Software Architecture Case Study Report for MySQL

Team 28 Members:

1. Sarthak Singhal (20171091)
2. Akash Kumar (2019201046)
3. Chaitanya Kumar P (2020201012)
4. Samanvaya Panda(2018121001)

18th April, 2021

1. Introduction

MySQL is an open source relational database management system. That is, it helps in organising the data in the form of tables which helps to preserve the data in a structured way. It can be broken into My+SQL. Here My stands for co-founder's daughter name and SQL for the query language. Using MySQL we can directly interact with the database using SQL. Along with this it is also used with other programs to implement large scale applications. Some of the popular customers of MySQL are Facebook, Youtube, Twitter etc.

MySQL was created by a swedish company, founded by Swedes David Axmark, Allan Larsson and Finland Swede Michael Wideinus. Original development of MySQL by Widenius and Axmark began in 1994 and the first version was released in 1995. Initially it was created for personal usage but gradually versions came up and it was released for everyone to use [9].

Release	General availability	Latest minor version	Latest release	End of support
5.1	14 November 2008; 12 years ago	5.1.73	2013-12-03	Dec 2013
5.5	3 December 2010; 10 years ago	5.5.62	2018-10-22	Dec 2018
5.6	5 February 2013; 8 years ago	5.6.51	2021-01-20	Feb 2021
5.7	21 October 2015; 5 years ag0	5.7.33	2021-01-18	Oct 2023
8.0	19 April 2018; 2 years ago	8.0.23	2021-01-18	Apr 2026

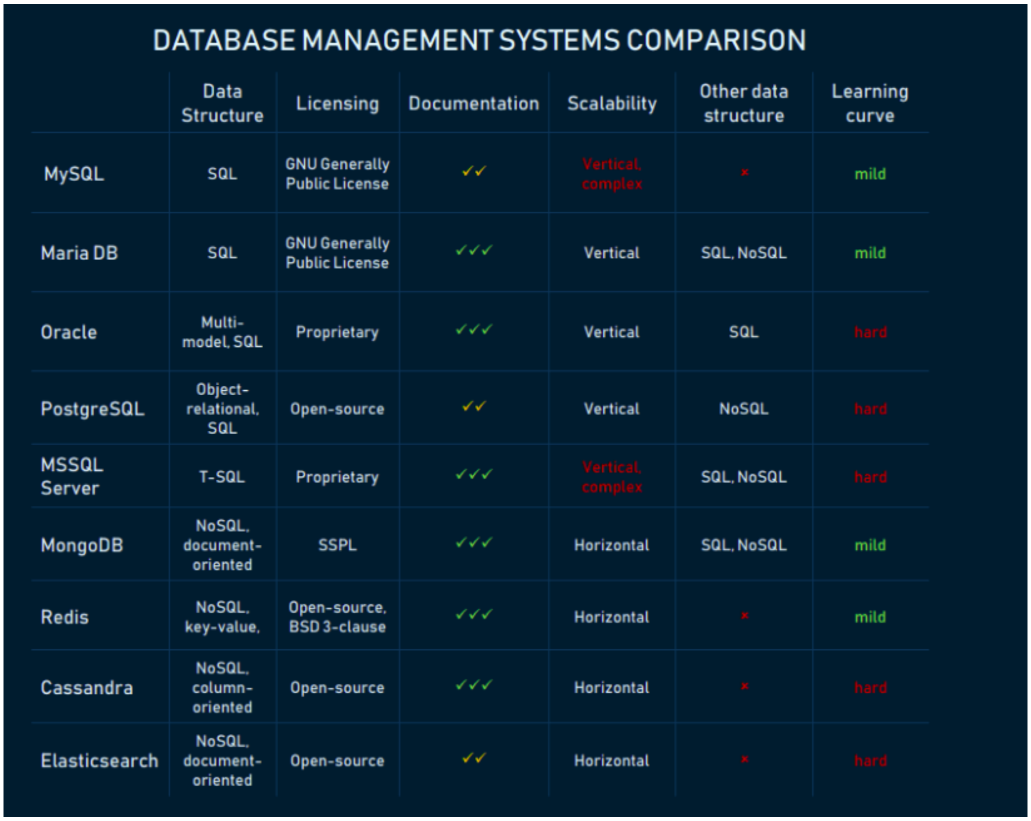
Recent release history

Some of the features that MySQL offers in processing the transactions are:

- cross platform support
- triggers
- updateable views
- ACID compliance
- query caching
- indexing and searching

Apart from these, several new updates are released approximately every 2-3 months which are available on its website or from Github repo. Also it is compatible with several third party user interfaces (MySQL Workbench being the most famous one of them). Apart from this, other GUI tools are Adminer, DBEdit etc. Other features include [8]:

- Scalability and flexibility
- High performance
- Robust transactional support
- Data protection
- Comprehensive app development
- Low costs



DATABASE MANAGEMENT SYSTEMS COMPARISON

	Data Structure	Licensing	Documentation	Scalability	Other data structure	Learning curve
MySQL	SQL	GNU Generally Public License	✓✓	Vertical, complex	×	mild
Maria DB	SQL	GNU Generally Public License	✓✓✓	Vertical	SQL, NoSQL	mild
Oracle	Multi-model, SQL	Proprietary	✓✓✓	Vertical	SQL	hard
PostgreSQL	Object-relational, SQL	Open-source	✓✓	Vertical	NoSQL	hard
MSSQL Server	T-SQL	Proprietary	✓✓✓	Vertical, complex	SQL, NoSQL	hard
MongoDB	NoSQL, document-oriented	SSPL	✓✓✓	Horizontal	SQL, NoSQL	mild
Redis	NoSQL, key-value,	Open-source, BSD 3-clause	✓✓✓	Horizontal	×	mild
Cassandra	NoSQL, column-oriented	Open-source	✓✓✓	Horizontal	×	hard
Elasticsearch	NoSQL, document-oriented	Open-source	✓✓	Horizontal	×	hard

Comparison among different DBMS

In section 2 we'll discuss the main functional requirements of the MySQL or any such RDBMS systems. Later we will go through the quality attributes that the system possesses and finally on the transaction principles of MySQL. In the section3 we'll look into the architectural characteristics of MySQL along with the kind of architecture that MySQL adopts with the help of graphics of different layers and their functionalities. We will also look at the storage technologies of MySQL and the features of the storage technologies available and how it achieves the recovery and crash management. Finally we'll look at the patterns of the architecture and how the architecture achieves several requirements and qualities.

