

CHAPTER 1

An Introduction to MySQL



In today's interconnected world, it's almost impossible to find a business that doesn't depend on information in some form or another. Be it marketing data, financial movements, or operational statistics, businesses today live or die by their ability to manage, massage, and filter information flow in order to achieve a competitive advantage.

More often than not, all this data finds a home in a business' relational database management system (RDBMS), a software tool that assists in organizing, retrieving, and cross-referencing information. A large number of such systems are currently available, and you've probably already heard of some of them: Oracle, Sybase, Microsoft Access, and PostgreSQL are well-known names. These database systems are powerful, feature-rich software applications, capable of organizing and searching millions of records at high speeds; as such, they're widely used by businesses and government offices, often for mission-critical purposes.

Recently, though, more and more attention has focused on a relatively new entrant in this field: MySQL.

MySQL is a high-performance, multithreaded, multiuser RDBMS built around a client-server architecture. Over the last few years, this fast, robust, and user-friendly database system has become the de facto choice for both business and personal use, notably on account of its advanced suite of data management tools, its friendly licensing policy, and its worldwide support community of users and engineers. This introductory chapter will gently introduce you to the world of MySQL by taking you on a whirlwind tour of MySQL's history, features, and technical architecture.

History

MySQL came into being in 1979, when Michael "Monty" Widenius created a database system named UNIREG for the Swedish company TcX. UNIREG didn't, however, have a Structured Query Language (SQL) interface—something that caused it to fall out of favor with TcX in the mid-1990s. So TcX began looking for alternatives. One of those alternatives was mSQL, a competing DBMS created by David Hughes.

mSQL didn't work for TcX either, however, so Widenius decided to create a new database server customized to his specific requirements. That system, completed and released to a small group in May 1996, became the first version of what is today known as MySQL.

A few months later, MySQL 3.11 saw its first public release as a binary distribution for Solaris. Linux source and binaries followed shortly; an enthusiastic developer community and a friendly, General Public License (GPL)-based licensing policy took care of the rest. Today, MySQL is available for a wide variety of platforms, including Linux, MacOS, and Windows, in both source and binary form.

A few years later, TcX spun off MySQL AB, a private company that had sole ownership of the MySQL server source code and trademark, and was responsible for maintenance, marketing, and further development of the MySQL database server. It was managed by Michael Widenius, David Axmark, and Allan Larsson, supported by both a full-time staff and the active support of a worldwide developer community.

In 2008, MySQL AB was formally acquired by Sun Microsystems, and in 2009, Sun Microsystems was in turn acquired by Oracle, which today owns and develops the MySQL database engine. Although Oracle operates commercially in a number of different markets, the MySQL source code remains available to the community under the GNU General Public License (users can, however, purchase commercial support from MySQL).

Unique Features

MySQL's popularity is due to a particular combination of unique features: speed, reliability, extensibility, and open-source code. The following sections discuss these features in greater detail.

Speed

In an RDBMS, speed—the time it takes to execute a query and return the results to the caller—is everything. By any standards, MySQL is fast, often orders of magnitude faster than its competition. Benchmarks available on the MySQL website show that MySQL outperforms almost every other database currently available, including commercial counterparts like Microsoft SQL Server 2000 and IBM DB2. For example, an eWeek study in February 2002 that compared IBM DB2, Microsoft SQL Server, MySQL, Oracle9i, and Sybase concluded that “MySQL has the best overall performance and that MySQL scalability matches Oracle ... MySQL had the highest throughput, even exceeding the numbers generated by Oracle.”¹

Reliability

Most of the time, high database performance comes at a price: low reliability. MySQL is, however, designed to offer maximum reliability and uptime, and it has been tested and certified for use in high-volume, mission-critical applications. MySQL supports transactions, which ensure data consistency and reduce the risk of data loss, and replication and clustering, two techniques that significantly reduce downtime in the event of a server failure. Finally, MySQL's large user base assists in rapidly locating and resolving bugs and in testing the software in a variety of environments; this proactive approach has resulted in software that is virtually bug-free.

