

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 services, product registration, and licensing information.

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

The Pervasive Software University database is provided as part of the Scalable SQL product and is frequently used in the documentation to illustrate database concepts and techniques. If you are already familiar with Scalable SQL, you may want to review the information in the *Programmer's Guide* to become acquainted with Pervasive Software University database.

Effective July 1, 1996, Btrieve Technologies, Inc. changed its company name to Pervasive Software Inc. The phrase Pervasive Software (or any derivation of Btrieve Technologies) implies assumed ownership under the Pervasive Software name.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

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. Please complete the User Comments form that appears on our Web site and fill in part number 100-003413-003.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

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 to you, including sales and technical support. This chapter also provides you with 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 Architecture”](#)

This chapter describes the basic architecture of Pervasive Software products, beginning with the MicroKernel Database Engine, which is the core engine for both Scalable SQL and Btrieve. This chapter also describes some sample configurations for Scalable SQL and Btrieve.

- [Chapter 5—“Product Overviews”](#)

This chapter describes the features and fundamental terms and concepts of Scalable SQL and Btrieve.

The manual also includes an index.

Conventions

Unless otherwise noted, command syntax and code 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.
[]	Square brackets enclose optional information, as in <i>[log_name]</i> . If information is not enclosed in square brackets, it is required.
	A vertical bar indicates a choice of information to enter, as in <i>[file name @file name]</i> .
< >	Angle brackets enclose <i>multiple choices</i> for a required item, as in <i>/D = < 5 6 7 ></i> .
<i>variable</i>	Words appearing in italics are variables that you must replace with appropriate values, as in <i>file name</i> .
...	An ellipsis following information indicates you can repeat the information more than one time, as in <i>[parameter ...]</i> .
::=	The symbol ::= means one item is defined in terms of another. For example, <i>a::=b</i> means the item <i>a</i> is defined in terms of <i>b</i> .

Pervasive Software Services

Pervasive Software is pleased that you have chosen our database management products. This chapter introduces you to our products and provides you with contact information. For the newest information about Pervasive Software's products, you can also contact us on the Web or via CompuServe.

This chapter includes the following sections:

- ["Company Information"](#)
- ["Contacting Pervasive Software"](#)
- ["Support Options"](#)
- ["Technical Support"](#)

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Company Information

Pervasive Software Inc. is a leading provider of high performance, multi-platform database engines. Pervasive Software's mission is to develop and deliver advanced information management solutions. This mission is realized through advanced database engines and tools that seamlessly integrate into an information management network offering users intuitive control over large amounts of distributed, complex, heterogeneous data. Pervasive Software is in a unique position to deliver this vision by virtue of its powerful MicroKernel Database Architecture, its channel of more than 50,000 registered professional developers, an experienced management team, and key industry partnerships.

Current Pervasive Software products include the Scalable SQL relational database, the Btrieve transactional database, and the ODBC Interface. These products are included in our flagship product, Pervasive.SQL. All products are built on Pervasive Software's MicroKernel Database Engine.

The MicroKernel Database Architecture separates the data access model from the low-level functions common to all databases. The MicroKernel supports relational, transactional, and ODBC access models while storing all data in a common, model-independent format. Both Scalable SQL and Btrieve applications can access all data concurrently. The MicroKernel provides a foundation for future object-relational and multi-dimensional access models.

The MicroKernel allows users to take full advantage of new applications with different and complex data types, while avoiding the expense of a costly database migration. For example, applications can manage mission-critical data using the Btrieve transactional model, and new applications using the relational model can subsequently access the same data without impacting the original system.

As the four time winner of Byte's Readers Choice Award for the best client/server relational database for workgroup computing, Scalable SQL provides the freedom to scale SQL applications from standalone to full client/server configurations without modifications to the application or database. Scalable SQL also offers low maintenance operation.

As the first transactional client/server database, Btrieve is the de-facto standard database used by professional developers, independent software vendors (ISVs), and value-added resellers (VARs) to build mission-critical applications. Designed for 3rd, 4th, and 5th generation language programmers, Btrieve offers rich transaction processing with a maximum degree of programming control, scalability from standalone to client/server environments, and low maintenance operation.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

Contacting Pervasive Software

You may contact Pervasive Software in the following ways:

USA

Pervasive Headquarters

Address	12365 Riata Trace Parkway Building II Austin, Texas 78727 USA
Phone	1-800-287-4383 Fax not available at publication (new headquarters site).
FAX	Not available at publication (new headquarters site).
E-mail	info@pervasive.com salessupport@pervasive.com techsupport@pervasive.com developer@pervasive.com docs@pervasive.com
World Wide Web	http://www.pervasive.com
CompuServe	GO BTRIEVE

Pervasive Software Nashville - Developer Relations (formerly Smithware Inc.)

Address	2416 Hillsboro Road, Suite 201 Nashville, TN 37212 USA
---------	---

Phone	1-800-828-7438
	1-615-386-3100
FAX	1-615-386-3135
E-mail	developer@pervasive.com
World Wide Web	http://www.pervasive.com/developer

Europe

Pervasive Software European Service and Support Center

Office Serves	Europe
Address	4th Floor 18-19 College Green Dublin 2 Ireland
Phone	+353 1 612 7403 +353 1 612 7400 (Technical Support)
Fax	+353 1 612 7405
E-mail	int-techsupport@pervasive.com

Pervasive Software Southern Europe

Office Serves	France, Greece, Italy, Middle East, North Africa, Portugal, Spain,
---------------	--

and
Turkey

Address	Rue des Trois Fontanots 20 esplanade Charles de Gaulle 92000 Nanterre France
Phone	+33 1 46 69 79 55
Fax	+33 1 46 69 79 84
E-mail	pidelson@pervasive.com

