Pervasive.SQL 2000i

Pervasive.SQL User's Guide

Guide to Using Pervasive.SQL

Pervasive Software, Inc.
12365 Riata Trace Parkway
Building II
Austin, TX 78727 USA

Telephone: +1 512 231 6000 or 800 287 4383
Fax: +1 512 231 6010
E-Mail: info@pervasive.com
Web: http://www.pervasive.com

disclaimer

PERVASIVE SOFTWARE INC. LICENSES THE SOFTWARE AND DOCUMENTATION PRODUCT TO YOU OR YOUR COMPANY SOLELY ON AN "AS IS" BASIS AND SOLELY IN ACCORDANCE WITH THE TERMS AND CONDITIONS OF THE ACCOMPANYING LICENSE AGREEMENT. PERVASIVE SOFTWARE INC. MAKES NO OTHER WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, REGARDING THE SOFTWARE OR THE CONTENT OF THE DOCUMENTATION; PERVASIVE SOFTWARE INC. HEREBY EXPRESSLY STATES AND YOU OR YOUR COMPANY ACKNOWLEDGES THAT PERVASIVE SOFTWARE INC. DOES NOT MAKE ANY WARRANTIES, INCLUDING, FOR EXAMPLE, WITH RESPECT TO MERCHANTABILITY, TITLE, OR FITNESS FOR ANY PARTICULAR PURPOSE OR ARISING FROM COURSE OF DEALING OR USAGE OF TRADE, AMONG OTHERS.

trademarks

Btrieve, Tango, Client/Server in a Box, and the Pervasive Software logo are registered trademarks of Pervasive Software Inc.

Built on Pervasive, Built on Pervasive Software, Extranet in a Box, Pervasive.SQL, Jtrieve, Plug n' Play Databases, SmartScout, Solution Network, Ultra-light Z-DBA, Z-DBA, ZDBA, UltraLight, MicroKernel Database Engine, and MicroKernel Database Architecture are trademarks of Pervasive Software Inc.

Microsoft, MS-DOS, Windows, Windows NT, Win32, Win32s, and Visual Basic are registered trademarks of Microsoft Corporation.

Windows 95 is a trademark of Microsoft Corporation.

NetWare and Novell are registered trademarks of Novell, Inc.

NetWare Loadable Module, NLM, Novell DOS, Transaction Tracking System, and TTS are trademarks of Novell, Inc.

All other company and product names are the trademarks or registered trademarks of their respective companies.

This product includes software developed by Powerdog Industries.

The ODBC Driver Manager for NetWare (ODBC.NLM) included in this product is based on the GNU iODBC software © Copyright 1995 by Ke Jin <kejin@empress.com> and was modified by Simba Technologies Inc. in June 1999.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

A copy of the GNU Lesser General Public License is included in your installation of Pervasive.SQL 2000i at \pvsw\doc\lesser.htm. If you cannot find this license, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA. You may contact Pervasive Software Inc. using the contact information on the back cover of this manual.

Pervasive.SQL User's Guide
March 2001
100-003672-004

Contents

Figures

Tables

About This Manual

This manual introduces you to Pervasive.SQL utilities for server, workstation and workgroup products and shows you how to perform the basic tasks necessary to work with the application. Topics include starting and stopping the database engine, setting up access to a database, and accessing data from other applications. This manual also gives you a tour of the Pervasive Control Center (PCC). PCC allows you to manage Pervasive.SQL utilities within a single, easy-to-use framework.

Who Should Read This Manual

This manual provides information for users who install and run Pervasive.SQL client/server, workstation, and workgroup products.

Pervasive Software would appreciate your comments and suggestions about this manual. Please send comments to docs@pervasive.com.

Manual Organization

This manual is divided into five chapters:

- Chapter 1—"Introducing Pervasive.SQL"

  This chapter provides an introduction to Pervasive.SQL, and an overview of utilities and Pervasive.SQL documentation.

- Chapter 2—"Using Pervasive.SQL"

  This chapter covers the basic tasks you need to do to work with Pervasive.SQL.

- Chapter 3—"Using the Pervasive Control Center"

  This chapter explains how to get your work done using Pervasive Control Center.

- Chapter 4—"Unix Supplementary Documentation"

  This chapter explains how to get things done on Unix server platforms.

- Chapter 5—"Basic Troubleshooting"

  This chapter provides information for troubleshooting and resolving problems.

- Chapter 6—"Pervasive.SQL Resources and Contacts"

  This chapter explains the resources and information at your disposal as a valued customer of Pervasive Software.

The manual also includes an index.

Conventions

Unless otherwise noted, command syntax, code, and examples use the following conventions:

CASE	Commands and reserved words typically appear in uppercase letters. Unless you are working with Unix or the manual states otherwise, you can enter these items using uppercase, lowercase, or both. For example, you can type `MYPROG`, `myprog`, or `MYprog`.
Bold	Words appearing in bold include the following: menu names, dialog box names, commands, options, buttons, statements, etc.
`Monospaced font`	Monospaced font is reserved for words you enter, such as command syntax.
[ ]	Square brackets enclose optional information, as in [*log_name*]. If information is not enclosed in square brackets, it is required.
|	A vertical bar indicates a choice of information to enter, as in [*file name* | *@file name*].
< >	Angle brackets enclose multiple choices for a required item, as in `/D=<5|6|7>`.
variable	Words appearing in italics are variables that you must replace with appropriate values, as in *file name*.
…	An ellipsis following information indicates you can repeat the information more than one time, as in [*parameter* …].
::=	The symbol ::= means one item is defined in terms of another. For example, a::=b means the item *a* is defined in terms of *b*.

chapter

1 Introducing Pervasive.SQL

Understanding Pervasive.SQL and its Capabilities

This chapter provides an explanation of what Pervasive.SQL is and what it can do for you. This chapter is divided into the following sections:

Understanding Pervasive.SQL

Pervasive.SQL is a comprehensive database management system built around Pervasive Software's MicroKernel Database Engine. Pervasive.SQL offers easy installation, uncomplicated maintenance, and high levels of performance and reliability.

This section explains the product and the components that make it up.

What is a Database?

Loosely defined, a *database* is simply a collection of structured data. Generally, the data is structured by dividing it into sub-sets of information that share the same characteristics. Some examples of a database are:

- A telephone book

  Each entry in the phone book consists of four characteristics: first name, last name, address, and phone number.

- A collection of digital photographs

  Each picture on your hard disk has two characteristics: a file name, and the data within the file that represents the image.

- A list of orchards and the fruit grown by each

  Each entry in the orchard list might consist of three characteristics: orchard name, address, and date founded. The related list of fruits might have five characteristics: the orchard name, the fruit name, its type (McIntosh, Fuji, and so on), its price, and a taste rating.

In the particular context of this product, a database is a specific, well-defined collection of related information. You can probably find one or more databases available on your computer or your network. For example, you may have a database of information related to vendors from whom you purchase supplies or raw materials, and you probably also have a database containing customer or member information. Each of these is a distinct, well-defined collection of related information.

What is a Database Management System?

As citizens of the computer age, we are surrounded by collections of information—databases—everywhere we go. Unfortunately, all this data is of no use to anyone without methods to sort it, search it, analyze it, and keep it up to date.

A *database management system, or DBMS,* is a computer program designed to manage large amounts of data and to allow other computer programs and people to interact with the data. A DBMS can also be referred to informally as a *database engine* or simply an *engine*. A DBMS performs the following tasks:

- *Controls access to the data.* The DBMS can act as a watchdog to prevent the wrong people from using the data.
- *Structures the data so it can be interpreted by other applications.* The DBMS ensures that all the data adheres to the database structure, so that other computer programs can work with the data using common methods.
- *Keeps the data safe and prevents the data from getting garbled or lost.* The DBMS facilitates backing up the data in case of catastrophic loss, and also accesses the data in a consistent manner to prevent the data from inadvertent damage.
- *Makes it easy to add new information, find it, update it, and delete it.* The DBMS readily accepts new data and provides tools that you can use to locate, update, and remove information as you see fit. It verifies that the data inserted fits within defined attributes for the database fields.
- *Allows you to analyze relationships among different sets of data.* The DBMS stores the data in a way that allows you to examine how any piece of data relates to any other piece of data.

In summary, the DBMS organizes your data, keeps it safe, and helps you to use it and understand it.

Components of Pervasive.SQL

The Pervasive.SQL DBMS consists of a variety of components designed to help you achieve your data management goals. For more detailed information about these components and how they interact, please see *Pervasive Products and Services*.

MicroKernel Database Engine

The MicroKernel Database Engine (MKDE) is the high-performance heart of Pervasive.SQL. The MKDE works directly with the data files on your computer's hard disk. When requested, it directly inserts

new data, deletes unnecessary data, and ensures the safety and integrity of the data files at all times, even when people and applications are working with the data.

SQL Relational Database Engine

The SQL Relational Database Engine (SRDE) interacts with the MKDE and the client (described below). It provides many powerful features including support for Microsoft ODBC, sophisticated search and analysis capability, and security.

Client or Requester

In client/server systems, the client resides on the computer workstation. It interacts with the client application and across the network with both the MKDE and the SRDE on the server.

Pervasive Control Center

The Pervasive Control Center (PCC) is an easy-to-use, graphical tool designed to help you create and manipulate databases and control your DBMS. It allows you to access nearly all the functions of the product from one place. For a brief tour of PCC, see Chapter 3, "Using the Pervasive Control Center."

Utilities

A variety of graphical and command-line tools provide support for testing, configuring, and manipulating the many features and options provided by Pervasive.SQL. These tools are covered in-depth in *Advanced Operations Guide*, but a brief introduction to them is provided in "Helpful Utilities" on page 1-16.

Documentation

Pervasive.SQL comes with a printed copy of *Getting Started with Pervasive.SQL*, and with a complete set of online documentation. For more information about the documentation, see "Using Pervasive.SQL Documentation" on page 1-18.

Key Concepts

This section explains some basic concepts of databases in general and some of the key concepts that distinguish Pervasive.SQL from other database products.

Basic Database Structures and Terms

Most database management systems in use today share a common set of basic structures. This section briefly explains those structures. The descriptions that follow refer to the diagram below:

"Phone Book" Table

	Column 1	Column 2	Column 3	Column 4
Col Names	**Name**	**Address**	**Zip**	**Phone**
Row 1	Fred Black	643 Oak	12346	555-2345
Row 2	Jane Doe	112 Elm	12345	555-1212
Row 3	John Doe	112 Elm	12345	555-1212

Value

The most basic element of a database is a *value*. A value is one piece of data, one characteristic, for a specific entity. For example, in the diagram, the name "John Doe" or the phone number "555-1212" is a value.

Column or Field

The next element is called a *column*, or a *field*. A column represents a characteristic with no specific value. Columns generally have names that describe the given characteristic. For example, in the telephone book, **Name** and **Phone** are columns. They do not have specific values unless you look up a particular person. *Field* is sometimes used to refer to the generic characteristic of a specific row, same as *cell* below. For example, someone might point at a specific box in the table above and ask, "What is the value of that field?"

Row or Record

The next element is called a *row*, or a *record*. A row is a collection of all the values for one particular instance. For example, one entry in

the phone book, complete with name, address, and phone number, is one record or row.

Cell

A cell is a column within a specific record. You can think of it as the intersection of a row and a column. Each cell has a specific value. For example, you might tell a co-worker, "The value of the cell located at row 2, column 3 is '12345'."

Table

A collection of rows and columns makes up a *table*. A table is a set of data that shares exactly the same structure. Tables generally have names that describe the contents of the table. For example, the table above is called "Phone Book." With Pervasive.SQL, each table is stored as a separate data file on the hard disk.

Index

An *index* is an ordered list of all the values in a particular column. A table can have zero or more indexes on it. The database engine uses indexes to find specific records in the database without having to step through every record one at a time. Creating indexes on columns which will frequently be used in database searches is likely to improve the performance of your database.

Database

A *database* is a collection of one or more tables. The data in the tables do not need to be related among the various tables, but usually there are many relations. For example, a database might consist of the "Food Preferences" table below, and the "Phone Book" table above. With Pervasive.SQL, a database consists of one or more data files and Data Dictionary Files (or DDFs) on your hard disk. The Data Dictionary Files are special data files that contain all the definitions for tables, columns, and other attributes that define the structure of your database.

Schema

The term *schema* refers to the complete set of definitions that describe the entire structure of a database. A typical schema includes

definitions for tables, columns, indexes, and many other attributes. The DDFs for a database contain the database's schema.

"Food Preferences" Table

	Column 1	Column 2	Column 3	Column 4
Col Names	**Name**	**Meat**	**Grain**	**Drink**
Row 1	Fred Black	sushi	wheat	sake
Row 2	Jane Doe	steak	oats	beer
Row 3	Ann Dean	cod	bran	spring water

Remote

The term *remote* refers to an object, such as a file server or a database, that is not located in the computer you are using now. When you connect to a database over the network, you are connecting to a *remote* database. *Remote* is the opposite of *local*. *Remote* can refer to either the client or the server, depending on whether you are currently seated at the server computer or a client computer. *Remote* always refers to an object that is not located on the system you are using.

Local

The term *local* refers to the computer you are using right now, or something stored on this computer. A *local* database is a database in which the data files are stored on the hard disk of the computer you are currently using. *Local* is the opposite of remote. *Local* can refer to either the client or the server, depending on whether you are currently seated at the server computer or a client computer.

Relational

The term *relational* refers to a method of organizing data that relies on having a certain amount of duplicate data in different sub-sets of data, so that relationships between the sub-sets can be established and analyzed. Most DBMSs in use today are relational systems.

Join

A *join* refers to a relation between two tables. Joins are the true beauty behind the *relational* DBMS. For example, you can see that both our example tables contain the **Name** column, and some of the names are the same. Because we can cross-reference the names in the **Phone** table with the names in the **Food** table, we have the power to ask and answer such questions as, "What is the phone number of someone who likes steak?"

In the real world, we have the power to answer such questions as, "Which consumer profile purchased the most product B after buying product A?" You can probably see now how powerful relational data access can be.

The SRDE component of Pervasive.SQL provides full relational access to your data.

Unique Benefits of Pervasive.SQL

One unique feature of Pervasive.SQL is that it allows applications to access data through either the industry-standard relational method outlined above, or through an ultra-high-speed transactional or hierarchical method known as the Btrieve interface. In fact, Pervasive.SQL allows applications to use both access methods at the same time to access the same data.

Transactional Interface

The transactional interface is a high-performance, low-overhead access method, capable of handling updates, inserts, and deletes much faster than other database products.

Applications that use the transactional interface bypass the relational interface and communicate directly with the MKDE to maximize performance.

In the interest of performance, the transactional interface offers only basic security, including file passwords and encryption. It does not allow SRDE data access to bypass transactional security.

Relational Interface

The relational interface uses industry-standard ODBC to provide a rich environment for data definition, security, reporting, stored procedures, and universal application access without requiring any

application programming. Databases that are ODBC-enabled can be accessed by any ODBC-standard software program.

As an end user of an application based on Pervasive.SQL, you may not be able to choose which access method your application uses, but your application vendor has most certainly taken this into account. No other DBMS available today offers this combination of flexible relational access and high-speed transaction throughput.

Terminology Revisited

When using the Btrieve interface, the terms *table* and *database* are generally not used, and data files are referred to directly as such. In addition, Btrieve users normally use the terms *records* and *fields* rather than *rows* and *columns*.

Why Pervasive.SQL?

You or your application vendor have chosen wisely in selecting Pervasive.SQL as the database of choice. Within the realm of small- to mid-sized enterprises, Pervasive.SQL provides the lowest-maintenance, highest-performance, best overall DBMS value on the market today.

Pervasive.SQL provides a number of advantages over other products. Here are just a few:

- *Lowest total cost of ownership.* An independent study conducted by Aberdeen Group concluded that no major database product can match Pervasive.SQL's low total cost of ownership.
- *No Database Administrator (DBA) required.* You can look in the newspaper any day of the week and see classified ads for Oracle, Sybase, or SQL Server database administrators, with sky-high salaries. Pervasive.SQL offers the unique Zero Database Administrator, or Z-DBA™, architecture. Its easy-to-use tools, bulletproof installation, and set-it-and-forget-it simplicity make it the perfect workhorse for desktop, workgroup, and departmental applications.
- *Scalable from the desktop to the Web.* Pervasive.SQL is available in three editions: the Ultra-light™ Workstation database engine supports applications running locally on the same computer as the engine; the Workgroup engine comes with a three-user license and scales up to about ten users; and the Server engine comes with a 10-user license and scales to thousands of concurrent users, including intranet and extranet applications. Upgrading to another configuration requires no changes to the supported application, just plug-and-play with the new database engine.
- *Cross-platform support.* Unlike some other products, Pervasive.SQL does not lock you in to a single platform. Pervasive.SQL databases are binary-compatible and supported across Microsoft Windows, Novell NetWare, Sun Solaris, and several varieties of Linux. No matter where your data is or where it is going to be, Pervasive.SQL is there for you.

- *Big database features at a small price.* Pervasive.SQL offers full security, encryption, management and monitoring tools, and a host of other features you would expect to see in more expensive DBMS products.
- *Legendary stability and reliability.* Pervasive database engines have been serving the needs of business users for nearly twenty years. There's no doubt why 70% of the Windows desktop accounting market uses Pervasive.SQL as the underlying DBMS of choice. When you've got to manage important data, you go for the DBMS that won't let you down.
- *Multiple access methods.* Your application vendor can use the transactional interface for blazing performance on bulk data operations, while offering the richness of ODBC, OLE-DB, pure Java, and JDBC interfaces for data reporting, security, analysis, and standard compatibility. No other database management system offers all these access methods.

Understanding the Product Family

Pervasive.SQL is available in three different packages. The major differences between the packages are price and multi-user features.

- The single-user Workstation engine is least expensive, but it does not provide support for multi-user databases.
- The Workgroup engine is designed for maximum flexibility in small multi-user network environments where there may or may not be a dedicated database server.
- Finally, the Server engine is designed for maximum scalability in high-volume, mission critical database applications where there is a dedicated database server. The Server engine quickly becomes most economical per-seat as you increase the number of users.

All three engines are plug-n-play compatible with any Pervasive.SQL database. To upgrade or downgrade from one package to another requires no changes to your application or to your database. Simply install the new package and you are ready to go. All three engines were designed with a common architecture and, with the exception of networking and multi-user features, offer exactly the same features.

In addition, there are several add-on packages that you can use to expand the capabilities of your database. This section explains all the different packages.

Workstation

Pervasive.SQL Workstation runs on a single computer, and it does not allow multi-user access to data. While a Workstation may connect to data files on a file server via a mapped drive, no other user can access those files if a Workstation engine is accessing them. Workstation does not offer the ability to share data. However, it does include the client software. If you have Workstation installed, you can connect to any Workgroup or Server engine available on your network, assuming you have been granted access rights to those databases.

Workstation offers the lowest price point in the product family. It offers a single-user version of the same powerful features and time-tested reliability of the Workgroup and Server engines.

Workgroup

The Workgroup engine offers a peer-to-peer network setup designed for small workgroups. The Workgroup engine is the only engine that offers multi-user access to Pervasive.SQL data located on a computer where no database engine is installed. For example, you can share data files stored on a NetWare or Windows server where no database server is installed, using a Workgroup engine installed on a different computer.

The Workgroup architecture is essentially the same as the Server architecture. One significant difference between the two is that the Workgroup engine does not support remote ODBC client connections. For example, you cannot monitor SRDE performance of a Workgroup engine from a remote workstation, nor can you connect to a Workgroup engine from another computer unless you have a Workgroup engine installed on the client computer.

Another major difference between Workgroup and Server is the Gateway feature of Workgroup. When there is no database engine running on the computer where the data is located, normally the first database engine to connect to that data handles all requests from other engines to access that data. This feature can be configured so that the same Workgroup engine always services that data, or the Gateway designation can be allowed to "float" based on which Workgroup engine connects to the data first during any given work day.

Workgroup engine performance is excellent up to roughly twenty users, however server engine licenses are less expensive once you reach 7-10 users (in most cases).

Server

The Server engine offers a full client/server architecture providing excellent performance and scalability for up to thousands of concurrent users. The Server engine can be monitored and configured remotely.

The Server engine must be located on the same computer as the data files it is intended to access.

The table below shows a comparison of Server, Workgroup, and Workstation features.

Table 1-1 Comparison of Server, Workgroup, Workstation Features

Feature	Server	Workgroup	Workstation
Supports Btrieve, ODBC, OLE-DB, Java, JDBC, and ActiveX interfaces	✔	✔	✔
Full-featured relational support (online backup, security, referential integrity, management tools, and so on)	✔	✔	✔
Binary compatible data files across all platforms and engine editions	✔	✔	✔
Easy plug-n-play upgrading, no application changes required	✔	✔	✔
Includes complete online documentation	✔	✔	✔
Can access data on a file server where no database engine is installed		✔	✔
Supports remote ODBC client connections	✔		
Requires a Workgroup engine on all computers expected to access remote data	N/A	✔	N/A
Engine runs on Windows	✔	✔	✔
Engine runs on NetWare	✔		
Engine runs on Linux	✔		
Engine runs on Solaris	✔		
Multi-user for small groups	✔	✔	
Scales to thousands of users	✔		
Extranet license available	✔		

Replication

Pervasive.SQL Replication is an optional product that allows you to synchronize two or more databases that have the same structure and are always or occasionally connected by a network.

This type of product is typically used to keep the data of traveling employees synchronized with the home database, or to allow regional offices to maintain fully synchronized copies of the corporate database without having to download the entire database to each location on a regular basis.

For more information about Replication, contact your sales representative or visit the Pervasive web site at: http://www.pervasive.com.

Crystal Reports for Pervasive.SQL

Another optional product, Crystal Reports provides rich capabilities for creating and formatting reports based on Pervasive.SQL databases. Reports can be customized in thousands of ways and published as HTML, Microsoft Word document, Microsoft Excel document, or other formats.

For more information about Crystal Reports, contact your sales representative or visit the Pervasive web site at: http://www.pervasive.com.

Helpful Utilities

Pervasive.SQL comes with a variety of utilities designed to help you control and manage your databases. Most of the utilities run only on 32-bit Windows and allow remote function to NetWare or Unix database server engines.

Table 1-2 Summary of Pervasive.SQL Utilities

Utility name	Runs on these platforms	Description	Server, Workstation, or Workgroup
Pervasive Control Center	Win32	Utilities warehouse for Pervasive.SQL. Shows list of engines and databases available.	All
Configuration	Win32	Manipulates settings for Pervasive client and server components.	All
Monitor	Win32	Monitors server engine activity, useful for database administration and programming diagnostics.	Server, Workgroup, Workstation - local Server - remote
Function Executor	Win16, Win32	Executes Btrieve operations, enabling you to learn how the Btrieve interface works or test and debug an application.	All
Pervasive.SQL Maintenance	Unix, Win32, and NetWare	Performs common Pervasive.SQL file and data manipulations, such as importing and exporting data.	Unix, Windows, and NetWare servers All 32-bit Windows workstations for GUI version
SQL Data Manager - invoked automatically within PCC	Win32	Allows you to execute SQL Statements interactively. Creates and maintains Data Dictionary Files (DDFs) and database files. Checks and lists RI Constraints on data sources.	All - local Server - remote

Table 1-2 Summary of Pervasive.SQL Utilities

Utility name	Runs on these platforms	Description	Server, Workstation, or Workgroup
Rebuild	Win16, Win32, and NetWare	Converts previous versions of MicroKernel files into version 7.x format.	Windows and NetWare servers Windows 32-bit workstations
User Count Administrator	Win16 and Win32	Increases the Pervasive.SQL user count incrementally with a software key you obtain from Pervasive Software.	Windows and NetWare servers remotely Unix, Workgroup local
ODBC Administrator	Win32	Sets up Data Source Names (DSNs) for client and engine interfaces	All Windows engines - local Server - remote
Gateway Locator	Win32	Used to configure and maintain gateway configuration files for the workgroup engine.	Workgroup engine only
Pervasive System Analyzer	Win32	Analyzes system components, runs communication tests, and archives or restores previous database engine files on your system.	All

Using Pervasive.SQL Documentation

All Pervasive.SQL documentation, both printed and online, assumes you are familiar with the basics of using a computer, such as clicking and dragging, opening and saving files. If you need assistance with these tasks, please consult the documentation that came with your computer and/or operating system.

This section describes the Pervasive.SQL documentation. Of these titles, only *Getting Started* and *Status Codes Quick Reference* card are provided in hardcopy with Pervasive.SQL 2000i SP3; the rest are provided on the Pervasive.SQL 2000i SP3 CD-ROM. These online documentation files are installed on Windows when you choose the **Typical** installation procedure. They are available as an option in the **Custom** installation procedure. The content is accessible through the **Start** menu:
Programs | Pervasive | Pervasive.SQL 2000i | Documentation | Pervasive.SQL 2000i Documentation.

For NetWare users, you can access the online documentation on any Windows workstation that has the database client installed.

For Linux and Solaris users, you can access the online documentation on the server in HTML format. There are also manpages provided for certain utilities.

You can order printed copies of the documentation from your sales representative.

Getting Started Guides

Getting Started with Pervasive.SQL 2000i (Server Edition) and *Getting Started with Pervasive.SQL 2000i (Workstation/Workgroup Edition)* help you get Pervasive.SQL running with installation, setup, and troubleshooting information. Both *Getting Started* editions cover the following topics:

- Preparing to install Pervasive.SQL 2000i
- Installing Pervasive.SQL 2000i
- Upgrading from previous versions of Pervasive.SQL
- Configuring Pervasive.SQL 2000i
- Troubleshooting your Pervasive.SQL 2000i installation
- Where to go for Pervasive.SQL product information and technical support

User's Guide

Pervasive.SQL User's Guide offers you an introduction to Pervasive.SQL and describes common user tasks. The book discusses the database engine, Pervasive.SQL utilities and other key components; the differences between Server and Workstation engines; and the differences between ODBC and Btrieve access. *User's Guide* provides you with the basics in order to work with Pervasive.SQL successfully.

Topics include:

- a brief introduction to the product—a concise guide to the print and online documentation included with the product
- a "how to" chapter with procedures for creating and editing your data files, starting and stopping the database engine, setting up database access and logging in and accessing a database
- Troubleshooting, FAQs, and a list of resources and contacts

Advanced Operations Guide

Advanced Operations Guide provides detailed information at the administrative level, including the steps to perform common procedures and several new ones. Topics include:

- checking database consistency
- performing periodic backups
- configuring network protocols and understanding network topologies
- working with database security
- basic configuration guidelines
- configuration options reference
- moving, renaming, compressing and rebuilding files

SQL Engine Reference

SQL Engine Reference gives database programmers a complete reference guide to the SQL relational database language. It also covers SQL engine parameters and limitations.

Status Codes and Messages

Status Codes and Messages documents all possible status codes and numbered messages that can be received when using Pervasive software.

The *Status Codes and Messages Quick Reference* card is also included with your complete documentation set.

Pervasive Products and Services

Pervasive Products and Services provides an outline of how to work with Pervasive Software and describes the Pervasive.SQL database family.

Online Help

Pervasive.SQL comes with a full set of online documentation, available in the Pervasive program group on your **Start** menu.

You can download other formats and related documentation from this address:

http://www.pervasive.com/support/technical/product/psql2k.tml

You can also find additional information on the web site at:

http://www.pervasive.com/portals/psql.tml.

File System Security

The Pervasive.SQL engine adheres to the file system security defined by the specific operating system, such as Windows NT File Sharing or Novell Storage Services.

Note There is no user authentication when connecting to remote Workgroup engines. Users can bypass Windows security by opening a file through Btrieve using a UNC path name.

chapter

2

Using Pervasive.SQL

A Walk-through of Basic User Tasks

If you have not already done so, install Pervasive.SQL 2000i by following the instructions in *Getting Started with Pervasive.SQL*.

This chapter cover the basic tasks you need to know to work with Pervasive.SQL databases.

This chapter includes the following sections:

Starting and Stopping the Database Engine

This section outlines how to start and stop the Pervasive.SQL 2000i engine. For most engine configuration parameters, you need to stop and restart the engine in order for a particular change in your configuration to take effect.

To start and stop the MicroKernel, follow the instructions for your platform:

- "Starting and Stopping Services on Windows Server" on page 2-3
- "Starting and Stopping Services on NetWare" on page 2-4
- "Starting and Stopping Services on Unix" on page 2-5

Note *Btrieve v6.15 Users:* If you have any log (.LOG) files you want to roll forward, you must do so before you load the Pervasive.SQL 2000i MicroKernel, which uses a different logging scheme. You must use both the v6.15 or earlier engine and its accompanying Roll Forward Utility. Refer to your Btrieve v6.15 or earlier documentation for information about logging and instructions about how to roll files forward.

Starting and Stopping Services on Windows Server

In Windows NT or 2000, the MicroKernel runs as a service. The service is loaded as part of the installation process and is set to be always available if you followed the Typical installation.

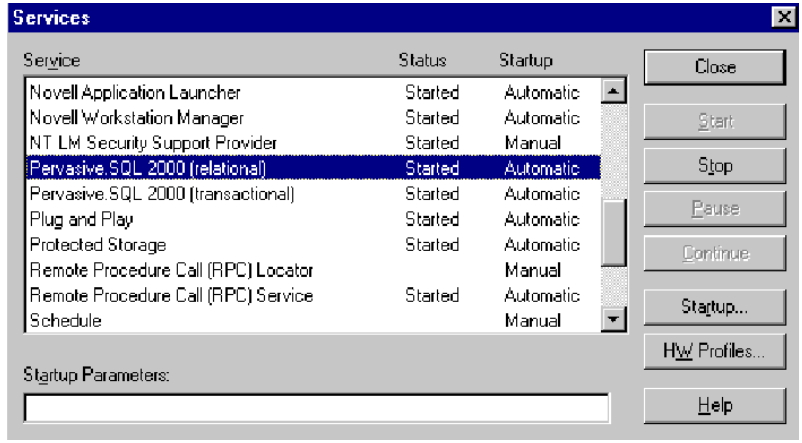
➤ To start or stop the Database Server on Windows NT or 2000

1. Click the **Start** menu, point to **Settings** and select **Control Panel**.
2. On Windows NT, double-click on the **Services** icon.

   On Windows 2000, point to **Administrative Tools**, then to **Services**.

   A dialog box similar to Figure 2-1 displays:

Figure 2-1 Windows NT Services Dialog Box



3. Select **Pervasive.SQL 2000** (**relational**) and **Pervasive.SQL 2000** (**transactional**) services from the list of services and click **Start**.

   By default, the MicroKernel Database Engine allocates resources and is ready to service clients.

➤ To stop the Database Server on Windows NT or 2000

1. Click the **Start** menu, point to **Settings** and select **Control Panel**.
2. On Windows NT, double-click on the **Services** icon.

   On Windows 2000, point to **Administrative Tools**, then to **Services**.

3 Select **Pervasive.SQL 2000i (relational)** and **Pervasive.SQL 2000i (transactional)** from the list of services and click **Stop**.

Starting and Stopping Services on NetWare

➤ To start the Database Engine on NetWare

Note You must load AFTER311.NLM before you load the MicroKernel.

Enter each of the following commands separately from the console window:

```
BSTART
MGRSTART
```

By default, the MicroKernel allocates resources and is ready to service clients.

Please note that Pervasive.SQL supports NetWare Storage Services (NSS) volumes on NetWare 5.0 and up. NSS volumes must be mounted before accessing the server with one of the database engines.

As an example:

```
LOAD NSS
MOUNT ALL
SYS:ETC\INITSYS.NCF
BSTART
MGRSTART
```

Also, please note that database updates performed against data files on NSS volumes may run more slowly than with earlier versions of NetWare. As noted in Novell TID 2952147 (http://www.novell.com), "NSS is optimized for reading files." Updates "will almost always perform a little faster on the legacy file system."

Based on this information, you may wish to store frequently-updated data files on regular NetWare volumes rather than NSS volumes.

➤ To stop the Database Engine on NetWare

Enter each of the following commands separately from the console window:

```
MGRSTOP

BSTOP
```

Note Never enter the **BSTOP** command before the **MGRSTOP** command.

Tip If NetWare does not allow you to stop the engines with these commands due to dependencies on other modules, first enter the command BTRV UNLINK, then perform the commands noted above.

