

Migrating from Access to SQLBase for Performance and Reliability: A Technical Overview

By Suren Behari
Product Manager

TABLE OF CONTENTS

Abstract	3
Introduction: Database Choice for Small and Medium Sized Businesses	3
The Benefits of Migrating to SQLBase	4
Migration Strategies.....	8
Methods of Migrating Access to SQLBase.....	8
Method 1: Using Access to Export Tables to Text Files.....	9
Method 2: Using Access and ODBC for Direct Migration.....	9
Method 3: Access to SQLBase Migration Kit	12
Conclusion.....	14
Appendix	15
SQLBase Architecture.....	15

Abstract

Many businesses that build business applications using Microsoft Access eventually begin to struggle against its limitations. Now they have an alternative: *Move data repositories to a system that provides better performance and reliability and that is more flexible in how it allows information to be used.* SQLBase, the original multi-user PC database, remains today a popular business strength database server that runs under Windows and Netware. SQLBase provides performance and reliability enhancements such as a cost-based query optimizer, transaction support, and automatic crash recovery. By using SQLBase as a data management platform, Access users can continue to manipulate their data through Access as a front end if they wish, but they can also exploit their data in other ways.

This paper outlines the advantages of moving business applications from Access to SQLBase and presents three methods to migrate to SQLBase.

Introduction: Database Choice for Small and Medium Sized Businesses

Most businesses focus first on finding a solution to meet their business needs; the technology behind the solution is secondary. Is the database that powers the solution important? Of course. The ultimate goal of a database designed for small to medium businesses is to be unobtrusive. Certainly, a database must meet some basic requirements, such as speed, ease-of-use, scalability and security, to be a viable choice. The ideal database delivers high performance with the lowest possible cost of ownership to the user in terms of management effort or limitations on application delivery choices.

For small to medium size companies without large IT staffs, the need for reliable, low-maintenance solutions becomes critical. Because these businesses do not have large internal support organizations, they typically rely on integrators to help install and deploy ready-built solutions. Moreover, those solutions must be easy and inexpensive to maintain over time.

Access is a good database for relatively simple business applications, owing largely to its easy to use interface. However, as a database technology for business applications that may need to scale over time, Access has several limitations in how it manages transactions, handles security and manages data storage.

Access does not support atomic transactions: it does not guarantee that all changes performed within a transaction boundary are committed or rolled back. This poses a limitation for developers wanting control over database transactions.

An important issue to corporate environments is security. Access offers the ability to setup User-Level Security and to encrypt the .mdb file. Access does not offer 128-bit Triple DES Encryption, database page alteration protection, database page encryption, advanced password validation or transmission security. The entire database (.mdb) is encrypted when you encrypt an Access database.

The ultimate goal of a database designed for small to medium businesses is to be unobtrusive

Because of its storage management, data integrity and security limitations, you may be considering migrating to a more suitable database for your business applications. This paper outlines some benefits that you stand to gain by using the SQLBase database to manage your data, and provides some guidelines to help you migrate locally stored Access tables to SQLBase.

The Benefits of Migrating to SQLBase

Whether one is creating a new application that will access a database or migrating an existing database application to a new database engine, it is important to consider the unique attributes of the underlying database technology.

This section highlights some of the important advantages of SQLBase.

Deployment of information

When your information resides in SQLBase, you're free to continue using it from Access if you wish, but a number of other possibilities open up as well. This allows your data to be exploited more fully in more contexts, and by more people. For example, other people can use the data through the standard SQLBase client programs or from GUI-based applications. Your database also becomes more accessible over the Web. Microsoft Access now provides some capabilities for making a database available on the Web, but if SQLBase manages the database, you have a wider range of options.

Multiple-user access

Although Access provides some data sharing capabilities, that's not really its strength. SQLBase, on the other hand, easily handles many simultaneous users. It was designed from the ground up to run in a networked environment and to be a multiple-user system that is capable of servicing large numbers of clients. SQLBase is capable of supporting up to 50 concurrent users at the highest levels of enterprise-class performance, depending on the database design and application architecture. SQLBase is generally recommended for 1 to 30 concurrent users.

Management of large databases

Effective management of disk space becomes critical as databases grow. If a database is currently larger than 2 GB, or has the potential to grow beyond this limit in the near future, a true multi-user relational database is typically the best option.

