
# Cost-Effective Oracle

*Getting the most from your Oracle RDBMS*

*Jay Stanley*
*Database Specialists*
*<jstanley@dbspecialists.com>*

## Preface

These opinions are my own; they may or may not be shared by my employer, Database Specialists, and I take sole responsibility. These ideas come from the 15 years that I've been working as a programmer and database administrator on the Oracle RDBMS.

## Introduction

There are quite a few companies, given the current global financial situation, that are investigating ways of lowering the cost of running their business using an Oracle database.

This paper does not involve Oracle Applications or Oracle Financials; they are strictly tips on how to get the most out of the Oracle RDBMS itself.

## Tip 1: Understand the amount of money spent on the software license, -vs- the hardware used, including both up-front and recurring costs, over the life of the database

Typically, the cost of an Oracle license is far, far greater than the cost of hardware, hardware support, the operating system, and operating system support, especially over the life of the installation.

So, it makes sense to spend more on hardware, and try to limit the cost of the Oracle license itself. Do an analysis that includes all of the factors involved, and include any others that may be particular to your case.

Generally, the separate items you pay for include:

- The hardware that will run the database, including any networking required,
- The electricity needed to run that hardware,
- The space that that hardware takes up (especially if in a co-located rented space)
- Hardware support from the manufacturer (ie Dell, HP, IBM, Oracle/Sun)
- The operating system itself (Red Hat Linux, AIX, Oracle Enterprise Linux, HP/UX, Windows)
- Operating system support (this includes security updates and bug fixes)
- Any external storage + replacement disks/associated hardware as they fail,
- Support for external storage,
- Electricity and rack space for external storage,

 

- The Oracle database software licenses, additional option packs (e.g. RAC, tuning pack), and Oracle Support for help and patches as required,
- The administrators (system administrators, network administrators, programmers, database administrators) to support the system.

Look at these costs over the *entire timespan of your target product*.

When I have done these exercises in the past, excluding human resources, frequently the cost paid for the RDBMS license to Oracle is more than 70-80% of the total.

## Tip 2: Buy the cheapest version of Oracle that will do the job

Oracle has several versions of its venerable RDBMS; they range in price from free, to quite expensive. Carefully analyze which version makes the most sense for you; in fact, if you haven't designed your application yet, you may be able to target your product to one of the cheaper versions.

The different versions are at the time of this writing:

[http://www.oracle.com/us/products/database/product-editions-066501.html?ssSourceSiteId=otnen]

| Edition | CPU Limit | RAM limit | Database size | Notes |
|---|---|---|---|---|
| Express Edition | 1 | 1Gb | 4GB | Free; Only on Windows, Linux, quite limited |
| Standard Edition One (SE1) | 2 sockets | OS max | No limit | Cannot use advanced features |
| Standard Edition (SE) | 4 Sockets | OS Max | No limit | Cannot use advanced features |
| Enterprise Edition (EE) | No limit | No limit | No limit | Extra cost for each advanced option |

Note that "Sockets" is the *maximum number of processor sockets that are available on the server*, regardless of whether the sockets are filled. It doesn't make financial sense, in terms of the Oracle license, to buy a server that has empty sockets, unless there's a special agreement with Oracle.

The nice thing about these different versions, is that it is usually quite easy to upgrade. On the other hand, downgrading, say from Enterprise Edition to Standard Edition, usually involves quite a lot of downtime, and frequently a full export/import of all of the data, with associated downtime.

Unless your application is very, very small, Express Edition may not give you what you really need; be wary.

 

Standard Edition One will work extremely well in many applications today. It offers configurations up to 2 sockets, which with processors today can many times be more than sufficient. You can buy SE1, and can have an unlimited amount of RAM, and unlimited amount of storage and whatever quality of storage with this license. You can even use ASM (Oracle Automatic Storage Management) with SE1, and this combination can be very cost effective.

Standard Edition also works very, very well, especially if you don't need any of the advanced options, and need more CPUs than SE1, though note that parallel operations are not available; that's only enabled in EE. Standard Edition is usually 1/3 of the cost of Enterprise Edition.