Starting and Stopping Services on Unix

In Unix, the MicroKernel runs as a daemon. The daemon is loaded as part of the installation process and is set to be always available if you followed the Typical installation.

You must be logged in as the `root` user to start and stop the Pervasive.SQL 2000i daemon process. While we strongly recommend you use the supplied shell script `psql`, you may start and stop individual services by using the `mkded` and `sqlmgr` utilities.

➤ To start the Database Engine on Unix

Enter the following at the command line:

`/etc/rc.d/init.d/psql start` for Linux

`/etc/init.d/psql start` for Solaris

➤ To stop the Database Engine on Unix

Enter the following at the command line:

`/etc/rc.d/init.d/psql stop` for Linux

`/etc/init.d/psql stop` for Solaris

Granting Administrative Rights for the Database Engine

This section begins by outlining those Pervasive.SQL 2000i tasks that require administrative-level access at the operating system level and those that do not. The section then walks you through the steps to grant a user administrative-level access for each of the supported operating systems.

Note The following section only applies to the Server engine.

Tasks Requiring Administrative Rights

Administrative-level rights are required to:

- create and configure named databases and tables
- set engine configuration options
- view and set engine monitoring values
- view certain engine configuration settings

How Administrative Rights are Granted

To have administrator-level access you must:

- possess administrator-level rights on the computer itself, or
- be a member of the operating system group Pervasive_Admin.

Note For Unix servers, administrator-level rights can only be granted by using the btadmin utility to add users and passwords to the btpasswd file.

The Pervasive_Admin option is offered so that you can grant users administrative rights to the database engine without granting them administrative rights to the operating system where the database engine resides.

Rights Provided to Normal Users

Runtime-only access enables a user without administrator-level rights to perform such functions as:

- extract a list of DSNs
- extract a count of DSNs
- extract information on a DSN

- extract information on the location of the DBnames configuration file (dbnames.cfg)
- connect to databases
- retrieve, update, insert, and delete data (as permitted by database security)

To grant a user administrative rights, follow the instructions for your platform:

- "Granting Administrative Rights on Windows Server" on page 2-9
- "Granting Administrative Rights on NetWare 3.2" on page 2-11
- "Granting Administrator Rights on NetWare 4.2 or 5.0" on page 2-11
- "Granting Administrator Rights on Unix" on page 2-12
- "Logging in as Administrator" on page 2-13

Granting Administrative Rights on Windows Server

Users who are members of **Pervasive_Admin** or of **Administrators** are permitted to perform administrative tasks on the database engine.

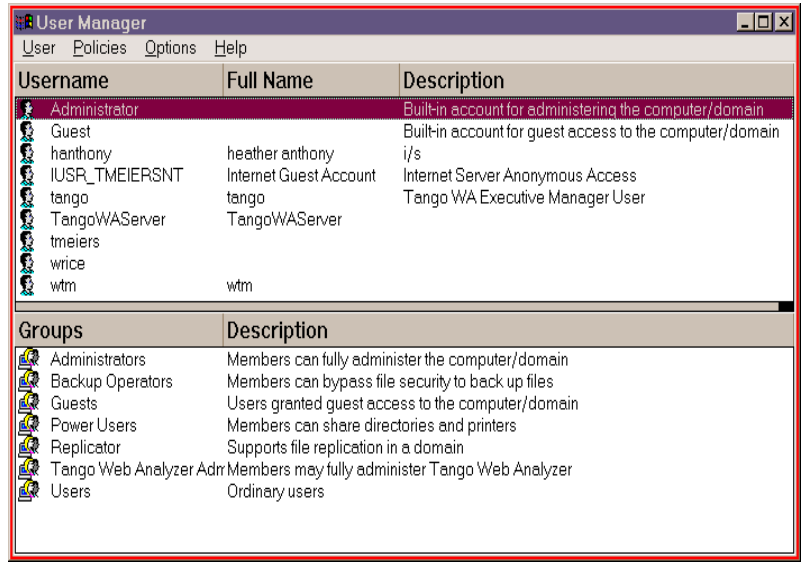
➤ To grant a user database administrator rights on Windows NT

Note You must be logged onto the Windows server as a user with administrative rights.

1 Open the Windows NT User Manager (from the **Start** menu, select **Programs**, then **Administrative Tools**, then **User Manager.**)

A dialog box like this appears:

Figure 2-2 User Manager Dialog Box



2 From the **User** menu, select **New Local Group**. A dialog box appears.

3 Type in `Pervasive_Admin` as the group name. You may now add users to this list by clicking **Add** and selecting them. When finished, click **OK**.

Users who are members of Pervasive_Admin or Administrators are permitted to perform administrative tasks on the database engine.

➤ To grant a user database administrator rights on Windows 2000

Note You must be logged onto the Windows server as a user with administrative rights.

1. From the **Start** menu, select **Settings | Control Panel | Users and Passwords**.
2. Select **Advanced** tab, then select **Advanced User Management**.
3. Select **Groups** folder. From the menu, choose **Action | New Group**.
4. Type in Pervasive_Admin as the group name. Click on **Add...** to add users to this group. When you are finished, click **Create**.

Granting Administrative Rights on NetWare 3.2

➤ To grant a user database administrator rights on NetWare 3.2

Note You must have NetWare administrator rights to change a user's or a group's rights.

Note The Novell Client must be installed. If it is not, contact your network administrator.

1. Run the **System** utility from the `public` directory in `syscon:` volume.
2. Select **Group Information.** The **Group Names** dialog box appears.
3. Press the insert key. The **New Group Names** dialog box appears.
4. Enter Pervasive_Admin as your new group name.
5. From the **Available Topics** menu, select **User Information**. The **User Names** dialog box appears.
6. Select the user you wish to add, and then **User Information/ Groups Belonged To**. A dialog box appears.
7. Press the insert key. The **Groups Not Belonged To** dialog box appears.
8. Select Pervasive_Admin.

Granting Administrator Rights on NetWare 4.2 or 5.0

➤ To grant a user database administrator rights on NetWare 4.2 or 5.0

1. Run the **Nwadmin** utility from the `public` directory of the `syscon:` volume. The NetWare Administrator dialog box appears.
2. Select the **Object** menu.
3. Select **Create.** The **New Object** dialog box opens.
4. Select **Group** and click **OK**. The Create Group dialog box opens. Enter Pervasive_Admin as the name of the group, and click **Create**.

5 Once Pervasive_Admin is created, you can double-click on the name, and the **Details** dialog box for that group will appear. **Members** is an options page (listed in the right-hand panel of the dialog box); click on it and then click **Add** to select a user and click **OK** to close the dialog box.

6 To grant a group of users (such as Pervasive_Admin) specific rights, bring up the **Details** dialog box for that group, and click on the **Rights to Files and Directories** options page on the right.

Note Users may not log in to Pervasive.SQL 2000i if they are in violation of any restrictions set in their NetWare user profile. These restrictions may be accessed and changed by double-clicking the user name in the NDS tree, which will bring up the **Details** dialog box for that user. The **Login Restrictions**, **Login Time Restrictions** and **Network Address Restrictions** options pages appear in the right-hand panel of the dialog box; these may be reviewed to ensure no violations are occurring.

Granting Administrator Rights on Unix

➤ To grant a user administrator rights on Unix

A user cannot remotely administer a Unix server engine unless the user has first been set up as a database user with administrative rights. You can perform this task by using the btadmin utility at the server command line.

1 Login to the Unix server as `root` or `psql`. No other user is permitted to run btadmin.

2 Create a new user with administrative rights by running btadmin:

```
btadmin -p passwd a+ user_name
```

For example, if you wanted to create an administrative user "tim" with password "tim56", you would enter the following command:

```
btadmin -p tim56 a+ tim
```

Note Users created with btadmin are not related to Unix system users. These users are known only to the database engine.

Logging in as Administrator

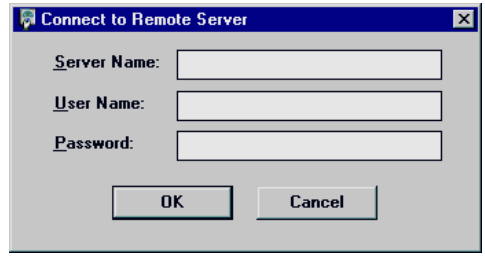
➤ To connect to a remote Pervasive.SQL 2000i server

1 Use the Monitor or Configuration utility to connect to a remote server.

Note The Maintenance, Rebuild and Function Executor do not require you to log on to connect to a remote server.

A dialog box like this appears: Enter your operating system user name and password, and click **OK**.

Figure 2-3 Connect to Remote Server Dialog Box



The password is encrypted before being sent over the network using a unique and pre-defined encryption key. The Pervasive.SQL 2000i engine unpacks and decrypts the user name and password, and verifies access. It then returns a status code to the client indicating the success or failure of the verification.

Setting Up ODBC Database Access

This section covers the steps to set up access to a database. We will review some conceptual information before detailing the steps to perform the tasks of setting up Client and Engine (Server) DSNs and creating a new database using Pervasive.SQL 2000i.

This section covers the following topics:

Basic Concepts ODBC Standard

Pervasive.SQL adheres to the Microsoft standard for ODBC database connections. According to the standard, applications must connect to databases through Data Source Names (DSNs) defined in the operating system.

Every Pervasive.SQL database that you expect to access using ODBC must have a Data Source Name (DSN) available on the same computer as the database engine, and (if applicable) another DSN on the client computer. The only exceptions are Pervasive tools, which can access remote databases without using DSNs on the client computer. A DSN created on the same machine as the database engine is called an *Engine* DSN. A DSN created on the client machine is called a *Client* DSN.

Note Pervasive.SQL databases that are accessed only through Btrieve do not need DSNs. However, in this case, the database is not visible in PCC nor can it be manipulated using PCC. Pervasive recommends using Data Dictionary Files (DDFs) and DSNs with all databases, including ones accessed only through Btrieve, to make them easier to manipulate.

Figures 2-4 and 2-5 show the DSNs that are needed for each configuration. Workstation and Workgroup engines use only an Engine DSN on the local computer where the database engine is installed. You cannot use Client DSNs with Workstation or Workgroup engines.

Figure 2-4 DSNs Needed for Server Engine

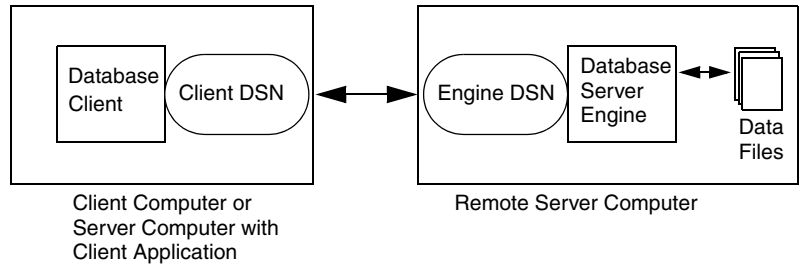
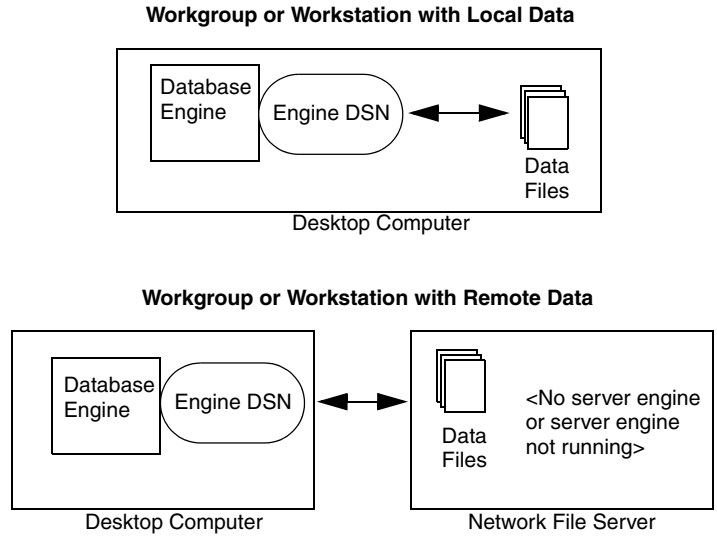


Figure 2-5 DSNs Needed for Workstation/Workgroup Engine



Servers and Clients

Pervasive.SQL servers are also clients. The client components of Pervasive.SQL are installed with every server engine or workgroup engine. So you can use your server machine to connect to other servers as a client.

Pervasive.SQL clients can connect to remote machines where a Pervasive.SQL server engine is installed.

Note For NetWare systems, use the following path UNC structure:
`\\server\vol1:\path`

For Unix systems (when Samba is not used), use the following UNC path structure:
`\\server\$PVSW$\[path on Unix from root '\' dir]`

For Windows systems, use the following UNC path structure:
`\\server\sharename\path`

Data Source Names

The client-server architecture calls for the naming of each specific data set so that it can be referred to by a well-known name. There are generally three ways to create DSNs:

1. Create an Engine DSN from the server console. Workstation and Workgroup engines require only Engine DSNs.
2. Create an Engine DSN remotely from a client machine.

Note On NetWare, the *only* way to create an Engine DSN is to create it remotely from the client.

3. Create a Client DSN on each client machine.
   While Pervasive tools can access remote databases without a client DSN present on the client machine, ODBC-based applications such as Microsoft Excel and Microsoft Access cannot do so. You must create a client DSN on each client computer that needs to access network databases from local applications.

Components of a Database Name

There are two components of a database name. Pervasive.SQL uses an internal Database Name (DBNAME) to identify the location of the Data Dictionary Files (DDFs) and the data files for each database. An ODBC Data Source Name (DSN) entry refers to one DBNAME.

You may set up more than one DSN that refers to the same DBNAME. If the physical location of the data files on the server is

changed, only the DBNAME needs to be updated. All DSNs remain unchanged.

What to Know if You are Recreating DSNs

Btrieve application users

Applications that only use Btrieve do not require you to follow the procedures in this section. However, you must follow these procedures to access Pervasive.SQL through ODBC. This includes the SQL Data Manager utility and all Microsoft applications. If you are not sure whether your application uses ODBC, check with your network administrator.

Pervasive.SQL v7 users

You must recreate all DSNs created with Pervasive.SQL 7 to access them in Pervasive.SQL 2000i. However, you do not need to rename existing Named Databases. To re-create DSNs, follow the instructions provides in "Deleting DSNs" on page 2-53.

Prerequisites to Setting up Database Access

Before you begin, verify the following:

- The data files and DDFs already exist. If these do not exist, ask your network administrator to create them, or refer to *Advanced Operations Guide* for information on creating a database.
- If the data files are located on a remote server, you must know the name of the server. You must also know the location of the data files on the server. If the database already has a DBNAME, this information can be used instead of the location of the data files. If you do not have this information, see your network administrator.
- If the data files are located on a remote server and an Engine DSN has not been created on the server, you must possess administrator rights on the remote machine that houses the database you wish to access, in order to set up an Engine DSN. Setting up local access requires OS system rights to create a System DSN.

Note If you are using Workgroup engine to access data files on a remote file server where no database engine is running, then you should follow the instructions provided in "Setting Up Database Access on a Windows Server or Workgroup/Workstation" on page 2-19. With the Workgroup engine, you will create a local Engine DSN but specify a mapped drive path or UNC path for the location of the data files.

The sections that follow allow you to set up database access as long as the conditions above are met. The following topics are provided:

- "Setting Up Database Access on a Windows Server or Workgroup/Workstation" on page 2-19
- "Setting Up Database Access on a NetWare Server" on page 2-28
- "Setting Up Database Access on Unix" on page 2-36
- "Setting Up Client Access" on page 2-39

Setting Up Database Access on a Windows Server or Workgroup/Workstation

➤ To Set Up Database Access on a Windows Server or Workgroup/Workstation

Note If you have not done so already, please review "Prerequisites to Setting up Database Access" on page 2-17 before proceeding.

1 Open Pervasive Control Center (PCC):
Start | Programs | Pervasive | Pervasive Control Center

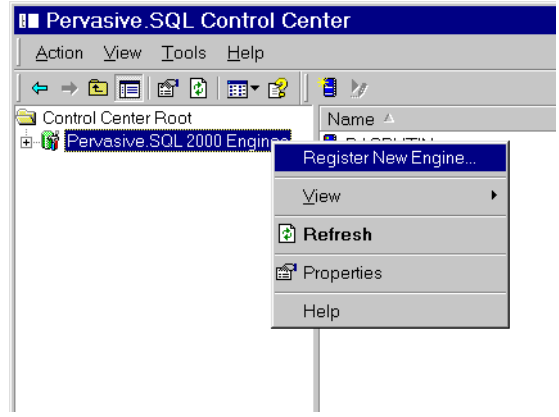
2 Double-click on **Pervasive.SQL 2000i Engines.**

If you are using Workgroup/Workstation, skip to step 5 now.

If you are not sitting at the Windows server you want to work with, and you do not see the name of the server you want to connect to, you must register the remote server with PCC. To do so, right-click on **Pervasive.SQL 2000i Engines** and select **Register New Engine**. Type in or choose the server you want to connect to.

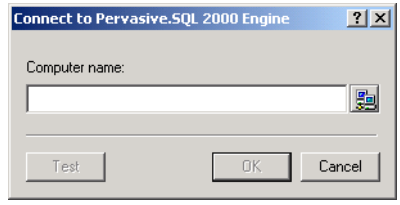
The Namespace is the column on the left side of PCC window that contains a list of database engines that have been registered.

Figure 2-6 Registering a New Engine



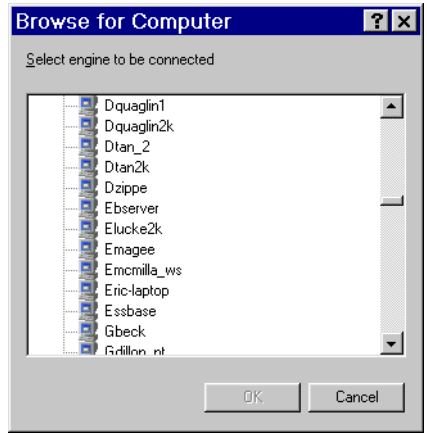
A dialog is displayed that allows you to choose the machine name where the Pervasive.SQL server resides.

Figure 2-7 Choosing a Computer Name



3 If you are unsure of the server name, click the button located to the right of the name field and browse from the Network list that appears, as shown below.

Figure 2-8 Choosing a Computer Name from a Network List



4 Enter the computer name in the field and press **OK**.

5 Double-click the icon representing the engine you are working with. Then double-click the **Databases** folder associated with the selected engine.

6 Inspect the databases that are listed to determine whether the database you wish to access has already been set up with an Engine DSN. If you do not see the database you wish to access, proceed to step 7.

If the database you wish to access is visible, then it already has an Engine DSN and you should consult table 2-1 to determine your next step.

Table 2-1 Next Step if Engine DSN Already exists

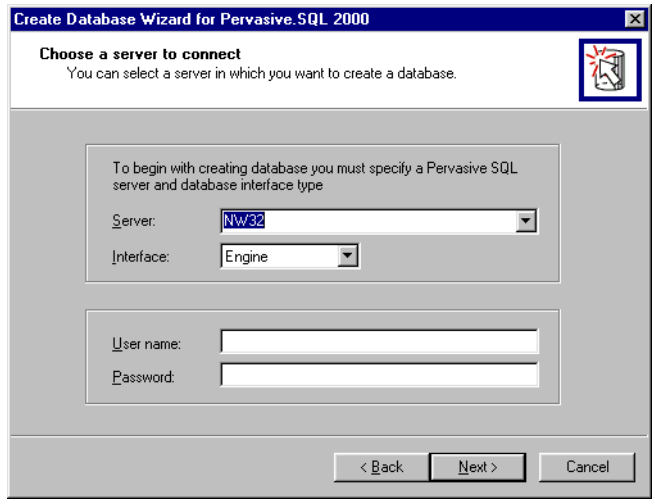
If you are using this engine ...	... then this is your next step:
Server	Go to "Setting Up Client Access" on page 2-39 for instructions on how to set up the client machines with Client DSNs.
Workgroup/Workstation	The database is fully set up. Refer to "Accessing Data via ODBC From Other Applications" on page 2-46.

7 Create a New Database for the engine:
Right-click on the database folder underneath the server engine name and select **New Database** from the shortcut menu.

If you are working with a Server engine and you are not logged into the server as a user with system administrator permissions, you see the login screen shown below.

If you are using Workstation/Workgroup, or you are using the server console or are already logged into the server with system administrator permissions, skip to step 9 as shown in Figure 2-10.

Figure 2-9 Create an Engine DSN and Connect to a Server Database

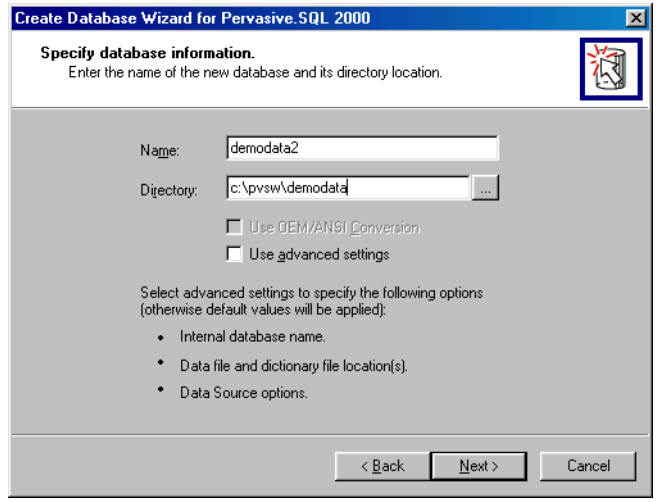


8 The **Server:** field should indicate the server that you are currently working with. Choose Engine for the **Interface** type, and enter your operating system user name and password for that server, in the **User name** and **Password** boxes.

Remember that you must have administrator-level rights on the server operating system to be able to complete this task.

Click **Next** to move to the next screen, shown below.

Figure 2-10 Naming the New Database on the Server



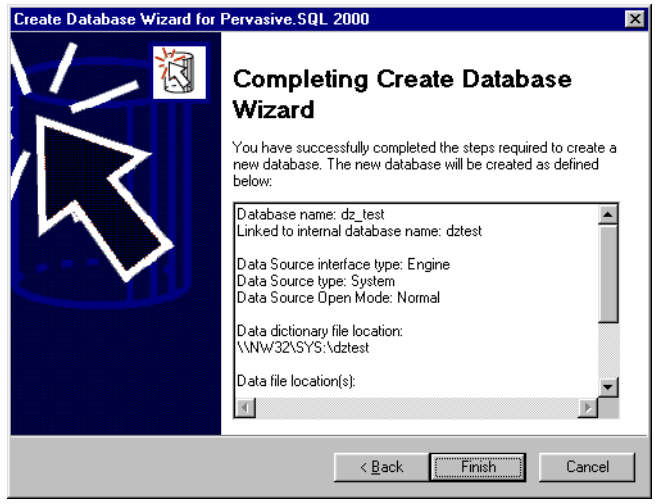
9 In the **Name** field, specify an Engine DSN name for the database. This name will appear in the database listing in PCC after you complete this task.

In the **Directory** field, specify the location of the data files.

Note If you are working with a Server engine, you must specify a full path (not a mapped drive) that makes sense on the server. The server engine cannot interpret mapped drive paths. For example, if you have a mapped drive S:\data1\datafiles on your client, referring to the folder C:\servers\data1\datafiles on the server, you must enter the server-based directory, C:\servers\data1\datafiles.

If you wish to use any advanced features, such as specifying that the DDFs and data files are not located in the same directory, that the DDFs and data files are located in multiple directories, or modifying the default DBNAME settings, you must click the **Use advanced settings** option. Click **Next**. If you clicked the **Use advanced settings** option, proceed to step 11.

Figure 2-11 Create Database Wizard Complete



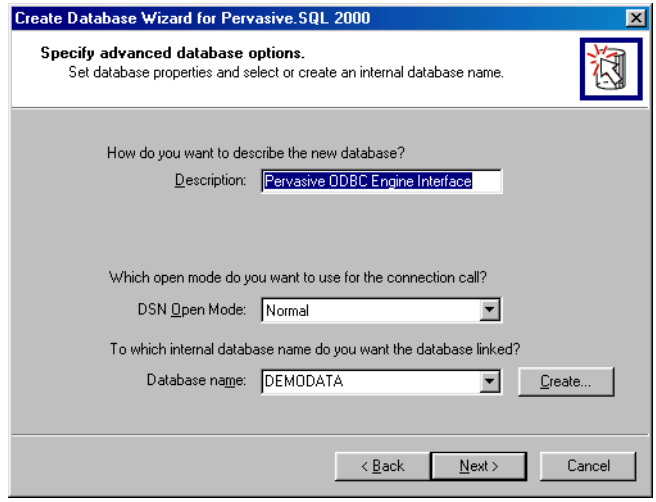
10 The final wizard step verifies the successful creation of the database. Click **Finish**. The database is ready to access.

If you just created a DSN for a Server engine, you must proceed to setting up client DSNs on each client workstation as explained in "Setting Up Client Access" on page 2-39.

Advanced settings procedure

11 If you clicked the **Use advanced settings** option, you see a window as shown below.

Figure 2-12 Advanced Settings Window



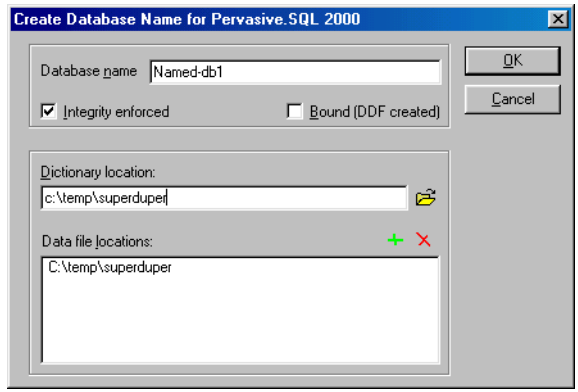
12 In the Advanced Settings window, you can type a description for the new database.

In the **DSN Open Mode** field, choose Normal unless you have reviewed the other options as described in *Advanced Operations Guide*.

In the **Database name** field, choose the DBNAME of the database for which you wish to create the Engine DSN. If you find the DBNAME you want, click **Next** and return to step 10.

If the DBNAME you want to access is not listed, click the **Create** button to create an internal DBNAME for the database. Click **Next** and continue with the next step.

Figure 2-13 Create DBNAME Window



13 In the Create Database Name window, type in the internal name you want to assign to the database. This name does not need to be the same as the DSN. This name is not displayed in PCC nor is it exposed to users trying to connect to the database during runtime.

For more information in **Integrity enforced** and **Bound**, see "Bound Databases and Enforced Integrity" on page 2-56

Click **Integrity enforced** if you plan to enforce the referential integrity rules defined in the database. Usually it is a good idea to do so.

Click **Bound** (**DDFs created**) if you want to prevent any other databases from being associated with these DDFs and data files. Checking this option creates a new set of empty DDFs. Do not click this option if you are setting up access to an existing database.

In the **Dictionary location** box, type or select the location of the DDFs. This location can be the location of existing DDFs if you are setting up an existing database for use with ODBC, or it can be an empty directory if you are creating a new database.

- For Server databases, this must be a full path name on the server—do not enter or browse a mapped drive path. Browsing for the location only works if you are seated at the server or you are browsing a local directory tree that is identical to the directory tree on the server.
- Workstation users must enter a full path name on a local hard drive.
- Workgroup users may enter a mapped drive or UNC name.

In the **Data file location** box, you must accept the default or browse to the location of the data files. This location can be the location of existing data files if you are setting up an existing database for use with ODBC, or it can be an empty directory if you are creating a new database.

- For Server databases, this must be a full path name on the server—do not enter or browse a mapped drive path. Browsing for the location only works if you are seated at the server or you are browsing a local directory tree that is identical to the directory tree on the server.
- Workstation users must enter a full path name on a local hard drive.
- Workgroup users may enter a mapped drive or UNC name.

Click **OK** to close the Create Database Name window. Back in the Create Database Wizard, click **Next** and go to step 10.

Setting Up Database Access on a NetWare Server

➤ To Set Up Database Access on a NetWare Server from a Client Workstation

Note This scenario is the only option for NetWare. You must create both the Client and the Engine DSNs from the client workstation using PCC or ODBC Administrator.

Note If you have not done so already, please review "Prerequisites to Setting up Database Access" on page 2-17 before proceeding.

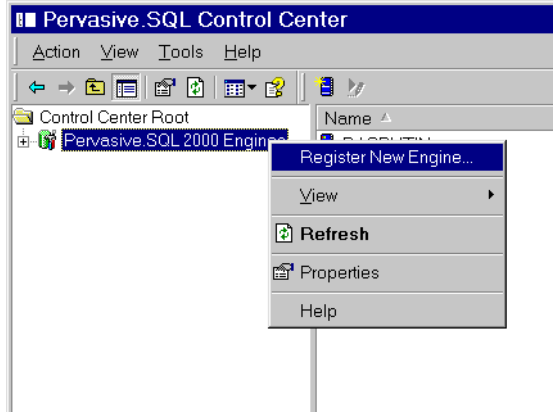
1 Open Pervasive Control Center (PCC):
Start | Programs | Pervasive | Pervasive Control Center

2 Double-click on **Pervasive.SQL 2000i Engines**.

You must register the remote NetWare server with PCC. To do so, right-click on **Pervasive.SQL 2000i Engines** and select **Register New Engine**. Type in or choose the server you want to connect to.

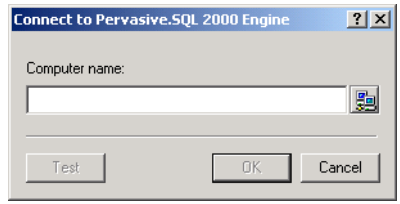
The Namespace is the column on the left side of PCC window that contains a list of database engines that have been registered.

Figure 2-14 Registering a New Engine



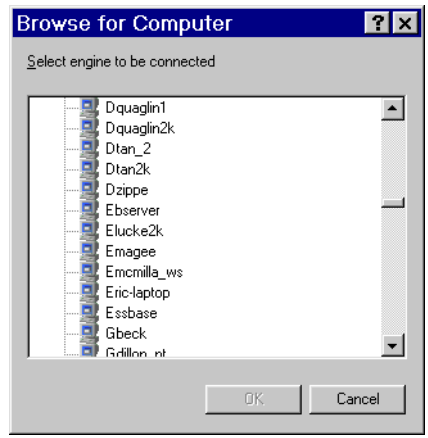
A dialog box is displayed that allows you to choose the machine name where the Pervasive.SQL server resides.

Figure 2-15 Choosing a Computer Name



3 If you are unsure of the server name, click the button located to the right of the name field and browse from the Network list that appears, as shown below.

Figure 2-16 Choosing a Computer Name from a Network List



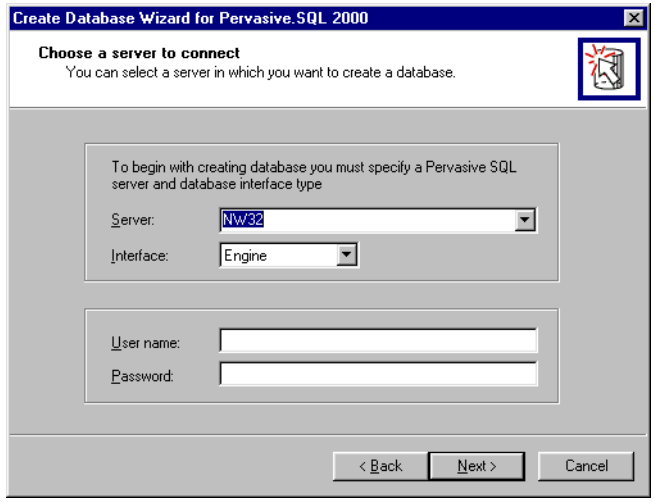
4 Enter the computer name in the field and press **OK**.

5 Back in the main screen of PCC, double-click the icon representing the server you are working with. Then double-click the Databases folder associated with your server.

Inspect the databases that are listed to determine whether the database you wish to access has already been set up with an Engine DSN. If the database already appears, then it already has an Engine DSN and you need only set up the client machines with Client DSNs. If you do not see the name of the database that you want to connect to, then continue with the step that follows to set up an Engine DSN.