Scalability

MySQL can handle extremely large and complex databases without too much of a performance drop. Tables of several gigabytes containing hundreds of thousands of records are not uncommon, and the MySQL website itself claims to use databases containing 50 million records. A 2005 test by MySQL Test Labs demonstrated that

¹ “A Look at MySQL 5.0 Performance Benchmarks: MySQL Technical White Paper”; <http://www.mysql.com>; May 2006.

What Makes MySQL so Fast?

Part of the reason for MySQL's blazing performance is its fully multithreaded architecture, which allows multiple concurrent accesses to the database. This multithreaded architecture is the core of the MySQL engine, allowing multiple clients to read the same database simultaneously and providing a substantial performance gain. The MySQL code tree is also structured in a modular, multilayered manner, with minimum redundancies and special optimizers for such complex tasks as joins and indexing.

MySQL also includes a *query cache*, which can substantially improve performance by caching the results of common queries and returning this cached data to the caller without having to re-execute the query each time. This is different from competing systems such as Oracle, in that those systems merely cache the execution plan, not the results. However, they still need to execute the query, including all joins, and re-retrieve the query results on every run. MySQL benchmarks claim that this feature improves performance by more than 200 percent, with no special programming required on the part of the user².

It is worth noting that MySQL's designers initially left out many of the features that cause performance degradation on competing systems, including transactions, referential integrity, and stored procedures. These features typically add complexity to the server and result in a performance hit. User requests for these features have, however, resulted in their inclusion in newer versions of the product.

"MySQL shows near-linear scalability in a multi-CPU environment,"³ with performance increasing in proportion to the number of CPUs added to the system. This ability to scale with demand has made MySQL popular with businesses like Eli Lilly, Alstom, Dun & Bradstreet, Epson, and the *New York Times*; high-volume websites such as Google, Facebook, and Slashdot; and government organizations such as NASA, the U.S. Census Bureau, and the Swedish National Police.

Ease of Use

MySQL is so easy to use that even a novice can pick up the basics in a few hours, and the software is well supported by a detailed manual, a large number of free online tutorials, a knowledgeable developer community, and a fair number of books. While most interaction with MySQL takes place through a command-line interface, a number of graphical tools, both browser-based and otherwise, are also available to simplify the task of managing and administering the MySQL database server. Finally, unlike its proprietary counterparts, which have literally hundreds of adjustable parameters,

² The MySQL manual; <http://dev.mysql.com/doc/refman/5.0/en/query-cache.html>

³ The MySQL website; <http://www.mysql.com/why-mysql/white-papers/performance.php>

MySQL is fairly easy to tune and optimize for even the most demanding applications. For commercial environments, MySQL is fully supported in terms of professional MySQL training, consultancy, and technical support.

Portability and Standards Compliance

MySQL supports most of the important features of the ANSI (American National Standards Institute) SQL standard, and often extends the ANSI standard with custom extensions, functions, and data types designed to improve portability and provide users with enhanced functionality. MySQL is also available for both UNIX and non-UNIX operating systems, including Linux; Solaris; FreeBSD; OS/2; MacOS; and Windows 95, 98, Me, 2000, XP, NT, and Vista; and it runs on a range of architectures, including Intel x86, Alpha, SPARC, PowerPC, and IA64.

Multiuser Support

MySQL is a full multiuser system, which means that multiple clients can access and use one (or more) MySQL database(s) simultaneously; this is of particular significance during development of web-based applications, which are required to support simultaneous connections by multiple remote clients. MySQL also includes a powerful and flexible privilege system that allows administrators to protect access to sensitive data using a combination of user- and host-based authentication schemes.

Internationalization

As a program that is used by millions of users in countries across the globe, it would be unusual indeed if MySQL did not include support for various languages and character sets. MySQL offers full Unicode support, as well as full support for most important character sets (including Latin, Chinese, and European character sets). Character sets are taken into account when sorting, comparing, and saving data.

Wide Application Support

MySQL exposes application programming interfaces (APIs) to many programming languages, thereby making it possible to write database-driven applications in the language of your choice. Currently, MySQL provides hooks to C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl, and connectors are available for JDBC, ODBC, and .NET applications.

