

**Experiment No. 3**

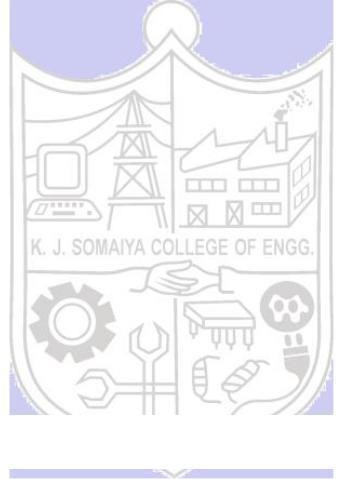
**Title: Execution of In-memory database queries**

**Batch: A1 Roll No.:16010422013 Experiment No.:3 Aim: To execute In-memory database queries**

# Resources needed: MySQL

**Theory**

In-Memory [database](http://www.webopedia.com/TERM/D/database.html) is a database that uses a system's main [memory](http://www.webopedia.com/TERM/M/memory.html) for data storage rather than the disk-based [storage](http://www.webopedia.com/TERM/M/mass_storage.html) typically utilized by traditional databases. In-memory databases, or IMDBs, are frequently employed in high-volume environments where response time is critical, as access times and database requests are typically considerably faster when system memory is used as opposed to hard disk storage.

The traditional databases and in-memory databases can be used together and referred as [hybrid databases,](http://www.webopedia.com/TERM/H/hybrid_database.html) which support both in-memory and disk-based storage in order to maximize performance as well as reliability of the system. All most all RDBMS systems available in market supports In-Memory databases.

# MySQL In-Memory database:

In MySQL DB, the MEMORY storage engine creates special-purpose tables with contents that are stored in memory. Because the data is vulnerable to crashes, hardware issues, or power outages, use of these tables are limited to temporary work areas or read-only caches for data pulled from other tables.

A typical use case for the MEMORY engine involves these characteristics:

* Operations involving transient, non-critical data such as session management or caching. When the MySQL server halts or restarts, the data in MEMORY tables is lost.
* In-memory storage for fast access and low latency. Data volume can fit entirely in memory without causing the operating system to swap out virtual memory pages.
* A read-only or read-mostly data access pattern (limited updates).
* MEMORY tables cannot contain [BLOB](https://dev.mysql.com/doc/refman/5.5/en/blob.html) or [TEXT](https://dev.mysql.com/doc/refman/5.5/en/blob.html) columns.

To create a MEMORY table, specify the clause ENGINE=MEMORY on the [CREATE](https://dev.mysql.com/doc/refman/5.5/en/create-table.html) [TABLE](https://dev.mysql.com/doc/refman/5.5/en/create-table.html) statement

**CREATE TABLE EMP (emp\_Id INT, name CHAR (30)) ENGINE = MEMORY;**

insert records in this table using following commands, you can try using script to populate this table with multiple records.

**insert into EMP values(1,’anil’); insert into EMP values(2,’sunil’);**

To display the records in EMP table use

**select \* from EMP**

As indicated by the engine name, MEMORY tables are stored in memory. They use hash indexes by default, which makes them very fast for single-value lookups, and very useful for creating temporary tables. However, when the server shuts down, all rows stored in MEMORY tables are lost. The tables themselves continue to exist because their definitions are stored in .frm files on disk, but they are empty when the server restarts.

To load the data in memory from other existing table use,

**CREATE TABLE EMP\_temp (emp\_Id INT, name CHAR (30))) ENGINE=MEMORY as SELECT \* FROM EMP;**

To move the data from In-Memory table to hard drive (using any text file) use the following syntax,

**SELECT \* INTO OUTFILE 'emp\_data.txt' FROM EMP;**

To populate a MEMORY table when the MySQL server starts, use the INFILE option. For example,

