



MySQL 5.0 vs. Microsoft SQL Server 2005

White Paper

Abstract

This paper describes the differences between MySQL and Microsoft SQL Server 2000.

MySQL 5.0 vs. Microsoft SQL Server 2005

Revised by Chris Rindal
QA Technician, Tometa Software, Inc.

Tometa creates custom software for you

Tometa Software designs and develops robust software solutions for virtually all industries including in-house (vertical market) and retail software, some of which is on the shelves at your local software store. We focus our unique combination of creative, technical, and problem-solving skills on meeting our client's objectives. Because of our clarity of purpose, commitment to process, and broad professional skill sets, we are able to provide our clients with world-class solutions that are functionally superior and fully aligned with our client's strategic focus.

Balancing development speed, quality and cost is what we are all about. Tometa combines agile development practices with fixed pricing, so you know what the cost, end product, and delivery time table look like—up front. If we underestimate the effort, we complete the overrun on our dime. Simple as that. That's why large enterprise firms like Alcoa and NASDAQ choose Tometa.

Tometa's agile development expertise and low-overhead US location keep our prices comparable to offshore vendors – without offshore challenges. Using a fixed pricing model, we provide upfront visibility into a project's ultimate costs, end product and delivery schedule. Our clients like knowing that we have “skin in the game” – a fixed price that aligns our goals with yours, incenting us to get the job done right and fast.

Lastly, as a Microsoft Certified Gold Partner, Tometa Software, can customize its products or create custom web, client/server, and traditional applications. With programming experience in C#, C++, Visual Basic, CGI, HTML, RPG, Delphi, Java and many others; Tometa Software is uniquely positioned to meet your needs as a development firm.

[Check us out today](#)

Database engines are a crucial fixture for businesses today. There is no shortage of both commercial and open source database engines to choose from. Microsoft SQL Server 2005 is Microsoft's next-generation data management solution that claims to deliver secure and scalable applications while making them easy to deploy and manage. MySQL has long been the DBMS of choice in the open source community. The recent release of MySQL 5.0 has seen major changes in both features and performance to bring the database system into enterprise-level standards.

This paper aims to give the low-down on features most desirable to database developers and compare both database management systems in light of these features.

Topics in this Paper:

Features

The Open Source vs. Commercial Database Paradigm

Performance

Replication

Security

Recovery

Features

The most obvious difference between the two products is in philosophy. SQL server is essentially a proprietary storage engine. Once you purchase the product, you are only limited to the Sybase-derived engine. By contrast, MySQL is an open storage engine offering multiple choices: InnoDB, BerkleyDB, MyISAM and Heap amongst other supported engines.

The second marked difference between the two database systems is in the technical features and specifications implemented. SQL Server is a fully-fledged database system developed specifically for large enterprise databases. All advanced features of a relational database are fully implemented. MySQL, on the other hand, has only come out of edge in the "relational" front, with recent support for foreign keys.

The latest release of MySQL, the 5.X offering, has rounded up on features that lagged commercial equivalents such as SQL Server. There is now full support for cursors, complete views and stored procedures according to the SQL 2003 syntax. Other features that were a major differentiator between MySQL and SQL Server are now part of the 5.X release. Triggers, stored procedures and foreign keys are fully implemented.

But is MySQL 5.0 really up to industry-level database standards? The features outlined above have only been implemented in the latest release and are yet to fully stabilise. They are yet to be rationalised across the different databases in the MySQL suite of products – InnoDB, MyISAM, MaxDB and the new data clusters. MySQL is still carrying four distinct database

architectures and it proves very challenging to fully implement replication, parallel processing, journaling and recovery across different databases.

SQL Server continues to have the edge, as the advanced features list has long stabilised. The latest release of SQL Server 2005 provides the necessary technological underpinnings to keep it in the higher-end of database systems. There is now a far greater integration with Microsoft's .NET Framework, a development environment that greatly facilitates coding without the need to learn advanced features of SQL. It is also tightly integrated with Visual Studio .NET. This will provide better support for XML, querying multi-dimensional data in the SQL server and a set of advanced reporting controls. Finally, XML is now a native data type within XML. This enables a DBA to modify an XML document within the DBMS environment, query the document and validate it against an XML schema.

The Open Source versus Commercial License Paradigm

Another difference between the two database engines is licensing costs. Both databases have a two-tiered licensing scheme, but have little else in common.

The first licensing scheme is essentially free. SQL Server provides a free license for "development use only". What this means is that the database system cannot be deployed in a commercial environment. MySQL, on the other hand, is free to use under any environment, provided one abides by GPL license rules.

This brings us to the second-tier of licensing. For use in a commercial environment, one would need to purchase the SQL Standard Edition license. It costs a whopping \$1,400, a substantial investment for a small business. However, it is a fully-fledged relational database system complete with all features needed to develop and deploy enterprise databases. This goes a long way towards justifying the hefty price tag.