Open-Source Code

The MySQL source code is freely available under the terms of the GNU General Public License—a key benefit, since it allows users to download and modify the application to meet their specific needs. This unique licensing policy has fuelled MySQL's popularity, creating an active and enthusiastic global community of MySQL developers and users. This community plays an active role in keeping MySQL ahead of its competition, both by crash-testing the software for reliability on millions of installations worldwide and extending the engine to stay abreast of the latest technologies.

High-volume, well-informed mailing lists and user groups assist in the rapid resolution of questions and problems, and a global network of committed MySQL users and developers provides knowledgeable advice, bug fixes, and third-party utilities. All of this has paid off: A code inspection study by Reasoning, Inc. concluded that the code quality of MySQL was six times better than that of comparable proprietary code⁴.

NOTE *It is worth noting that if your MySQL-powered application is not licensed under the GPL or other MySQL-approved open-source license and you intend to redistribute it (whether internally or externally), you are required to purchase a commercial license for this use. Oracle earns revenue both from the sale of these licenses and by providing support, training, and consultation services for the MySQL database server.*

Product Family

In addition to the core MySQL database server, Oracle makes available a number of MySQL-related products and tools. This section introduces you to some of the other members of the MySQL product family.

MySQL Server

This core product consists of a high-performance database server, which is the main software engine responsible for creating and managing databases, executing queries and returning query results, and maintaining security. This core product also includes a number of client-side tools, such as a command-line SQL client; tools to manage user permissions; and utilities to import, export, copy, and repair databases.

MySQL Cluster

MySQL Cluster is a version of the MySQL database server that supports “clustering,” a technology that allows data to be transparently distributed across two or more physical servers to increase redundancy. This clustering technology plays an important role in high-availability applications, as it ensures continuous data availability even if one of the nodes in the cluster fails. At the time of this writing, MySQL Cluster supports up to 255 nodes in a single cluster and uses synchronous replication to copy data between nodes.

MySQL Proxy

MySQL Proxy is a proxy server that serves as a “gatekeeper” between the MySQL database server and connecting clients. It includes the ability to intercept and rewrite queries, modify result sets, implement query queues, analyze query traffic for reporting purposes, and perform load balancing tasks.

⁴ The MySQL website; <http://www.mysql.com/why-mysql/quality>

Are there Different Versions of the MySQL Database Server?

MySQL's core database server comes in two flavors: Community and Enterprise. The Community server is "free": Users can download and use it at no cost under the terms of the GNU GPL, but by the same token, are required to perform all maintenance and administrative tasks themselves, with no support from the MySQL development team. For companies and individuals looking for a greater level of support, the Enterprise server is a commercial offering that provides regular updates and bug fixes, consultancy services and advice from MySQL engineers, and proprietary database-monitoring software in return for a subscription fee.

MySQL Administrator

MySQL Administrator is an all-in-one control center for a MySQL database server, allowing database administrators to track server status in real time. It includes visual tools for user administration, database backup and restore, and log analysis, as well as server fine-tuning.

MySQL Query Browser

MySQL Query Browser is a visual tool for graphically constructing queries and viewing the results. It includes tools to manage database connections, databases, and tables, as well as a debugger (with breakpoint support) to assist in optimizing and troubleshooting complex queries.

MySQL Workbench

MySQL Workbench is a visual design tool that enables database administrators and developers to graphically design and validate data models, generate database schema code, and manage changes to database schemas. It also includes the ability to visually compare and synchronize two versions of a database and create import/export scripts to transfer data from one system to another.

MySQL Migration Toolkit

The MySQL Migration Toolkit is a graphical, wizard-driven tool to port databases from other RDBMS products to MySQL. It includes support for Oracle, Microsoft SQL Server, and Microsoft Access, and provides automated tools to remap and rebuild table schemas; copy records; and transfer indexes, views, triggers, and stored procedures.

