Building Distributed
Microsoft SQL Server 2000 Database Applications on Dell PowerEdge 6650 Servers

Building out data centers using many smaller servers based on four or fewer CPUs rather than a few large servers based on eight or more CPUs can offer cost, redundancy, and ease-of-expansion advantages. To demonstrate how Microsoft SQL Server software can benefit from running on multiple servers, a team of Dell engineers built a 100 GB online store application to run on two Dell PowerEdge 6650 servers, each with four processors—one system received orders and the other processed financial reports against the order data. This article describes the online store application and the data replication features of SQL Server that connected the database instances on each server.

Microsoft SQL Server relational database management software is widely deployed in worldwide enterprises for online transaction processing (OLTP), online analytical processing, and data mining. Highly scalable, reliable, easy to deploy, and self-tuning, SQL Server is used for demanding, mission-critical applications. Whether used as the database engine behind such Microsoft products as Commerce Server and Content Management Server, or accessed by custom applications created with the Microsoft Visual Studio.NET development system, SQL Server can provide a robust environment to manage corporate data.

Dell best practices advocate building enterprise data centers by sharing the workload among farms of small servers that have four or fewer CPUs, rather than concentrating the workload on larger servers with eight or more CPUs. Scaling out the data center using smaller servers can offer cost, fault-tolerance, and ease-of-expansion advantages over larger, scaled-up servers. SQL Server applications can be designed to run on multiple servers through the use of high-speed data replication technology.1

1 For more information on Microsoft SQL Server replication technology, see “SQL Server 2000 Replication Overview” at http://www.microsoft.com/sql/evaluation/features/replication.asp.
In December 2003, a team of Dell engineers used SQL Server replication technology to build a 100 GB online DVD store application that ran across two Dell™ PowerEdge™ 6650 servers, each with four processors. One PowerEdge 6650 server, which was driven by a Web application (not modeled in this test), collected incoming orders. A second PowerEdge 6650 server generated financial reports from the order data. On a scheduled basis, the updated order data from the first server was replicated to the second server. By running the order-entry and report workloads on separate servers, with each server tuned for its particular workload, Dell engineers achieved performance scaling and redundancy. This article describes the SQL Server test performed by the Dell team, including factors such as the database application, the replication method used to update the second server, and the performance impact of the replication.

Building the database application
To demonstrate SQL Server running on multiple instances, a 100 GB online DVD store was implemented as two replicated SQL Server databases. One SQL Server instance handled the entry of new orders and replicated changes on a scheduled basis to the second SQL Server instance, which was used for generating financial reports. The DVD store (DS) database comprised a set of data tables organized according to a certain schema, along with a set of stored procedures that did the actual work of managing the data in the database as orders were entered and reports were requested. The database back end was designed to be driven from a Web-based middle tier, but because the focus of the Dell test was on the database servers, the back-end stored procedures were driven directly by custom C programs to simulate a Web-based middle tier.

The database schema
The DVD store was composed of four main tables and one additional table (see Figure 1). The Customers table was prepopulated with 200 million customers, including 100 million U.S. customers and 100 million customers from the rest of the world. The Orders table was prepopulated with 10 million orders per month, starting in January 2003 and ending in September 2003. The Orderlines table was prepopulated with an average of five items per order. The Products table contained 1 million DVD titles. In addition, the Categories table listed the 16 DVD categories. For the full DVD store database build script, visit Dell Power Solutions online at http://www.dell.com/magazines_extras.

The stored procedures
The DVD store database was managed through seven stored procedures. The first two were used during the login phase. For returning customers, the Login procedure retrieved the customer’s information, in particular the CUSTOMERID. For new customers, the New_customer procedure created a new row in the Customers table containing the customer’s data.

Following the login phase, the customer might search for a DVD by category, actor, or title. These database functions were implemented by the Browse_by_category, Browse_by_actor, and Browse_by_title procedures, respectively. Finally, after the customer completed the selections, the Purchase procedure was called to complete the transaction. Additionally, the Rollup_by_category procedure calculated total sales by DVD category for the previous month, quarter, and half-year periods. For the stored procedures, visit Dell Power Solutions online at http://www.dell.com/magazines_extras.