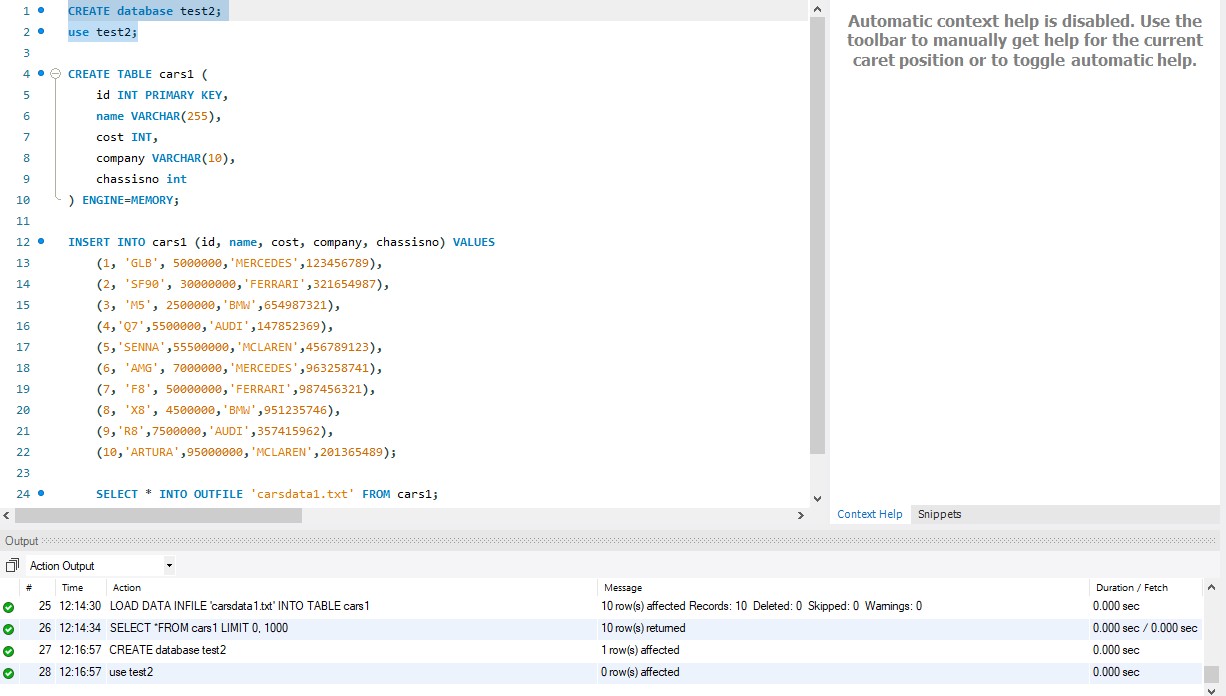
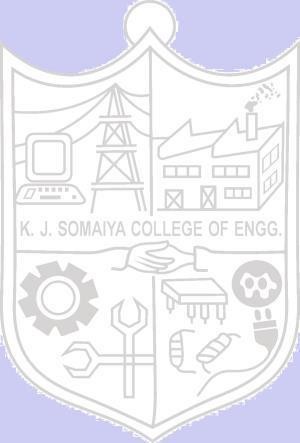
**LOAD DATA INFILE 'emp\_data.txt' INTO TABLE EMP;**

Where, emp\_data.txt is a data file.

# Procedure:

Perform following tasks:

1. Create In-memory table using Engine as Memory.
2. Insert values in that table.
3. Attempt to retrieve values from the table after restarting the database server.