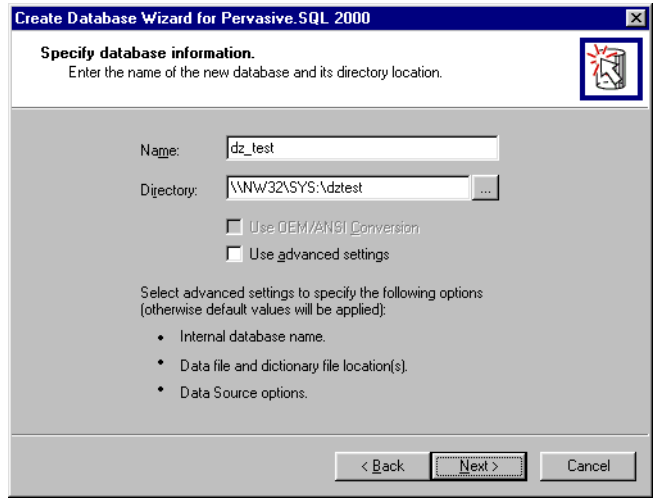
6 Create a New Database for the server engine:
Right-click on the database folder underneath the server engine name and select **New Database** from the shortcut menu. The Create Database Wizard appears as shown here.

Figure 2-17 Create an Engine DSN and Connect to a Server Database



7 The Server name should indicate the server that you are currently working with. Choose Engine for the DSN type, and enter your operating system User Name and Password for that server. Remember that you must have administrator-level rights on the server operating system to be able to complete this task. Click **Next** to move to the next screen, shown below.

Figure 2-18 Naming the New Database on the Server



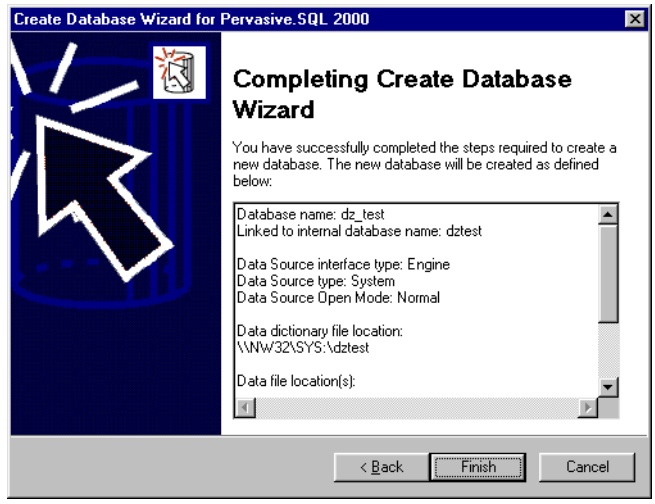
8 In the **Name** field, specify an Engine DSN name for the database. This name will appear in the database listing in PCC after you complete this task.

In the **Directory** field, specify the location of the data files.

Note The server engine cannot interpret mapped drive paths. You must specify a full path (not a mapped drive) that makes sense on the server. For example, if you have a mapped drive S:\data1\datafiles on your client, referring to the folder SYS:\servers\data1\datafiles on the server, you must enter the server-based directory, \\servname\SYS:\servers\data1\datafiles.

If you wish to use any advanced features, such as specifying that the DDFs and data files are not located in the same directory, that the DDFs and data files are located in multiple directories, or modifying the default DBNAME settings, you must click the **Use advanced settings** option. Click **Next**. If you clicked the **Use advanced settings** option, proceed to step 11.

Figure 2-19 Create Database Wizard Complete

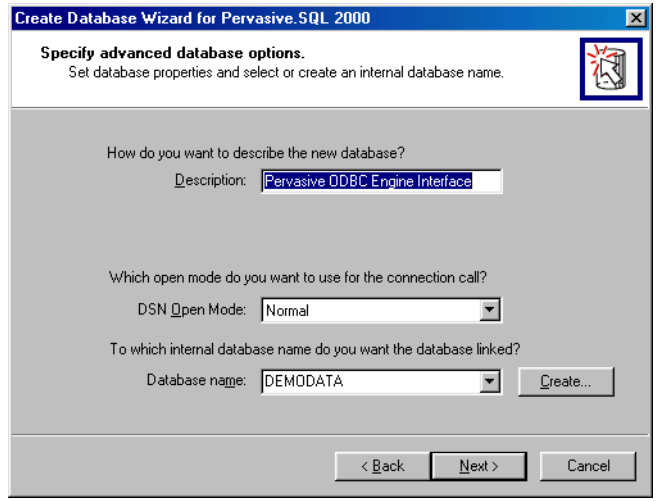


9 The final wizard step verifies the successful creation of the database. Click **Finish**. The database is ready to access. Proceed to setting up client DSNs as explained in "Setting Up Client Access" on page 2-39.

Advanced settings procedure

10 If you clicked the **Use advanced settings** option, you see a window as shown below.

Figure 2-20 Advanced Settings Window



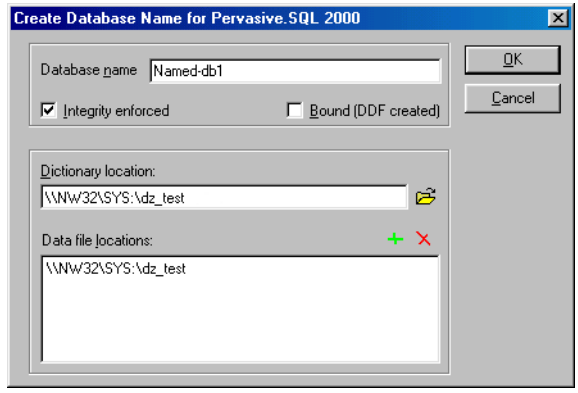
11 In the Advanced Settings window, you can type a description for the new database.

In the **DSN Open Mode** field, choose Normal unless you have reviewed the other options as described in *Advanced Operations Guide*.

In the **Database name** field, choose the DBNAME of the database for which you wish to create the Engine DSN. If you find the DBNAME you want, click **Next** and return to step 10.

If the database you want to access is not listed, click the **Create** button to create an internal DBNAME for the database. Click **Next** and continue with the next step.

Figure 2-21 Create DBNAME Window



12 In the Create Database Name window, type in the internal name you want to assign to the database. This name does not need to be the same as the DSN. This name is not displayed in PCC nor is it exposed to users trying to connect to the database during runtime.

For more information in **Integrity enforced** and **Bound**, see "Bound Databases and Enforced Integrity" on page 2-56

Click **Integrity enforced** if you plan to enforce the referential integrity rules defined in the database. Usually it is a good idea to do so.

Click **Bound (DDFs created)** if you want to prevent any other databases from being associated with these DDFs and data files. Checking this option creates a new set of empty DDFs. Do not click this option if you are setting up access to an existing database.

In the **Dictionary location** box, type or select the location of the DDFs. As previously cautioned, do not enter or browse a mapped drive path because the server engine cannot interpret mapped drive paths. You must enter a full path name.

Caution Browsing for the location only works if you are seated at the server or you are browsing a local directory tree that is identical to the directory tree on the server. Ensure that you enter a full path name.

In the **Data file location** box, you must accept the default or browse to the location of the data files. As noted previously, browsing for the location only works if you are seated at the server or you are browsing a local directory tree that is identical to the directory tree on the server.

Click **OK** to close the Create Database Name window. Back in the Create Database Wizard, click **Next** and go to step 10.

Setting Up Database Access on Unix

➤ To Set Up a Named Database and Engine DSN from the Unix Server

Database names are created in Unix by using the `dbmaint` utility at the server. For a complete description of `dbmaint`, see "dbmaint" on page 4-17 or read the `dbmaint` man page.

1 To create an empty database, use the following at the command line:

```
dbmaint a | d | l [-b] [-i] [-e] -nDbname
    [-ldictpath] [-ddatapath]
```

The list of commands for dbmaint include:

`a` – add database name
`d` – delete database name
`l` – list all database names

Options include:

`-b` – create Bound database
`-i` – create database with Relational Integrity enforced
`-e` – do not create dictionary files for database
`-n`*DBName* – specify database name
`-l`*Dictpath* – specify dictionary path
`-d`*Datapath* – specify data path
`-a` – show full data in the DBNames list

For example, to create DBName TEST with relational integrity, type:

```
dbmaint a -i -nTEST
```

Note Unless *datapath* is specified, the new database is created in the default location, $PVSW_ROOT/data. Likewise, if *dictpath* is not specified, the dictionary is created in the default location.

- To delete an existing database, use the following at the command line:

  ```
  dbmaint d -nDbname
  ```

  For example, to delete the newly created database TEST, type

  ```
  dbmaint d -nTEST
  ```

- To list all existing databases:

  ```
  dbmaint l [-a]
  ```

2 To set up an Engine DSN, modify the following files:

- `${PVSW_ROOT}/etc/bti.ini`.

  SQLMGR required settings:

  ```
  [SQLManager]
     MgrPort=1583
     MgrUseTransport=TCP
  ```

- `${PVSW_ROOT}/etc/odbc.ini`

Note The value of `${PVSW_ROOT}` is typically `/usr/local/psql`.

Server data source – the one to which remote calls will be redirected:

```
[DSN name]
   Driver=/usr/local/psql/lib/libsrde.so
   Description=Test Pervasive database
   DBQ=DBName
```

In addition, each data source should be mentioned in the section `[ODBC Data Sources]` as in the following example:

```
[ODBC Data Sources]
   dsnName1=Pervasive.SQL data base
   dsnName2=Pervasive.SQL data base
```

For example, if you have in `bti.ini`:

```
[MyDSN]
   Driver=/usr/local/psql/lib/libsrde.so
   Description=test
   DBQ=MyDB
```

then your `odbc.ini` should have:

```
[ODBC Data Sources]
   MyDSN=Pervasive.SQL database
```

Note Because Unix is case sensitive, the `[DSN name]` must be input exactly as listed under `[ODBC Data Sources]`.

- The engine can also be created using the `dsnadd` utility by typing the following at the command line:

```
% dsnadd -dsn=DSNname -db=DBName
```

An easy way to verify DBName and DSN configuration settings is to run the supplied odbctest program:

```
% /user/local/psql/bin/odbctest DSN=DEMODATA
```
on Linux

```
% /opt/PVSWpsql/bin/odbctest DSN=DEMODATA
```
on Solaris

3 Proceed to setting up client DSNs as explained in "Setting Up Client Access" on page 2-39.

Setting Up Client Access

➤ To make a Pervasive Client able to access a remote Pervasive.SQL Server database

This procedure explains how to create a Client DSN. Client DSNs are used only on client workstations when connecting to a database server. Client DSNs are not used with the Workstation or Workgroup database engine.

Note If you have not done so already, please review "Prerequisites to Setting up Database Access" on page 2-17 before proceeding.

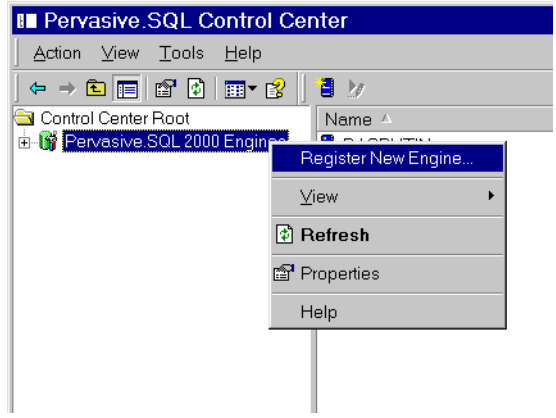
1 Open Pervasive Control Center (PCC):
Start | Programs | Pervasive | Pervasive Control Center

2 Double-click on **Pervasive.SQL 2000i Engines**.

If you do not see the name of the server you want to connect to, you must register the remote server with PCC. To do so, right-click on **Pervasive.SQL 2000i Engines** and select **Register New Engine**. Type in or choose the server you want to connect to.

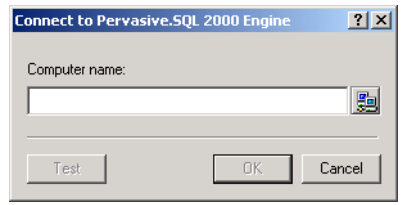
The Namespace is the column on the left side of PCC window that contains a list of database engines that have been registered.

Figure 2-22 Registering a New Engine



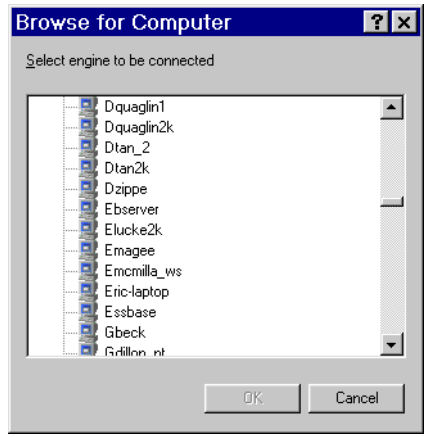
A dialog box is displayed that allows you to choose the machine name where the Pervasive.SQL server resides.

Figure 2-23 Choosing a Computer Name



3 If you are unsure of the server name, click the button located to the right of the name field and browse from the Network list that appears, as shown below.

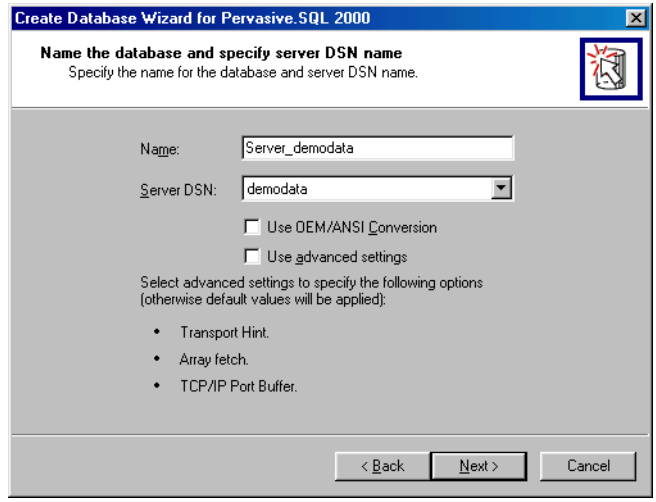
Figure 2-24 Choosing a Computer Name from a Network List



4 Enter the computer name in the field and press **OK**.

5 Back in the main screen of PCC, double-click the icon representing the server you are working with. Then right-click the **Databases** folder associated with the server you have selected. Choose **New Database** to start the Create Database Wizard.

Figure 2-25 Create Database Wizard—Client DSN



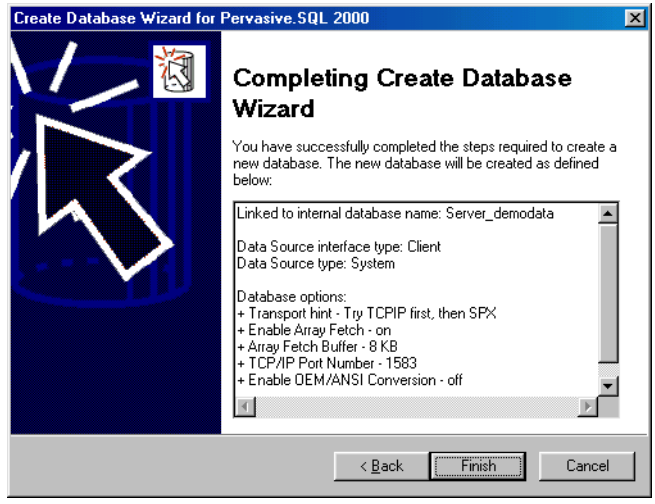
6 In the **Name** field, enter a descriptive name for the DSN. This name will be displayed in the ODBC Administrator and to users attempting to connect to ODBC data sources. The maximum length is 32 characters.

In the **Server DSN** field, select the Engine DSN on the server that you want to associate with the Client DSN you are creating.

Click **Use advanced settings** only if you have read about this option in *Advanced Operations Guide*.

Click **Next** to display the final confirmation screen.

Figure 2-26 Create Database Wizard—Client DSN Confirmation



7 Click **Finish** to complete the operation.

Note The Client DSN you have just created does not appear in PCC. It appears only in the ODBC Administrator as a System DSN. It also appears in the DSN list presented when local applications attempt to connect to ODBC data sources. Once you have created a Client DSN, you can remove or rename it from the ODBC Administrator, using the **Remove** or **Configure** buttons.

Tip You can set the option to have Client DSNs appear in PCC by right-clicking on the database engine is the list and selecting **View | Show Client Databases**. Once you have set this option, you can remove or configure it using PCC.

➤ Setting Up a Client DSN Using the ODBC Administrator

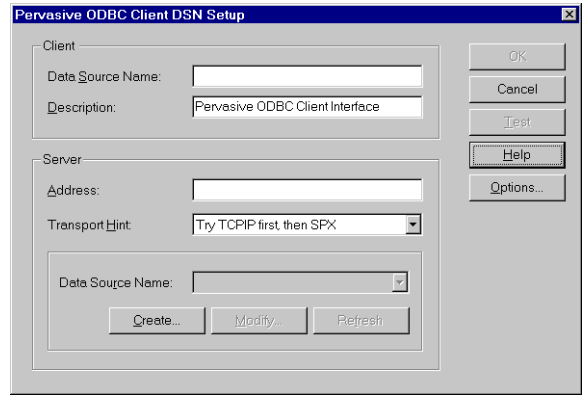
1 Open the ODBC Administrator (from the **Start** menu, select **Programs | Pervasive | Pervasive.SQL 2000i | Utilities | ODBC Administrator**).

2 Click on the **System DSN** tab, then on **Add**.

3 In the Drivers window, select **Pervasive ODBC Client Interface**.

The following dialog box appears:

Figure 2-27 Pervasive ODBC Client DSN Setup Screen

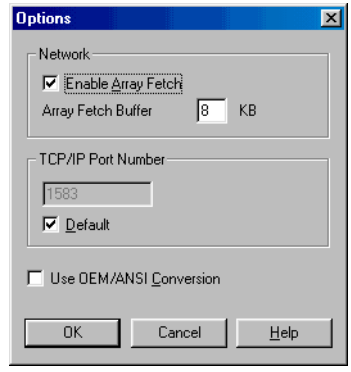


4 In the Client section, type in a DSN (with a maximum length of 32 characters) for the data source to which you wish to set up a connection. This DSN will help you identify the data source. It will be visible only on the current machine.

5 Type a description of the data source, if desired (with a maximum length of 80 characters).

6 If you want to enable OEM/ANSI conversion, click the **Options** button and make your selection in the dialog box that appears.

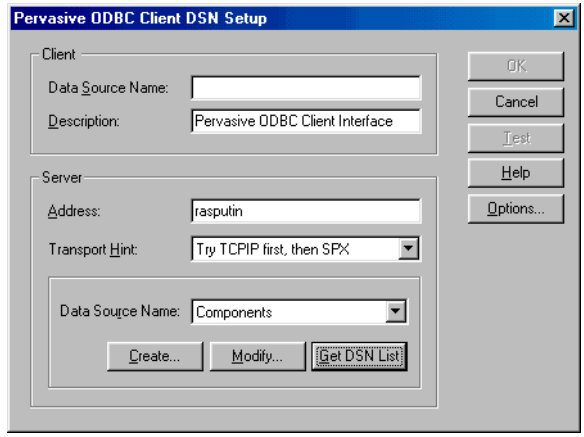
Figure 2-28 Pervasive ODBC Client DSN Options



Do not modify the **Network** settings or **TCP/IP Port Number** unless you have first reviewed the information about these Client DSN options in *Advanced Operations Guide*.

7 Click **OK** to return to the **Pervasive ODBC Client DSN Setup** dialog box.

Figure 2-29 Pervasive ODBC Client DSN Setup Screen #2



8 In the Server area, type in the host name of the computer where the data source resides. You can enter a machine name, TCP/IP address, or an IPX/SPX MAC address.

9 To use an existing database on the server, click the **Get DSN List** button and select the desired DSN from the drop-down list. In the **Server** area, **Data Source Name** refers to an Engine DSN on the server computer.

If no databases appear in the drop-down list, either you selected the wrong server, or you need to have your system administrator name the server databases and create Engine DSNs for each of them before you can access them.

10 Click **OK**.

11 You can now set up another Client DSN or click **OK** to exit the ODBC Administrator.

➤ Setting Up a Client DSN on a Unix Workstation

While it is possible to access a database from a Unix server by a Unix client, there are no Pervasive.SQL utilities (except `dsnadd`) that can be used on the client. A Unix client configuration would be used for independent applications, such as web applications.

To add a client data source, execute the following command:

`dsnadd -dsn=`*`myDSN`* `-desc=`*`datasource`*
`-host=`*`psqlhost`* `-sdsn=`*`svDSN`*

myDSN is a name you want to assign to the new Client DSN.

datasource is any string to describe the data source.

psqlhost is the name of the network host where your Pervasive.SQL database resides.

svDSN is the name of the Engine DSN on the Pervasive.SQL host.

Note The datasource on the server must be named first.

For example, to create a Client DSN named TEST on host NewDev, where the Engine DSN name for the database is NewTest, type

```
dsnadd -dsn=TEST -host=NewDev -sdsn=NewTest
```

For more information about the `dsnadd` utility, please see *Getting Started with Pervasive.SQL* (Server edition).

Accessing Data via ODBC From Other Applications

This section explains how to access data using Microsoft Access and Microsoft Excel.

The examples covered in this section are:

- "Accessing Data Using Microsoft Excel" on page 2-46
- "Accessing Data Using Microsoft Access" on page 2-49

Before You Begin

Does the Database Have a DSN Available?

- If you are connecting from a client workstation to a server, you must have a Client DSN defined on your workstation for the given remote database. Information on how to create a Client DSN is provided in "Setting Up Client Access" on page 2-39.
- If you have a Workstation or Workgroup engine installed on your computer, you must have an Engine DSN defined on your computer for either local or remote databases. Information on how to create an Engine DSN is provided in "Setting Up Database Access on a Windows Server or Workgroup/ Workstation" on page 2-19.

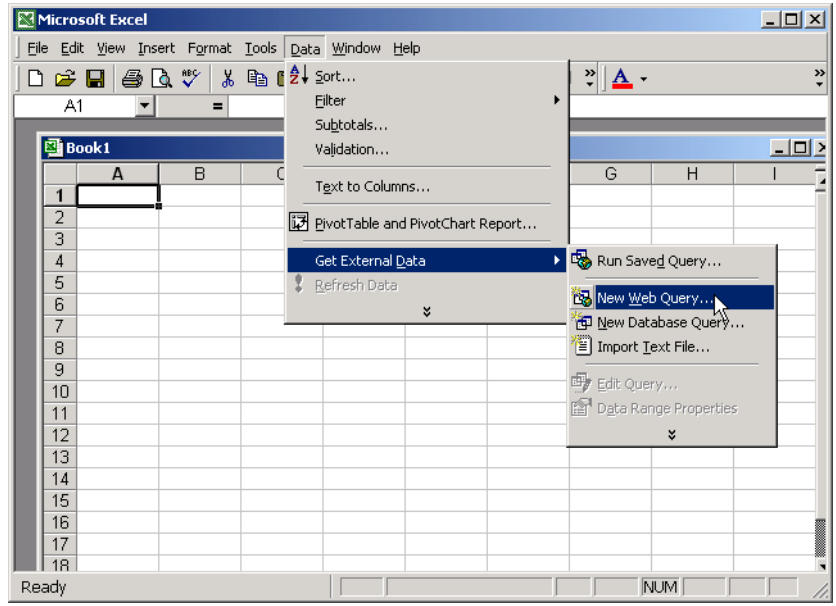
Note The instructions in this section apply only to Pervasive.SQL 2000i, not to previous versions.

Accessing Data Using Microsoft Excel

➤ To access Pervasive data using Excel

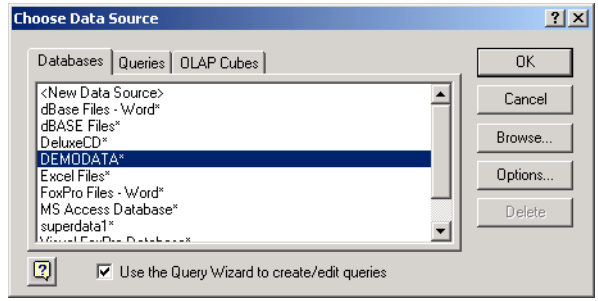
1. You must have the Pervasive.SQL client or any version of the Pervasive.SQL engine installed on the computer where you are using Excel.
2. Start Excel.
3. From the **Data** menu, choose:
   Get External Data | **New Database Query** as shown below.

Figure 2-30 Accessing Pervasive Data using Microsoft Excel



4 The **Choose Data Source** box lists the defined data sources for any ODBC drivers that are installed on your computer. From this list, click on the Client or Server DSN for the Pervasive database you wish to access, as shown in the example below.

Figure 2-31 Excel Display of ODBC Source List



If the database you want does not appear in the ODBC Source list, see“Before You Begin” on page 2-46.

5. Click **OK**. You may be prompted to login to the Pervasive.SQL database. If the database is not secure, leave the **User** and **Password** fields empty. Otherwise enter your assigned user name and password.
6. The **Query Wizard** opens. Simply follow the wizard to select your options such as which tables to query, how to filter and sort the data, and how you would like Excel to return the Pervasive data to you for your use.

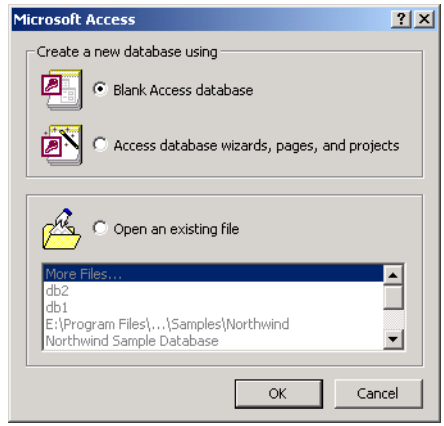
Accessing Data Using Microsoft Access

➤ To access data from Microsoft Access

1 Open Microsoft Access.

2 From the Access dialog box, choose **Blank Access database** as shown below. Click **OK**.

Figure 2-32 Create a New Database using Microsoft Access



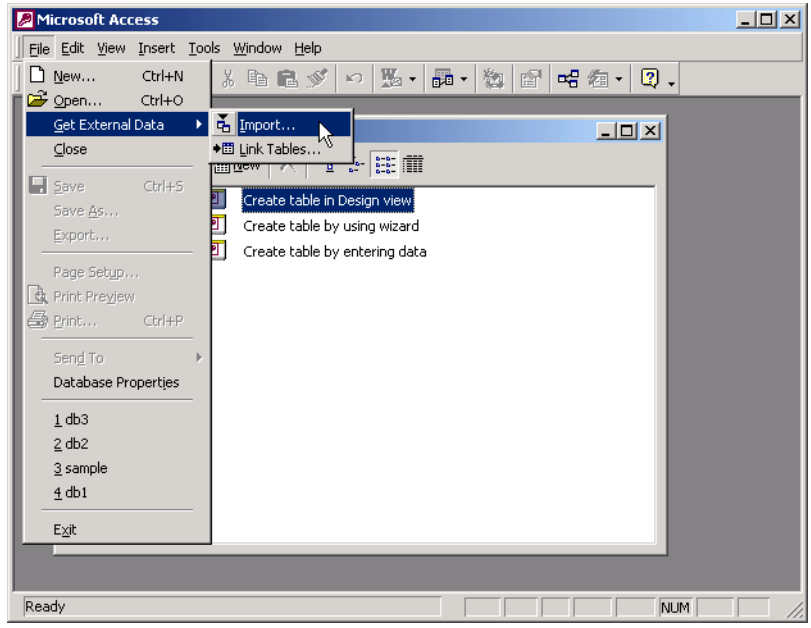
3 Next, the **File New Database** dialog box opens and asks you to name the new database. Name the database and click **Create**.

4 From the Access menu, choose:
File | Get External Data | Link Tables.

Note You have the option to **Import** data or **Link Tables** to the new database. When you choose **Import**, you break the link to the ODBC data source immediately following the import procedure. Essentially, **Import** creates a static copy of the data. When you choose **Link Tables**, Microsoft Access keeps the connection open and remains dependent upon the ODBC data source each time the data is accessed. This way, the data you see reflects any changes to the data at its source.

Note If you wish to link to a file on a local area network, make sure to use a universal naming convention (UNC) path, instead of relying on the drive letter of a mapped network drive in Windows Explorer. A drive letter can vary on a computer or may not always be defined, whereas a UNC path is a reliable and consistent way for Microsoft Access to locate the data source that contains the linked table.

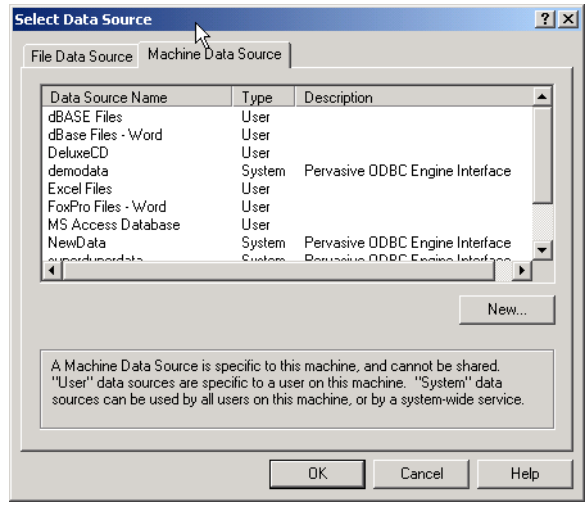
Figure 2-33 Importing External Data Using Access



5 In the **Link** dialog box, in the **Files Of Type** box, select **ODBC Databases**.

6 The **Select Data Source** box lists the defined data sources for any ODBC drivers that are installed on your computer. Click on the **Machine Data Source** tab as shown in the figure below.

Figure 2-34 Access Display of ODBC Source List



7 Select the ODBC data source that you want to link. If the ODBC data source that you selected requires you to log on, enter your user name and password (additional information might also be required), and then click **OK**.

Note To define a new data source for any installed ODBC driver, click **New**, and then follow the instructions in the **Create New Data Source** dialog box and the dialog boxes that follow it before proceeding.

Tip If you are linking a table, select the **Save The Login ID And Password** check box to store the information for the table in the current database, so that users will not have to enter it each time. If you leave the check box cleared, all users must enter the logon ID and password every time they open the table with Microsoft Access in each new session. Your network administrator can also choose to disable this check box, requiring all users to enter a user name and password each time they connect to the database.

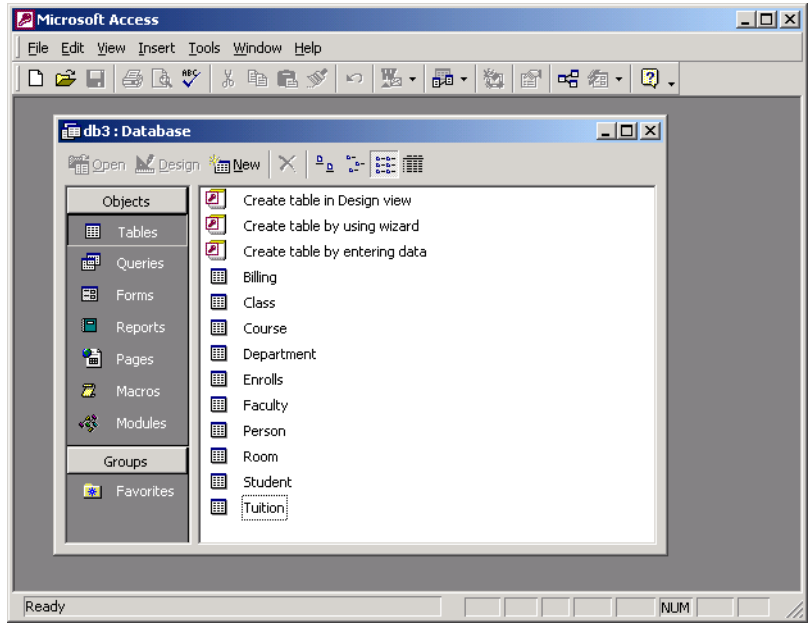
If the database you want does not appear in the ODBC Source list, see"Before You Begin" on page 2-46.

8 The Access **Link Tables** dialog box opens. Click each table that you want to import or link, and then click **OK**.

Note Microsoft Access cannot display more than 256 columns in a table. If you need to display more than 256 columns, you may wish to use a different tool.

