

Web 2.0 Database Strategies

New Applications, New Infrastructures

An EnterpriseDB White Paper

for DBAs, Application Developers, and Enterprise Architects

August, 2008

Executive Summary

The overwhelming success of social networking sites has proved that large interactive communities can be cultivated with user generated multi-media content leveraging application components such as blogs, wikis, forums, and other collaboration tools.

The usage patterns and volumes of these Web 2.0 sites have exposed critical performance, scalability, and reporting requirements that do not exist in the Web 1.0 world of host generated and relatively static content and shopping carts.

In general, solutions to Web 2.0 critical infrastructure issues, provide a roadmap for any enterprise application that needs to perform well, scale effectively and deliver critical intelligence to an organization's constituencies.

The Web 2.0 database strategies discussed are sufficiently generally to provide value to all readers in the context of detailed solutions using EnterpriseDB's Postgres Plus products.

An in depth discussion targeted specifically to your organization's requirements can be scheduled with an EnterpriseDB domain expert by sending an email to sales@enterprisedb.com.

Web 1.0 and the Application Server

The overwhelming success of social networking sites such as MySpace.com, Facebook.com, and hi5Networks.com is instructive to information technologists on two dimensions. First, these sites prove the thesis that large, interactive communities can be cultivated using application components such as blogs, wikis and user forums.

Second, the usage patterns and volumes of social networking sites have exposed critical computing infrastructure requirements that do not exist in the world of static content and shopping carts (i.e., Web 1.0).

Although these lessons may seem to be primarily useful to architects and developers of commercial Web 2.0 applications, in reality Web 2.0 architectures provide a roadmap for any enterprise application that needs to perform well, scale effectively and deliver critical intelligence to the organization.

Online stores such as Amazon.com and Dell.com are good examples of the Web 1.0 paradigm. These sites provide extensive product catalogs that are viewed in high volume, but the catalog content is fairly static. Although some Web 1.0 applications support large visitor populations, dynamic site data is limited to features such as custom personal profiles, shopping carts and product configurators.

Application Servers like JBoss are often the power plants of Web 1.0 architectures. To mitigate database contention, frequently-accessed content is stored in the server's memory cache and delivered to site visitors on demand. Since the data is relatively static, synchronizing the cache with content changes is not highly burdensome on the infrastructure.

When the application applies a database transaction (insert, update or delete), the application server synchronizes its cache with the new content, and then propagates any insert, update or delete changes to other servers in the infrastructure. This cache coherency approach scales acceptably for applications that have high query, low transaction profiles.

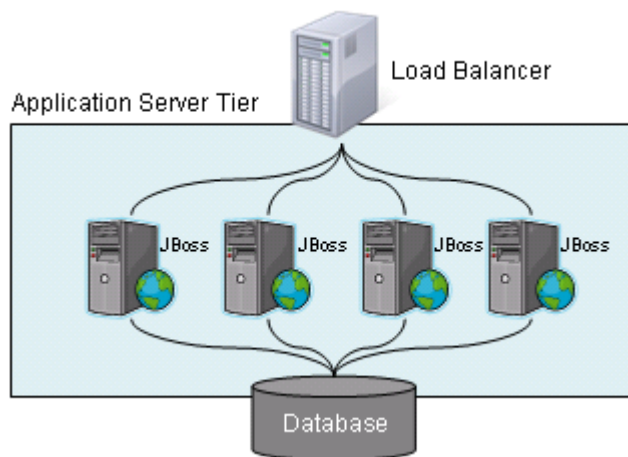


Figure 1: Web 1.0 Architecture

Web 2.0 – New Applications, New Challenges

Social networking is the central theme of Web 2.0 applications. Wikis, blogs and community forums are examples of social networking application components that provide high levels of user supplied content and user-driven interactivity. These interactive components place new demands on the application database, as this paper will explore in detail.

What's more, social networking sites face a particularly thorny scaling dilemma – they must accommodate explosive growth, even though the magnitude and timing of that growth is difficult to predict. Myspace.com and hi5Networks.com, for example, now have more than 300 million and 75 million registered users, respectively, and face tremendous scaling challenges as their communities evolve.

By offering high levels of interactivity, sites like hi5Networks aim to capture and retain the attention of large numbers of users. The goal is to provide an experience that encourages users to visit often, stay connected for long periods and interact continuously with other community members. Thus, Web 2.0 applications pose important infrastructure challenges not present in their first generation counterparts, including the need to:

- Handle large amounts of dynamic, user-driven data
- Scale the underlying infrastructure in the face of uncertain demand
- Maintain durable user connections and sessions as usage increases

As described earlier, Web 1.0 architectures mitigate database contention by maintaining static and semi-static data at the application server tier, with cache coherency managed primarily by the application servers

themselves. Web 2.0 applications process much higher volumes of dynamic data – blog posts, wiki entries, and forum threads – that can quickly overwhelm the application server tier.

