

---

# Server Virtualization and MySQL: Your Options

# A MySQL® Business White Paper

## Table of Contents

<a href="#">1 INTRODUCTION.....</a>	<a href="#">3</a>
<a href="#">2 WHAT IS VIRTUALIZATION?.....</a>	<a href="#">3</a>
<a href="#">3 HARDWARE-BASED VIRTUALIZATION.....</a>	<a href="#">3</a>
<a href="#">4 THE HYPERVISORS.....</a>	<a href="#">5</a>
<a href="#">4.1 Baremetal Hypervisor.....</a>	<a href="#">5</a>
<a href="#">4.2 Hosted Hypervisor.....</a>	<a href="#">8</a>
<a href="#">5 OS VIRTUALIZATION.....</a>	<a href="#">9</a>
<a href="#">6 GENERAL VIRTUALIZATION BENEFITS.....</a>	<a href="#">10</a>
<a href="#">7 WHY IS MYSQL THE MOST FREQUENTLY USED?.....</a>	<a href="#">11</a>
<a href="#">8 HOW TO FIND MYSQL VIRTUALIZATION OPPORTUNITIES.....</a>	<a href="#">11</a>
<a href="#">9 OTHER IDEAS.....</a>	<a href="#">12</a>
<a href="#">10 GENERAL ADMINISTRATION HINTS.....</a>	<a href="#">12</a>
<a href="#">11 CONCLUSION.....</a>	<a href="#">12</a>
<a href="#">12 LINKS AND REFERENCES.....</a>	<a href="#">13</a>

# 1 Introduction

Virtualization technology has now gone main stream, and its footprint within IT infrastructures continues to accelerate. If you are responsible for MySQL within your enterprise you need a firm understanding of these technologies and offerings and how they work when combined with MySQL deployments. This paper will explain virtualization technologies and how they work with MySQL to help optimize you infrastructure investment while at the same time increasing the performance and availability of your business-critical applications. Along the way I'll also provide pointers and links to resources with additional details.

## 2 What is Virtualization?

In general terms virtualization abstracts the physical computing hardware. More specifically, Server virtualization hides server resources – the physical server, CPU, disks, and operating systems. IT professionals use server virtualization products to divide physical server into multiple virtual servers. These virtual servers are often referred to by many different names including virtual private servers, partitions, instances, containers, jails, and others.

This paper focuses on *server virtualization* which will be referred to from here on as simply *virtualization*.

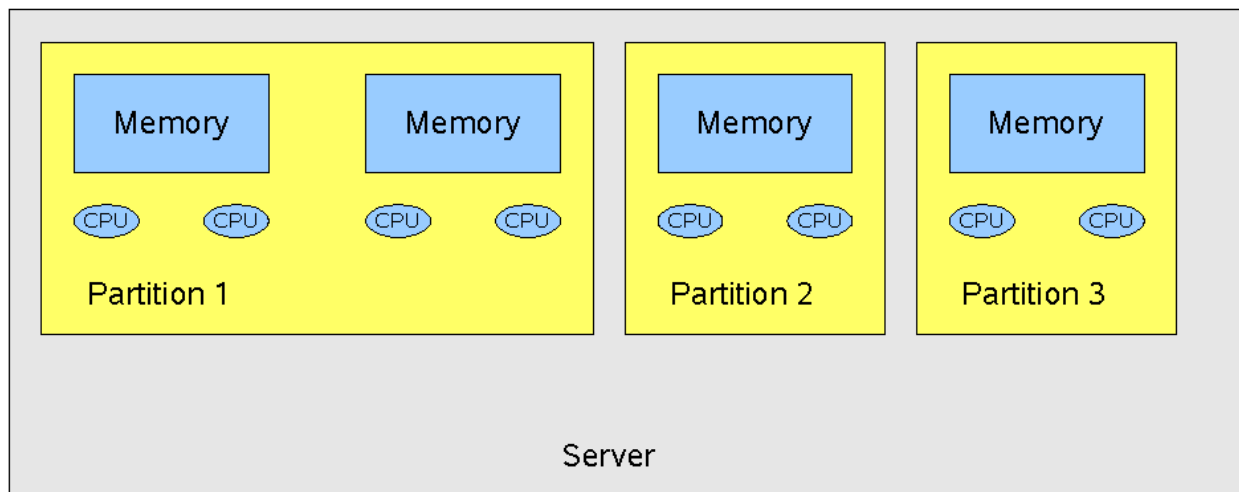
There are really 4 basic categories of virtualization, which I will cover in order from most hardware oriented to entirely software oriented:

1. **Hardware-based**
2. **Native Hypervisors**
3. **Hosted Hypervisors**
4. **Partitioned Environments**

## 3 Hardware-based Virtualization

Also known as hard partitioning or virtualized hardware, this option has been around the longest, around 10 years. Many fail to list this in the virtualized environment discussion, but as these provide dynamic configuration of multiple Operating Systems within a single server, it fits my definition, as it abstracts the hardware resources. Hardware Partitioning divides a physical server into multiple by assigning resources (CPUs, Memory, etc) into partitions – each partitions pool of resources represent a virtual server. Administrators add or subtract resources to individual partitions as needed either to boost performance for a partition that has become resource limited or take away if resources are underutilized.

Diagrammatically, partitioned hardware looks like this. As you can see partition 1 has been assigned twice the computing power of partition 2 and 3:



**Figure 1 – Hardware-based Virtualization**

Most partitioned systems allow hardware to be added or removed on the fly and assigned to or unassigned from partitions. If cost is no object and a high degree of flexibility and uptime is desired, virtualization via hardware partitioning is your best option. From a MySQL perspective, running within a partition will work the same and performance will be the same as running on equivalent standalone server hardware, but with the added flexibility of virtual resource management and other features provided by this method.

**Pros**

- High fault tolerance, higher degree of separation - “Electrical Isolation”
  - If one fails the rest are fine so higher levels of business continuity
  - Hardware can fail and application keeps going. You can hot swap in hardware.
  - No bits exposed outside of partition as needed for highly secured applications
- Workload balancing
  - Able to dynamically add resources
    - so if you need more add the hardware and assign to the partition that needs it.
  - No workload side affect
    - one partitions load does not affect the others
- High performance
  - As fast as dedicated and in some cases better due to isolation and contention avoidance.
  - Partitions don't add any performance overhead.
- Can run different OS versions in each partition

**Cons**

- Expensive
  - The resources are totally dedicated
  - Only available on high to mid range hardware
- Does not optimized hardware resource utilization

### **When you want this for MySQL**

- High-end enterprise MySQL database applications that you expect to scale up and want the flexibility to add compute resources later
- Cases where “electronic” isolation is needed for security reasons
- Cases where uptime requirements are very, very high.
- Cases where resources need to be dedicated.

### **Examples**

- Sun Sparc M-series Servers
- IBM, HP, and others offer similar products.

### **Deployment Process**

1. Acquire a server that supports hardware partitioning
2. Create partitions
3. Assign hardware to these partitions
4. Install an Operating Systems on that server
5. Add MySQL
6. Repeat 2-5 tuning resources or adding new hardware components to a partitions

### **Tip**

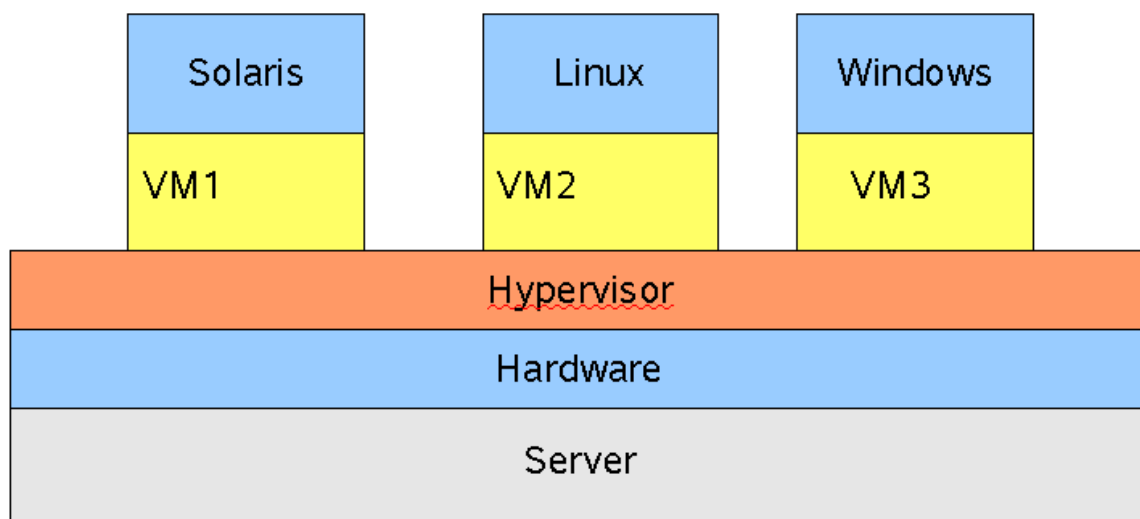
In some cases more CPU cores and thus more threads do not lead to better MySQL performance and can even degrade it. For some MySQL database engines, high end hardware with 8 threads (core) support may provide the best performance. Thus the flexibility to create an 8 core partition should be evaluated if MySQL is the lone application on a partition..