MySQL Embedded Server

MySQL Embedded Server is a low-footprint version of the MySQL database server that is intended specifically for use in embedded applications, such as networking equipment, diagnostic tools, or point-of-sale (POS) systems. This embedded database also includes a number of useful administrative features: automatic space expansion, auto-restart, and dynamic reconfiguration.

MySQL Drivers and Connectors

MySQL provides drivers and connectors for many different programming languages, thereby making it possible to build database-driven applications using any one of several different development toolkits. Currently, MySQL provides drivers and connectors for C, C++, Java, Perl, PHP, Python, Ruby, JDBC, ODBC, and .NET applications.

Technical Architecture

MySQL is based on a tiered architecture, consisting of both primary subsystems and support components that interact with each other to read, parse, and execute queries, and to cache and return query results.

Subsystems

There are three primary subsystems within the MySQL architecture, as discussed in the following sections.

Memory and Connection Management

This subsystem manages user connections, via modules for network connection management with clients, and synchronizes competing tasks and processes, via modules for multithreading, thread locking, and performing thread-safe operations. It also handles all memory management issues between requests for data by the query subsystem and the data storage subsystem.

Query Parsing and Execution

Query parsing and execution is handled by two interrelated components: the syntax parser and the query optimizer. The syntax parser decomposes the SQL commands it receives from calling programs into a form that can be understood by the MySQL engine. It also checks the objects being referenced to ensure that the privilege level of the calling program allows it to use them. The query optimizer then prepares the most efficient plan for query execution, making decisions about table-versus-index scans, join methods, and range optimization, and using a bottom-up methodology to detect the optimal execution plan.

Data Storage

The data storage subsystem interfaces with the operating system (OS) to write to disk all of the data in the user tables, indexes, and logs, as well as the internal system data. MySQL 5.1 also introduced a new pluggable architecture, which allows developers to create new table storage mechanisms and “plug them in” to the server at run-time. This pluggable architecture also creates a level of abstraction between the data storage subsystem and the rest of the MySQL server, making it possible for developers to add new data storage engines that interact with the other MySQL subsystems through a standard API.

Connectivity

MySQL is designed on the assumption that the vast majority of its applications will be running on a TCP/IP (Transmission Control Protocol/Internet Protocol) network. This is a fairly good assumption, given that TCP/IP is not only highly robust and secure, but is also common to UNIX, Windows, OS/2, and almost any other serious operating system you'll likely encounter. When the client and the server are on the same UNIX machine, MySQL uses TCP/IP with UNIX sockets, which operate in the UNIX domain; that is, they are generally used between processes on the same UNIX system, as opposed to Internet sockets, which operate between networks.

Standards Compliance

The Structured Query Language (SQL) is an open standard that has been maintained by the American National Standards Institute (ANSI) since 1986. Although it's true that the implementation of this standard does differ in varying degrees from vendor to vendor, it's fair to say that SQL is today one of the most widely used cross-vendor languages. As with other implementations, such as SQL Server's T-SQL (Transact-SQL) and Oracle's SQL, MySQL has its own variations of the SQL standard that add power beyond what is available within the standard. Beginning with v5.1, MySQL also includes support for data import and export using Extensible Markup Language (XML), a widely accepted, vendor-neutral format for data markup and sharing.

Transactions

In the SQL context, a transaction consists of one or more SQL statements that operate as a single unit. Each SQL statement in such a unit is dependent on the others, and the unit as a whole is indivisible. If one statement in the unit does not complete successfully, the entire unit will be rolled back, and all the affected data will be returned to the state it was in before the transaction was started. Thus, a transaction is said to be successful only if all the individual statements within it are executed successfully.

The MySQL transaction system fully satisfies the ACID tests for transaction safety via its InnoDB and BDB table types (older table types, such as the MyISAM type, do not support transactions).

- *Atomicity* is handled by storing the results of transactional statements (the modified rows) in a memory buffer and writing these results to disk and to the binary log from the buffer only once the transaction is committed. This ensures that the statements in a transaction operate as an indivisible unit and their effects are seen either collectively or not at all.
- *Consistency* is primarily handled by MySQL's logging mechanisms, which record all changes to the database and provide an audit trail for transaction recovery. In addition to the logging process, MySQL provides locking mechanisms that ensure that all of the tables, rows, and indexes that make up the transaction are locked by the initiating process long enough to either commit the transaction or roll it back.

