Pervasive.SQL 2000i

Pervasive Products and Services

Overview of Pervasive.SQL 2000i Products and Support

Contents

Figures

Tables

About This Manual

This manual contains information about Pervasive Software product concepts, architecture, and terminology, as well as more general information about database models. It also includes details on Pervasive customer services, product registration, and licensing information.

The Pervasive.SQL 2000i product includes a relational database management system (formerly called Scalable SQL), and a transactional database management system (formerly called Btrieve). These products may be used separately or simultaneously, in applications ranging from single workstation configuration to complete client/server access.

Pervasive also offers the Pervasive.SQL Software Developer Kit (SDK). The SDK is a comprehensive collection of rapid application development resources, including the I*Net Data Server, a direct ActiveX interface, a pure Java Interface, and support for all major Windows development environments. Environments supported include Microsoft's Visual Studio (Visual Basic and Visual C++), Inprise (Delphi, C++ Builder, and JBuilder), and Symantec Visual Cafe. The SDK also supports C, C++, Java, and COBOL programming languages.

Who Should Read This Manual

This manual provides information for users who are new to Pervasive Software products or who are new to databases and database terminology. This manual is also useful for anyone who is new to Pervasive Software application development.

Pervasive Software would appreciate your comments and suggestions about this manual. As a user of our documentation, you are in a unique position to provide ideas that can have a direct impact on future releases of this and other manuals. If you have comments or suggestions for the product documentation, post your request at http://webforum.pervasive.com or send e-mail to docs@pervasive.com.

Manual Organization

The following list briefly describes each chapter in the manual:

- Chapter 1 — "Pervasive Software Services"

    This chapter describes the different Pervasive Software services that are available. These include Pervasive Partner Programs and information concerning sales and technical support contracts. This chapter also provides contact information for Pervasive Software in Europe and Japan.

- Chapter 2 — "Registration and Licensing"

    This chapter describes Pervasive Software's product registration and licensing and provides information about product bundling and upgrades.

- Chapter 3 — "Database Models"

    This chapter describes different types of database models and explains how Pervasive Software products support these models.

- Chapter 4 — "Product Overviews"

    This chapter describes the different products offered by Pervasive Software.

- Chapter 5 — "Product Architecture"

    This chapter describes the basic architecture of Pervasive Software products, beginning with the core engines of Pervasive.SQL. This chapter also describes some sample configurations for Pervasive.SQL.

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 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*.

Pervasive Software Services

Contact Information for Pervasive Software, Sales, Ordering, and Support

Pervasive Software is pleased that you have chosen our information management products. This chapter introduces you to our products and provides you with contact information should you desire to reach us. For the newest information about Pervasive Software products, visit us on the Internet at our corporate Web site: http://www.pervasive.com.

This chapter includes the following sections:

- "Contacting Pervasive Software" on page 1-2
- "Sales and Ordering Information" on page 1-5
- "Pervasive Software Partner Programs" on page 1-6
- "Support Services" on page 1-9
- "Pervasive's Professional Services Group" on page 1-11

Contacting Pervasive Software

You can contact Pervasive Software in the following ways:

North America

Pervasive Headquarters	
Address	Pervasive Software Inc. 12365 Riata Trace Parkway Building II Austin, Texas 78727 USA
Phone	1-800-287-4383 1-512-231-6000
Fax	1-512-231-6010
E-mail	info@pervasive.com salessupport@pervasive.com developer@pervasive.com docs@pervasive.com
World Wide Web Pervasive Support Web Site	http://www.pervasive.com http://www.pervasive.com/support/ Email_Support.taf

Europe

Pervasive Software European Service and Support Center	
Office Serves	Europe
Address	Pervasive Software N.V. Bessenveldstraat 25a B-1831 Diegem Belgium
Phone	+32-2-710-1660 (Technical Support)
Fax	+32-2-718-0331
World Wide Web Pervasive Support Web Site	http://www.pervasive.com/support/ Email_Support.taf

Pervasive Software Southern Europe Sales Office	
Office Serves	France, Greece, Italy, Middle East, North Africa, Portugal, Spain, and Turkey
Address	Pervasive Software Southern Europe Immeuble Atria 21, Avenue Edouard Belin F-92 500 Rueil Malmaison, France
Phone	+33-1-55-47-17-00
Fax	+33-1-55-47-17-07
E-mail	dbellens@pervasive.com

Pervasive Software United Kingdom Sales Office	
Office Serves	United Kingdom and Ireland
Address	Pervasive Software Ltd 110A High Street Chesham Buckinghamshire UK HP5 1EB
Phone	+44-1494-791119
Fax	+44-1494-793929
E-mail	hgooder@pervasive.com

Pervasive Software Northern Europe Sales Office	
Office Serves	Baltic States, Belgium, Denmark, Finland, Iceland, Luxembourg, Netherlands, Norway, and Sweden
Address	Pervasive Software N.V. Airport Boulevard Office Park Bessenveldstraat 25A B-1831 Diegem Belgium
Phone	+32-2-718-0330
Fax	+32-2-718-0331
E-mail	gvcutsem@pervasive.com

Pervasive Software Central Europe Sales Office	
Office Serves	Austria, Eastern Europe, Germany, Switzerland, and South Africa
Address	Pervasive Software GmbH Frankfurter Strasse 151 D-63303 Dreieich Germany
Phone	+49-6103-96 22 00
Fax	+49-6103-96 22 05
E-mail	bmartin@pervasive.com

Japan

Pervasive Software Japan	
Address	Pervasive Software Co., Ltd. World Trade Center Building, 33F 2-4-1 Hamamatsu-cho Minato-ku, Tokyo 105-6133 Japan
Phone	+81-3-5405-2261
Fax	+81-3-5405-2269
E-mail	info@pervasive.co.jp

Sales and Ordering Information

Pervasive Software provides a sales team to assist you in evaluating offerings from Pervasive Software and complementary third-party applications to streamline your development efforts. Pervasive Software's sales representatives are knowledgeable, trained experts with technical backgrounds and an in-depth understanding of all Pervasive Software products.

You can order direct from Pervasive Software or through an authorized Pervasive Software Solution Integrator. For sales assistance or more information, call 800-287-4383, or 512-231-6000. You can also send a fax to 512-231-6010, send e-mail to salessupport@pervasive.com, or visit the Web site at http://www.pervasive.com.

Pervasive Software Partner Programs

In an effort to provide our partners with greater value, amidst the constant change of today's business world, Pervasive Software proudly announces our new partner offering, Profit with Pervasive.

Profit with Pervasive

Profit with Pervasive is Pervasive Software's strategic channel partner offering, designed to help you better develop, market, support and deploy your business applications and/or services that use Pervasive's software. Profit with Pervasive is based on your desired level of involvement, not just your revenue commitments or contributions. There are no minimal requirements to become a Profit with Pervasive partner, and no up-front, out-of-pocket costs. It puts the power directly into your hands, allowing for a truly flexible and progressive relationship.

Your Pervasive territory manager can help grow your business through co-marketing and tech support. "Profit with Pervasive" is designed to be simple, but more importantly, it's designed to provide real value to your business.

The flexibility of Profit with Pervasive allows partners with varying business models to participate in select Pervasive programs on an elective basis. Many benefits will be available to you through Profit with Pervasive. You'll also want to check out the additional Profit with Pervasive offerings as well. Again, your level of involvement is completely up to you.

Profit with Pervasive Benefits

- Marketing assistance to execute approved promotional campaigns
- Invitations to Webinars
- Free limited support incidents
- Enrollment in DevWire and TechWire
- Listing in Pervasive Software Solutions Directory
- Access to sales and marketing tools, white papers, and technical information
- Notification of Pervasive's public Beta programs
- Expand your opportunities

For complete requirements for each program, please contact your Pervasive Software representative. For information on Profit with Pervasive you may visit http://www.pervasive.com/pwp/pwp.tml, or send e-mail to partners@pervasive.com. You may also telephone us directly at 800-287-4383 ext. 2, or you can send a fax to a representative at 512-231-6199. We will respond with detailed information about the features, benefits, and requirements of joining Profit with Pervasive.

Types of Pervasive Partnerships

Pervasive offers the following partnership programs:

Profit with Pervasive

Pervasive Software's Profit With Pervasive provides Value Added Resellers (VARs), Independent Software Vendors (ISVs), and consultants with easy access to new products and information needed to service and solve their customer's complex needs. Profit with Pervasive supports partners with marketing assistance, free limited support sessions, invitations to webinars, and notification of beta programs. You can receive demo and testing software, and access to marketing literature, data sheets and white papers. In addition, a consistent pricing structure and a unique licensing scheme make building and deploying applications with Pervasive.SQL affordable. Partners have the opportunity to be featured on our widely consulted corporate Web site and listed in our Solutions Directory. Best of all, participation is absolutely free!

Pervasive OEM Partner

OEM Partners acquire the right to integrate, manufacture, and distribute Pervasive's embedded database solutions in their own high-volume software products. As an OEM Partner, you'll receive advertising assistance, worldwide seminars, training classes, public relations, promotions, and marketing literature. You'll also have a senior account manager who will provide invaluable one-on-one guidance, and schedule regular account planning sessions with opportunities to discuss product and marketing strategy, explore reseller and end-user issues, and identify new opportunities.

OEM Partners receive substantial volume discounts based on estimated volumes, with discounts deepening as sales exceed expectations. There is also an allowance for variances in seasonality.