Enterprise Edition has a lot going for it, especially when purchased with some of the more-used extra options; however, it's usually about three times (300%) the cost of Standard Edition. Of course, this ends up being not only a purchase cost, but a recurring support cost as well for the life of the database.

There are quite a few advanced features that can be enabled only for Enterprise Edition, which may be absolutely required by some applications. For example, if you have a product that is under constant development with fast release cycles, and really requires constant tuning, then the Enterprise Edition with the SQL Tuning Pack extra-cost option may save you time in the long run; similarly, for compliance reasons you may need the Advanced Security option. The Partitioning option is also quite popular.

Enterprise Edition has quite a few features that make disaster recovery better as well, such as tablespace point-in-time recovery, and the ability to have a physical or logical standby slaved in real-time with the primary database. There are many other advanced features as well; too many to include in this presentation.

If you are not sure, ask your Database Administrator for help.

## Tip 3: Limit the number of CPU sockets in the database server to save licensing costs; investigate the named-user-plus license

There are several plans which Oracle offers; named-user-plus and processor-based are two of them. Take a very close look at their current licensing agreements and discounts available here:

[http://www.oracle.com/us/corporate/pricing/specialty-topics/index.html]

If your application faces the internet with an unlimited number of end-users, you'll need to purchase the CPU license. The CPU license is affected by the number of sockets, or CPUs that you have. This means that you should get the fastest CPU you can, but also that you limit the number of sockets; a two-socket machine will cost double (200%) of the Oracle RDBMS license than a single socket in most cases.

If you are using a "Processor" based plan (for unlimited users), see if you can get by with a less powerful processor. This can save quite a bit of money.

## Tip 4: Don't use RAC unless you need it

Remember, when you use RAC, you have to buy a separate license for each box, and since this is very

 

likely the biggest-ticket item, you will pay *much* more when you use RAC. In addition, RAC is an additional option that is purchased in addition to the Enterprise Edition license.

RAC is useful for when you cannot buy a machine with enough CPUs or memory at any cost for your target application, or that cost is extremely prohibitive; it allows you to scale your application horizontally. The main successful applications I've seen are generally data-warehouse type applications with extremely large data volumes.

In my experience:

- Buying RAC for two nodes of one processor apiece is usually far more expensive than buying a non-RAC license for one two-processor box.
- Buying a machine with 8 processors will cost less in overall costs than buying 4 servers with two processors apiece.
- In fact, buying RAC for any number of nodes is more expensive than buying one box that has the same amount of processing power/memory.
- Your database will run faster on a single box with say 4 processors and 64G of RAM, than it will on four boxes with 1 processor apiece and 16G of RAM apiece, and you'll save on licensing costs as well.

Oracle RAC uses the memory in all of the instance nodes in order to cache database blocks for queries – this is called 'cache fusion' by Oracle. When one instance requires a read-consistent block, it requests it from other nodes, to see if it is already in memory. If the block is in memory on another node, the node which has the block will move it over the interconnect to the instance which requested it. The speed of the network interconnect between machines (used to move blocks between instances in a RAC configuration) is an order of magnitude or more slower than the speed of inter-processor communication on a single machine with multiple processors. In terms of obtaining a read-consistent block, a machine with all CPUs together will be able to obtain the block an order of magnitude faster than obtaining that block over an interconnect.

Current CPUs today are far, far faster today than they have before in history; most database servers today are oversized in terms of processing power. If using the 'unlimited users/CPU' license structure, RAC therefore is far more expensive in licensing costs than a single machine with multiple processors.

In 2001, when Oracle first released RAC, servers were far, far less powerful than they are today. At the time, it was relatively easy to max out the amount of CPU and memory bandwidth used by an Oracle database. For example, in 2000 the fastest Intel processor (Pentium III) did about 2,054 MIPS. Today, the fastest Intel processor, the Intel Core i7 875k does 92,100 MIPS (https://secure.wikimedia.org/wikipedia/en/wiki/Instructions_per_second) – high-end processors today are roughly 60 times faster than they were in 2001. Memory bandwidth and bus speed has increased quite a bit as well, which is a major part of the equation. In 2000, it was extremely expensive to buy multi-processor computers; today it is the norm, with some single sockets even boasting 64 processors, and even most laptops, and even smartphones, built with multiple CPUs in single sockets. It is actually quite rare in the authors experience to see CPU maxed out on a current high-end box except in very special applications; the author has witnessed single non-RAC machines being able to do in excess of 2 million logical reads per second! For this reason, with today's high-end hardware, RAC really doesn't

 

make much sense when viewed in a cost-benefit, accounting based analysis for most applications.