- Server-side semaphore variables and locking mechanisms act as traffic managers to help programs manage their own *isolation* mechanisms. MySQL's BDB table handler, for example, uses page-level locking to safely handle multiple simultaneous transactions, while the InnoDB table handler uses a more fine-grained row-level locking.
- MySQL implements *durability* by maintaining a binary transaction log file that tracks changes to the system during the course of a transaction. In the event of a hardware failure or abrupt system shutdown, recovering lost data is a relatively straightforward task by using the last backup in combination with the log when the system restarts.

Because transactional tables incur some performance overhead, it's also possible to specify whether to use transactions on a per-table basis.

Query Caching

If a query returns a given set of records, repeating the same query should return the same set of records, unless the underlying data has somehow changed. As obvious as this sounds, few of the other major RDBMS vendors provide features that take advantage of this principle. Other database products are efficient in storing optimized access plans that detail the process by which data is retrieved; such plans allow queries similar to those that have been issued previously to bypass the process of analyzing indexes yet again to get to the data.

Result-set caching takes this principle a step further by storing the result sets themselves in memory, thus circumventing the need to search the database at all. The data from a query is simply placed in a cache, and when a similar query is issued, this data is returned as if in response to the query that created it in the first place.

The MySQL engine uses an extremely efficient result set-caching mechanism, known as the Query Cache, that dramatically enhances response times for queries that are called upon to retrieve the exact same data as a previous query. This mechanism is so efficient that a major computing publication declared MySQL queries to be faster than those of Oracle and SQL Server (which are both known for their speed). If implemented properly, decision support systems using MySQL with canned reports or data-driven web pages can provide response speeds far beyond those that would be expected without the Query Cache.

Extensibility

In keeping with its open-source roots, MySQL makes the original source code available as part of the distribution, which permits developers to add new functions and features that are compiled into the engine as part of the core product. MySQL also allows separate C and C++ libraries to be loaded in the same memory space as the engine when MySQL starts up.

MySQL also allows developers to add new functions at run-time through a special user-defined function interface. User-defined functions are created initially as special C/C++ libraries and are then added and removed dynamically by means of the `CREATE FUNCTION` and `DROP FUNCTION` statements.

Symmetric Multiprocessing Support

To take advantage of multiprocessor architecture, MySQL is built using a multithreaded design, which allows threads to be allocated between processors to achieve a higher degree of parallelism. This is important to know not only for the database administrator, who needs to understand how MySQL takes best advantage of processing power, but also for developers, who can extend MySQL with custom functions. All custom functions must be *thread-safe*—that is, that they must not interfere with the workings of other threads in the same process as MySQL.

MySQL makes use of various thread packages, depending on the platform. POSIX threads are used on most UNIX variants, such as FreeBSD and Solaris. LinuxThreads are used for Linux distributions. For efficiency reasons, Windows threads are used on the Windows platform, but the code that handles them is designed to simulate POSIX threads.

Because MySQL is a threaded application, it is able to let the operating system take over the task of coordinating the allocation of threads to balance the workload across multiple processors. MySQL uses a global connection thread to handle all connection requests and creates a new dedicated thread to handle authentication and SQL query processing for each connection. In addition, in replication, master-host synchronization is handled by separate threads.

Of course, another way to take advantage of multiprocessing is to run multiple instances of MySQL on the same machine, thereby spawning a separate process for each instance. This approach is especially practical for hosting companies and even for internal hosting within corporate environments. By running multiple instances of MySQL on the same computer, you can easily accommodate multiple user bases that need different configuration options.

Security

The process of accessing a MySQL database can be broken down into two tasks: connecting to the MySQL server itself and accessing individual objects, such as tables or columns, in a database. MySQL has built-in security to verify user credentials at both stages.