Pervasive Software United Kingdom

Office Serves	United Kingdom
Address	Pervasive Software Ltd Victoria House Desborough Street High Wycombe Buckinghamshire England HP11 2NF
Phone	+44 1494 601085
Fax	+44 1494 601091
E-mail	hgooder@pervasive.com

Pervasive Software

Northern Europe

Office Serves	Baltic States, Belgium, Denmark, Finland, Iceland, Ireland, Luxembourg, Netherlands, Norway, and Sweden
Address	Airport Boulevard Office Park Bessenveldstraat 25A B-1831 Diegam Belgium
Phone	+32 2 716 40 44
Fax	+32 2 716 47 22
E-mail	gvcutsem@pervasive.com

Pervasive Software Germany

Office Serves	Austria, Eastern Europe, Germany, and Switzerland
Address	Hessenring 121 D-61348 Bad Homburg Germany
Phone	+49 61 72 925 830
Fax	+49 61 72 925 889
E-mail	gkrauss@pervasive.com

Japan

Pervasive Software Japan

Address	Pervasive Software Japan, Ltd. World Trade Center Building Floor 24 Minato-ku, Tokyo 105-6124 JAPAN
Phone	+011-81-3-5405-2261
Fax	+011-81-3-5405-2269
E-mail	info@pervasive.co.jp

Sales and Ordering Information

Pervasive Software provides both a sales team to assist you in evaluating additional 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 Authorized Pervasive Software Solution Integrators. For sales assistance or more information, call 1-800-287-4383 or 1-512-794-1719. You may also send a fax to 1-512-794-1778, or send e-mail to salesupport@pervasive.com.

Pervasive Software's Solution Partner Program

The Pervasive Solution Partner Program is designed to build long term relationships with advanced technology organizations that provide application software, support, integration services, consulting, and training services necessary to deliver complete information management solutions.

The Solution Partner Program provides authorization and certification programs for individuals.

More and more companies today are realizing that teaming with a leader in the industry will help expand their businesses and gain greater profits, increase access to leading edge technologies, and network with other highly successful organizations.

Pervasive's Solution Partner Program provides Value Added Resellers and Independent Software Vendors with easy access to new products and information needed to service and solve customer's complex needs. Partners participating in this program receive technical and marketing training, sales/marketing information, and qualified leads and referrals. In addition to providing on-going technical and marketing seminars, Pervasive provides software for testing and demonstration purposes.

In addition, the Solution Partner Program offers certification programs for database administrators, developers, and trainers.

Highlights of the Solution Partner Program

- Authorization to sell Pervasive Software server and database engines, as well as support contracts
- Qualified leads and referrals
- Priority product support
- Training in database development, integration, and management
- Special discounts based on meeting certain requirements
- Access to demonstration and training software
- An opportunity to apply for beta tester status
- Joint marketing and sales opportunities
- Product literature and sales tools
- Free subscription to Pervasive's quarterly *Advantage* newsletter
- Free listing in our Solutions Directory
- Marketing and sales support from our highly experienced staff

To join Pervasive's Solution Partner program, attend a seminar, or receive more information, contact Pervasive Software at 1-800-287-4383 and request a program kit or you may fax a representative at 1-512-794-1760. Pervasive Software will respond with detailed information about the features, benefits, and requirements of joining one of the Solution Partner programs.

Pervasive Software Solutions Directory

The Pervasive Software Solutions Directory CD includes a directory of applications, services, tools, and utilities that support Scalable SQL and Btrieve. With a customer base of more than 50,000 developers, there are hundreds of thousands of Scalable SQL and Btrieve tools and applications that fall into virtually every vertical and horizontal market. To find out what is available, the Pervasive Software sales group, system engineers, and resellers have this information in a comprehensive reference listing.

If you have an application, service, tool or utility that supports Scalable SQL or Btrieve, we would like to list it in the Solutions Directory. Product profile forms are available through Pervasive Software's CompuServe Forum and Library 10 (Third Parties), or call 1-800-287-4383 or 1-512-794-1719 to have a product profile either faxed or mailed to you.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

Support Options

Pervasive Software offers a variety of support options as detailed in the following sections. The tables following these sections detail the options, features, and pricing for each support plan.

Premium Platinum Support

This highest level of support is designed for customers who deal with large volumes of Pervasive products and need dedicated resources available. In addition to the Platinum features, Premium Platinum customers receive unlimited support incidents for five named customer contacts, an assigned support account manager for escalation, access to a special priority phone queue, and pre-arranged pager access to a technician during critical situations. Five free training days at Pervasive and free "new release" training for two people are also provided, as well as two days of shadow training and two seats at the Pervasive Partner Conference. Premium Platinum customers are eligible for a 20% discount on Pervasive Consulting Services, along with two free days of access to the Pervasive "Performance Tune-Up" lab.

Platinum Support

Extensive technical resources are available with this annual fee-based plan. Benefits include three named customer contacts and 60 incidents, along with our anytime "no-questions-asked" backline support escalation policy. Platinum customers have access to a special priority email queue and our electronic technical newsletter, and are automatic enrolled in all Beta programs. Three free training days at Pervasive are also provided, as well as free product training for one person for new major releases, and a seat at the Pervasive Partner Conference. We also provide a support technician for one day of free "shadow training" to assist Platinum customers with their own training efforts.

Gold Support

An annual fee-based Gold support contract ensures that Pervasive developers and users will have comprehensive support coverage. In addition to the benefits of electronic and per-incident support, the Gold plan gives you two named customer contacts who can call in for assistance with up to 30 incidents per year. Gold support customers receive a complementary Pervasive.SQL developer kit as well as two free training days at Pervasive and our electronic technical newsletter, which includes notification of all software updates and patches.

