

WHITEPAPER

**INGRES**

# VECTORWISE

## Simply FAST

A technical whitepaper

**INGRES**

**vectorwise**



## TABLE OF CONTENTS:

Introduction .....	1
Uniquely fast – Exploiting the CPU.....	2
Exploiting Single Instruction, Multiple Data (SIMD) .....	2
Utilizing CPU cache as execution memory .....	3
Other CPU performance features .....	3
Leveraging industry best practices .....	4
Optimizing large data scans .....	4
Column-based storage .....	4
VectorWise’s hybrid column store .....	5
Positional Delta Trees (PDTs) .....	5
Data compression .....	6
VectorWise’s innovative use of data compression .....	7
Storage indexes .....	7
Parallel execution .....	8
VectorWise use cases .....	8
Financial services.....	9
Social media.....	9
E-commerce .....	10
Conclusion .....	10



## INTRODUCTION

There is no lack of data. Internet users and devices generate more and more each day. Companies and organizations recognize the need to analyze data, whether it is data generated by business processes or public data. Companies and organizations create data warehouses and data marts in relational databases to store and analyze terabytes of data.

The market for relational database solutions for data warehousing or data marts has evolved rapidly over the last few years. Multiple purpose-built products are available for reporting, data analysis and Business Intelligence. Some product offerings are available only as a hardware and software combination – a data warehouse appliance – whilst others are software-only solutions that support a variety of hardware installations.

VectorWise is relational database software for reporting, data analysis and Business Intelligence. VectorWise exploits performance features in today's x86 CPUs that most other relational databases do not take advantage of. As a result VectorWise can process data much faster than most other relational databases.

Much faster data processing performance opens up opportunities. Think not only about support for larger data sets, more users and more complex workloads, but also about the ability to directly query detail data when previously query performance would only be acceptable after extensive indexing and materialization of intermediate results. Faster performance significantly reduces the amount of lag-time until you can first look at results, and faster performance increases flexibility in the ways you can access your data.

But there is more. VectorWise enables you to run a workload on a server when other databases require a much larger machine, a cluster of servers, or both, to achieve similar results. You can lower costs instantly by better utilizing your hardware and over time because you don't have to carefully tune the system with hard-to-find experts.

This paper explains why VectorWise achieves extremely fast performance for typical data warehouse and data mart workloads. But don't just read this paper – you should experience VectorWise in action in your own environment. Get your copy of a trial version today through <http://vectorwise.ingres.com>!

## UNIQUELY FAST – EXPLOITING THE CPU

VectorWise is unique because it takes advantage of powerful CPU features that most other databases don't.

During the past three decades CPU processing capacity has roughly followed Moore's Law<sup>1</sup>. In recent years however the improvements required to keep CPU data processing performance in line with Moore's Law are not just the result of increases in clock speed and the number of transistors on the chip. CPU manufacturers have also introduced numerous additional performance features in the last several years such as multi-core CPUs and multi-threading that are transparently leveraged by most database software.

There are however other optimizations that were introduced in the last decade that are typically not transparently leveraged by most database software. Examples include so-called SIMD<sup>2</sup> instructions and their associated instruction sets, large chip caches, out-of-order execution and hardware-accelerated string-based operations. In fact, most of today's database software that was originally written in the 1970s or 1980s has become so complex that in order to take advantage of these performance features a complete rewrite of the database software would be required.

VectorWise was written from the ground up to take advantage of performance features in modern CPUs, resulting in dramatically higher data processing rates compared to other relational databases.


### Exploiting Single Instruction, Multiple Data (SIMD)

SIMD enables a single operation to be applied on a set of data at once. VectorWise takes advantage of SIMD instructions by processing vectors of data through the Streaming SIMD Extensions instruction set. Because typical data analysis queries process large volumes of data the use of SIMD may result in the average computation against a single data value to take less than a single CPU cycle.

---

<sup>1</sup> Moore's law describes a long-term trend that the number of transistors that can be placed inexpensively on an integrated circuit doubles roughly every two years. See [http://en.wikipedia.org/wiki/Moores\\_Law](http://en.wikipedia.org/wiki/Moores_Law). Although Moore's Law specifically talks about the number of transistors it is casually used to describe technology improvements that double performance every two years.

