Introducing Teradata Columnar

Prepared by
William McKnight
www.mcknightcg.com

A guide to help the technical manager learn and use Teradata Columnar

Sponsored by TERADATA.
Teradata.com

Contents

Columnar Database Structure 2

Teradata 14 Hybrid Columnar 5

Mixing Columns in Containers 6

Partition Elimination and Columnar 7

Compression in Teradata Columnar 7

- Run-Length Encoding 8
- Dictionary Encoding 8
- Delta Compression 9

In Conclusion 10

About The Author 10

Columnar Database Structure

All database management systems constantly tinker with the internal structure of the files on disk. Each release brings an improvement or two that has been steadily improving analytic workload performance. However, few of the key players in relational database management systems (RDBMS) have altered the fundamental structure of having all of the columns of the table stored consecutively on disk for each record. The innovations and practical use cases of "columnar databases" have come from the independent vendor world, where it has proven to be quite effective in the performance of an increasingly important class of analytic query.

These columnar databases store data by columns instead of rows. This means that all values of a single column are stored consecutively on disk. The columns are tied together as "rows" only in a catalog reference. This gives a much finer grain of control to the RDBMS data manager. It can access only the columns required for the query as opposed to being forced to access all columns of the row. It's optimal for queries that need a small percentage of the columns in the tables they are in but suboptimal when you need most of the columns due to the overhead in attaching all of the columns together to form the result sets.

Columnar databases ensure the proper assignment of the column values to the other columns in the same row by position. For example, the fourth values in the first name column and the last name column are both for the fourth record. This way, "McKnight" gets attached to "William" instead of "John" when the result set is presented. This positional representation eliminates column-value-level metadata overhead on most column data pages, which further makes the reads more precise to the needs of the query.[1] The high likelihood of the same value being referenced successively on disk also opens up a variety of possibilities for compression.

Not only is the DBMS now operating only on data that is clearly interesting to the query, it is becoming more efficient in the use of the input-output (I/O) function. I/O has become the primary bottleneck in analytic queries, taking that distinction from the CPU long ago. The idle CPUs in many data warehouses are largely the result of congestion in the pre-processing layers of disk, memory, and cache. Complete rows being sent up the pre-processing layers often causes the cache layer(s) to be skipped due to its limitations on what it can be processing at any point in time. The cache adds a lot of processing value to the CPU operation and removing this expensive component from processing would not be a desired goal of any system administrator. However, these shortcomings are inevitable when complete rows are implicated in I/O, as is the case with row-based DBMS.

Architectural recommendations for columnar databases place them at the same architectural tier as the data warehouse itself due to the similar workloads supported by the two structures. As the *power of performance* becomes understood by an organization, it will do everything it can to speed up data access, which allows for deeper analysis and an increased likelihood of getting to root causes in organizational performance. This recommendation to supplement the data warehouse with a columnar structure usually involves crossing a chasm of technologies, skills and culture as well as a reduced expectation level regarding the robustness, utility and manageability of all implemented solutions. Columnar DBMS, developed for a specialized set of queries, tend to trail the row-based DBMS in many of the technical architecture requirements IT management has come to depend on. See Figure 1.

1 As well as creating space for more actual values on the page, further improving the I/O.

Scalable – The solution should be scalable in both performance capacity and incremental data volume growth. The solution should scale in a near-linear fashion and allow for growth in database size, the number of concurrent users, and the complexity of queries. Understanding hardware and software requirements for such growth is paramount.

Powerful – The platform should be designed for complex decision support in an advanced workload management environment. The optimizer should be mature enough to support every type of query with good performance. Determine the best execution plan based on changing data demographics. Check on conditional parallelism and the causes of variations in the parallelism deployed, and on dynamic and controllable prioritization of resources for queries.

Manageable – The solution should be manageable with minimal support tasks requiring DBA/System Administrator intervention. There should be no need for the proverbial army of DBAs to support an environment, and the system should provide a single point of control to simplify administration. You should be able to create and implement new tables and indexes at will.

