DB storage architectures:
Rows, Columns, LSM trees

COS 518: Advanced Computer Systems
Lecture 7
Michael Freedman
Row-based storage: variable lengths

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>URL</td>
<td>Size</td>
<td>Code</td>
<td>Fetched</td>
</tr>
<tr>
<td>BIGINT</td>
<td>VARCHAR(255)</td>
<td>INT</td>
<td>SMALLINT</td>
<td>DATE</td>
</tr>
</tbody>
</table>

How do you walk through all the URLs?
No longer at fixed offsets

Row-based disk layout

- Data stored in fixed-sized pages on disk
  - E.g., typically 8K in PostgreSQL
  - Page includes metadata and actual data items
  - Items = indexes, data rows

Row-based storage: variable lengths

<table>
<thead>
<tr>
<th>ItemIdData:</th>
<th>[(0, 18), (18, 273), (291, 59)]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>Id</td>
<td>URL</td>
</tr>
<tr>
<td>BIGINT</td>
<td>VARCHAR(255)</td>
</tr>
</tbody>
</table>

Row-based disk layout

- Data stored in fixed-sized pages on disk
  - E.g., typically 8K in PostgreSQL
  - Page includes metadata and actual data items
  - Items = indexes, data rows

READ 32 bytes at positions (0 + 8), (50 + 8), (100 + 8), (150 + 8)
Types of database workloads

- **OLTP = OnLine Transaction Processing**
  - Write-heavy
  - Transactions

- **OLAP = OnLine Analytical Processing**
  - Read-heavy
  - Analytical scans or “rollups” along column
    - “SELECT AVG(latency) FROM system WHERE time > now() – interval("1h")

Comparison of disk layouts

- **Row-oriented layout**
  - [Diagram showing row-oriented layout]

- **Column-oriented layout**
  - [Diagram showing column-oriented layout]

Good discussion of benefits of columns...

- **C-Store: A Column-oriented DBMS**
  - Mike Stonebraker, Daniel J. Abadi, Adam Birnbaum, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amos Rosenberg, Sam Madden, Elizabeth O’Neil, Paul O’Neil, Alex Russell, Nga Tran, Stan Zdonik
  - [Link to good discussion]

- **Column-Stores vs. Row-Stores: How Different Are They Really?**
  - Daniel J. Abadi, New Haven, CT, USA
  - Sam Madden, MIT CSAIL, Cambridge, MA, USA
  - [Link to column-stores vs. row-stores discussion]
LSM Trees: Discussion

1. On-disk SSTable indexes are always loaded into memory
2. All writes go directly to the MemTable index
3. Reads check the MemTable first and then the SSTable indexes
4. Periodically, the MemTable is flushed to disk as an SSTable
5. Periodically, on-disk SSTables are “collapsed together”

• LSM Trees: Write to memory, then flush to disk