**MAIN MEMORY DATABASE SYSTEMS: AN OVERVIEW**

**Summary:** As the title suggests this paper gives an overview of the Main Memory Database systems (MMDB’s). The paper discusses the differences between the DRDB’s and MMDB’s based on: *Access methods, application programming interface, commit processing, concurrency control, data clustering, data representation, main memory database system (MMDB), query processing, recovery.*

**Introduction:** Main memory DB and disk DB both have copies of the data however, both database systems differs because the main memory and disk differs in the following aspects: a) The access time of main memory is much less. b) Main memory is volatile. c) Disk access is uniform and independent of work; main memory access is not. d) Disk data layout is crucial to performance, which doesn't hold for MMDB. e) Main memory is directly accessible by the processor, makes data vulnerable.

**Literature Review:** The paper discusses about certain queries regarding MMDBs. Although it is reasonable to assume that an entire database can fit into a main memory however, there are real life applications that have *hot and cold* data and therefore just a part of the database needs to be in memory. Author predicts that there will be time when DRDB will be completely turned into MMDBs. Moreover, with advent of special hardware it will be possible to turn memory into nonvolatile resulting in decrease of media failures and better efficiency. Additionally, less backups will be required. However, for now it is important to keep a backup on the disk to endure potential failures. The factors for frequent backups: a) Memory is vulnerable to OS errors. b) Media failure means loss of the entire database, and recovery is slow. c) Sources like battery lead to higher data loss probability.

**Implementation:** The paper discusses the impacts of memory resident data on the lines of *concurrency control* (large lock granules can be used), *commit processing* (logs are needed), *access methods* (index structures can store pointers to indexed data), *data representation* (tuples represented as sets of pointers to values), *query processing* (construct compact data structures to speed up queries), *recovery* (do checkpoint backup and failure recovery), *performance* (performance of checkpoint and backup is critical), *application interface* (memory position pointer provided to user and allows better performance, but have authorization issues and system can't log all changes ), and *data clustering* (since tuples store pointers to values, migrating data to disk is tricky ). Some DBMS for memory resident data (OBE, MM-DBMS, IMS/VS, MARS, HALO, TPK) are introduced and compared. They differ in the aspects mentioned in the above section.

**Results:** With the increase inavailability of cheaper memory storage, it becomes cost effective to keep more data permanently in memory. Therefore, memory resident database systems will become more common in the future.

**Discussion:** The paper clearly defines the advantages of MMDBs over the traditional DRDBs however, the paper lacks experimentation to illustrate how MMDB performs better than DRDB and in what situation MMDB performs better.

**Conclusion:** The main aim of this paper is to give a thorough overview on the main properties, existing design and implementations of memory database systems. The weak point of this paper falls on that it only introduces the current models that exist but fails to do in depth comparison.