Frontbase®

Users Guide

Because of last-minute changes to Frontbase, some of the information in this manual may be inaccurate. Please read the Release Notes on the Frontbase distribution for the latest up-to-date information.

Revised: 00/12/06 gab

How to Contact Frontbase:

U.S.A. and international	FrontBase Datavej 52 DK-3460 Birkerød Denmark
Ordering	Voice: +45 4582 6262 Fax: +45 4582 0816
World Wide Web	`http://www.frontbase.com`
Registration information	`register@frontbase.com`
Technical support	`support@frontbase.com`
Sales, marketing, & licensing	`info@frontbase.com`

Table of Contents

5 Administration 63

1

Document Guide

FrontBase is a document that was designed for ease of use as well as providing operation guidelines.

This document contains the following chapters:

2

Forward

FrontBase is a scalable relational database server. A few general concepts will help explain what that means.

Architecture

Client/server architectures

are familiar to anyone who has used the world wide web. With the web, your browser (the client) makes a request for a page from some web site (the server). The web site processes the request and delivers a result (the web page). The FrontBase server is similar in some ways to a web server. It listens for requests from FrontBase clients, processes requests, and returns results.

Relational Database server

However, FrontBase is a relational database server. While a web server typically serves web pages for display in a browser, FrontBase handles requests to store and retrieve data. By implementing the "relational" model, FrontBase stores data in application-defined tables, among which, application-defined relations may exist. Tables are defined by their columns. For example, a phone book table might have the following columns: name, phone_number. The actual data is entered in rows of a table. A row in the phone book table might have "John Doe" in the name column and "555-1212" in the phone_number column.

Highly Scalable

FrontBase is highly scalable, which means it can handle very large data sets and high request loads. The size of a data set is typically described by the number of rows in a table. Using its column index-

ing features, FrontBase can efficiently insert and lookup data in tables with millions of rows. In applications with high request loads, FrontBase's replication and clustering features can be exploited to run the same database on multiple servers.

SQL 92 query language

FrontBase implements the industry-standard SQL 92 query language. FrontBase clients use this language to store and retrieve data on the server. They also use the language to manage users, tune performance, and do other administrative tasks. FrontBase implements the SQL 92 standard quite strictly, but also has extensions to handle FrontBase-specific issues. FrontBase uses other standards, such as TCP/IP for client/server communication and Unicode for character value storage.

Inherent Interfaces

With its features, scalability, and adherence to standards, FrontBase is the ideal database for today's custom client/server applications and for building dynamic web sites. Application developers can use FrontBase's FBCAccess C library, JDBC and ODBC adaptors, and other interfaces to develop custom applications. They can use FrontBase's PHP, Perl, and WebObjects adaptors to develop dynamic websites.

3

Introduction

FrontBase is a high performance relational database engine conforming to the standards and demands of today's quality minded developers and users.

This chapter contain the following sections:

Overview

The engine is written in pure ANSI C and benefits from 15+ years of experience with compiler and run-time systems, embedded systems, object-oriented programming, database systems, and command-and-control systems.

The topics in this section are:

Supported Platforms

– "Solaris" on page 38.
– "RedHat 6.x Linux (x86)" on page 42.
– "SuSE Linux (x86)" on page 45.
– "YellowDog Linux (PPC)" on page 48.
– "Debian Linux (x86)" on page 51.
– "Mandrake Linux (x86)" on page 54.
– "FreeBSD (x86)" on page 58.

Designed with forethought

FrontBase provides high performance and conformance to both international and de facto standards such as:

SQL 92: FrontBase is the first industrial strength database engine in compliance with this important international standard. This includes full integrity constraint checking built right into the engine.

Unicode: FrontBase uses Unicode 2.0 exclusively for handling all CHARACTER data, while conserving space by using the UTF-8 standard for representation. This provides easy support for mixed client environments and their varying character sets.

Communication: FrontBase uses sockets for communicating with clients, making it easy for developers to support a wide variety of client platforms.

Administration: FrontBase databases can be administrated from any computer on the internet - a normal Web browser is all that is needed.

Solid Foundation

The underlying truly relational oriented engine provides a solid foundation for dealing with databases very efficiently and without limitations:

– Stringent transaction control
– 100% resiliency against crashes

- Super-fast start-up times
- Terabyte size databases
- Gigabyte size CHARACTER / VARCHAR strings and BLOBs
- Multi-column optimized B-tree indexing with very low overhead
- Host OS filesystem independency
- In-memory caching of tables
- Serializable isolation level with versioned reads
- Row-level locking facilities
- Read-only databases

Standards Compliance

This section describes international standards to which FrontBase adheres. These include SQL 92, Unicode, TCP/IP, and ANSI C.

FrontBase adheres to several international standards, ensuring that you can leverage these standards when developing and deploying your application with FrontBase.

This section will discuss the following:

Full SQL 92 Compliance

FrontBase implements the full SQL 92 standard. Great care has been taken to implement all features defined by the standard and implement them as defined by the standard. While the FrontBase documentation provides several tutorial examples of using SQL 92 with FrontBase, the ultimate guide to SQL 92 is the standard itself. You can obtain "SQL 92 Standard at ANSI" the standard from ANSI for a fee at:

```
http://webstore.ansi.org/ansidocstore product.asp?sku=ISO%2FIEC+9075%3A1992
```

An excellent book by renowned database experts C. J. Date and Hugh Darwen is:

A Guide to the SQL Standard, Fourth Edition

```
http://www.amazon.com/exec/obidos/ASIN/0201964260
```

While it offers an academic approach, it is a very amusing book. Date and Darwen are not fans of many of the decisions that resulted in the final SQL 92 standard.

Unicode Character Representation

FrontBase stores all character data (including CLOBs) using Unicode. Combined with FrontBase's support for CHARACTER SETs and COLLATIONs, this ensures that server-side string comparisons work with character sets other than standard ASCII.

FrontBase's support for Unicode works seamlessly across client platforms. Character data is converted to Unicode on the client side, using the client operating system's native Unicode library. Character data in Unicode format is then passed to the FrontBase server, where it is stored. When a client extracts character data from the server, the client then converts from Unicode format to a format suitable for use on the client platform.

TCP/IP Client/Server Interaction

FrontBase clients and servers use the standard TCP/IP Internet protocol to communicate. This allows tremendous flexibility in how you design and deploy systems that incorporate FrontBase servers. For example, in a WebObjects deployment, you may connect several application servers to a cluster of FrontBase servers in a server area network (SAN). Or, if you don't require such performance and/or redundancy, you can run WebObjects, Apache, and FrontBase on a single server.

ANSI C Codebase

FrontBase is written in ANSI C. This ensures a stable cross-platform codebase and allows us to move FrontBase to new UNIX-based platforms with minimal effort. This offers FrontBase customers flexibility in development and deployment platforms.

The FBCAccess client library is also written in ANSI C. Source code for FBCAccess is available by special request, so if you need to deploy FrontBase clients on platforms (such as PalmOS or PocketPC) where FrontBase servers have not been ported, you can.

Key Features

This section introduces key features of FrontBase which make it the ideal database for Internet applications.

This section introduces key features of FrontBase which make it the ideal database for Internet applications. FrontBase supports most of the important features of "the big boys" and is more standards-compliant than free and open source solutions.

This section will discuss the following:

- Terabyte Databases, Gigabyte Column Values
- FrontBase Security
- Transactions
- Row Level Priveleges
- Live Backup
- Caching
- Multi-server Deployment

Terabyte Databases, Gigabyte Column Values

FrontBase supports terabyte-sized databases, gigabyte-sized character column values, and gigabyte-sized binary large objects (BLOBs) and character large objects (CLOBs).

FrontBase implements its own file system within the files used to store the database. The file system consists of up to 2^32 512-byte blocks, yielding over 2 terabytes of usable space. A character column values, BLOB, or CLOB can occupy up to 2^32 of these 512-byte blocks as well. In practice, large objects will be much smaller so that one does not consume the entire addressable space for the database. Thus, we advertise gigabyte-sized column values.

On platforms where the logical file size cannot exceed the gigabyte range, FrontBase will, when necessary, break its storage for a database into several files that fit within the file system's limits. Since FrontBase maintains its own file system within these files, this parti-

tioning does not impose any limits on sizes of character column values, BLOBs, or CLOBS.

FrontBase Security

FrontBase uses passwords, encryption, and client IP address checks for security.

Passwords

FrontBase provides two layers of passwords for protecting access to databases: database passwords and user passwords.

1. Database password

   If the database password is set, a client must send the database password to the server as part of the connection protocol. If the server cannot verify the password, the client connection is closed immediately.

2. User password

   Each database user can have a password. The password is verified by the server when a session is created for that user. If the verification fails, the session is not created. When a session is successfully created, the protection defined by the SQL 92 standard takes over.

3. Password handling in general

   Passwords may be of any length. Passwords are never exposed outside the client software and they are not even in the database. As soon as an application sends a password to the FrontBase client library, a one-way function is applied to generate a password digest. The function will throw away parts of the password so that it is impossible to deduce the password from the digest. The user name is part of the digest, so two users with the same password will not have the same digest. The password digest is transmitted to the server and used for verification in place of the password.

Encryption

Encryption is used to protect communication channels and data storage. When you create a FrontBase, you may optionally specify

that data stored on the disk should be encrypted. You may also optionally specify that communication channels between the server and its clients must be secure. You must provide an encryption key for each option specified.

1. Encryption of data

   Data stored on the disk is encrypted using a triple DES in cipher block chaining mode on 512 byte blocks. The data store itself is block-oriented with 512 bytes / block, so this effectively encrypts all data, including table definitions, table contents, character data, and BLOBs. The initialization vector depends on the logical position of the block within the system, thus blocks with the same contents will never generate different cipher text blocks. The key used for encryption of data is a 64 bit initialization vector, and 3x56 bits for the DES encryption.

2. Secure channels

   A client and the server are able to establish a secure channel. When a client connects to the server, it receives a public RSA key from the server. The client then generates a set of random session keys: one for outgoing data and one for incoming data. It encrypts those session keys with the public RSA key and sends the results to the server. The server decrypts the session keys sent by the client using its private key. Thus, the client and the server have established a common set of secret keys.

   The algorithm used for encryption of communication data is a triple DES in byte stream mode with cipher text and clear text feed back. The clear text feedback ensures that an error will propagate to all bytes following the error. This ensures simple detection of errors and introduces only a small amount of redundancy.

IP Address Checks

FrontBase implements black and white lists (also known as the "whiskey list") for determining which clients may connect.

When a client connects to the FrontBase server, the IP address of the client is checked against a black and white list. If the IP address is on

the black list, the connection is refused. If the IP address is on the white list, the connection is accepted.

If an IP address is on the white list, you can specify if a secure communication channel is required for that address. In most cases, it will be ok to allow local connections to run without encryption.

FrontBase can also run in a mode where it accepts only local connections. This may be useful when backing up a WebObjects or PHP powered web server, as it ensures that outsiders cannot connect directly to the database.

Transactions

The "Understanding Transactions" on page 68, section describes FrontBase's transaction support.

Row Level Priveleges

FrontBase offers a unique feature called "Row Level Privileges" on page 75, which allows you to specify access privileges for individual rows. Each row is said to be owned by a specific user and belonging to a specific group. Access privileges (SELECT, UPDATE, and DELETE) for a row can be specified for the owner, the group, and the world.

Live Backup

The "Backup and Restore" on page 72, section describes FrontBase's versioning system allows it to perform backups of live databases (i.e. while clients continue to access and modify the database).

Caching

The "Tuning FrontBase" on page 78, section describes FrontBase's caching schemes.

Multi-Server Deployment

This section describes FrontBase's replication and clustering features, which allow multi-server deployment for both redundancy and enhanced performance.

Drivers and Adaptors

FrontBase has drivers for ODBC and JDBC. It has adaptors for EOF, PHP, and Perl. This section gives an overview of these drivers and adaptors.

ODBC and JDBC

The ODBC and JDBC drivers provide general connectivity to FrontBase from applications which rely on these standards.

EOF adaptor

The EOF adaptor allows FrontBase to work with Apple's WebObjects.

PHP3 and PHP4 adaptors

The API for the PHP3 and PHP4 adaptors are based on the PHP3 and PHP4 adaptors for MySQL respectively. Functions that begin with mysql_ in the MySQL adaptors instead begin with fbsql_ in the FrontBase adaptors. This should make porting a PHP application to FrontBase fairly easy.

Perl adaptor

The Perl adaptor is a Perl Database Interface (DBI). This abstraction layer should make porting Perl application to FrontBase from other databases fairly easy.

Migration

FrontBase has tools for importing from FileMaker, MySQL, OpenBase, and Sybase.

FileMaker

"FileMaker" on page 83, migration is a two step process. The tables of a FileMaker database are exported from FileMaker. The exported files are moved to MacOS X, where an application imports them into FrontBase.

MySQL

The "MySQL" on page 84, migration tool uses JDBC to extract table data from a MySQL database and import it into a FrontBase database.

OpenBase

"OpenBase" on page 85, is supported directly through extensions to the SQL 92 INSERT command.

Sybase

"Sybase" on page 86, is supported directly through extensions to the SQL 92 INSERT command.

FrontBase-Related Processes

There are three main process that run in a FrontBase installation: FBExec, FrontBase, and FBWebEnabler. This section describes these processes, how to determine if they are running, and how to start them.

FBExec

FBExec acts as a broker between FrontBase databases running on your computer and client software running on your computer or over the network. One instance of FBExec should be running

When you install FrontBase, your computer will be set up so that FBExec is launched at startup. To make sure that FBExec is running on a UNIX-based installation, enter the following in a terminal session:

```
ps axc | grep FBExec
```

If FBExec is running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If FBExec is not running, you should start it as follows (assuming you have added FrontBase/bin to your $PATH):

```
FBExec &
```

Be sure to include the ampersand ("&") at the end so that the task doesn't end with your terminal session.

If, however, you are running FrontBase on Windows NT, you can use the Task Manager to check whether FBExec is running. If it is not running, go to the Service Manager and start it. FBExec should have been installed as a service so that it would start on system startup.

FrontBase

One instance of FrontBase will be running for each database that is running on your computer. Each FrontBase instance is started directly from the command-line (UNIX installations), by the Service Manager (Windows NT), or using the FBManager and FBWebManager tools.

The command-line options for FrontBase, which affect the behavior of a server during execution, follow. These options can also be set when using the "FBManager" on page 91, and "FBWebManager" on page 129, tools to start and create FrontBase databases.

FBWebEnabler

The FBWebEnabler process maintains persistant connections for the FBWebManager system, allowing you to perform administrative functions on your FrontBase server from any web browser, either locally or from another computer.

When you install FrontBase, your computer will be set up so that FBWebEnabler is launched at startup. If it stops running, you will likely find out when you attempt to access cgi-bin/FBWebManager through a web browser. "FBWebManager" on page 129, will then indicate that it cannot connect to FBWebEnabler.