9 Linking to your Pervasive data is complete. As shown in the figure below, Access presents you with options for designing the new database. View the linked tables by double-clicking on the table name.

Figure 2-35 Using Pervasive Data in Microsoft Access



Note If you are linking a table and it does not have an index that uniquely identifies each record, then Microsoft Access displays a list of the fields in the linked table. Click a field or a combination of fields that will uniquely identify each record, and then click **OK**.

Deleting DSNs

This section describes how to delete a DSN. One reason you may have to perform this task is if you are upgrading from Pervasive.SQL 7. If this is the case, you must delete your existing DSNs and re-create them using Pervasive.SQL 2000i.

This procedure does not delete Data Dictionary Files (DDFs) or data files.

➤ To Delete a DSN using ODBC Administrator (for Windows DSNs only)

1. From the **Start** menu, choose **Programs | Pervasive | Pervasive.SQL 2000i | Utilities | ODBC Administrator**.
2. In the **ODBC Administrator** window, click on the **System DSN** tab.
3. Click on the DSN you wish to remove, and click **Remove**. You are prompted to confirm removal of the DSN. Click **Yes**.
4. After the DSN has been removed, click **OK** to exit ODBC Administrator.
5. If you are simply deleting an unwanted DSN, you are finished. If you need to re-create the DSN, you should refer to one or more of the following sections:

If you need to do this	... refer to this section:
Re-create an Engine DSN on a server engine or a Workgroup/Workstation engine	One of: ◆ "Setting Up Database Access on a Windows Server or Workgroup/ Workstation" on page 2-19 ◆ "Setting Up Database Access on a NetWare Server" on page 2-28 ◆ "Setting Up Database Access on Unix" on page 2-36
Re-create a Client DSN on a client workstation	"Setting Up Client Access" on page 2-39

➤ To Delete a DSN Using PCC

1. From the **Start** menu, choose **Programs | Pervasive | Pervasive Control Center.**
2. Double-click **Pervasive.SQL 2000i Engines.**

   If you wish to remove an Engine DSN on the server, you must have administrative rights on the server to do so. If you do not see the name of the server computer listed in the left-hand window of PCC, you must register the server. See "Registering or Removing a Server" on page 3-4.
3. Double-click the icon representing the computer where you want to remove the DSN. Double-click **Databases.**
4. Right-click on the database you wish to remove. Choose **Delete** from the pop-up menu.
5. In the Drop Database Wizard that appears, clear **Delete database name.**

Caution Be certain **Delete database name** is not checked. If this box is checked, the internal database name as well as the DSN will be deleted, and you will have to re-create the internal database name.

Also, if you want to save the system files and DDFs, make sure the **Delete database name** is not checked. One reason that you may need to preserve these files is because a Btrieve application is accessing them.

6. Click **Next** to display the confirmation screen and then click **Finish**. You will notified of a successful deletion. Click **Close** to exit the **Delete Database Wizard.**

7 If you are simply deleting an unwanted DSN, you are finished. If you need to re-create the DSN, you need to select one or more of the following options:

If you need to do this	... refer to this section:
Re-create an Engine DSN on a server engine or on Workstation/ Workgroup	One of: ◆ "Setting Up Database Access on a Windows Server or Workgroup/ Workstation" on page 2-19 ◆ "Setting Up Database Access on a NetWare Server" on page 2-28 ◆ "Setting Up Database Access on Unix" on page 2-36
Re-create a Client DSN on a client workstation	"Setting Up Client Access" on page 2-39

Bound Databases and Enforced Integrity

When creating a database, you are offered the options of **Integrity Enforced** and **Bound** (**DDFs created**).

- **Integrity Enforced** means it enforces any triggers and referential integrity defined within the database. Generally, the only time you would *not* want this option checked is when doing a large download of data from an existing database.
- A **Bound** database associates a database name with a single set of Data Dictionary Files (DDFs), which refer to only one set of data files. If DDFs are bound, you can't use those DDFs for several databases, nor can you refer to the data files by more than one set of DDFs. This protects the database from tampering.

Note If you choose to secure a Bound database, you will not be allowed to access the data through the Btrieve API.

chapter

Using the Pervasive Control Center

3

A Brief Tour of Pervasive Control Center

Pervasive Control Center (PCC) is an easy-to-use, graphical tool designed to help you create and manipulate databases and control your DBMS. It allows you to access nearly all the functions of the product from one place. This chapter leads you on a tour of PCC to help you learn the interface, the variety of Pervasive.SQL 2000i tools, and common operations launched from PCC.

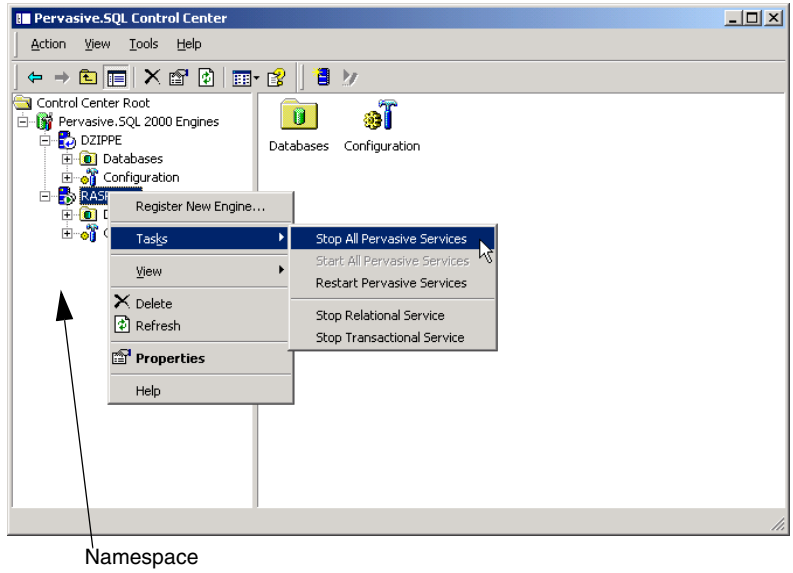
The topics in this chapter include:

An Overview of Pervasive Control Center

Pervasive Control Center (PCC) is an integrated framework in which users can connect to Pervasive.SQL engines, set up and modify databases, query and update data, and tune engine performance.

PCC uses a Windows Explorer-like motif—a tree of objects—referred to elsewhere as the "Namespace." This tree of objects can be opened or expanded to reveal more detail. Examples of objects include engines, databases, tables, users and engine configuration settings. Figure 3-1 shows a picture of PCC.

Figure 3-1 Pervasive Control Center



Each object in the tree, when selected, has its own set of functions and its own set of tools to enable these functions. For example, if one of the **Databases** located under a Pervasive.SQL engine is selected, a different set of tools appears in the toolbar than if **Configuration** is selected. The **Configuration** tools function separately from the Pervasive.SQL engine tools.

Much of the functionality of PCC is implemented as wizards that act upon objects in the namespace. To perform a function on a given object, right-click on the object and choose from the list of actions in the **Tasks** menu.

Some utilities have not yet been tightly integrated within the PCC framework. However, they may still be started from within PCC by selecting them through the **Tools** menu, which is available when anything but the Control Center Root is selected. The non-integrated tools that may be selected this way include:

- Function Executor
- Pervasive System Analyzer
- Gateway Locator (Workstation and Workgroup only)
- Maintenance
- Monitor
- Rebuild
- User Count Administrator
- Custom (third-party) tools

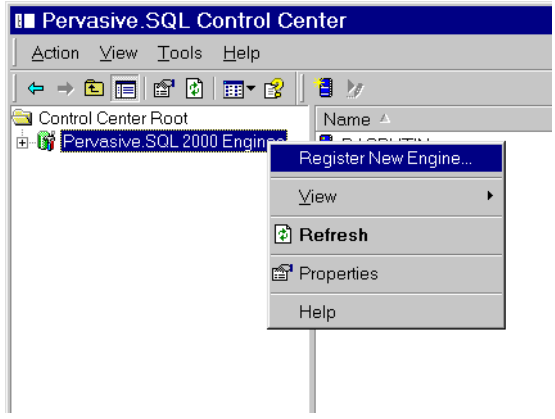
Registering or Removing a Server

You can use Pervasive Control Center (PCC) to work with database engines that are on your machine or on remote server engines. To work with a remote server engine, you must introduce it to PCC. This procedure is called *registering* the server.

➤ To register a remote server with PCC

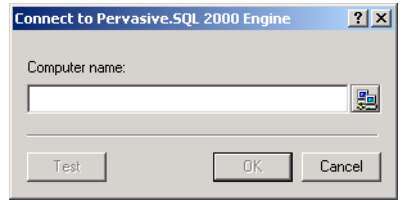
1. Open PCC:
   Choose **Start | Programs | Pervasive | Pervasive Control Center.**
2. Within PCC, double-click on **Pervasive.SQL 2000i Engines** to see a list of engines that are already registered.
3. Right-click on **Pervasive.SQL 2000i Engines** and select **Register New Engine**. Type in or choose the server you want to connect to.

Figure 3-2 Registering a New Engine



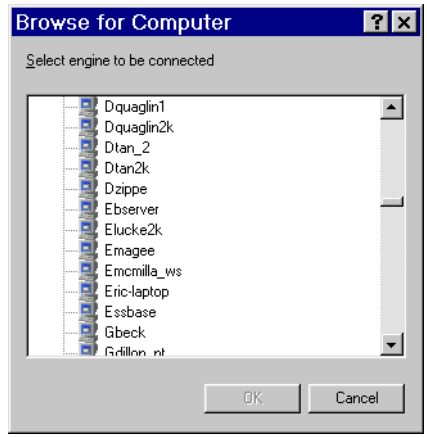
A dialog is displayed that allows you to choose the machine name where the Pervasive.SQL server resides.

Figure 3-3 Choosing a Computer Name



4 If you are unsure of the server name, click the button located to the right of the name field and browse from the Network list that appears, as shown below.

Figure 3-4 Choosing a Computer Name from a Network List



5 Select the computer name from the list or enter the computer name in the field and click **OK**.

The server should now appear in the Namespace window of PCC.

Note If you have engines registered that are not running, you may encounter delays in Configuration as PCC periodically attempts to contact these engines. To eliminate the delay, unregister the "dead" engines. You can minimize the delay by performing the following procedure:

In PCC, right-click **Pervasive.SQL 2000i Engines** and choose **Properties**. Set **Poll interval (seconds)** to 999. Click **OK**.

➤ To remove a remote server from the Namespace

This procedure does not erase the data or database from the server. It only keeps the server from appearing in PCC on your computer.

1. Open PCC:
   Choose **Start | Programs | Pervasive | Pervasive Control Center.**
2. Within PCC, double-click on **Pervasive.SQL 2000i Engines**.
3. Right-click on the server you want to remove and select **Delete**. PCC disconnects from the server, and it is removed from the Namespace.

Viewing Database Engines

Pervasive Control Center (PCC) provides a unique view of available database engines, databases, and actual data. This section explains how to view these objects in PCC.

➤ To view registered database engines

1. Open PCC:
   Choose **Start | Programs | Pervasive | Pervasive Control Center.**
2. You can view the database engines that you have registered in PCC by double-clicking **Pervasive.SQL 2000i Engines** in the left-hand pane (the Namespace) of PCC.

   If you want to add more database engines to the list, follow the instructions provided in "Registering or Removing a Server" on page 3-4.

   By right-clicking on a particular database engine, you can choose to view its properties, refresh its database list, or delete it.

Note If you have engines registered that are not running, you may encounter delays in PCC as it periodically attempts to contact these engines. To eliminate the delay, unregister the "dead" engines. You can minimize the delay by performing the following procedure:

In PCC, right-click **Pervasive.SQL 2000i Engines** and choose **Properties**. Set **Poll interval (seconds)** to 999. Click **OK**.

Interpreting Server Status Icons

Computer names are listed in the namespace tree under the Pervasive.SQL 2000 Engines node. An icon indicating the state of the Pervasive.SQL engine on that computer accompanies each engine

name. The possible states of the database engine are outlined in the following table.

Table 3-1 Pervasive.SQL Machine States

Engine Status	Small Icon	Large Icon
Workstation/Workgroup (local or remote)		
Windows NT/2000 Server (Green triangle) (relational and transactional)		
Windows NT/2000 Server (Green triangle) (transactional only)		
NetWare Server (Blue triangle)		
Unix Server (Brown triangle)		
Engine stopped		
Engine detection in progress		
No Service Found		

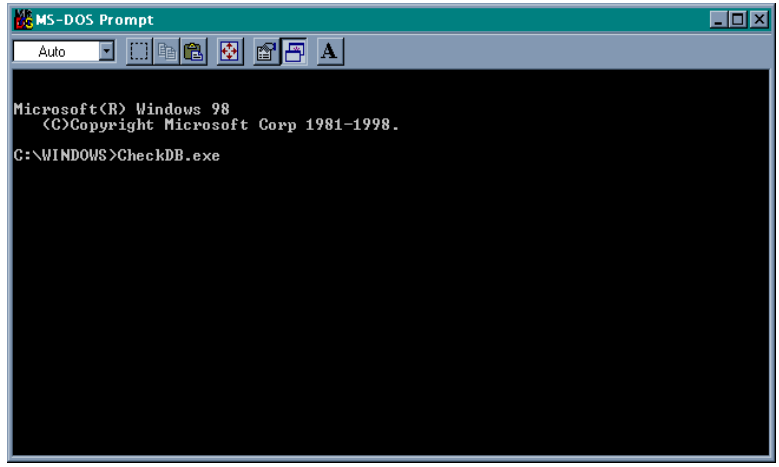
Note When PCC is next started, engines remain in the Namespace even if the engines are not running. Engines that are not running are listed in the namespace with an appropriate icon indicating the engine is not running.

Pervasive Control Center Wizards

This section provides information about wizards available from Pervasive Control Center (PCC). These wizards can be invoked using a **DOS command line** or through a window in PCC. In this manual, PCC is the default starting point for creating and deleting tables, data and databases.

If you invoke a wizard from the command line (see Figure 3-5), a login screen requires you to specify a **Server** and **Database** to use. If the database you want to access is secure, the login screen also requires you to enter a **User Name** and **Password**. The login screen also appears when creating a new database. The following screen is an example of a wizard invocation using a DOS command line:

Figure 3-5 DOS Command Line Screen Example



The following table provides the file names of the wizards if you wish to start them from the command line:

Table 3-2 File Names of Wizards

File Name	Wizard
createdb.exe	Create Database
checkdb.exe	Check Database
crtblwzd.exe	Create Table

Table 3-2 File Names of Wizards continued

File Name	Wizard
dropdb.exe	Drop Database
droptab.exe	Drop Table
expwizrd.exe	Data Export
impwizrd.exe	Data Import
nulcnvwz.exe	Null Conversion
psawizrd.exe	Pervasive System Analyzer
w3sqlqpv.exe	SQL Query Plan Viewer

Adding or Creating a Database

To add or create a database, use the Create Database Wizard. Newly created databases are empty and may be populated with tables using the Add Table Wizard described in "Adding a Table" on page 3-18.

You also use the Create Database Wizard to create Engine and Client DSNs for pre-existing databases. This allows a database to be accessed using ODBC. For information on how to do this, see "Setting Up ODBC Database Access" on page 2-14.

Note If you wish to add a database to a Server engine, you must have administrative rights on the server operating system. If you do not have administrative rights, you will not be permitted to add the database.

➤ To add or create a database

1. Within PCC, right-click on **Databases** under the machine on which you want to create the database.

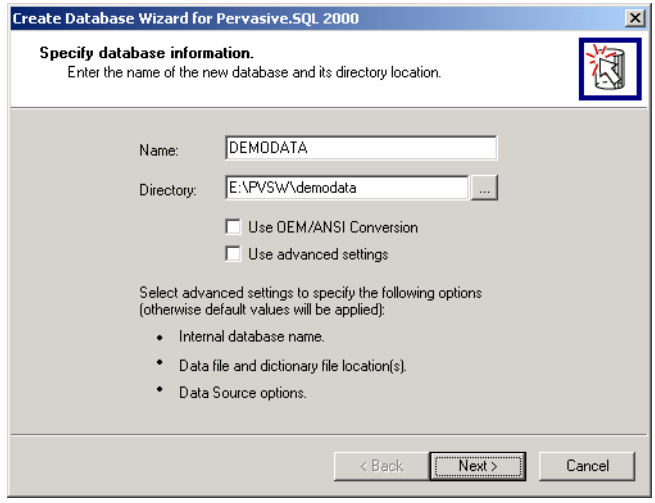
   If you do not see the name of the machine where you want to create the database, see "Registering or Removing a Server" on page 3-4.

2. Select **New Database** from the shortcut menu.

3. On a Server engine, you may be prompted to login. If so, supply a user name and password for the server operating system.

   The Create Database Wizard starts with the following dialog box:

Figure 3-6 Create Database Wizard Dialog Box



Enter the name of the new database and the location where its files will reside.

Note Be careful when entering directories on remote machines. When creating a server database, remember that the directory entered in this dialog box must be valid as if it was entered on the remote machine. This means that paths with mapped drives may not be valid. In this case UNC paths should be used instead.

Note For NetWare systems, use the following path:
`\\server\vol1:\path`

For Unix systems (when Samba is not used), use the following path:
`\\server\$PVSW$\[path on Unix from root '\' dir]`

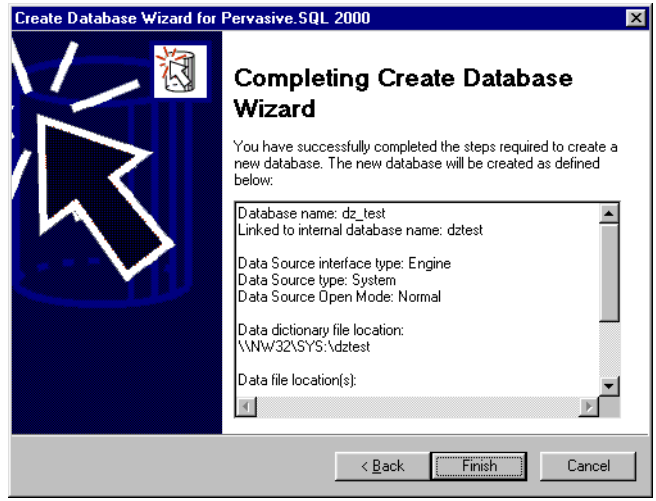
For Windows (or Unix systems with Samba), use the following path:
`\\server\sharename\path`

Optionally, you may check the **Use advanced settings** box to specify more details on how the new database should be created, or if you want to create DDFs for existing data files. If this option is selected, an additional dialog box appears later. This option is not required for most database creation tasks. For instructions on using the advanced settings, see "Advanced settings procedure" on page 2-24.

Click **Next** when done.

4 Now that all of the required information has been gathered, the Create Database Wizard presents a screen indicating the actions it is about to take. This screen allows you to confirm the settings before they are committed to the engine. Click **Finish**.

Figure 3-7 Create Database Wizard - Complete Dialog Box



5 If no errors have occurred, the wizard indicates success and displays a final dialog box (not shown). Click **Close**.

Now, if you look in the PCC namespace, a new entry should appear for the database just created. You may have to refresh the list in the Pervasive Control Center for the new database to show. To do this, point the cursor at the menu, click **Action**, then **Refresh.**

Note If you have created a brand new database, where no DDFs or data files existed previously, your new database is empty. No tables or columns have been created. You can use the Create Table Wizard, described in "Adding a Table" on page 3-18, to define tables and columns.

Deleting a Database

PCC includes a Delete Database Wizard. You can use the wizard to delete a DSN, a Database Name, or an entire database.

To delete an Engine DSN, a Database Name, or a complete database from a server, you must have administrative rights on the server.

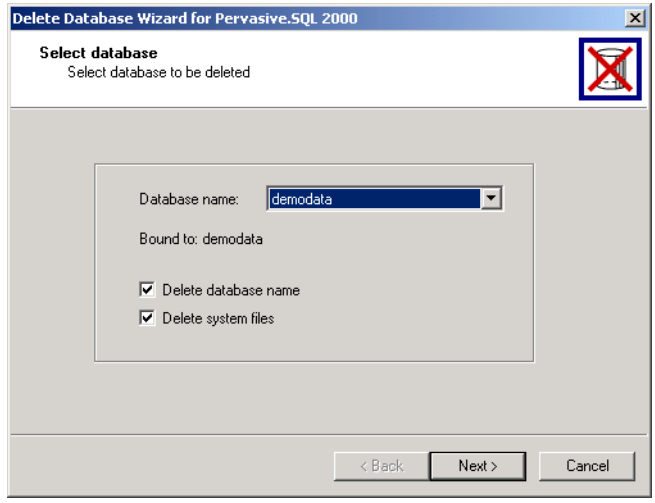
Caution This procedure is different than removing an engine from the PCC Namespace. This procedure may permanently delete data files and data dictionaries.

Caution Do not use the Delete Database Wizard (or the PCC Delete command) to delete replication-enabled databases or replication templates. Use the Deactivation Wizard to remove replication-enabled databases and the Template Remover Wizard to remove templates. The Deactivation Wizard and the Template Remover Wizard can be accessed from the PCC after the Pervasive.SQL Replication product is installed.

➤ To remove an existing database from an engine

1. In PCC, right-click on the database to be deleted.
2. Select **Delete** from the shortcut menu. The following dialog box appears:

Figure 3-8 Delete Database Wizard Dialog Box



This dialog provides options to customize how much of the database to remove from the system. By default the database DSN, the internal database name and all data files are removed.

If no options are selected, only the DSN is deleted. The internal database name, the DDFs, and the data files are not affected.

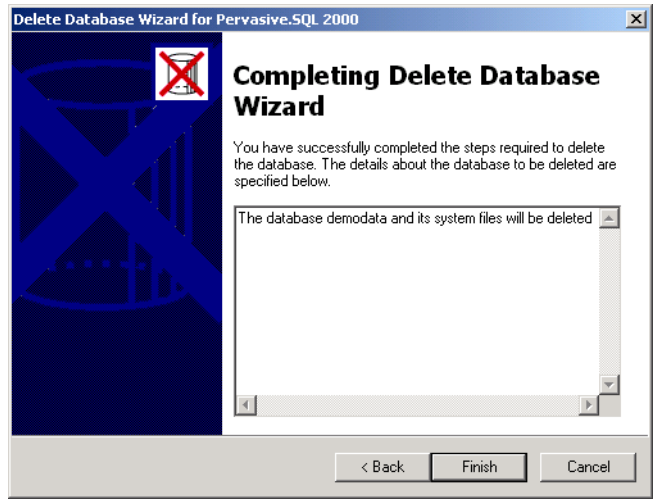
If **Delete database name** is the only option checked, the DSN and the internal database name are deleted, but the DDFs files and the data files are not affected.

If **Delete system files** is checked, the DDFs and data files are deleted. You cannot select this option without also selecting **Delete database name**.

Caution Depending on the options you have selected, deleting a database may erase the related data files and data dictionaries from your hard drive. Do not delete a database unless you are certain you want to do so.

3 Click **Next**. To delete the database, click **Finish**. To quit without deleting the database, click **Cancel**.

Figure 3-9 Completing Delete Database Wizard Dialog Box



4 After you click **Finish,** the wizard displays a final dialog box (not shown) that indicates success. Click **Close**. At this point the database should disappear from the PCC Namespace. You may need to refresh the screen in the PCC to view this change. To do this, go to the toolbar, click **Action**, then **Refresh.**

Adding a Table

Tables are the objects in which databases store data. PCC's Create Table Wizard enables you to easily add and remove tables from any existing database.

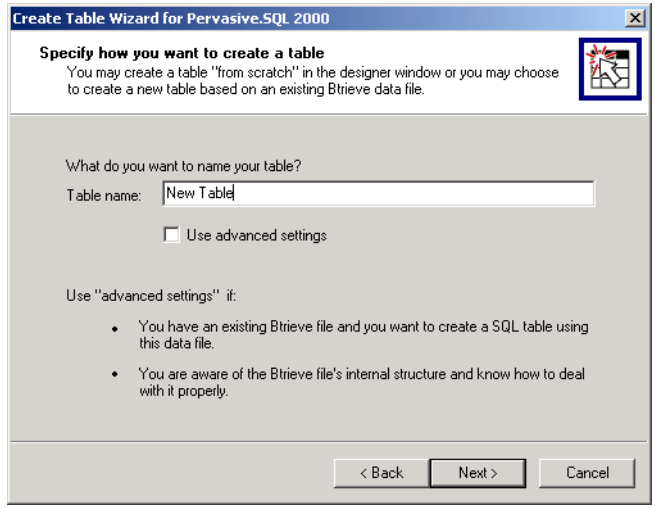
Note To create a table in a database, database security must be turned off or you must have access rights to create tables.

Caution It is highly recommended that you backup all your DDFs and data files before you perform functions through Table Designer such as create table, alter table, convert nulls, or import of an external file. This powerful tool is designed to give you the ability to modify your database schema and data. If you inadvertently set the options incorrectly or enter incorrect data, you could change your files in an irreversible manner. Full recovery will be possible if you have performed a backup.

➤ To create a new table within a database

1. Click on the **Tables** Namespace node beneath the database in which the new table is to be created.
2. Right-click on the background in the table list in the right pane of the Control Center window or right-click on the "Tables" icon underneath a given database icon.
3. Select **New Table** from the shortcut menu. The following dialog box appears:

Figure 3-10 Create Table Wizard Dialog Box

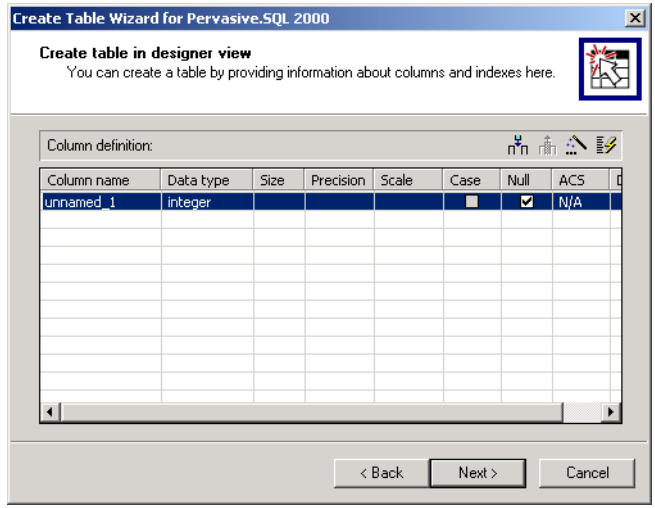


Enter the name of the new table. Click **Next** when done.

For advanced users only: you may select the **Use advanced settings** box in order to create a table definition for an existing Btrieve file. The steps to use this advanced option are described in *Advanced Operations Guide*, Chapter 12, "Adding Relational Access to Btrieve Files."

4 The columns that are to appear in the new table must now be defined. The following dialog box presents a grid display in which columns may be defined:

Figure 3-11 Create Table Wizard - Designer View Dialog Box



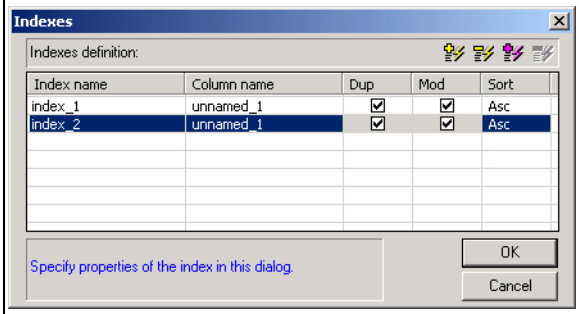
Create Table Wizard Design Buttons

Use the design buttons (located at the right of the window) for ease in creating a table. Their functions are described in Table 3-3.

Table 3-3 Table Wizard Tools

Function	Icon
Add Column - Click this button to add a column.	
Remove Column - Highlight the column to be removed and click this button.	

Table 3-3 Table Wizard Tools

Function	Icon
Index Window - Click this button for a dialog box in which you can specify properties of the index.	 
Index Window Tools:	
Add Index	
Remove Index	
Add Segment	
Remove Segment	

Create Table Wizard Column Attributes

Name
Enter an alpha-numeric name for the column. The limit is 20 characters. You can have spaces in the name, but if you have spaces, you must always enclose the name in quotes whenever you are referring to it in SQL statements. Avoiding the use of spaces is recommended.

Data Type
Choose a data type for the column.

Size
For character or binary fields, specify how many bytes are permitted in this field.

Precision
For floating point values, specify the number of significant digits.

Scale
For floating point values, specify the number of significant digits that are to the right of the decimal point.

Case
For character data types, checking the **Case** attribute forces the database engine to use case-insensitive comparisons when searching for values in the database.

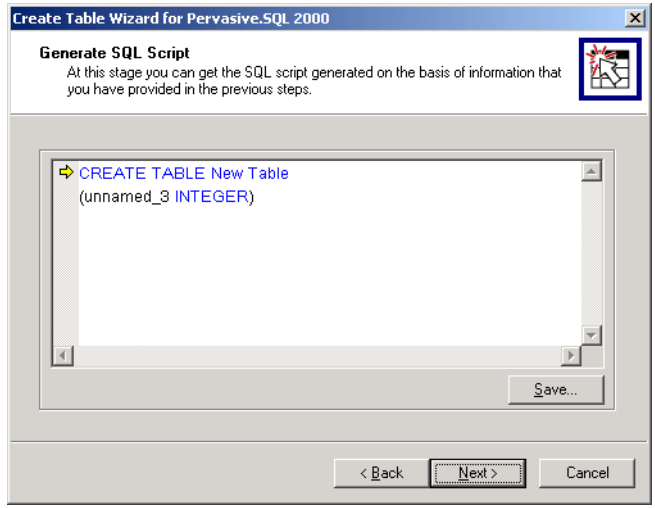
Null
For data types that allow NULL values (IDENTITY and BIT data types cannot be null), checking the **Null** attribute permits NULL values in the column.

ACS
If you wish to use an alternate collating sequence (ACS) for this column, specify the path name to the ACS here.

Default
If you wish to specify a default value for this column, enter the value here. The default value is used if a row is inserted and no value is provided for this column.

5 The next dialog box (Figure 3-12) displays the SQL statement that could be used to recreate this table at a later date. This statement can prove useful in replicating a database structure at another location, or as a safety measure in case the table must be re-created.

Figure 3-12 Create Table Wizard - Generate SQL Script Dialog Box



6 You can click the **Save** button to save the script so that it can be run in the Pervasive.SQL data manager at any time. Click **Next**.

7 By clicking **Finish** in the dialog box below (Figure 3-13), the changes that have been entered thus far will be committed to the database.

Figure 3-13 Create Table Wizard - Complete Dialog Box

8 The wizard displays a final dialog box (not shown) indicating that the operation was a success. Click **Close**.

Creating Tables for Existing Data Files

Sometimes a data file already exists, but it has no table definition allowing it to be accessed with ODBC. The advanced settings of the Create Table Wizard allow for adding table definitions for existing data files. For detailed instructions on how to perform this complex procedure, see *Advanced Operations Guide*.

Modifying a Table Definition

You can add, delete, or change the characteristics of columns within a table by using Table Designer. You can apply these changes only to the table definition, or to the actual data in the data file as well.

Note To modify a table definition in a database, database security must be turned off or you must have access rights permitting you to modify table definitions.

Caution It is highly recommended that you backup all your DDFs and data files before you perform functions through Table Designer such as create table, alter table, convert nulls, or import of an external file. This powerful tool is designed to give you the ability to modify your database schema and data. If you inadvertently set the options incorrectly or enter incorrect data you could change your files in an irreversible manner. Full recovery will be possible if you have performed a backup.

Linked or Unlinked Mode

When Table Designer is in Linked mode, the changes you make are reflected in both the table definitions and the corresponding data files. When Table Designer is in Unlinked mode, the changes you make are reflected only in the table definitions; the data files are not modified in any way.

Unlinked mode is provided so that advanced users can modify table definitions to match existing data files. Under normal circumstances, it is recommended that you always use Table Designer in linked mode.

In the main Table Designer window, the lower right corner indicates your current mode: UNLINKED MODE or LINKED MODE.

Caution By using Table Designer in Unlinked mode, it is possible to make your data files inaccessible. Only advanced users should use Unlinked mode.

