# Column Database: Solution to Data Warehousing Performance Problems?

Upasna Setia[1], Shruti Saxena[2]
[1,2]Department of Computer Science &Engineering,
Ganga Institute of Technology and Management, Kablana,
Jhajjar, Haryana, India

***Abstract:*** **It is widely accepted that a data warehouse is the central place of a Business Intelligence system. It stores all data that is relevant for the company, data that is acquired both from internal and external sources. Such a repository stores data from more years than a transactional system can do, and offer valuable information to its users to make the best decisions, based on accurate and reliable data. As the volume of data stored in an enterprise data warehouse becomes larger and larger, new approaches are needed to make the analytical system more efficient. This paper presents column-oriented databases, which are considered an element of the new generation of DBMS technology. The paper emphasizes the need and the advantages of these databases for an analytical environment and makes a short presentation of two of the DBMS built in a columnar approach.**

## I. INTRODUCTION

In the evolution of computing science, three generations of database technology are identified since the 60’s till nowadays.

The first generation started in the 60’s and its main purpose was to enable disparate but related application to share data otherwise than passing files between them. The publishing of “ A Relational Model of Data for Large Shared Data Banks ” by E. F. Codd marked the beginning of the second generation of DBMS (database management systems ) technology. Codd’s premise was that data had to be managed in structures developed according to the mathematical set theory. He stated that data had to be organized into tuples, as attributes and relations. A third generation began to emerge in the late 90’s and now is going to replace second-generation products. Multi-core processors became common, 64-bit technology is used largely for database servers, memory is cheaper and disks are cheaper and faster than ever before. A recent IDC study examines emerging trends in DBMS technology as elements of the third generation of such technology. It considers that, at the current rate of development and adoption, the following innovations will be achieved in the next five years:

- most data warehouses will be stored in a columnar fashion;
- most OLTP (On-Line Transaction Processing ) databases will either be augmented by an in-memory database or reside entirely in memory;
- most large-scale database servers will achieve horizontal scalability through clustering;
- many data collection and reporting problems will be solved with databases that will have no formal schema at all.

This study examines how some innovations in database technology field are implemented more and more. Most of these technologies have been developed for at least ten years, but they are only now becoming widely adopted. As Carl Olofson, research vice president for database management and data integration software research at IDC, said, “many of these new systems encourage you to forget disk-based partitioning schemes, indexing strategies and buffer management, and embrace a world of large-memory models, many processors with many cores, clustered servers, and highly compressed columnwise storage ”. From the innovations that the study considers that will be achieved in the next years, this paper presents the columnar data storage.

## II. NEED OF COLUMN ORIENTED DATABASE

Column-oriented databases are making a regular appearance on technology sites as a bit of a silver bullet for database performance issues.It is often presented and perceived as an evolution of database designs, much as was seen with the emergent NoSQL options (on the topic of NoSQL, some claim that MongoDB is column-oriented. It is in actuality moving in exactly the opposite direction).

In this case many seem to believe that column-oriented databases correct the mistake of row-oriented storage.

## III. DIFFERENCES BETWEEN the ROW-ORIENTED and COLUMN ORIENTED APPROCHES

- In row store data are stored in the disk tuple by tuple. Where in column store data are stored in the disk column by column

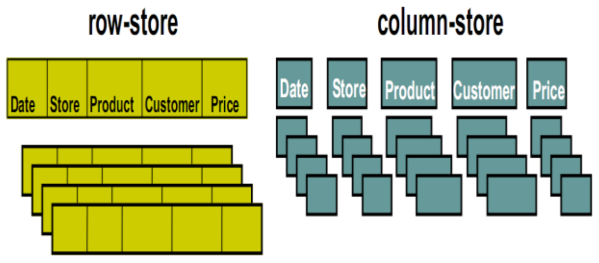

*Figure 1*

- Most of the queries does not process all the attributes of a particular relation.

  For example the query

  Select c.name and c.address

  From CUSTOMES as c

  Where c.region=Mumbai;

  Only process three attributes of the relation CUSTOMER. But the customer relation can have more than three attributes.