2. Requirements and Qualities:

Like any other relational database management system, MySQL provides various functionalities that are essential for its users. Other than providing basic functionalities like reading, writing and deleting, MySQL provides its users with aggregate functions, control-flow functions, etc. Some of the functionalities provided by MySQL are:-

- **Aggregate Functions** - Aggregate functions like MAX, MIN, SUM, AVG and COUNT take a set of entries as input and output a single number.
- **Comparison Functions** - Comparison functions like GREATEST, LEAST, ISNULL and COALESCE help in comparing different attributes.
- **Control-Flow Functions** - Control-Flow functions like CASE, IF, IFNULL and NULLIF provide the users with the option of changing the flow of execution of SQL queries based on values of certain attributes.

MySQL also provides various other functionalities like date functions, string functions, window functions, math functions, etc. MySQL is very easy to use which is one of its key quality attributes. Anyone with minimal knowledge of SQL can use the system without much difficulty. It also uses a client/server architecture that makes interacting with data simpler. MySQL provides an essential security layer to protect sensitive data from malicious adversaries. Another key quality attribute of MySQL is that being open source, it is supported by most of the operating systems like Windows, Linux, Unix, FreeBSD, etc. Moreover, it is faster, scalable, and cheaper for its storage engine architecture. MySQL also supports the recovery of data through rollbacks and log management. It is stable, highly flexible, provides high productivity and efficient data management. The quality attributes based on the utility they provide to the MySQL system could be ranked in the following order:-

- **Security** - MySQL provides one of the most secure and reliable database management systems by providing an additional security layer through various authorization and authentication mechanisms.
- **Scalability** - MySQL provides high scalability which can handle terabytes of data. It also supports on-demand scalability which allows users to scale up or scale down their infrastructure based on their requirements.
- **Performance** - MySQL provides high performance with a variety of configurations for the users. It also provides an assurance of 24X7 uptime and other high availability solutions like clustered servers and master/slave replication configurations.

- **Transactional Support** - MySQL provides robust transactional support that is completely atomic, consistent and isolated. It also provides rollbacks, multi-version transaction support, etc to ensure full data integrity. It can be updated asynchronously and is eventually consistent in terms of all the transactions.
- **Workflow Control** - MySQL allows customized configurations for each user making it easier for data management irrespective of the operating system used.
- **Open Source** - MySQL is open source. So it reduces the cost of ownership for users who could opt for MySQL instead of enterprise software. It also makes maintenance and upgrades fast and easy.
- **Separation of concerns**: MySQL follows the layered architecture. So, each functionality/ concern is handled at each level. It follows a hierarchical layered approach which will be discussed in the following sections.

MySQL also abides by the ACID principles that ensure that all its transactions are processed reliably. ACID stands for Atomicity Consistency Isolation Durability. These properties are ensured by MySQL in the following ways:-

- **Atomicity** - The atomic property implies that either the entire transaction is executed or it is discarded. Partial execution of a transaction is not allowed. To ensure atomicity, MySQL commits to the log only when the entire transaction is completed successfully.
- **Consistency** - Consistency property refers to the fact that a transaction doesn't violate any data integrity constraint. MySQL ensures consistency by discarding a transaction and rolling back to the previous commit if the database enters into an illegal state.
- **Isolation** - Isolation property states that all the transactions are executed in a particular order and no two transactions are executed simultaneously that affect each other. Isolation property is ensured in MySQL by using optimistic and pessimistic transactions.
- **Durability** - The durability property implies that all the transactions that have been executed successfully have been committed properly and will be available in the future even if the system fails. It ensures that the data in the database is not corrupted even if the system crashes or there is a power outage. MySQL ensures durability through logs.

3. Architectural Solution:

MySQL is different from other database servers. The characteristics of it's architecture makes it useful for several purposes. It is also a poor choice for some purposes. It works very well in applications such as web, data warehouses, delivery software, online transaction processing(OLTP), etc.

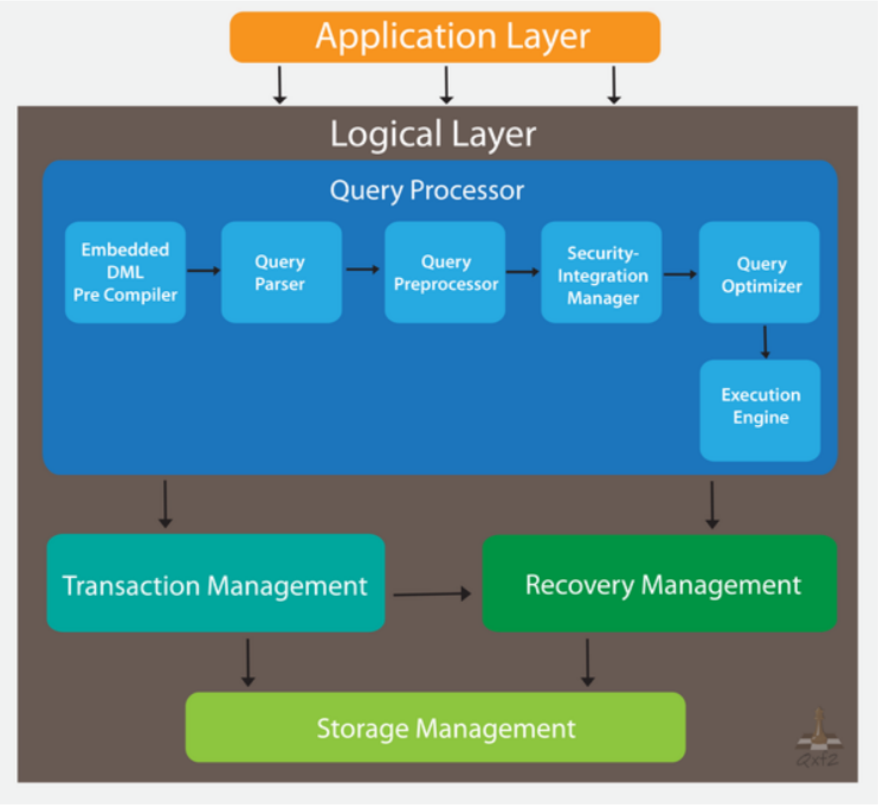
Before learning about MySQL architecture, we need to know the general RDBMS architecture. According to *Garlan and Shaw* all database systems can be viewed as layered architecture[3].