Some people consider RAC more reliable, but in my experience it most certainly is not; there is still a single-point of failure, that being the storage subsystem, and the complexity of RAC makes it far less bug-free. It is true, though, that if a single node in a RAC cluster does go down, and the storage/interconnect are not affected, then it does allow some failover ability. Database sessions that were connected to the bad node can reconnect, and if the underlying application is designed for it, can do so semi-transparently (using TAF); they can typically reconnect far faster than using a standby database or using dataguard under Enterprise Edition. So, RAC does help protect against CPU/Memory/Network issues, but it does not protect against physical or logical storage corruption.

Note that in order to really use RAC, you will be required to buy the Enterprise Edition. The only Standard Edition RAC setup supported is very small; only two nodes can exist, and both of them are limited to one socket apiece.

Unless you really don't care about uptime, never invest in RAC if you haven't already invested in a standby database; a standby is much more important. Also, if you are considering really using RAC to its fullest, e.g. using 'instance fencing' or service-based connection types, remember that to reliably implement these, you will need a setup with the same number of nodes for your standby database, as well as for your development/QA team(s), as you are using in production.

Using RAC does require a higher level of skills from your DBA team.

RAC can provide more availability in some situations if properly configured, administered, and monitored, but it is not for everyone.

## Tip 5: Investigate Oracle Enterprise Linux (if considering Linux)

If considering Linux, be sure to check the pricing of Oracle Enterprise Linux (OEL) compared to Red Hat Enterprise Linux, and include support in there. This is a moving target, but each time I check, OEL is much cheaper than Red Hat Enterprise Linux, especially when support is figured in. There is also the added advantage that when something is wrong between the operating system and the RDBMS software, you can point your finger at one place for a solution: Oracle. Oracle also claims that there is improved performance using their kernel rather than the stock Red Hat Enterprise Linux supplied kernel. CentOS, the free alternative to Red Hat, is typically 9 months behind in both performance and security updates at the time of this article (October, 2011). It can save some money, but not much when compared with the actual cost of the Oracle license over the lifespan of the database, given the low cost of OEL.

## Tip 6: Do buy a boatload of RAM

Oracle *loves* RAM for most applications. Especially if you expect your application to scale, RAM becomes paramount. Oracle doesn't cost more if you have more RAM in your system. Remember also that, like faster CPUs, RAM is a one-time cost. It is also quite cheap (for most applications) to increase throughput compared to other options, e.g. adding processors. It is particularly good for applications which re-read a lot of data, which in real life is most of them.

 

## Tip 7: Do spend more money on top-quality storage

The Oracle license does not cost more if you have better storage.

In my experience, the main bottleneck most databases experience is almost never CPU; they are rarely CPU-bound. Instead, they are I/O bound, and in particular IOPS bound (input/output operations per second); that is to say, not limited by the bandwidth of the storage subsystem, but rather the latency of single block reads. For this reason, don't focus on the disk bandwidth, focus on IOPS. There is no other program that I know of that can stress storage as much as Oracle at high loads.

Saving money on disks will ultimately cost more, as there is no substitute for disk subsystems that can handle very high IOPS reliably.

Oracle provides a utility called 'Orion' which can help qualify storage subsystem choices. If you have the luxury of benchmarking them, use it.

Failing disks will happen; it is not "if" they will fail, it is "when". If your database has no way for the disk subsystem to recover from this (and you do not have a standby database), then your database will go down for awhile; the bigger the database, the longer it will be unavailable during recovery. With today's larger databases exceeding a petabyte, recovery can take days or weeks. Better storage subsystems will make it much easier/more transparent to trade out failed disks, and to add capacity.