Per Incident / Silver support

This fee-based option is designed for Pervasive customers who do not need the protection and assurance of an annual support program, but who do require occasional assistance in using Pervasive products. Here, for example, we can address your questions about client/server implementation - or help you on a per-incident basis with any of a wide range of other complex issues. Silver support provides a 5-incident pack to eliminate the overhead of processing payments each time you call. Silver support also gives you access to our electronic technical newsletter.

Electronic Support

The program offers cost-free electronic support in a variety of topics - known issues, frequently asked questions, technical tips - 24 hours a day, 7 days a week, through Pervasive's website and CompuServe forum. Maintenance releases and utilities are available for download from Pervasive's FTP site. All questions submitted to our support department - whether by email or our CompuServe forum - receive a response within three business days.

Complimentary 30-day Support

Because every customer needs support while they install, configure, and get started with a new product, Pervasive provides complimentary support to registered users for 30 days after the first call to Pervasive support. Complimentary support will be limited to installation, configuration, and first-time usage questions to ensure that new users—or existing users installing new Pervasive products—get the support they need to be successful.

Support Features and Options

The following tables describe the Pervasive Premium Support Program options, features, and pricing.

Table 1-1
Premium Customer Support Offerings

Options	Premium Platinum	Platinum	Gold	Silver	Per Incident	Electronic
Number of contacts	5	3	2	N/A	N/A	N/A
Number of incidents (including developer assistance)	Unlimited	60/year	30/year	5	N/A	N/A
Hours of coverage Monday through Friday (Central Standard Time)	7 a.m. to 7 p.m.	7 a.m. to 7 p.m.	7 a.m. to 7 p.m.	7 a.m. to 7 p.m.	7 a.m. to 7 p.m.	N/A
Priority in problem analysis and product defect fixes	1	2	3	4	5	6
Toll free number	Yes	Yes	Yes	Yes	Yes	N/A
Monthly usage reports	Yes	Yes	Yes			
Notification of maintenance releases and patches	Yes	Yes	Yes			
Emergency on-site support (pre-arranged)	Yes	Yes				
Early access to Beta software	Yes	Yes				
Anytime, no-questions asked escalation	Yes - real-time transfer	Yes - call back				

Free days of training at Pervasive HQ	4	2	1		
Special priority email queue	Yes	Yes			
Special priority phone queue	Yes				
Free Software Developer Kit	Yes	Yes	Yes		
Free attendance at Pervasive Partner Conference	2 seats	1 seat			
Free product training at Pervasive HQ with major releases (X.0)	2 person	1 person			
Free shadow-training with Pervasive Support Technicians at Pervasive HQ	2 person-days/year	1 person-day/year			
Tune-up in Performance Lab at Pervasive HQ	2 days/year free	Call for per-day pricing	Call for per-day pricing		
Pre-arranged pager access to Customer Engineering in critical situations	Yes				
Pre-assigned back-line Support Account Manager	Yes				
Consulting Services (available Summer '98 in United States)	20% discount, high priority	Standard priority	Lower priority	Referred through sales channel	Referred through sales channel
Automatic shipment of all product	Yes	Yes			

maintenance
releases

Automatic shipment of
all BETA test releases

Yes

Yes

Table 1-2
Electronic Services

Options	Premium/ Platinum	Gold	Silver	Per Incident	Electronic
CompuServe	Yes	Yes	Yes	Yes	Yes
Interactive Forum	Yes	Yes	Yes	Yes	Yes
Bug fix and patch information	Yes	Yes	Yes	Yes	Yes
Technical Tips	Yes	Yes	Yes	Yes	Yes
Frequently asked questions	Yes	Yes	Yes	Yes	Yes
Ability to download fixes	Yes	Yes	Yes	Yes	Yes
WWW	Yes	Yes	Yes	Yes	Yes
Access to Pervasive knowledge base	Yes	Yes	Yes	Yes	Yes

Table 1-3
Support Pricing

Options	Premium Platinum or	Gold	Silver	Per Incident	Electronic
----------------	--------------------------------	-------------	---------------	-------------------------	-------------------

	Platinum				
Annual Agreement	Call for pricing	Call for pricing	N/A	N/A	No Charge
Additional contacts	Call for pricing	Call for pricing	N/A	N/A	No Charge
Additional incidents	Premium - N/A	Use Silver add-ons	N/A	N/A	No Charge
	Platinum - use Silver add-ons				
Per incident	N/A	Use Silver add-ons	Call for 5-incident pricing	Call for pricing	No Charge

Technical Support

Pervasive Software currently provides technical support from 7 a.m. to 7 p.m. central standard time, Monday through Friday. For technical support and information about Pervasive Software's technical support programs, call 1-800-287-4383 or 1-512-794-1719, or e-mail techsupport@pervasive.com.

Knowledge Base

Pervasive's Knowledge Base (<http://www.pervasive.com/support/knowledge>) 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 Manufacturing 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 electronic mail to Pervasive's Manufacturing Partners, international distributors, MPP 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 Web site or submit a question directly to techsupport@pervasive.com.

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 Relations web site contains important information for developers, forums for discussion, and contact points for more information. Visit the site at:

<http://www.pervasive.com/developer>

In addition, Pervasive DevWire can keep you informed of the latest updates regarding Pervasive.SQL development. Pervasive DevWire is a free developer-oriented e-mail news service. For more information, visit:

<http://www.smithware.com/contact/DevWire.html>

CompuServe (GO BTRIEVE)

CompuServe is an electronic information subscription service. Users who have accounts on CompuServe can access the Pervasive Software Forum.

- Messages

The message section allows you to post questions which will be answered by Pervasive support personnel. The messages and subsequent replies are visible to the whole forum community, which provides another source of technical information for you to review.

- Libraries

The library sections contain downloadable updates and patches to our product offerings as well as additional debugging tools, documentation, third-party tools, and betas.

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.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Registration and Licensing

This chapter introduces Pervasive Software product information such as 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 our licensing, distribution policies, and product features. Forward any comments or suggestions to us via our registration card or via our CompuServe forum or Internet addresses.