SQLBase enables you to spread individual databases and logs across multiple disk partitions or volumes. In addition to allowing SQLBase to handle large databases, partitioning can enhance performance when using multi-disks and controllers. SQLBase provides partitioning on all supported platforms. SQLBase also supports non-partitioned data up to 512 gigabytes on 32-bit Windows NT and 95 operating systems with file systems that support files greater than 2 gigabytes.

SQLBase also enables you to define multiple named database areas that are physical files or devices where database data and log files can be stored. These areas can be spread across multiple disk drives to take advantage of parallel disk input/output operations.

**SQLBase easily
handles many
simultaneous**

**SQLBase
supports
transactions of
unlimited size**

Backup Management

Vital to any database management system is a comprehensive and effective backup and restore mechanism. SQLBase has been designed since its inception with this in mind. In addition to allowing incremental backups on-line, full on-line backup are also possible, giving you the flexibility you need to manage operations successfully and unobtrusively.

Transaction Support

Defined as a "logical unit of work," a *transaction* is one of the features common to most database management systems. By wrapping multiple database operations into a single unit, transactions offer the developer the ability to enforce data integrity by making sure multiple operations can be treated by the engine as an "all or nothing" proposition, thereby never allowing the database to end up in an inconsistent state.

The most common example of transaction processing involves a bank's automated teller machine. The processes of dispensing cash and then debiting the user's account are considered to constitute a logical unit of work and are therefore wrapped in a transaction: The cash is not dispensed unless the system is also able to debit the account. By using a transaction, the entire operation either succeeds or fails. This maintains the consistent state of the ATM database.

Transactions can be defined by what are known as the *ACID* properties. The following attributes of transactions make up the ACID acronym:

Atomic denotes that transactions are all-or-nothing operations. Each operation wrapped in a transaction must be successful for all operations to be committed.

Consistent denotes that a transaction enables data operations to transform the database from one consistent state to another.

Isolated denotes that all transactions are "invisible" to other transactions. That is, no transaction can see another transaction's updates to the database until the transaction is committed.

Durable denotes that after a transaction is committed, its updates survive — even if there is a subsequent system crash.

SQLBase supports transactions of unlimited size, constrained only by available memory, on all platforms, from the high-end workgroup server down to the laptop server engine. It also gives the protection of fully automated crash recovery, ensuring that uncommitted transactions are backed out and committed transactions rolled forward on restart of the server, even if a power failure occurs during the commit phase.

Two-phase commit gives you upscale server functionality in an easy-to-use package. SQLBase supports transactional integrity across multiple databases using two-phase commit technology. This allows a single transaction to execute changes to multiple SQLBase databases on completely separate hardware systems. This occurs as long as they are on the same network-functionality that was only previously possible using large, expensive systems with complex database and networking components.

Distributed operations can be very important and require the guarantee that transactions execute cleanly across functionally separate but related databases. SQLBase supports transparent, automatic two-phase

Database authority and security is paramount

commit functionality that works between both workgroup servers and standalone systems. For example, a two-phase commit transaction could be issued against a NetWare multi-user server and a Windows NT desktop server. No special programming is required by applications to implement such transactions simply a single API call to enable the processing.

Remote Administration

SQLConsole is a fully graphical database management tool for SQLBase administration and performance monitoring. It offers unparalleled ease of use and it lets you conduct all operations needed to create, maintain, and administer your SQLBase databases and servers with no need to write SQL. A copy is supplied with every SQLBase server.

SQLConsole can run either on Windows-based servers (including single-user systems), or on client workstations, allowing remote management. It puts control of all SQLBase functions at your fingertips, with the emphasis on a consistent graphical interface, and is designed both as an aid in application development, and as a tool for DBAs.

Security

Database authority and security play major roles today in all organizations, due to the increasingly sensitive information applications have access to, such as a company's accounting, banking and payroll information.

Database Authority

Database authority controls who can access a database and what they can do once connected to it.

SQLBase controls access with usernames and passwords. SQLBase has 4 levels of users.

SQLBase Authority levels are hierarchical and serve to tightly control data access:

- **SYSADM:** Creates user accounts and designates users' authority levels and passwords.
- **DBA:** Grants, changes, or revokes the object privileges of any user.
- **RESOURCE:** Creates and drops objects. Grants, changes, or revokes the privileges of other users on those objects.
- **CONNECT:** Accesses objects, but cannot create them.