## **4 The Hypervisors**

A hypervisor is a program that supports running multiple operating systems on a single server. It allows each running Operating System to think it has got its own hardware when it is actually shared. The hypervisor sits in the middle and controls each virtual servers interaction with the physical hardware. Hypervisors bring virtualization to commodity hardware. A hypervisor can be either pure software or a combination of software/hardware(firmware). There are 2 general categories of hypervisors – Baremetal and Hosted.

### **4.1 Baremetal Hypervisor**

Also called native or type 1, baremetal hypervisors are where most of the current virtualization buzz has been focused. A baremetal hypervisor runs directly on the server hardware. As seen here, it's basically a highly specialized, very small operating system that creates this hypervisor layer.



**Figure 2 – Baremetal Hypervisor**

Many companies are now running baremetal hypervisors technology within their production environments as well as for development and test systems. Baremetal hypervisors provide great flexibility as well as quite respectable performance in most cases and especially when compared to their Hosted hypervisor counterpart.

Using the hypervisor, virtual servers are easily created, managed and migrated. Many companies will create virtual standby servers as backups or mirrors for physical servers. There's huge flexibility gained by going with this style of virtualization. There is some overhead in running the hypervisor so you don't get 100% of the performance of the native hardware, but with new baremetal software and firmware technologies you may get as low as 5-10% performance degradation, higher in some cases, lower in others, but this cost is falling with each new release. Also in many cases where you migrate servers and applications from older 32 bit to newer faster 64 bit systems with hypervisor enhanced hardware - you may not see any difference in performance or even experience improvement.

Since baremetal hypervisors share hardware across the virtual servers you will see higher sustained utilization numbers and thus better use and balance across existing hardware. On the downside, say you have 16 virtual servers running on a single physical server. If that server goes down, all 16 go down. Because of that, most of these systems have methods for dynamically migration or rapid fail-over to other hypervisor based virtual servers.

Also Hypervisor based virtual servers are easy to migrate. If a virtual server gets too overloaded, its relatively straight forward to migrate to another computing resource, or to add a computing resource.

Companies are continually evolving chip design; Sun UltraSparc T2, Intel VT and AMD-V have been designed specifically with features to support hypervisors. In addition some Operating Systems support paravirtualization techniques. Paravirtualized Oses are specifically ported to take advantage of specific virtual machine service interfaces that result in improved performance.

Baremetal hypervisors are the most popular virtualization option in use by the database community for production use and going more and more mainstream in enterprises.

## Deployment Process

1. Acquire a server (preferably one that has built-in hardware hypervisor support)
2. Install the hypervisor product of your choice
3. Create and configure a virtual server
4. Install an Operating System on that server
5. Add MySQL
6. Repeat 3-6

## Examples

### Hypervisors

Sun xVM Server and LDOM (Solaris), VMware ESX Server 3.0.1, Microsoft Hyper-V, Citrix XenServer, HP Vpar and Integrity VM, IBM DLPAR and Micropartition

### Paravirtualized Operating Systems

Solaris, Open Solaris, Linux 2.6 kernels, NetBSD

## Pro's

- Good price/performance ratio,
- Great flexibility and manageability,
  - May reduce management costs.
- Higher efficiency/resource utilization, Less hardware,
- Power and space reducing
- Higher Reliability
- Server consolidation and containment
- Added data protection options,
- Application isolation.