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

- ["Product Registration"](#)
- ["Licensing"](#)
- ["Developing with Pervasive Software Products"](#)
- ["Upgrades"](#)

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

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.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

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 (unlimited users). The following two tables list the different licensing plans available for Pervasive.SQL 7, Scalable SQL, Btrieve, and the ODBC Interface.

Table 2-1
Pervasive.SQL 7 Product Licensing

License	Server	Workstation
SDK – <i>Limited Use Software License</i>	ü	ü
Single Seat – <i>Per Machine License</i>	N/A	ü
Unlimited Distribution – <i>Derivative Software License</i>	MPP Contract Only ^{**1}	ü
Client/Server – <i>Network and Intranet License</i>	ü	N/A
Client/Server – <i>Network License Amendment for Internet Service</i>	ü	N/A
Commercial Deployment	N/A	N/A

^{**1}Pervasive's Manufacturing Partner Program permits distribution on an OEM basis.

Table 2-2
Btrieve 6.15, Scalable SQL 4, and ODBC Interface Licensing

	Btrieve 6.15		Scalable SQL 4		ODBC Interface
License	Server	Workstation	Server	Workstation	

SDK – <i>Limited Use Software License</i>	ü	ü	ü	ü	N/A
Single Seat – <i>Per Machine License</i>	N/A	ü	N/A	ü	Included
Unlimited Distribution – <i>Derivative Software License</i>	MPP Contract Only1	ü	MPP Contract Only ^{**1}	MPP Contract Only1	CDK
Client/Server – <i>Network and Intranet License</i>	ü	N/A	ü	N/A	ü
Client/Server – <i>Network License Amendment for Internet Service</i>	ü	N/A	ü	N/A	ü
Commercial Deployment	N/A	N/A	N/A	ü	ü

^{**1}Pervasive's Manufacturing 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 1-800-287-4383 or 1-512-794-1719 or send e-mail to salesupport@pervasive.com.

Pervasive Software is committed to providing our 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 1-800-287-4383 for approval which depends on your licensing agreement.

Documentation is available in hard copy and on-line formats. We currently provide on-line documentation in both HTML and HLP (Windows Help) format.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

Upgrades

Upgrades are available directly from Pervasive Software by calling 1-800-287-4383 or 1-512-794-1719 or sending e-mail to salesupport@pervasive.com. Upgrades are also available through Pervasive Software Channel Partners. Refer to [Chapter 1, "Pervasive Software Services."](#) for more information.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Database Models

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



Note: The word “relational” refers to Scalable SQL and the SQL engine that is part of Pervasive.SQL. The word “transactional” refers to Btrieve (the transactional engine in Pervasive.SQL) and files that were formerly called “navigational.”

This chapter includes the following sections:

- [“Relational Model”](#)
- [“Transactional Model”](#)

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

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 coexist in the same table.

The relational database model is currently very popular. This is mostly due to 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. Scalable SQL is an example of a relational database.

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 university database, the Room table (entity) contains the attributes Type and Capacity. (The university database is found in Pvsw\Demodata.)

In the relational model, 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 university 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, etc. (See [Table 3-1](#)).

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.

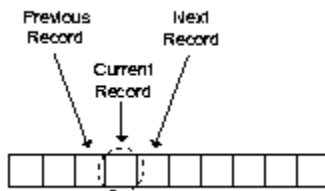
For more information about Pervasive.SQL and relational databases, refer to [Chapter 5, "Product Overviews."](#)

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 may be called next, are also placed in memory because they are a part of the same page as the current record. [Figure 3-1](#) illustrates an example page.

Figure 3-1
Transactional Files and Records



The MicroKernel Database Engine is the basis of all Pervasive Software products; it manages all of the low-level database functions for both Scalable SQL and Btrieve. In either case, the MicroKernel 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 Architecture

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 Scalable SQL and Btrieve interfaces.

The following sections describe Pervasive product architecture:

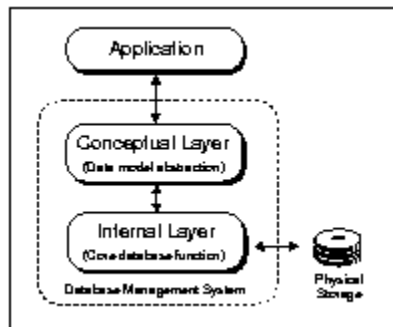
- ["Database Architecture"](#)
- ["The MicroKernel Database Engine"](#)
- ["Btrieve and the MicroKernel Database Engine"](#)
- ["Scalable SQL and the MicroKernel Database Engine"](#)
- ["Scalable SQL and Btrieve Environment Configurations"](#)

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Database Architecture

Conceptually, the architecture of most Database Management Systems (DBMS) is divided into an internal layer and a conceptual layer as shown in [Figure 4-1](#). 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.

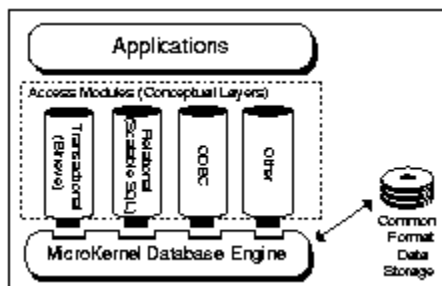
Figure 4-1
General DBMS Architecture



The MicroKernel Database Architecture expands on this model, as shown in [Figure 4-2](#). The MicroKernel provides the low-level functions of the internal layer, and the plug-in access modules (such as Scalable SQL, Btrieve, 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 4-2
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.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

The MicroKernel Database Engine

The MicroKernel Database Engine is the basis for all Pervasive Software products. 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 university 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.