Buying cheap disks, if the company scales, will ultimately cost a lot of money to replace. Buying not enough storage will ultimately cost a lot of money in Oracle administration time, juggling storage, and this obviously adds risk to the equation as well. Quality disk subsystems of sufficient capacity, while not cheap, are very inexpensive compared to trying to make do with inadequate solutions.

## Tip 8: Always use a standby database

If your application requires high availability, there is no substitute for a standby database: see [http://www.dbspecialists.com/blog/database-backups/need-uptime-use-and-oracle-physical-standby-database/].

It will double your costs, as you will need equivalent hardware for the standby, including Oracle licenses. If you are not using Enterprise Edition, failover will very likely be a manual operation, but it will still be an order of magnitude faster than restoring it from a backup unless the size is trivially small.

Investigate the cost if your application is down for the time it would take for a full recovery from a backup. Very frequently, this cost is far, far more than the cost of a standby database, including all of the costs mentioned in tip #1.

## Tip 9: Do buy Oracle Support

In my experience, not purchasing Oracle Support, is best described as "penny-wise and pound-foolish", even though it is a recurring cost that usually runs about 1/3 of the purchase price yearly for the life of the database.

If you don't require 24/7 access, or don't have the money for it, go with one of the cheaper support

 

options. Remember, Oracle releases security patches four times a year, and if you do not have support, you cannot download or legally apply them. It also releases bug fixes often, and again, if you have not bought support, you cannot legally apply those bug-fixes, or even discover them using the Oracle Support website.

Even the cheapest option gives you free patches and upgrades, which in reality are absolutely required. If you do not have support, and you run into an insurmountable software issue with Oracle, you will not be able to fix it, period. Buying emergency support from Oracle can be terribly expensive.

If you are using the current version of Oracle, released within the past 6 months to a year, you will most certainly see benefit from purchasing Oracle support.

This year (2011), Oracle has really tightened their release and support cycle; as of July, you cannot download or apply the latest security patches for version 10, unless you have purchased "extended support", which is an extra-cost option. This means in practice that release cycles are and will be shorter than they have been in the past, and the length of time that you can get Oracle support for a version without paying extra for the "extended support" option is shortened compared to what it was in previous years. This will encourage the adoption of the current version of Oracle; and without purchased support for it, you may well find the database to have bugs which can affect production in a very fundamental way.

## Tip 10: Engage Oracle Sales

The Oracle Sales team is well-known for being able to cut a deal.

In my experience, especially if your company is a new one using Oracle technology, there is a good chance that you can negotiate lower prices. Some of my colleagues have mentioned that engaging them at the end of a sales quarter is productive, but I have no hard evidence of this.

If your company runs a significant number of databases, there too, you might be able to cut a deal.

You will never know until you try, and in the worst case, you won't pay more than the published values.

## Tip 11; if your application is in-house, engage your development team

There are smart ways of using an Oracle database in an application. There are also many ways of using it can dramatically increase the demands on the database. If the application is predicted to grow a lot; to scale, then it's very important to not use the database for things that it doesn't need to be used for. Develop metrics that can measure how much effort the database needs to expend to satisfy a given requirement; if these are tracked and provided to the development team in an easy and timely fashion, it can make quite a big difference in the machine capacity required, and therefore the ultimate licensing costs.

If the application is in the development phase, it may be possible to target one of the cheaper licenses (SE1 or SE) instead of Enterprise Edition. If you become used to the EE extra features, it can be quite difficult to eliminate their usage.

Be careful what you use Oracle for – Oracle is among the best RDBMSs out there, but many problems

 

just don't require ACID or relational algebra to solve. Using it as a syslog/application log destination for your data center, including web logs, may not be the best use of it. Using it to store and report various performance metrics of machines in your datacenter may be convenient, but it will increase the amount of work your database has to do – investigate RRDtool, or similar specialty databases. If you are trying to use it to store documents, especially with schemas that evolve often, investigate Hadoop or a more appropriate document-centric data-store solution. If you're using it for a graph database with nodes/edges/properties, again, investigate a more appropriate datastore. Using Oracle as a key-value store usually is vastly over-engineered; look into Oracle BDB, or one of the other well-known key-value datastore databases. Using your Oracle RDBMS for the parts of the application that really require ACID, work in sets of data, and require full transactional support will usually reduce the amount of CPU you need to buy, and therefore the licensing costs.