Database Security

If information is a corporate asset, then it is important to the organization that it be protected from prying eyes. There are three potential areas of vulnerability, each of which must be strongly guarded:

- On the wire, between the client application and the database. Network analyzers may compromise networks; but no less susceptible is the notebook, where people may introduce sniffers or tracing products, that can monitor the conversation between the application and the database.

SQLBase offers an extensive range of application programming interfaces and tools

- In the database: the storage systems (the files) for the database must also be strongly encrypted, because the data must remain safe and private even if the database engine has somehow been shut down.
- In the backup files and log files; because the information in there contains copies of the information contained within the database.

SQLBase offers very good protection against unauthorized data access. Encryption is offered at the server level, database level, database page level, and transmission level resulting in end-to-end security, a database industry first.

SQLBase database security levels refer to the type of encryption used to hide the data within the database. There are four security levels of encryption available:

- None = SQLBase Standard (No encryption)
- Low = SQLBase Standard (Cryptogram encryption)
- Medium = SQLBase Treasury Edition 56 (56-bit DES encryption)
- High = SQLBase Treasury Edition 128 (same features as SQLBase Treasury Edition 56, with 128-bit Triple DES encryption)

SQLBase Treasury Editions offer security at every level of the client/server process. Features include:

- Enhanced server security
- Advanced password validation
- Database page encryption
- Database page alteration protection
- Transmission security

Connectivity

SQLBase offers an extensive range of application programming interfaces (APIs) and tools.

- SQL/API - a rich and thoroughly documented set of functions that provide total control of both data access and manipulation, and server operations from C.
- SQLBase++ - an extension to the Microsoft Foundation Class Library (MFC) and is fully integrated into the Visual C++ development environment. It gives developers an object-oriented alternative to the standard C API.
- JDBC Driver - a level 4 driver, which means it's a pure Java driver that communicates with SQLBase using a low level protocol known as the Message Level Interface (MLI). No other gateways or other layers of middleware is needed, hence the SQLBase JDBC driver is well suited for Java applets on the Web and Intranets.
- ODBC Driver - is implemented as a layer on top of the SQL/API. This driver allows developers to concentrate on writing business logic with the development tool of their choice, without having to implement the specifics and complexities of their middleware API.

- OLE DB Data Provider - takes advantage of Microsoft's Universal Data Access strategy to simplify access to SQLBase functionality from third-party programming tools such as Visual Basic, Delphi, and C++. Since the OLE DB Provider provides a set of COM interfaces, consumers can access information from SQLBase from any language.
- .NET Data Provider will be available in January 2003.

Migration Strategies

Should you wish to migrate from Access to the SQLBase database, you can do so either partially or completely. For those users who are more comfortable working with the Access interface as the front-end, they can continue to use the interface by migrating partially: Transfer locally stored Access tables to SQLBase, then set up links in the Access interface that point to the tables managed by the SQLBase database server. This way you retain the familiarity of the Access, but also take advantage of the strengths of SQLBase for data storage, management, and security. If you're less tied to the user interface, you can migrate completely away from Access. Transfer your Access tables to SQLBase, then use your information with tools intended for working with the SQLBase database.

Because of the "quick start" approach to many of these Microsoft Access implementations, designing a new data model may be the best approach, to get the most benefit from converting to SQLBase. If the conversion includes a data model design change, then custom applications will need to be developed to read the data in the existing ACCESS data structures, convert the data to the new data structures format, and write the data to the SQLBase database.

Methods of Migrating Access to SQLBase

In general, to migrate information from Microsoft Access to the SQLBase database, the first step is to copy the contents of your tables from an Access database to the SQLBase server. (To perform the operation of transferring the tables to SQLBase, you can choose from several methods, described below.) If you plan to continue using Access for the interface to your data, the next step after transferring the tables is to replace them with links: Delete the tables stored in your Access database, establish an ODBC connection from Access to the SQLBase server, and recreate the tables as links to the SQLBase tables. (Naturally, before you delete anything, it's prudent to make a backup first, just in case something goes wrong.) If you don't plan to continue using Access, you need not create any links.

Some transfer methods require making an ODBC connection to the SQLBase database server. Gupta SQLBase includes the SQLBase ODBC driver, and can set it up for you during the SQLBase installation procedure.

For those applications that have outgrown the features offered by Microsoft Access and need to scale up to a more sophisticated and robust database such as SQLBase, the following describes three methods that can be used to convert the data.