If FBWebEnabler is not running, you should start it as follows (assuming you have added FrontBase/bin to your $PATH):

```
FBWebEnabler &
```

Be sure to include the ampersand ("&") at the end so that the task doesn't end with your terminal session.

If, however, you are running FrontBase on Windows NT, you can to the Service Manager and start FBWebEnabler. Like FBExec, FBWebEnabler should have been installed as a service so that it would start on system startup.

4

Installation

We offer a different installation of FrontBase for each supported server platform. Each installation contains the FrontBase server and the server administration tools and client libraries available for the platform. Complete documentation covering the FrontBase server, administration tools, and client libraries on all platforms is included with each installation.

Downloading

The following platforms are supported:

– "WebObjects" on page 30.
– "Linux" on page 41.
– "BSD" on page 57.

The FrontBase License String

The license string enables features in your FrontBase server. This article will help you obtain and install a free or paid license from FrontBase.com.

– "Obtaining License String" on page 61.
– "Installing License String" on page 62.

WebObjects

The Following Administrative Tools are provided:

- "FBManager" on page 91.
- "FBWebManager" on page 129.
- "sql92" on page 131.

The following Client Libraries are available:

- FBAccess
- FBCAccess
- EOF Adaptor
- PHP3/4, Perl
- ODBC
- JDBC 2.0

The following Operating Systems are supported:

- "MacOS X Server" on page 31.
- "MacOS X Public Beta" on page 33.
- "Windows NT" on page 36.
- "Solaris" on page 38.

MacOS X Server

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

3. Log into your MacOS X Server computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
4. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website. The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
5. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
6. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher):

```
gunzip FrontBase-MacOSX-2.18.pkg.tar.gz
gnutar xvf FrontBase-MacOSX-2.18.pkg.tar
```

   Use the Workspace Manager to navigate to the directory into which the downloaded file was installed. You should now see the Installer icon for the .pkg file, double-click on this icon and install the software by following the instructions provided by Installer.app.

By default, the installer will install FrontBase into the /Local/Library/FrontBase directory.

The installer will attempt to install FBWebManager files into appropriate directories for use by the Apache web server.

7. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /Local/Library/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

8. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should re-start those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

9. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

MacOS X Public Beta

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

1. Log into your MacOS X Public Beta computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher):

```
gnutar xvf FrontBase-MacOSX-2.18.pkg.tar
```

   Use the Workspace Manager to navigate to and start the installer application:

```
/System/Administration/Applications/InstallerX.app
```

Select the menu item **Package** and then select the sub-menu **Open**. Select the expanded file from above and click **Open**. Follow instructions provided by the installer application.

By default, the installer will install FrontBase into the /Local/Library/FrontBase directory.

The installer will attempt to install FBWebManager files into appropriate directories for use by the Apache web server.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /Local/Library/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should re-start those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

Windows NT

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: FBManager, FBAccess, and EOF Adaptor on Windows NT require Apple's WebObjects to be installed.

1. Log into your Windows NT computer as "administrator". FrontBase currently needs to be installed by administrator so that it can run like IIS and similar services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. Use your favorite zip file utility (such as PKZip) to unzip the downloaded archive. Unzipping will yield three executable (.exe) files. Run each of them and follow the installation instructions provided.

   FrontBase will be installed into the <drive>:/usr/FrontBase directory, where <drive>: is the drive onto which you've installed FrontBase.

   Windows NT specific components will be installed into the <drive>:/Program Files/FrontBase Tools and <drive>:/Apple/Library/Frameworks directories.
4. By default, FrontBase expects to be installed on the C: drive. If you have installed it on another drive (e.g. F:), you'll need to add an environment variable to NT.

   Go to Start —> Settings —> Control Panel, double-click the System icon, and add the FB_HOME_DRIVE environment variable, setting it to the letter of the drive onto which you installed FrontBase (e.g. F:).

5. You then need to define the FBExec as an NT service. Bring up a shell (e.g. a Bourne or DOS shell) and enter the following command:

```
<drive>:/usr/FrontBase/bin/FBExec -install
```

   Start the FBExec service using the Service Control Manager (Start —> Settings —> Control Panel, double-click the Services icon). Using the service Control Manager, you can also specify that the FBExec service should be started automatically whenever the computer is restarted.

   You can verify that FBExec now is running by launching the "Windows NT Task Manager" (Ctrl-Alt-Del, click on Task Manager).

6. Adjusting the search path to include the <drive>:/usr/FrontBase/bin directory will make your life on the command-line simpler. Use Windows Explorer to navigate to and open the ??? whatever ?? file with a text editor. Add the following line at the end of the file:

```
(path variable command here...)
```

7. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

Solaris

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: FBManager is not available on Solaris. FBAccess and EOF Adaptor on Solaris require Apple's WebObjects to be installed. FrontBase for Solaris has been generated using SunOS 5.8. For other versions, please contact us at info@frontbase.com

1. Log into your Solaris computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher):

```
mkdir FrontBase-2.18
mv FrontBase-2.18.tar FrontBase-2.18
```

```
cd FrontBase-2.18
tar xvf FrontBase-2.18.tar
sh install.sh
```

5. By default, the script will install FrontBase into the /opt/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.
6. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /opt/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

7. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should restart those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

8. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
9. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

Linux

The Following Administrative Tools are provided:

- “FBWebManager” on page 129.
- “sql92” on page 131.

The following Client Libraries are available:

- FBCAccess
- PHP3/4, Perl
- ODBC
- JDBC 2.0

The following Operating Systems are supported:

- “RedHat 6.x Linux (x86)” on page 42.
- “SuSE Linux (x86)” on page 45.
- “YellowDog Linux (PPC)” on page 48.
- “Debian Linux (x86)” on page 51.
- “Mandrake Linux (x86)” on page 54.
- If you are interested in FrontBase for Linux for IBM S390, please contact us at info@frontbase.com.

RedHat 6.x Linux (x86)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: RedHat-based Linux installs simply with RPM.

1. Log into your RedHat Linux computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to Basic Administration if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher). If you are installing FrontBase for the first time:

```
rpm -i FrontBase-2.18.rpm
```

If you are updating FrontBase, use:

```
rpm -U FrontBase-2.18.rpm
```

By default, the script will install FrontBase into the /usr/local/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /usr/local/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should restart those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

SuSE Linux (x86)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: SuSE Linux installs simply with RPM.

1. Log into your SuSE Linux computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to Basic Administration if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher). If you are installing FrontBase for the first time:

```
rpm -i FrontBase-2.18.rpm
```

If you are updating FrontBase, use:

```
rpm -U FrontBase-2.18.rpm
```

By default, the script will install FrontBase into the /opt/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /opt/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should re-start those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

YellowDog Linux (PPC)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: YellowDog Linux installs simply with RPM.

1. Log into your YellowDog Linux computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher). If you are installing FrontBase for the first time:

```
rpm -i FrontBase-2.18.rpm
```

If you are updating FrontBase, use:

```
rpm -U FrontBase-2.18.rpm
```

By default, the script will install FrontBase into the /opt/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /opt/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should restart those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

Debian Linux (x86)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: Debian Linux installs with a shell script.

1. Log into your Debian Linux computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher):

```
tar xvf FrontBase-2.18.tar
cd FrontBase-2.18
sh install.sh
```

By default, the script will install FrontBase into the /usr/lib/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /usr/lib/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should restart those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.

8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer.

You should now spend some time reading the documentation which accompanies your FrontBase server.

Mandrake Linux (x86)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print this section before continuing.

NOTE: Mandrake Linux installs simply with RPM.

1. Log into your Mandrake Linux computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to Basic Administration if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.
4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher). If you are installing FrontBase for the first time:

```
rpm -i FrontBase-2.18.rpm
```

If you are updating FrontBase, use:

```
rpm -U FrontBase-2.18.rpm
```

By default, the script will install FrontBase into the /usr/local/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.

5. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

If one or both are not running, try to launch them from the command-line:

```
cd /usr/local/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

6. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP, etc.) running prior to starting your upgrade, you should restart those now.

   First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

7. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
8. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

BSD

The Following Administrative Tools are provided:

- “FBWebManager” on page 129.
- “sql92” on page 131.

The following Client Libraries are available:

- FBCAccess
- PHP3/4, Perl
- ODBC
- JDBC 2.0

The following Operating Systems are supported:

- “FreeBSD (x86)” on page 58.

FreeBSD (x86)

Following the instructions below, you should be able to install FrontBase in about 10 minutes. After installing FrontBase, you can read about your new FrontBase server. You may find it useful to print section before continuing.

NOTE: FreeBSD installs with shell script and requires some manual configuration.

NOTE: FrontBase for FreeBSD has been generated using FreeBSD 3.4 and requires compat3x libraries for use on FreeBSD 4.x.

1. Log into your FreeBSD computer as "root". FrontBase currently needs to be installed by root so that it can run like Apache and other root level services. It makes no use of special ports nor does other things that may cause security concerns.
2. To download FrontBase, please visit www.frontbase.com/download.html of the FrontBase website.

   The archive weighs in at less than 3 MB and should require only a few minutes to download on a 56K modem.
3. If you have a previous version of FrontBase running on your server, the install script (which you'll run in the next step) will ask whether you wish to stop FrontBase related processes. You should usually let the script stop these processes for you. Refer to "Administration Tasks" on page 59, if you wish to stop things manually.

   If you are also running client software (such as WebObjects, PHP, etc.), you should disable it so it does not access FrontBase during the upgrade process. Some client software may detect that FrontBase is not running and attempt to restart it while you are upgrading! Your installation should only be offline for a few minutes while you upgrade.

4. From the terminal, expand FrontBase and run the installation script as follows (note that the version number may be higher):

```
tar xvf FrontBase-2.18.tar
cd FrontBase-2.18
sh install.sh
```

5. By default, the script will install FrontBase into the /usr/local/FrontBase directory. The script will attempt to install FBWebManager related files into the relevant Apache directories.
6. Verify that FrontBase has been successfully installed and started by entering the following in a terminal window:

```
ps axc | grep FB
```

If the FBExec process (a key FrontBase component) and the FBWebEnabler process (web admintstration tool process) are running, the system should reply with something like:

```
374 ? S 0:00 FBExec
375 ? S 0:00 FBWebEnabler
```

7. If one or both are not running, try to launch them from the command-line:

```
cd /usr/local/FrontBase
```

(deafault install location for FrontBase)

```
bin/FBExec &
bin/FBWebEnabler &
```

If launching FBExec or FBWebEnabler results in an error, please send e-mail to install-support@FrontBase.com. We will be happy to help you.

8. If you are upgrading from a previous version of FrontBase and had databases or client software (e.g. WebObjects, PHP,

etc.) running prior to starting your upgrade, you should restart those now.

First, start your databases as described in "Administration Tasks" on page 59. Then restart your client software.

9. Adjusting the search path to include the FrontBase/bin directory will make your life on the command-line simpler.
10. Congratulations, you have successfully downloaded and installed FrontBase! You do not need to restart your computer. You should now spend some time reading the documentation which accompanies your FrontBase server.

Obtaining License String

You can obtain a license string from the FrontBase.com website by supplying the IP address of the server and your e-mail address. A "license string" and "license check" for your server will be sent to you via e-mail.

Obtain a Free License

From the FrontBase.com website:

www.frontbase.com/cgi-bin/WebObjects/FrontBaseLicense.woa/wa/freeLicense

NOTE: The free license **does not** support backup and restore and is not to be used for deployment.

Purchase a License

From the FrontBase.com website:

www.frontbase.com/cgi-bin/WebObjects/FrontBaseLicense.woa/wa/store.

You can also apply for a free developer license (an "Enterprise" license without deployment rights) through this link.

Installing License String

There are three ways to install the license string:

1. If you are running FrontBase on a platform that includes FBDatabaseManager, you can install it from there. Choose **License**... from the **Edit** menu, select the server, and paste the license string and license check which were e-mailed to you.
2. If you have FBWebManager running on the server, connect to it with your web browser and click **License** in the left column. Paste the license string and license check which were e-mailed to you.
3. Although not highly recommended, you can create the license file yourself using a text editor. The file is called **LicenseString** and resides in the main **FrontBase** directory. It's format is as follows:

```
<license string>:<license check>
```

NOTE: The colon (:) between the license string and license check. The file is just one line – there is no carriage return after the license check.

5

Administration

FrontBase is designed to be "zero-maintenance", and for many applications, starting up a database may be all you need to do. More advanced or critical applications may involve managing user, backing up, keeping your installation current, tuning, and other tasks. These tasks are designed to be simple and straight-forward. This section explains how to do them.

This chapter contain the following sections:

Directory Structure

The FrontBase directory contains all the files required for your FrontBase installation. The files are subdirectories of the FrontBase directory are explained in this section.

Platforms

The location of the FrontBase directory depends on the platform on which you install FrontBase

Platform	Path
"MacOS X Server" on page 31.	/Local/Library/FrontBase
"MacOS X Public Beta" on page 33.	/Local/Library/FrontBase
"Windows NT" on page 36.	<drive>:/usr/local/FrontBase
"Solaris" on page 38.	/opt/FrontBase
"RedHat 6.x Linux (x86)" on page 42.	/usr/local/FrontBase
"SuSE Linux (x86)" on page 45.	/opt/FrontBase
"YellowDog Linux (PPC)" on page 48.	/opt/FrontBase
"Debian Linux (x86)" on page 51.	/usr/lib/FrontBase
"Mandrake Linux (x86)" on page 54.	/usr/local/FrontBase
"FreeBSD (x86)" on page 58.	/usr/local/FrontBase

A future version of FrontBase and its platform-specific installers will allow you to install FrontBase anywhere. Currently, the FrontBase directory needs to be owned by "root" or "administrator", but that mostly artificial requirement should also disappear in a future version.

Contents of the FrontBase directory

The FrontBase directory will normally contain the following subdirectories and files:

1. **Databases** is a directory which holds all databases served from the host. For each database, there are two files: <database-name>.fb and <database-name>.fb.log. The former contains the actual data in the database, while the latter contains miscellaneous status and error messages. You can delete a database by deleting the <database-name>.fb file. Of course, you can also log into the database as _SYSTEM and issue a DELETE DATABASE command. On Windows NT, you should make sure that you've removed the database as a service before deleting it.
2. **Collations** is a directory which holds all defined collations (orderings of Unicode characters which can be instantiated as COLLATIONs in your SCHEMAs).
3. **Translations** is a directory which holds all defined translations.
4. **bin** is a directory which contains FrontBase executables. You may want to add the FrontBase/bin directory to your $PATH environment variable for easy access.
5. **FBExec.log** is a file to which the FBExec process appends miscellaneous status and error messages.
6. **.BootstrapFiles** is a directory containing miscellaneous files required to create a new database.

Administration Tools

This section introduces the three primary tools for administering your FrontBase server: the "sql92" command-line tool, the web-based "FBWebManager" tool, and the "FBManager" application (for MacOS X and Windows NT only).

The following Tools are available:

– "FBManager" on page 66.
– "FBWebManager" on page 66.
– "sql92" on page 66.

FBManager

"FBManager" on page 91, introduces the application that lets you monitor and administer local and remote database servers. It is available for MacOS X and Windows installations of FrontBase.

FBWebManager

"FBManager" on page 91, introduces the FBWebManager web-based application, which is available for all installations of FrontBase and is accessed through any browser that supports dynamic HTML (aka JavaScript).

sql92

"sql92" on page 131, introduces the sql92 command-line tool, which is available for all installations of FrontBase.

Removing FrontBase

Should you ever need to completely remove FrontBase from your computer, this section describes how to do that.

Understanding Transactions

This section describes the database concepts of transactions, isolation levels, locking discipline, and updatability as implemented by FrontBase to manage simultaneous access.

Simultaneous access

One of the basic features of a database server is to provide users with parallel access to shared data. The database server must ensure that updates made to a database are performed in an orderly manner so that data is not corrupted or lost.

Transactions

A transaction is used to control users access to the database. A user cannot access the database without a transaction, and all operations are performed in the context of a transaction. All changes made to the database by a user in the context of a transaction are made visible to other users when the transaction is committed. A transaction is, as seen from the outside, a single atomic operation.

During its existence a transaction may fail. When a transaction fails, it may not be committed. The only course of action is to start all over again with the hope that the transaction will not fail the next time around. A database server can, in principle, fail transactions at will, but a good server will only fail a transaction for a good reason. The only good reason is a deadlock.

When a transaction is created it is assigned an isolation level, an updateability, and a locking discipline. The isolation level determines how isolated a transaction is from other transactions. Updateability determines if the access is read only or read write. The locking discipline determines the type of lock used to synchronize access to database.

Updatability

The updatability can be READ ONLY or READ WRITE. A transaction which has the updatability of READ ONLY cannot modify the database.

Isolation level

SQL92 defines 4 isolation levels:

- READ UNCOMMITTED
- READ COMMITTED
- REPEATABLE READ
- SERIALIZABLE

FrontBase defines one more isolation level:

- VERSIONED

Users accessing data in the database may experience the following phenomena:

Dirty reads

One transaction is writing some data to the database. The second is then reading that data, but the first rolls the transaction back. The second transaction has now read data that did not really exist.

Non-repeatable read

A transaction reads a row. A second transaction updates the values of the row and does a COMMIT. If the first transaction reads the row again it will get a different result.

Phantom

One transaction select some data in the database. A second transaction updates or inserts rows that satisfy the predicates that the first transaction used. The second transaction is committed. If the first transaction performs the select again, it will get a different result.

	Dirty Reads	Non-Repeatable	Phantom
READ UNCOMMITTED	Yes	Yes	Yes
READ COMMITTED	No	Yes	Yes
REPEATABLE READ	No	No	Yes
SERIALIZABLE	No	No	No
VERSIONED	No	No	No

The amount of data that is locked is reflected by the isolation level. With READ UNCOMMITTED, nothing is locked. With READ COMMITTED, nothing is locked, but only data that is committed is read. REPEATABLE READ locks rows as they are selected. SERIALIZABLE locks the whole table. When you use FrontBase, READ UNCOMMITTED is upgraded to READ COMMITTED.

The VERSIONED isolation level is only valid for READ ONLY transactions and will keep the current version of the database for the duration of the transaction. Other transactions may modify the database, but the changes will not be visible to the VERSIONED transaction. Any number of VERSIONED transactions can be ongoing at the same time, sharing committed versions of the database.

Locking Discipline

FrontBase introduces the concept of locking discipline. The locking discipline may have the following values:

1. **PESSIMISTIC** locking assumes that the given object will be changed. A transaction must wait until the object is available (unlocked). When a transaction is waiting there is a possibility for deadlocks. Deadlocks are detected and broken by failing one of the transactions causing the dead lock.
2. **OPTIMISTIC** locking assumes that a given object isn't changed by other transactions, and any changes are performed without delay. When the transaction is committed, FrontBase checks that the accessed objects weren't changed

during the the transaction. If they were changed, the commit fails.

3. **DEFERRED** (sometimes called "upgradable") is a version of PESSIMISTIC locking which assumes that objects are only read. Initially, the lock is a read lock and if the object is updated, the lock is upgraded to a write lock.

Backup and Restore

FrontBase allows you to backup a complete database to flat files. It allows you to restore content data from ASCII flat files directly into your database. FrontBase can even backup a "live" database.

You should backup critical databases on a regular basis to protect from loss due to hardware failures, lost files, or other catastrophes. You will also need to backup when moving your FrontBase databases to a new server or when sending them to FrontBase for diagnosis.

This section discusses the following:

– "Exporting Schema and Content Data" on page 72.
– "Importing Content Data from Flat Files" on page 73.

Exporting Schema and Content Data

FrontBase allows you to export a complete database to flat files. The SQL statement has the following syntax:

```
WRITE ALL OUTPUT('<output-directory>' [,'YES']);
```

All generated files will be saved in the specified <output-directory> The file <output-directory> / schema.sql contains SQL statements necessary to recreate the database. If 'YES' argument is supplied, a file for each base table will be generated as well. Each base table file holds the content data (rows) for the table. The <output-directory> / schema.sql file will include the necessary import statements.

Note that it is important to capitalize 'YES' if that option is used. This should be corrected in a future version of FrontBase.

A flat file holding the content data for a given table has the following lay-out:

Line 1: <schema name>.<table name>

Line 2: Column count

Line 3: <column #> <column name>,"~"

Line 4: ...

Line n: <column #> <column name>,"~\n"

-- n-2 == column count

Line n+1: <row data>

Line n+2: -"-

Example:

A database with a single user, FB, and a single table , T0, exported as:

```
WRITE ALL * OUTPUT('/tmp/db5', 'YES');
/tmp/db5/schema.sql:
CREATE USER "FB";
CREATE SCHEMA "FB" AUTHORIZATION "FB";
CREATE TABLE "T0" (

"C0" INTEGER, "C1" INTEGER);
COMMIT;
SET SESSION AUTHORIZATION "FB";
INSERT INTO "FB"."T0" FROM 'FrontBase' INPUT('/tmp/db5/2_13');
COMMIT;
/tmp/db5/2_13:
FB.T0
2
  1 "C0","~"
  2 "C1","~\n"
1~2~
3~4~
```

Importing Content Data from Flat Files

FrontBase allows you to import content data from ASCII flat-files directly into a database. Use the following SQL statement:

INSERT INTO SCHEMA <schema-name>

FROM 'FrontBase'

INPUT('<path name of file with data>');

The format of the file must be as if generated by the export functionality of FrontBase.

A common issue for all import formats is that the schema must be properly defined before the import will work.

WARNING! The import filter is very unforgiving about incorrectly formatted input. Please experiment with a test database before working with a production database!

Row Level Privileges

FrontBase offers a unique feature called Row Level Privileges, which allows you to specify access privileges for individual rows. Each row is said to be owned by a specific user and belonging to a specific group. Access privileges (SELECT, UPDATE, and DELETE) for a row can be specified for the owner, the group, and the world.

Invoking Row Level Privileges

To use the Row Level Privileges feature, a given database has to be initialized with the feature given as an option. Start a FrontBase from the command line as follows:

```
FrontBase/bin/FrontBase -rlpriv <database name>
```

You can also specify the -rlpriv option when creating a database with the FBManager and FBWebManager tools.

Once invoked, the option is recorded in the database. You don't need to specify the option when the database server is subsequently stopped and started.

Managing the meta data

You can create and drop groups when CURRENT_USER is _SYSTEM using these SQL statements:

```
CREATE GROUP <group-name>;
DROP GROUP <group-name> RESTRICT|CASCADE;
```

You can add and drop users to a group when CURRENT_USER is _SYSTEM using these SQL statements:

```
ALTER GROUP <group-name> ADD USER <user-name>;
ALTER GROUP <group-name> DROP USER <user-name>;
```

You can change the default group of a user when CURRENT_USER is either _SYSTEM or <user-name> using this SQL statement:

```
ALTER USER <user-name> SET DEFAULT GROUP <group-name>;
```

You can change the default row privileges for a table when CURRENT_USER is either _SYSTEMor <user-name> using this SQL statement:

```
ALTER TABLE <table-name> SET DEFAULT PRIVILEGES(<row-privileges>)
    [USER <user-name>];
```

If <user-name> is not given, the current user is assumed. The <row privileges> argument has the following BNF form:

```
<row privileges> ::= <row privs> | <row privileges> , <row privs>
<row privs>      ::= <owner privs> | <group privs> | <world privs>
<user privs>     ::= USER = * | <priv mask>
<group privs>    ::= GROUP = * | <priv mask>
<world privs>    ::= * = * | <priv mask>
<priv mask>      ::= <priv> | <priv mask> + <priv>
<priv>           ::= SELECT | UPDATE | DELETE
```

Example:

```
ALTER TABLE T0
    SET DEFAULT PRIVILEGES(USER=*, GROUP=SELECT+UPDATE, *=SELECT);
```

Managing the content data

You can change the privileges, owning group, or owning user of rows of a table using the UPDATE SQL statement. CURRENT_USER must either be _SYSTEM or own the affected rows.

```
UPDATE <table-name> SET PRIVILEGES (<row-privileges>)
    [WHERE <cond-expr>];
UPDATE <table-name> SET GROUP <group-name>
    [WHERE <cond-expr>];
```

```
UPDATE <table-name> SET USER <user-name>
    [WHERE <cond-expr>];
```

SELECTing the access privileges for a row

The owner, group, and privileges for a given set of rows can be fetched with this SQL statement:

```
SELECT USER, GROUP, PRIVILEGES FROM <table>
    WHERE <cond-expr>;
```

By wrapping the SELECT in a VIEW, the values can be used in queries:

```
CREATE VIEW(ROW_OWNER, ROW_GROUP, ROW_PRIVS) T0_PRIVS
    SELECT USER, GROUP, PRIVILEGES FROM T0;

SELECT * FROM T0_PRIVS WHERE ROW_OWNER = '<user-name>';
```

Tuning FrontBase

As your FrontBase database gets large or your performance demands increase, you can tune FrontBase to improve overall and specific-query performance. This section describes how to tune FrontBase using the Raw Device Driver feature (overall performance) as well as table and index caches (specific query performance).

Database server performance

Database server performance depends upon three things: CPU, RAM, and disk subsystem speed.

When your database is small (i.e. up to a hundred thousand rows per table) the performance is mainly dependent on the CPU speed: how fast the server can move data around. As your database grows larger, less of it fits into memory. Many databases are too large to fit in RAM at all times. Caches, caching strategies, and caching options greatly affect perceived performance. Eventually, your database becomes many times larger than the amount of available RAM. At that point, database server speed becomes most dependent on disk speed: how fast the server can get to the data.

Quite often, rather than upgrading your machine, you can install more RAM and upgrade the disk subsystem. You can also give the database server hints about how to cache data so that it can use the available RAM more effectively.

FrontBase's caching mechanisms

FrontBase offers an elaborate caching strategy with two major components accessible to and tunable by the database developer: table caching and the raw device driver (or global cache). The crucial quality of FrontBase caches is that the integrity offered by transactions is maintained. When data in the cache is updated, the same data is also written to the disk upon executing a COMMIT. The COMMIT succeeds only after the data is successfully written to disk.

When should caching be tuned?

If hitting the database server continually with SELECTs makes the CPU utilization percentage drop significantly, you should tune the caching mechanisms. In this case, the database server is waiting for the disk subsystem to deliver the data. You may be able to increase performance simply by adding RAM to the machine, or may need to tune caching.

FrontBase offers the following caching:

– "Table Caching" on page 79.
– "Raw Device Driver (RDD)" on page 80.

Table Caching

Table caching allows the developer or administrator to adjust how many of the rows of a given table that should be cached:

- Min. row count, the minimum number of rows to be cached.
- Max. row count, the maximum number or rows to be cached.
- Percentage, the percentage of the total row count to be cached.
- Persistent, keep the cache across transactions.
- Preload, cache the table the first time the table is referenced.

The Persistent setting is available (and not always ON) because in some scenarios, you may wish to control the actual memory usage while allowing certain bursts of memory usage to occur. An example is report generation, where each transaction can span many SELECTs and will likely reference some tables that normally aren't in use that much. If Persistent is set to OFF for such tables, the cache will get loaded as used and then flushed when the transaction is COMMITted.

When should table caching be used?

Table caching is typically used on smaller "feeder tables" which can be cached 50% to 100% without an explosion in memory usage.

Adjusting table cache settings

Using the sql92 command-line tool, you can adjust table caches using the following extensions to SQL 92 provided by FrontBase. You can also adjust table cache settings using the FBManager and FBWebManager tools.

```
ALTER TABLE <table name> SET CACHE ([<lower>], [<upper>], [<percent>]);
```

Sets or adjusts the cache parameters for the given table. <lower> and <upper> are given as absolute row counts. The server will keep a minimum of <lower> rows and a maximum of <upper> rows in memory. Defaults are 2000 for <lower> and 20,000 for <upper>. The <percent> value tells the server to keep a varying number of rows in the cache, while still obeying the <upper> value. The default value for <percent> is 20.

```
ALTER TABLE <table name> SET CACHE PRESERVE FALSE | TRUE;
```

Instructs the server to maintain (TRUE) the cache for the given table even if there are no references to the table. FALSE means that the server will discard the cache when there are no more references to the given table.

