X(4)
double*	PIC X(B)	PIC X(B)	PIC X(B)
char(n)	PIC X(N)	PIC X(N)	PIC X(N)
varying	PIC X(N)	PIC X(N)	PIC X(N)
date	PIC S9 (18) COMP	PIC S9 (18) COMP	PIC S9 (18) COMP
blob	PIC S9 (18) COMP	PIC S9 (18) COMP	PIC S9 (18) COMP

* Assumes the user has converted data to or from the corresponding C data types.

Using DSQL in COBOL

To enable use of DSQL in COBOL, it will be necessary to get variable addresses to complete the SQLDA. Refer to chapter one in the InterBase *DSQL Programmer's Guide* for information on variable addresses. If your program supports the **giving** clause, use the function **isc_baddress()**, where the value returned is the address of the function argument. If your program does not support the **giving** clause, use the subroutine **isc_baddress_s**, where the value returned as the second argument is the address of the first argument. For information on using these calls, refer to *dsql.ecbl* located in */examples*.

Using Shared Libraries on SGI

Note

The InterBase Version 3.3 shared library is not compatible with the InterBase Version 3.1A shared library. To use the V3.3 shared library, you **must relink** any applications that were previously linked with the V3.1A shared library.

The InterBase access method has two shared libraries: *gds_s.a* and *gdsf_s.a*. Shared library *gds_s.a* is loaded at addresses 0x0b800000 through 0x0bffffff; *gdsf_s.a* is loaded at addresses 0x0b000000 through 0x0b7fffff. If these addresses conflict with the addresses of other shared libraries used by your application, you can link to either of the InterBase archive libraries, *gds.a* or *gds_b.a*, instead of the shared libraries.

To compile user defined functions in C on SGI, use this command instead of the UNIX commands listed on page 9-5 of the *Data Definition Guide*:

```
cc -c -G 0 <filename>
```

Creating a function library under SGI is similar to creating a function library under SunOS4.0. Creating a Function Library on SunOS4.0 is described on pages 9-10 through 9-12 of the *Data Definition Guide*.

1. Log onto the node where the database resides.

2. Copy *functions.c, udf.c, shrudf.c,* and *udf.bind* from the */usr/interbase/examples* directory into your local directory.

3. Follow the instructions for Step 2 on page 9-10 of the *Data Definition Guide* to define each external symbol and define a pointer to hold that symbol.

4. Compile *functions.c, udf.c, shrudf.c,* and *udf.bind*:

```
cc -c -G 0 -DSHLIB_DEFS functions.c
cc -c -G 0 -DSHLIB_DEFS udf.c
cc -c -G 0 -DSHLIB_DEFS shrudf.c
```

5. Change the name of each output file from ".o" to ".bin". For example:

```
mv functions.o functions.bin
```

6. Add your functions to the *udf.bind* file.

7. Create the shareable library:

   ```
   mkshlib -s udf.bind -h gdsf_s.a -t gdsf_s -q
   ```

8. To make the function library available by using shareable libraries:

   — Modify the */usr/lib/libgdsf_s* link to point to the new *gdsf_s* that you created in the current directory.

   — Modify the */usr/lib/libgdsf_s.a* link to point to the new *gdsf_s.a* that you created in the current directory.

9. Test your functions.

10. To make the new shared library the standard library, move *gdsf_s* and *gdsf_s.a* to the */usr/interbase/lib* directory and set the links in */usr/lib* to once again point to */usr/interbase/lib*.

Using Journaling on SCO

If during your SCO machine setup, you added the host domain name to your machine name, you must add the full machine name, *<machine name>.<host name>* as a primary name or as an alias to the */etc/hosts* file. If the complete machine name is not in the */etc/hosts* file, InterBase journaling will not work.

InterBase Files Installed On Your System

When the installation is done, the following InterBase files and links are added to your system.

Note

Full file names are shown below. If your system uses short file names, the file names are truncated after the fourteenth character.