Why a layered approach ?

"Layered architecture implies that separation of concerns has been achieved. It is the reason MySQL went with layered architecture and it is easy to maintain when things can be distinguished easily."

It has three main components:

- Application Layer
- Logical Layer
- Physical Layer



Application Layer:

Users and clients interface with the MySQL RDBMS[6].

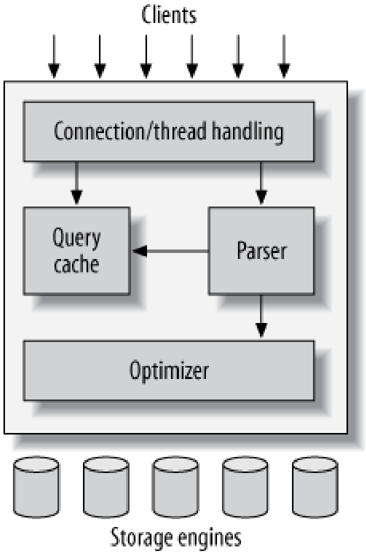
There are three components:

- Administrators
- Clients
- Query Users

Query users interact with MySQL RDBMS with the help of "mysql". Administrators use various types of interfaces and utilities like mysqladmin, isamchk etc[4]. Various interfaces and APIs help clients to communicate with MySQL RDBMS. Users can enter their queries(SQL statements) in the "mysql query interface" and observe the results. MySQL typically follows the '*component-and-connect'* architecture pattern. Specifically speaking, it follows the '*client-server*' pattern. This way we are ensuring the *separation of concerns.*

Logical Layer:

Usually, in a client-server pattern, the clients are distributed across many locations and wish to access shared resources and services (It is a many to one relationship). The server is the central program that manages the database contents and client programs connect to the server to retrieve or modify the data. The top layer is common in many network-based client/server architectures.



MySQL also includes *non-client utility* programs and scripts. Following are the services present in the topmost layer[1].

- Connection Handling.
- Authentication.
- Security.

The second layer comprises query parsing, analysis and optimization, caching and other built-in functions(e.g., dates, times, math and encryption). Any functionality that makes use of storage engines is provided in this layer. This layer is also called a query layer. The third layer contains storage engines. Storage engines allow us to store and later retrieve all the information. It is just like a file system and it had its pros and cons. The *storage engine API* allows for server communication. The API contains some low-level functions that perform operations such as *begin a transaction*, *fetch the row* that has the primary key. Storage engines simply respond to the requests from the server and they don't parse any query.

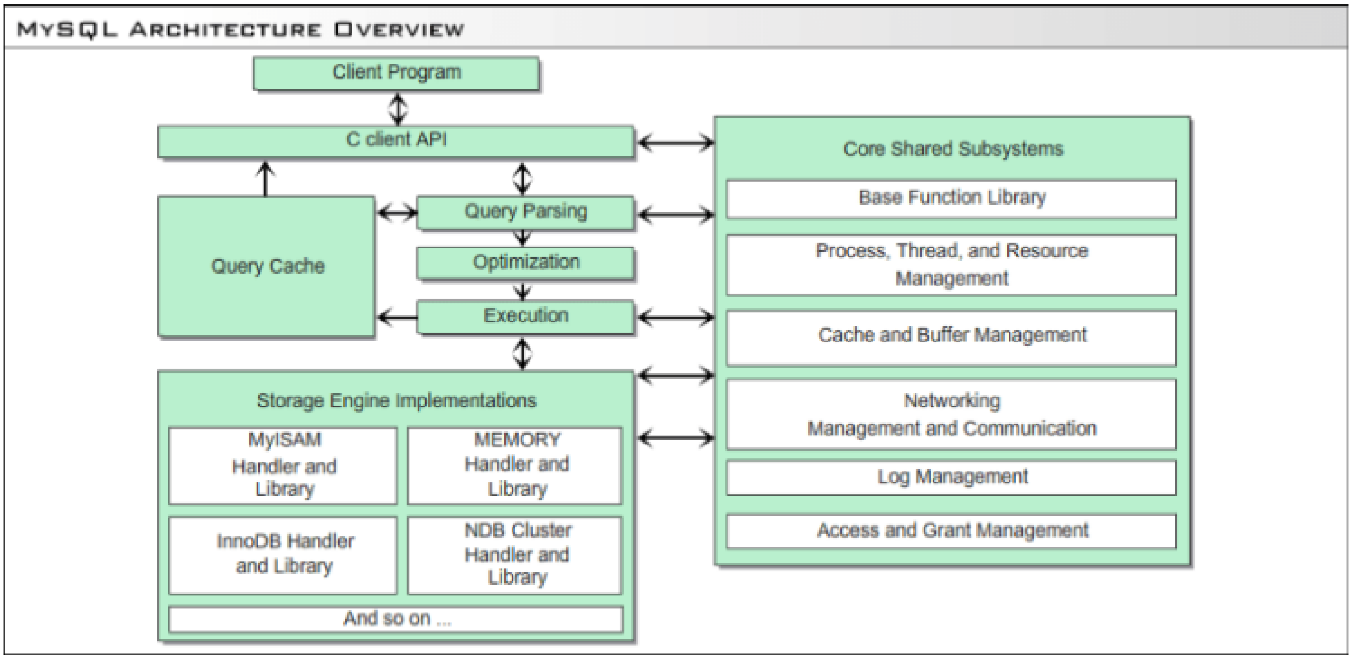
If we see the architecture overview, the client connects with the help of a connector and different systems have different types of connectors as client API. Few such examples are 'Connector/ODBC', JDBC, .NET, PHP, Python Json which supports connectivity from Windows, Unix, and macOS platforms. Many such connectors can be found in [2].