Pervasive Distribution Partner

The Distribution Partner and Master Distribution Partner programs are designed to support large software distributors who provide Pervasive products to ISVs and VARs. Master Distributors have exclusive territories located in emerging international marketplaces. Distribution Partners and Master Distributors receive premier discounts–along with market development funds to help build and expand your customer base. A dedicated account manager makes sure you receive up-to-the-minute product information, forwards sales referrals, and helps you plan your strategic direction.

Support Services

Pervasive Software offers three levels of technical support to resolving your support questions quickly and thoroughly. Pervasive's support options include telephone, web-based, and account management support. Of course, these offerings are just one part of a comprehensive system designed to further your success.

Managed Support Options

With Pervasive's Managed Support option, you can take advantage of a one-on-one relationship with a technical support engineer. Your personal Technical Account Manager (TAM) becomes your voice at Pervasive. First, your TAM will visit your place of business, taking time to understand your unique organizational needs. Your TAM will then oversee, track, and personally manage all of your support activity. Pervasive offers two levels of managed support: one with free TAM access and one with an option TAM component.

Premium Managed Support

- Technical Account Manager (TAM)
- Free installation support
- Web-based support programs
- Unlimited incidents
- Direct access to Pervasive's top level support team
- Up to eight persons in your organization will have access to Pervasive support centers
- Access to information (betas, bug fixes, etc.) from the Pervasive list server
- Free Software Development Kit (SDK)
- Incidents will adhere to published service level guidelines
- Customized monthly reporting - your TAM will work with you to develop monthly reports designed specifically for your needs

Standard Managed Support

- Free installation support
- Web-based support programs
- Unlimited incidents
- Direct access to Pervasive's top level support team

- Up to five persons in your organization will have access to Pervasive support centers
- Access to information (betas, bug fixes, etc.) from the Pervasive list server
- Free Software Development Kit (SDK)
- Technical Account Manager (optional)

General Pervasive Support Options

IT Professional Support

The Pervasive IT Professional Support options are fee-based and designed for IT professionals that require additional assistance while using Pervasive products. Options are flexible and suited to those needing specific, periodic support.

Basic Support

The Basic Support package offers free installation support and unlimited access to Pervasive's Web-based support, where you'll find answers to frequently asked questions, technical tips, and maintenance releases. You'll also have access to our comprehensive Knowledge Base - 24 hours a day, 7 days a week.

Other Pervasive Services

Pervasive offers a range of additional services to assist you in the planning, development, deployment, and usage of your solution, including off-hour support (scheduled in advance and available 24x7), a Technical Account Manager, and field services support. Contact your Pervasive account manager for more information on additional services.

About Pervasive Software

Pervasive Software is committed to driving simplicity into the complex world of data management, a pledge that's evidenced by our powerful, yet easy-to-use software products. It's this dedication that has made Pervasive Software's years of database business so successful, and the reason developers worldwide continue to depend on our high-performance database engines to power their applications.

For additional program and pricing information, please call 1-800-287-4383 or visit http://www.pervasive.com/support.

Pervasive's Professional Services Group

Pervasive's Professional Services Group (PSG) provides very targeted services to Pervasive.SQL application developers that leverage the expertise of the manufacturer and enhances their development and deployment efforts.

PSG offers expert, customized professional services by utilizing our highly trained and qualified staff. PSG is dedicated to providing outstanding service, cost-effective solutions and fast time-to-market.

Contact our Professional Services Group today to discuss how we can assist your Pervasive.SQL application development needs. Some of our packaged offerings are listed below.

- Web Application Migration Services. Migrate your current application to the Web, or simply enable parts of your application that are best suited for the Web. Our middleware and web development expertise, combined with our database knowledge, ensures that your application migration will be a success.

- Replication Services. PSG will help tailor a customized replication solution for your application needs. We will maximize your database application through replication across networks, minimize transaction latency and balance workloads.

- Datatecture Review. Optimization of the Btrieve API and SQL call performance are the main objectives. PSG will provide a data definition strategy, database schema design and review, metadata modeling and performance testing.

- Kick Start Process. The process is designed to get you up and running quickly. PSG will provide project planning, data modeling, system architecture and business analysis, through to actual implementation. PSG's qualified professionals will provide the assistance you need to accelerate your application development lifecycle.

- Database ASP Model (DASPM). DASPM provides an external, secure and centralized interface for migrating your database to an ASP model. With faster access for replicated data, common middleware, external data management and synchronization with internal databases, your organization can expand past its walls by leveraging the Internet and DASPM.

■ Application Upgrade Services. Need assistance upgrading your application to the latest Pervasive.SQL release? In order for you to take advantage of the newest features, we will provide the experience, design specifications and programming needed to optimize the performance, reliability and scalability of your application.

For more information call PSG at 1-800-287-4383 ext. 6349, or visit our website at http://www.pervasive.com/psg.

Technical Support

Pervasive Software currently provides technical support from 7 A.M. to 7 P.M. central time, Monday through Friday. For technical support and information about Pervasive Software technical support programs, call 800- 287-4383. To submit a technical support request visit the Pervasive Support Web Site at http://www.pervasive.com/support/Email_Support.taf.

Knowledge Base

Pervasive's Knowledge Base (http://support.pervasive.com/kb/) provides customers access to technical information on installation, configuration, component management, product defects, and answers to the most frequently asked questions (FAQs). If you are a OEM Partner, you are granted access to Pervasive's secured information on the Web.

Pervasive Tech Talk

The *Pervasive Tech Talk* newsletter is published bi-weekly and distributed directly via e-mail to Pervasive's OEM Partners, international distributors, OPP VARs, support contract customers, and other strategic customers. It contains the latest information on beta cycles, current topics, FYIs, Q&As, product defects, and enhancement requests. The information found here can also be found as technical tips, FAQs, and technical papers on the Support section of Pervasive's Web site (http://www.pervasive.com/support).

Electronic Mail

Customers can submit requests for assistance via a form on Pervasive's Support Web site at http://www.pervasive.com/support/Email_Support.taf.

FTP

The FTP site (ftp://ftp.pervasive.com/support) contains downloadable updates and patches to our product offerings as well as additional debugging tools, documentation, third party tools, and beta releases.

Developer Relations

Pervasive Software strives to maintain close ties to developers using Pervasive.SQL for their database solutions. The Developer Zone Web site is where you can find what you need to make developing applications with Pervasive database technology faster, easier, and more powerful. Sample code, tutorials, access to our expansive technical library, and links to third-party tools and utilities are just a

few of the practical resources you'll find at http://
www.pervasive.com/developerzone/index.html

In addition, the Pervasive DevWire can keep you informed of the
latest updates regarding Pervasive.SQL development. The Pervasive
DevWire is a free developer-oriented e-mail news service. For more
information, visit http://www.pervasive.com/developerzone/
index.html

Newsgroup   Pervasive Software is represented in the newsgroup community at
comp.databases.btrieve. The newsgroup is managed by the end-user
community, posting and answering questions as they see fit.

Registration and Licensing

Detailed Product Information for Users and Developers

This chapter introduces Pervasive Software product information, including product registration, licensing, product bundling, and upgrades. We urge you to read through this chapter carefully to understand how to use your product to the fullest extent. Pervasive Software is always interested in improving its licensing, distribution policies, and product features. Forward any comments or suggestions to us at salessupport@pervasive.com.

The following sections detail Pervasive's registration and licensing information:

Product Registration

Understanding our customers' needs is of the utmost importance to us at Pervasive Software; therefore, we have made it as simple as possible to register your product and provide us with feedback. After installing your software, complete Pervasive Software's registration card to provide important information necessary to assist us in providing features and programs to meet your needs. We will use this information to notify our customers regularly of current product and update information. When you send in the registration card, either by mail or fax, you are entitled to 30 days of complimentary customer support.

Licensing

The Pervasive Software product family is designed to provide developers simplified pricing and licensing structures for both building and distributing new products. SDKs contain software licensed for building, testing, and demonstrating applications using Pervasive's MicroKernel Database Engine. When ready to distribute your client/server application, you need licensing for only the targeted number of users per server.

Workstation-based applications can be covered entirely through one of the industry's lowest cost-per-seat licensing plans, the Derivative Software License. The following table lists the different licensing plans available for Pervasive.SQL 2000i. To find out the latest licensing plans for all the different Pervasive Software releases, please contact sales at 800-231-4383 or e-mail them at salessupport@pervasive.com.

Table 2-1   Pervasive.SQL 2000i Product Licensing

License	Server	Workstation	Workgroup
SDK – Limited Use Software License	✔	✔	✔
Single Seat – Per Machine License	N/A	✔	N/A
Derivative Software License	OEM Contract Only[1]	✔	✔
Client/Server – Network and Intranet License	✔	N/A	N/A
Client/Server – Network License Amendment for Internet Server	✔	N/A	✔
Commercial Deployment	N/A	N/A	N/A

1 Pervasive Software OEM Partner Program permits distribution on an OEM basis.

Developing with Pervasive Software Products

Developers interested in bundling the Pervasive.SQL client/server or single-user product with their commercially sold applications should contact Pervasive Software's Sales Department at 800-287-4383 or 512-231-6000 or send e-mail to salessupport@pervasive.com.

Pervasive Software is committed to providing its developers with the highest quality support in the industry. To help us maintain this standard, we require that you provide the first line of support for the applications you deploy based on Pervasive.SQL technology. As the application developer, we expect you to support and assess the interactions between your application and Pervasive.SQL.