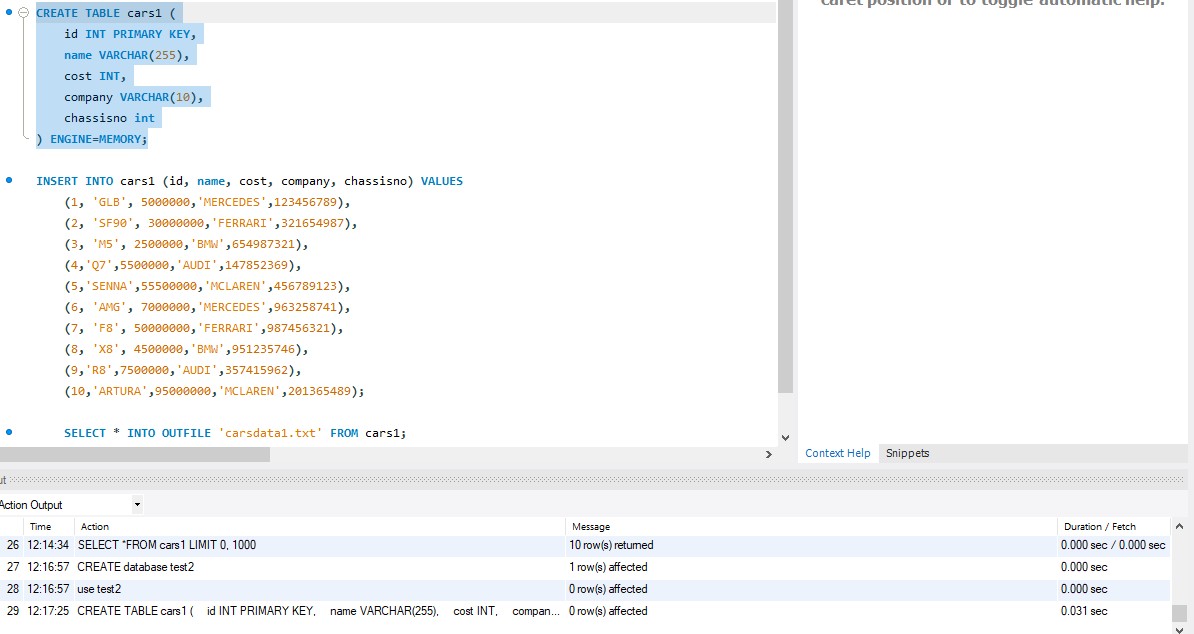


1. Load the data into table using file load.

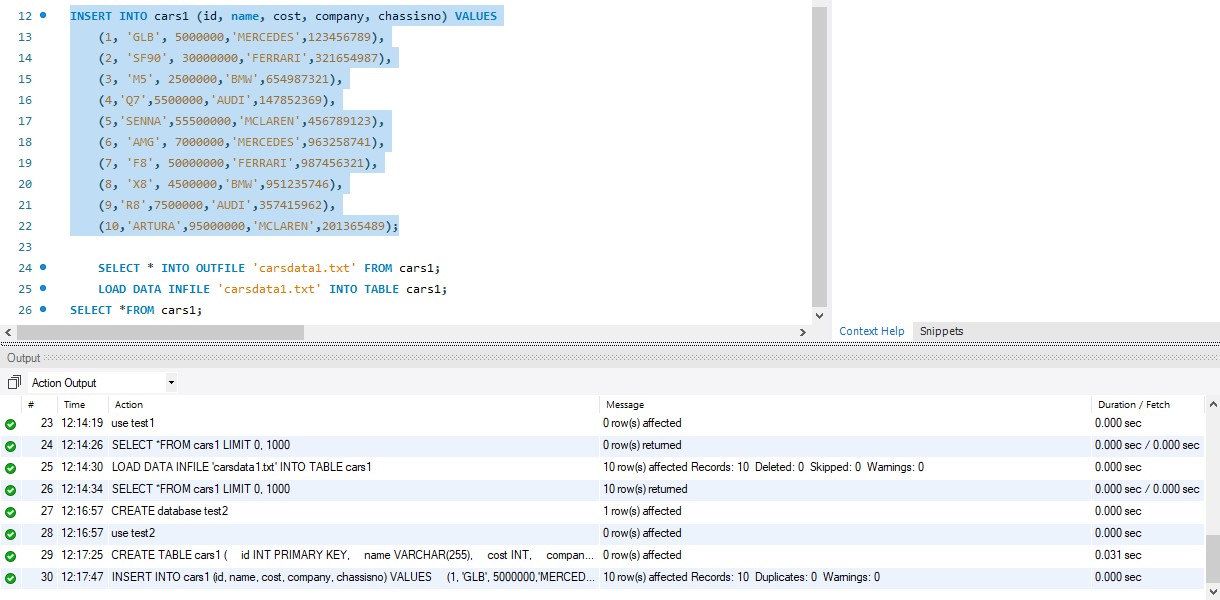
# Results: (Program printout with output)