As application usage increases, simply adding more application servers to the mix is not an effective scaling solution. The overhead of maintaining consistent cache data across an expanding server farm increases latency and starves the application servers of cycles that are needed by the application (e.g., to manage rich user interfaces).

The solution to these new architectural challenges is a high performance, massively scalable database tier that complements and enhances the application server infrastructure.

Performance, Scaling and Reporting

Although performance and scaling are different concepts, in practice they are both critical to creating effective application architectures. Performance is a measure of the response time of a single operation, and scaling is the measure of the stability of that response time as application load increases.

Reporting (a.k.a. business intelligence) is also germane to the Web 2.0 architecture discussion for two reasons. First, Web 2.0 applications generate huge amounts of click stream and usage information. Second, site managers need to continually analyze that information to enhance their visitors' experiences. In concert with performance and scaling, reporting represents the "third leg of the stool" in a viable Web 2.0 architecture.

Performance, scaling and reporting are each discussed in more detail below, using EnterpriseDB's Postgres Plus product family for further elaboration.

Performance

As discussed above, Web 2.0 applications pose complex database throughput challenges. Users are continually adding and changing site information, requiring the database to handle high transaction volumes while also processing very large result sets. Many Web 2.0 applications require complex queries on unstructured and semi-structured data.

Developers often need to optimize query execution by explicitly defining the sequence of index searches. Clearly, Web 2.0 applications require an advanced database that can handle both transactional and inquiry

operations at very high throughput levels. Postgres Plus provides several capabilities that are specifically aimed at performance-critical applications.

Dynamic Database Tuning

The foundation of a high performance Web 2.0 application is a database tier that is tuned to optimize system resources, application requirements and workload profiles. The Postgres Plus Dynatune™ function allows systems administrators to configure a variety of performance and database workload parameters that optimize runtime execution. DynaTune automatically recalibrates the database execution profile based on available system resources, ensuring that all system upgrades (e.g., to CPU and memory) immediately yield optimal database performance.

Bulk Data Processing

Due to the nature and volume of the user generated content (text and multi-media), Web 2.0 applications must frequently process large result sets at the database tier. For example, discussion threads on public forums can spider into thousands of posts that must be retrieved as a single database result set. Looping over large result sets within a stored procedure is an intensive database operation.

Fortunately, Postgres Plus includes advanced processing features – bulk collecting and bulk binding – that offer significant performance gains when compared to inefficient cursor based looping operations. In bulk loading applications such as nightly integration jobs, the Postgres Plus loader bypasses the SQL engine and provides the capability to load large volumes of data directly into the database.

Database Compression and Optimization

Postgres Plus includes enhancements that reduce physical database size, resulting in commensurately better performance for I/O intensive operations. Postgres Plus also supports Optimizer Hints, a feature that allows developers to influence the way the database runs a query, resulting in better performing SQL.

For example, using an Optimizer Hint, the developer can instruct Postgres Plus to use a specific index as the basis of a particular search, optimizing both query and subsequent join operations. Additionally, Postgres Plus supports an advanced transaction feature called Heap-Only Tuples (HOT), which significantly increases performance for all update and delete operations while maintaining full multi-version concurrency control (MVCC).

Scaling

As the load of a Web 2.0 application grows, response times begin to deteriorate unless the infrastructure is appropriately scaled. The data intensive nature of these applications limits the scaling benefits that can be achieved by adding more application servers. Thus, the database tier must be scaled vertically and/or horizontally.

Vertical scaling involves increasing the computing power (CPU, memory and disk resources) available to a database server. It may include expensive proprietary server technology and require significant upfront resource commitments.

Horizontal scaling involves partitioning and/or replicating the database across multiple instances using commodity hardware and software. It is often the most cost effective means through which to build a Web 2.0 infrastructure, but requires the database to efficiently perform complex distributed computing operations.

Postgres Plus delivers several capabilities that are critical to both vertical and horizontal scale-out of the database tier.

Distributed Memory Caching

Many data-intensive applications cache frequently accessed data at the application server tier. As discussed earlier, however, Web 2.0 data is often highly dynamic in nature. Thus, Web 2.0 applications burden the application server tier with the chores of maintaining coherent, synchronized cache data. As application usage grows, the overhead of maintaining cache coherency across an expanding server farm increases latency and starves the application servers of cycles that are needed elsewhere in the infrastructure (e.g., to manage rich user interfaces).

Postgres Plus provides a sophisticated distributed memory caching feature that allows critical caching chores to be performed at the database tier (see Figure 2). Special purpose cache servers offload the chores of maintaining coherent cache data from the application server tier without adding overhead to the primary database instance(s).