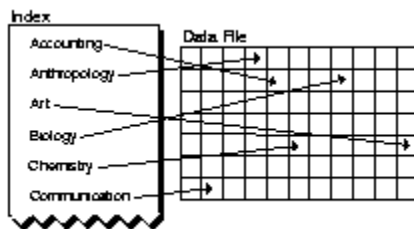
Indexes

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. [Figure 4-3](#) demonstrates this using the department name index.

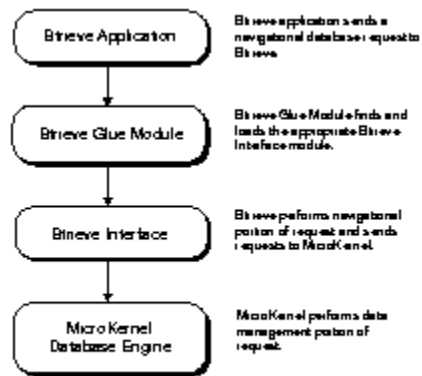
Figure 4-3
Pointers in Indexes



Btrieve and the MicroKernel Database Engine

Btrieve uses the MicroKernel to access data files. [Figure 4-4](#) illustrates the basic architecture of a Btrieve and MicroKernel configuration.

Figure 4-4
Btrieve and the MicroKernel Database Engine

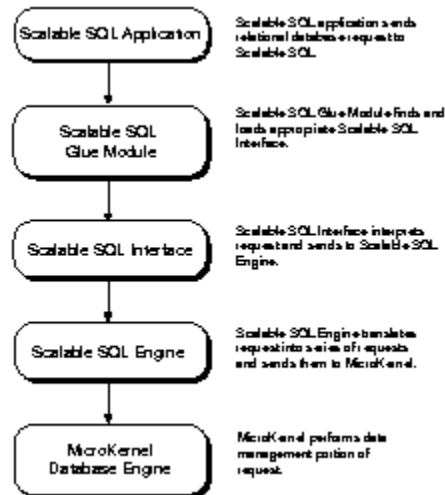


The Btrieve architecture can accommodate a variety of configurations, from single workstation use to client/server. Btrieve also provides a set of utilities that help you install and manage the software.

Scalable SQL and the MicroKernel Database Engine

Scalable SQL uses the MicroKernel to access data. As [Figure 4-5](#) illustrates, the MicroKernel performs file- and record-level operations, while Scalable SQL performs more abstract relational operations.

Figure 4-5
Scalable SQL and the MicroKernel Database Engine



The Scalable SQL architecture can accommodate a variety of configurations, from single workstation use to client/server. Scalable SQL also provides a set of utilities that help you install and manage the software.

Scalable SQL and Btrieve Environment Configurations

Scalable SQL and Btrieve 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 Scalable SQL and Btrieve simultaneously to access data, allowing you to benefit from both relational and transactional database models.

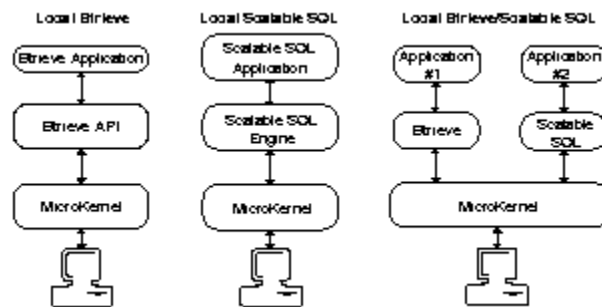
The MicroKernel Database Architecture allows you to distribute data processing and physical data files across a network without affecting the operation of applications. You can upgrade single-user applications running on standalone workstations to multi-user operation with a simple configuration change 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 Scalable SQL and Btrieve applications can integrate into a network and share distributed data.

Local Workstation

The local workstation configuration provides 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 Scalable SQL, Btrieve, and both simultaneously are shown in [Figure 4-6](#).

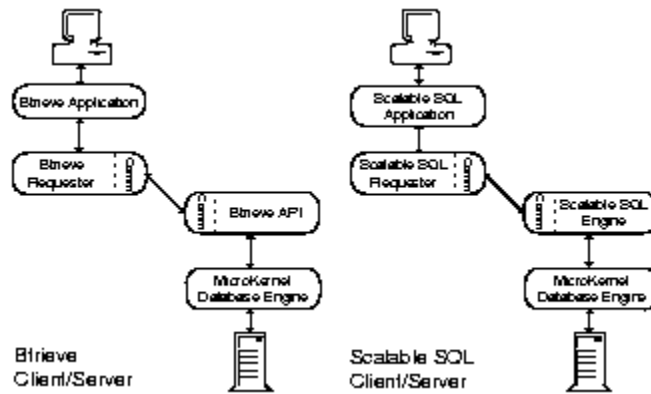
Figure 4-6
Local Workstation Configuration



Client/Server

In a client/server configuration, database requests are processed on a NetWare, Windows NT, or Warp Server 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. Scalable SQL and Btrieve client/server configurations are illustrated in [Figure 4-7](#).

Figure 4-7
Client/Server Configuration



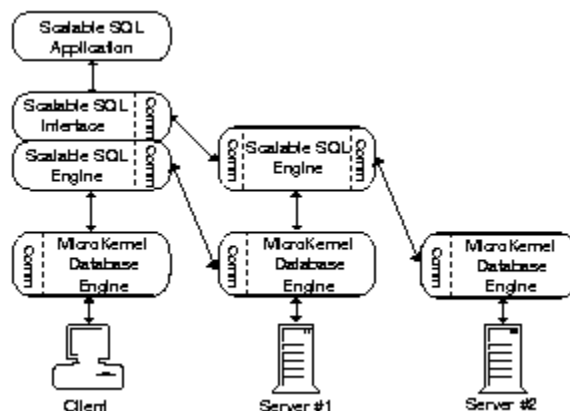
Although the figure shows only single client machines connected to single servers, the following configurations are also possible:

- 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. A Scalable SQL configuration illustrating this is shown in [Figure 4-8](#). The Scalable SQL Requester on the client machine routes data requests to either the local or server Scalable SQL Engine, depending on where the database is defined.