```
ALTER TABLE <table name> SET CACHE PREPARE FALSE | TRUE;
```

Instructs the server to load (TRUE) the cache fully for the given table the first time the table is referenced. FALSE means that the cache is loaded as dictated by the actual use of the given table.

The above statements are transaction initiating. A COMMIT is required to make the changes visible to other users.

Raw Device Driver (RDD)

With today's fast computers, most performance issues in a database server are related to how fast you can get data off the hard disk. One way to increase the performance in this area, is to bypass the host OS filesystem. In FrontBase this is done through the deployment of

the Raw Device Driver (RDD) module. RDD even allows you to specify a raw partition to be used as datastore. Additionally, RDD allows you to specify the size of its combined write-through and read cache.

When should RDD be used?

RDD is typically used in combination with table caching so that smaller tables are cached 100%, while larger tables are cached via the RDD cache. An example is an indexing solution. You would typically have two smaller tables, WORDS and DOCUMENTS, with the third larger table being the relation table, HIT. With FrontBase, you would cache WORDS and DOCUMENTS 100%, while letting the RDD manage the HIT table. 100-300 MB of RDD cache memory might work well, depending on the size of the HIT table.

Adjusting RDD settings

RDD settings are specified when the database is created or started. Both the FBManager and FBWebManager let you specify RDD settings with their respective "Start Advanced" commands. From the command-line, you specify RDD settings when starting a FrontBase process.

Replication and Clustering

This article explains clustering and replication, which are useful for both redunancy (protect from downtime when a server becomes unavailable) and load distribution (increase response time in heavy load situations).

Migration

This section explains how to use FrontBase's import features to migrate databases from other vendors' database servers.

Currently FrontBase has mechanisms for importing from the following:

- "FileMaker" on page 83.
- "MySQL" on page 84.
- "OpenBase" on page 85.
- "Sybase" on page 86.
- Other import mechanisms will be added as demand arises.

FileMaker

The FileMaker Pro migration tool lets you move a FileMaker Pro database to FrontBase. It has two key restrictions. First, as FrontBase is a relational database server and not a front-end tool, only your FileMaker Pro tables will be migrated. Second, SQL 92 has no provision for calculated columns, so they can't be moved over to FrontBase if you have them in a FileMaker Pro database.

Currently, you'll need to download the FileMaker Pro migration tool separately. It is only available for the MacOS X platform. The tool is available from the downloads section of the FrontBase website. Eventually, it will be rolled into the FrontBase downloadable for every platform.

When you download and decompress the FileMaker Pro migration tool, you'll find the following items:

1. FM2FB.app

   This is the main application build for Mac OS X Server. It can be used right away, or it can be moved to one of the Application folders. The application requires the FrontBase client frameworks to run. These frameworks are installed when you install the FrontBase package.

2. MetaDumper.fp3

   This FileMaker script extracts information from FileMaker files. This script should be moved to a place where FileMaker can access it. It works with both MacOS and Windows version of FileMaker. In MacOS, you may not be able just to double-click the script file. Instead, start FileMaker and open it using the "Open..." menu item.

3. README

   An older "read me" document.

To access the documentation on how to use FM2FB and the MetaDumper script, start FM2FB.app and choose "FM2FB Help" from the "Help" menu. This will open the HTML documentation in your web browser. The HTML files are embedded in the application wrapper.

MySQL

The MySQL migration tool lets you move a MySQL database to FrontBase using JDBC drivers for both databases. It has one key limitation. It cannot handle MySQL with enum or set column types. The MySQL migration tool requires a Java virtual machine version 1.2 or higher.

Currently, you'll need to download the MySQL migration tool separately. The tool is available from the downloads section of the FrontBase website (www.frontbase.com). Eventually, it will be rolled into the FrontBase downloadable for every platform.

You will need to acquire a JDBC driver for MySQL. You can get one here. Simply install the MySQL JDBC driver in the directory containing the rest of the MySQL migration tool.

To run the MySQL migration tool, change your working directory to the directory containing the tool and execute the following from the command line:

```
java -cp mysql_2_comp.jar:frontbasejdbc.jar:MySQL2FB.jar MySQL2FB
```

It will ask you for some information to complete the migration:

Source

The source field is the location and name of the MySQL database to transfer. Input must have the following format: dbName@host.

Destination

The destination field is the desired location and name of the FrontBase database. There has to be a running version of FrontBase 2.0 on the specified host. Input must have the following format: dbName@host.

User name

The user name is the user name needed to log on to the MySQL database.

Password

The password is is the password for the MySQL user name.

OpenBase

FrontBase can directly import OpenBase databases which have been exported with the "Backup as ASCII..." feature of the OpenBaseManager application. Use the following SQL statement to import into FrontBase:

```
INSERT INTO SCHEMA <schema-name>
    FROM 'OpenBase'
    INPUT('<path name of file with backup data>');
```

A common issue for all import formats is that the schema must be properly defined before the import will work.

WARNING! The import filter is very unforgiving about incorrectly formatted input. Please experiment with a test database before working with a production database!

Sybase

FrontBase can directly import SyBase databases which have been exported with Sybase's bcp utility. Use the following SQL statement to import into FrontBase:

```
INSERT INTO <table-name>
    FROM 'Sybase'
    INPUT('<path name of file with content data>',
    '<path name of file with format data>');
```

A common issue for all import formats is that the schema must be properly defined before the import will work.

WARNING! The import filter is very unforgiving about incorrectly formatted input. Please experiment with a test database before working with a production database!

Indexing

FrontBase offers two strategies for maintaining indices:

Strategies

1. **PRESERVE SPACE**

   The default strategy, called PRESERVE SPACE, is very space efficient and works well with tables up to a few hundred thousand rows. An index on a table, no matter the number of colums, costs less than 5 bytes per row. If you have a table with 100,000 rows, creating an index on this table will thus increase the disk footprint by less than 500 KB. Memory efficeincy is attained since column values are not stored together with the index information (an optimized B-tree). The downside is that rows from the table must be loaded to get column values when using the index. This index mode works well in most cases. Users like the low disk space footprint of a database.

2. **PRESERVE TIME**

   The alternative strategy, called PRESERVE TIME is fast when searching through millions of rows at the expense of higher disk space consumption. The mode scales very well and can easily handle tables with many millions of rows. Column values are copied into the B-tree, which increases the disk space footprint, but speeds up lookups considerably. The actual rows are loaded only when needed. If a given SELECT only fetches column values that are part of an index, the actual rows are not loaded at all. Such a SELECT is very fast.

Example:

Consider a typical indexing setup with a WORD table, a DOCUMENT table and a RELATION table:

```
CREATE TABLE WORD(
    WORDPK INT PRIMARY KEY,   -- Implies an index
    WORD VARCHAR(64));

CREATE INDEX ON WORD(WORD);
```

```
CREATE TABLE DOCUMENT(
    DOCUMENTPK INT PRIMARY KEY,   -- Implies an index
    DOCUMENT CLOB);

CREATE TABLE RELATION(
    WORDFK INT,
    DOCUMENTFK INT,
    PRIMARY KEY(WORDFK, DOCUMENTFK));   -- Implies an index

CREATE INDEX ON RELATION(DOCUMENTFK, WORDFK);

COMMIT;
```

To get a list of DOCUMENTFKs identifying the documents in which a given word is found:

```
SELECT DOCUMENTFK
    FROM RELATION, WORD
    WHERE RELATION.WORDFK = WORD.WORDPK AND WORD.WORD = '<some-word>';
```

To get a list of WORDFKs identifying the words found in a given document:

```
SELECT WORDFK
    FROM RELATION
    WHERE DOCUMENTFK = <some-document-pk>;
```

To find a list of DOCUMENTFKs identifying the documents in which two given words are found:

```
SELECT DOCUMENTFK
    FROM RELATION, WORD
    WHERE RELATION.WORDFK = WORD.WORDPK AND WORD.WORD = '<word_1>'
    INTERSECT
    SELECT DOCUMENTFK
    FROM RELATION, WORD
    WHERE RELATION.WORDFK = WORD.WORDPK AND WORD.WORD = '<word_2>'
```

This could / should be wrapped into a VIEW.

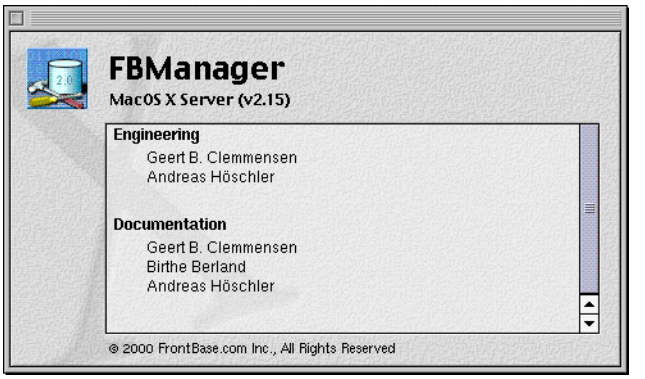
In a reasonable setup with 100,000 documents and 100 words on average per document, the RELATION table holds 10,000,000 rows and is a perfect candidate for PRESERVE TIME. In all of the above SELECT statements, the actual rows would not be loaded if the indices were set to PRESERVE TIME. The indices would hold the actual column values.

The WORD table is a less likely candidate for PRESERVE TIME. It will probably make more sense to use PRESERVE SPACE combined with a proper sized cache for this table.

6

FBManager

This chapter introduces the FBManager application, which is available for MacOS X and Windows installations of FrontBase. It is organized to give you a good overview of the features of the tool and how to accomplish common tasks with it



This chapter contains the following:

Monitoring Databases

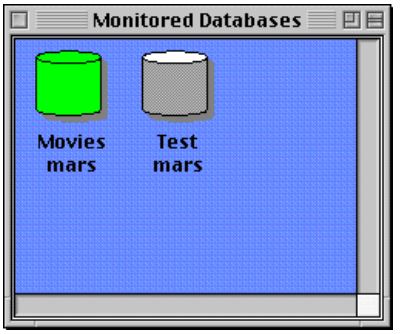
This section shows how to monitor servers and databases (perhaps on multiple servers) that interest you.

This section contain the following:

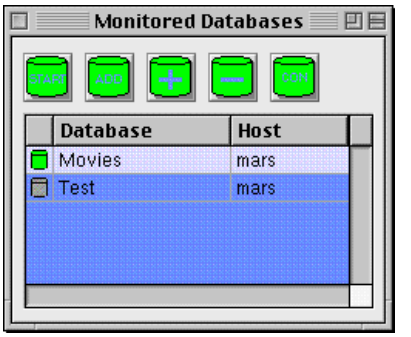
Starting FBManager

FBManager.app lets you monitor and control the state of FrontBase servers, i.e. whether they are running or stopped. When the FBManager.app is started for the first time by a given user, the following two windows appear:

The first window is called Icon View and shows monitored databases as icons indicating the state of the database together with the name of the database and the host computer on which the database is located. The example above shows the databases Movies and Test on host mars.

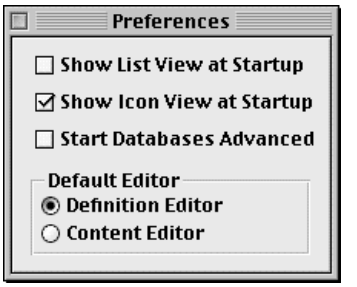


The second window is called List View and shows the same databases as rows in a tableview.



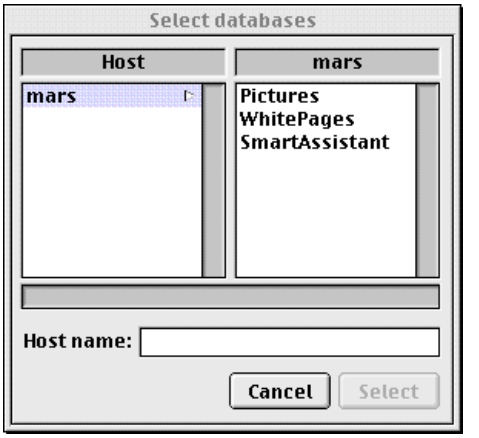
Preferences

Whether you use the icon view or the list view for administering databases is up to you. Which windows appear at application startup can be specified in the preferences panel.



You can open these windows at any time by clicking on the corresponding items in the File menu. To add existing databases to the monitor view(s), choose Database->Monitor... from the menu. The following panel appears.

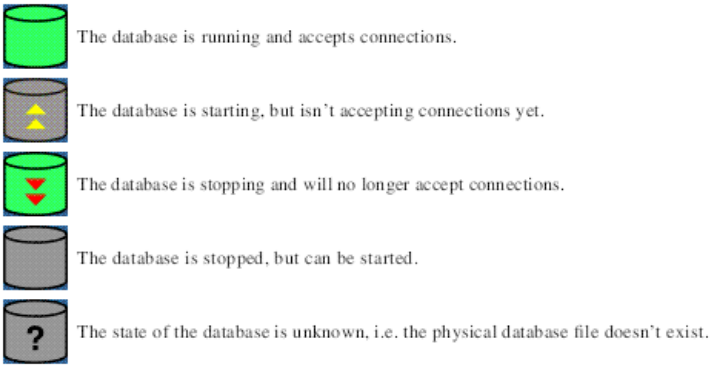
Monitor Panel



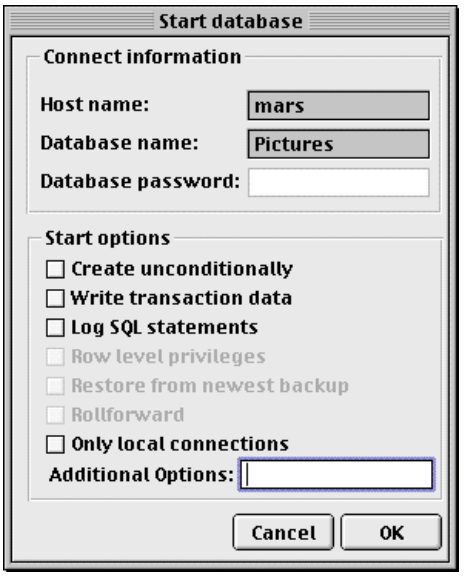
The first column of the browser shows all currently monitored hosts. The second column shows all existing but not yet monitored databases on the selected host. To monitor databases on a not yet shown host simply enter the hostname in the textfield below the browser and press Return. To add a database to the monitor view(s) select the database in the browser and click on Select. You can also add multiple database at once. You can remove a database from the monitor view(s) by selecting it and choosing Database->Demonitor from the menu.

Database States

A database can assume 5 different states:



To start a database that is not yet running select it in any monitor view and choose Database->Start or Database->Start Advanced from the menu. In the first case the database is simply started and the cylinder should become green within 1-2 seconds. In the second case the following panel appears allowing you to specify options before the database is started.

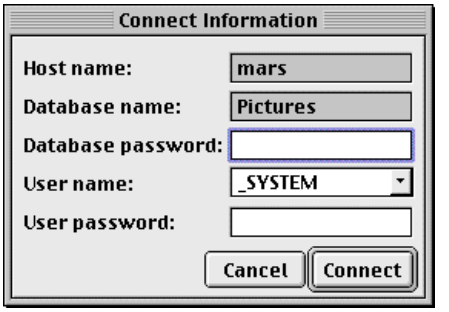


If you for example check the box next to Log SQL statements, all sql statements send to the database are logged in a file <database-name>.fb.log.

Connection Information

You can now connect to a database by either choosing Database->Connect from the menu or simply doubleclicking on a database icon. If you doubleclick on a not yet startet database, it is started automatically. Whether it is started in advanced mode - allowing you to specify options - or not depends on your selection in the preferences panel.

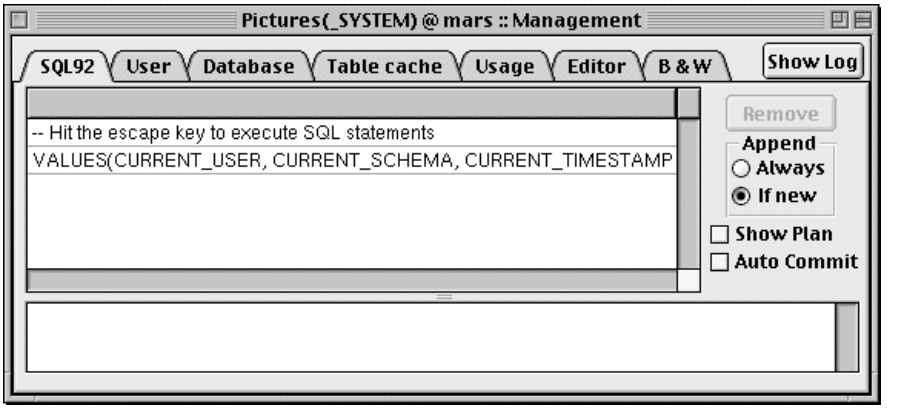
After making sure the database is running the following panel is raised.



Enter a username and a password and/or database pasword if any in the corresponding fields and click on "Connect". You can also select a user from a list of usernames that have been used at earlier connects.

Management Information

If you connect to the database for the first time (e.g. immediately after creating it), only superuser _SYSTEM is offered. If a connection can be established, the following management window appears:

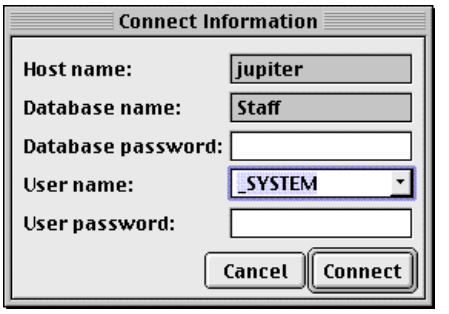


NOTE: The database, the user, and the host are shown in the title of the management window. You can have an arbitrary number of connections to the same database at once. Please note the "Show Log" button in the top right corner. A click on this button orders a log window to the front that shows all SQL statements that are sent to the database while working on the pages of the management window.

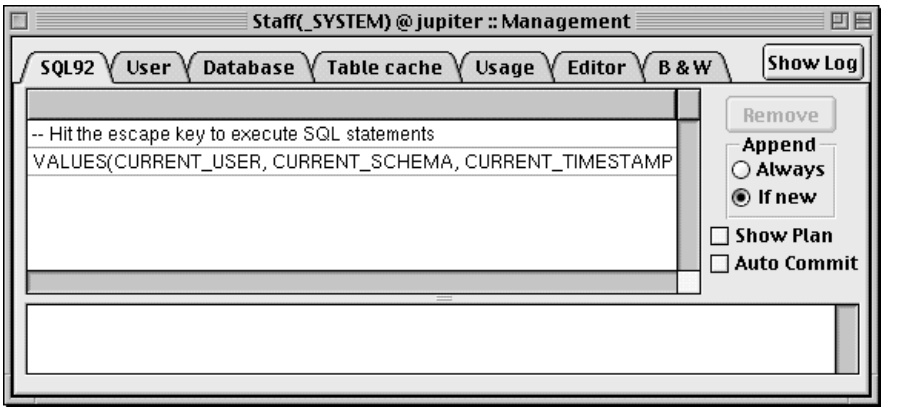
Managing Databases

This section explains how to use FBManager to work with databases, manage users, and perform other common tasks.

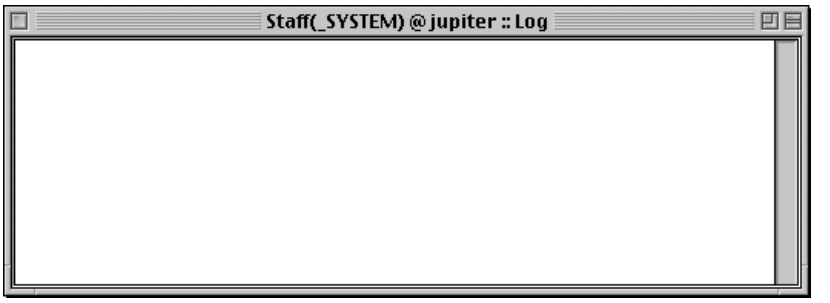
To enter the database management module, double-click on a running database in the monitor window. The following panel appears:



Enter the database password, if any, the user name and password, if any, and click on the "Connect" button. The user name is saved as a default and is automatically entered into the panel whenever appears. If a connection can be established, the following management window appears:



The window title has the format <database>(<user>)@<host>. A log window also appears:



SQL92

The SQL92 management option is an easy to do "command-line" SQL-based interaction with the server. Any SQL statement can be entered in the text field and by pressing the escape key, the following happens:

The SQL statement(s) are, if "seen" for the first time, inserted in the table view and send to the server for execution.

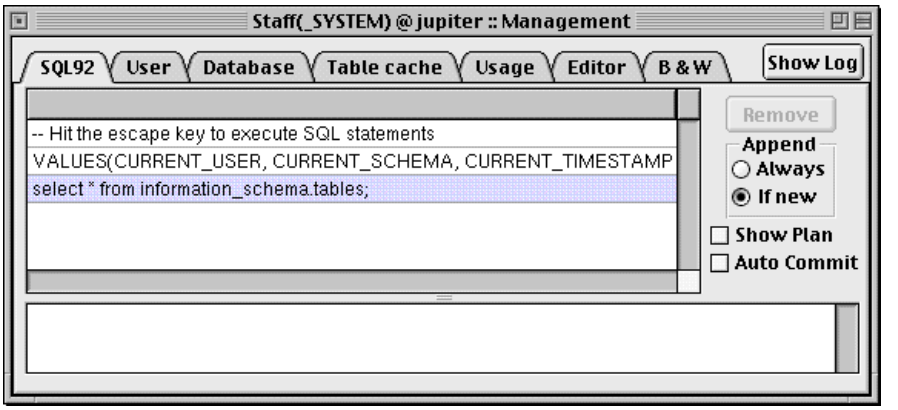
The SQL statement(s) are appended to the log view.

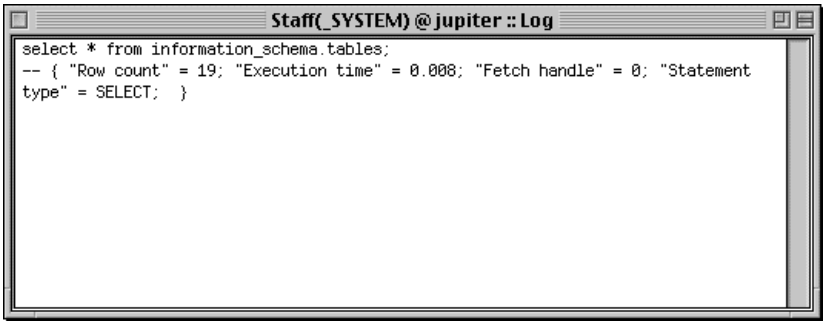
The execution result, inclunding any error messages, are appendend to the log view.

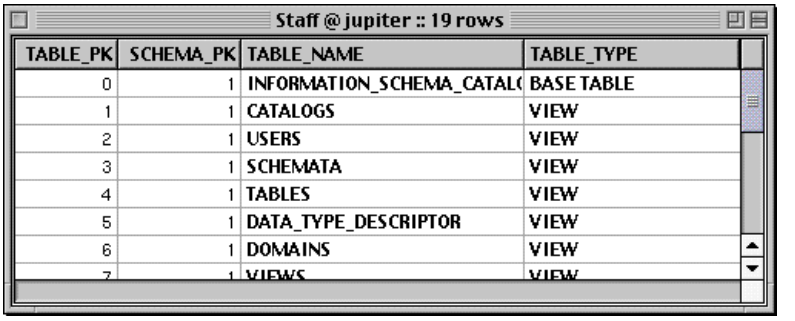
If the SQL statement is a SELECT, the corresponding table is fetched and displayed in a separate window.

Example:

After having entered "select * from information_schema.tables;" in the textview of the management window and pressed the escape key, the management window, log window and the result window look as follows:







By clicking on a row in the management window the corresponding statement(s), as previously entered, is inserted in the text field and can edited and executed again. By double-clicking on a row in the management window, the orresponding statement(s) is automatically executed again. Please examine the Management menu of the SQL92 page of the management window. It contains a menu item "Execute File...". If you click on this menu item a file browser is opened allowing you to select a file of SQL statements for execution as e.g. generated by Unload Database (see Management menu of the Database page).

Show plan

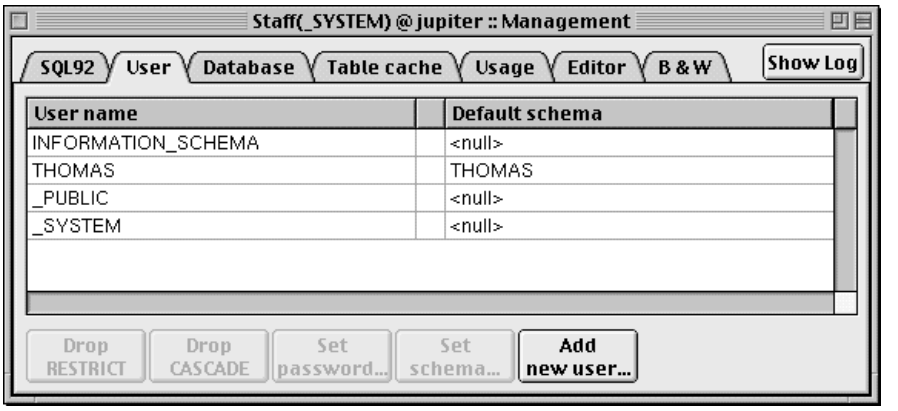
Check the box Show Plan next to the tableview on the SQL92 page and execute any select statement. The result window will now include not only the result of the fetch but also query plan information that can be used to decide whether indexes should be created for performance purposes.

Auto Commit

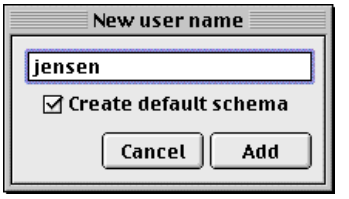
Checking the Auto Commit box next to the tableview on the SQL92 page executes an SQL statement that switches Auto Commit on. This disables transactions. All following statements will perform permanent modificatons in the database that canG0t be rolled back. Check the box again to switch auto commit off and enable transactions again.

Users

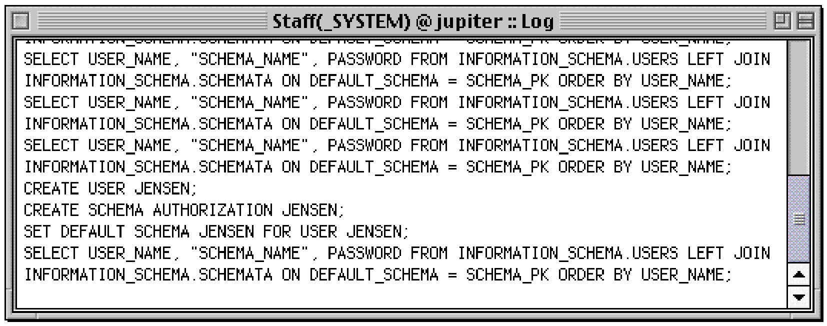
The user page is used to maintain the set of users:



To add a new user click on the "Add new user..." button. The following panel appears:

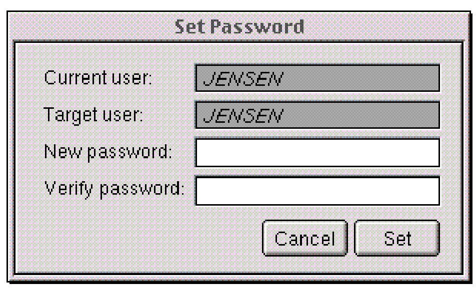


You need to be connected as user _SYSTEM to add new users. If the "Create default schema" switch is on, a schema with the same name as the user name is created. The corresponding SQL statements are shown in the log window:



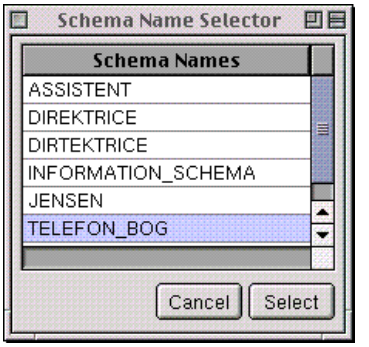
To delete a user name, you first have to select it in the table view and then click on the "Drop RESTRICT" or "Drop CASCADE" buttons. Dropping restricted will only succeed if the user doesn't own any schemas. Dropping cascaded will unconditionally drop all schemas owned by the user.

To set or change a password, select the user in the table view, click on the "Set password..." button and the following panel appears:



Enter the new password, verify the password, and click on the "Set" button. The user name table view is updated and now shows that a password has been set for the user:

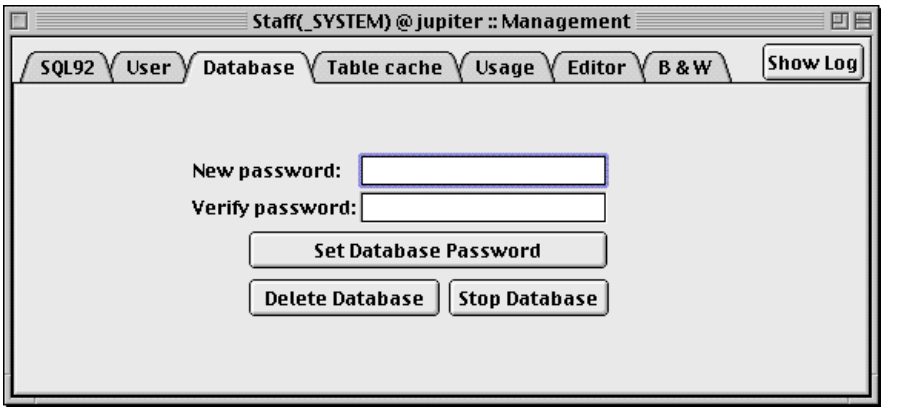
The default schema is the the schema that becomes the current schema for the user whenever a connection to the database is made. To set or change the deafult schema, click on the "Set schema..." button and the following panel appears:



Select a schema and click on the "Select" button, The user name table view is updated and now shows the new default schema for the user:

Database

The database page of the management window allows you to set a database password, and to stop or delete the database. These features are only enabled if you are logged in as _SYSTEM.



Enter the new password, verify the password, and click on the "Set Database Password" button. If you stop or delete the database, all concerned management windows will close automatically.

Please also examine the Management window of the Database page. It contains menu items for backing up and unloading databases.

BackUp

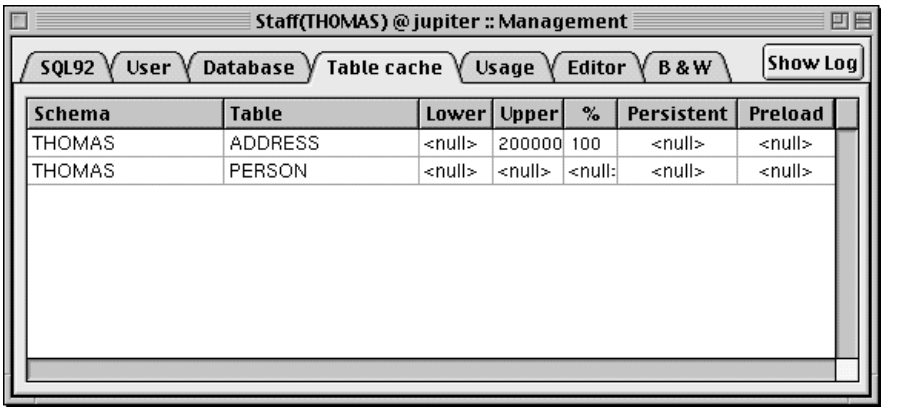
Backup Database initiates a backup of a live database. Once this operation has completed without errors, the backup is done and the corresponding file B_<yymmdd>_<hhmmss> can be found in /Local/Library/FrontBase/Backups/<database name>. The backup will latch onto the newest possible version of the database. To make sure that a rollforward (in connection with a restore operation) including ongoing transactions can succeed, you should click Yes in the dialog that is raised after choosing Backup Database... from the menu.

Unload Database

Unload Database writes the proper SQL statements to recreate the entire database into a file called schema.sql. This file will be created in the directory you specify in the dialog that gets raised when you choose Unload Database... from the menu. You can also specify whether the contents of all tables is to be written to flat files.

Table cache

This page can be used to finetune table caching.



All tables owned by the connected user are shown in the table view and the cache parameters for each table can be set. The parameters are:

1. Lower

   The minimum number of rows to be kept in the table cache.

2. Upper

   The maximum number of rows to be kept in the table cache.

3. %

   The percentage of total number of rows to be kept in table cache. Note that lower and upper take precedence over % if lower and upper are specified.

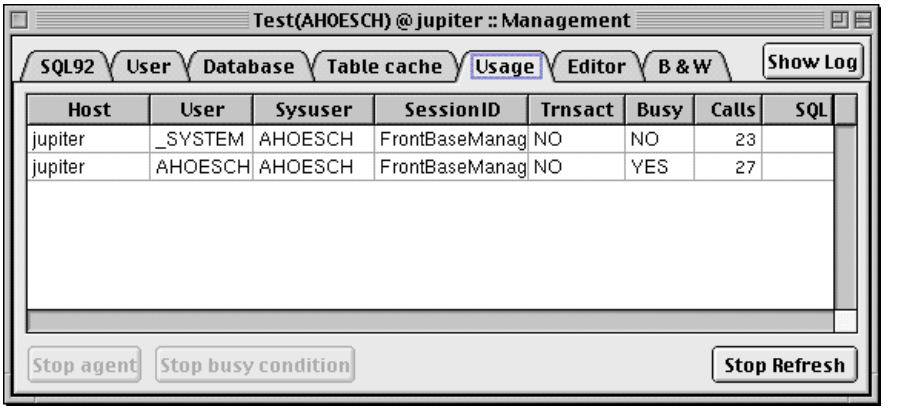
4. Persistent

5. If set to YES, the cache is kept across transactions, otherwise the cache is flushed after each COMMIT or ROLLBACK.
6. Preload

   If set to YES, all rows in the table will be loaded into the cache when the server is started, otherwise the rows are loaded upon the first reference to them.

Usage

This page gives you some usage information for the monitored databases, who is logged in to what database, how many transactions have been performed since starting the database, etc.

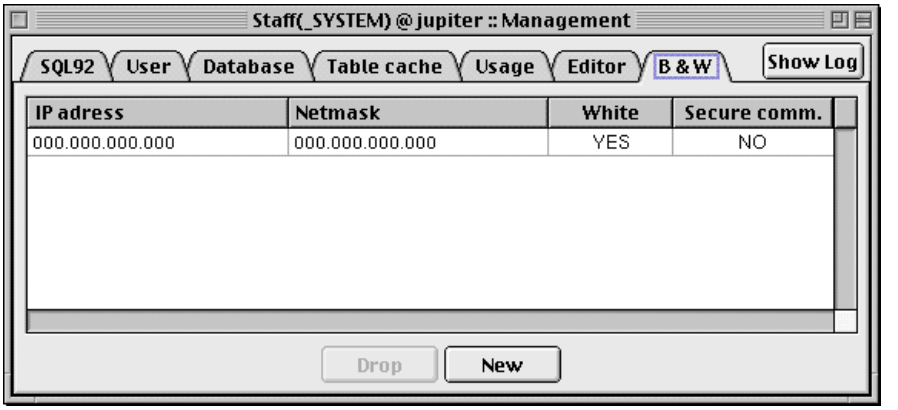


Please also note the two buttons in the lower left corner that get enabled if you are logged in as _SYSTEM and click on a row in the tableview. These two buttons allow you to close a connection to the database or stop a lengthy operation. If for example an application initiated a time taking operation (complex select statement on a very large table) and you want to break this operation, just select the corresponding row in the tableview and click on Stop busy condition.

White/Black Lists

When a client connects to a FrontBase server, its IP address is checked against the white and black list to see if the client is allowed to connect. For each entry in the white and black list, the IP address

of the client is ANDed with the netmask and the result is compared with the IP address in the white and black list. If a match is found and the White values is YES, the client is allowed to connect otherwise the connection is refused. If no match is found, the connection is refused. If a YES is entered in the secure field, the client is requested to uses encryption for all further communication with the server.



Editing Databases

This section explains how to use FBManager to edit databases – create tables, add columns, insert data, perform queries, etc.

Transaction Orientation

FrontBase is 100% transaction-oriented, which is your protection against corrupted data and unwanted conflicts stemming from multiple users updating the same tables at the same time. The deafult "transaction mode", which FBDatabaseEditor uses, is called SERIALIZABLE READ WRITE PESSIMISTIC. This means that the isolation level is SERIALIZABLE, the access mode is READ WRITE, and the locking discipline (a FrontBase extension to SQL 92) is PESSIMISTIC.

More generally, this transaction mode means that access to tables etc. is effectively serialized, i.e. if you want to access a particular table which is currently participating in another transaction, you will wait until that other transaction is completed.

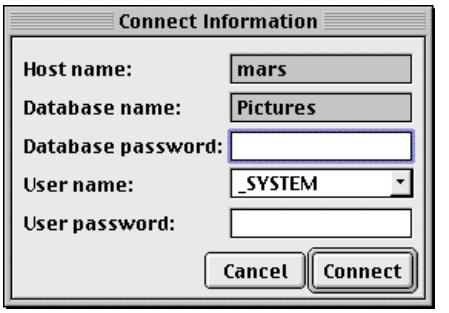
While the creation of a transaction is automatic, as defined by the SQL 92 standard, you will have to tell FrontBase when you want the transaction completed (COMMIT or ROLLBACK).

Whenever one or more windows show the edited state (icon in the upper left corner of a window), a transaction has been created. The COMMIT and ROLLBACK submenu items of the Transaction main menu item will also be enabled as soon as a transaction has been created.

As soon as you have completed the necessary modifications to your database, you should thus COMMIT or ROLLBACK the active transaction. Once a transaction has been successfully committed, the modifications are visible to other users of the database.

Connecting to a database

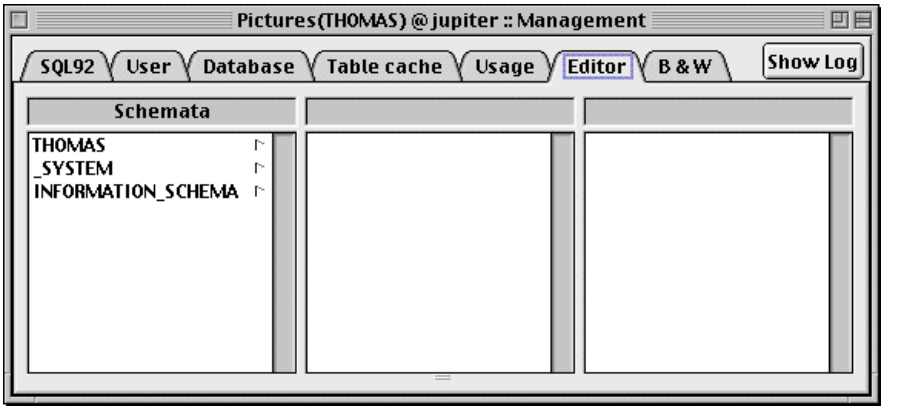
Connect to a database by doubleclicking on the corresponding icon or row in the monitor window (icon or list view). The following panel appears:



Click on the Connect button once the appropriate information has been entered. You can have open connections to several databases at the same time.

Database editor

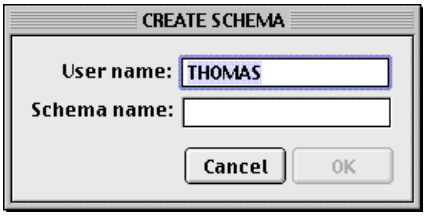
If a connection to the database could be established, a corresponding management window is ordered front. In the following we assume, that a user thomas with corresponding schema has already been created. After clicking on the Editor tab you should see something like the following.



The title of the management window indicates, that user thomas is connected to database Pictures on host jupiter. The first column of the browser is a list of the defined schemas in the database, including schema THOMAS that has been created automatically while creating user thomas. Please note that INFORMATION_SCHEMA cannot be edited via this application.

Creating a schema

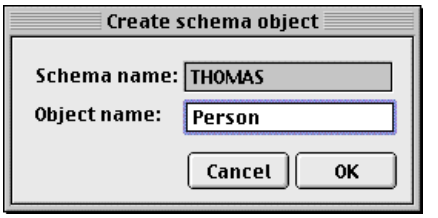
Objects like tables, views, collations,... are always created in and owned by a so called schema. While creating user thomas a schema thomas has automatically been created. However, you can also create additional schemas on your own. Choose Management->Create->Schema from the menu. The following window appears:



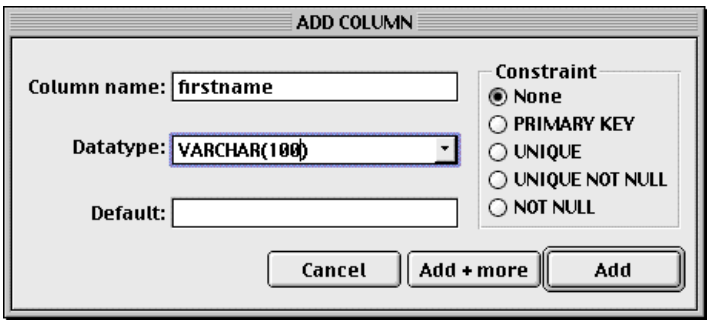
The user name is important because this is the user which will be the owner of the new schema, i.e. only this user will be able to modify the schema. Enter a name for the new schema and click on the OK button. If the OK button doesn't become enabled after you have typed a schema name, it is because a schema with the specified name already exists.

Creating a table

Select any schema owned by the connected user (e.g. thomas) by clicking on it in the editor browser and choose Management->Create->Table from the menu. The following panel appears:

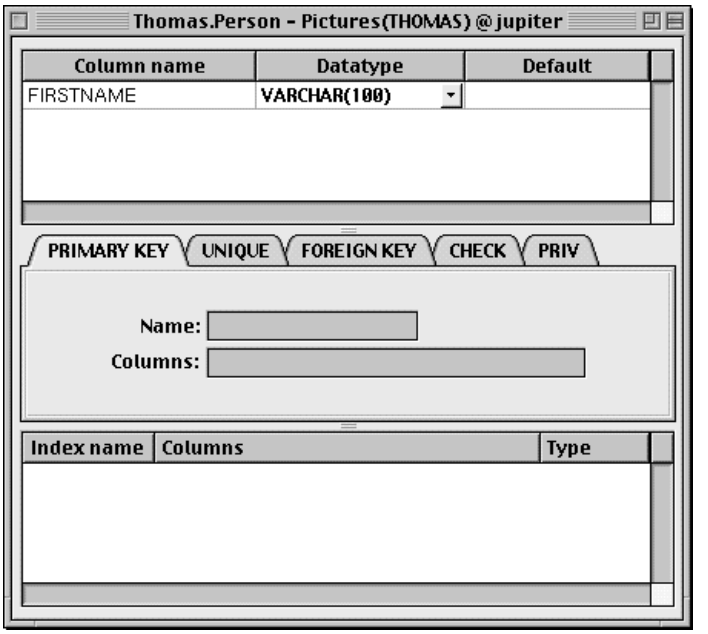


Enter the name of the table you are going to create and click on the OK button. A table must have at least one column that is to be entered into the now upcoming panel.



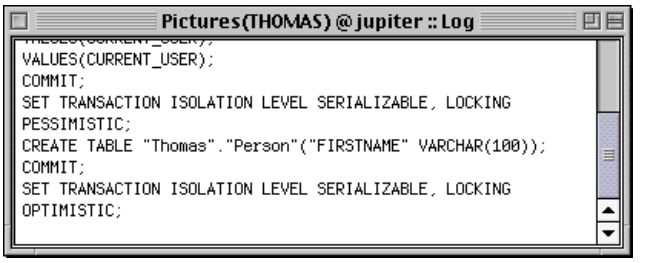
At a minimum, you need to enter a column name, but also both the datatype and the default value can be changed / set. Single column constraints can be created by selecting the appropriate constraint. Please note that PRIMARY KEY implies NOT NULL.

Enter firstname or whatever in the Column name field and choose varchar(<len>) from the Datatype combobox. Then replace <len> with an integer, e.g. 100. That is the maximum length of the varchar column being created. Click on Add. The following table editor window appears.



Note that the icon in the upper left corner of the window indicates an uncommited transaction. More precisly no create statement has been sent to the database yet at all. You can add additional columns by choosing Management->Create->Column from the menu (this can be done at any time after creating the table as well) or let FB-Manager send the create table statement to the database by either choosing Transaction->Commit from the menu or simply pressing Command-s.

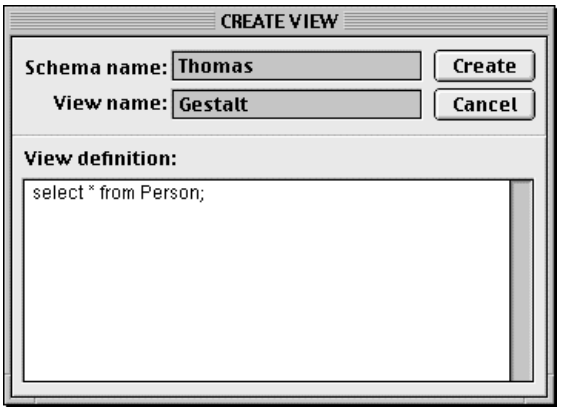
Have a look in the log window. It should show the table create statement followed by a commit.



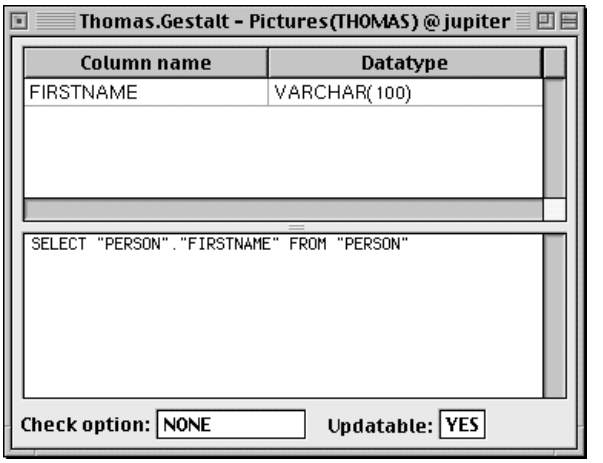
Please note that indexes and multi-column constraints cannot be created until the table has been physically created in the database, i.e. you need to do a COMMIT once the columns have been added.

Creating a view

Select the schema, by clicking on a schema in the browser, and choose Management->Create->View from the menu. Enter the name of the view (see Creating a table) and press Ok. The following window appears:



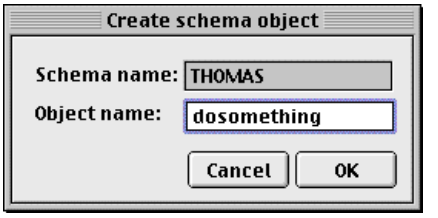
Enter the view definition and click on the Create button. The view will be created in the given schema and a view editor appears:



The names of the columns of the view can be changed by double-clicking on a column name and entering the new name. You may want to press Command-s now to commit the transaction.

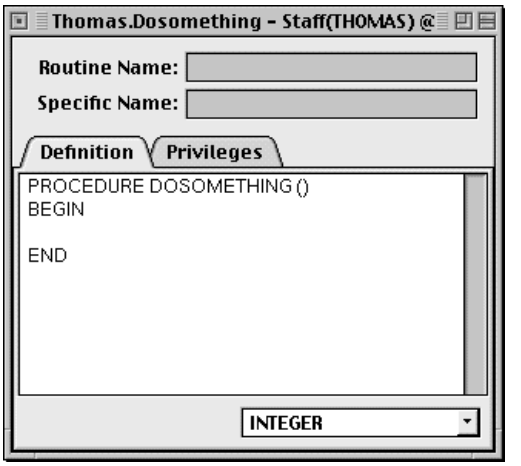
Creating a stored procedure

To create a stored procedure select a schema in the editor browser and choose Management->Create->procedure from the menu. The Create schema object window appears.



To create a stored procedure select a schema in the editor browser and choose Management->Create->procedure from the menu. The Create schema object window appears.

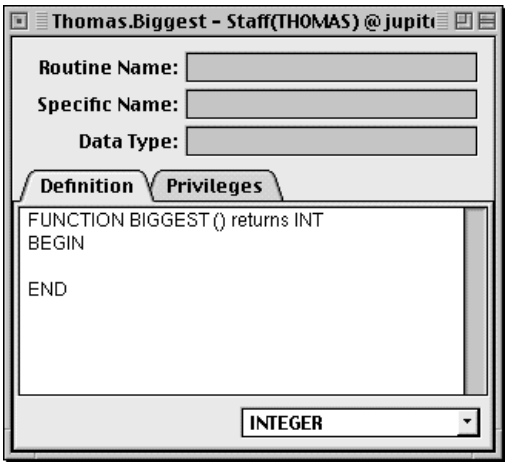
Enter a name for the procedure (e.g. dosomething) and press Return. The following window appears:



Note that a procedure template is created automatically. All you have to do is fill in the body, if necessary specify some parameters (see Editing a stored procedure) and press Command-s to commit the transaction (create the procedure).

Creating a stored function

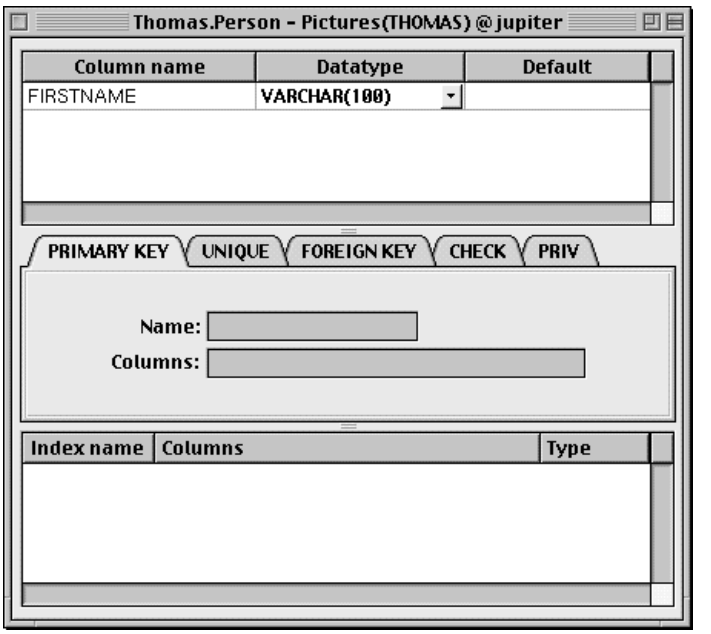
Make sure you have selected a schema in the editor bwoser, then choose Management->Create->Function from the menu. Enter a function name in the Create Object panel and press Return. The following window appears:



Note that a function template with default return type INT is created automatically. All you have to do is fill in the body, if necessary specify some parameters (see Editing a stored functions) and press Command-s to commit the transaction (create the function).

Editing a table

Click on a schema in the first column of the browser and on Tables in the second column. All tables of the selected schema appear in the third column. Select the table you want to edit and choose Management->Definition Editor from the menu or simply doubleclick on the table name. The second approach assumes that you have specified "Definition Editor" as the default editor in application preferences. The table editor appears.

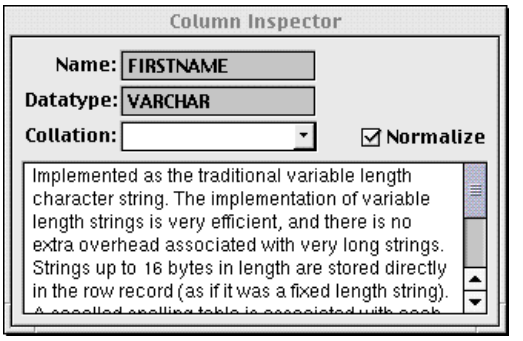


Click on a schema in the first column of the browser and on Tables in the second column. All tables of the selected schema appear in the third column. Select the table you want to edit and choose Management->Definition Editor from the menu or simply doubleclick on the table name. The second approach assumes that you have specified "Definition Editor" as the default editor in application preferences. The table editor appears.

The table editor lets you:

- Rename the table - Management->Rename...
- Change column names - Double-click and edit
- Changing the datatype of columns - select from the combobox
- Set a collation for character columns - select from the combobox
- Set normalize for character columns - click the checkbox
- Add / Drop columns - Management->Column->Add / Drop
- Add / Drop deafults - Double-click and edit (blank to drop)
- Add / Drop constraints - Management->...
- Add / Drop indexes - Management->...
- Add / Drop checks - Management->...

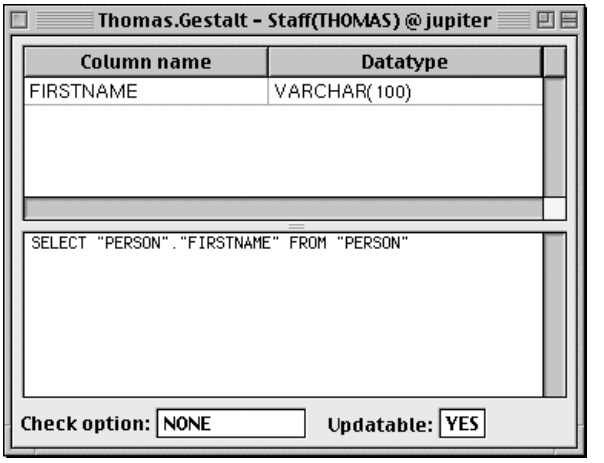
If you select a column in the table editor window, a column inspector is ordered front that for character columns allows you to set a collation and enable / disable normalization. Moreover it contains a brief descripton of the selected datatype.



Editing a view

Double-click on the view name in the editor browser of the management window if you have specified "Definition Editor" in application preferences of choose Management->Definition Editor from the menu.

The following window appears:



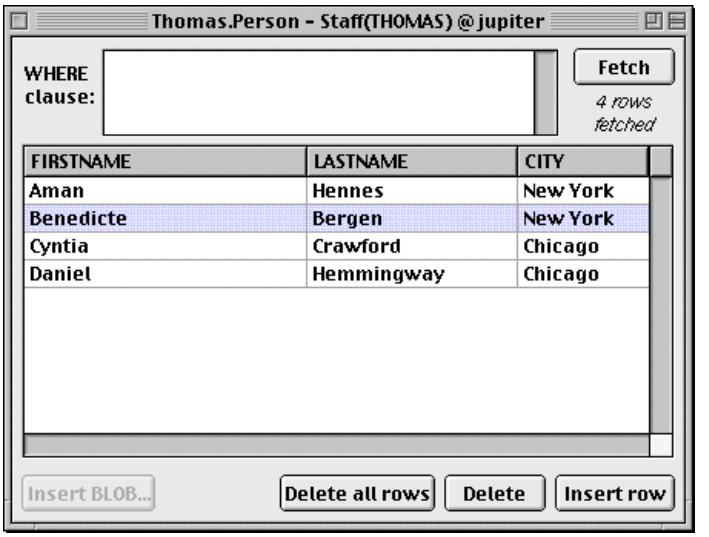
Double-click on the view name in the editor browser of the management window if you have specified "Definition Editor" in application preferences of choose Management->Definition Editor from the menu. The following window appears:

The view editor lets you:

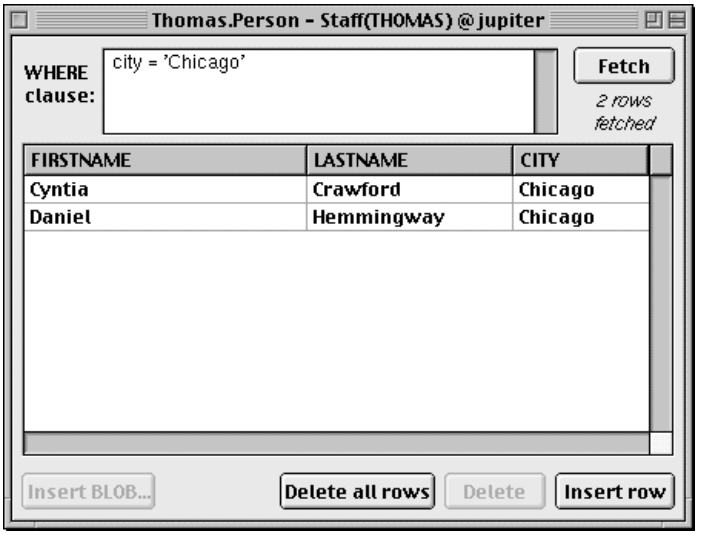
- Rename the view - Management->Rename...
- Change column names - Double-click and edit

Viewing and editing content data (rows)

Select a table or view in the editor browser and choose Management->Content Editor from the menu. The following window appears:



If you click on the Fetch button all rows of the table (or view) will be fetched. If you want to limit the number of rows to be fetched you need to enter an appropriate WHERE clause as in the following example. After entering the clause either click on fetch or simply press **Return**.

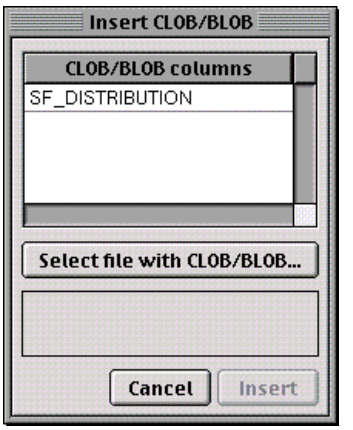


The content editor lets you:

- Delete all rows ("Delete all rows" button)
- Delete selected rows (hilite and hit "delete" button)
- Insert rows ("Insert row" button)
- Update column values (double-click and edit)
- Insert BLOB / CLOB values ("Insert BLOB..." button)
- Drop BLOB / CLOB values (double-click and make blank)

Inserting a BLOB/CLOB value

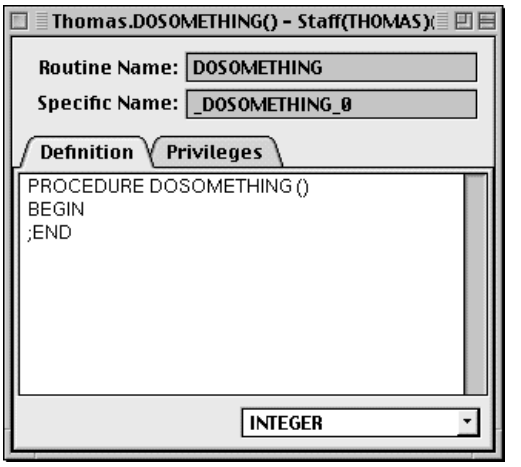
Bring up the content editor for the given table, select a row, click on the "Insert BLOB..." button and the following window will appear:



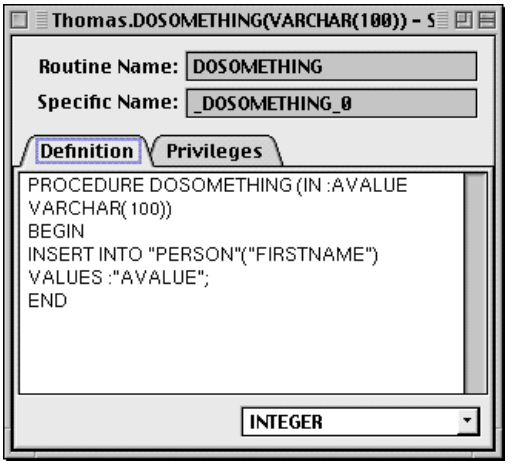
Select a column (only BLOB or CLOB columns will be shown), select a file by clicking on the "Select file with CLOB/BLOB" button, and click insert.

Editing a stored procedure

Select the schema that contains the procedure you are going to edit in the editor browser. If this schema contains any procedures a row Procedures appears in the second column. Click on this row. All procedures of the selected schema are listed in the third browser column. Double-click on the procdure you want to edit. The corresponding editor window appears.



Click into the textview and modify the procedure definition to fit your needs. If you select a FrontBase database in the combobox, the corresponding string is inserted into the textview for your convenience. After entering some logic and commiting the transaction (save changes), the window might look as follows.



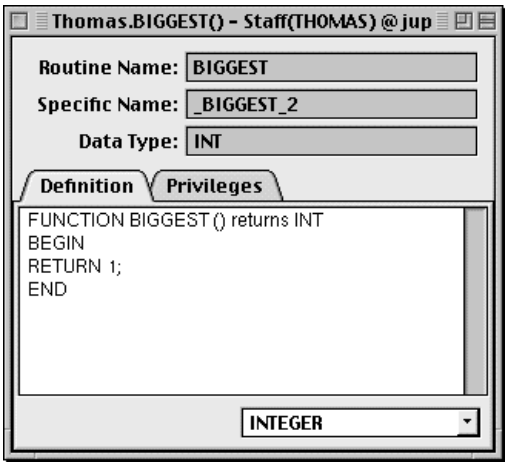
You can modify the set of parameters and the procedure definition at any time. To call the newly created procedure click on the SQL92 tab of the management window and enter:

```
call dosomething('Steve');
```

in the textview. Then press Esc to execute the statement. Commit the transaction and verify that a new row has been inserted into table Person.

Editing a stored function

Select the schema that contains the function you are going to edit in the editor browser. If this schema contains any functions a row Functions appears in the second column. Click on this row. All functions of the selected schema are listed in the third browser column. Double-click on the function you want to edit. The corresponding editor window appears:

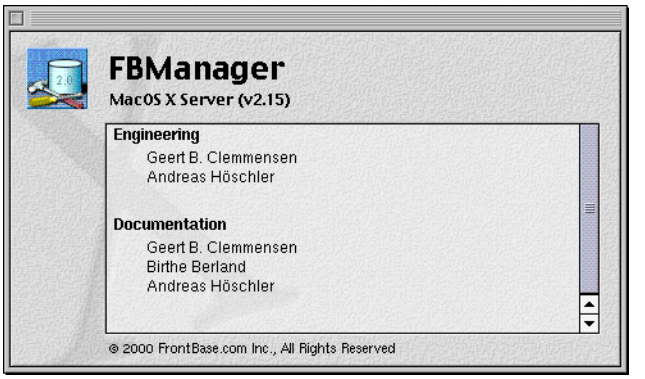


The only difference to editing a stored procedure is that you have to specify a return type and a return value in the function definition.

7

FBWebManager

This chapter introduces the FBWebManager web-based application, which is available for all installations of FrontBase and is accessed through any browser that supports dynamic HTML (aka JavaScript).



This chapter contains the following:

8

sql92

This chapter introduces the sql92 command-line tool, which is available for all installations of FrontBase. sql92 is a command line tool for execution of SQL 92 statements.

This chapter contains the following:

Command syntax