You can retain the familiarity of the Access, but also take advantage of the strengths of SQLBase for data storage, management, and security

Method 1: Using Access to Export Tables to Text Files

One approach to migrating data from Access to SQLBase is to use the export feature provided by Access itself to write out the contents of each table as a text file. Each file then can be loaded into SQLBase using the LOAD statement in SQLTalk.

The advantage of this approach is that it requires no special conversion tools. It can be used to produce data files even on machines that have no SQLBase database support. (If you don't have the SQLBase database client components installed on your Access machine, create the data files, then copy them to another machine where the SQLBase software is installed and load the files into SQLBase from there.)

The disadvantage is that the SQLBase database tables must already exist before you can load data into them, so you must issue the appropriate CREATE TABLE statements yourself.

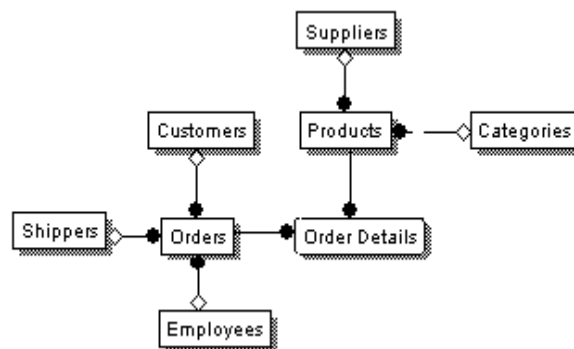
Method 2: Using Access and ODBC for Direct Migration

By using the Microsoft Access import/export feature, it is possible to convert an Access database, table by table, to a new SQLBase database.

First we will look at the Microsoft Access sample database "Northwind", and then I will describe the steps to convert it to SQLBase database.

Northwind Database:

The database contains 8 tables:



Note: Products are dependent on Suppliers and Categories

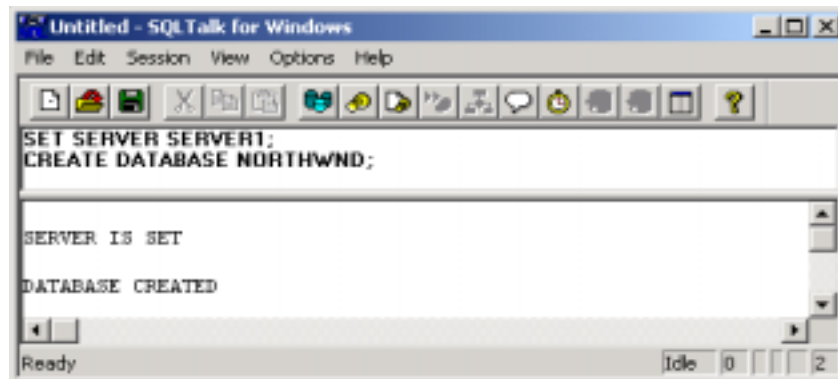
Orders are dependent on Shippers, Customers and Employees

Tables that are not depend on other tables should be created first:

- (1) Customers, (2) Suppliers, (3) Categories,
- (4) Shippers, (5) Employees, (6) Products,
- (7) Orders, (8) Order Details.

Step 1 – Create a SQLBase database called NORTHWND

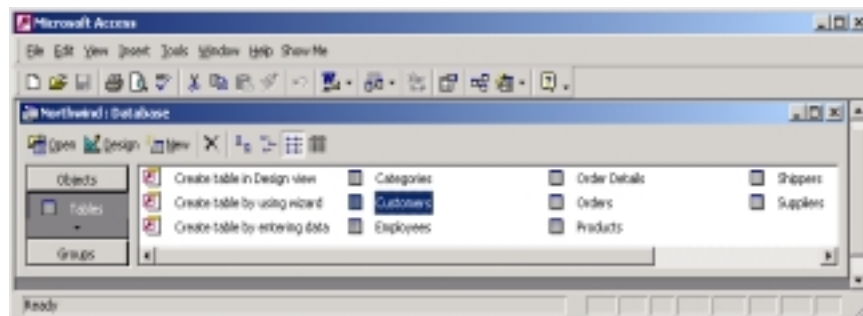
SQLBase databases can be easily created by establishing communications with your SQLBase Server (*SET SERVER servename*), and then using the interactive SQL tool (SQLTalk) and executing the SQL statement *CREATE DATABASE NORTHWND*, or even easier by using the DBA utility called SQLConsole.