Figure 4-8
Client/Server Configuration with Local Access



[Figure 4-8](#) also illustrates the ability of a Scalable SQL server engine to access MicroKernels on other database servers. 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.



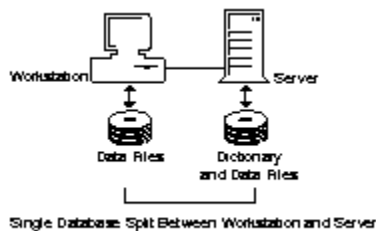
Note: Scalable SQL does not support this configuration on DOS machines.

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.

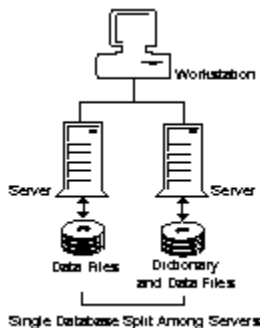
However, you can choose to split your database so that some files are stored on a workstation and some on a server, as shown in [Figure 4-9](#). This storage option is useful when either disk space or processing capability is limited on a particular workstation or server.

Figure 4-9
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 [Figure 4-10](#).

Figure 4-10
Database Split Among Servers

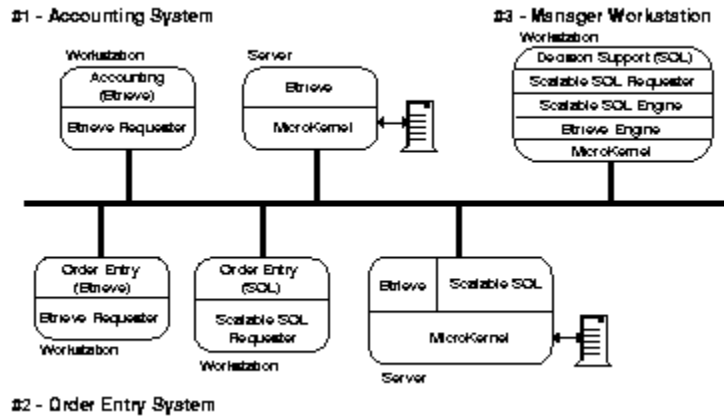


Combining Btrieve and Scalable SQL

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

As shown in [Figure 4-11](#), you can 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. [Figure 4-11](#) also illustrates how you can distribute data and processing throughout a network.

Figure 4-11
Example Scalable SQL / Btrieve Network



In [Figure 4-11](#), 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 Scalable SQL and Btrieve and client machines running either Scalable SQL or Btrieve order entry applications. This situation commonly occurs when new Scalable SQL applications are added to an existing Btrieve environment. Users can access data on the server using both the Scalable SQL and Btrieve applications concurrently.

The manager workstation is running a 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 Scalable SQL requests to the order entry server. The Scalable SQL workstation engine issues requests for accounting data to the MicroKernel located on the accounting server.

Product Overviews

This chapter introduces some of the features and fundamental concepts of Pervasive Software products. This chapter refers to terms and features of the MicroKernel Database Engine, which is discussed in [Chapter 4, “Product Architecture.”](#) If you have not already done so, read [Chapter 4](#) before proceeding with this chapter.

The following sections provide detailed Pervasive product information:

- [“Btrieve”](#)
- [“Scalable SQL”](#)
- [“Migrating from Btrieve to Scalable SQL”](#)

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Btrieve

Btrieve offers easy installation, uncomplicated maintenance, and high levels of performance and reliability. Btrieve provides a foundation on which you can run Btrieve applications or migrate to a relational database system, such as Scalable SQL.

Btrieve Benefits

For 15 years, Btrieve 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 Btrieve. Many application developers choose Btrieve for its speed, data integrity, easy scalability, and low maintenance costs:

- **Speed.** Btrieve uses Pervasive Software's 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. Btrieve provides easy scalability from workstation to client/server environments. In addition, the MicroKernel architecture allows you to move easily from transactional (Btrieve) to relational (Scalable SQL) models.
- **Low Cost.** The low support costs experienced by Btrieve developers translate into low maintenance costs realized by Btrieve application end users. Btrieve eliminates the need for sustained database administration through automatic data recovery functions and easy-to-use utilities.

Btrieve Development Environment

Pervasive.SQL includes many new features to ease the burden of application development. The ActiveX Interface, a complete sample application, and an online Developer's Resource Center all help to get new developers up to speed in record time.

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

Btrieve provides an open interface that allows you to develop many front-end applications, all of which can share a common, transactional 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 Btrieve workstation engine with your application is easy with an unlimited distribution license.

Btrieve Features

Btrieve is 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 Warp Server server engines.
- Robust transactions for both single-server systems and distributed, multi-server systems.

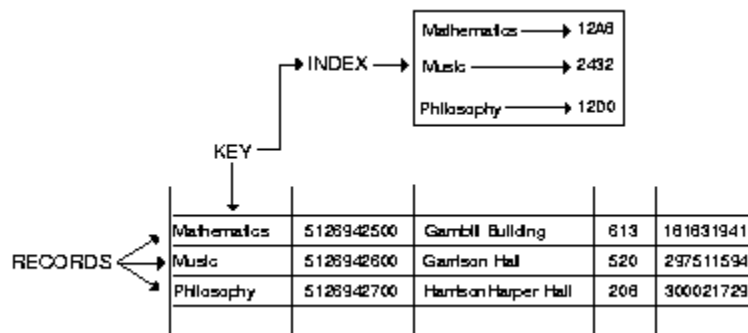
Btrieve Fundamentals

This section introduces fundamental concepts of Btrieve. For more info, refer to the *Programmer's Guide*. The Btrieve 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 gigabytes), 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.