## Con's

- Lower availability,
- Some added complexity for management and support,
- Some performance degradation
- More deployment design considerations

## When you want this for MySQL

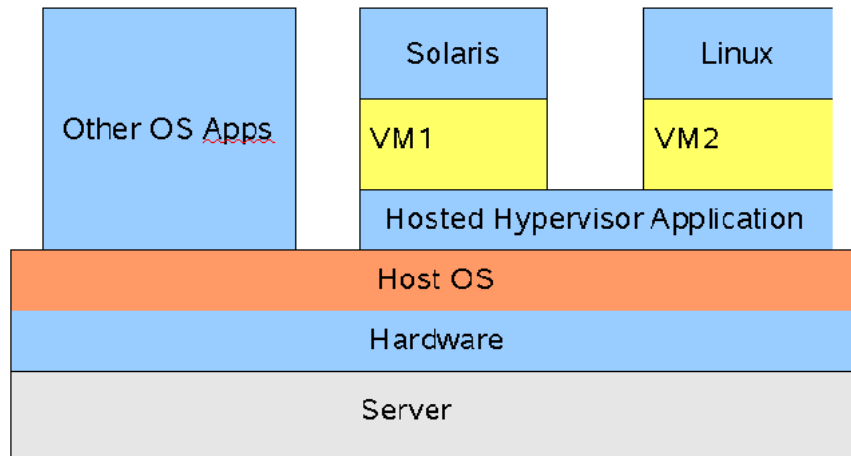
- Low to moderate enterprise MySQL database applications that can take a small performance hit.
- MySQL database needing OS isolation is needed for security reasons
- Where uptime requirements aren't absolutely mission critical.
- Testing that requires more performance than available from a hosted hypervisor

## Hints

- With higher improved host utilization many find they need to add additional network cards to avoid NIC card contention.
- Inexperienced Administrators are notorious for misconfiguring network interfaces in this environment
- Just like a physical server, be sure you give the virtual machines adequate physical memory
- Think about dedicating CPUs for MySQL virtual servers
- Always use direct physical disk configurations for database files if it available
- For optimum performance separate the OS and database disks.
- Use high performance SANs and RAIDs for best performance.
- Spread your virtual disks across your physical disks
- Be careful not to dynamically reconfigure virtual disks with the MySQL database running. This applies to regular hardware as well, but its a common goof on VMs.

## 4.2 Hosted Hypervisor

A hosted hypervisor is a “pure software” solution. As shown here the hosted hypervisor runs within another operating system, and thus it's never working directly on the hardware.



**Figure 3 – Hosted Hypervisor**

Hosted hypervisors are usually very easy to install and setup. However because its all in software, running a hosted hypervisor is more resource intensive and thus performance of its virtual servers is slower than the other 3 options. This significantly hampers MySQL performance. Also because of its architecture, hosted hypervisors are less secure as it shares files with the hosted OS and processes and memory are easily probed.

Hosted hypervisors are most frequently used for simple test and development environments and MySQL running in hosted hypervisors and as part of simple development and test systems is a good option. The availability of “virtual appliances” which are really just pre-configured virtual machine images (a file), makes setting up a LAMP stack server, for example, trivial. Often complaints you may see about the performance of a database on a virtual platform are those running on a hosted hypervisor. Unfortunately this is often over-generalize and applied to this to all virtualization technologies. So if you hear this just remember to first ask which technology was used.

### Deployment Process

1. Acquire a server with an installed OS
2. Install the hypervisor product of your choice
3. Create and configure a virtual server
4. Virtual Install an Operating Systems on that server
5. Add MySQL
6. Repeat 3-5

### Examples

Sun xVM VirtualBox , VMware Server, Microsoft Virtual PC and Virtual Server,

**Pros**

- Very inexpensive,
- Very Simple
- Great flexibility
- Less hardware,
- Power and space reducing
- Server consolidation and containment
- Application isolation.

