Advanced Data Management Technologies

Unit 21 — Main Memory Databases

J. Gamper

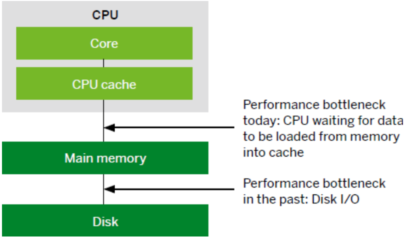
Free University of Bozen-Bolzano
Faculty of Computer Science
IDSE

Outline

Outline

Technological Transition

- Computer architecture has changed a lot in the past decades.
- Today's multicore, multi-CPU server provide fast communication between processor cores via main memory or shared cache.
- Main memory is no longer a limited resource.
  - In 2012 servers with more than 2 terabytes of RAM are available.
- Server processors with 100 cores and more are able to process more and more data per time unit.
- With all data in memory, disk access is no longer a limiting factor for performance.
- New bottleneck is CPU waiting for data from memory!



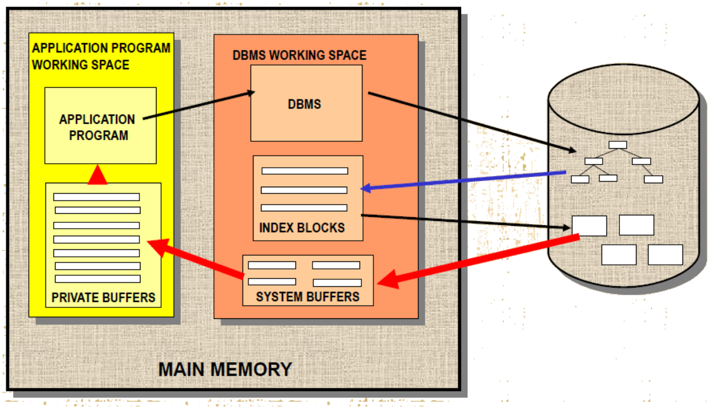
- Modern computer architectures create new possibilities and challenges for data management and processing → main memory databases.

Definition

- **Disk resident database (DRDB)**
  - The primary copy of data is permanently disk resident.
  - Data can be temporarily cached in main memory for access speed-up.
- **Main memory database (MMDB)**
  - The primary copy of data lives permanently in main memory.
  - There can be a backup copy resident on disk.
- Advantages of MMDBs
  - MMDBs avoid the disk IO bottleneck of DRDBs
  - No buffer cache management
  - High throughput
  - High availability

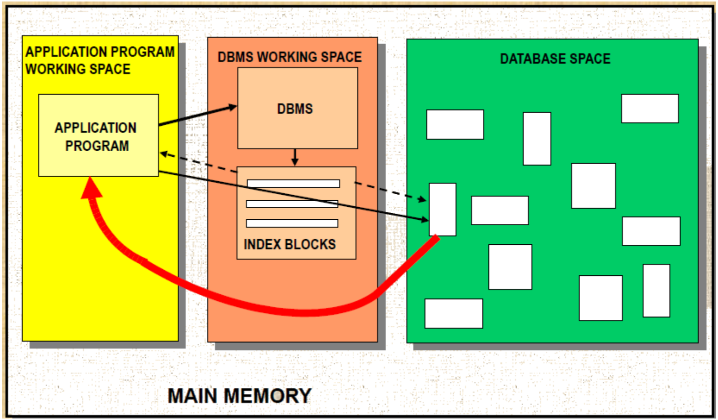
DRDB

- Data are accessed via a buffer manager, which (given the disk address) checks if the relevant block is in MM cache and then copies it to the MM application working area.