Currently, Table Designer can perform the following operations to modify existing tables:

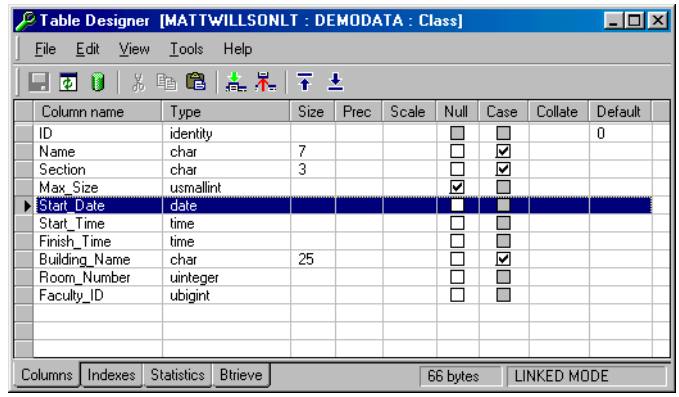
Table 3-4 Table Designer Alter Table Functions

Function	Linked Mode	Unlinked Mode
Change column name	✔	✔
Add column	✔	✔
Drop column	✔	✔
Change Case attribute	✔	✔
Change ACS attribute	✔	✔
Change column size	No	✔
Change column data type	No	✔
Modify index	No	✔
Modify table with Btrieve owner name	No	✔
Modify table with security enabled	No	No
Modify table with referential integrity constraint	No	No
Add index	✔	✔

➤ To access the Table Designer

1. Click on the **Tables** namespace node under the database whose table you wish to modify.
2. Right-click on the table to be modified. From the pop-up menu, choose **Tasks | Edit Table Design**. Table Designer appears, as shown in Figure 3-14.

Figure 3-14 Table Designer



➤ To specify whether changes affect only the table definition or also the data file

1 From Table Designer menu, choose **Tools | Options....**

2 In the Options window, make sure **Table link mode** is checked if you want changes to apply to both the table definition and the data file.

If you want changes to apply to only the table definition, then make sure **Table link mode** is not checked.

Caution Do not use Unlinked mode unless you are an advanced user. By making changes to your table definition that are not reflected in your data file, you may make it impossible to access the data file.

3 In the main Table Designer window, the lower right corner indicates your current mode: UNLINKED MODE or LINKED MODE.

➤ To add, delete, or modify columns or indexes

1 See "Create Table Wizard Design Buttons" on page 3-20 for information about the controls.

2 See "Create Table Wizard Column Attributes" on page 3-21 for information about the column attributes.

3 Make changes to the table definition as you see fit. When you are finished, choose **File | Save** to save your changes. Choose **File | Reset** to erase your changes and revert to the last saved version of the table definition.

Restrictions

The ability to modify the Null attribute or data type of a column is subject to the following restrictions:

- The target column cannot have a PRIMARY/FOREIGN KEY constraint defined on it.
- If converting the old type to the new type causes an overflow (arithmetic or size), the ALTER TABLE operation is aborted.
- If a nullable column contains NULL values, the column cannot be changed to a non-nullable column.

If you must change the data type of a key column, you can do so by dropping the key, changing the data type, and re-adding the key. Keep in mind that you must ensure that all associated key columns in the database remain synchronized.

For example, if you have a primary key in table T1 that is referenced by foreign keys in tables T2 and T3, you must first drop the foreign keys. Then you can drop the primary key. Then you need to change all three columns to the same data type. Finally, you must re-add the primary key and then the foreign keys.

For additional information, refer to ALTER TABLE in *SQL Engine Reference*.

Dropping a Table

Dropping a table using the PCC's Drop Table Wizard is very similar to dropping a database.

Note To delete a table from a database, database security must be turned off or you must have access rights to delete tables. See Chapter 2—"Using the Pervasive Control Center," for information on access rights.

Caution This procedure may permanently delete data files and DDFs.

➤ To delete an existing table from a database

1. Click on the **Tables** Namespace node under the database whose table you want to drop.
2. Right-click on the table to be dropped in the table list pane on the right side of the Control Center window.
3. Select **Delete** from the shortcut menu.
4. The following dialog box appears, allowing you to delete the table.

Figure 3-15 Drop Table Wizard

The option **Delete Data file** not only removes the table from the database but also removes the actual data file itself from the machine's hard drive. Click **Finish**.

5 The final **Drop Table Wizard** dialog box (not shown) indicates to you that the table was successfully dropped. Click **Close**.

Note When a table is in use, it cannot be dropped. If you are unable to drop a table, use Monitor to verify whether the table is in use.

Setting Database Security

For further information on database security, see *Advanced Operations Guide*, Chapter 7, "Owner Names and Relational Security."

Database security helps prevent intruders from accessing the data on your databases or unintentional damage to data by authorized users. By default, security is turned off. Security is enabled on Pervasive.SQL databases through the PCC or by executing.

Note When turning security on or off, all database connections must be closed. Because SQL Data Manager uses additional connections, you must close all SQL Data Manager windows before attempting to turn security on or off. If you are viewing the contents or properties of any table in the database, then you must close that window before proceeding.

Turning Security On and Off

➤ To set database security from within the PCC

1. Right-click on a database icon in PCC Namespace. Select the **Properties** menu item.
2. Select the **Security** tab in the Database properties dialog box.

Figure 3-16 Database Properties Dialog Box

3 Enter a password for the Master user.

Caution Be sure to specify a password with significant length, at least five characters. Do not leave the password field blank because doing so creates a major security risk for your database.

Click **OK** when done.

The database is now secured, and a single user named "Master" has been created with the password you specified. Until additional users are created and assigned permissions in the database, only the Master user is permitted to view or update data.

A secure database supports individual users and groups of users who have the same set of permissions. Additional users and groups may be added to the database through the **Users** namespace node under the database. For more information about adding users and groups, see "Working with Groups and Users" on page 3-32.

➤ To turn off database security

Caution Turning off database security deletes all users, groups, and permissions. If you turn security back on, you must re-create all users and groups.

1 Right-click on the icon for your database in the PCC Namespace and choose **Properties** in the shortcut menu.

2 Select the **Security** tab in the Database properties dialog box.

3 Since security is on, the pane includes a check box entry to turn security off. Select it and press **OK** to turn off security. Unless you have an open database session, Pervasive.SQL prompts you for the Master password before turning off security.

Working with Groups and Users

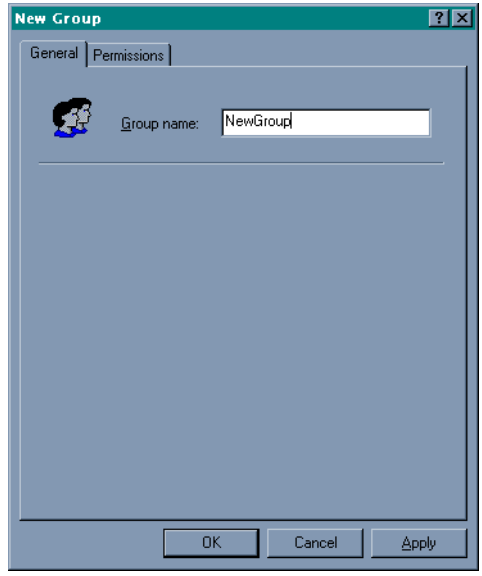
For any data files in your database with Btrieve owner names set, you cannot use PCC to grant access to these files. You must use the GRANT statement with the appropriate owner name.

➤ To add a new group to the database

1 Select the Namespace node **Users** under the database in which you would like to add a new group.

2 Right-click **Users**. Select **New Group** from the shortcut menu. The following dialog box appears:

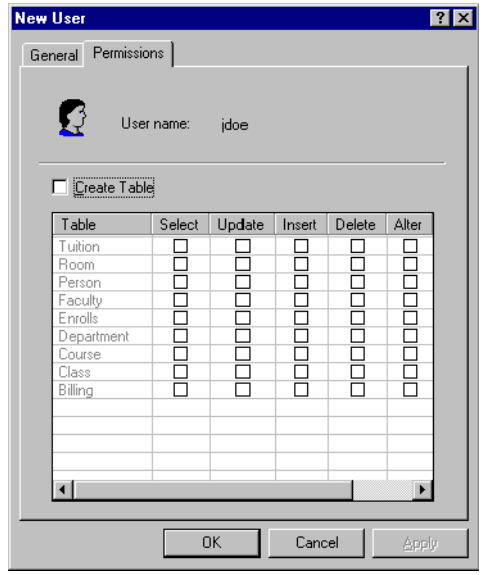
Figure 3-17 Add New Group Dialog Box



3 Type in a group name.

4 Click on the **Permissions** tab to select the specific operations that this Group is entitled to perform on the database:

Figure 3-18 Add New User / Group Permissions Dialog Box



Each check box represents a specific permission for a specific table. For example, if you want members of this group to be able to perform SELECT operations on the "Billing" table, click the box on the row labeled "Billing" under the column "Select."

5 Click **OK** when done.

You can also perform the task of creating a group and assigning permissions using the SQL statements CREATE GROUP and GRANT.

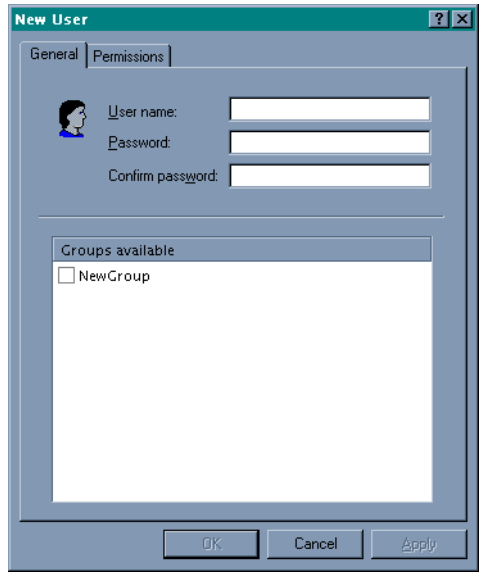
➤ To add a new user to the database

Note If you wish to use groups, you must set up the groups before creating users. You cannot add a user to a group after you have already created the user.

1 Select the Namespace node **Users** under the database in which you would like to add a new user.

2 Right-click **Users**. Select **New User** from the shortcut menu. The following dialog box appears:

Figure 3-19 Add New User Dialog Box

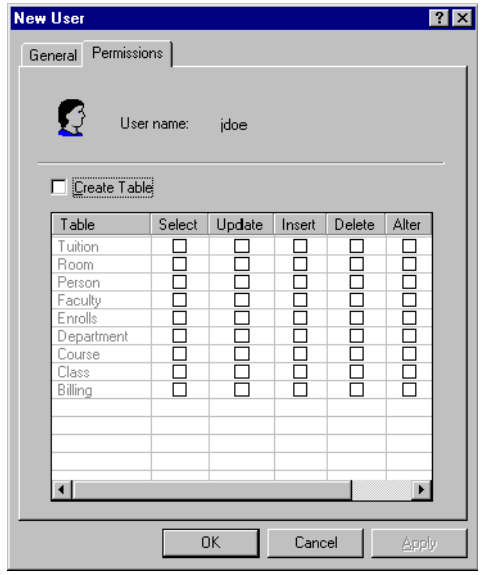


Enter the new user information in the name and password fields and select the group, if any, to which the new user belongs.

Note Group members do not have individual permissions. Every member of a group has exactly the permissions that have been defined for the group.

3 Click on the **Permissions** tab to select the specific operations that this user is entitled to perform on the database:

Figure 3-20 Add New User / Group Permissions Dialog Box



Each check box represents a specific permission for a specific table. For example, if you want members of this group to be able to perform SELECT operations on the "Billing" table, click the box on the row labeled "Billing" under the column "Select."

4 Click **OK** when done.

You can also perform the task of creating a user and assigning permissions using the SQL statement GRANT.

➤ To delete a user or group

Note To delete a group, you must first delete all users in the group. You cannot delete a group that contains users.

1 Double-click the **Users** node in PCC. Right-click on the icon representing the user or group you want to delete.

2 From the pop-up menu, click **Delete**.

You can also perform this task using the SQL statement REVOKE LOGIN FROM *user*.

➤ To add an existing user to a group

Existing users cannot be directly added to a group. Follow these steps:

1. Delete the user by following the steps provided in "To delete a user or group" on page 3-36.
2. Re-create the user by following the steps provided in "To add a new user to the database" on page 3-34.

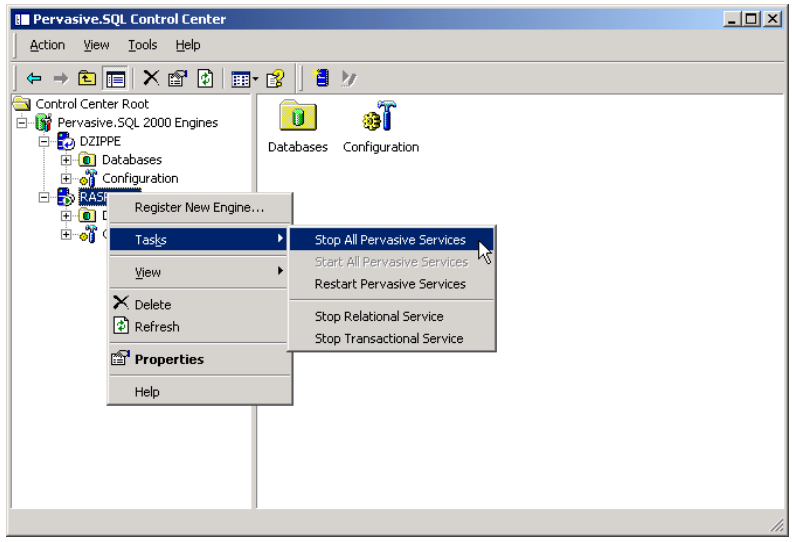
Stopping and Restarting Services on Windows Servers

PCC offers a convenient way to start and stop Pervasive.SQL servers on Windows NT/2000 machines without having to use the Windows Services control panel.

Services on remote machines may be started and stopped provided that you have database administration rights on the remote machine.

1 To start or stop a relational service, right-click on the machine in the PCC Namespace and select **Tasks** from the shortcut menu. The submenu options allow you to start or stop the transactional engine, the relational engine or both.

Figure 3-21 Stopping and Restarting Services in the Pervasive Control Center



Note You must stop the relational *and* transactional services to completely stop Pervasive.SQL. Stopping just one of the services does not stop the database engine completely.

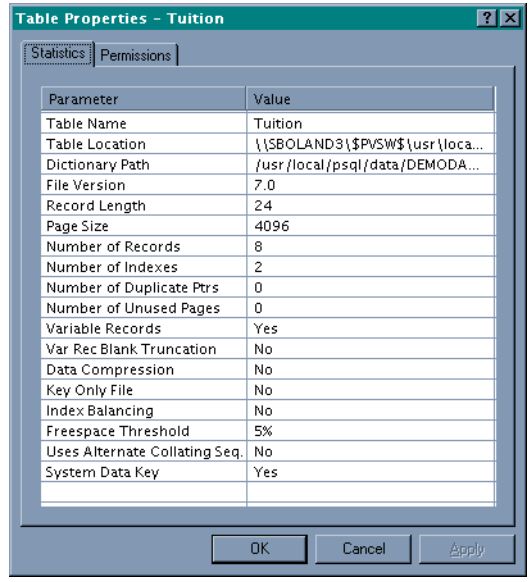
Viewing and Modifying Table Properties

In PCC, it is very easy to view the properties of a table. You can view and modify table properties as needed.

➤ To view the properties of an existing table within a database

1. Click on the **Tables** Namespace node under the database whose table you wish to view.
2. Right-click on the desired table in the table list in the Control Center's right pane.
3. Select **Properties** in the shortcut menu. The following dialog box appears:

Figure 3-22 Table Properties Window



Tip To view or modify column properties, right-click on the table and select **Tasks | Edit Table Design** to start Table Designer.

The Table Properties dialog box presents the following information:

Table 3-5 Existing Table Properties

Statistic Name	Description
Table Name	Shows the name of the table as it appears in the database schema.
Table Location	Shows the physical location of the data file associated with the table.
Dictionary Path	Displays where the database's DDF files are located.
File Version	Shows the earliest Btrieve engine version that can read the file format.
Record Length	Shows the length of the data file's records.
Page Size	Shows the page size (in bytes) of the data file. The page size determines the maximum number of index segments that can be defined in a table.
Number of Records	Shows the number of records currently contained in the data file.
Number of Indexes	Shows the number of indexes defined for the table.
Number of Duplicate Pointers	Shows the number of linked duplicate indexes that can be added.
Number of Unused Pages	Shows the number of pre-allocated pages available. If pre-allocation is enabled, the MicroKernel pre-allocates a specified number of pages when it creates the data file. Pre-allocation guarantees that disk space for the data file is available when the MicroKernel needs it.
Variable Records	Shows whether the data file contains variable-length records.
Variable Record Blank Truncation	Shows whether blank truncation is enabled. If it is, the MicroKernel truncates the blanks in variable-length records. Blank truncation is applicable only if the Variable Records statistic is Yes and Data Compression is set to No.
Data Compression	Shows whether data compression is enabled. If it is, the MicroKernel compresses each record it inserts into the data file.
Index Balancing	Shows whether balanced indexing is enabled.

Table 3-5 Existing Table Properties

Statistic Name	Description
Free Space Threshold	Shows a percentage (5%, 10%, 20% or 30%) if the data file has a free space threshold. The MicroKernel stores the variable-length portions of records on their own pages (called variable pages), separate from the fixed-length portions (which are stored on data pages). The MicroKernel uses the threshold to determine whether to add data to an existing variable page or to create a new one. A higher free space threshold reduces fragmentation of variable-length records across several pages but uses more disk space.
Use Alternate Collating Sequence	Shows whether the table uses an alternate collating sequence for sorting.
System Data Key	Shows whether the data file has system data keys enabled.

Occasionally, a table requires modifications to its list of columns or indexes. Table Designer provides a mechanism through which such changes can be made.

Viewing and Modifying Data

When you double click a table, PCC starts up SQL Data Manager (SQL DM) to allow you to work with the data in the table. You can use SQL DM to run SQL statements to view, update, and delete records, or insert new records.

SQL DM provides a graphical query builder for constructing queries, and gives you the full power of SQL to work with your databases.

This section is not intended to cover SQL or SQL DM in depth, but rather to give you a brief introduction to the many uses of this utility. For detailed information on SQL syntax, see *SQL Engine Reference*.

Viewing Data

To view all the data in a table, simply double-click on the table icon in PCC. When SQL DM comes up, its default behavior is to display a grid containing all records in the selected table.

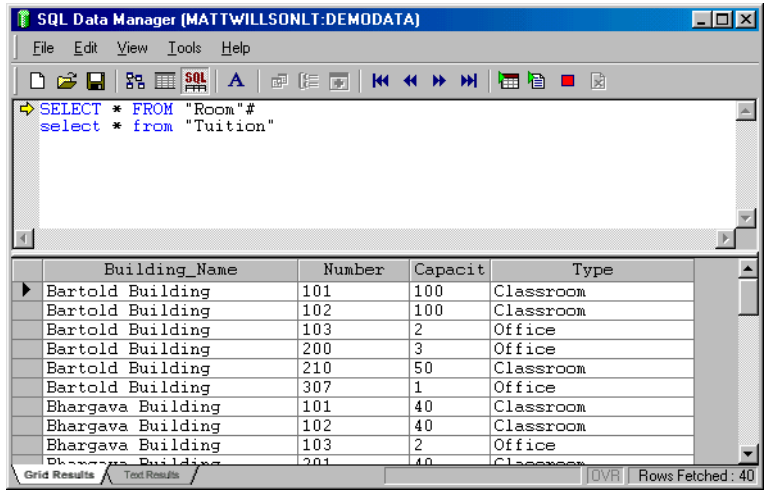
The window bar at the top of the window shows the machine name where the database is located, and the name of the database containing the current table.

If you want to see a limited set of rows or a sub-set of the columns in the table, see the next section on writing SQL statements.

Writing and Executing SQL Statements

By default, SQL DM appears with the query pane immediately beneath the toolbar, and the results grid at the bottom, as shown in Figure 3-23.

Figure 3-23 SQL Data Manager



You can type SQL statements directly in the query pane. If you wish to enter more than one SQL statement, separate each statement using the pound sign (#) as shown above. You can change this delimiter character to a semi-colon by selecting **Tools | Properties** from the menu.

By default, SQL DM runs only the current statement. The yellow arrow in the left-hand margin points to the current statement. You can select the current statement by placing the cursor in the statement you wish to select, or by using the blue arrow buttons to move forward and back through the statements.

You can run all the SQL statements by clicking on the Tools menu and selecting **Query | Run All SQL Statements**. The table below shows the toolbar buttons and the functions they perform:

Table 3-6 SQL Data Manager Buttons

Buttons	Functions
	Respectively, these buttons **start** a new SQL script file, **open** an existing SQL script file, or **save** the current screen of SQL statements to a script file. A script file is a text file containing SQL statements.
	Respectively, these buttons open the **Query Builder Diagram**, the **Query Builder Grid**, and the **Query Text Pane**.

Table 3-6 SQL Data Manager Buttons

Buttons	Functions
	This button allows you to set the **font characteristics** in the Query Text Pane.
	These buttons are available only when the Query Builder Diagram or Grid is open. **Choose statement**: The first button allows you to choose a SELECT, INSERT, UPDATE, or DELETE statement. **Group By**: The second button allows you to specify a GROUP BY clause in a SELECT statement. **Add Table**: The third button allows you to add a table to the query.
	These buttons move the current statement pointer through the list of statements. If the cursor is placed within the body of a given statement, the “Previous” button moves the cursor to the beginning of the given statement.
	This button **executes the current statement** and places the results into an **active grid**. After the grid has been populated, you may directly update your database by making changes to the values in the grid. The grid does not allow cutting and pasting data into another application. The grid caches results locally and only displays the current 40 records. Statements that generate a large amount of output continue to run in the background as you scroll through the early results.
	This button **executes the current statement** and places the results in a **text window**. You cannot change the actual values in the database by changing the values in the text window. You may cut and paste the output into another application. If your statement generates a large amount of output, using the text window may take a significant amount of time to complete the results. Unlike the active grid, results do not display in the text window until the statement is complete. Because I/O to the screen is slow, output of large results may take some time.

Table 3-6 SQL Data Manager Buttons

Buttons	Functions
	This button **interrupts** the current query. If you are executing a statement into the active grid, any results are cleared by clicking this button. If you are executing a statement into the text window, the results remain in the window after clicking this button.
	This button is available only if you are executing a statement into the text window. This button **clears all results** in the text window.

Using the SQL Statement Builder

The graphical Query Builder allows you to create database queries without typing out SQL statements.

Note These procedures assume that you already have PCC open, and you have started SQL DM by double-clicking on a table icon in PCC.

➤ To build a query graphically

1. Start up Query Builder by clicking on the Query Builder Diagram button. 
2. Click the Add Table button.  In the window that appears, select the table(s) you wish to query.
3. To create a join, click on a column in one table and drag it to the corresponding column in the other table. For example, if you want to select rows where “person.id” is equal to “faculty.id”, then click on the “id” column in the “person” table and drag it onto the “id” column in the “faculty” table.
4. Click the appropriate check box for each column you want to return in the result set, or each column you want to insert, or update.
5. When the query is finished, you may display the results in an active grid or in a text window by clicking the appropriate button in the toolbar.

Exporting/Importing Data

Pervasive.SQL provides an ODBC interface, which makes it easy to access your data from a large number of third-party programs. Sometimes data must be made available in other ways or in specific formats. To ease the migration of data to and from Pervasive.SQL databases, Pervasive Control Center (PCC) includes import and export wizards.

The Import/Export wizards in PCC support two different data formats, which can be read and written. The formats are shown in Table 3-7.

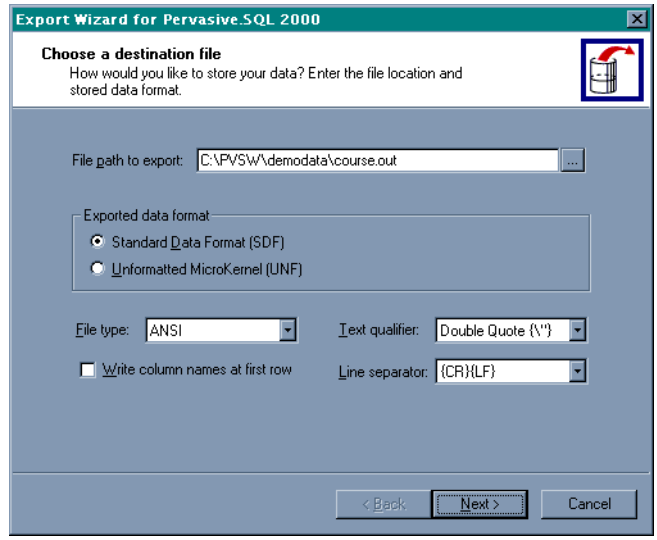
Table 3-7 Import/Export Wizards Data Formats

File Format	Description
Standard Data Format	Data is represented as standard ASCII, OEM or Unicode characters. Commas separate the columns. By default, the quote character is used to enclose columns, and carriage return/line feed is used to separate records. The column separator and record separator characters are configurable.
Unformatted MicroKernel	The wizard does not convert the data to ASCII. Binary columns remain in binary format. Each record is preceded by its length in bytes and is followed by a comma delimiter. A carriage return/line feed terminates each record.

➤ To export data from a Pervasive.SQL database table

1. Select the **Tables** node in the namespace under the database from which you are exporting data.
2. Right-click on the desired table object in the table list in PCC's right pane.
3. Select **Tasks**, then **Export Data** from the shortcut menu.
4. Enter the location of the output file that you want to contain the records exported from the table.

Figure 3-24 Export Wizard - Choose a Destination File Dialog Box

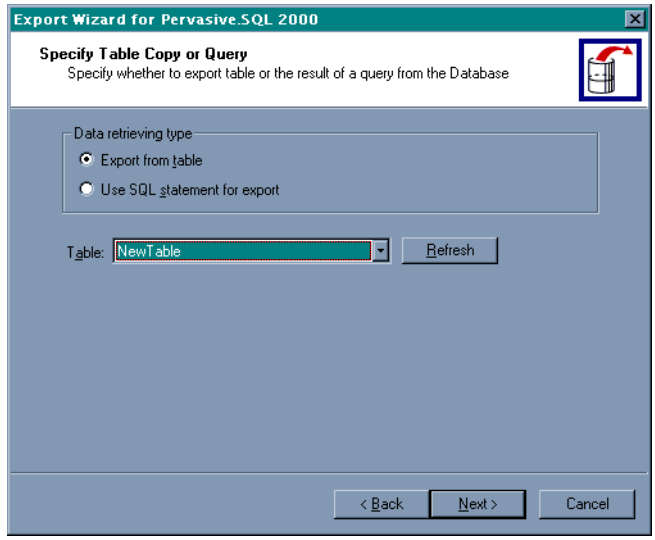


In this example, we are exporting comma delimited records.

Select the box marked **Write column names at first row** if you would like the Export Wizard to create a header row at the beginning of the file that includes the name of each column from the DDF. This would be particularly useful if you are exporting to another application, such as Excel, so that the contents will be identified.

5 Click **Next**. The Export Wizard now displays a dialog box that allows records to be filtered or processed using SQL. The option chosen here is to output the table records unmodified.

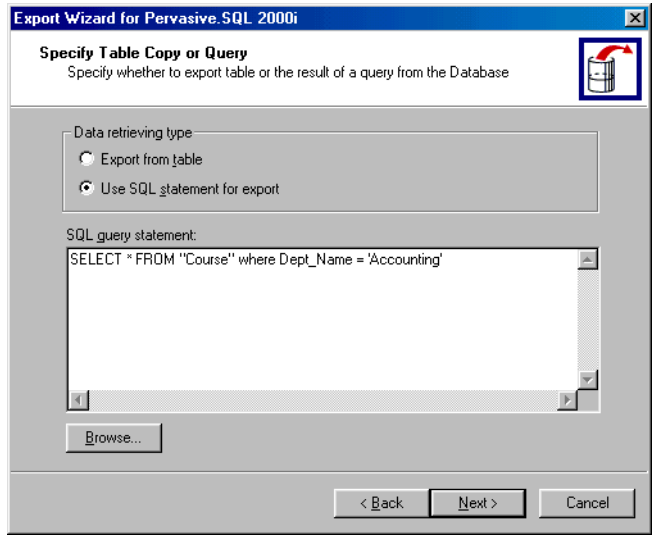
Figure 3-25 Export Wizard - Specify Table Copy or Query Dialog Box



6 Click **Use SQL statement for export** if you would like to filter the output records using SQL (Figure 3-26).

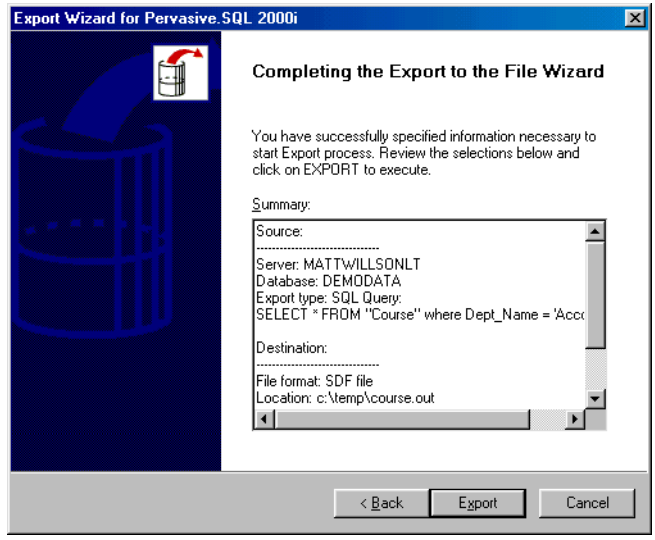
Click the **Export from table** if you want to export the entire table without any specifying any filters.

Figure 3-26 Export Wizard - Use SQL Statement for Export Dialog Box



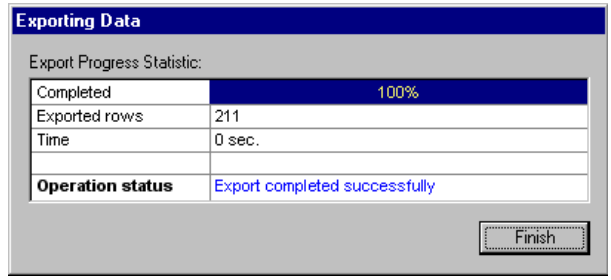
7 Click **Next**. Now that all of the information has been entered, the Export Wizard displays a dialog box listing the actions to be taken.

Figure 3-27 Export Wizard - Completing the Export to the File Wizard Dialog Box



8 Click **Export**. A dialog box provides immediate feedback as the records are exported from the table. Once all records are processed, the **Finish** button becomes enabled. Click it to exit the wizard.

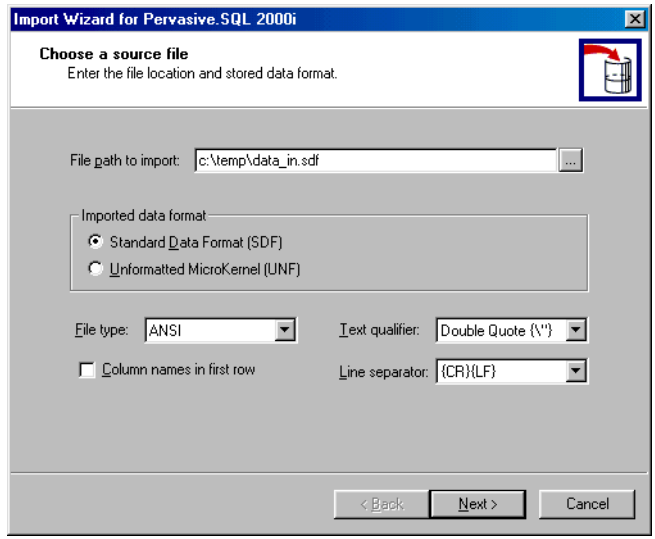
Figure 3-28 Exporting Data Status



➤ To import data into an existing table

1. Click on the **Tables** node in the Namespace under the database into which you are importing data.
2. Right-click on the desired table object in the table list in the Control Center's right pane.
3. Select **Tasks**, then **Import Data** in the shortcut menu.
4. Enter the location of the file that contains the data to be imported into the table. Also indicate the format of the data.

