Hyrise - a Main Memory
Hybrid Storage Engine

Philippe Cudré-Mauroux
eXascale Infolab
U. of Fribourg - Switzerland & MIT

joint work w/
Martin Grund, Jens Krueger, Hasso Plattner, Alexander Zeier (HPI)
and Sam Madden (MIT)

October 6, 2010
SAP Labs Palo alto
Outline

- Motivation (OLTP VS OLAP)
- Hyrise Architecture
- Hybrid Layouts
- Query Execution
- Cost Model
- Physical Layouters
  - Optimal Layouter
  - Scalable Layouter
- Performance
- Current & Future Work
Motivation: the DBMS Divide

- **OLTP**
  - Data entry
  - Mix of reads and writes to few rows at a time
  - Index structures (B+ Trees)

  ➡ Transaction Processing System

- **OLAP**
  - Aggregates
  - Bulk updates (ETL)
  - Large sequential scans spanning few columns but many rows
  - Warehousing
  - Rise of column-stores

 VS
Problems w/ current divide

- Increasing needs for “real-time analytics” and hybrid workloads
  - e.g., ATP (Available To Promise) applications involve both OLTP queries and OLAP aggregates
  - fundamental bottleneck implied by implicit separation

- Duplication of data management systems is cumbersome and costly for SMEs
  - ... and even for large companies
Hyrise

- Hyrise: a
  - Main memory
    - Simplicity
    - Efficiency
    - Future of DBMS market for enterprise data

- Hybrid storage engine
  - Partitions tables into vertical partitions of varying widths depending on how columns accessed (e.g., transactionally or analytically)
Hyrise Architecture
Query Processing

- Query plans
  - Trees of operators

- Standard operators
  - Look-ups, aggregations, joins, etc.

- Support both early and late-materialization strategies
Examples of query plans

Join Plan

Position Lookup
Position AND
Positions
Values
Hash Join
Filter
Filter
Dimension Tables
Fact Table

Non-Join Plan

Position Lookup
Position AND
Positions
Values

Filter
Filter
Filter

1,3,4,5
1,2,3,4
3,4,5
3,4,5
1,2,3,4
Obviously, performance heavily depends on the way attributes are partitioned.
Physical Database Design (2)

- Optimal physical layout $\lambda_{opt}$

\[
\lambda_{opt} = \arg\min_{\lambda} (\text{Cost}_{DB}(W))
\]

dependent on

- Set of attributes and #tuples ($DB$)
- Important / frequent queries ($W$)
- Cost of various queries given a layout ($\text{Cost}$)

- Two issues
  - Accurate cost-model
  - Optimal and efficient layout-selection algorithm
Cost-Model (1)

- What to model?
  - layout-dependent VS layout-independent costs

- Where does time go in main-memory layout-dependent operations?
  - cache misses (L1/L2)

⇒ Highly accurate model for cache misses in hybrid DBMSs
Cost Model Basics

- Partial projections:

\[ \text{Miss}_i(\pi, C) = \frac{C.n}{v} \sum_{r=0}^{v} \text{Miss}_i(r, \pi) \]

\[ \text{Miss}_i(r, \pi) = \left[ \frac{\text{lineoffset}_i(r, \pi) + \pi.w}{L_i.w} \right] \]

\[ v = L_i.w / \text{gcd}(C.w, L_i.w) \]
Cost Model (2)

- Additional expressions for
  - Arbitrary combination of partial projections
  - Selections
  - Joins, aggregates
  - Container padding, cache collision, prefetching decisions etc.
We can discriminate layouts thanks to the cost-model.

How many layouts shall we consider for n attributes?

\[ a(n) = (2n-1)a(n-1) - (n-1)(n-2)a(n-2) \]

with \( a(0) = a(1) = 1 \)

3,535,017,524,403 for a table of 15 attributes

➡ Efficient optimal algorithm
➡ Scalable approximate algorithm
Optimal Layouter

- Three phases
  - Candidate Generation
    - Primary partitions (no container overhead cost)
  - Candidate Merging
    - Combine candidates
    - Discard candidates that are subsumed by a set of more efficient candidates
      e.g., Cost( |A1|,|A2| ) < Cost ( |A1,A2| )
  - Layout Generation
    - Generate valid layouts
    - Exponential in the worst-case
    - Relevant for small workloads
Scalable Layouter (1)