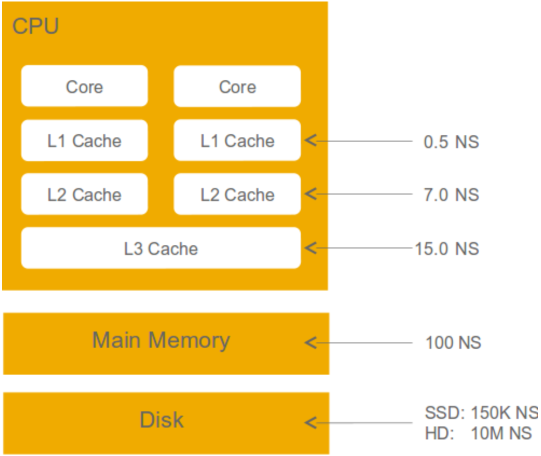
MMDB

- Data are accessed by directly referring to their memory address.



Memory Hierarchy

- DRAM is 100,000 times faster than disk, but DRAM access is still 6-200 times slower than on-chip caches.



Main Memory vs. Disk Storage

- Access time
  - Access time of MM orders of magnitude faster than for disks (100 nsec vs. 10 msec)
- Access pattern
  - Memory is better for random access than disks.
  - Disks have high fixed cost per access, independent of the amount of retrieved data (block-oriented access)
  - MM does not care of sequential access (?).
- Stableness
  - Memory is volatile; content lost if system crashes.
  - If a single memory board fails the entire machine must be powered down loosing all the data.
  - Even if special HW can enhance MM reliability, periodic backup is necessary.
  - Disk is nonvolatile (permanent).
- Security
  - Memory is more vulnerable to software errors, since memory can be directly accessed by the processor/applications.

Hybrid MM-DR Database Systems

- Some DB are so large that they will never fit in MM
- Data can belong to different classes
  - Hot: frequently accessed, low volume, timing sensitive (e.g., bank account records)
  - Cold: rarely accessed, voluminous, non time critical (e.g., bank customers records, historical records)
- Hybrid MM-DR DBMSs consist of a collection of databases, some MM others DR
- Objects can migrate among the dbms, changing their structure accordingly (e.g., IBM IMS Fast Path)

MMDBMs Concurrency Control/1

- Lock duration is short
  - Reduced contention
  - Large granules (up to the entire database)
- This almost eliminates the need of concurrency control

  $\longrightarrow$ mainly serial transaction processing
- Concurrency control still necessary when
  - mixed length transactions coexist
  - a multiprocessor system shares the DB among the different units

MMDBMs Concurrency Control/2

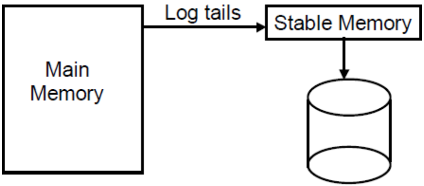
- Traditional implementation
  - Lock (hash) tables holding entries for currently locked objects
  - No lock information attached to data
- Implementation in MMDBs
  - Add some bits of locking information to the data, e.g.,
    - 1st bit is the X-LOCK SET bit
    - 2nd bit is the WAITING FOR bit

MMDBMs Commit Processing/1

- ACID properties of transactions
- Durability of transaction forces a log record to be written to stable storage before committing
- Logging affects response time and throughput
- Problem: Log I/O becomes a bottleneck!

MMDBMs Commit Processing/2

- Solution 1: Store log tail in stable memory
  - Reduces response time



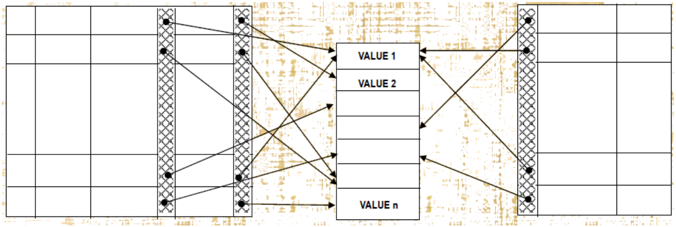
- Solution 2: Group commit
  - Accumulate log until page is full
  - Flush log page to disk only once
  - Reduces the total number of disk accesses
- Solution 3: Precommit transactions
  - Release lock (i.e., precommit) when log is written to log buffer
  - Commit when log buffer flushed to disk
  - Reduces blocking time of other transactions