All the functionalities like Query Parsing, Optimization, and Execution that we see below the client API come under the Logical layer's Query processing step. Both client API and Query processor interact with the shared subsystems to effectively process the query and fetch the results. These things can be encountered when we move further down in the architecture.

The *logical layer* of MySQL architecture comprises the following subsystems.

- Query Processor
- Transaction Management

- Recovery Management
- Memory Management


MYSQL ARCHITECTURE OVERVIEW
Client Program
C client API
Query Cache
Query Parsing
Optimization
Execution
Storage Engine Implementations
MyISAM Handler and Library
MEMORY Handler and Library
InnoDB Handler and Library
NDB Cluster Handler and Library
And so on ...
Core Shared Subsystems
Base Function Library
Process, Thread, and Resource Management
Cache and Buffer Management
Networking Management and Communication
Log Management
Access and Grant Management

Overview of the functionalities of each layer in the logical diagram.

1) Query Processing:

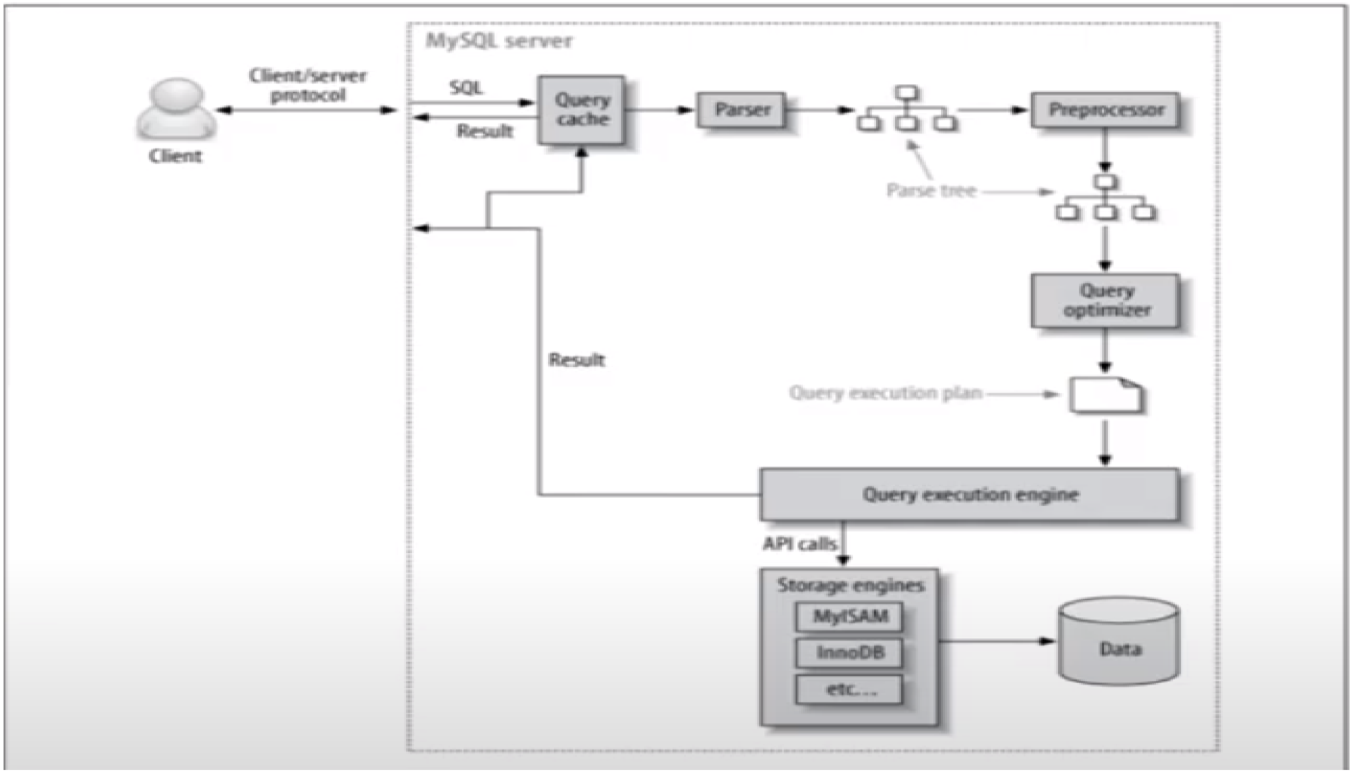
MySQL parses the queries to create an internal structure (the parse tree) and then applies various optimizations. These can be rewriting the query, determining the order in which it needs to read the tables, choosing indexes on certain columns. Optimizations are configurable by the user. It does not depend on the storage engines [1]. The process of query parsing and execution is shown in the below diagram.