The Pervasive.SQL engines include all end-user utilities to run diagnostics and analyze your data files. These utilities may be distributed with your application at your discretion.

You are required to support any utilities you distribute.

Distributing Pervasive Documentation

If you would like to distribute Pervasive documentation with your product, you must first contact Pervasive Software's Product Marketing group at 800- 287-4383 for approval, which will depend on your licensing agreement. Documentation is available in hard copy and on-line formats. We currently provide online documentation in both PDF and HLP formats.

Upgrades

You can obtain upgrades directly from Pervasive Software by calling 800-287-4383 or 512-231-6000 or by sending e-mail to salessupport@pervasive.com. You can also obtain upgrades through Pervasive Software Channel Partners. Refer to Chapter 1, "Pervasive Software Services" for more information.

Database Models

Overview of the Primary Database Management Systems

This chapter introduces two major models for database management systems: relational and transactional. Both models describe a way to view information in a database, and each is based on a different method of modifying the data and extracting information.

Note The word "relational" refers to the SQL engine that is part of Pervasive.SQL 2000. The word "transactional" refers to the transactional engine in Pervasive.SQL 2000 and files that were formerly called "navigational."

This chapter includes the following sections:

- "Relational Model" on page 3-2
- "Transactional Model" on page 3-4

Relational Model

The relational database model represents data to the user in the form of tables. Each table is related to other tables in the database in some way. This relationship enables the end user to search the database for particular pieces of information that may not co-exist in the same table.

The relational database model is currently very popular. This is because of the level of abstraction that it provides from the actual data. End users do not see, nor do they need to know about, the file-level operations that the database engine performs. Instead, an end user deals with a conceptual layer that makes the data appear to have a certain logical organization.

The basic structures of a relational database (as defined by the relational model developed by E.F. Codd) are entities and attributes:

- *Entities* are the primary objects around which a relational database is organized. They include tables, columns, rows, keys, and indexes. For example, in the sample database, entities include courses, classes, instructors, students, grades, etc. (You can find the sample database in the \pvsw\demodata directory.)

- *Attributes* are descriptors for a column, table, or index. Attributes are the components of entities that define the uniqueness and usefulness of an entity. In the sample database, the Room table (entity) contains the attributes Type and Capacity. (The sample database is found in pvsw\demodata.)

In the relational model shown below, the end user perceives and manipulates data as tables, consisting of rows and columns. A database usually consists of several tables, with all information in a given table being related in some way. For example, in the sample database, there is a Class table. Each row in the table contains columns of information relevant to each class, such as class ID, name, section, maximum size, and so on.

Table 3-1   Relational Database Table

ID	Name	Section	Max Size	Start Date	Start Time	Finish Time	Building Name	Room Number	Faculty ID
91	ENG 305	001	50	06/06/95	02:00 PM	04:50 PM	Bartold Building	210	110-65-25-96

When searching for information in a relational database, the results are generated as a new table, using information from existing tables. This new table is called a *derived table*, or *view*, which in turn can itself be used to generate another, different view.

For more information about Pervasive.SQL 2000 and relational databases, refer to Chapter 5, "Product Architecture."

Transactional Model

The transactional database model is part of a class of nonrelational data models which includes the hierarchical, network, and inverted list models. The term *transactional* refers to the data access method used to navigate up, down, and sideways through data records. This method provides direct control and allows a developer to optimize data access based on knowledge of the underlying structure of the data. Btrieve is an example of a transactional database engine.

A transactional database allows you to find a specific record within a file and navigate to an adjacent record, the first record, the last record, or yet another specific record. This type of database access is quite fast. This is because the section of the file that contains the requested record (typically referred to as a page) is stored in the cache, or memory. Adjacent records, which are probably related to the current record and could be called next, are also placed in memory because they are a part of the same page as the current record. The following figure illustrates an example page.

Figure 3-1 Transactional Files and Records
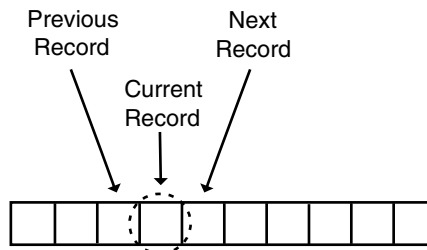
The powerful Pervasive.SQL 2000 engine is the basis of all Pervasive Software database products; it manages all of the low-level database functions for both the relational and transactional interfaces. In either case, the engine allows you to access records in their physical order, their logical order, or both. Physical order refers to the physical location of the data in the file. Whenever a new record is inserted into the file, it is written into the first available free space. Logical order, on the other hand, is a conceptual construct that organizes the records according to some specified system. Examples of logical ordering include alphabetical ascending or descending order.

Product Overviews

Getting to Know the Pervasive.SQL Product Family

This chapter introduces some of the features of Pervasive Software products, including Pervasive.SQL, the I*net Data Server and the Pervasive Software Developer Kit (SDK).

The following sections provide detailed Pervasive product information:

- "Pervasive.SQL 2000 SDK" on page 4-2
- "I*net Data Server" on page 4-3
- "The Pervasive.SQL 2000i Transactional Interface" on page 4-8
- "The Pervasive.SQL 2000i Relational Interface" on page 4-10
- "Migrating from Scalable SQL to the Pervasive.SQL Relational Database Engine" on page 4-12
- "Year 2000 Compliance" on page 4-13

Pervasive.SQL 2000 SDK

The Pervasive.SQL 2000i SDK includes many new features to ease the burden of application development. These include:

- Java. The Java Interface gives you the option of developing Btrieve applications in an object-oriented, platform-independent manner. It includes support for true null and Unicode values as well as for Binary Large Objects (BLOBs).

- Low-level APIs. Direct programming to the Btrieve API gives you the fastest possible data access execution and the most control over the way in which your application reads and writes data. If these considerations are important to your application and you are willing to manually develop code, you may find direct API programming highly useful. For relational access to data, you may also code directly to the Microsoft ODBC API.

- The ActiveX Interface. The ActiveX Interface allows you to leverage the power and speed of the Pervasive.SQL engine with a minimum of manual coding. These controls are designed for easy use with third-party grid controls as well.

- Complete sample application. Pervasive.SQL SDK includes a complete sample application designed to run a video rental store. Full sample code in Visual Basic, Delphi, Java, and C/C++ is supplied. Examples using ODBC, ActiveX RDO, third party controls, and direct API calls are shown.

- Developer's Resource Center. The Developer's Resource Center provides detailed tutorials and "recipe cards" designed to guide you through both easy and complex programming tasks, using the code from the sample application. All of this information is provided in industry-standard Windows Help format.

Development Environment

Pervasive.SQL provides an open interface that allows you to develop many front-end applications, all of which can share a common, transactional, or relational database. You can use popular programming languages such as Java, Delphi, BASIC, Visual BASIC, C, C++, COBOL, Pascal, ODBC, PowerBuilder (through ODBC), and FoxPro (through ODBC). In addition, bundling a Pervasive.SQL workstation engine with your application is easy with a Derivative Software License.

I*net Data Server

Pervasive Software's I*Net Data Server (IDS) is a revolutionary way to connect users to information stored in Pervasive.SQL files regardless of whether they are using an intranet or the Internet (thus, I*net). The IDS is a multi-threaded server using Windows NT domain security which provides for data compression and virtual sites. The technology employed by the IDS eliminates many of the preconditions for end-user connectivity, allowing you to choose your development environment as well as the type of product to enable across TCP/IP. The single-user I*net Data Server (IDS) test server is included in the Pervasive.SQL SDK installation

The current state of database connectivity lies in two camps. You can either choose wide-area connectivity and write a Web application, or you can choose network connectivity and write for the local network. In any case, a developer who wants full database connectivity in all situations is forced to make difficult choices: either write for the Web exclusively, or maintain two separate sets of source code in (very likely) two different programming environments.

The first choice, developing new applications exclusively for the Web, is initially appealing to many IS departments. It is not until the effort is made to create the actual sites that the difficulties arise. First, there is the learning curve of the new development environment, which is followed closely by discovery of the limitations of that environment. Then, there is the inevitable workaround of built-in HTML limitations.

Finally, the application is ready to be deployed and the first user logs on. All of a sudden, the same application the user has been running successfully in a browser-less mode is several times slower because of the resources allocated to the browser and the virtual machine running inside of it. This is nothing that upgrading every machine in the office will not fix — until the next browser release.

This is not to say that the Web does not have useful applications. There are times when the advantages of a Web-based application outweigh the disadvantages, such as when end-user browsing is essential (for example, online catalogs) or platform independence is a high priority. However, an optimal solution would provide the ability to not only write Web applications when needed, but also to

develop an Internet application using the same development and end-user skills that already exist in the organization.

The IDS can provide Pervasive.SQL access in both of these situations. By combining complementary clients, the IDS provides Internet connectivity to the Web as well as to most 32-bit Windows development environments. ActiveX is the tool of choice for applications developed with Rapid Application Development (RAD) tools, such as Visual Basic, Delphi, C++ Builder, and Visual C++. Additionally, for Web applications, platform-independent applications, or Java gurus, there is a Java class library that provides IDS client services.

The IDS runs on a Windows NT machine and processes Pervasive.SQL requests made by both of these clients. It can handle requests from all types of clients simultaneously, and compresses data using the algorithm used by each client request, so compression can be turned on and off on a per-request basis. The server acts as a proxy between Pervasive.SQL and the IDS clients, passing information as depicted in the following illustration.

