ColumnStores vs. RowStores: How Different Are They Really?

a LIT review by Megan Fantes
First, what are column-stores and what are row-stores?

Row-stores:

Column-stores:
Is one better than the other?

Answer: it depends

Are you WRITING a lot?

OR

Are you READING a lot?
Is one better than the other?

Answer: it depends

Are you WRITING a lot?

OR

Are you READING a lot?
Now, on to the paper...

1. Advantages of column stores in read-mostly environments
2. Tests the performance of a row-store database that has been modified to act like a column-store
3. Tests each optimization of a column-store to figure out which gives the largest increase in performance
Advantages of Column-Stores

What are some advantages of column stores that you can think of?
Advantages of Column-Stores

What are some advantages of column stores that you can think of?

- Late materialization
- Block iteration
- Compression
- Invisible joins (new in this paper)
What in the Heck are “Invisible Joins”? 
What in the Heck are “Invisible Joins”? 

Phase 1: Apply region = ‘Asia’ on Customer table

Phase 2: Fact Table

Phase 3: Join Results
Modifying a Row-Store Database

3 ways to make a row-store behave like a column-store:

1. Vertical partitioning
2. Index-only plans
3. Materialized views
Results:

Figure 6: (a) Performance numbers for different variants of the row-store by query. Here, T is traditional, T(B) is traditional (bitmap), MV is materialized views, VP is vertical partitioning, and AI is all indexes. (b) Average performance across all queries.
Testing Column-Store Optimizations:

Testing the 4 advantages of column-stores:

- Late materialization
- Block iteration
- Compression
- Invisible joins
Results:

Figure 7: (a) Performance numbers for C-Store by query flight with various optimizations removed. The four letter code indicates the C-Store configuration: T=tuple-at-a-time processing, t=block processing; I=invisible join enabled, i=disabled; C=compression enabled, c=disabled; L=late materialization enabled, l=disabled. (b) Average performance numbers for C-Store across all queries.
Figure 7: (a) Performance numbers for C-Store by query flight with various optimizations removed. The four letter code indicates the C-Store configuration: T=tuple-at-a-time processing, t=block processing; I=invisible join enabled, i=disabled; C=compression enabled, c=disabled; L=late materialization enabled, l=disabled. (b) Average performance numbers for C-Store across all queries.
What Would I Add to This?

A case study

- Awesome, we have these cool column-stores
- But how are we going to use them?
Thanks for listening to my LIT review