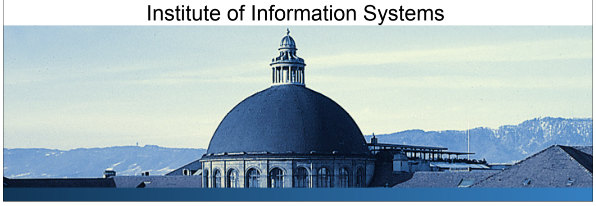
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Data Warehousing
SoSe 2006

Dr. Jens-Peter Dittrich
jens.dittrich@inf
www.inf.ethz.ch/~jensdi
Institute of Information Systems



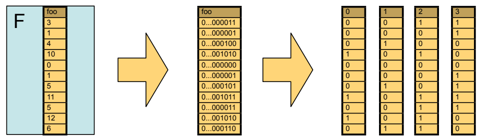
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Query Processing and Indexing (Part 2)

Projection Index Details

- assume \`foo´ is a column of F
- the projection index then consists of the sequence of column values from \`foo´ ordered by RID (holes may exist for unused row numbers)
- if column \`foo´ is 4 bytes in length, then we can fit 1000 values from \`foo´ on a single 4 KByte disk page
- for any given row numer n = m(r) in F we can access the proper disk page p and slot s to retrieve the appropriate \`foo´ value with a simple calculation:
  p = n / 1000
  s = n % 1000
- the row number of a given page p and slot s is calculated as:
  n = 1000 * p + s

Bit-sliced Index Example

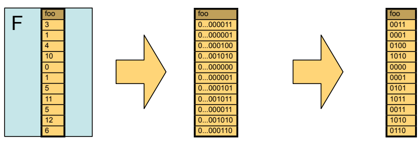


- for this example only 1/8 of the original space for a PI is required
- for FTS on vertical partition this translates roughly into **times 8** performance improvements!
- Assuming 100 million rows in projection index this is 50 MB versus 400 MB to read (versus possibly dozens of gigabytes for reading F)

Bit-sliced Index

- Assumptions
  - \`foo´ is a column of F of a countable number type
  - domain of numbers is limited, all numbers in 1,...,N
- To represent 1,...,N numbers we need $n = \lceil \log_2 N \rceil$ bits
- Idea
  - for column \`foo´ define a bit list D(RID,i) for each bit of each value
  - D(RID,0) = true ⇔ 2^0 bit of value \`foo´ of row RID is set
  - D(RID,1) = true ⇔ 2^1 bit of value \`foo´ of row RID is set
  - ...
  - D(RID,i) = true ⇔ 2^i bit of value \`foo´ of row RID is set
  - ...
  - D(RID,n) = true ⇔ 2^n bit of value \`foo´ of row RID is set

Compressed Vertical Partitions



- Alternative: instead of cutting a vertical partition into bit-slices compress it directly into a single compressed vertical partition
- same effect as bit-sliced partitions
- many different encoding schemes exist for compressing (see Managing Gigabytes book)

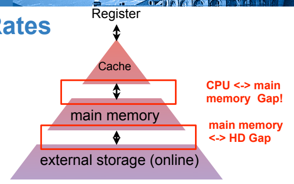
Pagesize

- OLTP works best with small page sizes
  - space- and update-efficient system
  - may require lots of seek operations
  - high data fragmentation
- OLAP works best with large page sizes
  - query-efficient system
  - OLAP queries may require large amounts of data to be read to answer a query
  - avoid seek operations whenever possible
  - low data fragmentation

Cache Hit Rates

K-M Bytes
2-20 ns

M-G Bytes
50-100 ns

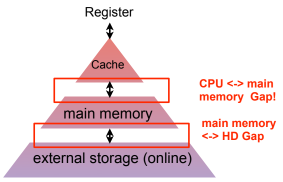


- Lesson learned so far: Vertical partitions + compression lowers I/O-cost on the hard disk(s) considerably
- Now: vertical partitions also lowers **I/O-cost in main memory!**
- I/O-cost in main memory?
  - data not in cache, cache miss: read it from main memory
  - access to main memory is much slower than accessing cache
  - CPU has to wait until data is loaded into the cache
  - also: many CPUs are able to prefetch (either automatically guessing access patterns or through pragma directives)

Cache Hit Rates: Goals

K-M Bytes
2-20 ns

M-G Bytes
50-100 ns



- Ideal world: only \`relevant´ data should go into the cache
- \`relevant´ data: data that has to be touched by the CPU
- everything else just waists cache space or stalls CPU
- Algorithms should be designed to **not** waist cache space
- Performance gains may be tremendous (factors)
- Products: Sybase IQ, Applix TM1, SAP BI Accelerator, ...

Cache Hit Rates: Approaches

- 2 approaches to optimize cache hit rates
  - cache-oblivious algorithms
    - no detailed knowledge (obliviousness) about memory hierachy and size of memory levels
    - generic approach
    - not tailored for a specific architecture (see lecture by Prof. Widmayer)
  - cache-aware algorithms
    - detailed knowledge (awarenbess) about memory hierachy and size of memory levels
    - specialized approach
    - not tailored for a specific architecture (see hardware lecture by Prof. Thiele)

MOLAP: Multidimensional OLAP

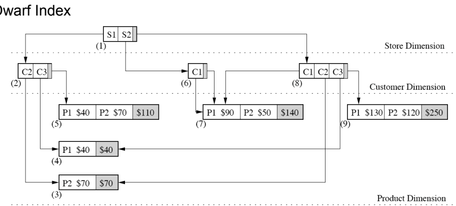
- Non-relational approach to OLAP
- Core Idea
  - Do not use DBMS for query processing
  - invent clever index structures that
    - materialize (almost) all aggregates
    - store aggregates in a highly compressed manner
  - at query time just look-up aggregates values or do light-weight post-aggregation