It consists of the following components:

- Embedded DML Precompiler
- DDL compiler
- Query Parser
- Query Preprocessor
- Security/Integration Manager
- Query Optimizer
- Execution Engine

2) Transactions Management:

- It facilitates the concurrent data accesses.
- Provides the locking facility
- Make sures that data or sessions can be accessed simultaneously in a consistent manner..
- Prevent data corruption or data damage.
- Lock Manager is responsible for handling locking and it is the sub component of the Transactions Management.



3) Recovery Management:

Log Manager:

- Logs every operation executed in the database.
- Stores the operating logs as MySQL commands.
- In the case of SYSTEM crash executing these commands will bring back the database to its last stable state

Recovery Manager:

- Recovery manager checks the logs and identifies the elements that are changed and returns the database to its stable state
- Logs are created by the log manager and recovery manager makes use of it to recover the database to a stable state.

4) Memory Management:

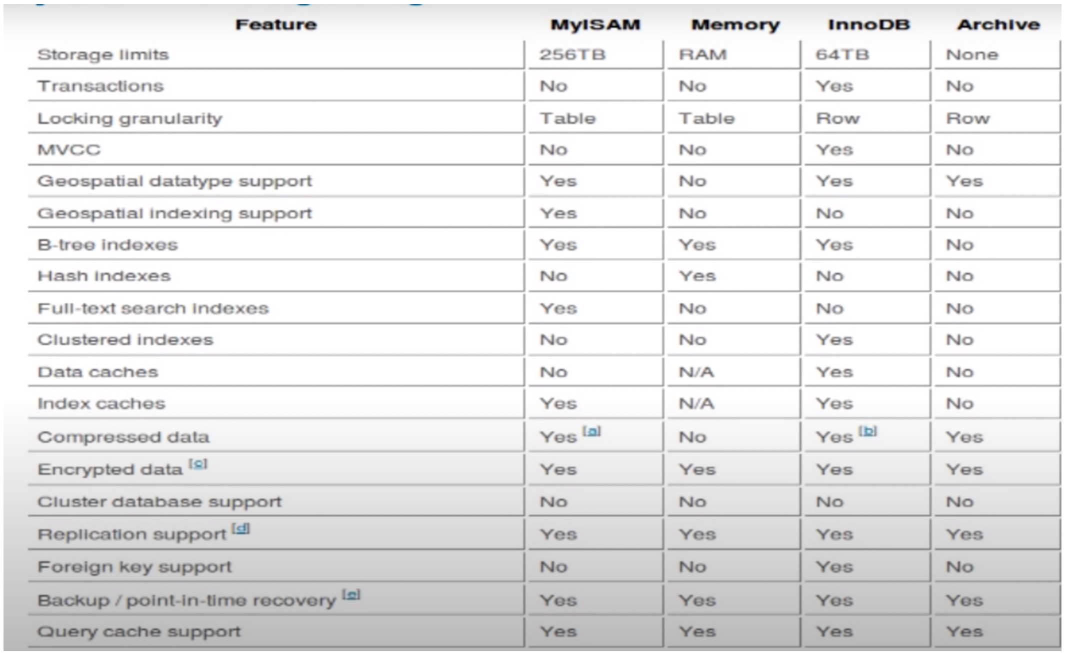
Buffer Manager:

- It allocates the memory resources for various transactions.

Resource Manager:

- Accepts the requests from the execution engine.
- Requests the details from the buffer manager.
- It receives addresses of the data in the memory with the help of a buffer manager.
- Returns this data to the upper layer

Storage management as shown in the above architecture overview contains many *pluggable storage elements.* MyISAM is a storage engine where the data will be gone when we restart the server. It is used to contain schema of the database and some metadata . InnoDB is more persistent and used for transaction based purposes. NDB is used in case of clustered data management i.e some data is present in one node and other data is present in some other node like that. Similarly there are several types of storage engines with different purposes. The default storage engine is *InnoDb* for MySQL at present.(MySQL 5.5 and above)[5].



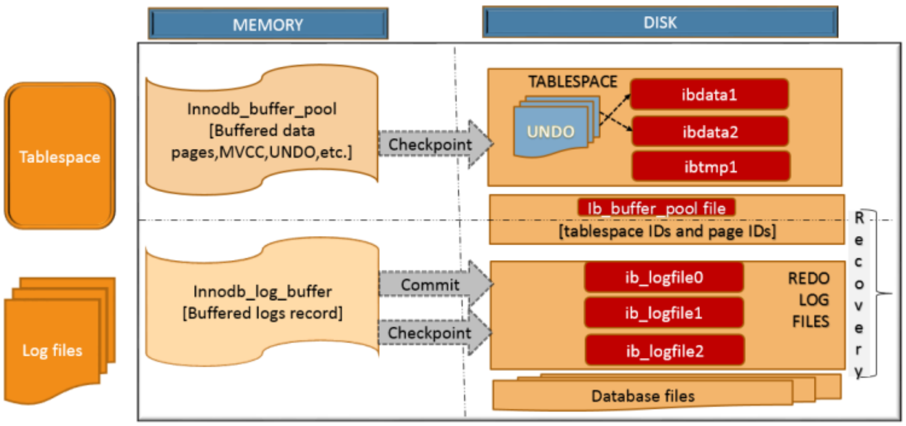
Feature	MyISAM	Memory	InnoDB	Archive
Storage limits	256TB	RAM	64TB	None
Transactions	No	No	Yes	No
Locking granularity	Table	Table	Row	Row
MVCC	No	No	Yes	No
Geospatial datatype support	Yes	No	Yes	Yes
Geospatial indexing support	Yes	No	No	No
B-tree indexes	Yes	Yes	Yes	No
Hash indexes	No	Yes	No	No
Full-text search indexes	Yes	No	No	No
Clustered indexes	No	No	Yes	No
Data caches	No	N/A	Yes	No
Index caches	Yes	N/A	Yes	No
Compressed data	Yes [a]	No	Yes [b]	Yes
Encrypted data [c]	Yes	Yes	Yes	Yes
Cluster database support	No	No	No	No
Replication support [d]	Yes	Yes	Yes	Yes
Foreign key support	No	No	Yes	No
Backup / point-in-time recovery [e]	Yes	Yes	Yes	Yes
Query cache support	Yes	Yes	Yes	Yes