Figure 4-1 IDS as Proxy Server
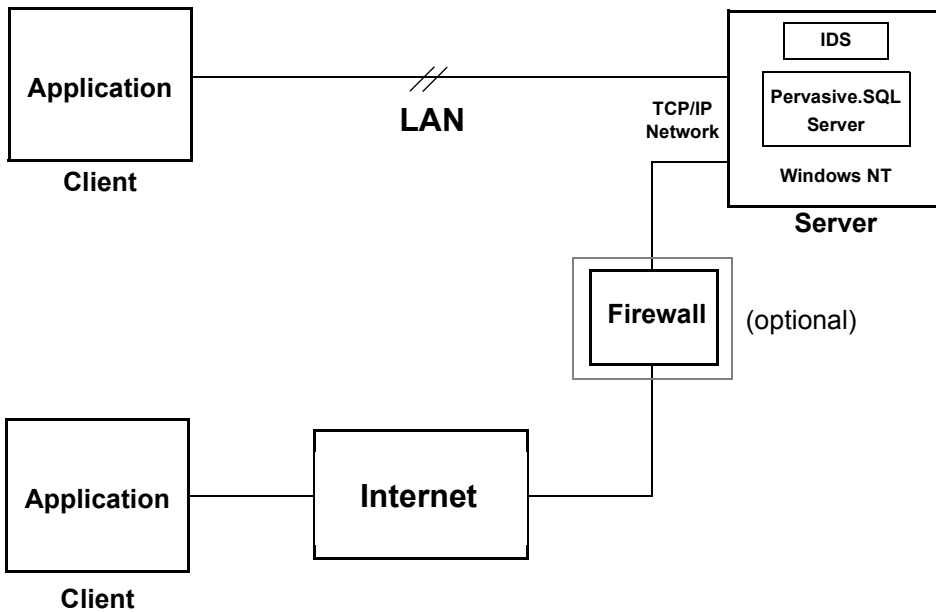
The IDS adds services in addition to straight-through Pervasive.SQL services. It is optimized to use bandwidth efficiently, as demonstrated with optional data compression. Likewise, DDF handling can be performed in only two calls: one to read the list of tables and one to read all the field information for a given table. These calls are processed internally in both clients. Furthermore, the send and receive buffers are not necessarily the same length.

This architecture optimizes calls across the I*net that have significantly different amounts of information in the send buffer and the receive buffer. For example, a GetDirect operation only needs to send four bytes of information yet may need to receive a 2-KB record. The IDS allows the client to send a data buffer of only four bytes while retrieving the full record.

Another advantage of the IDS that makes it ideal for an intranet is its easy configuration. While other Pervasive.SQL clients depend on a particular constellation of Pervasive.SQL settings (and as a result may even be incompatible with other, equally needed, Pervasive.SQL clients), the IDS clients do not need Pervasive.SQL settings at all. In fact, Pervasive.SQL does not need to be installed on a computer accessing Pervasive.SQL data via the IDS. This has the potential for dramatically reducing the number of support calls and making for much simpler installation programs.

The IDS can be used to provide access to an unsecured site or to any number of virtual sites. Virtual sites are implemented whenever security is enabled; a basic, secured installation would have only one virtual site—called database sets in the server — but a more complex installation could maintain any number of sites. Each site/database set has its own set of permissions which use Windows NT domain users and groups to validate logon access, and each site/database set can provide one of three levels of access: full access, read-only access, and write-only access.

Finally, each access level in each site/database set can have its own virtual root directory — all Pervasive.SQL calls are processed relative to this root, and none are allowed to traverse back up the directory tree (although any may go down the tree as shown in Figure). These virtual roots may actually exist on other machines on the local network, if desired. Because users and groups are established at the domain level, maintaining IDS security is only a small add-on to normal Windows NT domain maintenance.

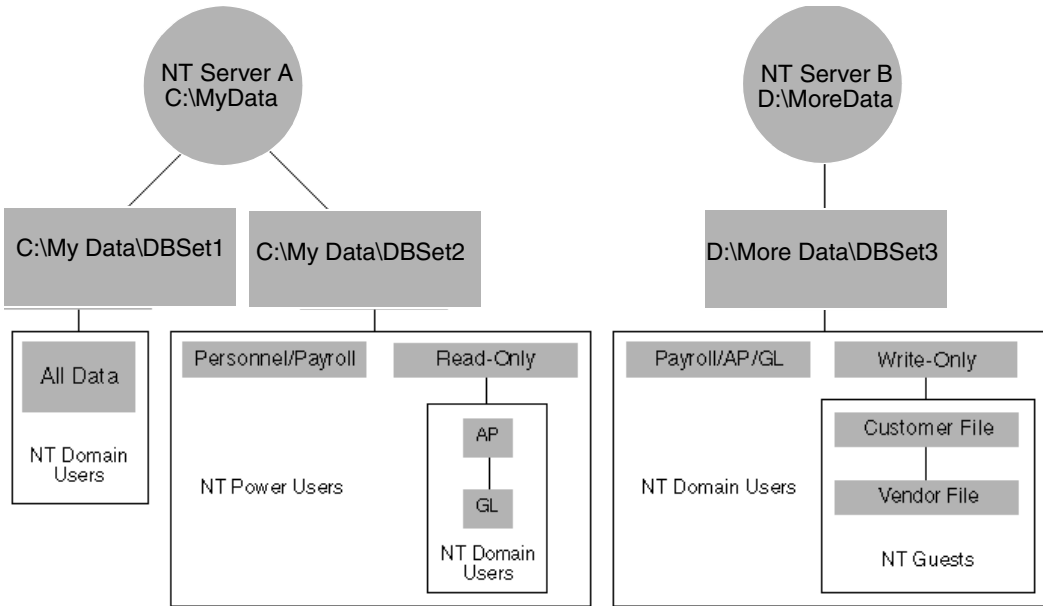*Figure 4-2 Examples of Different Configurations for IDS Database Sets*

Table 4-1   Example Permissions

Site #	Virtual Root	Example Permissions	Possible Scenario/Description
1	NT Server A\n\nC:\MyData\DBSet1	IDS Full Access: NT Domain Users (All users have full access to the files in C:\MyData\DBSet1)	This would be the simplest secured setup. Each user would have to offer a valid user name and password, but would have access to all of the data files allowed by the particular application. Different security levels for different tasks could be implemented via owner names and application-driven protection.
2	NT Server A\n\nC:\MyData\DB Set2	IDS Full Access: NT Power Users\n\nIDS Read-Only Access: NT Domain Users\n\nNT Power Users would have access to files on C:\MyData\DB Set2*.* and below\n\nNT Domain Users would have access to files on C:\MyData\Site2\ReadOnly\ *.* and below	This would be a situation where inquiries into the accounting system were fairly common for most employees, but only a limited subset would need entry rights or rights to payroll information. Again, security distinctions between the files in the Full-Access group could be made at the application level and/or by owner name.
3	NT Server B\n\nD:\MyData\DB Set3	IDS Full Access: NT Domain Users\n\nIDS Write-Only: NT Guests (or Web clients)\n\nNT Domain Users would have access to files on NT Server B\My Data\DB Set3*.* and below\n\nInternet clients would have write-only access to files on NT Server B\My Data\DB Set3\ WriteOnly*.* and below	This setup would be appropriate if the company maintained internal security via owner names or application protection, but wanted customers and vendors to be able to add or edit their own (address) information without the possibility of their looking up someone else's data.

The Pervasive.SQL 2000i Transactional Interface

The Pervasive.SQL transactional interface, built on Btrieve, offers easy installation, uncomplicated maintenance, and high levels of performance and reliability. Pervasive.SQL provides a foundation on which you can run transactional applications or migrate to a relational database system.

Benefits
Pervasive.SQL's transactional interface is Btrieve, which has been the data management system of choice for tens of thousands of applications around the world. In the highly competitive accounting software market—where reliability and performance are paramount—seven of the top 10 vendors choose Pervasive.SQL. Many application developers choose Pervasive.SQL for its speed, data integrity, easy scalability, and low maintenance costs. As part of Pervasive.SQL, Btrieve's transactional interface offers:

- Speed. Pervasive.SQL uses the highly-evolved MicroKernel Database Engine, capable of sub-second response rates, even when building multi-gigabyte databases for hundreds of users. The MicroKernel achieves these high speeds through features such as internal indexing algorithms that cache pages for fast data retrieval and updates, and automatic index balancing to keep data access speeds fast, even as your files grow.

- Data Integrity. The MicroKernel guarantees data integrity through rich transaction processing support, referential integrity controls, and automatic file recovery. In the event of a server or system failure, logging and roll forward utilities allow you to recover data up to your last completed transaction.

- Scalability. Many client/server database applications begin on the desktop and scale with corporate growth. Pervasive.SQL provides easy scalability from workstation to client/server environments.

- Low Cost. The low support costs experienced by Pervasive.SQL developers translate into low maintenance costs realized by Pervasive.SQL application end users. Pervasive.SQL eliminates the need for sustained database administration through automatic data recovery functions and easy-to-use utilities.

Features   Pervasive.SQL provides a comprehensive transactional database management system that offers many features, including the following:

- MicroKernel Database Engine as the underlying data manager.
- Access to databases distributed across multiple NetWare, Windows NT, and Unix server engines.
- Robust transactions for both single-server systems and distributed, multi-server systems.

The Pervasive.SQL 2000i Relational Interface

Pervasive.SQL's relational interface, built on the SQL Relational Database Engine (SRDE), offers easy installation, uncomplicated maintenance, and high levels of performance and reliability.