Also, place your business logic on servers that are not on the database, unless they need to gather lots of data to perform their function – again, this will reduce the CPU requirements of your database server, and therefore your Oracle licensing costs.

## Tip 12; be wary of virtualization today

If you are using the Oracle Unlimited user license, which is licensed by CPU, be wary of using virtualization, unless it is the Oracle VM product. If you are using a few larger servers to host your database and application servers, and not using Oracle VM, note that the license language, as it stands today, requires you to purchase the license based on the total number of CPU/sockets that are in use in the machine, even if you are assigning only a limited number to the Oracle virtual machine itself. In other words, if you have a 16-processor box, with a virtual machine (VM) within it running the Oracle database, you will need to pay a license based on all 16 of those processors/sockets. This may make sense technically, but from an accounting standpoint it does not, at least at the time of the writing of this paper.

The Oracle VM product itself does look quite interesting; it is a type-1 hypervisor, and you can license it for just the subset of CPUs on the machine that you dedicate to the Oracle Database; this is unlike licensing any other VM product at the time of this publication. The main downside of using Oracle VM, is that so far, it's quite rare to see companies using it in the author's experience.

## Tip 13; Hire the best database administrator(s) that you can

A good, experienced database administrator will save a lot of money over the life of your product. An inexperienced database administrator will take far longer to complete tasks, especially during critical times, and will be far more likely to make a 'fatal' mistake. Like much of the computer industry, the difference in productivity can be one to two orders of magnitude when comparing DBAs with little experience, with those that have many years of experience.

Of course, the trouble with hiring and retaining good Oracle database administrators is that they typically cost more per hour, they are hard to find and retain, and many times you really don't require a full-time DBA. They are a scarce resource. The best database administrators that I've had the opportunity to work with have invariably been ones with many years of experience.

 

If you are having problems finding a qualified applicant, one option might be to engage us here at Database Specialists. You will gain access to an organization that is top in the field. You can customize the number of hours that you require for your business, and renegotiate the number of hours you use monthly. In addition, we offer unparalleled 24/7 deep monitoring of your database, reviewed by experts, so that any issues that begin to happen can be mitigated quickly.

We pride ourselves here at Database Specialists in having the most experienced, most highly trained, most informed, most cost-effective, and fastest database administrators in the business. Feel free to give us a call for your requirements; at the end of the year, the total cost to your business will be far less than hiring less qualified individuals or companies.

## Conclusion

I've collected my thoughts after my many experiences over the years. This is in no way a complete list of ways that you can save money, but it should give food for thought before your purchase decisions are made.

## About Database Specialists, Inc

About Database Specialists, Inc.

Database Specialists, Inc. provides remote database administrator (DBA) services and onsite database support for your mission critical Oracle systems. Since 1995, we have been providing Oracle database consulting in Solaris, HP-UX, Linux, AIX, and Windows environments. We are DBAs, speakers, educators, and authors. Our team is continually recognized by Oracle, at national conferences and by leading trade publications. Learn more about our remote DBA support, Oracle database administration, and dba outsourcing services by visiting our website, or call us at **415-344-0500** or **888-648-0500**.

## About the Author

Jay Stanley began his IT career in 1986 with Xerox, specializing in data analysis, system administration, programming and database administration. Since that time, he's held a variety of IT jobs in database administration, programming/development, and systems administration. He has been working with Oracle databases since 1995, and has been an Oracle Certified Professional (OCP) since 1997 (7.3.4, 8.0, 8i, and v10). Since then, he has worked with a very wide variety of businesses, databases, and applications, ranging from small web-based startups, to medium-sized companies, to Fortune 500 multinational companies.

Today he works as a Senior Database Consultant for Database Specialists, Inc. He considers his colleagues at Database Specialists to be the best Oracle database administrators in the business.