- Divide-and-conquer approximate algorithm
  - Graph storing affinities between primary partitions
    - min-cut graph-partitioning to generate sub-graphs containing at most $K$ primary partitions
      - approximate multilevel k-way partitioner
Scalable Layouter (2)

- Generate optimal layouts for each subset
  - worst-case exponential in $K$

- Combine sub-layouts to generate final layout
  - Requires $O(|P| \times \binom{|P|}{2})$ layout evaluation in the worst case ($|P|$ = total # partitions from previous step)

- Approximate result
  - Very tight upper-bound on penalty incurred

- Scales to hundred of primary partitions and thousands of queries
  - e.g., a few minutes for 1000 queries and 500 att.
Hyrise Performance (1)

- Detailed performance evaluation on a realistic hybrid workload (Krueger 2010)
Workload (OLTP-Style)

Q1: Search for a customer by first or last name (ADRC)
    select ADDRNUMBER, NAME_CO, NAME1, NAME2, KUNNR
    from ADRC where NAME1 like (...)
    OR NAME2 like (...);

Q2: Read the details for this customer (KNA1)
    select * from KNA1 where KUNNR = (...);

Q3: Read all addresses belonging to this customer (ADRC)
    select * from ADRC where KUNNR = (...);

Q4: Search for a material by its text in the material text table (MAKT)
    select MATNR, MAKTX from MAKT where
    MAKTX like (...);

Q5: Read all details for the selected material from the material table (MARA)
    select * from MARA where MATNR = (...);

Q6.a: Insert a new row into the sales order header table (VBAK)
    insert into VBAK (...) values (...);

Q6.b: Insert a new row into the sales order line item table based on the results of query Q5 (VBAP)
    insert into VBAP (...) values (...);

Q7: Display the created sales order header (VBAK)
    select * from VBAK where VBELN = (...);

Q8: Display the created sales order line items (VBAP)
    select * from VBAP where VBELN = (...);

Q9: Show the last 30 created sales order headers (VBAK)
    select * from VBAK order by VBELN desc limit 30;
Workload ("OLAP-Style")

Q10 Show the turnover for customer KUNNR during the last 30 days
   select sum(item.NETWR), header.KUNNR from
   VBAK as header, VBAP as item where
   header.VBELN = item.VBELN and
   header.KUNNR = $1 and header.AEDAT >= $2;

Q11 Show the number of sold units of material MATNR for the next 10
days on a per day basis
   select AEDAT, sum(KWMENG) from VBAP where
   MATNR = $1 and AEDAT = (...) group by AEDAT;

Q12 Show the number of sold units of material MATNR for the next 180
days on a per day basis
   select AEDAT, sum(KWMENG) from VBAP where
   MATNR = $1 and AEDAT = (...) group by AEDAT;

Q13 Drill down through the material hierarchy starting on the highest level
   using an internal hierarchy on the table, each drill-down step reduces
   the selectivity, starting from 40% selectivity going down to 2.5% selec-
tivity.
## Results (1)

<table>
<thead>
<tr>
<th></th>
<th>Q1</th>
<th>Q3</th>
<th>Q12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid</td>
<td>9707</td>
<td>2585</td>
<td>101615</td>
</tr>
<tr>
<td>Column</td>
<td>26488</td>
<td>2356</td>
<td>101957</td>
</tr>
<tr>
<td>Row</td>
<td>14995</td>
<td>2764</td>
<td>840862</td>
</tr>
</tbody>
</table>

![Bar chart showing hybrid, column, and row distribution across percentages for Q1, Q3, and Q12]
Results (2)

- Hyrise combines the best of both-worlds (OLTP & OLAP systems) in one system.

- Final speed-up depends on query weighting.
  - 40% to 400% speed-up compared to all-column / all-row layouts on a hybrid workload.
Current Status

- Fully functional prototype
- Cost-model framework
- Automated layouter

- Article in PVLDB 2010
  - presentation at VLDB 2011

- Current work on
  - Multi-core optimizations
  - Hybrid horizontal + vertical partitionings
Future Work

- Data replication & partial materialization
  - Maximize utility function given a space budget

- Hybrid storage for new application domains
  - multidimensional / scientific / multimedia data
  - graph data (Linked Data?)
Hyrise - a Main Memory Hybrid Storage Engine

Philippe Cudré-Mauroux
eXascale Infolab
U. of Fribourg - Switzerland & MIT

pcm@unifr.ch

October 6, 2010
SAP Labs Palo alto