Figure 3-29 Import Wizard - Choose a Source File Dialog Box

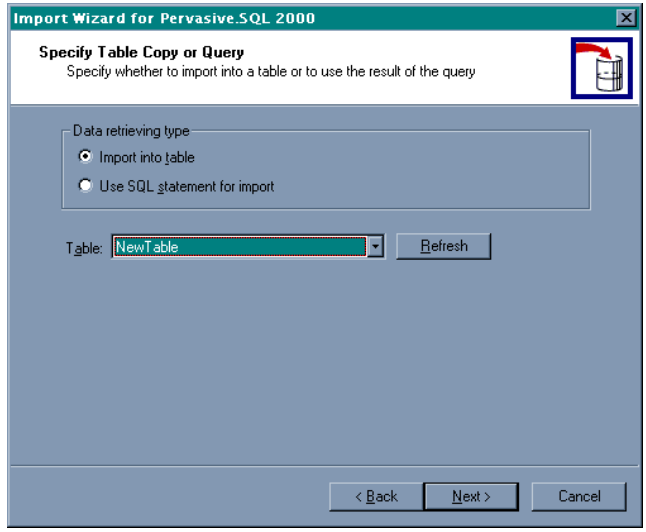


Select the box marked **Column names at first row** if you selected the option **Write column names at first row** when exported the file (Figure 3-24).

Click **Next** when done.

5. Indicate what should be done to the data as it is imported. The options are to import all of the data and insert it into the table or to specify an SQL statement, which gives you control over how the import data fields correspond to the columns in the table.

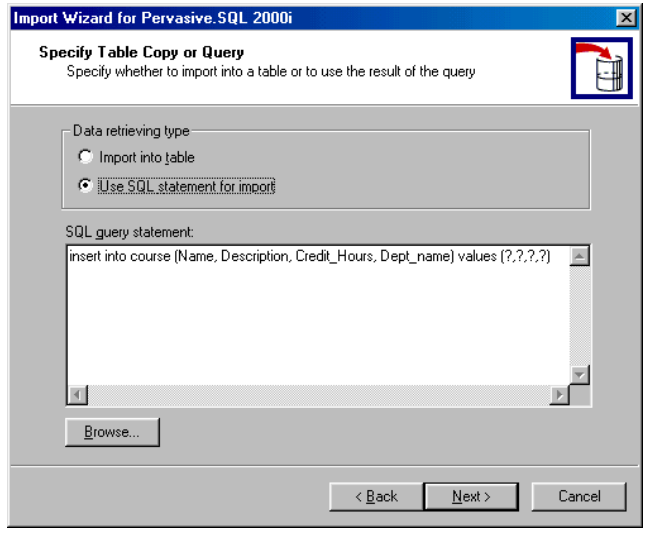
Figure 3-30 Import Wizard - Specify Table Copy Dialog Box



6 Alternatively, you can click **Use SQL statement for import** if you would like to use a SQL statement to control how the import file columns correspond to the table columns. In the example shown in Figure 3-31, the question marks (?, ?, ?, ?) represent the fields from the source file.

Select **Import into table** if you selected the option **Export from table** (Figure 3-25) to import the entire table without filters.

Figure 3-31 Import Wizard - Use SQL Statement for Import Dialog Box



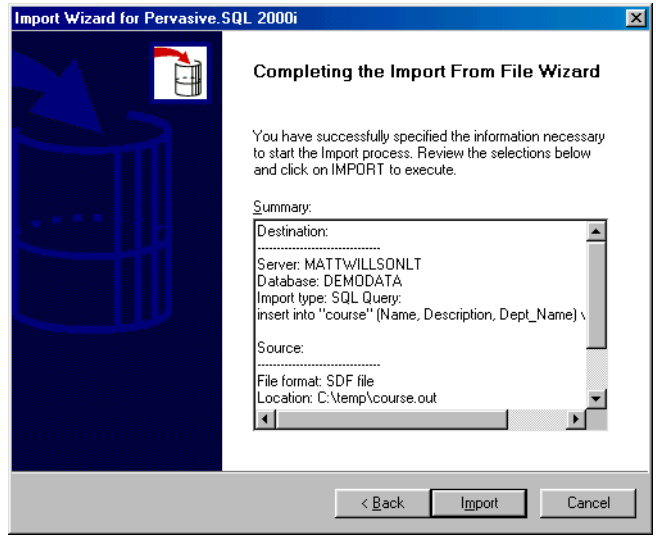
You cannot import fewer fields than exist in the table or in the import file. Both the import file and the table must have the same number of columns. The only factor you can control with the INSERT statement is how the import file columns correspond to the table columns.

The first named column receives the data from the first column in the import file, the second named column receives the data from the second column in the import file, and so on.

For example, if you want the data in the first column of the import file to go into the third column of your table, you must put the name of the third column first in the column list.

7 Now that all of the information has been entered, click **Next**. The Import Wizard displays a dialog box listing the actions to be taken.

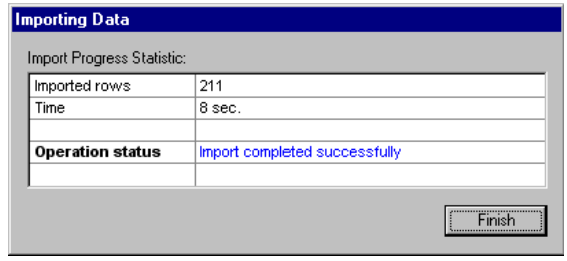
Figure 3-32 Import Wizard - Completing the Import from File Wizard Dialog Box



To stop the import, click **Cancel**, or else click **Import**.

8 A dialog box provides immediate feedback as the records are imported into the table. Once all records are processed, the **Finish** button becomes enabled. Click it to exit the wizard.

Figure 3-33 Importing Data Dialog Box



Checking Consistency and Referential Integrity

Checking the consistency and the integrity of a file and displaying the constraints on a file is a simple process using the Check Database wizard in PCC. Using this wizard, you can perform a consistency test, a referential integrity (RI) test or an index consistency test.

A consistency test checks logical and physical consistency of the selected database, including table and indexes; a referential integrity test checks referential integrity constraints applied to the selected database.

To perform the tests, you must be logged into the database using its data source name.

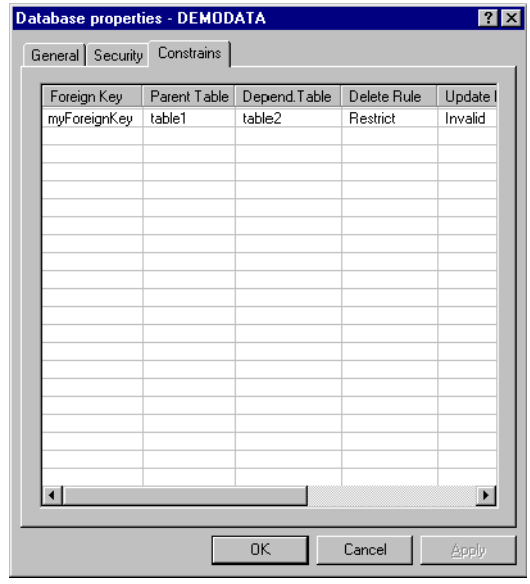
If you are performing the RI test, your database must be named, stored on the same server as the SRDE, and have referential constraints defined (whether or not RI is enabled).

Listing Referential Constraints

➤ **To produce a report that lists all foreign key definitions and related information for any database**

1. Right-click on the database in the PCC Namespace whose referential constraints are to be viewed.
2. Select **Properties** from the shortcut menu.
3. Click on the tab marked **Constraints.** The following dialog box appears:

Figure 3-34 Database Properties



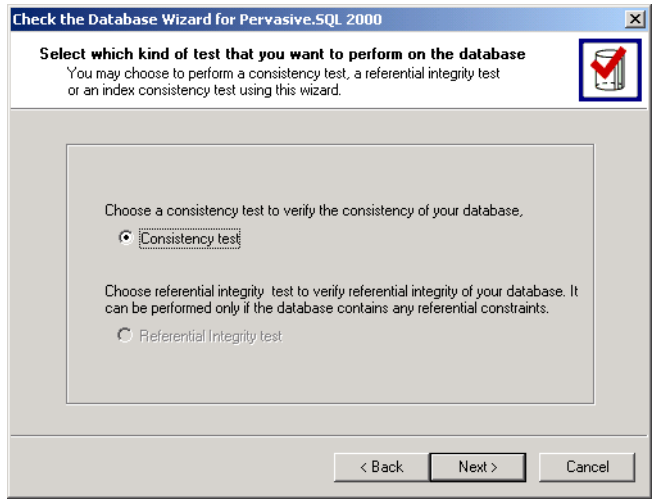
Click **OK** to close the window.

Checking Consistency

➤ To check whether the table definitions in your DDFs match the file structure of the data files

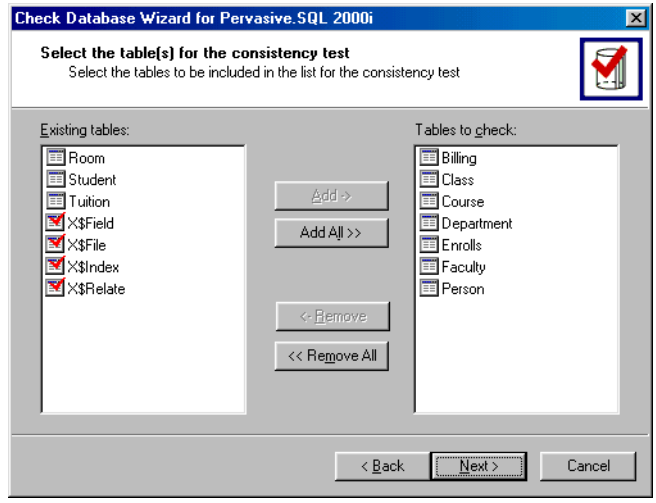
1. Right-click on the database in the PCC Namespace whose referential integrity is to be checked.
2. Select **Tasks**, then **Check Databases** from the shortcut menu. The following dialog box appears:

Figure 3-35 Check Database Dialog Box



3 Click **Consistency test**, then click **Next**. The select table dialog box allows you to specify which tables of the database are to be checked:

Figure 3-36 Select Tables for the Referential Integrity Test Dialog Box



Select the files you want to check, click **Next**, and follow the remaining prompts.

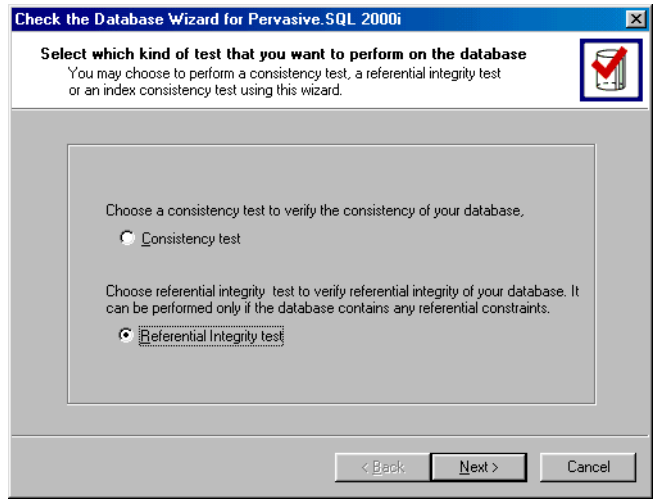
Checking Referential Integrity

➤ To check for orphan rows in a database or verify the consistency of a data file's referential integrity constraints

1 Right-click on the database in the PCC Namespace whose referential integrity is to be checked.

2 Select **Tasks**, then **Check Databases** from the shortcut menu. The following dialog box appears:

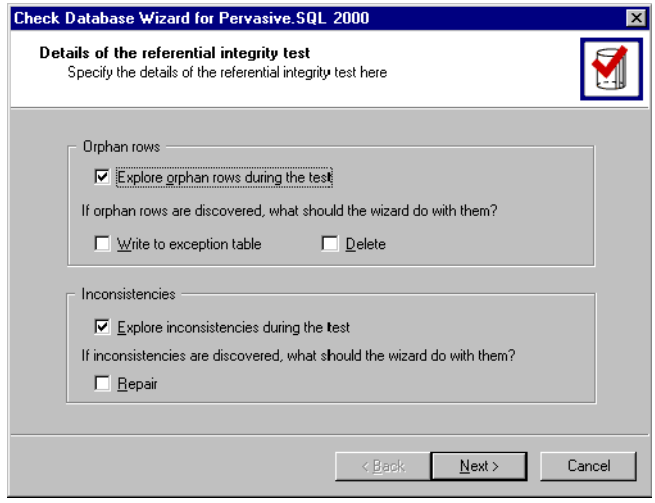
Figure 3-37 Check Database Dialog Box



3 Select **Referential Integrity test** and click **Next**.

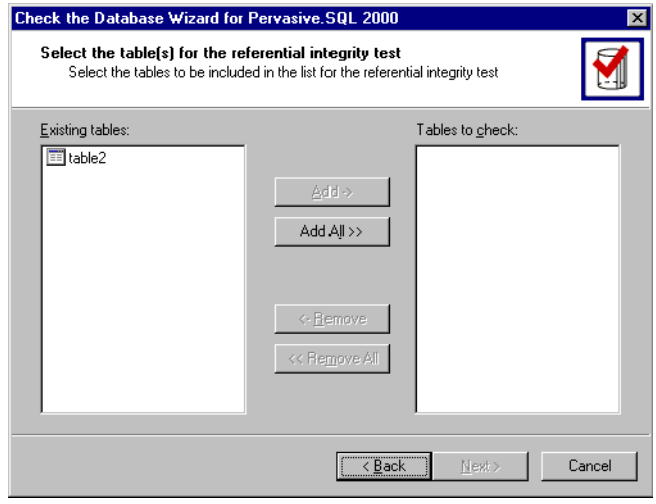
4 The **Details** dialog box provides options to specify what actions should be taken when inconsistencies or orphan rows are found. Select any desired options and click **Next**.

Figure 3-38 Details of the Referential Integrity Test Dialog Box



5 The select table dialog box allows you to specify which tables of the database are to be checked:

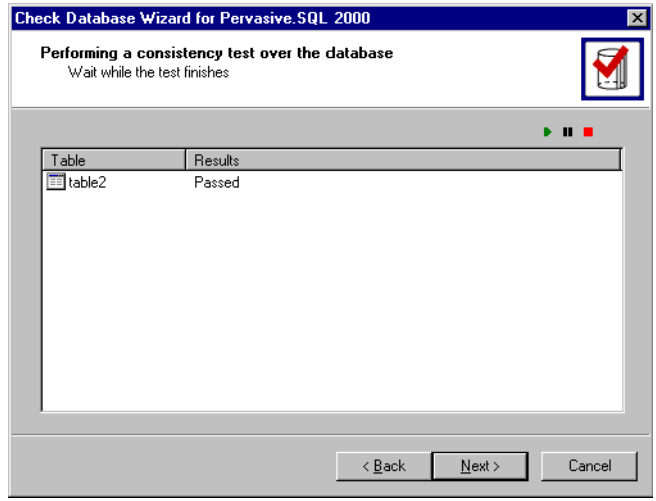
Figure 3-39 Select Tables for the Referential Integrity Test Dialog Box



Make your selections, then click **Next**.

6 The resulting dialog box shows the results of the referential integrity check on the tables as the wizard checks each table selected:

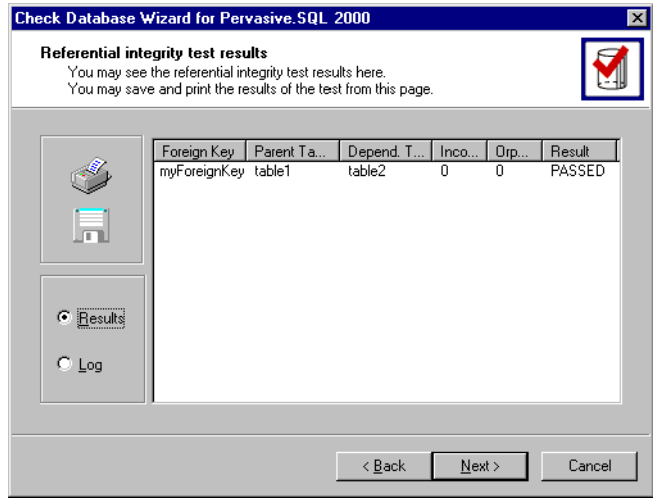
Figure 3-40 Check Database Results Dialog Box



When the wizard has finished checking all tables, click **Next**.

7 The Check Database Wizard's result dialog box shows a complete list of foreign keys and the results of each test. You can save the test results to disk or print them out at this time:

Figure 3-41 Referential Integrity Test Results Dialog Box



8 Follow the wizard prompts until the you receive confirmation that the tests are complete.

Exception Tables

By default, SQL Data Manager checks for orphan rows but does not write them to an exception table or delete them.

You can change the relevant settings as follows:

Orphan Rows	If you do not want SQL Data Manager to check for orphan rows, deselect this option button.
Write to Exception Table	If you want SQL Data Manager to create an exception table, select this option.
Delete	If you want SQL Data Manager to delete the orphan rows after writing them to the exception table, select this option.

If the referential integrity test was run with the option to save orphan rows to an exception table, then the exception table becomes a part of the database with the same location and file name as the original file, but with an .EXC extension. For example, if SQL Data Manager generated an exception table on table2, the exception table would be named EXC_table2 and would be stored in the data file named TABLE2.EXC.

The first field in the exception table is an index and contains the parent table name. The remainder of each row contains the same field values as the original orphan row and can contain up to 4,090 bytes. SQL Data Manager truncates rows larger than 4,090 bytes. You can use SQL Data Manager to review the exception table by issuing SQL statements, just as you would with any other table.

Inconsistencies of Referential Definitions

Optional: By default, SQL Data Manager checks for but does not repair any inconsistencies between the information in the data dictionary and that in the individual data files.

You can change the relevant settings as follows:

Inconsistencies	If you do not want SQL Data Manager to check for inconsistencies, deselect this option button.
Repair	If you want SQL Data Manager to repair inconsistencies, select this option.

For example, you may create an inconsistency if you move a data file from one database to another, because the old data source name stored in the data file does not match the new data source name stored in the data dictionary. The Check Database wizard checks for the following inconsistencies:

Data Source Names	Checks the data source name stored in the data file against the data source name stored in the data dictionary.
Primary Key	Checks the number of referencing foreign keys stored in the data file against the referential constraints stored in the data dictionary.
Foreign Keys	Checks the number of foreign keys defined in the data file against the referential constraints stored in the data dictionary.

To repair inconsistencies, the Check Database wizard updates the information stored in the individual data files to match that in the data dictionary.

chapter

4

Unix Supplementary Documentation

Additional Information on Unix Utilities

This chapter discusses information specific to Pervasive.SQL 2000i for Unix. It also maps out which sections in the existing documentation that are not used with the Unix product.

The chapter contains the following sections:

User Manual Exclusions for Unix

Because the Unix platform is unique, Unix users are advised to look through this section when referring to *Advanced Operations Guide* and the rest of *Pervasive.SQL User's Guide*. This section contains exclusions and differences that apply only to the Unix release of Pervasive.SQL.

Utilities

All Win32 utilities must be run from a Windows client. If a utility must be run at the server, then its Unix equivalent will be found in this manual.

Changes and Exclusions

The following sections of the *Advanced Operations Guide* do not work in the same manner for Unix. Any necessary explanations are given below.

- The sections, "Understanding the Pervasive Component Architecture" of the *Advanced Operations Guide* regarding "Overview of Smart Components", "Component Identification", or "Unique Component Naming" do not apply to Pervasive.SQL 2000i.
- The section, "Understanding the Pervasive Component Architecture" of the *Advanced Operations Guide* regarding "Pervasive.SQL Event Logging" is different for Pervasive.SQL 2000i on Unix.

  Pervasive.SQL 2000i uses the standard Unix logging system. Depending on the configuration of `/etc/syslog.conf`, messages are sent to the `syslogd` daemon, which does one of the following:

  - logs it in an appropriate system log
  - writes it to the system console
  - forwards it to a list of users
  - forwards it to `syslogd` on another host over the network

  More information can be found in the man pages for `syslogd` and `syslog.conf`.

- The section, “Tuning Components Using the Configuration Utility” of the *Advanced Operations Guide* regarding the settings for “System Cache” and “Accept Remote Requests” are ignored in Pervasive.SQL 2000i.
- The chapter, “Manipulating Pervasive.SQL 2000i Data” of the *Advanced Operations Guide* works only on the client for Pervasive.SQL 2000i.

Man Pages

The following man pages are available for the Pervasive.SQL 2000i for Unix product:

```
btadmin
butil
dbmaint
dsnadd
mkded
sqlmgr
ucutil
```

To make these man pages available, add `$PVSW_ROOT/man` to your MANPATH environment variable. If you need more detailed information on a utility or application, see "Available Utilities" on page 4-5.

Available Utilities

The following are utilities available in Pervasive.SQL 2000i. These are specific to the Unix product and are not found in the *Advanced Operations Guide*.

btadmin

Description

The btadmin utility is used to create and update the flat file `btpasswd, which` stores user names and passwords for authentication of Pervasive.SQL users. Users given administrator rights can monitor engine status and configure the engine remotely.

Synopsis

`btadmin [ -p` *password*`] [a+] [a-] [-r]` *username*

Availability

- Sun Solaris 2.6, Solaris 7 (SPARC only)
- Red Hat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

`-p`	Specify the password. If this option is not specified, you will be prompted to enter the password.
`a+`	Gives administrator rights for this user.
`a-`	Removes administrator rights for this user.
`-r`	Remove user name from btpasswd file.

username

Creates or updates the username in the `btpasswd` file. If *username* does not exist in this file, an entry is added. If it does exist, the password is changed.

See Also

mkded(1), butil(1)

Notes

To administer the engine from a remote workstation, you must supply a user name and password. Upon initial installation of Pervasive.SQL 2000i, the supplied default is admin with an empty password.

Use btadmin to add more administrators:

```
% btadmin [-p password] [a+] username
```

This utility creates a record in btpasswd for user *username* with password *password* (if option -p is not used, then you will be asked to enter a password). If a user already exists, then his password is changed as specified.

By default a user is created without administration permissions. You can use the a+ option to give administration rights to the user. You can remove this right by using a-.

To remove a user record from the password file, enter:

```
% btadmin -r username
```

Every time the btpasswd file is changed, the previous version is backed up to btpasswd-.

butil

Description

The Pervasive.SQL Maintenance Utility, or `butil`, is a command line utility that performs command file and data manipulations on a MKDE.

The maintenance utility performs the following file and data manipulations:

- Starts and stops continuous operations for use in performing server backups.
- Recovers changes made to a file between the time of the last backup and a system failure.
- Imports and exports ASCII, unformatted, and SDF sequential data.
- Copies data between data files.
- Returns MKDE version information.

Continuous operation is an MKDE feature that enables you to back up files while they are in use by Pervasive.SQL-based applications. Two Maintenance Utility commands, `startbu` and `endbu`, begin and end continuous operation on a file or set of files.

Synopsis

```
butil
    -clone outputFile sourceFile [/Oowner]
    -clrowner sourceFile [/Oowner]
    @commandFile [commandOutputFile]
    -copy sourceFile outputFile [/Oowner1] [/Oowner2]
    -create sourceFile descriptionFile [Y | N]
    -drop sourceFile <keyNumber | SYSKEY> [/Oowner]
    -endbu </A | sourceFile | @listFile>
    -index sourceFile indexFile descriptionFile [/Oowner]
    -load unformattedFile outputFile [/Oowner]
    -recover sourceFile unformattedFile [/Oowner]
    -rollfwd sourceFile unformattedFile [/Oowner]
    -save sourceFile unformattedFile
    [Y indexFile | N <keyNumber | -1>] [/Oowner]
```

```
-setowner sourceFile owner level
-sindex sourceFile <descriptionFile | SYSKEY>
[keyNumber] [/Oowner]
-startbu <sourceFile | @listFile>
-stat sourceFile [/Oowner]
-ver
```

Availability

- Sun Solaris 2.6, Solaris 7 (SPARC only)
- Red Hat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

Note Maintenance Utility command options are not case sensitive unless the option is a filename.

If you run `butil` without specifying a command option or with an invalid command option, a usage message is printed. The usage message indicates that there is an optional `/S` command line argument to `butil`. This argument, which is used to suppress screen-at-a-time printing on NetWare, is ignored under Unix.

Commands

`-clone` *outputFile sourceFile* `[/O`*owner*`]`

The clone command creates a new, empty Pervasive.SQL formatted file with the same file specifications as an existing file (including any supplemental indexes, but excluding the owner name). The new data file includes all of the defined key characteristics (such as key position, key length, or duplicate key values) contained in the existing file. Unless specified, the new data file will not have an owner name.

`-clrowner` *sourceFile* `[/O`*owner*`]`

The `clrowner` command clears the owner name of a Pervasive.SQL data file.

`@`*commandFile* `[`*commandOutputFile*`]`

The Maintenance Utility allows you to specify the `butil` command options in a *commandFile* that can be specified to `butil`. An `<end>` keyword must be placed after each command option in the file.

The following is an example command file:

```
create xface.btr xface.dsc
<end>
stat xface.btr
<end>
```

`-copy` *sourceFile outputFile* `[/O`*owner1*`] [/O`*owner2*`]`

The `copy` command copies the contents of one Pervasive.SQL formatted file to another. `Copy` retrieves each record in the source data file and inserts it into the output data file. The record size must be the same in both files. After copying the records, `copy` displays the total number of records inserted into the output data file.

`Copy` performs the same function as `recover` and `load` in a single step.

`-create` *sourceFile descriptionFile* `[Y | N]`

The `create` command generates an empty Pervasive.SQL formatted data file using the characteristics specified in the description file. If the path name is the name of an existing Pervasive.SQL formatted file, this command creates a new, empty Pervasive.SQL formatted file in place of the existing Pervasive.SQL formatted file. Any data that was stored in the existing file is lost and cannot be recovered.

`Y|N` indicates whether to replace an existing file. If you specify N but a Pervasive.SQL formatted file with the same name exists, the utility returns an error message and does not create a new file. The default is Y.

`-drop` *sourceFile* `<`*keyNumber* `| SYSKEY> [/O`*owner*`]`

The drop command removes an index from a Pervasive.SQL formatted data file and adjusts the key numbers of any remaining indexes, subtracting 1 from each subsequent key number. If you do not want to renumber the keys, you can add 128 to the key number you specify to be dropped.

`-endbu </A |` *sourceFile* `| @`*listFile*`>`

The `endbu` command ends continuous operation on a data file or set of data files previously defined for backup. Execute this command after you have issued the `startbu` command and your backup utility has finished running. To back up data files using continuous operation, first issue the `butil –startbu` command, followed by the data file or set of data files. Next, run your backup program. Then, stop continuous operation by using the `butil –endbu` command.

`/A` stops continuous operation of all the files defined for the backup.

`-index` *sourceFile indexFile descriptionFile* `[/O`*owner*`]`

The `index` command builds an external index file for an existing Pervasive.SQL formatted file, based on a field not previously specified as a key in the existing file. Before you can use the `index` command, you must create a description file to specify the new key characteristics. The records in the new file consist of the following:

- The 4-byte address of each record in the existing data file.
- The new key value on which you want to sort.

If the key length you specify in the description file is 10 bytes, the record length of the external index file would be 14 bytes (10 plus the 4-byte address).

`-load` *unformattedFile outputFile* `[/O`*owner*`]`

The `load` command inserts records from an unformatted input sequential file into a Pervasive.SQL formatted file. It performs no conversion on the data in the input sequential file. After the utility transfers the records to the data file, it displays the total number of records loaded.

Before running the load command, you must create the input sequential file and the data file. You can create the input sequential file using a standard text editor or an application; the input sequential file must have the required file format (as explained below). You can create the data file using either `butil -create` or `butil -clone`.

Records in the input sequential file must be in the following format:

- The first field must be a left-adjusted integer (in ASCII) that provides the length of the record. This field does not include the end-of-line markers. For files with fixed length records, the length you specify should equal the record length of the data file. For files with variable length records, the length you specify must be at least as long as the minimum fixed length of the data file.
- A separator (either a comma or a blank) must follow the length field.
- The record data follows the separator. The length of the data must be the exact number of bytes specified by the length field.
- An end-of-line marker must terminate each line. The end-of-line marker is not included in the length value at the beginning of the line. Note that the Solaris version of `butil` will accept both PC style end of line marker (e.g., carriage return and new line) and Unix style end of line marker (e.g., new line).

The Solaris version of `butil` accepts both DOS and Unix styles with respect to end-of-file markers. DOS files use CTRL+Z, and Unix does not have an end-of-file marker.

`-recover` *sourceFile unformattedFile* `[/O`*owner*`]`

The `recover` command extracts data from a Pervasive.SQL formatted file and places it in a sequential file that has the same format as the input sequential file used by the `load` command. This command is often useful for extracting some or all of the data from a damaged Pervasive.SQL formatted file. The `recover` command may be able to retrieve many, if not all, of the file's records. You can then use the `load` command to insert the recovered records into a new, undamaged Pervasive.SQL formatted file.

`-rollfwd` *sourceFile* `[ /L`*dumpfile* `/W`*dumpfile* `/T`*datalength* `/E`*keylength* `/H /V /A /O`*owner*`]`

The `rollfwd` command recovers changes made to a data file between the time of the last backup and a system failure. The MKDE stores the changes in a log.

If a system failure occurs, you can restore the backup copy of your data file and then use the `rollfwd` command, which applies all changes stored in the log to your backup copy.

`-save` *sourceFile unformattedFile*
`[Y` *indexFile* `|` `N` `<`*keyNumber* `|` `-1>]` `[/O`*owner*`]`

The `save` command retrieves records from a Pervasive.SQL formatted data file using the specified index path and places them in a sequential file that is compatible with the required format for the load command. `Save` generates a single record in the output sequential file for each record in the input data file. Upon completion, save displays the total number of records saved.

The Maintenance Utility performs no conversion of the data in the records. Therefore, if you use a text editor to modify an output file containing binary data, be aware that some text editors may change the binary data, causing the results to be unpredictable.

indexFile is used if you do not want to save records by using the default of the lowest key number.

keyNumber is used if you do not want to save records using the default of the lowest key number.

`-1` is the specification for saving the records in physical order using the Pervasive.SQL Step operations.

`-setowner` *sourceFile owner level*

The `setowner` command creates an owner for a Pervasive.SQL formatted file and assigns an access restriction level to that owner. For more information about owner names, see *Advanced Operations Guide*.

`-sindex` *sourceFile* `<`*descriptionFile* `|` `SYSKEY >` `[`*keyNumber*`]` `[/O`*owner*`]`

The `sindex` command creates an additional index for an existing Pervasive.SQL formatted file. The key number of the new index is one higher than the previous highest key number of the data file. An exception is if a `drop` command previously removed an index without renumbering the remaining keys, thus producing an unused key number. In this case, the new index receives the first unused number.

Before you can use the `sindex` command, you must create a description file to key specifications for the index.

keyNumber is used when you do not want to save records using the default of the lowest key number.

`-startbu <`*sourceFile* | *@listFile*`>`

The `startbu` command places a file or set of files into continuous operation for backup purposes. To back up files using continuous operation, first issue the `butil -startbu` command, followed by the data file or set of data files.

Next, run your backup program. Then issue the `butil -endbu` command to stop continuous operation.

When you place a data file into continuous operation mode, the MKDE creates a temporary file with the same name as the data file, but with a `.^^^` extension. Therefore, do not create multiple data files with the same names but different extensions. For example, do not use a naming scheme such as INVOICE.HDR and INVOICE.DET for your Pervasive.SQL data files.

`-stat` *sourceFile* `[/O`*owner*`]`

The `stat` command reports the defined characteristics of a data file and statistics about the file's contents.

`-ver`

The `ver` command returns the version number of the MKDE loaded at the server.

Variables

descriptionFile
is the path name of a description file containing the description of the index you wish to use. Is an ASCII text file containing information the Maintenance Utility needs to perform create and index operations. Description files are made up of one or more elements, where each element consists of a keyword followed by an equal sign (=) and a value (with no space separator). Please refer to the *Advanced Operations Guide* for details of description files.

indexFile
is the path_name of the index file in which the MKDE stores the external index.

keyNumber
is the key number (other than 0) of the index specified.

level
is the type of access restriction for the data file. The possible values for this parameter are as follows:

0 - Requires an owner name for any access mode (no data encryption).