# 

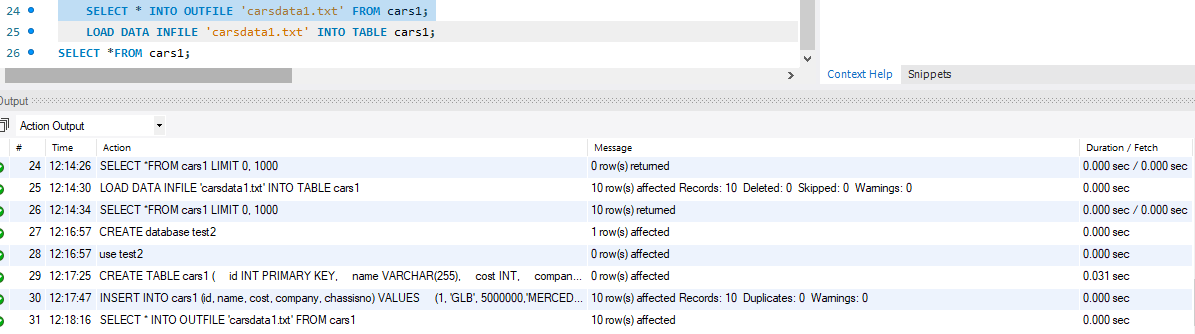
**TIME FOR CREATING AND USING DATABASE: 0.000s**



