**EPM HW Sizing and Capacity Planning Notes**

* It is strongly recommended that you deploy a server with dual quad-core processors, 8 GB or RAM or more, and 200 GB of disk space at RAID 1.
* Factors that may affect throughput and performance
* Number of users
* The type, complexity, and frequency of user operations
* The number of postbacks in an operation
* The performance of data connections.
* Workloads, and subsequently the hardware needs, will differ in relation to the following variables

|  |  |
| --- | --- |
| Projects | * *Number of projects* * *Typical project sizes in terms of tasks* * *Number of project level custom fields* * *Level of linking (dependencies) between tasks* |
| Users | * *Concurrency of users*. How many users will be hitting the system at the same time? What is the average load, what are the spikes in traffic? * *What security permissions do users have*? This affects both the amount of data the server needs to present to the user at a given time, along with the complexity of the security checks the server has to do. * *Geographic distribution of users*. When users are spread over large geographical areas there can be detrimental performance effects due to network latency. This also impacts usage patterns insofar as users are likely to hit servers at different times during the day, making it harder to find low-traffic periods in which to run maintenance tasks such as backups, reporting, or Active Directory sync. |
| Usage Patterns | * *Workload conditions*. Which set of features are being commonly utilized? For example, a deployment that uses time-sheeting heavily will have different characteristics than one that does not use time-sheeting. * *Average time between page requests*. * *Average session time.* * *Payload of Pages* (How many web parts do you have on a given page? How much data do they contain?) |

* Task formula fields tend to take the largest toll on performance because they need to be computed for each task.
* The more you utilize Timesheets, the more resource demands will be placed on the SQL Server.

**For the SQL Server some important considerations are**

* Ideally, you should separate and prioritize data among disks. Place your data files and your SQL Server 2008 transaction logs on separate physical hard disks
* RAID 5 should provide a good compromise between reliability, and throughput.
* For a large scale implementation; separate your reporting database onto a separate database server.
* Project Server 2010 does support running on virtualized machines.

### Network Requirement

For most Project Server deployments, network bandwidth tends to not be the bottleneck on performance. The table below lists the recommended specifications of network components

|  |  |  |
| --- | --- | --- |
| **Component** | **Small and Medium** | **Large** |
| # of NICs | 1 | 2 |
| # NIC Speed (Network) | Any speed greater than 100mbps should be fine | 1 Gb/s |
| Load Balancer Type | NLB or hardware, both are acceptable | NLB or hardware, both are acceptable |

**Note:**

Project Server 2010 only provides limited scale-out capabilities. While additional servers can be added to act as Web Front End servers, and Application servers, on the backend the SQL Server has limited scale-out possibilities. At most, you can separate Project Server databases onto two dedicated SQL Servers – one holding the reporting database, and the other housing the archive, draft, and published databases. What this implies is that for large datasets the SQL Server becomes the primary bottleneck, and a primarily “scale-up” strategy needs to be employed, where additional hardware resources are added to the SQL Server.

### To provide for more user load:

* Scale-out by adding more Web Servers to serve as dedicated Application and Web Front End servers.
* You may also scale up any Web Front End or Application Server to improve performance by increasing the capacity of those servers.
* Note that as you add more Web Front End servers, and Application servers, the load on the SQL Server will increase, and your SQL Server could in turn become your bottleneck. What this implies is that for large datasets the SQL Server becomes the primary bottleneck, and a primarily “scale-up” strategy needs to be employed, where additional hardware resources are added to the SQL Server.

### To provide for more data load:

* To provide for more data load, add capacity to the database server by increasing the capacity of that single server.
* Separate the Project Databases from the SharePoint Databases by moving the four Project databases onto their own dedicated database server.
* Beyond this, it is also possible to detach the Reporting Database to a separate database server. The server with the Reporting database can then additionally act as a failover for the server with the other three Project databases. It is possible to achieve this by using SQL mirroring, and setting up a cross-fail relationship between the server housing the Reporting Database, and the server housing the remaining Project databases.

### Recommended server role ratio:

As a rule of thumb, a recommended ratio for maintaining a manageable load on the SQL Server is:

2 Web Front Ends: 1 Application Server : 1 SQL Server

### Investing rules of thumb:

* As a rule of thumb, in the early stages of scaling your Project Server deployment, you will want to invest primarily in purchasing additional memory. Most often, the subsequent areas you would want to invest in are disk spindles, and then network resources.

### Database Server Optimizations

* Separate the database files and the transaction log files away from the OS drives – preferably each to their own partition. This helps by reducing IO contention between the host operating system and SQL Server, and then also between SQL database files and log files, which tend to have different update patterns depending on what recovery strategy is used.
* SQL Server supports the use of statistics. Ensuring that the statistics are up-to-date will improve the ability of the SQL Query Optimizer.
* Ensure that your SQL Server indexes are not fragmented.
* Separate the TempDB onto its own partition. Split the database into several physical files – ideally, splitting it into as many files as you have processors on your database server.
* Consider utilizing a RAID subsystem for your data needs
  + RAID 5 is recommended for medium data set sizes.
  + RAID 5 is acceptable for large dataset sizes, but RAID 10 is ideal
* Move indexes onto their own partition
* Project Server has been optimized to utilize the benefits of SQL CLR. Although SQL CLR is not a requirement, if available and enabled, it has the potential for increasing the performance of some operations

### Security Setting Optimizations

* Use groups and categories where possible rather than more granular permissions that require additional complexity in the security checks
* Try to restrain people’s security permissions to the projects they need to have access to, that way they load only the data they need to when interacting with Project Server

### Custom Field Optimizations

* Generally, the prescription is to try to limit the number of custom fields used, especially at the task level.
  + As a rule of thumb, try to use less than 10 -15 task level Enterprise Custom Fields.
* Task and assignment custom fields are the primary bottleneck in saving from WinProj to the server in most observed customer data sets.

Three things that may influence system performance are

* Platform
* User
* Data

Performance Degradation has been observed in following conditions

* WFE more than 4
* Application Servers more than 3
* SQL Server shared across multiple applications and contention across system resources
* Total queue more than 12 for each type of Queue

In 64 bit scenario, the focus is more on system resources and not system numbers