Extensible – Look for flexible database design and system architecture that keeps pace with evolving business requirements and leverages existing investment in hardware and software applications. Know the answers to questions such as: What is required to add and delete columns? What is the impact of repartitioning tables?

Interoperable – The system should have integrated access to the Web, internal networks, and corporate mainframes.

Recoverable – In the event of component failure, the system must continue providing value to the business. It also should allow the business to selectively recover the data to points in time – and provide an easy-to-use mechanism for doing this quickly.

Affordable –The proposed solution (hardware, software, services) should provide a relatively low TCO over a multi-year period.

Flexible – The system should provide optimal performance across the full range of normalized, star, and hybrid data schemas with large numbers of tables. Look for proven ability to support multiple applications from different business units, leveraging data that is integrated across business functions and subject areas.

Robust In-Database Management Systems Features and Functions – Make sure there are DBA productivity tools, monitoring features, parallel utilities, robust query optimizer, locking schemes, security methodology, intra-query parallel implementation for all possible access paths, chargeback and accounting features, and remote maintenance capabilities.

Figure 1. Requirements for a technical architecture.

Teradata 14 Hybrid Columnar

The unique innovation by Teradata, in Teradata 14, is to add columnar structure to a table,[2] effectively mixing row structure, column structures and multi-column structures directly in the DBMS which already powers many of the largest data warehouses in the world. With intelligent exploitation of Teradata Columnar in Teradata 14, there is no longer the need to go outside the data warehouse DBMS for the power of performance that columnar provides, and it is no longer necessary to sacrifice robustness and support in the DBMS that holds the post-operational data.

A major component of that robustness is parallelism, a feature that has obviously fueled much of Teradata's leadership position in large-scale enterprise data warehousing over the years. Teradata's parallelism, working with the columnar elements, are creating an entirely new paradigm in analytic computing – the pinpoint accuracy of I/O with column and row partition elimination. With columnar and parallelism, the I/O executes very precisely on data interesting to the query. This is finally a strong, and appropriate, architectural response to the I/O bottleneck issue that analytic queries have been living with for a decade. It also may be Teradata Database's most significant enhancement in that time.

Teradata offers a true hybrid row and column database.

The Teradata Family Goes Hybrid Columnar

All of the Teradata Purpose-Built Platforms can benefit from Teradata Columnar features for specific analytical workloads. However, platforms can benefit differently based on their unique architectural differences. For example:

- **Teradata Active Enterprise Data Warehouse** – High compression could make solid state feasible for more data, taking performance to an even higher level.
- **Teradata Data Warehouse Appliance** – Reduced need for I/O improves performance – sometimes dramatically – of the typical query scan pattern on the Teradata Data Warehouse Appliance.
- **Teradata Extreme Data Appliance** – Columnar advantages, like compression and query performance, are magnified for the larger data sets usually placed in the Teradata Extreme Data Appliance.
- **Teradata Extreme Performance Appliance** – Fast becomes faster with columnar driving reduced I/O on this SSD platform, and compression allows more data to take advantage of the Teradata Extreme Performance Appliance.

2 And join indexes.

Mixing Columns in Containers

Many shops, with columnar databases, are carrying the same data in both row- and columnar DBMSs. Given the sensitivity of the platform to the workload, this is seen as a reasonable alternative to forcing all processing into one or the other. In particular, most corporate data that is kept in columnar databases is also kept in the row-based data warehouse for the general-purpose, many-column reporting that is typically done there. Teradata 14 allows the data