Step 2 – Use Microsoft Access to export the 8 tables to SQLBase

To start the Microsoft Access export, follow the steps given below:

- Start MS Access
- Click on File → Open
- Find the Microsoft Sample database (Nwind.mdb), and click the Open button.
- Click on the Tables Tab and select the Customers table by clicking on it.

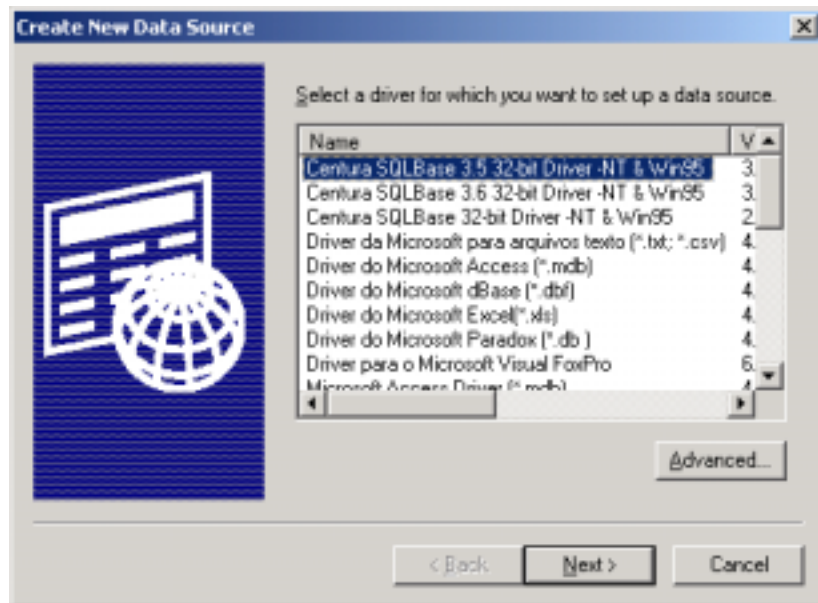


To export the Customers table to SQLBase, follow the steps given below:

- Click on File→ Export menu item
- In the Save As type drop down list, scroll down and select ODBC Databases()
- When the Export popup window appears, change the table name from "Customers" to "CUSTOMERS", then click OK
- **Important:** Make sure to enter the table name in all uppercase i.e. CUSTOMERS, the default in MS Access is uppercase and, if you create the table in uppercase and lowercase, you will have to place double-quotes around the table name each time you refer to it in SQLBase.



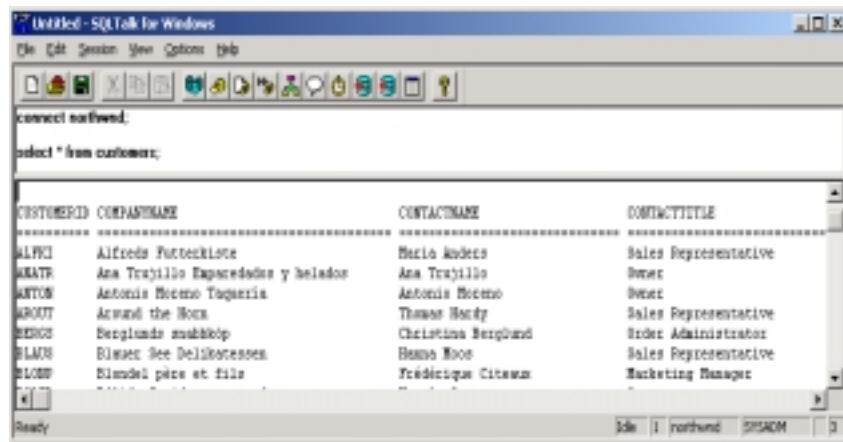
- If you do not already have a SQLBase ODBC Data Source, when the Data Source window appears, click on the New button in order to create a SQLBase Data Source by selecting the appropriate SQLBase ODBC Driver.



- Click on the Next button, enter a name for the data source, and then click on the Finish button.
- A popup window will appear, prompting you to login to the SQLBase database. Enter your servername and database name and click on the Ok button.



- Click on the Ok button to export the CUSTOMERS table to SQLBase.
- Open up SQLTalk and verify that the database table has been loaded into the SQLBase database NORTHWND.