**Cons**

- Poor performance
- Low security
- Low availability,

**When you want this for MySQL**

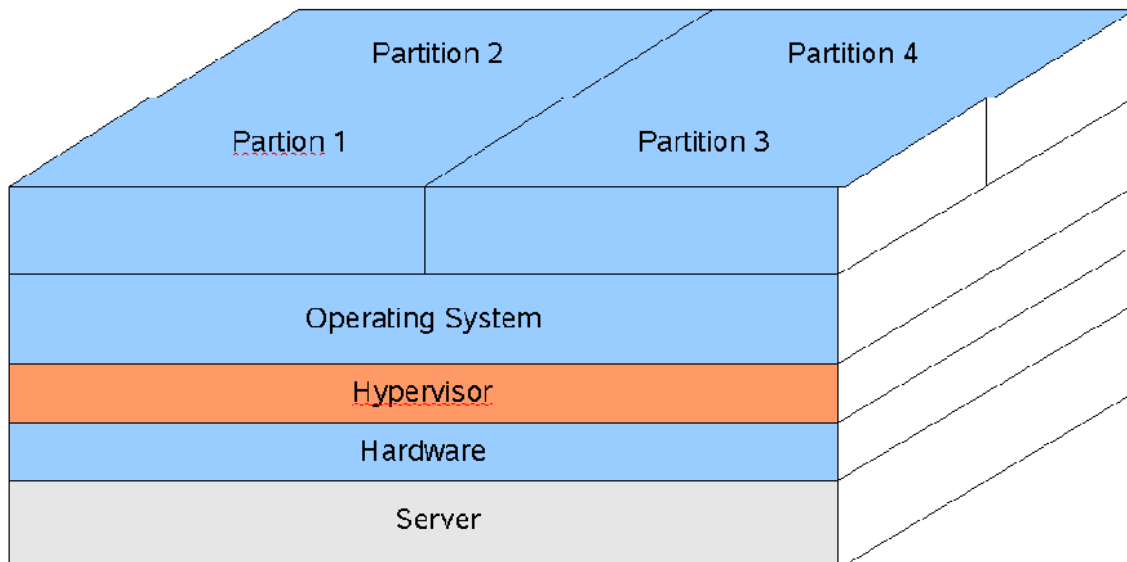
- For test and development environments.

**Hints**

- Just use for simple development and test environments
- Don't forget to backup and take snapshots
- Don't use in production
- Expect slow performance and migrate to a baremetal hypervisor if too slow..

## 5 OS Virtualization

Virtualization can also take place within an individual operating system. In this case the operating system kernel allows you to create multiple user-spaces or partitions (also referred to as containers, zones, VEs, VPs, or jails) instead of just one. If you only need to be running 1 OS version, but want isolated “virtual” servers this mature technology is a great option. Its common for applications and especially database applications like MySQL to be separated into individual zones.



## Figure 4 – OS Virtualization

OS virtualization creates very little overhead since its running on a shared kernel directly on the hardware. Thus MySQL performance will be more or less equivalent between an OS Virtualized environments and a native OS. Using a dedicated partition MySQL can have its own networking and I/O paths which may actually improve performance in some cases.

### Examples

Sun Solaris Containers/Zones, Parallels Virtuozzo Containers, FreeBSD Jail, AIX WPARs

### Pros

- Performance - Little to no overhead,
- More easily combine applications on a single system, added application isolation, enhances security, simpler to manage
- Additional Isolation
- No added expense, can even save if you avoid buying dedicated hardware.
- Reduce Hardware
  - Combine low utilization systems
- Isolate applications from faults
- Maintain Service Levels
- Fine tune response times

### Cons

- Shared kernel resources
- Not as flexible as other virtualization choices
- Single OS
- Fewer OS options

### When you want this for MySQL

Use this to Isolate the MySQL server from other components and get finer grained control over the servers resources, obtain a higher security level, and maintain a more consist MySQL service level..

## 6 General Virtualization Benefits

So know that you have a better understanding of virtualization you can probably see the many possible benefits that can be derived by virtualizing servers and running MySQL. Of course, each virtualized solutions covered attains varying degrees of these benefits, and there are more than listed here, but here's a few to consider.

### Improve Efficiencies and Reduce Costs

- Maximize asset utilization
- Fight Server sprawl
- Reduced power and cooling needs
- Lower capital expenditure
- Lower operating expenditure
- Lower carbon footprint
- Increased IT staff productivity
- Reduced software licensing and hardware management costs

### Increased Business Agility

- Dynamically scale up or back

- Rapidly response to changing business needs
- Scale IT infrastructure to available power
- Faster provisioning of services and infrastructure

#### **Maximize Business Continuity**

- Optimize availability
- Application isolation
- Centralized/remote management
- Reduce backup windows
- Simplify disaster recovery
- Increase security.

## **7 Why is MySQL the most frequently used?**

Relative to other databases in the market MySQL more often presents compelling candidate for virtualization than other competitive products.