<sup>2</sup> SIMD stands for Single Instruction, Multiple Data. Traditionally CPUs would process using a SISD model: Single Instruction, Single Data. For more information see <http://en.wikipedia.org/wiki/SIMD>.



At the CPU level traditional databases process data one tuple at a time spending most of the CPU time on overhead to manage tuples and not on the actual processing. In contrast VectorWise processes vectors of hundreds if not thousands of elements at once which effectively eliminates these overheads. As a result the CPU resources are used to perform the actual work.

## Utilizing CPU cache as execution memory

The majority of the improvements to database server memory (RAM) over the last number of years have resulted in much larger memory pools, but not necessarily faster access to memory. As a result, relative to the ever-increasing clock speed of the CPU, access to memory has become slower and slower over time. In addition, with more and more CPU cores requiring access to the shared memory pool contention can be a bottleneck to data processing performance.

In order to achieve maximum data processing performance VectorWise avoids the use of shared RAM as execution memory. Instead VectorWise uses the private CPU core and CPU caches as execution memory delivering significantly greater data processing throughput.

## Other CPU performance features

On an ongoing basis the VectorWise Development team looks for ways to improve data processing performance using modern chip technology.

For example recent Intel chips support hardware-accelerated string-based operations. VectorWise exploits these<sup>3</sup>. Operations that benefit from the hardware-accelerated string-based optimizations include selections on strings using wild card matching, aggregations on string-based values and joins or sorts using string keys. However, not all modern CPUs support hardware-accelerated string-based operations and VectorWise also works fine – just a little less optimal – if this performance feature is not available.

---

<sup>3</sup> For more information and performance results see the Ingres VectorWise Sneak Preview on the Intel Xeon® Processor 5500 series-based platform at <https://www1.vtrenz.net/imarkownerfiles/ownerassets/978/IngresVectorWisePreview-WP.pdf>



## LEVERAGING INDUSTRY BEST PRACTICES

Various specialized data warehouse products use a number of well-known techniques to achieve fast performance. In general, because of the data-intensive nature of a data warehousing workload, most techniques focus on limiting and optimizing Input/Output (IO).

For VectorWise – because of its much higher per CPU core data processing power – to limit IO is an absolute requirement to achieve good data processing performance. VectorWise implements industry best practices to limit IO whilst it introduces innovations to overcome some of the traditional weaknesses associated with these techniques.

### Optimizing large data scans

The use of large data blocks makes sense for data-intensive applications such as data warehouses or data marts. To maximize bandwidth a single IO should retrieve as much relevant data as possible. Disks, in particular spinning disks, provide much higher IO bandwidth if data is stored sequentially as opposed to randomly on disk. The use of large data blocks ensures that IOs are performed as efficiently as possible, maximizing the IO bandwidth a disk provides.


### Column-based storage

When relational database software was first written it implemented so-called row-based storage: all data values for a row are stored together in a data block (page). Data is always retrieved row-by-row, even if a query only accesses a subset of the columns. This storage model works very well for On-Line Transaction Processing (OLTP) systems in which data is stored highly normalized and tables are relatively narrow, queries often retrieve very few rows and many small transactions come through.

Data warehouse databases are different:

- Tables are often (partially) denormalized resulting in many more columns per table, not all of which are accessed by most operations.
- Most queries retrieve very many rows.
- Data is added through a controlled rather than ad-hoc process and often large data sets are added at once.

As a result of these differences a row-based storage model typically generates a lot of unnecessary IO for a data warehouse workload. A column-based storage model, in



which data is stored together in data blocks (pages) on a column-by-column basis, is generally accepted as a superior storage model for data analysis queries.

Column-based databases have been available commercially for more than a decade. In addition to the benefit of data elimination when accessing fewer than all table columns in a query an additional significant advantage of column-based storage is better data compression. One of the biggest challenges with most column-based databases is incremental small inserts, updates or deletes (as opposed to large bulk data load operations).