- Column-stores are more I/O efficient for read-only queries as they read, only those attributes which are accessed by a query.

| Row Store | Column Store |
|---|---|
| (+) Easy to add/modify a record | (+) Only need to read in relevant data |
| (-) Might read in unnecessary data | (-) Tuple writes require multiple accesses |

- So column stores are suitable for read-mostly, read-intensive, large data repositories

## IV. ADVANTAGE OF COLUMN ORIENTED DATABASE

Column-oriented databases provide important advantages towards the row- oriented ones, some of them being presented below. Column-oriented databases provide a better performance for analytical requests. In the row-oriented approach, the system performance decreases significantly as the number of simultaneous queries increases. Building additional indexes in order to accelerate queries becomes uneffective with a large number of diverse queries, because more storage and CPU time are required to load and maintain those indexes. In a column-oriented system indexes are built to store data, while in a row-oriented system they represent the way to point to the storage area that contains the row data. As a result, a column-oriented system will read only the columns required in a certain query. On the other hand, as they store data as blocks by columns rather than by rows, the actions performed on a column can be completed with less I/O operations. Only those attributes requested by users are read from disk. Although a row-oriented table can be partitioned vertically, or an index can be created for every column so it could be accessed independently, the performance is significantly lower than in a column-oriented structure [7]. And taking into consideration that I/O operations are the bottleneck of a database application, the column-oriented approach proves its superiority against the row-oriented one. Unlike the row-oriented approach, the column-oriented approach allows rapid joins and aggregations. Tables are already sorted, so there is no need to sort them before merge or join them. In addition, accessing data along columns allows incremental data aggregation, which is very important for BI applications. In addition, this approach allows parallel data access, improving the system performance. Thereby, complex aggregations can be fulfiled by the parallel processing of columns and then joining data in order to achieve the final result. Column-oriented databases need a smaller disk space to store data than row- oriented databases. To accommodate the sustained increase of volume of data, additional structures – as indexes, tables for pre-aggregation and materialized views, are built in row-oriented systems. Column- oriented databases are more efficient structures. They don’t need additional storage for indexes, because data is stored within the indexes themselves. Bitmap indexes are used to optimize data store and its fast retrieval. That’s why in a column- oriented database queries are more efficient than in a row-oriented one. Moreover, a higher data compression rate can be achieved in a column-oriented database than in a row-oriented one. It is well known that compression is more effective when repeated values are presented, and values within a column are quite similar to each other. A column-oriented approach allows the ability to highly compress the data due to the high potential for the existence of similar values in adjacent rows of a certain column. In a row-oriented database, values in a row of a table are not likely to be very similar; therefore, they cannot be compressed as efficient as in a column-oriented database.

## V. EXAMPLES OF COLUMN ORIENTED DATABASE SYSTEMS

Besides the column-oriented approach, another important innovation applied in data warehousing consists in the way in which data is processed. Two major techniques are used to design a data warehouse architecture: symmetric multiprocessing (SMP) and massively parallel processing (MPP). It couldn’t be certified the superiority of one approach against the other. Each of these solutions has its own supporters, because both of them are valid approaches and, when properly applied, lead to notable results.

Two database systems are presented in the next sections, each of them using one of the two types of architecture.

### 5.1. Sybase IQ

SAP IQ (also known as SAP Sybase IQ and Sybase IQ) is a column-based, petabyte scale, relational database software system used for business intelligence, data warehousing, and data marts. Produced by Sybase Inc., now an SAP company, its primary function is to analyze large amounts of data in a low-cost, highly available environment. SAP IQ is often credited with pioneering the commercialization of column-store technology.

At the foundation of SAP IQ lies high-performance column store technology that allows for speed compression and ad-hoc analysis without additional tuning, as well as high scalability and cloud enablement. SAP IQ also provides application services enabling developers to build smarter apps. SAP IQ comes with in-database analytics, multilingual client APIs, federation and web enablement. It includes IQ database drives for web 2.0 programming environments and an extensibility framework for embedding analytics inside the database. The top layer comes with an ecosystem of partner solutions, tools, and apps that are embedded within SAP IQ through the application services layer. This includes Business Intelligence (BI) tools, data integration tools, database admin (DBA) tools, and packaged apps. Everything is encapsulated within a grid architecture.

SAP IQ has an open interface approach towards its ecosystem. Many popular commercial and open source business intelligence and data integration tools are certified to work with SAP IQ. Moreover, now as part of SAP, SAP IQ is also integrated with SAP's Business Intelligence portfolio of products to form an end-to-end business analytics software stack, and is an integral component of SAP's In-Memory Data Fabric Architecture and Data Management Platform.

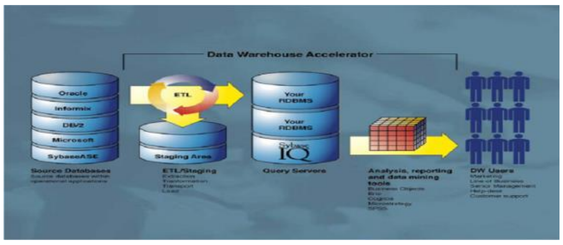

*Figure 2*

## 5.2. Vertica

Vertica is one of a growing band of vendors offering column-based analytic databases for data warehousing. Its Analytic Database is designed specifically for storing and querying large datasets. Vertica's differentiator is that it combines a columnar database engine with massively parallel processing (MPP) and shared-nothing architecture, aggressive data-compression rates, and high availability. Column-based databases can be slow when it comes to deleting and updating data, and Vertica addresses this by taking advantage of a hybrid store that handles write, update, and insert operations. The store also makes the data available for queries in- memory. In addition, the product benefits from smart capabilities for data access and disk I/O. Ovum believes that Vertica's technology could be applicable across a range of markets among companies that have a mix of analytical requirements.