Data Representation/1

- Relational data are traditionally stored in flat files
  - Slotted page structure
  - Tuples are store sequentially
  - Attribute values are "embedded" in the tuples
    - Space consuming due to duplicate values.
  - Access is local
- Indexes for efficient access

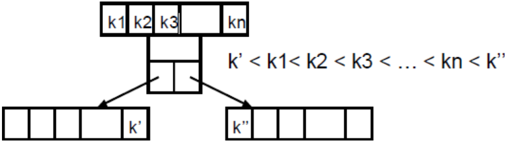
Data Representation/2

- Access locality is not an issue in MMDBs
  - Any location can be accessed at the same speed
- Variable length fields are not problematic
  - Pointers to heap space
- Compressing data size is a major goal of MMDBs → domain storage
  - Store domain values of enumerated types in a domain table
  - In the tuples, store pointers to the domain table
  - Domain tables can be shared among columns and relations
  - Yields fixed size tuples



T-Tree Index

- **T-tree** is the most important index structure in MMDBs
  - Modified binary AVL tree
    - binary search
  - A node contains more than two values
    - Storage and update efficiency (as in B-trees)
  - Balanced by rotating nodes



- Advantages
  - Space efficient
  - Logarithmic performance

Outline

SAP Vision and Challenge

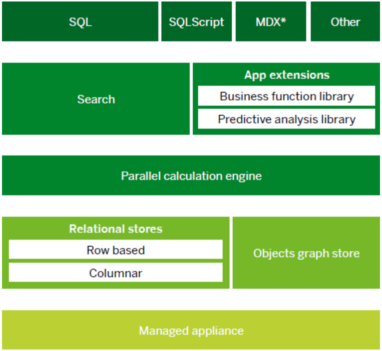
- Unify transaction processing and analytics
- Single system
- Same data instance
- Run analytics in real-time
- Run analytics and transactions at the "speed of thought"

- Solution: in-memory computing
  - Store large blocks of data directly in random access memory (RAM)
  - Keep it there for continued analysis

→ SAP HANA

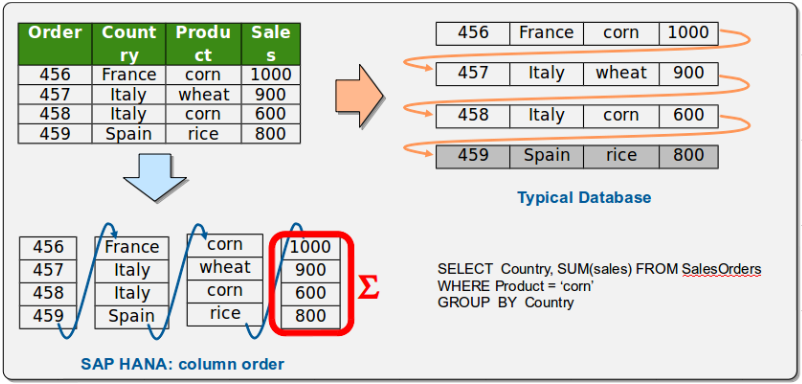
SAP HANA Database

- The SAP HANA platform implements a new approach to big data analytics
- At the core is a full database management system with a standard SQL interface, transactional isolation and recovery (ACID), and high availability
- But includes much more than the DBMS
- In-memory is much more than simple caching of disk data structures in main memory
  - Data is completely stored in main memory
  - Highly tuned access structures
  - Row-based and column-based stores
  - Data compression techniques
  - Parallelization of query processing
  - etc.