The driver applications
Separate multithreaded driver programs were written to model the OLTP order-entry workload and the report request workload.

Online transaction processing. Each thread of the OLTP driver application connected to the database and made a series of stored procedure calls that simulated customers logging in, browsing, and purchasing. Because no customer think times or key times were factored in, the database connections remained full, simulating what happens in a real multitiered application—a few connections are pooled and shared among Web servers that may be handling thousands of simultaneous customers. In this way, Dell engineers achieved a realistic simulation of database activity without needing to model thousands of customers.

Each thread of the OLTP driver modeled a series of customers going through the entire sequence of logging in, browsing the catalog several ways, and finally purchasing the selected items. Each sequence completed by a customer counted as a single order. The driver measured order rates and the average response time to complete each order. Several tunable parameters were

---

2 This is just one example of spreading a SQL Server workload across multiple servers. For an overview of different scale-out choices, see The Definitive Guide to Scaling Out SQL Server by Don Jones at http://www.dell.com/sql/ebook.
used to control the application, as described in Figure 2.

**Reports.** The report request driver program was similar to the OLTP driver in that each thread connected to the database and started making stored procedure calls. Each thread made repeated calls to the Rollup_by_category stored procedure, which calculated total sales by DVD category for the previous month, quarter, and half year, until every report for all 16 categories was completed. In each test, eight simultaneous reports were run.

**Replicating the SQL Server databases**

Microsoft SQL Server 2000 provides several configuration options when setting up replication, but they all use the same basic model of publishers, distributors, and subscribers. The test environment simulated the requirement to run reports and accept new transactions against the same database. Replication allowed two synchronized copies of the same data. Through the use of two PowerEdge systems running SQL Server, connected by a replication mechanism, all new transactions occurred in one instance of SQL Server and the reports were run on the other instance.

**Replication types and terminology**

SQL Server uses the model of publishers, distributors, and subscribers to replicate data, and it provides a set of wizards to help administrators set up each of these components. The publisher database is the source for data to be replicated. The distributor service pushes out the data from the publisher for replication. The subscriber database receives the data from the distributor. Multiple subscribers can exist, allowing for multiple replicated copies of the same data. In the test environment described in this article, the server accepting the new transactions was configured as the publisher and the distributor of all tables and stored procedures for the DVD store database.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value(s) used in test</th>
</tr>
</thead>
<tbody>
<tr>
<td>n_threads</td>
<td>Number of simultaneous connections to the database</td>
<td>1</td>
</tr>
<tr>
<td>warmup_time</td>
<td>Warm-up time before statistics are kept</td>
<td>1 minute</td>
</tr>
<tr>
<td>run_time</td>
<td>Runtime during which statistics are kept</td>
<td>Varied</td>
</tr>
<tr>
<td>pct_returning</td>
<td>Percent of customers who are returning</td>
<td>95 percent</td>
</tr>
<tr>
<td>pct_new</td>
<td>Percent of customers who are new</td>
<td>5 percent</td>
</tr>
<tr>
<td>n_browse_category</td>
<td>Number of searches based on category</td>
<td>Range: 1–3 Average: 2</td>
</tr>
<tr>
<td>n_browse_actor</td>
<td>Number of searches based on actor</td>
<td>Range: 1–3 Average: 2</td>
</tr>
<tr>
<td>n_browse_title</td>
<td>Number of searches based on title</td>
<td>Range: 1–3 Average: 2</td>
</tr>
<tr>
<td>n_line_items</td>
<td>Number of items purchased</td>
<td>Range: 1–9 Average: 5</td>
</tr>
<tr>
<td>net_amount</td>
<td>Total amount of purchase</td>
<td>Range: $0.01–$400.00 Average: $200.00</td>
</tr>
</tbody>
</table>

Figure 2. OLTP driver parameters

**Scaling out the data center using smaller servers can offer cost, fault-tolerance, and ease-of-expansion advantages over larger, scaled-up servers.** Additionally, SQL Server offers different types of replication, which are defined by how often the updates to the database are published to the subscribing database and how those updates are sent. Using snapshot publication for SQL Server replication means that a complete and updated snapshot of the data is periodically sent to the subscriber. A merge publication means that changes made to both the publisher and subscriber copies of the database are merged periodically. Transactional publication allows for data from the publisher to be sent out as incremental changes on a scheduled basis.