Benefits

Many relational database application developers choose Pervasive.SQL because it provides scalability, maintenance-free operation, and a small memory footprint:

- Speed. Pervasive.SQL offers direct ODBC access to the database engine. Many competitive products use a translation layer to translate ODBC calls to proprietary "native" relational API calls that then access the database engine. In contrast, the Pervasive.SQL ODBC driver calls the database engine directly, without translating ODBC calls to a proprietary relational API.
- Scalability. Pervasive.SQL allows you to scale applications from single-user to large client/server environments without changing the application or the database.
- Maintenance-free Operation. Pervasive.SQL is simple to install and use. It requires no extensive performance setup or ongoing tuning by a database administrator.
- Small Memory Footprint. Pervasive.SQL has a small footprint, requiring only 4 MB of memory.

Features

The Pervasive.SQL relational interface provides a flexible architecture that helps you easily scale your database applications from large client/server systems to single-user environments without additional coding. Pervasive.SQL offers easy installation, uncomplicated maintenance, high levels of performance and reliability, and a smooth migration path for data. In addition, bundling Pervasive.SQL with your application is easy with the Pervasive.SQL distribution component, which provides multi-user and single-user run-time support.

Pervasive.SQL has a comprehensive relational database management system interface that offers many features, including the following:

- Application scalability from standalone to client/server.
- Fully functional workstation and client/server engines.
- Declarative Referential Integrity.

- Bi-directional, updateable, and scrollable cursors.
- Named database support providing location transparency for applications.
- Comprehensive, industry standard data type support.
- Programming extensions such as triggers and stored procedures.
- Cost-based optimization from statistical analysis and enhanced fetch algorithms.
- Transaction processing enhancements such as full transactional logging.
- Standards enhancements, including ODBC support.
- Other new features include additional Windows utilities, large file support (up to 64 GB), and additional data type variables such as TIMESTAMP, UNSIGNED, and CURRENCY.

The SQL Data Manager

The SQL Data Manager utility includes the following functions:

- 100% compatible with ODBC.
- Create new table definitions.
- Create table definitions for existing Btrieve data files.
- Drop table definitions.
- Add/Drop table columns.
- Add/Drop named indexes.
- Display/Print table data, table definitions, and statistics.
- Check the database for inconsistencies.
- Enable/Disable database security.

You will make your database more efficient if you take the time to map a logical database structure from the conceptual model of the information to the physical file and disk layout.

➤ To map a logical data structure, follow these general steps:

1 Map your Btrieve data files to Pervasive.SQL tables and views.

2 Determine the relationships between the tables.

3 Normalize the tables by eliminating redundant columns. Normalization saves time in data entry, makes searches faster, and increases the stability and reliability of your database.

Migrating from Scalable SQL to the Pervasive.SQL Relational Database Engine

The Pervasive.SQL 2000 MicroKernel Database Engine (MKDE) will continue to support access from Scalable SQL engines released in previous versions of Pervasive.SQL to provide a smooth migration path to the SQL Relational Database Engine (SRDE). The SRDE uses the MKDE as a storage manager. While both the Scalable SQL engine and the SRDE can access the MKDE concurrently, we highly recommend you upgrade all your relational engines to Pervasive.SQL 2000i. Refer to the *Pervasive.SQL 2000 Application Migration Guide* available with the Pervasive.SQL 2000 SDK for further details.

Year 2000 Compliance

Pervasive.SQL 2000i and previous versions of Btrieve (v6.15) and Scalable SQL (v4.*x*) have been tested and are fully year 2000 compliant. However, each application accessing the Btrieve interface must also be properly engineered for year 2000 compliance. Please check with each application vendor using Btrieve for their compliance information.

Older versions of Btrieve (including v5.x and v6.10) may be compliant, however, testing has been limited to version 6.15 and later. We recommend upgrading to Pervasive.SQL 2000i if you have any compliance concerns.

Included in the Pervasive.SQL 2000i server package is Pervasive Software's ODBC v2.*x* driver. Pervasive's ODBC Interface v2 is fully level 2 compliant and does not present any year 2000 limitations. Older versions of our ODBC driver have not been tested, and it is recommended that you upgrade to the current version. Please contact our Sales Department at 800-287-4383, or visit our Web site at http://www.pervasive.com, for more information about upgrading your ODBC driver.

Product Architecture

Overview of the Pervasive.SQL 2000i Database Engines

This chapter introduces database management system architecture and the architecture of Pervasive Software products. The MicroKernel Database Engine is the basic, or core, layer for Pervasive.SQL. The MicroKernel fundamentals introduced in this chapter are relevant to both the SQL and Btrieve interfaces. The following sections describe Pervasive product architecture:

Database Architecture

Conceptually, the architecture of most Database Management Systems (DBMS) is divided into an internal layer and a conceptual layer as shown below. The internal layer provides low-level database functions such as physical data management, data caching, transaction processing, and data integrity enforcement. The conceptual layer implements the data abstraction required for the specific data model; in other words, it provides a representation of the data, as well as the definitions and functions necessary to translate between the layers.

Figure 5-1 General DBMS Architecture
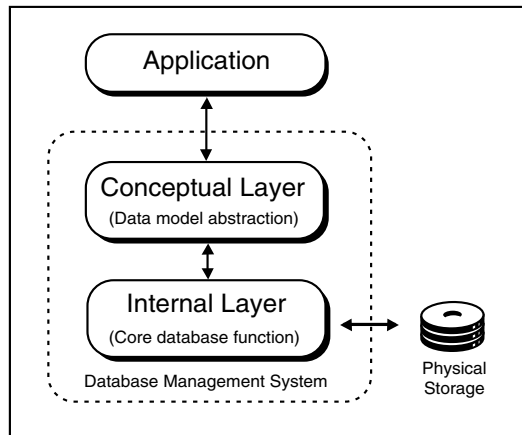
The MicroKernel Database Architecture expands on this model, as shown below. The MicroKernel provides the low-level functions of the internal layer, and the plug-in access modules (such as the SRDE, MKDE, or both) provide the conceptual layers. Multiple access modules can connect to a single MicroKernel Database Engine.

Data is stored in MicroKernel format, which is independent of the data model, operating system, and hardware platform. Applications using any Pervasive Software database engine can concurrently access data through the MicroKernel. You can configure the location of database engine components within a network, allowing applications to scale from standalone to client/server operation transparently.

Figure 5-2 MicroKernel Database Architecture
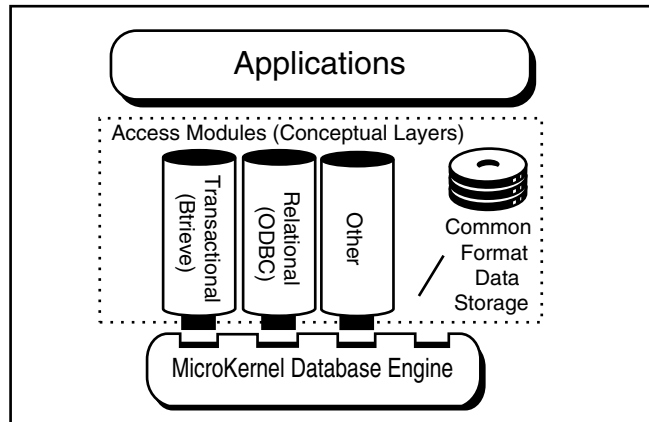
The MicroKernel provides low-level data management services for the access modules, such as the following:

- Physical data access
- Transaction processing
- Data integrity enforcement
- Referential integrity enforcement
- Data caching
- Logging and roll forward

In addition, the MicroKernel makes efficient use of disk space with automatic data compression, variable length data management, and other functions. The access modules are implementations of specific data models; they provide appropriate data structures and access techniques. (See Chapter 3 "Database Models," for more information about different models.) Access modules receive requests from applications and make calls to the MicroKernel to perform the required core data operations.

The Pervasive.SQL 2000i Database Engines

The Pervasive.SQL 2000i engine consists of two database sub-engines:

- MicroKernel Database Engine (MKDE), which provides Btrieve and MicroKernel API support for Pervasive.SQL 2000i applications.
- SQL Relational Database Engine (SRDE), which provides ODBC support for Pervasive.SQL 2000i applications.

MicroKernel Database Engine

The MicroKernel Database Engine (MKDE) provides Btrieve/ MicroKernel API support for Pervasive.SQL 2000i applications. The Workstation and Workgroup MicroKernel Database Engine (which runs on Windows 9x/NT) supports local applications running on the same computer as the engine. The Server MicroKernel Database Engine (which runs on Windows NT/2000, NetWare, and Unix) supports both local applications and remote (client/server) applications. The Workgroup MicroKernel Database Engine supports applications running on remote machines as well.

There are three versions of the MicroKernel engines in this release.

- Server Engine. Runs on Windows NT, NetWare, and Unix. Client platforms include DOS and Windows 3.x/9x/NT. Win16 applications running on a Win32 workstation that is executing a Pervasive.SQL 2000i engine locally are supported using the existing Btrieve thunk mechanism. Thunking is the default configuration for remote Pervasive.SQL 2000i engine access from a Win16 Btrieve application running on a Win32 workstation. It is possible to build transactional applications as NLMs (NetWare Loadable Modules) that access a local MKDE on NetWare.
- Workstation Engine. It is a stand-alone engine that runs on Windows 9x/NT and cannot be accessed from remote machines. The workstation MKDE is loaded when a Pervasive.SQL 2000i application starts running and a Btrieve or ODBC API call is made. The workstation MKDE remains loaded in memory until all Btrieve or ODBC applications have correctly released engine

resources (that is, logged out, closed files, issued correct number and type of Stop operations). Win32 workstations may run a DOS, Win16, or Win32 application and access a workgroup MKDE running on a different machine.