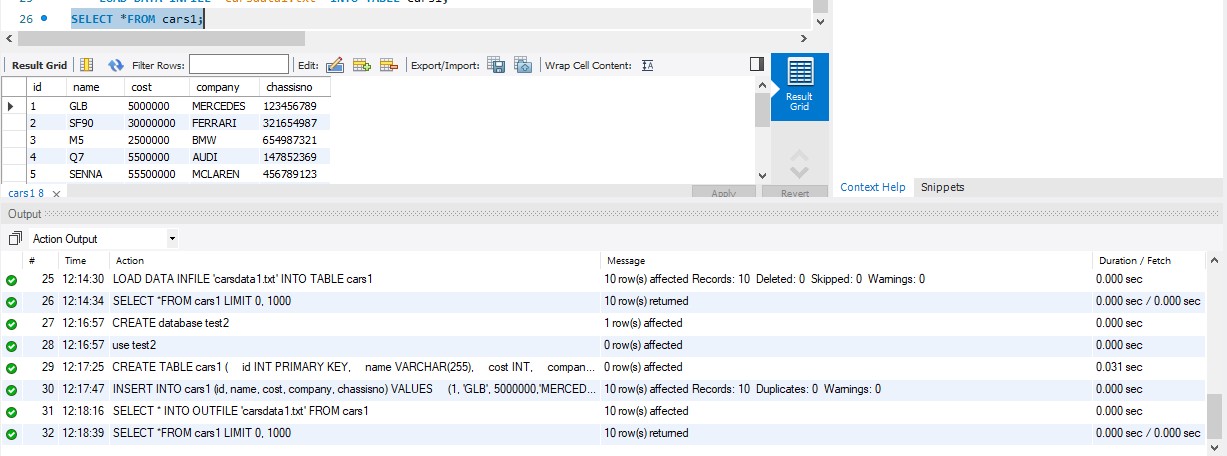
**TIME FOR CREATING TABLE: 0.031 sec**



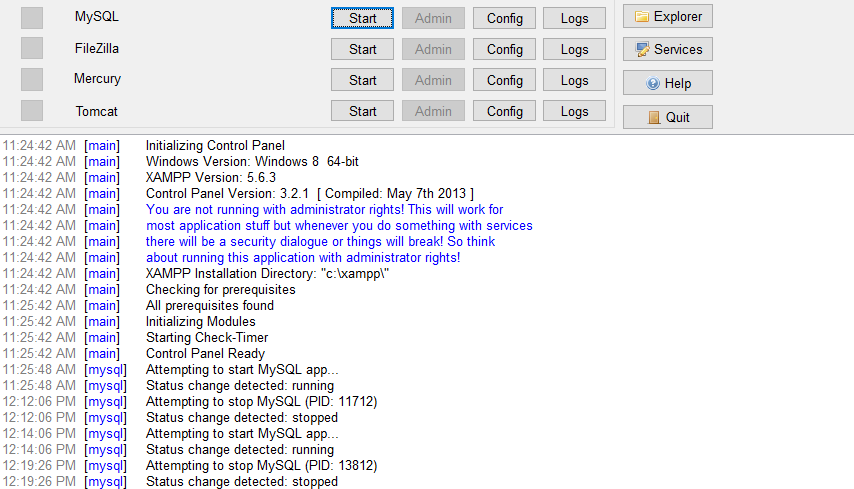
**TIME FOR INSERTING DATA INTO TABLE: 0.000 sec**



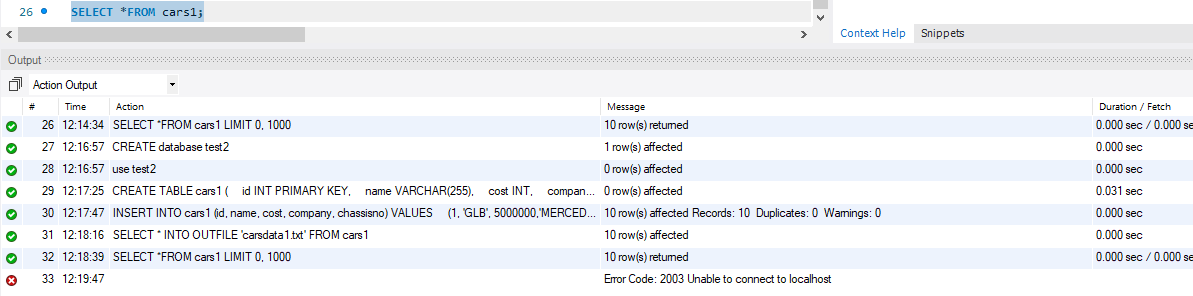
**TIME FOR PUTTING DATA IN TXT FILE: 0.000 sec**



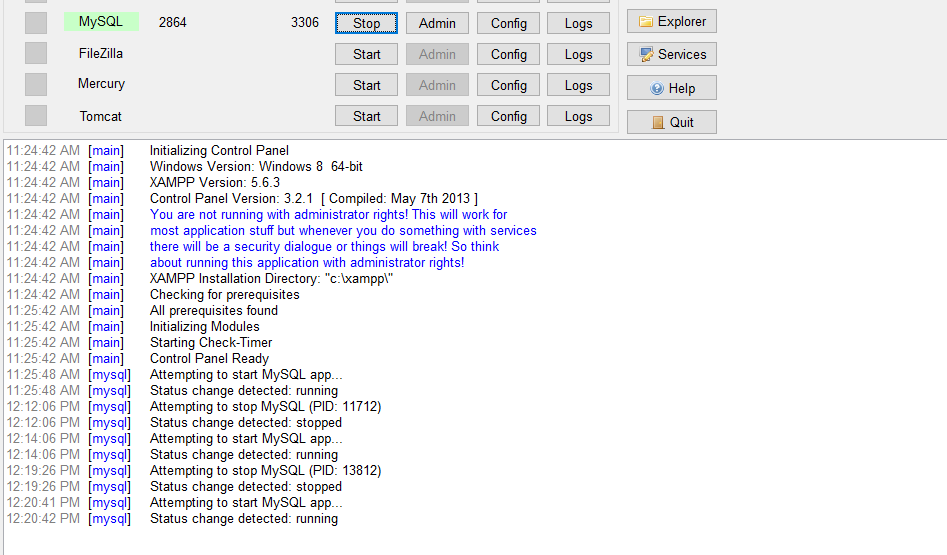
**TIME FOR DISPLAYING TABLE: 0.000 sec**