MySQL also provides licensing schemes to circumvent some of the restrictions of the GPL license. This is especially important for companies that deal with proprietary information. These commercial licenses are piloted by MySQL AB, the company behind the development of MySQL, and cost a very affordable \$400. Non-profit organisations and educational establishments are exempt from this fee.

Performance

In terms of performance, MySQL fairs better than SQL on a variety of platforms thanks to the default table format of its MyISAM database. They are compact on disk and use less memory and CPU cycles. While the database system performs well on Windows, it is better suited for UNIX and UNIX-like systems. The performance can further be tuned on 64-bit processors (such as SPARC stations) because of the internal use of 64 integers in the database. The latest release of MySQL 5.0 has seen further improvements in engine

performance, through compact mode support. Engines such as InnoDB and NDB Cluster uses 20% less space than it required in previous versions.

For additional non-default MySQL features, there is an increased demand on resource usage and this has obviously an effect on performance. For instance, alternative table formats on MyISAM or transactions on Berkeley DB will require additional memory usage. These features will, however, offer additional functionality.

For SQL Server, the full-set of powerful features that surpasses that of most competitors has a negative effect on performance. It's true that many of these features are geared towards performance tuning, but overall the system is more complex, places additional requirements on memory and disk storage. This results in a poorer performance compared with MySQL. The performance will benefit greatly with RAID and a dedicated hard drive for the data store.

Replication

Both Database systems are scalable and support replication to a different degree of complexity.

Replication on MySQL is easy because all SQL statements that change data are kept in a binary log. Because of the binary nature of the records, data can be replicated easily and quickly to one or more slave machines. This also means that data remains intact and replication takes place even when the server goes down. On the scalability front, MySQL scales easily into large, query-heavy databases.

Unlike MySQL one-way replication, SQL Server offers replication in a number of models: snapshot, transactional and merge. A snapshot application is a simple snapshot of the entire replicated database. It is a time consuming process but can be useful for databases that rarely change or as a way to establish a baseline for replication between systems. A transactional replication is a more flexible solution for databases that regularly change. The database is monitored for any changes by a replication agent monitor. When changes do take place, they are transmitted to the subscribers. Finally, merge replication allows simultaneous changes to the database by both the publisher and subscribers. Changes can be made without an active network connection, and any conflicting changes are resolved through a predefined conflict resolution algorithm.

However, increased replication support comes at the expense of a greater degree of complexity. This is due to SQL's complex transaction and record locking mechanism, cursor manipulation and synchronisation of dynamic data replication. If you're skilled in these elaborate mechanisms, then replication and migration shouldn't be an issue.

Security

Security remains a major concern for most businesses and a compelling consideration in choosing a database system.

Both DBMS support security at the base level. MySQL is limited to supporting basic security at the table level, via the SQL command. By contrast, SQL server fully supports security at the column level.

Another important consideration is security certificates - the verification of the database security by a third party. SQL Server has been certified as C-2 compliant, which means the database system has adequate security for government applications. MySQL has no such certification.

Moving on to more advanced features of protecting data on the database, the SQL Server 2005 have implemented more advanced authentication and authorisation features. The database supports native encryption capabilities, obfuscating the DBA from writing user-defined functions using column encryption APIs. The encryption mechanism is based on a combination of third-party certificates, symmetric keys and asymmetric keys. You can specify asymmetric keys for increased security or symmetric keys for better performance. A DBA has also the choice of specifying his own user-defined security functions through the encryption facility implemented in the .NET Framework.

Recovery

SQL Server is more failsafe and less prone to data corruption. SQL has a robust checkpoint mechanism whereby the data passes from the keyboard to the hard drive before showing in the monitor. Even if the databases shut down unexpectedly without warning, the data can be recovered.

New features in the SQL 2005 release provide enhanced mechanisms to manage data protection and rapid restoration. Mirrored backups allow you to create multiple copies of the backup file. These backups have identical content, so you can always mix the files in case one of the sets becomes corrupt.

Copy only backups enable you to make a copy of the database without interrupting the sequence of other backup files. This copy can be used to restore your database, instead of going through the full backup and translation log. You can also save time by using partial backups for all filegroups, except those marked as read-only.

MySQL falls short in recovery with its default MyISAM mechanism. The UPS assumes uninterrupted data, and in the event of an unexpected shutdown your data can be lost and the data store corrupted.

Concluding Thoughts

From a database developer's perspective, choosing between a MySQL and SQL Server DBMS is a matter of the scale of the database application. For enterprise-level applications, SQL Server wins hands down. It has advanced set of SQL features, superior replication, clustering, security and management tools.

For lower-tier database applications, MySQL can offer the core functionality you require at a very low cost. Some might argue that the latest offering from MySQL has made the open source database system enterprise "worthy", but this remains to be seen. The advanced functionalities implemented are yet to stabilise and be rationalised across the database engine. What's more, Microsoft has upped the ante with even more advanced features of its own. It's up to MySQL to rise up to the challenge, but at this point in time MySQL is nowhere near the competitive enterprise field of the more established SQL Server 2005.