```
sql92 <options> [<filename>]
```

If a filename is specified, sql92reads and executes the SQL 92 statements stored in the named file. If the input file is omitted, SQL 92 statements are read from standard input. All output is written to standard output.

Options:

- s Silent. Do not print commands as they are executed. By default, statements are printed when the the input source is not a terminal device.
- v Verbose. Print statements as they are executed.
- i Ignore. Normally sql92 will exit when an error is encountered. When the ignore option is specified, sql92 does not exit.
- e Exit on error.

- p Prompt. By default sql92 will prompt for statements when the input source is a terminal device. When the prompt option is specified, sql92 will always prompt.
- c Compare. Perform a test, compare the output with the input, write a passed comment if they are equal, and a failed when not.
- u <encoding>Input encoding. The <encoding> argument specifies the encoding of the input files. The default is UTF8 which will also accept us-ascii.

General

A sql92 command is always terminated by a ';' character. When you type input to sql92 it will not interpret the statement until a ';' character is input. This allows a SQL 92 statement to span several lines. Most SQL 92 statements are interpreted by the FrontBase SQL server, but a number of commands are interpreted by sql92. These commands typically deal with connections to the server and administration of the input files.

Comments start a with '#' character or with the '--' string and is terminated by the end of line. The '#' as a comment delimiter allows you to write interpreters. Comments are considered part of the input and are thus part of the verbose output. Lines beginning with the character '>' are also regarded as comments, but are not regarded as part of the verbose output. Result from executing SQL 92 statements are written with lines beginning with '>'. This allows you to write test scripts where the output of the script is the same as the input, simplifying regression testing.

When sql92 is invoked it attempts to execute an initialization file, with the name .slq92rc.sql. sql92 searches the current working directory and the user's home directory.

Commands interpreted by sql92:

Connect

Syntax:

```
CONNECT TO <database-name>
    [DATABASE_PASSWORD <database-password>]
    [ON <database-name>]
    [AS <connection-name>]
    [USER <user-name>]
    [PASSWORD <password>];

CONNECT TO DEFAULT;
```

The first form establishes a connection to the database named <database-name> with the optional database password <database-password> on the host named <host-name>. If the <host-name>is not specified, the local host is assumed. The connection is named with <connection-name>, which is also used as prompt value. If the connection name is not specified, the connection is named with the name of the database name and the name of the host. The <user-name> specifies the authorization identifier used for the SQL 92 session established. If the <user-name>is not specified, the login name of the host operating system is used. The <password> specifies the user password. The connect command is an SQL 92 statement but is interpreted by sql92.

The second form establish the default connection.

Create Database

Syntax:

```
CREATE DATABASE <database-name> [ON <host-name>];
```

Create a new database with the name <database-name>on the host with the name <host-name>. If the host name is not specified the name of the local host is used.

Define Blob and Define Clob

Syntax:

```
DEFINE BLOB <blob-name> LENGTH <length> VALUE {<hex-bytes>};
DEFINE CLOB <clob-name> LENGTH <length> VALUE {<hex-bytes>};
```

Define a blob (or clob) object with the name <blob-name> (or <clob-name>), length <length> in bytes, and the value which is a list of <hex-bytes>. A <hex-byte> is two hexadecimal digits. The list of <hex-bytes> may contain white space (newlines, tabs, spaces, etc.). The blob (or clob) object is defined and can be referenced in subsequent SQL 92 statements. A reference has the form @'<blob-name>' (or @'<clob-name>') and will create the blob (or clob) object for that particular column. The blob (or clob) object will disappear when the next commit or rollback is executed.

Delete Database

Syntax:

```
DELETE DATABASE <database-name> [ON <host-name>];
```

Delete the database with the name <database-name> on the host with the name <host-name>. If the host name is not specified, the local host is assumed.

Disconnect

Syntax:

```
DISCONNECT <connection-name>; DISCONNECT CURRENT; DISCONNECT ALL;
```

The first form disconnects the connection with the name <connection-name>. The second form disconnects the current connection, and the third form disconnects all connections.

Execute

Syntax:

```
EXECUTE <file-name>;
```

Reads and executes the sql92 commands from the file named <file-name>.

Exit and Quit

Syntax:

```
EXIT
QUIT
```

Exits sql92. The EXIT and QUIT commands are the only ones that do not need to be terminated with a ';'.

Set Connection

Syntax:

```
SET CONNECTION <connection-name>;
SET CONNECTION DEFAULT;
```

The first form makes the connection with the name <connection-name>the current connection. The second form makes the default connection current connection.

Set Database Password

Syntax:

```
SET DATABASE_PASSWORD [<password>];
```

Sets the database password to <password>. If <password> is omitted, future connections will not require a password.

Set Default

Syntax:

```
SET DEFAULT LOCKING [OPTIMISTIC | PESSIMISTIC];
```

Set the default locking discipline that is used when a new session is created.

Set Password

Syntax:

```
SET PASSWORD
    [USER <user-name>]
    [OLD <old-password>]
    [NEW <new-password>];
```

Sets the password for the user with the name <user-name>. If the user already has a password it must be specified and the new password <new-password> is established for the user. If <new-password> is omitted, the user may log in without a pssword.

Show Connections

Syntax:

```
SHOW CONNECTIONS;
```

Show all the connections that have been established.

Start Database

Syntax:

```
START DATABASE <database-name> [ON <host-name>];
```

Start the database with the name <database-name> on the host with the name <host-name>. If the host name is not specified the name of the local host is used.

Stop

Syntax:

```
STOP;
```

Stop the database connected to the current connection.

Stop Database

Syntax:

```
STOP DATABASE <database-name> [ON <host-name>];
```

Stop the database with the name <database-name> on the host with the name <host-name>. If the host name is not specified the local host is assumed.

9

Keeping Current

FrontBase is continually improved. If you have obtained FrontBase from a CD-ROM or bundled with your operating system or computer, you most likely are not running the latest version (or reading the latest documentation!).

This chapter contain the following sections:

Determining the Latest Version

Using sql92

Establish a connection to a FrontBase database, then execute the following SQL statement:

```
VALUES(SERVER_NAME);
```

The version of FrontBase currently running as well as the version of FrontBase used to create the database will be returned.

FBWebManager

Establish a connection to a FrontBase database, then execute the following SQL statement:

```
VALUES(SERVER_NAME);
```

The version of FrontBase currently running as well as the version of FrontBase used to create the database will be returned.

FBManager

(MacOS X and Windows NT only) Establish a connection to a FrontBase database, then execute the following SQL statement:

```
VALUES(SERVER_NAME);
```

The version of FrontBase currently running as well as the version of FrontBase used to create the database will be returned.

Upgrading your FrontBase server

When you need to upgrade, proceed to section "Installation" on page 29. Choose your platform and download the FrontBase software.

NOTE: If you currently have a previous version of FrontBase installed and running on your server, pay careful attention to the upgrade instructions!.

A

SQL92 Primer

SQL 92 is the latest official standard for SQL from the ANSI/ISO bodies and as such represents an amalgamam of many years of experience with the language SQL and the various implementations

Overview

FrontBase is the first industrial strength database server that implements virtually all of SQL 92. This may not be important to the work you want to do with a database server, but there are at least a couple of issues that you need to be aware of. The single most concept we have seen that can create a little confusion is the concept of SCHEMAs. While many products on the market use the word SCHEMA in their literature, very few implement this concept (at least in the SQL 92 sense).

Actually, there is a layer on top of SCHEMAs called a CATALOG. An SQL 92 database is comprised of a number of CATALOGs, each holding a number of SCHEMAs. A SCHEMA can be viewed as the container for a number of objects: TABLEs, VIEWs, DOMAINs, COLLATIONs, etc.

The topics in this section are:

CATALOGs

Currently, FrontBase offers the support for one CATALOG in a database. This CATALOG inherits the name of the database, e.g. if the database was created with the name Movies.fb, the catalog is named MOVIES. This catalog name is always used as the default catalog and thus you don't really need to worry about CATALOGs.

SCHEMAs

As mentioned earlier, a catalog can contain many SCHEMAs. A SCHEMA is owned by a user (see further below) and only this user can add or drop objects in the SCHEMA. A SCHEMA object can be a table, a view, a domain, ... By means of the GRANT/REVOKE statements, other users can be granted a number of privileges pertaining to the objects within a schema, e.g. the INSERT privilege on a table.

Objects in a SCHEMA can be referenced via so-called qualified names:

```
[[<catalog name>.] <schema name>.] <object name>
```

SQL 92 offers a rich "default for everything" setup, so whenever you want to reference objects in the current SCHEMA, only the <object name> needs to be given.

When a new database is created, two SCHEMAs are created as well: DEFINITION_SCHEMA and INFORMATION_SCHEMA. The DEFINITION_SCHEMA holds all the objects used to maintain all other SCHEMAs. The INFORMATION_SCHEMA holds various objects, which offer access to the objects in the DEFINITION_SCHEMA, and a number of convenience objects. For example if you want to see which TABLEs have been defined in a database, the following SQL 92 statement could be executed:

```
SELECT * FROM INFORMATION_SCHEMA.TABLES;
```

INFORMATION_SCHEMA.TABLES is actually a VIEW defined like:

```
CREATE VIEW INFORMATION_SCHEMA.TABLES AS
   SELECT * FROM DEFINITION_SCHEMA.TABLES;
```

The VIEWs in INFORMATION_SCHEMA are all non-updateable, i.e. an INSERT like:

```
INSERT INTO INFORMATION_SCHEMA.TABLES ....will fail.
```

The DEFINTION_SCHEMA is maintained exclusively by FrontBase and cannot be accessed or manipulated directly by any user.

To create a new SCHEMA (which the current user will then own):

```
CREATE SCHEMA <schema name>;
```

NOTE: See the SQL 92 standard for the complete syntax of which the example is only but a tiny fragment.

To make a schema the current schema:

```
SET SCHEMA '<schema name>';
```

NOTE: (Please note that the schema name is given using a character string).

To see the list of defined SCHEMAs:

```
SELECT * FROM INFORMATION_SCHEMA.SCHEMATA;
```

To see what is the current schema, the CURRENT_SCHEMA string function is available. Please note that CURRENT_SCHEMA is a FrontBase extension to SQL 92.

USERs

The concept of users in SQL 92 is relatively simple, but is tied very closely to the concept of schemas.

To access a database, a user name is required, otherwise access is denied (FrontBase offers password protection as an extension to SQL 92).

When a new database is created, a number of user names are also created, among which are: _SYSTEM and _PUBLIC. Both these user names are considered by SQL 92 as special user names, in fact the leading underscore cannot be used in regular identifiers, and is not to be used.

To create a new user:

```
CREATE USER <user name> [DEFAULT SCHEMA <schema name>];
```

To change the default schema:

```
ALTER USER <user name> SET DEFAULT SCHEMA <schema name>;
```

The optional <schema name>, which must exist when the user name is created, will be the default schema for the user whenever the database is accessed. If no default <schema name> is given, a schema with the same spelling as the user name is created and used as default (this will happen the first time the user accesses the database).

To see who is the current user

The USER and CURRENT_USER string functions can be used (USER is simply a shorthand for CURRENT_USER).

To make a user name the current user:

```
SET SESSION AUTHORIZATION <user name>;
```

To see the list of defined user names:

```
SELECT * FROM INFORMATION_SCHEMA.USERS;
```

DATE, TIME and TIMESTAMP

SQL 92 has an elaborate time concept which includes the following datatypes:

- DATE
- TIME
- TIME WITH TIME ZONE
- TIMESTAMP
- TIMESTAMP WITH TIME ZONE

DATE holds year, month and day, i.e. NO time components.

TIME holds hour, minute, and second.

TIMESTAMP holds year, month, day, hour, minute, and second.

When a TIME or TIMESTAMP literal is inserted into a database, the server's time zone is added to the literal. Example: If the server is running in Denmark and it is August, the server's time zone is GMT+02:00. If TIMESTAMP '1999-08-02 11:49:00' is inserted, the literal is thus adjusted with +02:00. If, however, TIMESTAMP '1999-01-02 11:49:00' is inserted, the literal is adjusted with +01:00 (because there is no daylight savings in January).

If you want to be in full control over the time zones, you should use the TIMESTAMP WITH TIME ZONE datatype.

```
Example: TIMESTAMP '1999-08-02 11:49:00-08:00'.
```

The same comments apply to TIME and TIME WITH TIME ZONE.

Keywords and Identifiers

SQL 92 has a very extensive set of keywords and you may run into some surprises when selecting the spelling for an identifier of yours. It may very well collide with the spelling of an SQL 92 keyword. Please also note that an identifier cannot begin with an underscore.

There are a couple of ways to reduce the "collision problems":

- There is no keyword in SQL 92 ending with an underscore, e.g. it will be perfectly legal to use SELECT_ as an identifier.
- By enclosing the identifier in double quotes, essentially any spelling can be used as an identifier, e.g. "SELECT" is a legal identifier.

SQL 92 is case insensitive, .e.g Movies as an identifier is considered identical to MOVIES.

Learning more about SQL 92

You can always get hold of a copy of the standard itself from either ANSI or ISO, but the standard is not really aimed at users. A better way to get acquainted with SQL 92 is to buy "A Guide to The SQL Standard, Fourth Edition" by Chris J. Date and Hugh Darwen. This book explains all concepts and constructs of SQL 92, sometimes with quite an academic viewpoint, but nonetheless very complete and absolutely readable and understandable.

The book is published by Addison-Wesley and has ISBN #: 0-201-96426-0. An easy way to order this book is to go to www.amazon.com, but your local bookstore will most likely be able to help you as well.

Index