Importantly, cache invalidation is triggered from the database, ensuring that the cache does not return stale data even if the data is altered by external applications. By isolating the complex, resource intensive tasks of cache management to cache data servers, distributed memory caching provides an effective means through which to massively scale Web 2.0 infrastructures.

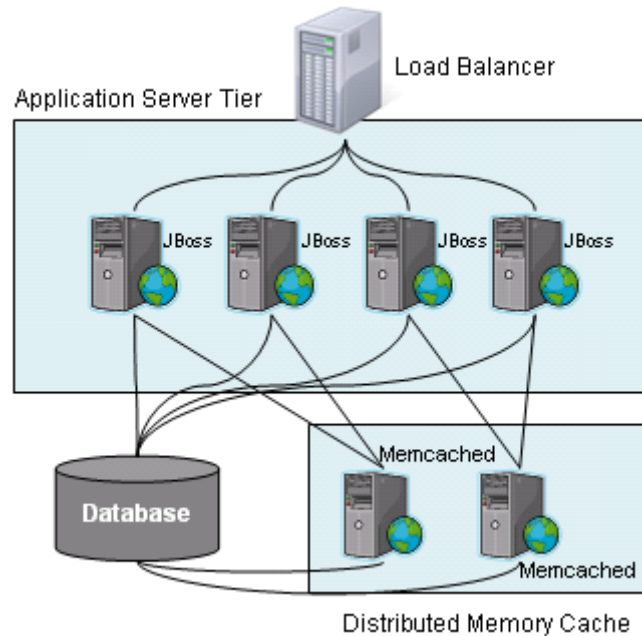


Figure 2: Distributed Memory Cache Architecture

Transaction Processing

As discussed earlier, the application database must excel at processing insert and update transactions. Although advanced capabilities like distributed memory caching offload resource-intensive query operations to special purpose cache servers, thus optimizing the database for OLTP operations, the database itself must also have the raw power to handle transactional workloads.

Postgres databases are renowned for transaction processing, functioning flawlessly as the workhorse behind many of the most demanding OLTP applications in production today. Postgres Plus includes multi-version concurrency control (MVCC) and table-level partitioning to further enhance the performance of transactional systems.

MVCC is an “optimistic” concurrency strategy that allows read operations to acquire query contexts without blocking write (insert/update/delete) operations. MVCC significantly reduces data contention, and thereby improves performance, in transactional applications by minimizing locking requirements.

Table-level partitioning allows data to be sliced horizontally and distributed across database instances, optimizing access to frequently needed data while ensuring fast, consistent access to all information.

Advanced Database-level Partitioning

As a Web 2.0 application community grows, additional scale can be achieved by partitioning the main application database into multiple physical servers. A horizontal partitioning strategy called “sharding” is useful in many Web 2.0 scale-outs.

Sharding is a federated model of database instances containing subsets of the full dataset. By partitioning datasets into smaller pieces across many physical servers, the transactional bandwidth to handle the growing application is divided across several servers. However, distributed architectures such as sharding require sophisticated database capabilities that can efficiently retrieve data from multiple partitions and return unified result sets.

Postgres Plus includes database federation capabilities that support cross-database joins by linking federated tables into a single schema (see Figure 3). These advanced capabilities enable the implementation of partitioning strategies such as sharding, thereby supporting efficient, cost effective scale-out of the Web 2.0 infrastructure.

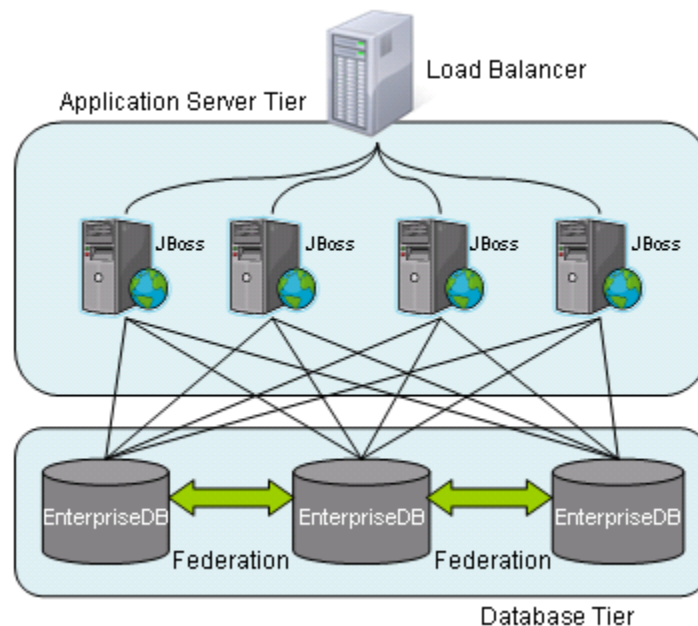


Figure 3: Database Tier with Federated Caching

Integrated Connection Management

In Web 1.0 architectures, database connection pools are typically managed at the application server tier. Each server allocates and manages its own pool. As the number of servers increases, pool

management often becomes a performance bottleneck – some pools thrash while others are underused.