- Repeat the steps used in this approach to export the remaining tables

Method 3: Access to SQLBase Migration Kit

A third option for migrating from Access to SQLBase is to utilize a migration tool provided by Gupta Technologies. The migration tool is designed to interrogate a Microsoft Access database and reproduce its tables in a SQLBase database and migrate the data from the Access database to the corresponding tables in the SQLBase database.

This version of the migration tool is designed to work with Access 2000 and Access 97 databases migrating to SQLBase versions 7.5.x and 7.6.x databases.

Implementation

Launching the "Rightsizing Wizard.exe" starts the migration tool. This application relies on two files: dbcap21.dll and odin.dll.

The user is prompted for a valid ODBC connection to the Access source database. Next, the utility needs to initialize itself using analysis data from the source database. You can perform the analysis at this stage or you can load previously saved analysis data from an earlier session. When the utility is ready to analyze the source you may optionally

specify a file name to store type and name information for customization and use in a later analysis session. The utility then prepares to create the SQLTalk-compatible load script, prompting one for a file name. The Load script will contain all the SQL statements to create a database schema in SQLBase that closely matches the schema from the source database. By default the data from the source database is also migrated by adding it to the Load script file. One then has the option of executing the Load script immediately against an existing database or a newly created database, or saving the execution of the script for later.

Software Requirements

- SQLBase versions 7.5.x or 7.6.x
- MS Access 97 or 2000
- Team Developer 2.1 runtime support
- Migration Tool files (supplied in the Migration Kit)

Known Limitations

The current version of the Migration Kit will not allow one to migrate Access Tables that have a space in their name.

[Download Migration Kit](#)

**The Optimal
Database
Solution is
Transparent To
The Users**

Conclusion

Getting the most value from an application requires an underlying database that enables the highest performance and most features while providing the least number of limitations. With SQLBase, developers can choose access methods and languages that best suit the application requirements. Applications can be delivered on a wide variety of operating systems and hardware platforms. And data can be readily shared among different applications. Migrating to SQLBase offers many benefits including multiple-user availability, security, and ability to manage large amounts of information. Migrating from Access to SQLBase is easy by using any one of the following methods: Use Access to export tables to text files, use Access and ODBC for direct migration, or use the provided migration toolkit.

Appendix

SQLBase Architecture

SQLBase is a client-server database management system. Unlike file-based databases, client-server database engines manage read-write operations to the database. Because of this, client-server databases such as SQLBase can typically handle many more concurrent users and vastly greater amounts of data. SQLBase is designed to meet the most demanding database application requirements for the enterprise, including operational and decision support systems implemented today and in the future.

SQLBase was designed as a true multi-user database server

Powerful databases such as SQLBase are built with core data manipulation architecture. This is extended by special functionality, and augmented with interfaces and tools to access and utilize the full features of the database. Its extreme capabilities such as performance, reliability, manageability, data security and more measure the power of a database. Its ability to handle complex queries and meet developer needs, is a function of how well all these features have been designed to work together.

The architecture of SQLBase is relational database technology, accessed via Structured Query Language (SQL). As its name implies, SQLBase was built to support SQL natively rather than being a database that was converted from another paradigm using a simplified SQL front-end.

Extensions to SQLBase's core include:

- Triggers to automatically propagate changes in one table to others to ensure business rules are met.
- Stored procedures to encapsulate business functions into callable server operations.
- External functions allowing centralized functions outside the database to be called by clients.

These extensions provide a performance boost to applications and allow logic to be placed in the database. The end-result being, that applications and system administrators need to do less. Other advanced extensions, such as its cost-based query optimizer and transaction control capabilities, also enhance performance.

Unlike other embedded databases that have attempted to grow from single-user systems to support multiple users, SQLBase was designed from its inception as a true multi-user database server with a small footprint to fit comfortably into low-end systems. This gives SQLBase the resilience and efficiency that ISAM¹-based systems simply do not possess and is a requirement for business-critical applications, whether for single or multi-user systems.

The key to this robustness is the Server that carries out all database file access and is responsible for transactional integrity, locking, and crash recovery. Programmatic access to data is through the Server, using the client API. Interfaces are provided that enables this API to be used from a wide variety of tools and development environments.

¹ Indexed Sequential Access Method - data supported by Jet tabular data stored in Microsoft Excel workbooks and Microsoft Outlook

Copyright © 2002 Gupta Technologies LLC. Gupta, the Gupta logo, and all Gupta products are licensed or registered trademarks of Gupta Technologies, LLC., All other products are trademarks or registered trademarks of their respective owners. All rights reserved.