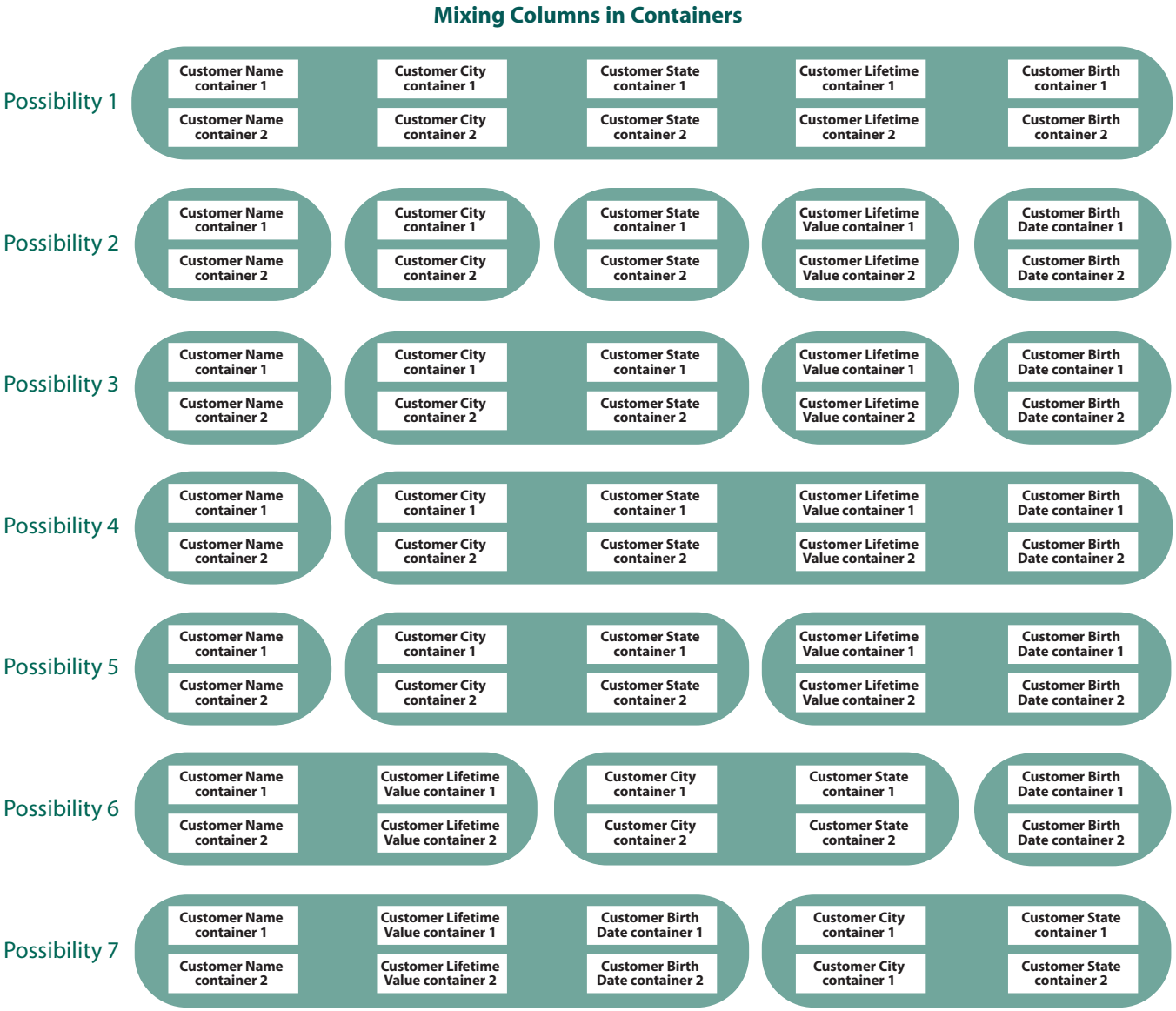

Figure 2. This figure shows the possibilities for a five-column table. Possibility 1 shows all columns grouped together in one container. Possibility 2 shows each column in a separate container. Possibility 3 shows one two-column container and three single containers, and so on through the remaining possibilities.

The physical structure of each container can also be in row (extensive page metadata including a map to offsets) which is referred to as "row storage format," or columnar (the row "number" is implied by the value's relative position).

manager to store some columns by themselves in a series of "containers"[3] and to store other columns with other columns in other containers. For the possibilities for a five-column table, see Figure 2.

It is still necessary to put a single column[4] into a single container and, as was mentioned, a container can have one or multiple columns. A column cannot be in multiple containers to take advantage of single- and multiple-column structures. When columns are usually accessed together, you could place them in the same container. This would eliminate the attaching process. When a table has a large number of columns, it is frequently a candidate for effective multiple-column containers. If the container is wide, Teradata Columnar will automatically use row storage format. Of course, if most of the row fits this criterion (accessed together), the concept of columnar may not apply to the table. I do, however, foresee valuable consideration work being given to pre-14 structures to assess the advantages of Teradata Columnar.

