UDC 004.652(043.2)

Shevchenko I.O.
National Aviation University, Kyiv

COLUMN-ORIENTED DATABASE MANAGEMENT SYSTEMS

For many of users, the database management systems (DBMS) they have been working with or have heard about have been "row-oriented". They may never have applied that label to them, but almost everyone should know how database storing principles affect database performance and what alternatives exist. The current main alternative to "row-oriented" is "column-oriented". "Column-oriented" databases claim to provide a better value proposition for a specialized analytic workload or even the entire data warehouse.

Columnar is a much clearer and simpler term for the term "column-oriented". Columnar database management systems store data by column instead of by row. The defining concept of a column-store is that the values of a table are stored contiguously by column. Thus the relational database which shows its data as two-dimensional tables, of columns and rows, is stored as one-dimensional strings of column cells. For example let's see how simple database of students (Table 1) is stored using two distinct approaches.

Table 1

ID	**Name**	**Surname**
1	John	Smith
2	Mary	Jones

A "row oriented" database serializes all of the values in a row together, then the values in the next row, and so on. The serialized database table is looked as:

1, John, Smith; 2, Mary, Jones;

A column-oriented database serializes all of the values of a column together, then the values of the next column, and so on. The serialized database table is looked as:

1, 2; John, Mary; Smith, Jones;

From this simple concept flow all of the fundamental differences in performance, for better or worse, between "column-oriented" and "row-oriented". For example, a column store will excel at doing aggregations like totals and averages, but inserting a single row can be expensive, while the inverse holds true for row stores. Column data is of uniform type; therefore, there are some opportunities for storage size optimizations available in column-oriented data that are not available in row-oriented data.

Generally the most significant concepts "column-oriented" databases differ from traditional row-oriented databases are performance, storage requirements, ease of modification of the schema and specific use cases of DBMS. Beware that "column-oriented" DBMS were commonly introduced to address a particular need or specific problem (say analyzing footprint, highly compressible distribution of data or spare matrix emulation) rather than to provide a general purpose column-oriented DBMS.