■ Workgroup Engine. This engine can service requests from other workgroup engines or from the Pervasive.SQL requesters running on a remote machine.

A "tray icon" is displayed to provide a graphical indication when a Pervasive.SQL 2000i workstation or workgroup MKDE is running. No tray icon is displayed when the workstation or workgroup MKDE is not running. The tray icon does not display for the server engine.

The Btrieve and ODBC APIs in Pervasive.SQL 2000i support writing distributed database applications that hide the details of connecting to a local or remote database engine from an application. Using this architecture, an application can access data that is co-located with the application (that is, running on the same computer as the application) while also accessing data on a remote computer. Moreover, a Pervasive.SQL database can be distributed by having DDFs (data dictionary files) serviced by a local MicroKernel Database Engine and data files (tables) serviced by remote MicroKernel Database Engine. This type of Pervasive.SQL database, which is not serviced exclusively by a local MicroKernel Database Engine, is referred to as a "mixed access database."

Mixed-access databases are subject to the following constraints:

■ The following features are not supported: bound databases, triggers, distributed transaction atomicity (requires two-phase commit).

■ The SRDE *and* the MKDE must be running on the same computer to access DDFs.

■ Data files for tables that are involved in an RI relationship, or those that have any triggers defined for them or are in a bound named database cannot be opened by a remote MicroKernel Database Engine.

■ When opening a file, the SRDE does not verify the version of the MicroKernel Database Engine servicing the request. If an operation that requires v6.30 or higher, MicroKernel Database Engine API support (for example, shared locking) is issued to a

MicroKernel Database Engine less than v6.30, then an error code is returned. When opening DDFs or when attempting to bind a DDF or data file, the SRDE verifies that the local MicroKernel Database Engine is servicing the request.

The MicroKernel Database Engine is the basis for Btrieve and MicroKernel API functions. It serves as the internal layer of the database and performs the basic tasks of data maintenance and retrieval.

To understand MicroKernel basics, you must understand data files, records, keys, and indexes.

Data Files and Records

The MicroKernel stores information in data files. Inside each data file is a collection of *records*. A record contains bytes of data. Using the sample database as an example, that data might represent a *row* consisting of a department name, phone number, building name, room number, and department head. These categories are called *fields*.

When an application retrieves the record for the Department of Music, it might display the information as follows:

```
Music 5126942600 Garrison Hall 520 297511594
```

Because the MicroKernel interprets a record only as a collection of bytes, the application is responsible for the appearance or formatting of the data once it has been retrieved. The MicroKernel does not view the data as "name," "phone number," and so forth; it must be told how the data within a record should be displayed. For example, an application that inserts or retrieves information about the Music Department might use a data structure based on the following format:

Field	Dept Name	Phone Number	Building Name	Room Number	Dept Head
Length	20 bytes	10 bytes	25 bytes	4 bytes	9 bytes
Data Type	Blank-padded string	ASCII Numeric	Blank-padded string	Single precision IEEE floating point	ASCII Numeric

Note that in this example, the application provides a name for the different fields in the record, the size of each field, and the data type (such as numeric or character string).

Keys

The MicroKernel can recognize information in a file that is defined as a *key*. A key is a specified field that is common to all the records in the file. If you think of a record as a row, then a key can be thought of as a column by which the file (a collection of records) can be sorted. An application or a user can designate any byte or set of bytes in a record as a key. In the previous example, the field called "Dept Name" is a key.

The purpose of a key is to provide the MicroKernel with fast, direct access to records. The MicroKernel can find a particular record based on a specified key *value* (using the previous example, the value of Dept Name for that record is Music). The MicroKernel also sorts records on the basis of the values in any specified key. For example, a sample application could use the Dept Name key to obtain a listing of all department names beginning with the letter "E", or it could obtain a listing of all departments and then display that listing, sorted alphabetically or by some other rule.

Each key has an *index*, which is used to locate information in the data file, much like an index of a book is used to find a particular piece of information in the book. If the key has a unique value, it identifies a single record, whereas a non-unique key value might point to several different records.

Indexes

Normally, when accessing or sorting information for an application, the MicroKernel does not search through all the data in its data file. Doing so would slow down searches significantly. Instead, the MicroKernel uses an index to perform the search and then manipulates only those records that meet the application's request.

For every key defined in a data file, the MicroKernel builds an index. An index functions like an address book for the data. The index is stored within the data file itself, and it contains a collection of pointers (addresses) to the actual data within that file. A key value is associated with each pointer.

Using the preceding example, the key "Dept Name" has an index. Inside that index is a collection of department names: one name for every department. For every department name in the index, a pointer indicates where the information about that department is physically located in the data file. The following diagram illustrates this using the department name index.

Figure 5-3 Pointers in Indexes
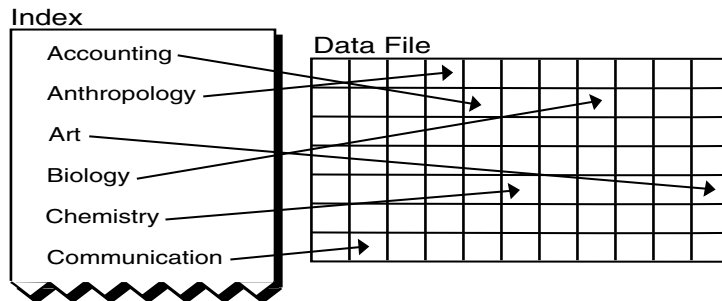
SQL Relational Database Engine

The SQL Relational Database Engine (SRDE) provides ODBC support for Pervasive.SQL applications. The SRDE runs as a service on Windows NT.

SRDE platforms include Windows 9x/NT, NetWare and Unix. The same SRDE is included in both the Win32 workstation or workgroup engine and NT server engine versions of Pervasive.SQL 2000i. On NetWare servers, NLM versions of the ODBC communications server, ODBC Driver Manager, and SRDE are provided.

ODBC client platforms include Windows 9x/NT. Remote ODBC application access to the SRDE requires installation of the ODBC client, which is a specialized ODBC driver that routes client-side ODBC calls to the ODBC communications server over the network.

Win16 ODBC applications running on a Win32 workstation that is executing a Pervasive.SQL 2000i engine locally are supported using the ODBC thunk mechanism. Thunking with the Microsoft-provided ODBC thunk DLLs is required for remote Pervasive.SQL 2000i engine access from a Win16 ODBC application running on a Win32 workstation.

Features of the SRDE include:

■   Atomic statements

- Bidirectional cursors (using the ODBC Cursor Library)
- Outer join support
- Updatable views
- ODBC data type support
- Multiple variable length columns in a table

Transactional Architectural Overview

The following diagram illustrates the basic architectural components of the Pervasive.SQL 2000i transactional interface.

Figure 5-4 Pervasive.SQL 2000i Transactional Architecture

Btrieve Application	Btrieve application sends a navigational database request to Btrieve.
Btrieve Glue Module	Btrieve Glue Module finds and loads the appropriate Btrieve Interface module.
Btrieve Interface	Btrieve performs navigational portion of request and sends requests to MicroKernel.
MicroKernel Database Engine	MicroKernel performs data management portion of request.
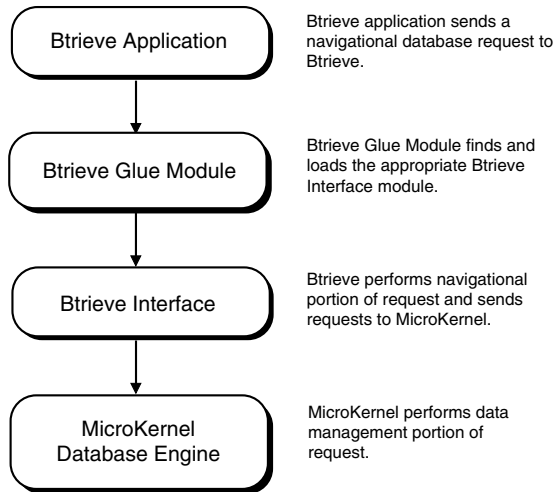
The transactional architecture can accommodate a variety of configurations, from single workstation use to client/server. The transactional engine also provides a set of utilities that help you install and manage the software.

Fundamentals This section introduces fundamental concepts of the transactional architecture. For more information, refer to the *Pervasive.SQL Programmer's Guide* available in the Pervasive.SQL SDK. This product provides a transactional interface for MicroKernel data files, which contain the actual information that an application stores, retrieves, or modifies. Data files consist of a series of pages. Because data files can be quite large (up to 64 GB), it is not practical or always possible to transfer such a huge file between memory and the disk; therefore, data files must be divided into units called pages.

Data File Components

As discussed in "MicroKernel Database Engine" on page 5-4, data files consist of records, keys, and indexes. A record can be thought of as a row of data in the data file. A key is a particular field in the record that can be used to sort the data. Each key contains one or more indexes that enable the MicroKernel to determine the physical location of the desired data. The relationship between these file elements is illustrated below.