Characteristics of the storage engines

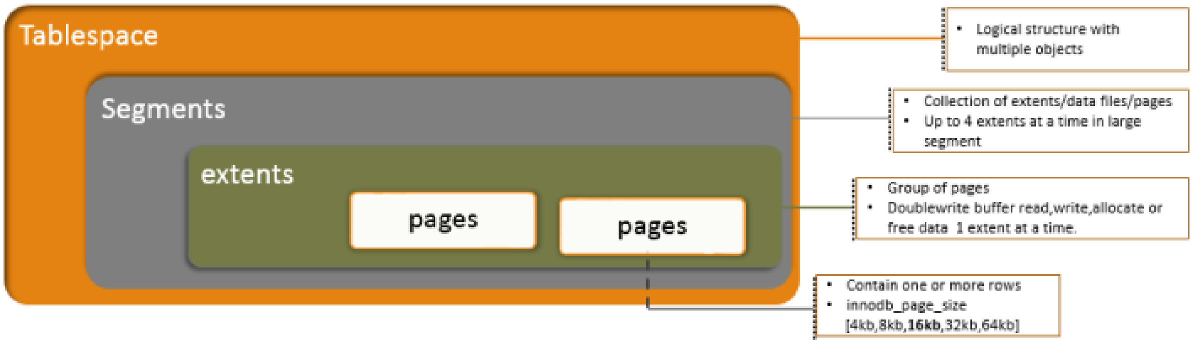
InnoDB:

- InnoDB storage engines provide safe transactions on tables such as commit, rollback, and crash recovery.
- It supports row-level locking.
- Fully transactional ACID.
- Offers REDO and UNDO for transactions.
- Table can extend to any size and supports power loss recovery and it is quick.
- This reduces the I/O for common queries based on primary keys.
- InnoDB is majorly used for applications that have data integrity.
- Uses a shared file to store objects i,e. Both the Data and Index can be in the same file.
- InnoDB reads a physical data and builds a logical structure i.e. blocks and rows

InnoDB storage engine architecture:

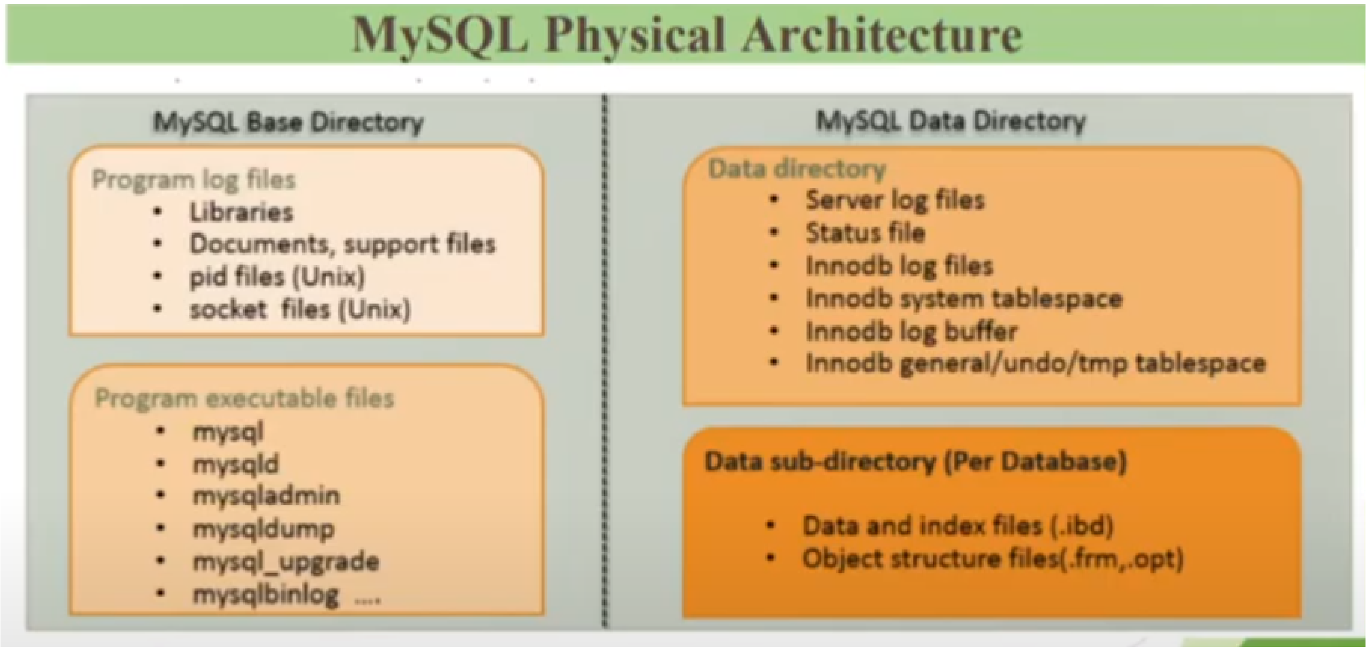


Logical Storage in InnoDB is called TABLESPACE. Storage for InnoDB is divided into tablespaces. A tablespace is a logical structure which contains multiple data files(objects). Each table space contains pages(blocks), extents and segments. It is shown below.


