Column-Oriented Database

How it works

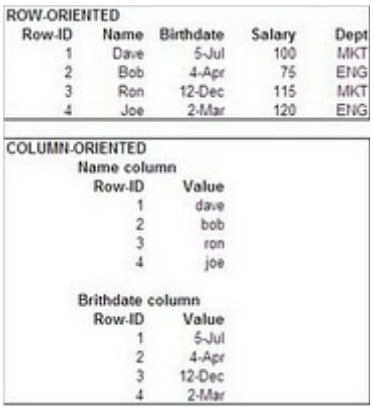
Content is stored by column rather than by row.

Given this table:

Emp ID	Last Name	First Name	Salary
001	A	B	5000
002	C	D	5100
003	E	F	5200

A one-dimensional representation of this by a row-oriented database will be:
Emp ID, Last Name, First Name, Salary; 001, A, B, 5000; 002, C, D, 5100; 003, E, F, 5200;

A one-dimensional representation of this by a column-oriented database will be:
Emp ID, 001, 002, 003; Last Name, A, C, E; First Name, B, D, F; Salary, 5000, 5100, 5200;



ROW-ORIENTED

Row-ID	Name	Birthdate	Salary	Dept
1	Dave	5-Jul	100	MKT
2	Bob	4-Apr	75	ENG
3	Ron	12-Dec	115	MKT
4	Joe	2-Mar	120	ENG

COLUMN-ORIENTED

Name column

Row-ID	Value
1	dave
2	bob
3	ron
4	joe

Brithdate column

Row-ID	Value
1	5-Jul
2	4-Apr
3	12-Dec
4	2-Mar

Similar to row-oriented database, SQL can be used to interact with the system.

Why the need for column-oriented database?

Traditional row-oriented (legacy) databases are good for On-Line Transactional Processing (OLTP), which maps to a single row in the database, e.g. entering employee details.
Most database queries are column-intensive. e.g. SELECT employees WHERE salary per month >$3000.

> In a column-oriented database would only require 2 columns, employee id and employee salary.
> In a row-oriented database it would have to read every single row one at a time to find out if salary per month >$300.

In this instance query on a row-oriented database will be more time-consuming and inefficient as compared to a column-oriented database.

Row-based database query speed can be increased by utilizing indexing, horizontal partitioning, materialized views, summary tables, parallel processing. But these have their drawbacks too. For example indexing will require more storage for the indexes and more maintenance, because when an entry is written the index has to be updated too.

Benefits

No need for indexing:

- -less storage required
- -data load speed is faster because no index needs to be maintained
- -no index design or maintenance required
- -cost savings

Faster query results - no need to access individual rows when only certain columns are needed

Better compression - compression algorithms perform better on data with lower entropy (high data value locality)

Drawbacks

Write operations:

- -inserted tuples have to be broken up to their attributes (columns), and each attribute written separately
- -dense-packed layout makes moving tuples within a page near impossible

Tuple construction:

- -information about a single entity (row) is stored at different places