#### ***VectorWise's hybrid column store***

VectorWise implements a hybrid column store. The term that is used in the research world for the type of storage VectorWise uses is PAX<sup>4</sup>.

- By default data is stored using a pure column-by-column approach.
- For tables that are indexed on more than one column VectorWise stores the indexed columns together in a single data block (but within the block data is still stored column-by-column to optimize compression). In VectorWise you should only index a table if the table is predominantly accessed on this column or set of columns. I.e. the columns in the index would always be accessed together and there is no penalty for storing the columns together in a single data block.
- Row-based storage. The user may choose to store data row-by-row if data allocation for column-by-column storage requires too much up-front data allocation. The minimum allocation for a table that is stored using columnar storage is one data block per column. The choice for row-based storage can make sense for extremely wide tables or tables with relatively few rows. This functionality is currently available as a beta feature.

#### ***Positional Delta Trees (PDTs)***


VectorWise implements a fully ACID<sup>5</sup>-compliant transactional database with multi-version read consistency. Any new transaction will see all previously committed transactions, both small incremental transactions and large bulk data loads.

---

<sup>4</sup> PAX stands for Partition Attributes Across. For more information visit <http://www.pdl.cmu.edu/Database/index.shtml>

<sup>5</sup> ACID is an abbreviation that stands for Atomicity, Consistency, Isolation, Durability – a set of properties that guarantee database transactions are processed reliably. For more information visit <http://en.wikipedia.org/wiki/ACID>.





Irrespective of the actual choice of data storage VectorWise uses Positional Delta Trees (PDTs) to store small incremental changes (inserts that are not appends as well as updates, and deletes (except truncates)).

Conceptually a PDT is an in-memory structure that stores the position and the change (delta) at that position. Queries efficiently merge the changes in PDTs with data stored on disk. Because of the in-memory nature of PDTs small DML statements can be processed very efficiently.

PDTs only use a configurable amount of memory. Once the memory pool is exhausted VectorWise automatically triggers an event (COMBINE) to write PDT changes to persistent storage. Depending on the total table size this can be an expensive operation since the trigger effectively rewrites the entire table to optimize data storage. You should bulk-append data whenever possible as opposed to individually inserting it, or at least consider manually triggering the COMBINE event at a point in time that works well for your application.

## Data compression


Most relational databases support some sort of data compression, and so does VectorWise. VectorWise compresses data on a column-by-column basis using any one of the following algorithms, or a combination of those:

- Run Length Encoding (RLE)<sup>6</sup>: a data value is stored as well as the number of subsequent values that are the same. This compression algorithm is very efficient on ordered data with relatively few unique values.
- Patched Frame Of Reference (PFOR): a base value is determined per data block and other values in the same block are encoded by storing the difference with the stored value using as few bits as possible. This is beneficial as the range of the actual data is typically much smaller than the range of a used data type. What makes PFOR special compared to similar solutions found in other products is the treatment of outliers. For example, if 99% of values are in range 0..255, and 1% of values is very large (e.g. around a million), then with PFOR the vast majority of the data will be stored using only one byte, while other solutions would use 2.5 bytes.
- Delta encoding on top of PFOR: in order to reduce the values of the integers with PFOR it is sometimes more efficient to store the delta from the previous value. This can be very efficient on ordered data.

---

<sup>6</sup> See [http://en.wikipedia.org/wiki/Run-length\\_encoding](http://en.wikipedia.org/wiki/Run-length_encoding).



- 
- Dictionary encoding: stores pointers to a dictionary of unique values. This algorithm is very efficient for a limited number of very frequently occurring values. At this point in time this is the only algorithm supported for character data types in VectorWise.

The algorithms VectorWise uses to compress data have been selected for their speed of decompression over a maximum compression ratio. The compression ratio you can achieve with VectorWise is highly data-dependent and also somewhat data-load-dependent. 3-5x compression ratios are very common for real-world data but both lower and higher compression ratios have been observed.