Figure 5-5 Data File Components
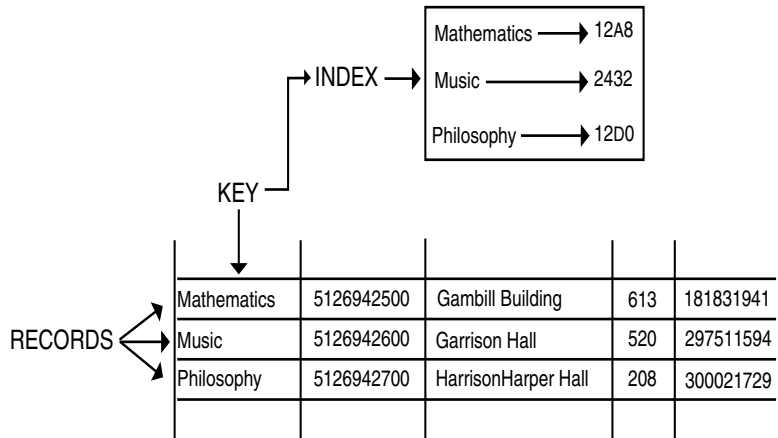
The above diagram shows three records from the sample database. These records belong to a data file that contains information about the different sample departments. Each record contains the department name, phone number, building name, room number, and department head ID.

In this example, the department name is a key. The records are sorted alphabetically by department name. The key contains an index that holds pointers to the physical location of each record in the data file. These pointers enable the MicroKernel to find information.

Enhanced B-trees

Pervasive.SQL 2000i transactional engine uses a hierarchical index type called a *b-tree*. The Pervasive.SQL 2000i b-tree is an enhanced, multi-level, or tree-structured, index. Because the MicroKernel searches indexes for information, a single-level index could, depending on the size, take some time to scan. By creating multiple

layers, however, this process can be shortened considerably and become much more efficient.

A multi-level index is essentially an index file containing one or more indexes. The sub-indexes do not contain all of the record entries for the file; if they did, they would be as large as the root (main) index itself. Instead, each sub-index contains a different set of the available entries.

Each index is a node of a b-tree. These nodes provide a search path for the MicroKernel. As the MicroKernel searches for a piece of data, it evaluates the search query at each node; essentially, the node is a fork in the road, so to speak, and the MicroKernel determines whether the information it needs is to the left or to the right. The search proceeds in this manner until the MicroKernel locates the desired record or information. Following is a simplified diagram of a b-tree.

Figure 5-6 Btrieve Indexes (B-trees)
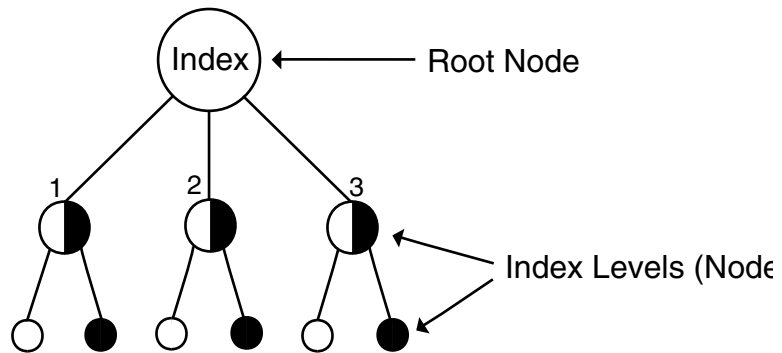
The MicroKernel begins at the root index and evaluates which path to take in search of the data. It may look for a physical record number, such as 78. If node 1 contains data for record numbers 1-50, node 2 contains data for numbers 51-100, and node 3 contains data for numbers 101-150, then the MicroKernel proceeds to node 2 and starts the evaluation process over again until it finds record 78.

Relational Architectural Overview

The following diagram illustrates the architectural components of the Pervasive.SQL 2000i relational ODBC interface for the server version.

Figure 5-7 Pervasive.SQL 2000i Relational Architecture: Server
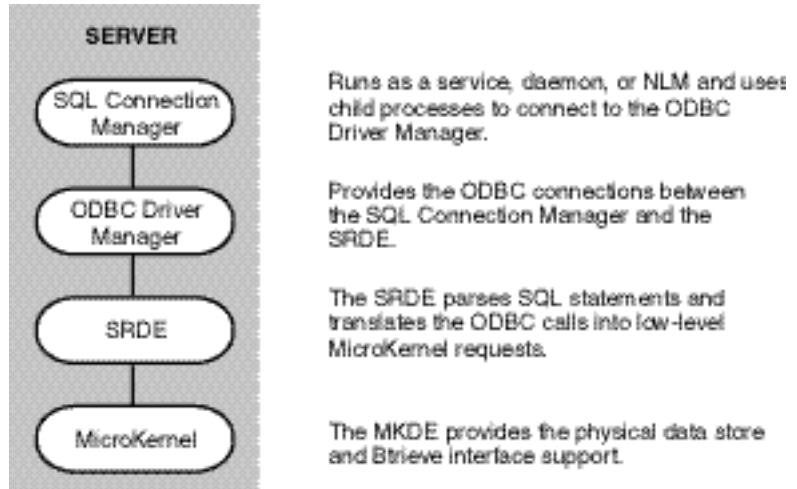
The SQL Connection Manager for the server consists of one executable and three DLL files:

- W3SQLMGR.EXE - SQL Connection Manager
- W3MGRMSG.DLL - Message File
- W3MGRRES.DLL - Resource DLL
- W3MGRRSX.DLL

In Windows, once the first SQL Connection Manager is opened, it invokes a child process of the same service. The SQL Connection Manager uses the ODBC Driver Manager to make calls to the SQL Relational Database Engine (SRDE), which in turn rests on top of the MicroKernel.

This following diagram illustrates the relational architecture of Pervasive.SQL 2000i on the client machine. The client talks to the SQL Connection Manager on the server through TCP/IP or SPX.

Figure 5-8 Pervasive.SQL 2000i Relational Architecture: Client
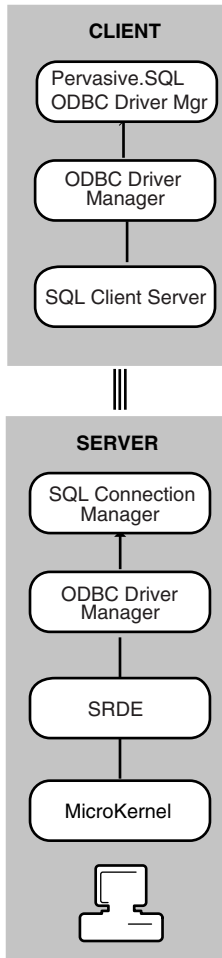
The following diagram illustrates the relational architecture of Pervasive.SQL 2000i on a standalone workstation.

Figure 5-9 Pervasive.SQL 2000i Relational Architecture: Workstation
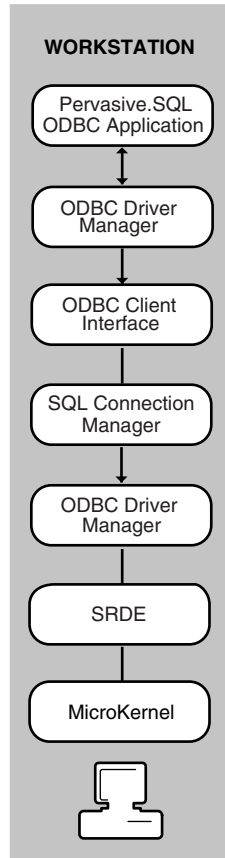
The following diagram illustrates the relational architecture of Pervasive.SQL 2000i in a workgroup computing environment, in which all participants in a workgroup can access data stored on an individual client.

Figure 5-10 Pervasive.SQL 2000i Relational Architecture: Workgroup Computing Environment

WORKGROUP

Pervasive.SQL ODBC Application	Pervasive.SQL ODBC Application	Pervasive.SQL ODBC Application
ODBC Driver Manager	ODBC Driver Manager	ODBC Driver Manager
ODBC Client Interface	ODBC Client Interface	ODBC Client Interface
SQL Connection Manager	SQL Connection Manager	SQL Connection Manager
ODBC Driver Manager	ODBC Driver Manager	ODBC Driver Manager
SRDE	SRDE	SRDE
MIF	MicroKernel	MIF

Common
Format
Data Storage

Pervasive.SQL 2000i Environment Configurations

SRDE and MKDE have one important thing in common: the core, low-level database functions for each are performed by the MicroKernel. This common base makes it possible for you to use both the Btrieve and SQL interfaces simultaneously to access data, allowing you to benefit from both relational and transactional database models.

The MicroKernel Database architecture allows you to distribute data processing and physical data files across a network without affecting the operation of applications. You can upgrade single-user applications running on standalone workstations to multi-user operation with a simple configuration change to the database environment.

The following examples illustrate the range of available configuration options, such as local workstation and client/server with local access environments. An additional example illustrates how Pervasive.SQL 2000i applications can integrate into a network and share distributed data.

Local Workstation

The local workstation configuration provides single-use standalone operation. All access module and MicroKernel components reside locally, and data files are stored on the workstation's disk drive. This configuration is used when the workstation is not connected to a network or when data files do not need to be shared. Examples of this configuration for the SQL interface, Btrieve, and both simultaneously are shown in the following diagram.