1 - Permits read access without any owner name (no data encryption).

2 - Requires an owner name for any access mode (with data encryption).

3 - Permits read access without an owner name (with data encryption).

listFile
is the name of the text file containing the pathnames of files to be included in either the `endbu` or `startbu` operations. These pathnames must be separated by a space or end-of-line marker.

outputFile
is the path name of the data file into which you want to insert records. The output file can be empty or have existing data.

owner
is the owner of any specified files, if any. Can be used more than once in a command line when more than one file is specified. For example, `copy sourceFile outputFile [/Oowner1][/Oowner2]`. The MKDE enables you to restrict access to a file by specifying an owner name. Since owner names are optional, the files you use with this utility may or may not require an owner name. Owner names are case sensitive.

sourceFile
is the path_name of an existing data file, except when using the `create` command, which creates an empty data file. Generally refers to a Pervasive.SQL file.

unformattedFile
is the pathname of an ASCII sequential file.

Examples

- The following command creates a Pervasive.SQL formatted file named `patients.btr` using the description provided in the `BUILD.dsc` description file.

```
% butil -create patients.btr BUILD.dsc
```

- The following command copies the records in `patients.btr` to `newpats.btr`. The `patients.btr` input file does not require an owner name, but the `newpats.btr` output file uses the owner name Pam.

```
% butil -copy patients.btr newpats.btr /O /OPam
```

- The following command creates the `newapp.btr` file by cloning the `patients.btr` file.

```
% butil -clone newapp.btr patients.btr
```

- The following command clears the owner name for the `newpats.btr`. The owner name for the file `newpats.btr` is Pam.

```
% butil -clrowner patients.btr /OPam
```

- The following command sets the owner name for the `newpats.btr` to Ron with a restriction level of 1.

```
% butil -setowner patients.btr /ORon 1
```

- The following command creates an external index file called `newpats.idx` using a data file called `patients.btr`. The `patients.btr` file does not require an owner name. The description file containing the definition for the new key is called `NEWidx.dsc`

```
% butil -index patients.btr newpats.idx
NEWidx.dsc
```

The description file shown below defines a new key with one segment. The key begins at byte 30 of the record and is 10 bytes long. It enables duplicates, is modifiable, is a string type, and uses no alternate collating sequences.

```
position=30 length=10 duplicates=y modifiable=y
type=string alternate=n segment=n
```

The following two examples illustrate how to use the save command to retrieve records from a data file.

- The first example uses the `newpats.idx` external index file to retrieve records from the `patients.btr` data file and store them in an unformatted text file called `patients.sav`.

```
% butil -save patients.btr patients/.sav Y
newpats.idx
```

- The next example retrieves records from the `patients.btr` file using key number 3 and stores them in an unformatted text file called `patients.sav`.

```
% butil -save patients.btr patients/.sav N 3
```

- The following example loads sequential records from `patients.adr` file into the `patients.btr` file. The owner name of the `patients.btr` file is Sandy.

```
% butil -load patientsa.adr patients.btr /OSandy
```

- The following example adds an index to the `patients.btr` file. The name of the description file is `suppidx.dsc`. The owner name of the `patients.btr` is Ron.

```
% butil -sindex patients.btr suppidx.dsc /ORon
```

See Also

`butil(1), syslogd(1)`

API Programmer's Reference – describes the Pervasive.SQL API

$PVSW_ROOT/README – contains useful configuration information and release notes ($PVSW_ROOT denotes the directory where Pervasive.SQL for Unix is installed)

dbmaint

Description

The dbmaint utility manages named databases.

Synopsis

`dbmaint a | d | l [-n`*Dbname*`] [-a] [-b] [-i] [-e] [-l`*dictpath*`] [-d`*datapath*`]`

add new database name	a -n*Dbname* [-b] [-i] [-e] [-l*dictpath*] [-d*datapath*]
delete database name	d -n*Dbname*
list database names	`l [-a]`

Availability

- Sun Solaris 2.6, 7 (SPARC only)
- Red Hat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

Commands

`add, a`	add dbname
`del, d`	delete dbname
`list, l`	list dbnames

Options

`-b`	create bound database
`-i`	create database with relational integrity
`-e`	do not create dictionary files for database
`-n`*DBName*	specify database name

`-l`*dictpath*	specify dictionary path
`-d`*datapath*	specify datapath
`-a`	show detail information about dbnames in database list

Examples

To create DBName TEST with relational integrity type:

```
% dbmaint a -i -nTEST
```

Note Unless a datapath is specified, the new database will be in the default location, $PVSW_ROOT/data. Likewise, if a dictionary path is not specified, the dictionary will be created in the default location.

To delete the same database, type:

```
% dbmaint d -nTEST
```

To list all DBNames with full information, type:

```
% dbmaint l -a
```

See Also

`butil(1), btadmin(1), syslogd(1), smb.conf(5)`

API Programmer's Reference – describes the Pervasive.SQL API

dsnadd

Description

`dsnadd` simplifies the setup of a new ODBC data source that uses the Pervasive ODBC Client Interface driver. It modifies the `odbc.ini` file by adding appropriate information about the new data source. For the latest information on `dsnadd`, see the man page.

Synopsis

To add a data source, execute the following command:

```
dsnadd -dsn=server_DSN -db=DB_name
    [-ini=inifile] [-desc=DSN_description]
    [-drv-desc=driver_description]
```

or

```
dsnadd -dsn=myDSN -desc=datasource
    -db db_name -host=psqlhost -sdsn=svDSN
```

myDSN
is a name you want to assign to the new data source.

datasource
is any string to describe the data source.

psqlhost
is the name of the network host where your Pervasive.SQL is installed.

svDSN
is the name of the data source on the Pervasive.SQL host.

Availability

- Sun Solaris 2.6, 7 (SPARC only)
- RedHat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

`-help`	gives help on the `dsnadd` utility
`-odbc-ini, -ini`	ODBC.ini file name (`$HOME/.odbc.ini`)
`-dsn-name, -dsn`	Data Source Name
`-dsn-desc, -desc`	DSN description `[`[*driver_description*`:]` *host*`/`*DSN*`]`
`-srv-host, -host`	server host name
`-srv-port, -port`	server port number [1583]
`-sdsn`	server DSN [*localDSN*]
`-drv-path, -drv`	`[/usr/local/psql/lib:$HOME/lib]` for Linux `[/opt/PVSWpsql/lib:$HOME/lib]` for Solaris
`-drv-desc`	driver description [Pervasive ODBC Client interface]

Examples

To add a Client DSN named `localDSN` that references an Engine DSN named `remoteDSN` on a machine named `server12`, enter the following command:

```
dsnadd -dsn=localDSN -desc=any_text -host=server12
    -svr-dsn=remoteDSN
```

See Also

`btadmin(1)`

mkded

Description

`mkded` is the command used to start and stop the Pervasive.SQL database server, including the underlying MKDE, as either a daemon process or in console mode.

In daemon mode, Pervasive.SQL runs as a background daemon process. As a daemon, mkded relinquishes the control terminal and becomes the owner of the process group. Informational, warning, and error messages produced by the MKDE are printed to the console window (/dev/console), or to the system log (see syslogd(1)) if the MKDE cannot write to /dev/console.

When Pervasive.SQL is started in console mode, a `>>` prompt will be displayed allowing you to enter interactive commands. Console mode is functionally equivalent to daemon mode, except that MKDE messages are printed to standard out instead of /dev/console (or the system log).

While the Pervasive.SQL database server is running, the `butil` Maintenance Utility can be executed to perform a variety of common file and data manipulation operations. For example, run `butil -ver` to print information on the Pervasive.SQL version that is currently running.

Synopsis

```
mkded [ -start | -stop | -console | -help ]
```

Availability

- Sun Solaris 2.6, Solaris 7 (SPARC only)
- RedHat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

The options described below are supported by `mkded`. These options are case-insensitive.

`-start`	Runs the Pervasive.SQL database server as a daemon process.
`-stop`	Terminates execution of Pervasive.SQL engine that was previously started in daemon mode with the -start command line argument.
`-console`	Runs the Pervasive.SQL database server in console mode. When Pervasive.SQL is started in console mode, a '>>' prompt will be displayed allowing you to enter interactive commands. The following commands are supported within console mode: `status` prints client and file engine status, `quit` exits the MKDE process `?` displays help information.
`-help`	Provides an informative message on options to the mkded command.

Configuration

Pervasive.SQL for Unix uses a bti.ini file to obtain MKDE configuration parameters. On startup, the MKDE determines the location of this file as follows:

1. If the BTIINI environment variable is defined, then look for bti.ini in the directory specified by this environment variable.
2. If (1) does not apply, then look for bti.ini in the current working directory.
3. If (1) and (2) do not apply, then look for bti.ini in the directory where the Pervasive.SQL database engine executable (`mkded`) resides.
4. If (1), (2), and (3) do not apply, then look for bti.ini in all directories contained in the $PATH environment variable.
5. If (1), (2), (3), and (4) do not apply, then create a default bti.ini in the current working directory. When the MKDE creates a default bti.ini file, an informational message is printed by the MKDE.

The configuration parameters that can be specified in bti.ini are described below. A sample configuration file can be found in

$PVSW_ROOT/samples/config/bti.ini, where $PVSW_ROOT indicates the directory where Pervasive.SQL for Unix is installed.

The bti.ini file may contain several sections, denoted by section headings, each containing a set of configuration parameters. For this release, only the [MicroKernel] section is supported (although a [Database Names] section is created by default in the bti.ini file).

The MicroKernel performs a case-insensitive search for the name and value of each configuration parameter entry in the bti.ini file. Some entries can have more than one value. In such case, they are separated by a vertical bar (|). Values that are generic (such as, pathnames) are enclosed in angular brackets (<>).

See Also

`butil(1), btadmin(1), syslogd(1), smb.conf(5)`

API Programmer's Reference – describes the Pervasive.SQL API

$PVSW_ROOT/README – contains useful configuration information and release notes ($PVSW_ROOT denotes the directory where Pervasive.SQL 2000i is installed).

$PVSW_ROOT/doc/apinotes.txt – contains guidelines for programming with the Btrieve API on Unix RISC systems.

Notes

Daemon Mode

To start up the Pervasive.SQL database server as a daemon process (assuming $PVSW_ROOT/bin is in your path) run

```
% mkded -start
```

As a daemon, `mkded` relinquishes the control terminal and becomes the owner of the process group. To stop the database server process, run

```
% mkded -stop
```

Note The package installation script automatically launches `mkded` and incorporates its execution into rc scripts for automatically starting and stopping the database server as part of the system boot sequence.

Most MKDE errors are handled by returning status codes to applications that call the BTRV() and BTRVID() functions. Occasionally, special informational, warning, and error messages are printed directly by the MKDE (for example, creation of default configuration file, roll forward after for crash recovery, unable to open directory for log files). When running in daemon mode, the MKDE prints these messages to the Unix system's console window (/dev/console). You can bring up a console window on your system using

```
% xterm -C &
```

Depending on your system configuration, root access may be required to bring up a console window. If the MKDE is unable to print to /dev/console, then it will print messages to the Unix system log file. You can find out more about the system log by running `man syslogd`.

Console Mode

To start up the Pervasive.SQL database server in console mode (assuming $PVSW_ROOT/bin is in your path) run

```
% mkded -console
```

After executing this command, a `>>` prompt will be displayed allowing you to enter interactive commands. The `status` command prints client and file engine status. Entering `quit` will exit the MKDE process.

Console mode is functionally equal to daemon mode, except that MKDE messages are printed to standard out instead of /dev/console (or the system log).

sqlmgr

Description

`sqlmgr` serves remote requests from clients and returns them to the Pervasive.SQL database engine. To start `sqlmgr`, bti.ini should be placed into directory `/usr/local/psql/etc` for Linux and `opt/PVSWpsql` for Solaris. This file contains ODBC settings and description of ODBC DSNs for the server.

Port and network protocol in `bti.ini` should be specified as below:

```
[SQLManager]
MgrPort=1583
MgrUseTransport=TCP
```

Each DSN should be described in `odbc.ini` as follows:

```
[DSN name]
[Driver=/usr/local/psql/lib/libsrde.so |
Driver=/opt/PVSWpsql/lib/libsrde.so]
Description=Test Pervasive database
DBQ=DBName
```

To create a DBName see “dbmaint” on page 4-17.

Synopsis

```
sqlmgr [ -start | -stop | -console ]
```

Availability

- Sun Solaris 2.6, Solaris 7 (SPARC only)
- RedHat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

sqlmgr -start	starts sqlmgr
sqlmgr -stop	stops sqlmgr
sqlmgr -console	starts sqlmgr in console mode

Examples

1 SQLMGR required settings:

```
; configure sqlmgr port
    [SQLManager]
    MgrUseTransport=TCP
    MgrPort=1583
```

2 Server data source – the one to which remote calls are redirected:

```
; Test
    [dsn.srv]
    [Driver=/usr/local/psql/lib/libsrde.so |
    Driver=/opt/PVSWpsql/lib/libsrde.so]
    Description=Test Pervasive database
    ; DBName – named database
    DBQ=DEMODATA
```

In addition, each data source should be mentioned in the section [ODBC Data Sources] in `odbc.ini` such as the following:

```
[ODBC Data Sources]
    dsnName1=Pervasive.SQL data base
    dsnName2=Pervasive.SQL data base
```

An easy way to verify DBName and DSN configuration settings is to run the supplied odbctest program:

`% /user/local/psql/bin/odbctest DSN=DEMODATA` in Linux

`% /opt/PVSWpsql/bin/odbctest DSN=DEMODATA` in Solaris

See Also

`butil(1), btadmin(1), syslogd(1), smb.conf(5), dbmaint(1)`

API Programmer's Reference – describes the Pervasive.SQL API

Notes

Sqlmgr Daemon Mode

To start up the SQL Manager as a daemon process (assuming $PVSW_ROOT/bin is in your path) run

```
% sqlmgr -start
```

As a daemon, `sqlmgr` relinquishes the control terminal and becomes the owner of the process group. To stop the database server process, run

```
% sqlmgr -stop
```

Note The package installation script automatically launches `sqlmgr` and incorporates its execution into rc scripts for automatically starting and stopping the database server as part of the system boot sequence.

Sqlmgr Console Mode

To start up the SQL database server in console mode (assuming $PVSW_ROOT/bin is in your path) run

```
% sqlmgr -console
```

Console mode is functionally equal to daemon mode, except that system messages are printed to standard out instead of /dev/console (or the system log).

Press `^C` to exit `sqlmgr` console mode.

ucutil

Description

The Maintenance Names Database Utility manages the User License count.

Note This utility must be run from the `root` account.

Synopsis

`ucutil -D`*path* | `-G`*code* | `-K`*key* | `-S` | `[-T`*path*`]`

Availability

- Sun Solaris 2.6, Solaris 7 (SPARC only)
- Red Hat Linux 5.2, 6.0
- Caldera OpenLinux 2.2
- S.u.S.E. Linux 6.1

Options

`-D`*path*	Specifies the path where the license file is located. For example, `ucutil -D/mnt/fd`.
`-G`*code*	Returns the user count for the specified product code. The product code for Pervasive.SQL 2000i is 11. (To get a list of all products and codes, specify the `-G` option with no code.)
`-K`*key*	Specifies a key number with which to increase the user count.
`-S`	Displays the product serial number.
`-T`*path*	Specifies the target directory where Pervasive.SQL is installed, if not in the current directory.

See Also

`butil(1), btadmin(1), syslogd(1), smb.conf(5)`

API Programmer's Reference – describes the Pervasive.SQL API

chapter

5

Basic Troubleshooting

How to Identify and Solve Common Problems

This chapter provides information for troubleshooting and resolving the most commonly-encountered problems.

General Troubleshooting

This section provides some basic troubleshooting procedures to help you rule out possible causes for situations you may encounter. This section covers the following topics:

I get Error 1114 when trying to access my data

or

I get an error saying "'ServerDSN' or 'DBQ' was not found in the connection string"

PCC can access remote server data sources (DSNs) using connections without client DSNs. Many desktop applications, such as Microsoft Excel and Microsoft Access, cannot do this. You must create a client DSN on your local computer to provide access to data on the server through the remote server DSN. To create a client DSN, follow the instructions in "Setting Up Client Access" on page 2-39. You must first make sure that a server DSN exists on the server you want to access.

I get a message saying my Engine components' version is different than my client components' version

When a client requester first connects to an engine, the client requester compares its internal router version with the value returned from the engine by a Btrieve Version (26) call. If the client version is older than the engine, a message dialog box is displayed on the client system with the message "Engine components' Version is different from Client's" along with a suggestion to run Pervasive System Analyzer (PSA). The same message is also logged in the

client's PVSW.LOG file. This message is only a warning. Although the client is not prevented from connecting to the engine in this situation, keep in mind that older clients are not tested against newer engines. Pervasive only guarantees compatibility between engines and clients if the clients are the same version as, or newer than, the engines. When prompted by this message, if you choose not to run PSA and archive your old client components and install a newer client, you can expect the product to behave unpredictably until the client version is equal to or greater than the engine version.

I get Status Code 3111 and 3112 frequently

You may receive this error when attempting to connect to a Workgroup engine on Windows 95 if the machine has not been upgraded to Winsock 2. If you are not running Winsock 2 on your Windows 95 computer with Workgroup engine installed, you should download the WinSock 2 update from Microsoft: http://www.microsoft.com/windows95/downloads/contents/wuadmintools/s_wunetworkingtools/w95sockets2/default.asp.

I can't get to my data on the server engine

If you cannot get to data on the server engine, your most likely causes are:

- The server computer is down or the network has been interrupted
- You do not have operating system rights to access the server, or you are not logged into the correct network
- The client requester is not enabled
- The database server engine is not installed or not running
- The database server is not accepting remote connections
- The remote database does not have a DSN set up to advertise on the network
- The local client does not have a DSN to access the server
- The client or server network configuration is wrong

➤ To determine the actual cause of the failure

Follow the steps below to rule out certain root causes and narrow down the possible sources of failure.

1. From a Windows client, double-click **Network Neighborhood** and see if you can find the server computer that you want to connect to. If you can see the server, you can rule out that the server is down or disconnected from the network.

2. Next, try to map a drive to the file server or open a shared file on the server. If you can successfully connect to the file server and create a file on the mapped drive, then you can rule out lack of operating system rights. You can also rule out failure to login to the correct network. For example, if you have both NetWare and Windows NT servers in your environment, it is possible to be logged into the NetWare network but not the Windows NT network, or vice versa. If you're not logged into a particular network, you can't access the servers on that network at all.

Note If you are trying to create a new database on the server, to use Monitor against the remote server engine, or to use Configuration against the remote server engine, you must have administrative rights on the server, or be a member of Pervasive_Admin. A simple drive-mapping or shared-file read will not tell you whether you have administrative rights. This means you may be able to connect to the file server, but you still may not be able to connect to the database engine with Configuration, Monitor, or Create Database Wizard.

3. The next possibility is that the client requester is disabled.

   Choose **Start | Programs | Pervasive | Pervasive Control Center** to start PCC. Using PCC, double-click the icon that represents your local client computer. Double-click **Configuration** and choose **Client | Access | Requester**. Make sure this setting is set to **On**.

   You can now rule out the requester as the source of the problem.

4. Next, verify that Pervasive.SQL is installed and running on the target server.

   On Windows NT, go to the server console and open the Services Control Panel and verify that "Pervasive.SQL 2000 (relational)" and "Pervasive.SQL 2000 (transactional)" have been started. If not, start these services.

   On Windows 2000, go to the server console and open the Administrative Tools Control Panel and then double-click on the

Services icon. Verify that "Pervasive.SQL 2000 (relational)" and "Pervasive.SQL 2000 (transactional)" have been started. If not, start these services.

On NetWare, enter the command BSTART or MGRSTART at the NetWare prompt. If Pervasive.SQL is not loaded, these commands load Btrieve and the SRDE, respectively. If Pervasive.SQL is already running, you receive the message "Modules already loaded."

On Unix, type the following command at the Unix prompt on the server where the database engine is installed:

```
ps -e | egrep 'mkded|sqlmgr'
```

If the output from the command returns at least one line containing the text "mkde" and at least one line containing the text "sqlmgr," then Pervasive.SQL is running. If you do not see these lines, then you need to be logged into the root account and start the database engine by entering `/etc/rc.d/init.d/psql start` (on Linux) or `/etc/init.d/psql start` (on Solaris).

You can now be certain that the server engine is installed and running.

5 The next step is to ensure that the server engine is accepting remote communication requests.

In PCC, use **Configuration** to make sure that the remote database engine is configured to accept remote requests. If you are having difficulty accessing a Windows NT server engine remotely, then you must use **Configuration** at the server itself. You must have administrative permission on the server (or membership in the Pervasive_Admin group) in order to do so. Connect to the server in PCC, double-click **Configuration** for the target server, then choose **Server | Access | Accept Remote Request**. Be sure the value is set to **On**.

You can now rule out the possibility the server is not accepting remote requests.

6 Note: If your application uses pure Btrieve access only, without ODBC, then skip this step.

If everything checks out so far, but you still cannot get to the data you want to access, make sure a server DSN has been set up for your target data. Using PCC, connect to the server, open the **Databases** folder for that server, and inspect the databases that are present. Make sure one of the databases represents the data you want to access. If so, then a server DSN has been created for your data.

If you do not find the data you want to access, but you know it is on the server, then most likely you need to set up a DSN for the given data. You must have administrative rights on the server (or be a member of the Pervasive_Admin group) to do so.

Right-click on the **Databases** folder for the target server, and choose **New Database**. Follow the instructions in "Setting Up ODBC Database Access" on page 2-14 to set up a DSN for existing data files.

You can now rule out the server DSN as the source of the problem.

7 Note: If your application uses pure Btrieve access only, without ODBC, then skip this step.

If you have performed all the steps above and you still cannot get to your data, the next possibility is lack of a local client DSN for the remote data.

PCC can access remote server DSNs using connections without client DSNs. Many desktop applications, such as Microsoft Excel and Microsoft Access, cannot do this. You must create a client DSN on your local computer to provide access to the remote server DSN. To create a client DSN, follow the instructions in "Setting Up Client Access" on page 2-39. You must first make sure that a server DSN exists on the server you want to access.

You can now rule out the client DSN as the source of the problem.

8 The final task to perform is to ensure that your client and server are communicating on the appropriate network protocols. By default, Pervasive.SQL ships with all network protocols enabled, so connection time may be slow as it tries all protocols, but it should eventually connect. Some application vendors disable the protocols that are not typically used by their application(s).

First, determine what protocols ought to be used on your network. If you have a Unix network or a 100% Microsoft network, then your preferred protocol is TCP/IP. If you have a NetWare network, then you need to find out from your NetWare administrator whether you should be using IPX/SPX, SPXII, or TCP/IP.

Once you know what the protocol should be, you should ensure that your server is using this protocol. You must have administrative rights on the server operating system (or be a member of Pervasive_Admin) to perform this task. Using PCC, connect to the target server. Double-click **Configuration**, and click **Server | Communication Protocols | Supported Protocols**. Click **...** and ensure that the correct vendor and protocol stack is listed in the **Selected** column. Immediately above that setting, choose **ODBC Connection Manager Supported Protocol**. This setting should be set to TCP/IP unless you are configuring a NetWare server that does not have TCP/IP installed.

Ensure that your client is using the same protocol. Using PCC, double-click the icon for your local machine. Double-click **Configuration**, and choose **Client | Communication Protocols | Supported Protocols**. Click **...** and ensure that the correct vendor and protocol stack is in the **Selected** column.

9 If you have performed all of the above tasks with no success at accessing your data, refer to "Pervasive.SQL Resources and Contacts" on page 6-1 for more ways to get help.

Error Messages from PCC

You may receive several different messages when attempting to create or connect to databases in PCC. This section explains the likely causes for some of the most common error messages. This section explains the following messages:

- *Can't retrieve database names. You don't have access rights for the operation* on page 5-8
- *Unable to connect to the specified remote server. Verify that all of the communication components are loaded on the remote server and that there are available sessions and try again* on page 5-9
- *An error was encountered while connecting to the server* on page 5-10

Can't retrieve database names. You don't have access rights for the operation

This error may occur when you are attempting to create a new database on the server. The most likely cause is that you are logged in as an operating system user that has neither administrative rights in the server operating system, nor membership in the Pervasive_Admin group on the server. Another likely cause is that you forgot to enter a user name and password.

Solution: Be sure to enter a user name and password for the remote operating system. You must have administrative rights on the server or be a member of the Pervasive_Admin group in order to create a new database on the server. "Granting Administrative Rights for the Database Engine" on page 2-6 explains how to set up the Pervasive_Admin group.

For Windows NT/2000, be sure that you are set up as a local user on the system, not a network user. Network users have a domain name and a backslash preceding the user name, such as BOSTON\GILBERT. Be sure that the user who is a member of the Administrators group or Pervasive_Admin group is a local user.

If you have checked permissions and your user login does in fact meet one of the criteria above, then you should also check to make sure that you are logged into the correct network. For example, if you generally use the NetWare client to access servers on your network, but you are attempting to create a database on a Windows NT server,

you must make sure that you are logged into the Microsoft network, not only the Novell network on your LAN. You can verify whether you are logged into the correct network by attempting to read or write to a server that you are certain uses the target operating system.

Unable to connect to the specified remote server. Verify that all of the communication components are loaded on the remote server and that there are available sessions and try again

You may receive this error when attempting to register a new remote server in PCC. There are several reasons you may receive this error:

1. You mis-typed the server name. The database client tried to connect to a server that does not exist.

   Solution: Double-check the name of the server, and make sure you can see it in your Network Neighborhood, spelled exactly how you entered it.

   If you know the server exists but you can't see it in your Network Neighborhood, make sure that you are logged into the correct network. For example, if you generally use the NetWare client to access servers on your network, but you are attempting to connect to a database on a Windows NT server, you must make sure that you are logged into the Microsoft network, not just the Novell network on your LAN. Ask your network administrator for help.

2. The server user count has expired. If you have been using a temporary license, you will get this message for connection attempts after the license has expired.

   Solution: Run the User Count Administrator to check the status of licenses installed on the server. To start the program, choose **Start | Programs | Pervasive | Pervasive.SQL 2000i | Utilities | User Count Administrator**. In the window that appears, you can see detailed status information on each license that has been applied to your server. If your license has expired, purchase a permanent license from your reseller or from Pervasive Software.

3. There are no available sessions on the server. If you have a heavy load of users on the server, or if you have configured the server with a small number of sessions, you may receive this error.

Solution: Run Monitor to check the usage of sessions available on the server. You must have administrative privileges on the server (or membership in the Pervasive_Admin group) in order to do so. To start the program, choose **Start | Programs | Pervasive | Pervasive.SQL 2000i | Utilities | Monitor**. In Monitor, select **Options | Connect** and connect to the server in question. Then choose **MicroKernel | Communications**. In the window that appears, find **Total Remote Sessions**. If the **Peak** value and the **Maximum** value are the same, then it is likely that you have run out of sessions.

You can increase the number of sessions available by using Configuration in PCC. You must have administrative permission on the server (or membership in the Pervasive_Admin group) in order to do so. Connect to the server in PCC, double-click the Configuration icon for the target server, then choose **Server | Access | Number of Sessions**. Set the value to a number greater than the current setting.

4 The remote database server is not running.

Solution: Make sure that the remote database engine is running, or ask your network administrator to do so.

5 The remote database server is not accepting client requests.

Solution: Use Configuration to make sure that the remote database engine is configured to accept remote requests. You must have administrative permission on the server (or membership in the Pervasive_Admin group) in order to do so. Connect to the remote server in PCC, double-click **Configuration** for the target server, then choose **Server | Access | Accept Remote Request**. Be sure the value is set to **On**.

An error was encountered while connecting to the server

The most likely cause of this error is using the wrong operating system user name or password in an attempt to connect to the server.

Other possible causes include:

- The operating system may be expecting the user to change his/her password on the first logon. The Create Database Wizard does not allow this, so the connection fails. On Windows NT, this situation occurs if, in the User Manager, you have selected the **User Must Change Password at Next Logon** checkbox.

- If the user is a member of another group with lesser permissions, the lesser permissions will override the greater permissions. A user always has the most restrictive permissions of any group to which the user belongs.

Solution: Double-check the spelling of the user name and the password. Make sure the user and password have been set up on the remote server operating system.

Inspect the user's account information on the server. Make sure the operating system is not expecting the user's password to be changed at the next logon. Make sure the user is not also a member of a group that has restricted permissions.

For Windows NT/2000, be sure that the user is set up as a local user on the system, not a network user. Network users have a domain name and a backslash preceding the user name, such as BOSTON\GILBERT. Be sure that the user who is a member of the Administrators group or Pervasive_Admin group is a local user.

When trying to create a stored procedure in PCC, I get this error: ODBC Error: SQLSTATE = S1000, Native error code = -4994. The record has a key field containing a duplicate value (Btrieve Error 5)

A stored procedure already exists with the same name as the one you are trying to create.

Solution: Change the name of the stored procedure. Sometimes when you create a procedure with a SQL statement, you may not get a message from PCC confirming that the statement was successful. Then, when you run the statement again, thinking that it was not created, you receive this error code. To find out whether or not the procedure was actually created, you can get a list of the defined stored procedures by running this statement: `select * from X$proc#`

Frequently Asked Questions

This section answers some of the questions that customers ask most frequently. A list of the questions is provided below:

Installation

Security

Documentation

User Counts

Networking

Difficulty Accessing Data

ODBC and DDFs

Upgrading from Btrieve 6.15

Upgrading and Migration

Miscellaneous

Installation

Will I lose my data files if I uninstall my existing version of the product, or install a new version?

When you uninstall Pervasive.SQL or install a new version of Pervasive.SQL, your data files and DDFs are never affected. Even when Pervasive System Analyzer archives Pervasive.SQL files, or even if you have your data files in the same directory as Pervasive.SQL files, your data files are not affected.

What type of client install should I do—typical, custom, or network?

If you are not sure, always select typical. This option performs a standardized installation, which makes it easier to troubleshoot if problems occur.

How can I be sure what service pack level of client I am running?

If you are using Pervasive.SQL 7, start DDF Ease by choosing **Start | Pervasive | Pervasive.SQL 7 | Utilities | DDF Ease**. From the DDF Ease Help menu, choose **About**, and inspect the Build Level. The table below shows corresponding builds and service packs.

Build Number	Service Pack
76	SP 1
104	SP 2
112	SP 3
13x	SP 4
152	SP 5

If you are using Pervasive.SQL 2000i, start Monitor or Maintenance, choose **Help | About** from the menu, and check the Build Level. The table below shows corresponding builds and service packs.

Build Number	Service Pack
146	None
154	SP 1

Build Number	Service Pack
198	SP 2
230	SP 3

Is Pervasive.SQL 2000i supported on a Terminal Server?

Service Pack 3 provides full support for installation of the server engine on Terminal Servers. With Service Packs 2 and earlier, you can only install and run the client software on a Terminal Server.

Can I install Pervasive.SQL in a Failover environment? or

Can I install Pervasive.SQL in a Clustering environment?

Pervasive.SQL 2000i SP3 or later can be installed into a Windows 2000 Advanced Server Cluster or into a Novell NetWare 5.1 Cluster. Earlier versions of Pervasive.SQL are not supported in a Failover or Clustered environment. Linux and Solaris clusters are not supported at this time.

Can I install Pervasive.SQL in a Load Balancing environment?

That is not supported at this time.

Can I install Pervasive.SQL on a server running Btrieve v6.x or earlier?

No, you cannot run Pervasive.SQL and Btrieve 6.x on the same computer at the same time.

I installed Pervasive.SQL 2000 on my Netware 5.x server and it still says I am running the older version. What's wrong?

With Netware 5.x, you have to down the server and reboot for the new version of the database engine to be loaded into memory.

How do I keep my Workgroup Engine from starting up automatically when I reboot?

You must remove it from the Startup group under Start | Programs.

On Windows9X, the contents of this group are located at: c:\windows\start menu\programs\startup.