- Files added to the */usr/interbase* directory:
  - *RELNOTE.33*, InterBase Version 3.3 Release Notes
  - *RELNOTE.PRV*, InterBase previous versions release notes and documentation corrections
  - *gds.lock.<machine name>*, lock table file
  - *inetd.conf.isc*, addendum to */etc/inetd.conf*, the TCP protocol configuration
  - *install*, installation script
  - *interbase.log*, file of system messages, if any exist
  - *interbase.msg*, **qli** message file
  - *isc_event.gbl.<machine name>*, event communication file
  - *isc_ins_hlp.dat*, help text for the **iscinstall** utility
  - *isc_license.dat*, licensing information (*isc_lic.dat* on SCO)
  - *lock_header*, user-created lock_header file, if it exists
  - *lock_header_template*, default lock_header parameters
  - *services.isc*, addendum to */etc/services*, the TCP protocol configuration
  - *isc.gdb*, PC security database

- Images added to the */usr/interbase/bin* directory:
  - **fred**, InterBase forms painter
  - **gbak**, backup and restore utility
  - **gcon**, console program that communicates with the journal server
  - **gdef**, data definition utility

- — **gds_dnet_server**, DECnet server (if DECnet is supported)
- — **gds_drop**, utility to delete shared memory and semaphores
- — **gds_inet_server**, TCP server
- — **gds_lock_print**, lock manager's diagnostic utility
- — **gds_pipe**, stand-alone InterBase server (not on IBM)
- — **gds_relay**, process that handles cross-group signaling
- — **gfix**, database maintenance utility
- — **gltj**, journal server
- — **gpre**, GDML and SQL host language preprocessor
- — **grst**, database restoration utility used with after-image journaling
- — **gstat**, utility used by InterBase Customer Support
- — **iscinstall**, license and registration utility
- — *journal.gbak*, backup copy of the journal database
- — **qli**, interactive query and update facility
- — **gsec**, security administrator for SQL Link

- Files added to the */usr/interbase/examples* directory:
  - — backup files for the following sample databases:

    atlas.gbak

    c_guide.gbak

    emp.gbak

    nc_guide.gbak

    slides.gbak

    stocks.gbak
  - — sample GDML and SQL programs for supported languages on your system

- Files added to the */usr/interbase/help* directory:
  - — *help.gbak*, backup file for the **qli** help database

  - *help.gdb,* **qli help database created during the installation procedure from** */usr/interbase/help/help.gbak*

- Files added to the */usr/interbase/include* directory:
  - *functions.c,* template example for blobs and UDFs (unavailable on HP and DG)
  - *gds.h,* link to */usr/include/gds.h*
  - *perf.h,* file for performance analysis
  - *gds.hxx,* C++ include file

- Files added to the */usr/interbase/lib* directory:
  - *gds.a,* link to */usr/lib/libgds.a,* which invokes *gds_pipe*
  - *gds_b.a,* link to */usr/lib/libgds_b.a,* which is the full Access Method back end
  - *gds_pipe.a,* file for rebuilding *gds_pipe* with access to UDFs (not on IBM)
  - *gds_inet_server.a,* file for rebuilding *gds_inet_server* with access to UDFs
  - *gds_dnet_server.a,* file for rebuilding *gds_dnet_server* (if DECnet is supported) with access to UDFs on Ultrix
  - *gdsshr.a,* shared library (on IBM RS/6000 only)
  - *gds.sl,* shared library (HP only)
  - *gds.so,* shared library (DG only)
- InterBase language drivers named according to language subtypes.

InterBase License Worksheet

Use the information you receive from InterBase Customer Support and from the "STOP" message to complete the **iscinstall** utility's prompts. This information is required to license each InterBase installation.

PRODUCT	____________________	PRODUCT	____________________
OPTIONS	____________________	OPTIONS	____________________
NODE / ETHER	____________________	NODE / ETHER	____________________
CLASS	____________________	CLASS	____________________
UNTIL	____________________	UNTIL	____________________
COMMENT	____________________	COMMENT	____________________
ID	____________________	ID	____________________
KEY	____________________	KEY	____________________

PRODUCT	____________________	PRODUCT	____________________
OPTIONS	____________________	OPTIONS	____________________
NODE / ETHER	____________________	NODE / ETHER	____________________
CLASS	____________________	CLASS	____________________
UNTIL	____________________	UNTIL	____________________
COMMENT	____________________	COMMENT	____________________
ID	____________________	ID	____________________
KEY	____________________	KEY	____________________

PRODUCT	____________________	PRODUCT	____________________
OPTIONS	____________________	OPTIONS	____________________
NODE / ETHER	____________________	NODE / ETHER	____________________
CLASS	____________________	CLASS	____________________
UNTIL	____________________	UNTIL	____________________
COMMENT	____________________	COMMENT	____________________
ID	____________________	ID	____________________
KEY	____________________	KEY	____________________