- MySQL manages user authentication through user tables, which check not only that a user has logged on correctly with the proper username and password, but also that the connection is originating from an authorized TCP/IP address.
- Once a user is connected, a system administrator can bestow user-specific privileges on objects and on the actions that can be taken in MySQL. For example, you might allow *fred@thiscompany.com* to perform only `SELECT` queries against an inventory table, while allowing *anna@thatcompany.net* to run `INSERT`, `UPDATE`, and `DELETE` statements against the same table.

The actual data that travels over a network, such as query results, isn't encrypted and is, therefore, open to viewing by a hacker. To secure your data, you can use one of the Secure Shell (SSH) protocols; you'll need to install it on both the client applications and the operating system you're using. If you're using MySQL 4.0 or later, you can also use the Secure Socket Layer (SSL) encryption protocol, which can be configured to work from within MySQL, making it safe for use over the Internet or other public network infrastructures.

Application Programming Interfaces

For application developers, MySQL provides a client library that is written in the C programming language and a set of APIs that provide an understandable set of rules by which host languages can connect to MySQL and send commands. Using an API protects client programs from any underlying changes in MySQL that could affect connectivity.

Currently, MySQL provides hooks to C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl, and connectors are also available for JDBC, ODBC, and .NET applications.

Applications

MySQL's technical architecture, built as it is around the three tenets of performance, reliability, and ease of use, have made the product extremely popular, both on and off the Web. According to the MySQL website, more than 100 million copies of MySQL have been downloaded and distributed to date, and 50,000 more are added to that total every day. MySQL software today powers a variety of applications, including Internet websites, e-commerce applications, search engines, data warehouses, embedded applications, high-volume content portals, and mission-critical software systems.

Web Applications

It should come as no surprise that MySQL's primary applications today lie in the arena of the Web. As websites and web-based distributed applications grow ever more complex, it becomes more and more important that data be managed efficiently to improve transactional efficiency, reduce response time, and enhance the overall user experience. Consequently, a pressing need exists for a data management solution that is fast, stable, and secure—one that can be deployed and used with minimal fuss and that provides solid underpinnings for future development.

MySQL fits the bill for a number of reasons. Its proven track record generates confidence in its reliability and longevity; its open-source roots ensure rapid bug fixes and a continued cycle of enhancements (not to mention a lower overall cost); its portability and support for various programming languages and technologies make it suitable for a wide variety of applications; and its low cost/high performance value proposition makes it attractive to everyone from home users to small- and medium-sized businesses and government organizations. For these reasons and more, MySQL is a key component of modern web applications, particularly those built on the popular LAMP stack.

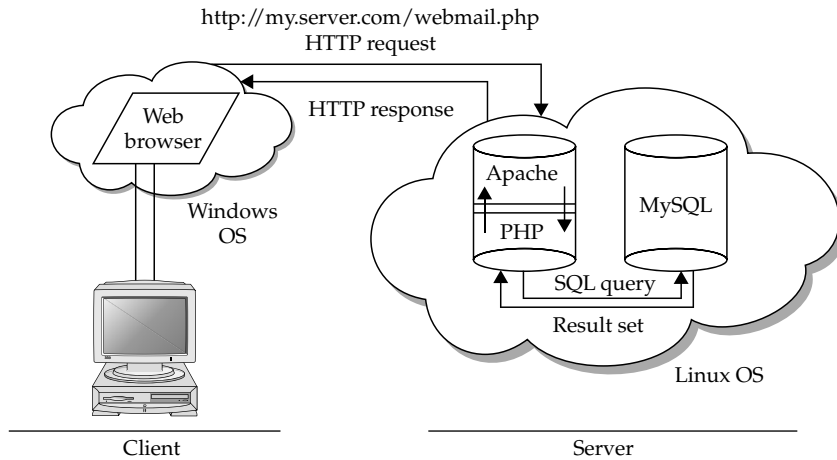


FIGURE 1-1 The LAMP development framework

Wondering what a LAMP stack is? Well, the term refers to a set of open-source software components that are commonly used in conjunction with each other to build web-based applications. These components are

- A base operating system and server environment (Linux)
- A web server (**A**pache) to intercept HTTP requests and either serve them directly or pass them on to the PHP interpreter for execution
- A database engine (**M**ySQL) that holds application data, accepts connections from the application layer, and modifies or retrieves data from the database
- A programming toolkit (**P**HP, **P**erl, or **P**ython) that parses and executes program code, processes database results, and returns results to the client

Figure 1-1 illustrates the four elements of the LAMP framework in action.