On Windows NT, the contents of this group are located at: c:\winnt\profiles*user*\start menu\programs\startup

On Windows 2000, the contents of this group are located at: c:\Documents and Settings*user*\start menu\programs\startup

On Windows NT/2000, *user* can be replaced by "All Users" or any user, as appropriate.

Security

Your security model is confusing. When do I login using an operating system user and password, and when do I login using a database user and password?

It may seem confusing at first, but in fact there is only one rule: use a database login only after you have already successfully connected to the server and are attempting to access a database directly. Up until this point, you should use an operating system login.

For example, if you run Monitor or Configuration to work with a remote server engine, you are prompted for a password. In both cases, you must supply a user name and password for an operating system account that has administrative permissions on the remote system, or an account that is a member of Pervasive_Admin. This applies also when you are creating a new database.

Once you start to work with the data itself, then you must supply a database user and password, if prompted. If database security is turned off, then you would never need a database user name or password. In this case, you would only need an operating system user and password to perform administrative tasks, as noted in the preceding paragraph.

Documentation

Why does my computer have a different format of online documentation than my co-worker's computer? We both installed the same software.

Starting with Service Pack 3, the type of online documentation that is installed depends on your system. Computers that support Microsoft HTML Help, also called Windows 98 Help, automatically receive that format of online documentation during the Pervasive.SQL installation. Computers that cannot support

Microsoft HTML Help automatically receive Windows 95 Help files. The actual content within each format is identical.

User Counts

How do I apply a User Count Upgrade?

For any engine running on Windows: From the **Start** menu of the computer where the server is installed, choose **Programs | Pervasive | Pervasive.SQL 2000i | Utilities | User Count Administrator**. Insert the license key diskette if you have one. Choose the path to the key file, ucmgr.key, or type in the key in the space provided.

For any engine running on NetWare: At the NetWare server console, insert the license key diskette if you have one. Change directories so that your current directory is where Pervasive.SQL is installed. Enter the following command:

`LOAD nwucutil -Da:` if you have a license key diskette in the drive

`LOAD nwucutil -K`*license_key* if you received your license key without a diskette (by email or fax, for example).

For any engine running on UNIX: Login as root and stop the transactional and relational database engines. Enter the command `su -psql` to become the owner of the Pervasive.SQL account. Run the appropriate command:

If you are adding a license key from a file:

`./ucutil -D`*path_to_dir_with_ucmgr.key_file*

If you are manually entering a license key:

`./ucutil -K`*license_key*

How does the Workgroup engine keep track of how many people are accessing the data? If people access the data with two engines at the same time, what happens?

The Workgroup engine keeps track of users just like the server engine does (that is, each combination of a set of requesters and an application creates a unique client ID, while licenses are tracked per computer based on the unique ID for the network card). Only one engine is ever permitted to access the files at a time. The second engine to try to open the files gets locked out, because the engines open the data files in exclusive mode (non-file sharing) so that corruption cannot occur.

Does the Workgroup engine use concurrent or per-seat licensing?

Concurrent—generally based on the unique ID embedded in the network interface card (NIC) of each workstation.

Aside from licensing, is there a limit to how many users can access a Workgroup engine simultaneously?

The Workgroup engine architecture is basically the same as the server engine. There are no theoretical limits, only practical ones such as bandwidth, system processing power, and economics. Client/server licenses are less expensive than Workgroup once you get to about 7-10 users.

Networking

How do I know which protocol I am using for communication? I can see other systems in Network Neighborhood but I can't get to my data.

From the **Start** menu of any Windows computer with the database client installed, choose **Programs | Pervasive | Pervasive.SQL 2000i | Utilities | Pervasive System Analyzer**. In the Welcome screen that appears, click **Next**. In the following screen, check the box **Test Network Communications** and make sure all the other boxes are not checked. Click **Next**. In the following screen, cancel the selected protocols that you do not want to test. Click **Browse** to select the drive that you have mapped to the installation directory (C:\PVSW by default) on the server. You must have a mapped drive; UNC names are not supported. Click **Next** to run the network tests. The results window tells you if there are any significant problems with your networking.

Difficulty Accessing Data

I upgraded from Btrieve v6.x or earlier to Pervasive.SQL 2000. Now I get error messages telling me that a file is unaccessible when everybody else can get to it. What's wrong?

Use Pervasive System Analyzer to be sure that all components from previous versions of Btrieve or Pervasive.SQL have been archived. Then, make sure your configuration settings are correct. Find the file pvsw.log and check for error messages indicating a status code 8505 or 8517. These status codes indicate that attempts were made to use a local Workstation engine to read the data files. From the Start menu, choose **Programs | Pervasive | Pervasive Control Center**.

Double-click **Pervasive.SQL 2000i Engines**. Double-click the icon representing your computer. Double-click **Configuration**. Double-click **Client**. Click **Access**. In the right-hand pane, make sure the following settings are set: Use **Local MicroKernel Engine** is "Off,"and **Use Remote MicroKernel Engine** is "On."

I have files sitting on the server that are shared and yet Pervasive.SQL cannot read them. What's wrong?

How are the files shared? Pervasive does not support mapping a drive letter using the Map Root under Novell, or using Redirected mapping under Microsoft, or using the hidden Admin share (C$) under Windows NT/2000.

Make sure that users have appropriate operating system login credentials to access the file server.

Run Pervasive System Analyzer and choose the Network Communications Test to be sure that you have proper connectivity.

I am using SQL queries to create a definition for an old table. The resulting record size is off. Why?

Starting with Pervasive.SQL 2000, fields that allow null values have an additional byte defined at the start of the field. This byte is the null indicator byte. You can work around this in one of two ways:

If you are using SQL statements to create a new table definition, enter the statement SET TRUENULLCREATE=OFF. For the remainder of your current session, any tables that you create will use the old record structure without the extra byte for each nullable column.

If you do not wish to use SQL statements, you can get the field sizes to align properly by creating all columns as not nullable.

I want to convert my data file version from 7 back to file format version 6 or 5. How do I do this?

If the files you wish to convert are serviced by a remote Server or Workgroup engine, you must have Administrator permissions on the remote system in order to perform these tasks. You must also have a network drive mapped to the remote data files.

Using Pervasive Control Center, double-click the icon that represents the server where the data files are located. Double-click **Configuration** for that server. Double-click **Server** then click

Compatibility. Click on **Create File Version** and set the value to the file version to which you want to convert. From the menu, choose **Edit | Apply**. Restart the database engine. These changes result in new files created to be in the version selected.

From the **Start** menu, choose **Programs | Pervasive | Pervasive.SQL 2000i | Utilities | Maintenance** to start Btrieve Maintenance Utility. Within this program, choose **Options | File Information Editor**. Click **Load Information** and choose the data file that you want to convert. Click **Create** and specify the name of the new, empty data file you want to create with the older version format. Click **OK** to create the file. Close the **File Information Editor** window, but do not exit Btrieve Maintenance Utility.

From the menu, select **Data | Copy**. Enter the name of the source data file and the name of the target data file (your newly created file with the older version file format). Click **Execute** to copy the records into the older version file. After the copying has finished, if you need the new data file to have the same name as it did previously, save your original data file with a different name, then save your new file using the original file name.

ODBC and DDFs

How can I tell if I am using ODBC to access my data files?

There are several ways to find out: first, look for .DDF files where the data files are located. If you see them, then most likely you can access the database using ODBC. Because it is possible to have DDF files located in a different directory, you should also use PCC to determine whether a database has been created for the data files you want to access. Finally, you can ask your application vendor whether their application uses ODBC to access the data files.

How can a hard-coded filepath in a DDF be changed?

Using PCC, right click on the table and choose **Tasks | Edit table design**. Click on the **Statistics** tab. Locate the parameter **Table Location** and change the value to the file path you wish to use. From the menu, choose **File | Save**.

It may appear that the path has not changed. To confirm the change, open the X$File system table and look at the Xf$Loc field for the given user table. If you cannot see the system tables in PCC, click on the **View** menu and choose **Show system tables**.

You can also use the ALTER TABLE USING statement in SQL to change the data file used by a particular table. Refer to *SQL Engine Reference* for further information.

I have DDFs from Scalable SQL 3.01. Are they compatible with Pervasive.SQL 2000i?

DDFs from Scalable SQL 3.01 are not compatible with Pervasive.SQL 2000i. You can use the CNVDDF utility provided on the Pervasive web site to convert the older DDFs to the new format. CNVDDF is a DOS utility that enables you to convert a database dictionary from Scalable SQL 3.01 to Pervasive.SQL 2000i format. To run the utility, make sure that either the Btrieve DOS Box or another DOS requester is loaded on the client workstation.

The utility is located at:

http://www.pervasive.com/support/updates/toolbox.tml

Note Converting database dictionary files in the CNVDDF utility does not modify any non-system table Btrieve files; it only modifies FILE.DDF, FIELD.DDF, and INDEX.DDF. It is recommended that you back up the original files to another directory before making any changes to the dictionary files using CNVDDF. This will enable you to restore your original DDFs.

What is the best way to ensure that my data dictionaries (DDFs) are safe?

Always keep a backup copy of your DDFs. Anytime you make changes to the runtime DDFs, be sure to make a backup copy of the DDFs before making changes. If you are turning on database security for the first time, you should make a backup copy of the dictionaries without security, and a backup copy with security.

How can I tell whether I have non-standard DDFs?

If you can edit your DDFs with a Btrieve utility, it means that you do not have standard dictionary files. A standard dictionary file does not permit direct Btrieve access. This lock out is a safety feature that ensures only the SRDE can write to the dictionary. DDFs are very special files that must remain synchronized with each other and with the data files at all times.

Standard dictionaries do not have case sensitive table names or field names. That is, the column definitions for column Xf$Name in file.ddf and column Xe$Name in field.ddf have the Case flag set, meaning the values are case insensitive.

DDFs are Btrieve files and thus can be opened and viewed (not updated) using the Function Executor. This is one way to confirm the contents of file.ddf or field.ddf.

On some non-standard dictionaries, the DDFs file.ddf, field.ddf, and/or index.ddf do not exist. Such dictionaries do not work with our products. For example, if you see a file called x$file.ddf, instead of file.ddf, you can assume your DDFs are non-standard.

Non-standard DDFs are unlikely to work properly with Pervasive Control Center or the relational engine.

Can I mix and match DDFs from different databases?

A complete set of DDF files must be considered a unit. No DDF file from one database may be intermixed with DDFs from a different database.

What happened to DDF Builder and DDF Sniffer?

DDF Sniffer and DDF Builder were added to the Pervasive product line with the acquisition of Smithware in 1998. Neither DDF Sniffer nor DDF Builder are available anymore. They were replaced by DDF Ease, which was part of the Pervasive.SQL 7 database engine. DDF Ease has since been replaced by Pervasive Control Center in Pervasive.SQL 2000i.

DDF Sniffer and DDF Builder used the Btrieve API to manipulate DDFs, which was less desirable than doing it through the native relational interface of Pervasive.SQL. These products contributed to issues that were difficult to isolate and fix.

Does Pervasive Control Center have the same functions as DDF Sniffer? Namely, can PCC read a Btrieve data file without DDFs and analyze the file to help me build DDFs?

Yes, it does in general, but it lacks some of the automation features offered by DDF Sniffer, such as the ability to recommend data types to use for non-indexed fields. However, PCC avoids many of the problems that DDF Sniffer and DDF Builder encountered in trying

to create and maintain DDFs via the Btrieve API rather than through the relational engine. The result is that you probably need to know a bit more about databases and data types than you would with DDF Sniffer, but you will wind up with fewer problems in the long run.

For more information, see the question and answer immediately above.

How do I set up ODBC on a NetWare server so that I can perform relational operations?

Pervasive.SQL includes an ODBC manager library for NetWare. All you need to do is use PCC to create a new database on the NetWare server, then create a client DSN on the client to make the remote database available to client applications.

You must have administrative permissions on the NetWare server to perform these tasks. Here are the steps in more detail:

1. Load the relational module on the NetWare server by issuing the command `mgrstart`.
2. At a Windows computer with Pervasive.SQL client installed, start PCC by choosing **Start | Programs | Pervasive | Pervasive Control Center**. Double-click **Pervasive.SQL 2000i Engines**. If you do not see the NetWare server name listed, then right-click and choose **Register New Engine**. In the window that appears, type in or browse to the NetWare server where the database engine is located. Click **OK**.
3. After the server icon is displayed in PCC, double-click on the icon that represents the server. Then double-click on the **Databases** folder. If you do not see the database you want to connect to, right-click on the **Databases** folder and choose **New Database**.
4. In the screen that appears, choose **Engine** interface and enter a NetWare user name and password with administrative permissions on the NetWare server. Click **Next**.
5. In the following screen, enter the name of the database and the directory where the data files are or will be located. This directory cannot be a mapped drive or relative to the client. It must be a full path name on the server, as if you were seated at the server console.

If you want to make existing DDFs and data files available for remote access, check **Use Advanced Settings**. If you are creating a brand new database from scratch, do not check this box.

6 Follow the prompts provided. For detailed procedures and options, see Chapter 2 of this book and/or see *Advanced Operations Guide*.

After you have created the Engine DSN on the server, you can access the database using PCC. If you want to access the database using ODBC-based applications, then you need to use ODBC Administrator to create a local client DSN on your workstation. You can do so by choosing **Start | Programs | Pervasive | Pervasive.SQL 2000i | Utilities | ODBC Administrator**. In ODBC Administrator, click the **System** tab, then click **Add**. In the window that appears, select **Pervasive ODBC Client Interface** and click **Finish**. In the following window, type in a name for the local DSN, and enter the server name. Click **Get DSN List**. In the box labeled **Data Source Name**, choose the DSN that you created on the NetWare server. Click **OK**.

You can now access the database on the NetWare server by connecting to the local client DSN you just created.

I have two similar Btrieve files, and I created a DDF for the first one. Since they are similar, can I use the same DDF on the second Btrieve file?

The answer depends on how similar the files are. If the two files differ only in the number of records, you can use the same DDF file. If there are any differences at all in the number, order, names, or types of fields or indexes, you cannot use the same DDF. In other words, you can only use the same DDF if the record structure of the two files is identical.

I have owner names set on my Btrieve files. After I created a DSN, I cannot open the files using ODBC. What's wrong?

If Btrieve files have owner names on them, you must use database security for ODBC access. This behavior changed in SP2 and later.

Turn on database security in PCC by following these steps: right-click on the database name, choose **Properties**, and click on the

Security tab. In the screen that appears, enter and confirm the master user password.

Caution Do not forget the Master user password. You cannot turn off security or perform administrative tasks within the database without it. You may want to make a backup copy of your DDFs before turning security on, in case you forget the password.

Next, you must grant the Master user access to the data files that have owner names defined. You can grant the access by issuing this SQL statement for each table that has an owner name:

```
GRANT ALL ON my_table 'ownername' TO Master
```

When you enter the statement, substitute the actual name of your table and the appropriate owner name for that table, as indicated above. Remember that each data file corresponds to an ODBC table. If you don't know which table corresponds to which data file, use PCC to find out: right-click on the table in PCC, and choose **Tasks | Edit Table Design**. In the Table Designer, click on the **Statistics** tab. The **Table Location** field shows you the file that is referenced by that table definition.

If security is important, then you must create users and assign permissions for all users expected to access the database. You do this by using CREATE USER, CREATE GROUP, and GRANT statements in SQL. You can also use the **Users and Groups** feature of PCC.

If security is not important to you, you can avoid creating many users and assigning privileges by granting access to PUBLIC, which means anyone on your network can access the data. You can use this statement:

```
GRANT ALL ON my_table 'ownername' TO PUBLIC
```

Is there a client side requester for the SRDE?

There is no DOS requester support for SQL applications, but the Pervasive.SQL client software for Windows includes ODBC client components allowing you to connect to a remote SRDE server engine.

Is ODBC the only method of access for Pervasive.SQL?

Definitely not! In addition to ODBC and the time-tested Btrieve API, you can also develop applications using our OLE DB provider, our JDBC driver, our pure Java interface, or our ActiveX controls.

Is there a single database file housing all the data, data definitions, stored procedures, security, table relationships, and so on as in some other products?

No. Pervasive.SQL stores data in separate files, one file per relational table definition. The meta data, such as data definitions, user/group definitions, and so on, are stored in a set of DDF files, where each file ends in the extension ".ddf."

Does the SQL engine (SRDE) have scheduler capabilities to run stored procedures or other types of scripts designed to access and affect data?

The SRDE does not have a scheduler.

Upgrading from Btrieve 6.15

After I upgrade the database engine on NetWare, is the SQL engine (SRDE) going to interpret Btrieve calls, or is it necessary to continue running the Btrieve.nlm?

Pervasive.SQL includes a new version of Btrieve.nlm. It also includes a new module named nwmkde.nlm. In the new architecture, Btrieve.nlm is a stub that calls nwmkde.nlm to perform operations. You must have both of these modules running.

On NetWare, will the 6.15 Btrieve NLM interface with the new MicroKernel engine, or will the 7.x version of Btrieve NLM have to be loaded to continue accessing the current Btrieve files?

You cannot use the 6.15 version of Btrieve.NLM with Pervasive.SQL. You must load all NLMs from the same product version.

My current files are in a 5.x format. Will they have to be converted to be accessed by the SQL engine (SRDE)?

The SRDE accesses 5.x format files through the MicroKernel. No file conversion is needed. You must convert the files to 7.x format to take advantage of new features offered by the 7.x MicroKernel. However,

the SRDE requires DDFs for your data files. If you do not have DDFs, in some cases you may be required to modify your file indexes before you can create valid DDFs. The steps required to create DDFs for Btrieve files are explained in *Advanced Operations Guide*, Chapter 12.

I have DDF files today that were used by a product called Xtrieve. They are in a 4.x file format. Can they be used by the SQL engine or will they have to be converted?

They must be converted to the Pervasive.SQL DDF format using the tool DDFCNV. For more information about this tool, see "I have DDFs from Scalable SQL 3.01. Are they compatible with Pervasive.SQL 2000i?" on page 5-22.

Is there a tool that replaces Xtrieve?

There is no direct replacement, but you should consider using Crystal Reports for Btrieve as an excellent upgrade from Xtrieve for reporting on and querying Btrieve data.

I expect to continue using my old applications and files and phase in new applications to access the same files through the SQL engine (SRDE). Is this a false expectation?

Generally speaking, no. Pervasive.SQL is designed to allow concurrent access between SRDE applications and Btrieve applications. If you are currently using Btrieve without ODBC, you may need to make some changes to the indexes on your files before you can create DDFs that provide SRDE access to the data. The steps required to create DDFs for Btrieve files are explained in *Advanced Operations Guide*, Chapter 12.

Upgrading and Migration

How can I migrate a Btrieve database from NetWare to Windows NT or vice versa?

Shut down the Btrieve engine on the source computer and copy all the database files from the source computer to the target computer.

Install Pervasive.SQL 2000i Server engine on the target computer.

Create a new database using existing DDFs and data files.

Share the server resource as appropriate.

Win32 Clients: No change

DOS Clients: If moving to Windows NT, replace BREQUEST with BREQNT or BREQTCP. If moving to NetWare, replace BREQNT or BREQTCP with BREQUEST.

When I create a table using an existing Btrieve file, the wizard displays fewer columns than there are in the Btrieve file. What's wrong?

Btrieve files contain a limited amount of information about the structure of the file. The table creation wizard can figure out some field definitions using the indexes, but after the indexes are exhausted, data segments may remain that contain more than one actual field. The wizard has no way of interpreting the contents. You must use your detailed knowledge of the record structure to split out these fields and build a table definition that matches all the fields in the record.

The procedure for this task is provided in *Advanced Operations Guide*.

Where can I find information on migration from earlier product versions to Pervasive.SQL 2000i? Where can I find migration and compatibility information?

Getting Started with Pervasive.SQL contains an entire chapter that provides detailed instructions on how to upgrade.

If your application uses Scalable SQL or ODBC, then you should review the *Application Migration Guide* available on the web site:

http://www.pervasive.com/developerzone/techlibrary/guides/mguide/index.htm

Miscellaneous

What does the "i" mean in Pervasive.SQL 2000i?

The 'i" stands for Internet-enabled. Pervasive.SQL offers the reliability and features needed for almost any data intensive web application. From OLE DB, JDBC, and pure Java interfaces to full security and file encryption on disk, Pervasive.SQL has what it takes to help your organization succeed in a wired world.

I dumped Btrieve records to a file and now I can't read the file. What happened?

If you use the Btrieve Maintenance Utility to save/dump the records, the resulting file contains the binary image of each record. Unless the record consists entirely of character data, it may not be readable to the human eye.

The only way that Pervasive.SQL can dump a record in ASCII readable format, is by reading the DDFs to get a description of the total contents of the record. Btrieve only has the record length, the data type of indexes and length of the indexes. Btrieve does not have information on how to interpret the entire contents of the record.

Does Pervasive.SQL take advantage of multiple processors?

Pervasive.SQL does not currently contain any code to optimize its behavior on multi-CPU hardware. Any performance benefits obtained in this type of environment are delivered by the operating system.

How do I run Pervasive.SQL in debug mode?

Server

You must have Pervasive.SQL 2000 SP2 or later installed. You must have administrator privileges on the machine where the engine is located that you want to run in debug mode. Using PCC, double-click **Configuration** for that engine. Choose **Server | Debugging | Trace Operation** and set the value to **On**. Click **Edit | Apply**.

You do not need to restart the engine.

Note After tracing operations, you should turn off Trace Operation, following the same procedure as outlined above, making sure to click Edit | Apply when finished. You may notice slower performance if you run Pervasive.SQL in debugging mode.

Win32 Client

Obtain the debug version of the client DLLs from Pervasive. Back up the original DLLs with the same names so you can restore them later. Install the debug DLLs on your client system. For example, to debug

32-bit applications you may need to install W3MIFxxx.DLL or W3NSLxxx.DLL, where xxx is a specific build number matching the non-debug version you originally had installed.

Run your Btrieve application on the client machine once to create the registry entries under

HKEY_CURRENT_USER\\Software\\Pervasive Software\\Communications Requester

Modify the registry entry **Trace Level**, setting its value to "99". Enter your desired trace file name and location, and set the value of **Append File** to **Yes**, if desired.

Btrieve Win16 Client

For the 16-bit client, follow the same steps as above, but instead of altering the registry, find the settings in BTI.INI:

```
[Diagnostics]
Trace Level=none  set this to 99
Trace Modules=all
Trace File=./COMM.TRC
Trace Buffer Length=3
```

The DLL names are W1MIFxxx.DLL or W1NSLxxx.DLL.

Does garbage collection occur in the data files and indexes? For example, is space from deleted records recovered or reused?

Yes, space from deleted records is re-used on subsequent inserts. Space in files is never de-allocated back to disk. If index balancing is turned on, then unused space in index pages is also re-used.

Is database shadowing available, allowing a complete up-to-data second copy of the database to exist on another drive or machine?

Pervasive.SQL does not contain specific functionality for this, but many customers have successfully used hardware mirrored drive arrays and solutions like Vinca's (now acquired by Legato) Standby Server to provide this functionality. Pervasive.SQL 2000i SP3 and later supports Pervasive's data replication product. Replication can be used to synchronize all or part of 2 or more databases.

What is the mechanism that allows the database to be backed up online? What happens if the server goes down in the middle of a backup with many open transactions?

Continuous Operations allows you to put a set of data files in a special "safe mode" so that they can be safely backed up while in use. While data files are in Continuous Operations mode, they are not modified, and special delta files store the results of any database operations. After the backup is complete, the data files must be removed from Continuous Operations mode, at which time the changes stored in the delta files are rolled into the live files.

If the server goes down while files are in continuous operations mode, the next time the data file is accessed, the database engine detects the existing delta file and rolls in the changes at that time.

You can put data files into Continuous Operations mode by using the BUTIL -STARTBU command or Maintenance utility described in *Advanced Operations Guide*.

chapter

6

Pervasive.SQL Resources and Contacts

A Guide to Pervasive.SQL Customer Information Resources

Pervasive Software strives to ensure that your experience with Pervasive.SQL is successful. This chapter describes the resources and information available to you as a valued customer of Pervasive Software.

The following variety of resources can help you get answers to your questions, troubleshoot problems, and interact with the Pervasive team as well as with other customers:

Printed Documentation

Pervasive.SQL 2000i SP3 comes with a printed copy of *Getting Started* and *Status Codes Quick Reference*. A complete suite of online documentation is installed on Windows when you choose the **Typical** installation procedure. It is also available as an option in the **Custom** installation procedure. The content is accessible through the **Start** menu:
Programs | Pervasive | Pervasive.SQL 2000i | Documentation | Pervasive.SQL 2000i Documentation.

Printed versions of the following titles are available for purchase:

- *Getting Started with Pervasive.SQL* (Server or Workstation/ Workgroup edition)
- *Pervasive.SQL User's Guide*
- *SQL Engine Reference*
- *Status Codes and Messages*
- *Pervasive Products and Services*
- *Advanced Operations Guide*

To order manuals, please contact Pervasive Software using one of the following methods:

Online: http://www.pervasive.com/products/manuals/

E-mail: salessupport@pervasive.com

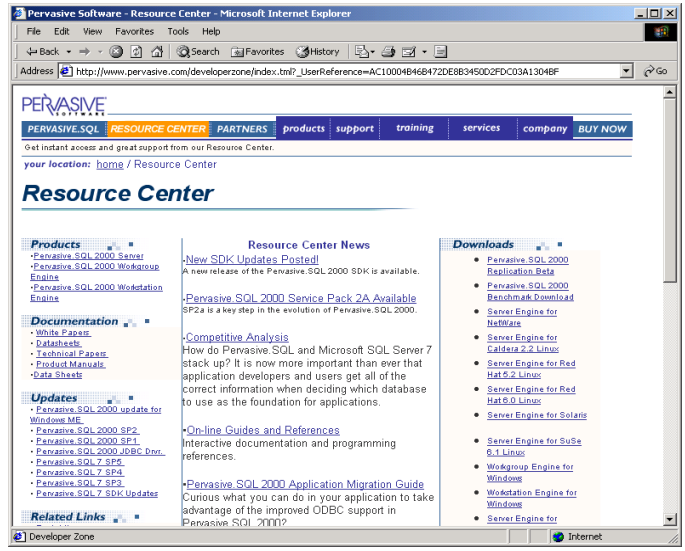
Phone: +1-800-287-4383

Developer Zone

The Pervasive Software Web site is a great source for Pervasive.SQL information: http://www.pervasive.com. It is your most immediate source for assistance with the product.

The link shown below is commonly referred to as Developer Center. It is a great starting point from which to navigate to available downloads, documentation, product updates, news articles, sample code and tutorials. Developer Center also provides access to an expansive technical library and training information.

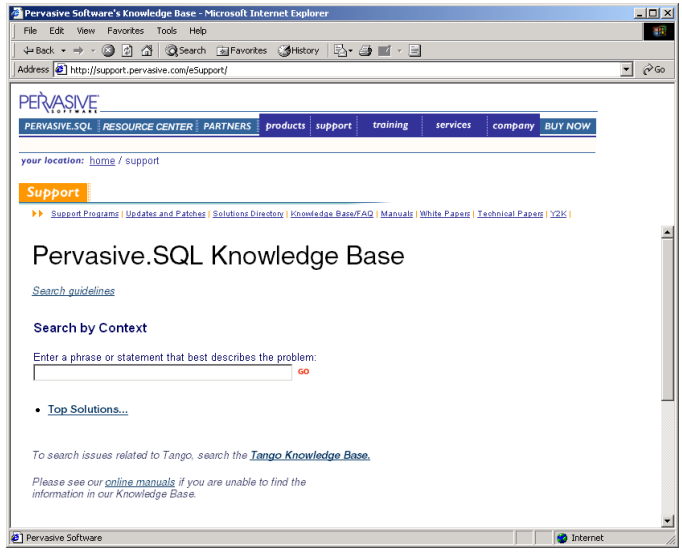
http://www.pervasive.com/developerzone/



Pervasive.SQL Knowledge Base

The Pervasive.SQL Knowledge Base Online is a searchable database for technical information regarding installation, configuration, component management, product defect status, and answers to the frequently asked questions (FAQs). The Knowledge Base, shown below, uses an associative problem-solving technology to perform contextual searches and can be used to quickly find specific answers to your questions about Pervasive products.

http://support.pervasive.com/eSupport/



FTP Site

Pervasive Software strives to maintain close ties to developers using Pervasive.SQL for their database applications. On the Pervasive FTP site, you can find practical resources such as downloadable updates and patches to our product offerings as well as additional debugging tools, documentation, third-party tools, and beta releases.

ftp://ftp.pervasive.com/support/

Online Documentation

The latest versions of Pervasive.SQL product manuals in CHM, HLP and PDF format are available for download from the Pervasive Software web site. These titles include:

- *Getting Started with Pervasive.SQL* (Server or Workstation/ Workgroup edition)
- *Pervasive.SQL User's Guide*
- *SQL Engine Reference*
- *Status Codes and Messages*
- *Pervasive Products and Services*
- *Advanced Operations Guide*

http://www.pervasive.com/support/technical/product/

DevWire

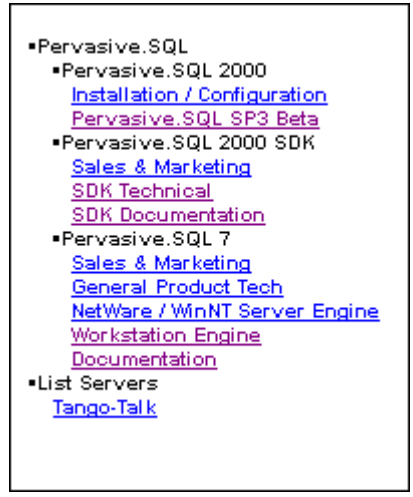
DevWire is a monthly newsletter on all things Pervasive: learn about Beta cycles and releases, Service Pack releases, current topics, FYIs, FAQs, Pervasive Software Events in your area, trade shows where you can find us, and much more!

To subscribe to DevWire, send an e-mail message with "**add**" as the subject line to devwire@pervasive.com.

DevTalk

Pervasive Software's DevTalk discussion forums are a great way to share ideas with other customers, get technical questions answered, and give feedback directly to Pervasive Software. Check out the figure below to see some of the topics currently being discussed and watch the web site for new topics as they appear!

http://www.pervasive.com/devtalk/



Newsgroup

Many Pervasive.SQL customers enjoy participation in a newsgroup—a learning environment in which users help users, with some participation by Pervasive Software. The newsgroup is managed by the end-user community, posting and answering questions as they wish.

Pervasive Software is represented in the worldwide network of news discussion groups at:

news://comp.databases.btrieve.

E-Mail

Pervasive Software welcomes your comments, suggestions and requests for assistance via e-mail. Please submit to the following contacts:

- docs@pervasive.com

  For comments or concerns regarding the content of Pervasive.SQL documentation. Requests for manuals should be directed to your sales rep or "salessupport@pervasive.com".

- uteam@pervasive.com

  For comments or concerns regarding the Pervasive.SQL utilities, such as PCC, PSA, installation, and other Pervasive tools.

- techsupport@pervasive.com

  For technical support for Pervasive.SQL. For a faster response, we recommend submitting your request via the e-mail support form located at:
  http://www.pervasive.com/support/Email_Support.taf

- salessupport@pervasive.com

  For information about Pervasive.SQL sales matters such as contacts, pricing, and product specifications. You may also submit the sales information form located at:
  http://www.pervasive.com/contact/email_sales.tml

- developer@pervasive.com

  For developer relations. A great way for developers to communicate their ideas about all Pervasive products, interfaces and programs.

- beta@pervasive.com

  For questions regarding beta products and general beta information, including Pervasive's Beta Program.

- info@pervasive.com

  For general information about the company, marketing efforts, public relations, and other general questions.

- investor.relations@pervasive.com

  For questions from investors.

Technical Support

Pervasive Software Technical Support Call Centers

- Support Center (Headquarters-Austin)

  Hours: 7 A.M. to 7 P.M. Central Standard Time.

  Phone: +1 512 231 6000

  Toll Free: +1 800 287 4383

- European Service and Support Center (Belgium)

  Hours: 9 A.M. to 6 P.M. Central Europe Time.

  Phone: +32 2 710 1660

For detailed contact information for Pervasive Software offices worldwide, refer to *Pervasive Products and Services*.

To receive a faster response to your technical support questions, we recommend using the electronic support method located at: http://www.pervasive.com/support/Email_Support.taf

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z