HANA Column Store

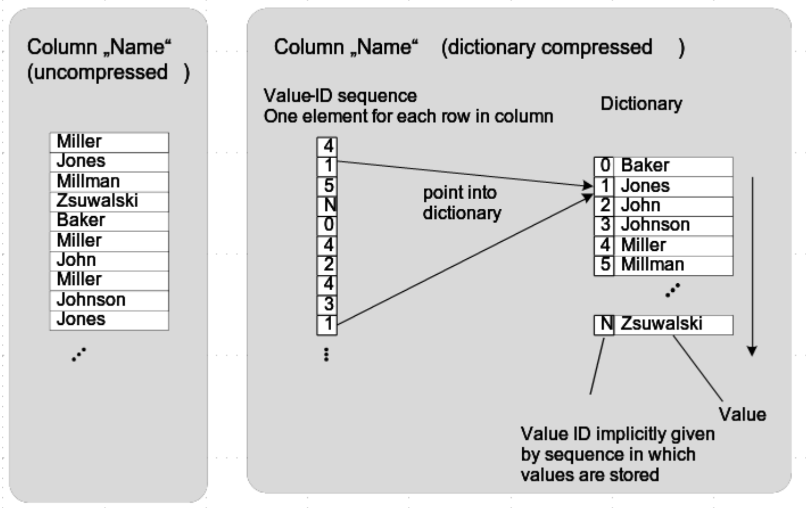
- In analytics, frequently only a small subset of columns is needed
- Extreme fast scan of columns
- Fast on-the-fly aggregation over columns



HANA Data Compression

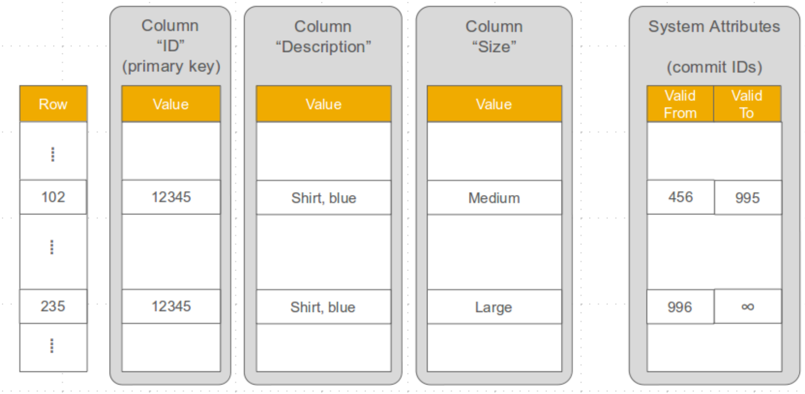
- Efficient compression methods (dictionary, run length, cluster, prefix, etc.)
- Compression works well with columns and can speedup operations on columns ($\approx$ factor 10)
- Because of compression (slow!), write changes into less compressed delta storage
  - High write performance not affected by compression
    - Data is written to delta storage with less compression which is optimized for write access
  - Merged into columns from time to time or when a certain size is exceeded
    - Delta merge can be done in background
  - Trade-off between compression ratio and delta merge runtime

HANA Dictionary Compression

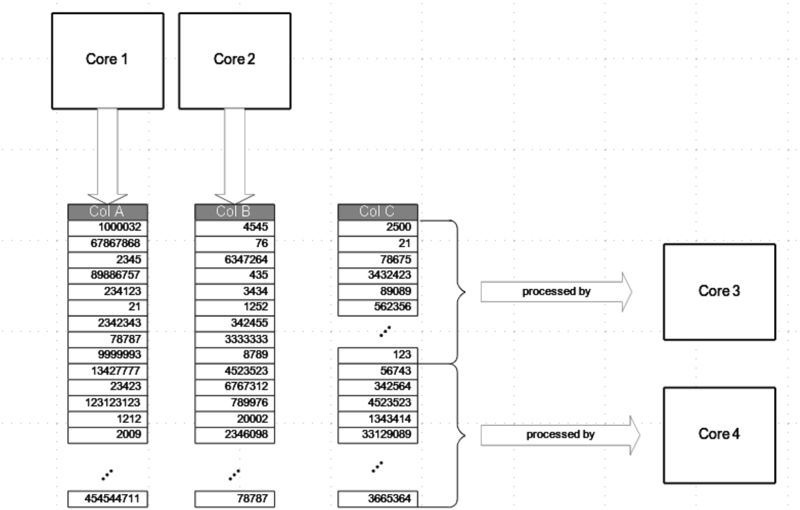


HANA Temporal Tables

- All updates and deletes are handled as inserts
- e.g., update T1 set Size = 'Large' where ID = '12345'