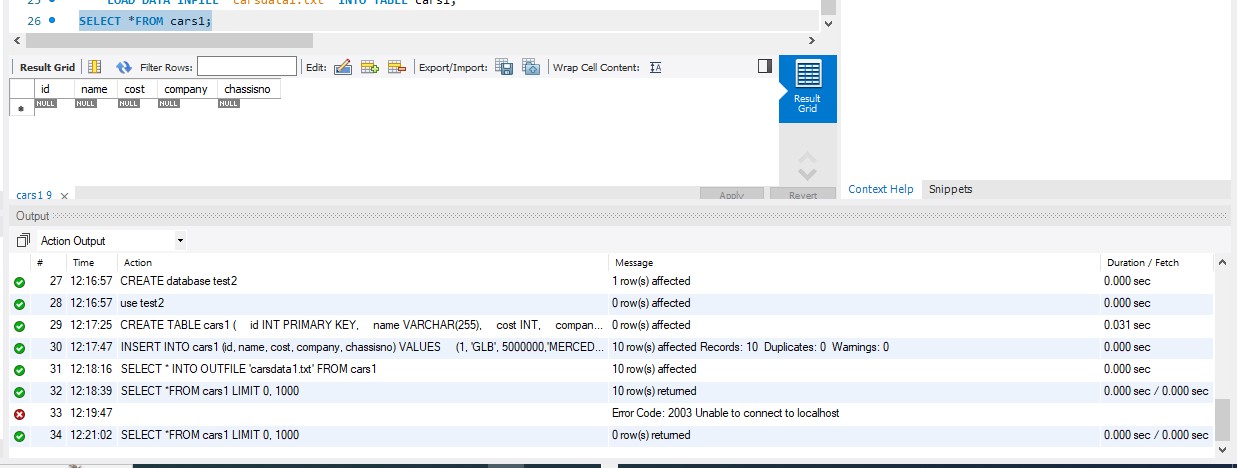
STOPPED XAMPP SERVER



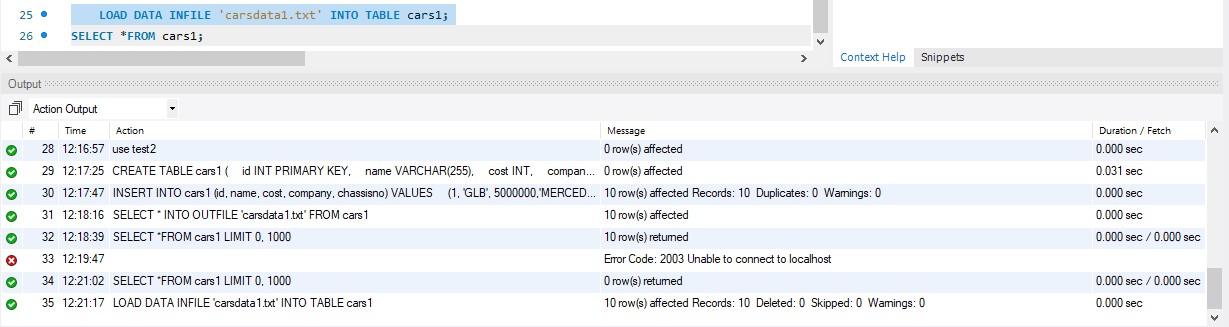
UNABLE TO CONNECT TO LOCALHOST AFTER STOPPING XAMPP SERVER



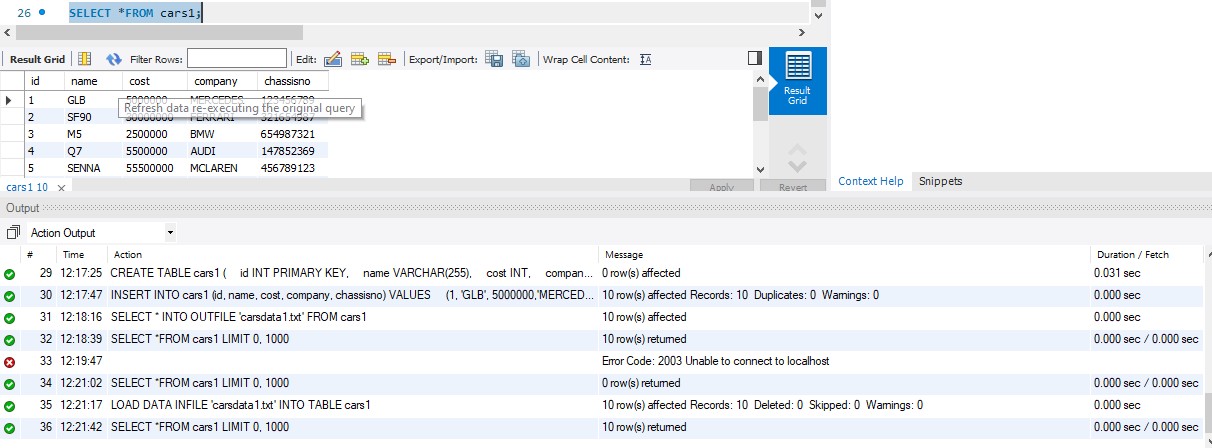
XAMPP SERVER RESTARTED



**TIME FOR DISPLAYING TABLE AFTER RESTRATING XAMPP SERVER: 0.000 sec**



**TIME FOR LOADING DATA INTO TABLE FROM TEXTFILE: 0.000 sec**



**TIME FOR DISPLAYING DATA AFTER LOADING FROM TEXTFILE: 0.000s/0.000s**

# Questions:

1. **What is the difference between traditional and In-memory databases?**

# Data Storage:

* Traditional databases store data on disk, leading to slower read and write operations.
* In-memory databases store data in the system's main memory, enabling faster access times.

# Performance:

* Traditional databases may have slower query performance due to disk I/O delays.
* In-memory databases deliver faster performance as they access data directly from RAM.

# Data Retrieval:

* Traditional databases fetch data from disk, causing latency in retrieving information.
* In-memory databases retrieve data directly from RAM, reducing retrieval time significantly.

# Optimization for Analytics:

* Traditional databases are optimized for transactional processing.
* In-memory databases are designed for analytical processing, making them suitable for complex queries and data analytics.

# Scalability:

* Traditional databases may face scalability challenges as data volume increases.
* In-memory databases can scale more easily by adding more RAM to the system, supporting increased data loads efficiently.