Partition Elimination and Columnar

The idea of data division to create smaller units of work as well as to make those units of work relevant to the query is nothing new to Teradata Database, and most DBMSs for that matter. While the concept is being applied now to the columns of a table, it has long been applied to its rows in the form of partitioning and parallelism. One of the hallmarks of Teradata's unique approach is that all database functions (table scan, index scan, joins, sorts, insert, delete, update, load and all utilities) are done in parallel all of the time. There is no conditional parallelism. All units of parallelism participate in each database action.

Teradata eliminates partitions from needing I/O by reading its metadata to understand the range of data placed into the partitions and eliminating those that are washed out by the predicates. See Figure 3.

Table partitioned by product_number (Part 1: 1-10, Part 2: 11-20, Part 3: 21-30, … Part 10: 91-100)

Select * from orders where product_number > 80

Partitions that need reading: 9, 10

Partitions "eliminated": 1-8

Figure 3. Basic partition elimination.

3 Unique physical structures

4 There are no restrictions on data types for Teradata Columnar.

There is no change to partition elimination in Teradata 14 except that the approach also works with columnar data, creating a combination row and column elimination possibility. In a partitioned, multi-container table, the unneeded containers will be virtually eliminated from consideration based on the selection and projection conditions of the query. See Figure 4.

Select product_number, product_name from product

Where size = 'large' and state_built = 'CA'

Selection containers = product_number, product_name

Projection containers = size, state_built

Figure 4. Selection and projection containers.

Following the column elimination, unneeded partitions will be virtually eliminated from consideration based on the projection conditions. For the price of a few metadata reads to facilitate the eliminations, the I/O can now specifically retrieve a very focused set of data. The addition of columnar elimination reduces the expensive I/O operation, and hence the query execution time, by orders of magnitude for column-selective queries. The combination of row and column elimination is a unique characteristic of Teradata's implementation of columnar.

Compression in Teradata Columnar

Storage costs, while decreasing on a per-capita basis over time, are still consuming increasing budget due to the massive increase in the volume of data to store. While the data is required to be under management, it is equally required that the data be compressed. In addition to saving on storage costs, compression also greatly aids the I/O problem, effectively offering up more relevant information in each I/O. Columnar storage provides a unique opportunity to take advantage of a series of compression routines that make more sense when dealing with well-defined data that has limited variance like a column (versus a row with high variability.) Teradata Columnar utilizes several compression methods that take advantage of the columnar orientation of the data. A few methods are highlighted below.

Run-Length Encoding

When there are repeating values (e.g., many successive rows with the value of '12/25/11' in the date container), these are easily compressed in columnar systems like Teradata Columnar, which uses "run length encoding" to simply indicate the range of rows for which the value applies.

'12/25/11' 134-267

Figure 5. Run-length encoding indicating the value of '12/25/11' applies to rows 134 through 267.

Dictionary Encoding

Even when the values are not repeating successively, as in the date example, if they are repeating in the container, there is opportunity to do a dictionary representation of the data to further save space. Dictionary encoding is done in Teradata Columnar by storing compressed forms of the

complete value. The dictionary representations are fixed length which allows the data pages to remain void of internal maps to where records begin.[5, 6] The records begin at fixed offsets from the beginning of the container and no "value-level" metadata is required. This small fact saves calculations at run-time for page navigation, another benefit of columnar.

For example, 1=Texas, 2=Georgia and 3=Florida could be in the dictionary, and when those are the column values, the 1, 2 and 3 are used in lieu of Texas, Georgia and Florida. If there are 1,000,000 customers with only 50 possible values for state, the entire vector could be stored with 1,000,000 bytes (one byte minimum per value).

Dictionary at beginning of container[7] :
1, Texas, 2, Georgia, 3, Florida, 4, null

Followed by column values:
1,3,2,3,1,4,1, …

Figure 6. Dictionary snippet showing state container on a data page.