#### **Some good reasons for this often include:**

- Small installation footprint
- Can limit install to components and engines needed and exclude others.
- Small runtime (as small as 50 Mb)
- Simple to configure and manage
- No licensing headaches - virtualization friendly licensing with the Open Source model and Enterprise
- Strong official support for users running on virtual environments
  - better support policy than most other vendors
- Proven package within cloud computing environments which are largely based on virtual servers
  - for example supported by Amazon EC2, Joyent, RightScale, Bungee Labs
- Runs very well on Operating Systems most commonly virtualized.

## **8 How to find MySQL Virtualization Opportunities**

When looking at your enterprises MySQL database its likely there will be databases and servers that are easy to qualify as candidates for virtualization. These are the “low hanging fruit”

Current MySQL installations such as

- Small databases
- Seldom used databases
- Failover “mirrors” or copies
- Low performance or used for batch processing
- Low RAM requirements
- Performance requirements not an imperative
- Transient databases or servers you may want to use and set aside
- Could benefit from added isolation or additional security

**Note:** The MySQL Enterprise Monitor provides a great tool for collecting the metrics needed to asses your all of your MySQL servers.

## 9 Other Ideas

Virtualization technology runs in places other than the internal LANs of the corporate world. More and more, datacenters and hosting providers are using virtualization as an integral part of their technology portfolio. Whether simply as a method to create cookie cutter servers for a customer with a database server running MySQL, an entire LAMP stack, or even a complex Cloud computing solutions, MySQL is most frequently the database of choice.

Similarly many SaaS (Software-as-a-Server) offerings take advantage of the benefits of virtualization in getting their products offerings packaged and too the market. The flexibility and security isolation are just 2 of the drivers for them.

## 10 General Administration Hints

Hypervisors and their installed virtual servers are similar to databases and other server applications. Thought must go into the installation and configuration. Poorly configured and designed deployments more often than not run poorly whether virtual or not.

Always remember, what's true on physical machines is also true on a virtual. If you poorly configure a database or use too few spindles your performance will be poor. Just because you're running on a VM doesn't mean you can forget about MySQL best practice. But this often happens and either the VM product is blamed when the same problem would occur on raw hardware.

Also a virtualized server is still a server, it's just virtualized. So just like you wouldn't do a panic shutdown on a hardware server running a database, don't just kill virtual servers, but take them down properly.

Many believe that MySQL Databases have too much overhead to virtualize. This depends on the virtualization solution you use:

- Its **False**, for hardware partitions and OS virtualization
- **Usually False**, for the latest baremetal hypervisors you're looking at a ballpark 10% performance hit for a properly configured system.
- **Usually True** for hosted hypervisors. The passing through both the hypervisor and the OS will be much slower.
- Its **False**, for Partitioned OS, these perform quite well.A

## 11 Conclusion

I hope this article provided a clearer picture into using virtualization technologies and MySQL together. There are certainly other interesting articles out there and we are putting efforts into more detailed white papers and references that deal specifically with MySQL and individual virtualization solutions, so look for those. Also, if you have a great virtualization success story we'd like you to share it so others can benefit, so please send me an email at [michael.frank@sun.com](mailto:michael.frank@sun.com)

Running MySQL on a virtual server is not right in many situations, but it is certainly has its advantages and its becoming more and more viable for a larger number of MySQL databases every day.

# 12 Links and References

**LDOMs Introduction**, Nick Klowki

[http://www.snpnet.com/customer\\_pub/sun/isv\\_LDOM/](http://www.snpnet.com/customer_pub/sun/isv_LDOM/)

**MySQL 5.1 Reference Manual Section 15.1 Common Issues with Virtualization**

<http://dev.mysql.com/doc/refman/5.1/en/ha-vm-commonissues.html>

**MySQL Virtualization Forum**

<http://forums.mysql.com/list.php?149>

**MySQL 2006 Quick Poll on Virtualization**

[http://dev.mysql.com/tech-resources/quickpolls/virtualization\\_software.html](http://dev.mysql.com/tech-resources/quickpolls/virtualization_software.html)

**Todor Ivanov's Weblog**

<http://ivanov.wordpress.com/2008/04/01/database-virtualization/>

**Database Systems on Virtual Machines: How much do You Lose?**

<http://www.cs.uwaterloo.ca/~ashraf/pubs/smdb08overhead.pdf>

**Database Virtualization: A New Frontier for Database Tuning and Physical Design**

<http://whitepapers.techrepublic.com.com/abstract.aspx?docid=289253>