Load balancers provide no relief because they target CPU and memory utilization, not database connections. Postgres Plus' Integrated Connection Management (ICM) feature solves this performance problem by managing connections at the database tier. As the number of application servers increases, ICM allocates and manages database connections dynamically, optimizing database access across all server instances, as well as across all shards in the database tier.

Reporting

Successful community managers are vigilant about understanding the profiles, content preferences and usage patterns of their visitor constituencies. As communities flourish and evolve, they produce massive amounts of click stream and usage information that help site managers enhance visitors' experiences, increase their return rates and prolong their interactions.

Although most decision support information can be stored directly in the database, business intelligence queries are extremely resource intensive and will degrade site performance if the infrastructure is inadequately configured.

Postgres Plus provides powerful solutions for high-volume reporting architectures. Figure 4 illustrates one possible architecture consisting of a federated data mart in the database tier and a grid of databases in the data warehouse tier. In this example, the data mart contains relatively fresh, two-dimensional data that can be accessed directly from the reporting server using standard queries.

The data warehouse contains older, multi-dimensional (e.g., time series) data that has been partitioned into a database grid and accessed through Postgres Plus' GridSQL parallel query processor. Postgres Plus solutions can be flexibly configured to meet the most demanding business intelligence applications while optimizing the performance of transactional front-end applications.

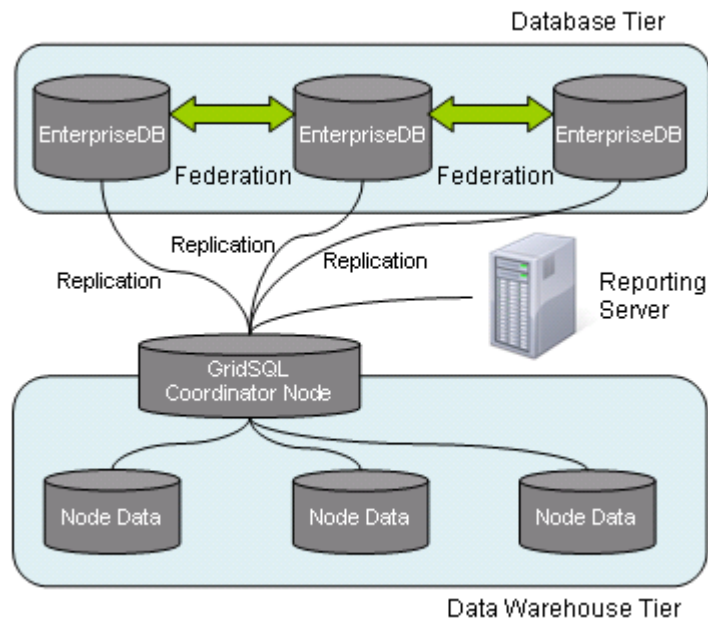


Figure 4: Postgres Plus High Volume Reporting Architecture

Conclusion

Advanced Web 2.0 applications offer exciting new ways to attract, retain, and cultivate vast user communities.

However, the sheer volume of visitors and the transactional content they create, pose very different computing infrastructure demands when compared with predecessor application architectures that depend primarily on application server based cache management.

Postgres Plus products solutions efficiently compliment application servers with the following features:

- Dynamic Database Tuning with DynaTune,
- Bulk Data Processing,
- Optimizer Hints,
- Heap Only Tuples,
- Distributed Memory Caching,
- Multiversion Concurrency Control,
- Table and Database level Partitioning,
- Integrated Cache Management, and
- GridSQL Parallel Queries.

These features provide the high performance, flexible scalability and sophisticated reporting access needed to implement advanced Web 2.0 applications.

Using Postgres Plus, Web 2.0 sites like hi5 Networks are delivering advanced application functionality to massive user communities, with the highest levels of performance available anywhere.

For more information or advice on your Web 2.0 architecture, please contact EnterpriseDB at:

https://www.enterprisedb.com/about/contact_us.do

About EnterpriseDB

EnterpriseDB is the leading provider of enterprise class products and services based on PostgreSQL, the world's most advanced open source database. The company's Postgres Plus products are ideally suited for transaction-intensive applications requiring superior performance, massive scalability, and compatibility with proprietary database products.

Postgres Plus also provides an economical open source alternative or complement to proprietary databases without sacrificing features or quality. EnterpriseDB has offices in North America, Europe, and Asia. The company was founded in 2004 and is headquartered in Edison, N.J. For more information, please call +1-732-331-1300 or visit <http://www.enterprisedb.com>.