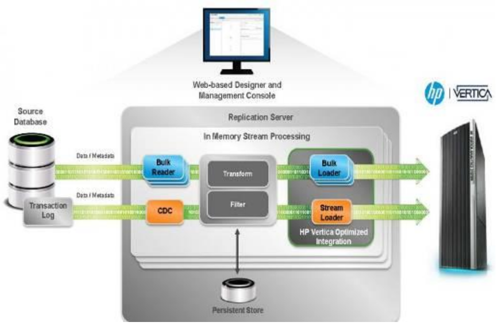

*Figure 3*

## VI. CONCLUSION

Despite their limitations, record-based relational databases have long been the prevailing data structure used in data warehouse systems.

However, as data volumes have increased and analytical needs have become more sophisticated, the sho rtcomings of the RDBMS—most significantly, the design compromise required between optimizing for performance versus query flexibility—have spurred the development of alternative data structures.

Columnar databases offer faster query performance and require less disk storage space than do RDBMSs, but they force their own compromise between optimizing for new record insertion versus record selection and retrieval.

The fundamental technical difference between the Sybase IQ and Vertica columnar DBMSes is the processing-storage model, SMP for Sybase IQ and MPP for Vertica. This difference manifests itself in ways that include: Verticas use of multiply-stored "projections" to speed data access and boost availability versus Sybases reliance on more typical, established storage methods, and also on the ability to add capacity. Sybase IQs independent scaling of compute-power and storage, which will prove advantageous for certain users, versus Verticas grid architecture that links compute and storage scaling. Columns provide better performance at a lower cost with a smaller foot-print: it is diffcult to understand why any company seriously interested in query performance would not consider a column-based solution. Using columns instead of rows means that you get greatly reduced I/O because you only read the columns referenced by the query. This means that you get dramatically improved performance. The use of compression improves I/O rates (and performance) still further.

In addition, de-pending on the supplier, you can eliminate or greatly reduce any need for indexes (thereby reducing on-going administration requirements) and, where they may be usefully used, they can be created automati-cally. In summary, this means that queries run faster (much faster), the database is much smaller that it would otherwise be (with all the upfront and ongoing cost benefts that implies) and there is less administration required than would otherwise be the case (with further ongoing cost benefits).

In addition, you may be able to run queries that simply could not be supported by more conventional means.

## REFERENCE

[1] https://dennisforbes.ca/index.php/2014/04/04/the-real-advantage-of-column-oriented-databases/
[2] http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=3&ved=0CCwQFjAC&url=http%3A%2F%2Fwww.cse.iitb.ac.in%2Fdbms%2FData%2FCourses%2FCS632%2FTalks%2FColumn-vs-Row.ppt&ei=qGrEVOYGwczyBbmjgOAL&usg=AFQjCNH3cd1S5Cp-5LtP6ovo1o0j-dtvIg
[3] http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=6&ved=0CD0QFjAF&url=http%3A%2F%2Fwww.ijcscn.com%2FDocuments%2FVolumes%2Fvol1issue2%2Fijcscn20110

10203.pdf&ei=b2PEVKiKD-K8mAXb1YDoBA&usg=AFQjCNGYac0qy5LQdDZs3qjfYhshPb33xA

[4] http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=5&ved=0CDQQFjAE&url=http%3A%2F%2Fwww.dbjournal.ro%2Farchive%2F2%2F1_Gheorghe_Matei.pdf&ei=CGLEVNGnC8XW8gXSlYDICw&usg=AFQjCNEgWKJ51vLVO_2VvCTk7t6FkyYZkQ

[5] http://en.wikipedia.org/wiki/Sybase_IQ

[6] http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=4&ved=0CC8QFjAD&url=http%3A%2F%2Fvertica.com%2Fwp-content%2Fuploads%2F2011%2F01%2FOvum-Vertica-Analytic-Database1.pdf&ei=kI_EVNbfBcK2mwW_-YGgDg&usg=AFQjCNHmQK4_3cqmE7-AGOIIHk3fyACFoA&bvm=bv.84349003,d.dGY

[7] http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=2&ved=0CCMQFjAB&url=http%3A%2F%2Fwww.sybase.in%2Fsb_content%2F1027874%2FWarehouseAccelerator.ppt&ei=B3_EVPesEcHsmAWEiICwDw&usg=AFQjCNGB2Uf_Sp4gtfpM-ciDQxTo74Ouvw