#### ***VectorWise's innovative use of data compression***

In order to improve IO performance VectorWise allocates a portion of physical memory for a memory-based disk buffer, the Column Buffer Manager (CBM). Data is automatically pre-fetched from disk and stored in the CBM mirroring the data as it is stored on disk. I.e. in contrast to many other databases VectorWise does not decompress data in the memory buffer, but rather data is decompressed only once it is ready for data processing.

VectorWise automatically chooses the most optimal compression on a page by page basis (i.e. per column – multiple pages – there can be multiple different algorithms in use). Decompression comes at almost no cost because it is directly integrated in the vector-based processing. VectorWise's decompression is far more efficient than previous speed-optimized compression libraries such as LZOP that many other products have utilized.

## **Storage indexes**

Although not many relational database implement storage indexes VectorWise is not unique in its use of this type of indexes. VectorWise automatically maintains a storage index per column storing minimum and maximum values for the data block. The storage index is very efficient in determining whether a database block is a candidate block for a particular query either because of explicit filter criteria or implicitly as a result of processing table joins.

In extreme cases the storage index provides the same benefit as data partitioning does for other databases without the overhead of multiple database objects and having to design and maintain a partitioning strategy.



## Parallel execution

Almost all relational databases support some means for a single operation to take advantage of multiple CPU core resources. For some databases, particularly the pure Massively Parallel Processing (MPP) databases, the use of multiple CPU cores is a mandate and virtually every operation uses all CPU cores in the system. Other databases use some form of a shared architecture and therefore support a wider range of possible degrees of parallelism.

VectorWise does not currently provide a clustering option and all data can be accessed by all CPU cores in the system. To dramatically reduce the execution time of a query VectorWise can execute the statement in parallel using any number of CPU cores up to the number of cores in the server. At present the degree of parallelism setting can only be changed in the VectorWise configuration file and a database restart is required to use a different setting.

Parallel execution must be balanced with the intended number of operations that execute concurrently. During execution an operation occupies as many CPU cores as its degree of parallelism. You can submit operations beyond the number of available CPU core resources but they will be queued and wait for resources to become available. At present VectorWise executes statements in the order they were submitted, as resources become available.

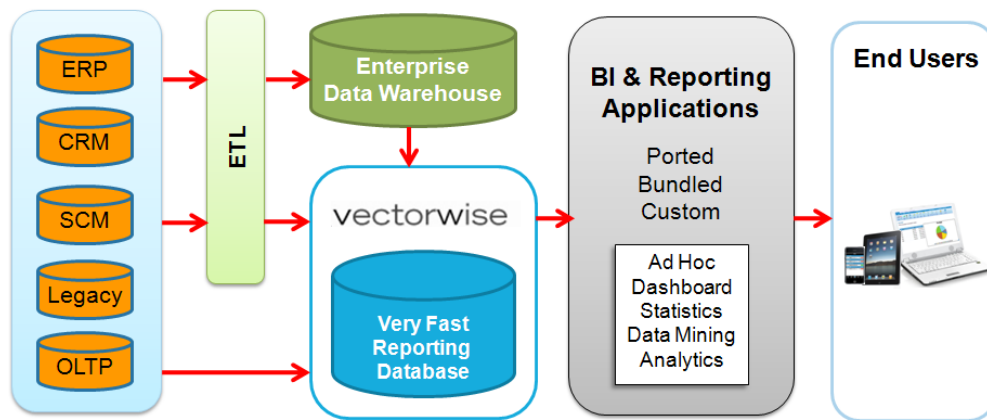
## VECTORWISE USE CASES

VectorWise provides relational database software that takes analytic data processing to a new level. With it, you can now achieve amazing performance with a simple, ANSI-compliant relational database – something that was previously only achievable with proprietary OLAP databases, and/or following lots and lots of careful design and tuning using complex features.

However VectorWise is still relatively new database technology. VectorWise does not provide all the rich feature/functionality that some of the databases with 20 or 30 years of history provide. At present you should not use VectorWise for multi-petabyte databases, nor should you use it for an active enterprise data warehouse that receives continuous updates and supports thousands of users.