Tablespace
Segments
extents
pages
pages
Logical structure with multiple objects
Collection of extents/data files/pages
Up to 4 extents at a time in large segment
Group of pages
Doublewrite buffer read,write,allocate or free data 1 extent at a time.
Contain one or more rows
innodb_page_size [4kb,8kb,16kb,32kb,64kb]

Physical Layer:

It contains the actual file systems and contains the libraries, program log files,status files etc.,. MySql employs a separate file for Table. Physical Layer is shown in the below image. The Physical Layer contains the storage engines. They take responsibility for storing and retrieving all the information or data in MySQL. The Physical Layer of MySQL is slightly different from other RDBMS. Physical system supports Pluggable Storage Engine architecture that enables us to load or unload storage engines to or form a running MySQL server. MySQL stores database schema as a special sub directory of its data directory. Every database has a corresponding data directory. Creation of tables in MySQL enforces the storage of the table definition in a *.frm* file whose name is the same as that of table. For example when you create a table named *EMPLOYEE* MySQL stores the table definition in *EMPLOYEE.form*.These .frm files do not store data but only have the format that contains the description of the table structure.



MySQL Physical Architecture

MySQL Base Directory

Program log files
- Libraries
- Documents, support files
- pid files (Unix)
- socket files (Unix)

Program executable files
- mysql
- mysqld
- mysqladmin
- mysqldump
- mysql_upgrade
- mysqlbinlog ....

MySQL Data Directory

Data directory
- Server log files
- Status file
- Innodb log files
- Innodb system tablespace
- Innodb log buffer
- Innodb general/undo/tmp tablespace

Data sub-directory (Per Database)
- Data and index files (.ibd)
- Object structure files(.frm,.opt)

The RDBMS stores a wide variety of information, usually kept in secondary storage and accessed with the help of a storage manager. Few of such data kept in the system are:

1. User data files
2. Meta data files which are called data dictionaries
3. Indexes were present on certain attributes for faster access to data.
4. Statistical Data, which stores statistical information about the data in the database; Query makes use of such statistical data to decide the effective query plans.
5. Log information files that helps achieving recovery management

At the physical level, storage locations of data objects are stored as information. Such information is abstracted from normal users and the physical level employs several storage means such as disks tapes and also stores insights on the storage details of such means.

1) Achieving Security requirement:

- Mysql enforces to select good passwords .
- Limits privileges to users.
- A client program does not know the identity of the person running it.
- MySQL will refuse to run as a root in the UNIX because it is extremely dangerous and any user with the privileges for file can read the data.
- MySQL has an extra connection for privileged users known as SUPER. Only they can login and check the server activity
- Supports logs for recovery in case of crashes and failures

These are some security guidelines that MySQL adopts to provide security and many such can be found at [10].

2) Achieving the Scalability Requirement:.

We need to scale the system such that it supports large terabytes of data. How does MySQL achieve this is as follows. It employs both *scale-out* and scale-up tactics. Users have the flexibility to go for any of the scalability tactics for MySQL server.

Scale-Up:

- It is Vertical scaling i.e increasing the procurement of expensive SMP hardware
- Using proprietary software for certain aspects.
- Used '*ForkLift'* inorder to increase the capacity and performance.

Scale-Out:

- It is Horizontal scaling i.e using commodity hardware.
- Going open source which is low cost and allows for rapid improvements.
- Achieving the platform independence.
- Adding more servers to match with the increasing load as well as increasing the capacity and performance.

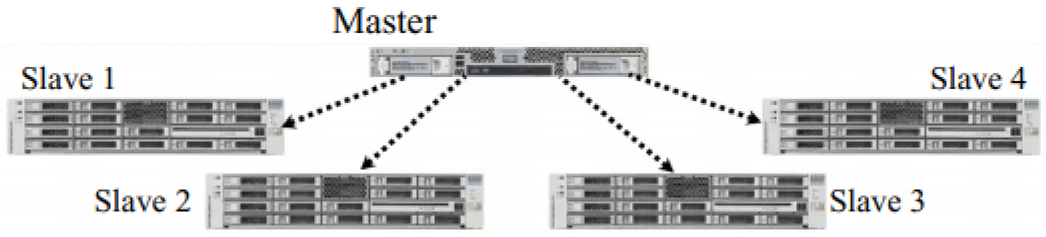
3) Achieving high performance:

MySQL supports multi-threading. It means that whenever a new connection is requested it will open a new thread along with the connection to the data and it will be handed over to the client to perform the queries. MySQL follows thread pool tactics to manage or limit the number of threads as the threads are memory consuming. Whenever a new connection is requested then MySQL picks one of the threads from the thread pool to serve the request. If all the threads were consumed then the new connections are not accepted. Threadpool size depends on the number of cores a machine has but crossing it beyond 16 is not good it actually degrades the performance.

Architectural Design patterns:

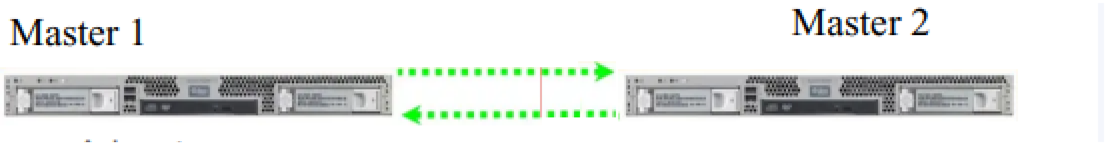
To achieve high availability, scalability and high performance there are some architectural design patterns that MySQL follows[13].

Master with Read only slaves:



A master can have as many slaves as possible with approximately 1%load on master per slave. More than 5 slaves are not unusual. Most often there will be around 80 slaves. Load balance can be read across slaves. Long running reports on slaves are possible. We can add more slaves as required. Slave can be used for backup of data. Usually JDBC drivers support this kind of pattern architecture.