Figure 5-11 Local Workstation Configuration
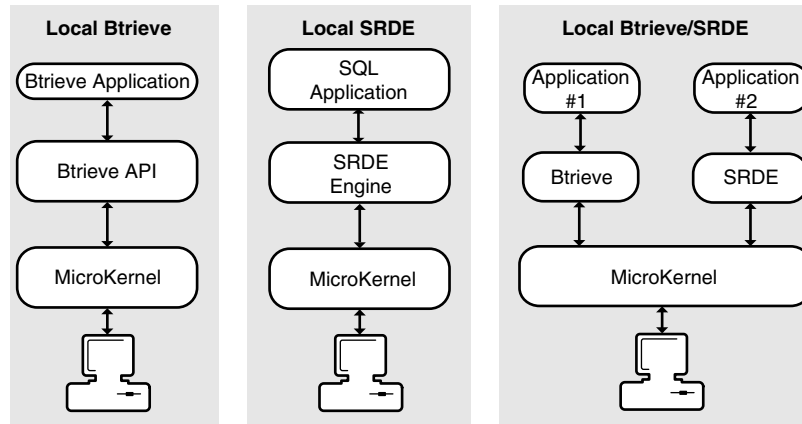
Client/Server   In a client/server configuration, database requests are processed on a NetWare, Windows NT/2000, or Unix server engine. A small requester module on the client machine routes requests from the application to a server database engine. Because all data processing and data files reside on the server, this configuration minimizes both network traffic and the use of client machine resources. The following diagram illustrates the Btrieve and SQL interface client/server configurations.

Figure 5-12 Client/Server Configuration
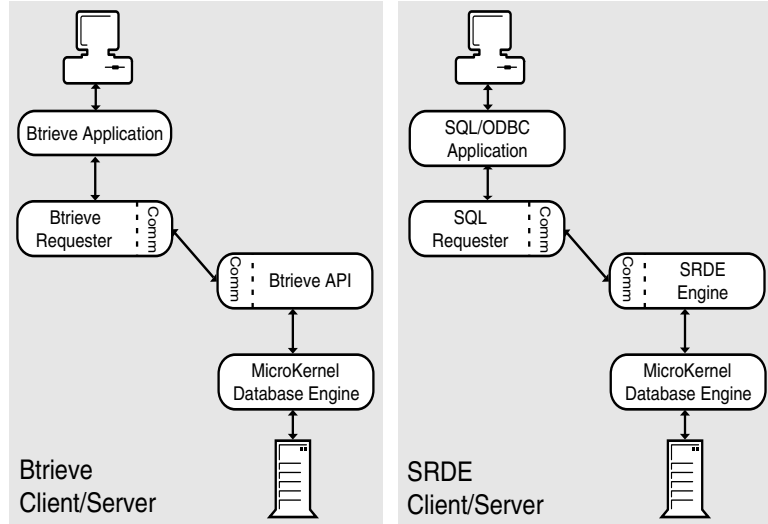
Although the figure shows only single client machines connected to single servers, the following configurations are also possible:

- A client machine can connect to multiple servers.
- Multiple client machines can connect to multiple servers.
- Multiple client machines can connect to a server simultaneously.

Client/Server with Local Access

Combining the client/server configuration with a local database engine allows applications to access local databases as well as databases on servers. The SQL Requester on the client machine routes data requests to either the local or server SQL Engine, depending on where the database is defined. The following configuration diagram illustrates the ability of a Pervasive.SQL server engine to access MicroKernels on other database servers.

Figure 5-13 Client/Server Configuration with Local Access
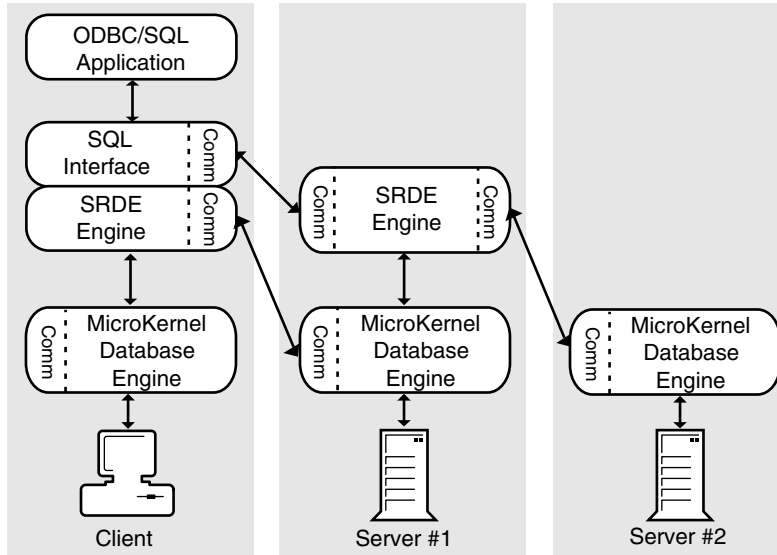
This allows the physical files that compose a database to be distributed across multiple servers, allowing you to build tables that contain data from multiple distributed databases.

Note Pervasive.SQL does not support this configuration on DOS machines.

Database File Storage

In most cases, you can achieve the best performance by storing as many of an application's data files as possible on the machine running the MicroKernel. Also, you can use referential integrity (RI) only on databases stored entirely on a single machine.

However, you can choose to split your database so that some files are stored on a workstation and some on a server, as shown in the figure below. This storage option is useful when either disk space or processing capability is limited on a particular workstation or server.

Figure 5-14 Database Split Between Workstation and Server

Single Database Split Between Workstation and Server
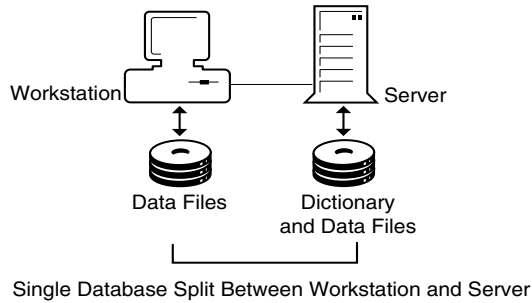
Similarly, you can choose to split your database among multiple servers, so that some files are stored on one server and some on another, as shown in the following figure.

Figure 5-15 Database Split Among Servers

Single Database Split Among Servers
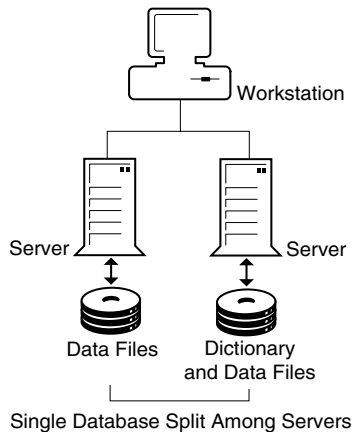
Pervasive.SQL: Combining Relational and Transactional Access

Because the MicroKernel provides the underlying data management for both the transactional and relational access methods in Pervasive.SQL, you can use both access methods simultaneously. In this hybrid relational/transactional environment, the MicroKernel high-transaction-rate applications remain accessible to hundreds or thousands of users, while other users simultaneously query relationally-modeled data through SQL applications. Pervasive.SQL realizes the best of both products without the high overhead costs associated with moving to more costly enterprise-targeted databases.

As shown in the following diagram, Pervasive.SQL allows you to integrate applications that use different data access methods within a network and still share data. You choose to implement the applications that use the most appropriate data model and extend existing environments with applications that use different data models. The diagram also illustrates how you can distribute data and processing throughout a network.

Figure 5-16 Example SRDE Network
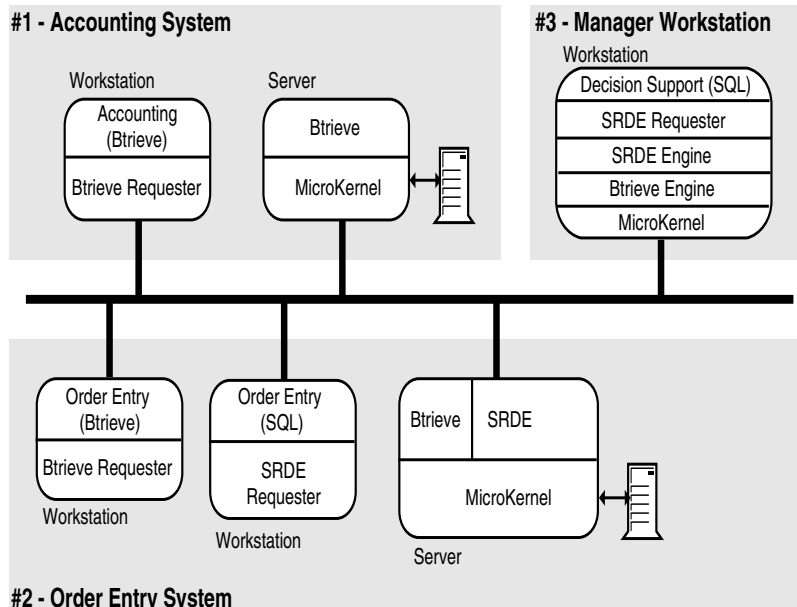
In the above diagram, shaded areas identify an accounting system, an order entry system, and a manager workstation.

The accounting system consists of a Btrieve server and client machine running an accounting application based on Btrieve.

The order entry system consists of a server running both SQL and Btrieve and client machines running either SQL or Btrieve order entry applications. This situation commonly occurs when new SQL applications are added to an existing Btrieve environment. Users can access data on the server using both the SQL and Btrieve applications concurrently.

The manager workstation is running an SQL-based decision support application and accesses data on the accounting server, the order entry server, and the workstation's disk drive. The workstation accesses order entry data by routing SQL requests to the order entry server. The SQL workstation engine issues requests for accounting data to the MicroKernel located on the accounting server.

Index