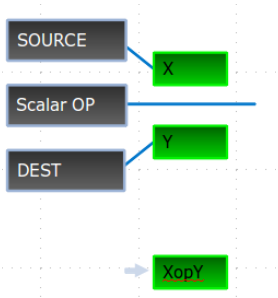


HANA Multi-core Parallelization

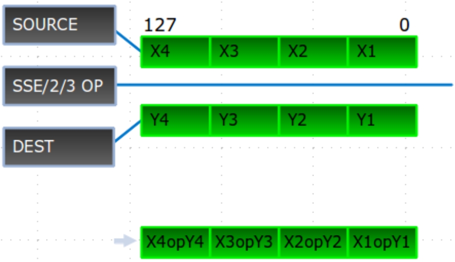


HANA Single Instruction Multiple Data (SIMD)

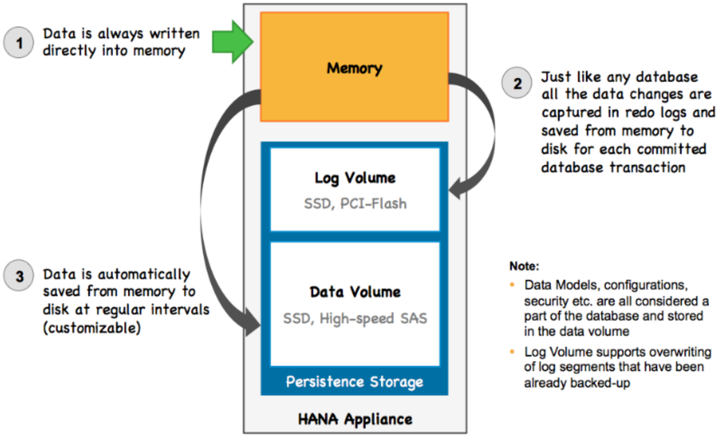
- Scalar processing
  - traditional mode
  - one instruction produces one result



- SIMD processing
  - with Intel SSE(2,3,4)
  - one instruction produces multiple results