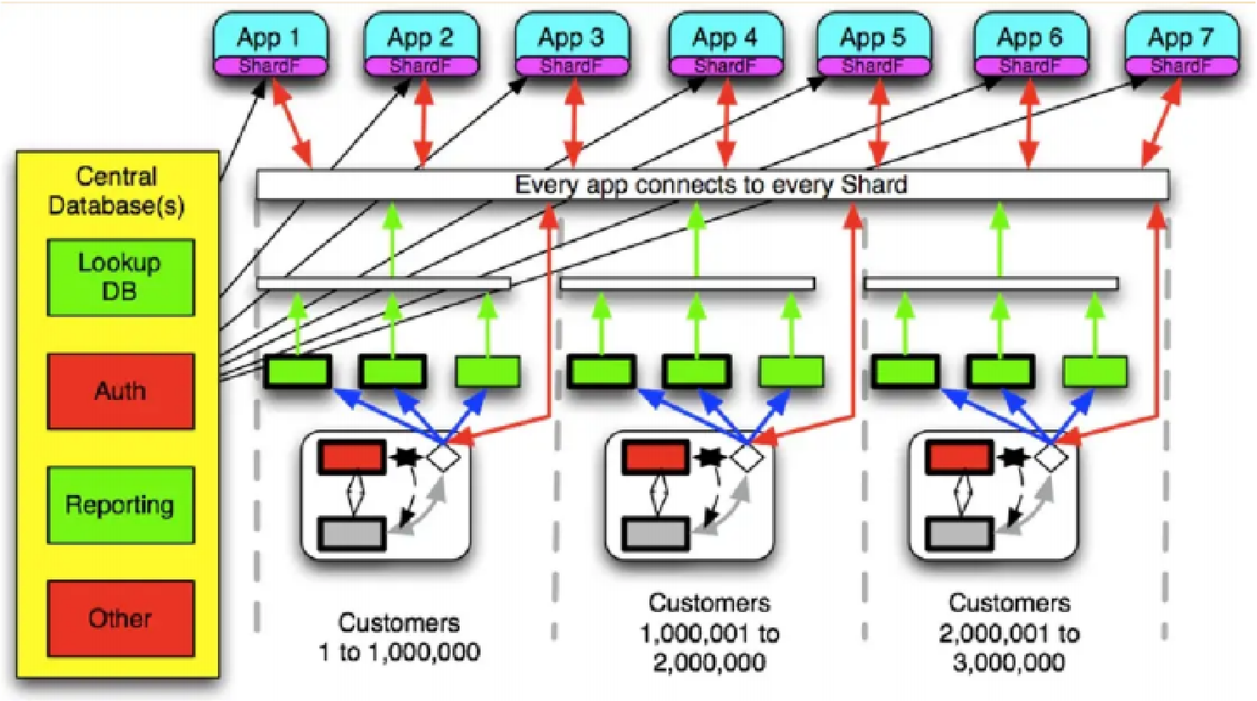
Master-Master replication :



Advantages of such a pattern are that the user can read/ write to either of the servers. It is usually employed when support is needed for more geographical locations. Either of the masters can be setup as active or passive at the time of initialization. We can increase performance by distributing the load across two masters. The above patterns actually achieves the High availability as well as scalability to some extent.

Large Sharing/Scale-out solution + DRBD:

It is also known as sharding. In this approach the database tables are partitioned into separate MYSQL RDBMS. Those are called shards. A key is associated with each shard and physical servers[11]. The architecture is shown below. It is purely scale-out architectural pattern, as we can see that as customers increase the number of shards is increasing. DRBD stands for Distributed Replicated Block device.[12].



4. Summary

Through this paper, we looked at the structure of MySQL. We started with what it is and briefly touched its history. We saw it's requirements and saw the various features that it provides. It is an important tool for processing the transactions and apart from this provides other features such as security, scalability, control etc. Through the facilities that it provides, it aims to achieve its goals which in our sense it completely achieves as it has now become the most popular RDBMS among its competitors. Through its excellent architecture and model, it serves as an exemplary model for any RDBMS based on it.

Some of the important features of the architecture is that it is layered which facilitates the maintainability, scalability and flexibility to plug in different kinds of storage engines depending on their needs makes it de facto for many small scale applications. It can be used in conjunction with other NoSQL databases and many giants like Netflix were also using this. As discussed in the above architecture, MySQL sharding results in high performance through parallel processing capability. Despite its many advantages and MySQL being open source it has some disadvantages. It is difficult to install a large cluster which needs to be consistent with the help of MySQL. It requires locking the entire table to change the structure of the large tables in case of clusters. It is one of the reasons why people prefer NoSQL databases.

Overall through the study of its architecture, we see that its efficient design is largely responsible for its long term usage and popularity which makes it a truly great architectural design.

5. References

[1] MySQL architecture and history. Retrieved from https://bit.ly/3stO6zu

[2] MySQL API's. Retrieved from https://bit.ly/3v2XADx

[3] Introduction to software architecture. Retrieved from https://bit.ly/2RLI5BN

[4] Linux man page. Retrieved from https://bit.ly/3agZEzR

[5] MySQL storage architecture. Retrieved from https://bit.ly/3dqMDFT

[6] MySQL architecture layers. Retrieved from https://bit.ly/3mUv5Fx

[7] MySQL components. Retrieved from https://bit.ly/3xdo5Zd

[8] MySQL features. Retrieved from https://bit.ly/32mTdXZ

[9] MySQL from Wikipedia. Retrieved from https://bit.ly/3ecjPAj

[10] Making MySQL secure. Retrieved from https://bit.ly/3v2XQT1

[11] DB sharding. Retrieved from https://bit.ly/3ssubBd

[12] MySQL performance. Retrieved from https://bit.ly/3x3B2Ex

8. Effort

Name	Time
Sarthak Singhal (20171091)	12 hours
Akash Kumar (2019201046)	12 hours
Chaitanya Kumar P (2020201012)	12 hours
Samanvaya Panda(2018121001)	12 hours