In addition to dictionary compression, including the "trimming"[8] of character fields, traditional compression (with algorithm UTF8) is made available to Teradata Columnar data.

Delta Compression

Fields in a tight range of values can also benefit from only storing the offset ("delta") from a set value. Teradata Columnar calculates an average for a container and can store only the offsets from that value in place of the field. Whereas the value itself might be an integer, the offsets can be small integers, which doubles the space utilization. Compression methods like this lose their effectiveness when a variety of field types, such as found in a typical row, need to be stored consecutively.

The compression methods are applied automatically (if desired) to each container, and can vary across all the columns of a table or even from container to container within a column[9] based on the characteristics of the data in the container. Multiple methods can be used with each column, which is a strong feature of Teradata Columnar. The compounding effect of the compression in columnar databases is a tremendous improvement over the standard compression that would be available for a strict row-based DBMS.

Customer Name container 1
Abc
Def
Ghi

Customer Name Dictionary for container 1	
Abc	Smith
Def	Jones
Ghi	Johnson

State container 1		
WY	1	10
TX	11	25
FL	26	34

Birth Date container 1 Original = 07-04-1970
+73
-198
+38

Customer Name container 2
Jkl
Mno
Pqr

Customer Name Dictionary for container 2	
Jkl	Williams
Mno	Jones
Pqr	Miller

State container 2		
TX	1	6
GA	7	8
NB	9	13

Birth Date container 1 Original = 12-20-1965
+66
0
-532

Local Dictionary Compression **Run-Length Encoding** **Delta Compression**

5 Including variable-length records, for which the fixed-length dictionary representation is used; this maintains the fixed length nature of each value.
6 When "row storage" format is selected for a container, offset maps are still used within pages; this provides a quicker access path for individual row accesses.
7 Each container has a separate dictionary.
8 Trimming essentially means characters are treated as variable-length characters.
9 Referred to as "dynamic compression."

In Conclusion

With intelligent conversion of existing tables, and consideration for new tables for storage as columns, Teradata 14 customers will be able to take a giant leap forward in analytic workload performance. Starting by analyzing the workloads to be used with the data and focusing on column-specific workloads, then grouping columns accessed together, the foundation for table creation, with its automatic compression, is laid.

Advantages will be seen immediately in fewer storage needs, improvements in I/O-bound query performance, and scan operations.[10] Teradata customers who apply Teradata Columnar to existing data warehouses and data marts will find a fast path to relief of these limitations, which have almost been accepted as unyielding. Companies who were considering pairing a Teradata data warehouse with a columnar data mart can now do so as well with the Teradata family of products. The value proposition of Teradata Columnar holds across the product line, with unique degrees of advantage found in each product.

> Teradata has waited out the emergent field of columnar and has been able to superset the functionality, including in the area of compression.

Teradata's massively parallel platforms are built on a foundation that has served the largest and most complex environments in the world for 30 years. Teradata has increased the granular possibilities of the data storage, the compression and I/O utilization and the sensitivity of the optimizer with Teradata Columnar. This is Teradata's most significant enhancement in a decade, and "piles on" to an already extensive industry-leading set of capabilities. Teradata continues to show its leadership in meeting the needs of analytical environments.

About the Author

William functions as Strategist, Lead Enterprise Information Architect, and Program Manager for complex, high-volume full life-cycle implementations worldwide utilizing the disciplines of data warehousing, master data management, business intelligence, data quality and operational business intelligence. Many of his clients have gone public with their success stories. William is a Southwest Entrepreneur of the Year Finalist, a frequent best practices judge, has authored hundreds of articles and white papers and given hundreds of international keynotes and public seminars. His team's implementations from both IT and consultant positions have won Best Practices awards. William is a former IT VP of a Fortune company, a former engineer of DB2 at IBM and holds an MBA. William is author of the book, "90 Days to Success in Consulting."

10 The larger the scan and the larger the data, the more pronounced the advantages of Teradata Columnar.

Teradata is a registered trademark of Teradata Corporation and/or its affiliates in the U.S. and worldwide.