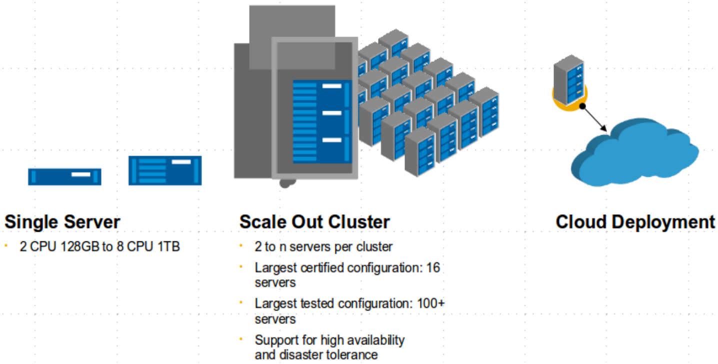


HANA Persistence Layer



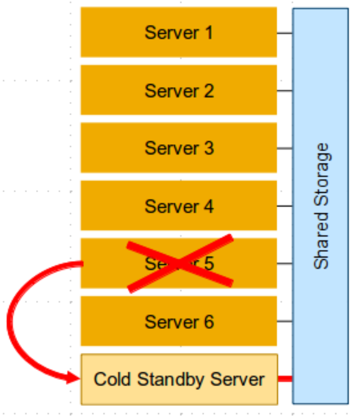
HANA Scalability

- Scales from very small servers to very large clusters

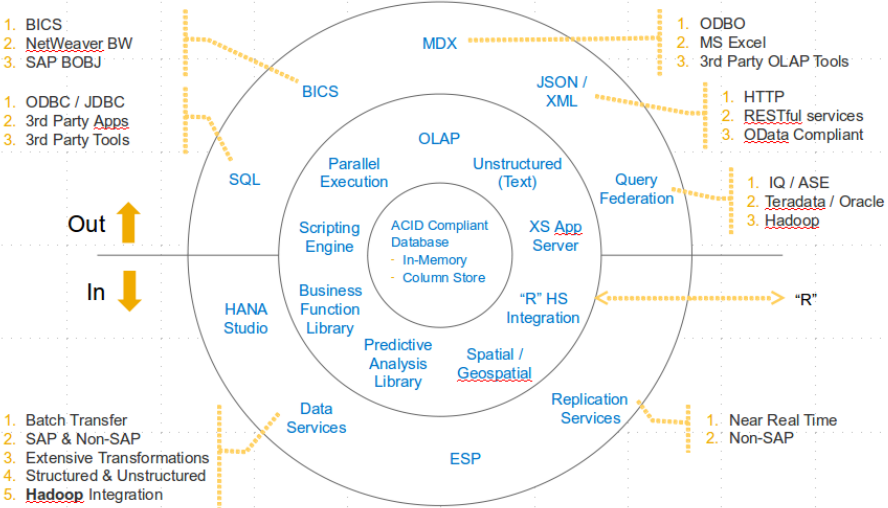


HANA High Availability

- High availability configuration
  - N active servers in one cluster
  - M standby server(s) in one cluster
  - Shared file system for all servers
- Services
  - Name and index server on all nodes
  - Statistics server (only on active servers)
- Failover
  - Server X fails
  - Server $N+1$ reads indexes from shared storage and connects to logical connection of server X



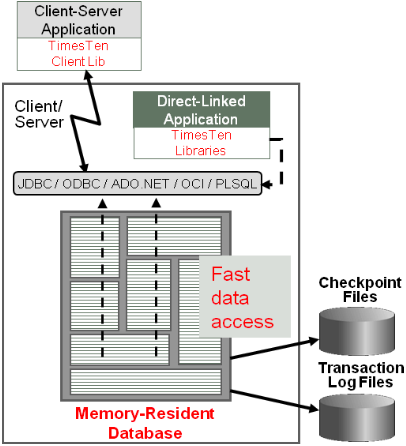
HANA Inside



1. BICS
2. NetWeaver BW
3. SAP BOBJ
1. ODBC / JDBC
2. 3rd Party Apps
3. 3rd Party Tools
Out
In
1. Batch Transfer
2. SAP & Non-SAP
3. Extensive Transformations
4. Structured & Unstructured
5. Hadoop Integration
MDX
BICS
SQL
HANA Studio
Data Services
ESP
Replication Services
Query Federation
JSON / XML
OLAP
Parallel Execution
Unstructured (Text)
Scripting Engine
XS App Server
Business Function Library
"R" HS Integration
Predictive Analysis Library
Spatial / Geospatial
ACID Compliant Database
- In-Memory
- Column Store
1. ODBO
2. MS Excel
3. 3rd Party OLAP Tools
1. HTTP
2. RESTful services
3. OData Compliant
1. IQ / ASE
2. Teradata / Oracle
3. Hadoop
"R"
1. Near Real Time
2. Non-SAP

Oracle TimesTen In-Memory Database

- In-memory RDBMS
  - Entire database in memory
  - Interfaces: Standard SQL with JDBC, ODBC, OCI, Pro*C, .NET, PL/SQL
  - Compatible with Oracle Database
- Persistent and durable
  - Transactions with ACID properties
- Extreme performance
  - Instantaneous response time
  - Very high throughput
- Embeddable



Summary

- New technological changes brought that main memory is no longer a limited resource → new opportunities for data processing
- Main memory databases keep the primary copy of data permanently in main memory
  - Backup copy on resident disk
- Data is accessed directly in memory and not via buffer manager
- Main memory is much faster than disk, and data locality is no longer an issue (any location can be accessed at the same time)
  - High fixed cost of disks due to block access is avoided
- Main memory is more vulnerable to software errors and volatile
- Concurrency is still there, but less important and crucial
- Optimized data representation
  - Use of pointers instead of repeating values or foreign keys
  - Advanced data compression techniques are applied
- T-tree is main index structure
- SAP HANA and Oracle TimesTen are two commercial main memory databases