Data Warehouses

As the opening paragraph in this chapter notes, businesses are becoming more and more intelligent in how they store, filter, and use information. Data warehouses are a key source of this business intelligence. Typically, data in a data warehouse is gathered from an enterprise's internal information systems, linked, and stored for long periods of time. In its simplest form, this data merely provides a record of past events; however, it can also be "mined" to detect patterns, which serve as input into an organization's decision-making. Speed of data retrieval is thus one crucial component of a data warehouse; long-term reliability is another.

MySQL scores high on both counts. It supports engine-level data integrity through the use of primary key and foreign key constraints. An extremely efficient query-caching mechanism dramatically enhances response times for queries that are called

upon to retrieve the exact same data as a previous query. MySQL InnoDB table format uses asynchronous I/O and a sequential read-ahead buffer to improve data retrieval speed, and a “buddy algorithm” and Oracle-type tablespaces for optimized file and memory management. For data storage reliability, MySQL supports replication, a data distribution mechanism that places copies of tables and databases in remote locations to reduce downtime in case of a server failure.

Business Applications

As a reliable, feature-rich database server, MySQL also has applications in business, education, science, and engineering—a fact amply demonstrated by its customer list, which includes such names as Motorola, Sony, NASA, HP, Xerox, and Silicon Graphics. Whether it is small, embedded applications or high-availability data processing systems, MySQL offers the scalability and performance needed to achieve business objectives. The MySQL website states that “MySQL scales to deal with billions of rows and terabytes of data, making it suitable for a wide range of transactional and analytic applications.”

To take advantage of multiprocessor architecture, MySQL is built using a multithreaded design, which allows threads to be allocated between processors to achieve a higher degree of parallelism. MySQL’s clustering technology allows data to be distributed across multiple nodes to achieve greater redundancy, while its fully ACID-compliant transactional engine provides a high degree of safety from undetected data loss. At the other end of the scale, MySQL’s embedded server library has a 1-MB memory/4-MB disk space footprint and provides a multithreaded, cross-platform data storage engine for use in kiosk-style applications or appliances. Finally, MySQL uses a two-tier privilege system (at the connection level and at the individual object level) to ensure the security and integrity of its data, and supports the SSL encryption protocol for client/server communication.

Summary

This chapter provided a gentle introduction to the world of MySQL, discussing the history and evolution of the product and highlighting some of its unique features and advantages vis-à-vis competing alternatives. It explained the various components of the MySQL architecture, discussed some of the key technical features of the MySQL engine, and illustrated how they interact with each other. Finally, it discussed some of MySQL’s real-world applications, notably with regard to web application development, data warehousing, and industrial applications.

If you’d like to learn more about the topics discussed in this chapter, consider visiting the following links:

- A more detailed history of MySQL at <http://www.linuxjournal.com/article.php?sid=3609>
- The MySQL development roadmap at <http://dev.mysql.com/doc/refman/5.1/en/roadmap.html>

- The MySQL manual at <http://dev.mysql.com/doc>
- An overview of MySQL's technical architecture at <http://dev.mysql.com/doc/refman/5.1/en/pluggable-storage-overview.html>
- MySQL case studies at <http://www.mysql.com/why-mysql/case-studies>
- MySQL customer listings at <http://www.mysql.com/customers>
- MySQL market share and usage statistics at <http://www.mysql.com/why-mysql/marketshare>
- MySQL performance benchmarks at <http://www.eweek.com/article2/0,3959,293,00.asp> and <http://www.mysql.com/why-mysql/benchmarks>
- Awards won by MySQL at <http://www.mysql.com/why-mysql/awards>