# List applications using in-memory database. Explain any one of it stressing upon advantage of using in-memory database.

In-memory databases find application in real-time analytics, where rapid data analysis is essential. For instance, in real-time fraud detection for financial transactions, in- memory databases provide a significant advantage by storing and analyzing transaction data instantly. The key benefit lies in the speed of processing, facilitated by keeping the entire dataset in memory. This enables quick identification of anomalous patterns and timely responses to potential fraudulent activities. In industries like finance, the use of in-memory databases enhances the ability to detect and prevent fraudulent behavior, leading to more effective and immediate risk mitigation.

# Outcomes: Design advanced database systems using In-Memory databases and its implementation.



**Conclusion: (Conclusion to be based on outcomes achieved)**

In conclusion, the experiment focused on executing MySQL queries using the MEMORY storage engine for in-memory databases. Characteristics and use cases of MEMORY tables were explored, emphasizing their susceptibility to data loss during server halts. Practical procedures included table creation, data insertion, and retrieval after server restarts. The theoretical background underscored the advantages of in-memory databases, particularly in real-time analytics. Overall, the outcomes highlighted the potential for designing and implementing advanced database systems using in-memory technology.

# Grade: AA / AB / BB / BC / CC / CD /DD Signature of faculty in-charge with date

**References:**

1. https://dev.mysql.com/doc/refman/5.5/en/memory-storage-engine.html
2. <http://opensourceforu.efytimes.com/2012/01/importance-of-in-memory-databases/>
3. <http://pages.cs.wisc.edu/~jhuang/qual/main-memory-db-overview.pdf>
4. [http://docs.memsql.com](http://docs.memsql.com/)