MOLAP Example: Dwarf

Fact table

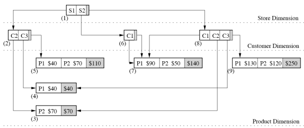
Store	Customer	Product	Price
S1	C2	P2	$70
S1	C3	P1	$40
S2	C1	P1	$90
S2	C1	P2	$50

Dwarf Index



- Literature: Yannis Sismanis, Antonios Deligiannakis, Nick Roussopoulos, Yannis Kotidis: Dwarf: shrinking the PetaCube. SIGMOD Conference 2002: 464-475

Dwarf: Worst Case Analysis



- Assume each dimension has cardinality N. Then each level has a fan-out of N+1.
- The number of nodes is then given by $(N+1)^{D-2}+1$ ($D \geq 3$).
- The number of leaves is given by $(N+1)^{D-1}$ ($D \geq 1$), i.e., number of leaves dominates
- $(N+1)^{D-1}$ may be huge for large cardinalities and high dimensions
- Example: N=1000, D=6, number of leaves= 10^{15}.
- Therefore, this approach is not **guaranteed** to scale well.
- (However approach may work for certain scenarios)

MOLAP: Discussion

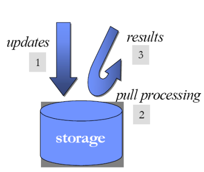
- Fight in industry MOLAP versus ROLAP
- Some companies offer MOLAP based tools: Essbase, MS Analysis Services
- companies do not tell people how they actually do it...
- reports on problems with many dimensions, high cardinalities and unexpectedly long indexing times
- therefore some companies favor HOLAP: Hybrid OLAP, i.e., part of the data is in ROLAP another part in MOLAP
- Recommendation
  - do not use it
  - given the current hardware charecteristics it is questionable whether MOLAP has a business case if the techniques taught in this lecture are well applied

Big picture: One size does not fit

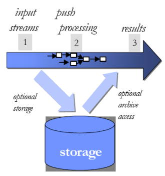
- Last 25 years of DBMS can be summed up in a single phrase: "One size fits all"
- DBMS vendors started with an OLTP centric relational architecture and extended that over the years to be able to cope with other applications.
- The idea was **one** architecture for **every** data processing application.
- This idea does not work. Vendors use separate engines.
- Examples include
  - data warehousing
  - stream processing
  - scientific databases
  - text search
  - personal information management
- Literature: Michael Stonebraker, Ugur Çetintemel: "One Size Fits All": An Idea Whose Time Has Come and Gone (Abstract). ICDE 2005: 2-11

Big picture: One size does not fit

- Example: stream processing



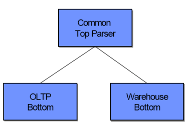
Outbound processing



Inbound processing

Note: Stream processing will be treated in a separate class (June 23rd 2006)

Big picture: One size does not fit



- Current approach of vendors: hide multiple engines behind a common frontend
- Cons
  - illusion of a single system is a marketing fiction
  - vendor is in fact selling multiple systems
  - pressure from both markets to include market specific features
  - common front-end strategy impractical, e.g., for stream processing systems

Big picture: One size does not fit

- Summary
  - hard to find one code line that works well for all applications
  - current OLTP engines bloated with add-on functionality that does not always perform well
  - in future they may be domain specific database engines (it is already happening)

Summary

- Several methods to speed-up OLAP queries were presented
  - star join plans
  - bitmaps
  - join-indexes
  - mat views
  - partitioning
  - vertical and horizontal partitioning
  - compression
  - cache awareness and oliviousness
  - page size
  - data layout
  - multidimensional databases
  - „one size fits all" approach etc.
- Applying all these techniques well gives excellent OLAP query performance even on TB data warehouses.
- Not all vendors provide all features.

Literature

- Don S. Batory: On Searching Transposed Files. ACM Trans. Database Syst. 1979.
- George P. Copeland, Setrag Khoshafian: A Decomposition Storage Model. SIGMOD Conference 1985: 268-279
- Patrick E. O'Neil, Goetz Graefe: Multi-Table Joins Through Bitmapped Join Indices. SIGMOD Record 24(3): 8-11 (1995)
- Patrick E. O'Neil, Dallan Quass: Improved Query Performance with Variant Indexes. SIGMOD Conference 1997: 38-4
- Chee Yong Chan, Yannis E. Ioannidis: Bitmap Index Design and Evaluation. SIGMOD Conference 1998: 355-366
- Clark D. French: ``One Size Fits All'' Database Architectures Do Not Work for DDS. SIGMOD Conference 1995: 449-450
- Clark D. French: Teaching an OLTP Database Kernel Advanced Data Warehousing Techniques. ICDE 1997: 194-198
- Michael Stonebraker, Ugur Çetintemel: "One Size Fits All": An Idea Whose Time Has Come and Gone (Abstract). ICDE 2005: 2-11
- Ian H. Witten, Alistair Moffat, Timothy C. Bell: Managing Gigabytes: Compressing and Indexing Documents and Images, Second Edition. 1999, Morgan Kaufmann
- Yannis Sismanis, Antonios Deligiannakis, Nick Roussopoulos, Yannis Kotidis: Dwarf: shrinking the PetaCube. SIGMOD Conference 2002: 464-475