Btrieve Data File Components

As discussed in [Chapter 4, "Product Architecture,"](#) 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 in [Figure 5-1](#).

Figure 5-1
Btrieve Data File Components



[Figure 5-1](#) shows three records from the university database. These records belong to a data file that contains information about the different university 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.

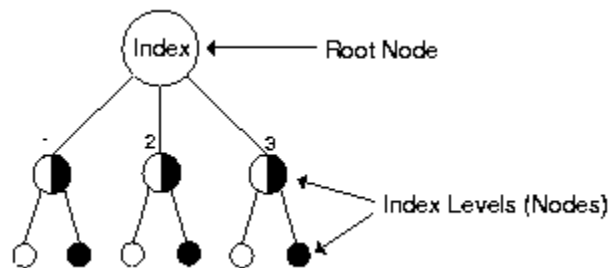
Btrieve Indexes (B-trees)

Btrieve uses a hierarchical index type called a *b-tree*. A b-tree is a 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 5-2
Btrieve Indexes (B-trees)



In [Figure 5-2](#), 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.

Scalable SQL

Scalable SQL offers easy installation, uncomplicated maintenance, and high levels of performance and reliability. Originally released as NetWare SQL, Scalable SQL is a four-time winner of *Byte Magazine's* Readers' Choice Award for Best Client/Server relational database management system.

Scalable SQL Benefits

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

- **Scalability.** Scalable SQL allows you to scale applications from mobile systems to client/server environments without changing the application or the database.
- **Maintenance-free Operation.** Scalable SQL is simple to install and use. It requires no extensive performance setup or ongoing tuning by a database administrator.
- **Small Memory Footprint.** Scalable SQL has a small footprint, requiring only 4 MB of memory.

Scalable SQL Development Environment

You can use popular programming languages such as BASIC, Visual BASIC, C, C++, COBOL, Pascal, ODBC, and PowerBuilder (through ODBC). In addition, bundling a fully-functional Scalable SQL workstation engine with your application is easy with its high volume deployment licensing.

Scalable SQL Features

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

Scalable SQL is a comprehensive relational database management system 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, updatable, and scrollable cursors.
- Named database support provides location transparency for applications.
- Comprehensive, industry standard data type support.
- Security features such as dynamic data encryption and decryption.
- Programming extensions such as triggers and stored procedures; also, Visual Basic compatible scripting via Inscribe.
- Cost-based optimization from statistical analysis and enhanced fetch algorithms.
- Transaction processing enhancements such as nested transactions and full transactional logging.

- Standards enhancements, including ODBC support.
- Other new features include additional Windows utilities, very large file support (up to 64 gigabytes), and additional data type variables such as TIMESTAMP, UNSIGNED, and CURRENCY.

Scalable SQL Fundamentals

This section introduces the fundamental concepts of Scalable SQL and relational database design. For more detailed information, refer to the *Programmer's Guide*.

The primary objects in a relational database are called *entities*. Tables, columns, rows, keys, and indexes are entities. A weak entity depends on the existence of another entity. In the sample database, Class is a weak entity because it is dependent on a Course being offered. We may not wish to retain Class if we delete the Course.

Attributes define the uniqueness and usefulness of an entity. The term *attribute*, as used in this manual, refers to a descriptor for a column, table, or index. Examples of column attributes are ranges, masks, default values, and value lists. Each attribute has various *properties*, such as data type, size, and relationships to other attributes. For example, the Name attribute of the Class table is a string data type and is 7 characters long.

Scalable SQL databases consist of *data files* and *data dictionary files*. A data file is a physical file stored on a disk. A data file is also sometimes referred to as a physical file or just a file. Scalable SQL data files contain the actual data, or information, that an application stores, retrieves, or modifies.

Data dictionary files contain relational information about the database, such as information about tables, columns, and indexes. Data dictionary files are also physical files stored on a disk that are themselves a relational database. You can look up data in these dictionary tables just as you can in the database itself.

Data and Tables

The most important concept in a Scalable SQL database is the *table*. A table is a collection of records, or *rows*. Each row in a table has the same number of *columns*. A column is the smallest unit of information in the database. It contains one type of information, such as a phone number or a company name.

A database can contain two types of tables:

Base Table	A set of rows that represents the data stored in a data file. Each row corresponds to a record in the associated data file.
View	A set of rows containing columns from one or more tables. Although data in a view may be stored in different base tables, you can treat a view as if all the data exists in a single logical table. (For example, you can insert and delete rows from a view.) Also, the columns in a view can be from the same table or multiple tables.

If you store the definition of a view so that you can recall it later, the view is a *stored view*. If you create a view by combining data from more than one table, the view is a *join*.

A table should contain information that is related in some way. For example, the university database has a Course Table that contains necessary information pertaining to each course. An excerpt from this table is as follows.

<u>Name</u>	<u>Description</u>	<u>Credit Hours</u>	<u>Dept Name</u>
-------------	--------------------	---------------------	------------------

GER101	German I	5	German
GER204	German IV	3	German
GER305	Studies in German Literature	3	German
HIS101	World History I	3	History

A table can have any amount of columns, but be sure that information is not repeated and that the column is related to the table. For example, columns containing information about faculty salary or a student's cumulative GPA do not belong to the Course table. It would be better to add faculty salary to a faculty table and cumulative GPA to a student table.

Scalable SQL supports *physical columns*, which are columns that store data, and *computed columns*. *Computed columns* are built from one or more column values or constants. For example, a table might include one column that contains a student's cumulative credit hours and another that contains the cost per credit. If you wanted Scalable SQL to calculate a computed column, such as tuition, you specify an *expression* to multiply the two column values.