Use VectorWise if you are looking for a relational database, supporting ANSI SQL and industry-standard JDBC/ODBC interfaces, that delivers extremely fast performance, is easy to use and cost-effective. VectorWise delivers performance faster

than popular in-memory type databases, without even having to load all data in memory and without the hard limit of memory availability. The diagram below shows a number of areas in which it makes perfect sense to position VectorWise.



**Figure 1. Positioning VectorWise.**

Following are examples of early adopters of VectorWise and their use of data-intensive applications.


## Financial services

A number of organizations in financial services chose VectorWise.

For example The Rohatyn Group, a Wall Street-based hedge fund focusing on emerging technologies, replaced a home-grown, in-memory database with VectorWise in order to continue to deliver at least as good as in-memory-like performance while not being limited to the total amount of memory. The analysts using the system had expressed a desire to query historical data as well as current positions and the data volume was simply too large to store cost-effectively in memory. VectorWise provides the in-memory performance, but now hundreds of millions of rows containing historical data are stored on-disk.

## Social media

Many social media websites are extremely popular and generate vast amounts of data about their users. Advertising is a commonly used approach to monetize user behavior. Nasza Klaza (<http://nk.pl>), a large social media site in Poland, selected



VectorWise to aggregate data in order to, amongst others, optimize its advertising strategy and maximize its profits.

## E-commerce

Data is core business for e-commerce organizations. GSI commerce, a data aggregator, ran into the limits of its existing database technology and decided to evaluate VectorWise. VectorWise not only improves the service GSI commerce delivers to its customers, but the company can now also offer access to up to 5 years of data. The previously used technology posed significant performance challenges with much less historical data.

## CONCLUSION

VectorWise is the first relational database that focuses on processing efficiency in modern CPUs. Its vector-based processing as well as other optimizations directly leverage improvements in modern chips.

In order to maximize performance the entire underlying database architecture is designed to eliminate any potential bottlenecks that would limit CPU processing. Column-based storage, data compression and smart storage indexes are all means to achieve this goal. In addition parallel execution can squeeze the absolute maximum performance out of a system.

If you need to analyze large volumes of data and you don't want to take the risk of an expensive or lengthy implementation project then you should evaluate VectorWise. Implement simpler and cheaper solutions, and benefit from better query performance than other relational databases.

VectorWise is the foundation for revolutionary performance gains in database processing. You should try VectorWise today, but rest assured that there is more to come! Future versions of VectorWise will not only introduce new functionality, but also continue to leverage CPU performance features and implement other optimizations to get absolute maximum query performance.

For more information and to download an evaluation version, go to <http://vectorwise.ingres.com>.



## About Ingres Corporation

---

Ingres develops and markets the leading open source enterprise-grade relational database and a breakthrough analytic database, VectorWise, designed to support the growing need for analytics and business intelligence. VectorWise customers claim performance improvements of up to 70X that of comparable databases, without the traditional overhead. The product is suitable for SMBs and enterprises and as an embedded database for ISVs and SaaS providers. VectorWise is quickly becoming a leader in interactive reporting and analysis for data-driven companies. More information is available at [www.ingres.com](http://www.ingres.com)

---

Ingres Corporation  
500 Arguello Street, Suite 200  
Redwood City, California 94063  
USA  
Phone: +1.650.587.5500

Ingres Europe Limited  
215 Bath Road  
Slough, Berks SL 1 4AA  
United Kingdom  
Phone: +44 (0) 1753 559550

Ingres Germany GmbH  
Ohmstrasse 12  
63225 Langen  
Germany  
Phone: +49 (0) 6103.9881.0

Ingres France  
7C Place Du Dôme  
Immeuble Elysées La Défense  
92056 Paris La Défense Cedex  
France  
Phone: +33 (0) 1.72.75.74.54

Ingres Australia  
Level 8, Suite 1  
616 St. Kilda Road  
Melbourne, Victoria, 3004  
Australia  
Phone: +61 3 8530.1700

For more information, contact [ingres@ingres.com](mailto:ingres@ingres.com)

©2011 Ingres Corporation. All rights reserved. Printed in the U.S.A. Ingres is a trademark of Ingres Corporation in the United States and in other countries. All other trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.