**Configuration for replicating the DVD store**

SQL Server 2000 was installed on two Dell PowerEdge 6650 servers. Then, the initial DVD store database was loaded with the same data on both systems. The objective was to run reports on one node and accept new transactions on the other node. To achieve this objective, the Dell engineers set up transactional replication whereby the publisher database accepted new orders and replicated updates nightly to the subscriber database, which ran reports. By moving the report generation to a second server, the test team ensured that running the reports would have no effect on the active transactions occurring on the publisher database. An option was specified during the initial setup of the transactional replication that the subscriber database already had the data and schema. Specifying this option allowed the replication to begin with new orders because both databases had already been pre-loaded with exactly the same data to begin testing.

A daily transactional replication between the two SQL Server instances was set for 12:05 A.M. so that all new transactions from the day before would be replicated to the reports server. This replication allowed orders and reports to be scaled across the two servers with minimal impact on the performance of either server.

**Observing SQL Server in action**

To demonstrate how SQL Server replication can be used to run the online DVD store application on two servers, Dell engineers started a simulated workload on the SQL_orders server of approximately 233 orders per minute (about 10 million per month) using the order-entry driver. After accumulating roughly one day’s worth of orders (about 335,000), with orders still coming in at the same rate, the test team manually initiated replication from the SQL_orders server (the publisher) to the SQL_reports server (the subscriber), thereby simulating the daily replication.
Figure 3 shows the results of the SQL Server replication. The response time—the total response time for all phases of the order, including login, browse, and purchase—rose slightly during the replication period but still averaged under 0.1 second, with a few individual orders as high as 0.5 second. The order rate was essentially flat. Thus, the replication occurred in real time, took about seven minutes, and had very little effect on the performance of the order-entry system as experienced by the customer.

Scaling out the enterprise with replication technology
Data replication is one method among many for building a Microsoft SQL Server database application across multiple servers. Using an online DVD store application, the Dell test team demonstrated how to replicate SQL Server data, with orders being received on one Dell PowerEdge 6650 server and financial reports being generated by a second PowerEdge 6650 server that had a copy of the same data. New orders were replicated nightly from the order-entry server to the reports server. The test results showed that replication had a minimal impact on the ability of the order-entry server to receive orders, while continuing to accept orders at a rate of 10 million per month.

Dave Jaffe, Ph.D. (dave_jaffe@dell.com) is a senior consultant on the Dell Technology Showcase team who specializes in cross-platform solutions. Previously, he worked in the Dell Server Performance Lab, where he led the team responsible for Transaction Processing Performance Council (TPC) benchmarks. Before working at Dell, Dave spent 14 years at IBM in semiconductor processing, modeling, and testing, and in server and workstation performance. He has a Ph.D. in Chemistry from the University of California, San Diego, and a B.S. in Chemistry from Yale University.

Todd Muirhead (todd_muirhead@dell.com) is an engineering consultant on the Dell Technology Showcase team. He specializes in storage area networks and database systems. Todd has a B.A. in Computer Science from the University of North Texas and is Microsoft Certified Systems Engineer + Internet (MCSE+) certified.

Will Claxton (will_claxton@dell.com) is the Dell alliance manager for enterprise applications, focusing on oversight and strategic direction for Microsoft applications. He has a bachelor’s degree from Texas A&M University and a graduate degree from The University of Texas at Austin.

FOR MORE INFORMATION
Dell and Microsoft SQL Server 2000:
http://www1.us.dell.com/content/topics/global.aspx/alliances/en/microsoft_sql?c=us&srs=5555&l=en&s=biz