Keys

A Scalable SQL key is a column or group of columns on which a table's referential integrity (RI) constraints are defined. In other words, a key (or combination of keys) acts as an identifier for the data in a row.

Each table must have a primary key. A *primary key* uniquely identifies each row in a table; no two primary keys can be alike. For example, in the university database Course table, the Course Name is unique; there can only be one course called German I. Because the key value is always unique, you can use it to detect and prevent duplicate rows.

A *foreign key* is a column that is common to more than one table and is the primary key of one of those tables. It is this relationship of a column in one table to a column in another table that provides the relational database with its ability to join tables.

RI can be defined only on named databases. For more information about keys and referential integrity, refer to the *Programmer's Guide*.

Indexes

You use indexes to define keys. An index points out one identifying column, such as a department name, that makes it easier and faster to find information.

Just as an index in a book helps readers quickly access information on a specific topic, a database table index speeds the process of searching and sorting rows. Although you can search and sort data without using indexes, indexes are more efficient. Use them to avoid or limit row scanning operations and sorting operations. If you frequently search and sort data by particular columns, you can create indexes on those columns, or if you regularly join tables to retrieve data, consider creating indexes on the common columns.

Indexes also serve to organize information in your database. An index provides the following:

- Faster row search and retrieval.
- Efficient access to data in multiple related tables.

- Rows are ordered automatically to support efficient data access. Using indexes, rows appear in indexed order instead of their stored physical order on disk.
- Each row is unique when you define an index as unique to ensure that duplicate rows do not occur.
- You can index a combination of columns to sort a table in several different ways at once.

If you create an index that contains more than one column, the index is segmented, and each column in the index is an index segment.

Structured Query Language

As the name implies, Scalable SQL uses Structured Query Language (SQL), a database language that consists of English-like statements that you can use to perform database operations. For example, SQL statements allow you to retrieve, insert, update, or delete data. You can also use SQL statements to create or drop tables and indexes, combine data from different tables, and set user permissions.

Both the American National Standards Institute (ANSI) and International Business Machines (IBM) have defined standards for SQL. (The IBM standard is called Systems Application Architecture (SAA).) Scalable SQL implements most of the features of both ANSI SQL and IBM SAA SQL and also provides additional extensions not specified by either standard.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](#) All rights reserved.

Migrating from Btrieve to Scalable SQL

Because Pervasive Software products share a common MicroKernel and file format, you can easily add Scalable SQL to your existing Btrieve environment to gain the benefits of a relational database management system. You may be an end user adding a Scalable SQL application to your existing Btrieve environment, running Btrieve and Scalable SQL applications concurrently on the same data. Or you may be a developer migrating your Btrieve application to Scalable SQL.

In either case, you can add Scalable SQL to most existing Btrieve databases without modification. Every Scalable SQL database consists of two basic parts:

- Data files that physically contain your data. (This is your existing Btrieve data.)
- A data dictionary that describes the data.

There are two ways you can add relational features to your Btrieve database: through Pervasive's SQLScope and DDF Ease utilities. You already have the data files, so you need only create the data dictionary. In Scalable SQL, the data dictionary consists of three required files (FILE.DDF, FIELD.DDF, and INDEX.DDF) and a number of optional ones.

Creating DDFs Using SQLScope



To create FILE.DDF, FIELD.DDF, and INDEX.DDF, follow these steps:

1. Create a directory in which to store the new data dictionary files. (For example, F:\MYDATA.)
2. Open SQLScope (a graphical tool included with Scalable SQL) and log in to the sample University database (also included with Scalable SQL).
3. In the SQL TEXT window, issue a CREATE DICTIONARY statement. For example:

```
CREATE DICTIONARY USING 'F:\MYDATA'
```

Scalable SQL creates default data dictionary files, which you must now edit to reflect the contents of your data files.

Once your database contains the three required data dictionary files, you can retrieve, insert and view your data using SQL statements. Refer to your Scalable SQL documentation for more information.

You can use SQLScope to create other DDFs (such as VIEW.DDF and RIGHTS.DDF) by copying default DDFs from the Scalable SQL install directory and editing the contents to match the new database's requirements.

However, 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 Scalable 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.

Creating DDFs Using DDF Ease

DDF Ease is a graphical application that allows you to create and maintain DDFs and database files. DDF Ease allows you to add relational capabilities to an existing Btrieve transactional database, create new databases, design new tables, check your database for inconsistencies with table definitions, and convert your dictionary from Scalable SQL 3.x to Scalable SQL 4.x dictionary format.

Because DDF Ease uses ODBC and Scalable SQL to create and maintain dictionary and table definitions, your database is able to work with Scalable SQL, ODBC, and ODBC-based third-party tools.

DDF Ease provides the following capabilities:

- Support for Scalable SQL 3.x and Scalable SQL 4.0 DDF file formats.
- 100% compatible with ODBC and Scalable SQL.
- Create, open, and delete DDFs.
- 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.
- Convert database dictionary formats.
- Check the database for inconsistencies.
- Enable/disable database security.

For more detailed information about creating DDFs with DDF Ease, launch the utility and run the application's context-sensitive on-line Help.

Send comments to docs@pervasive.com. Copyright © 1998 [Pervasive Software Inc.](http://www.pervasive.com) All rights reserved.

Year 2000 Compliance

Pervasive.SQL 7 and previous versions of Btrieve (version 6.15) and Scalable SQL (version 4.x) have been tested and are fully Year 2000 compliant. However, each application accessing the Btrieve engine 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 5.x and 6.10) may be compliant, however, testing has been limited to version 6.15 and later. We recommend upgrading to Pervasive.SQL 7 if you have any concerns.

Included in the Pervasive.SQL 7 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. A free upgrade is available for previous ODBC Interface customers. Please contact our Sales Department at 1-800-287-4383, or visit our web site at www.pervasive.com, for more information about upgrading your ODBC driver.

