MySQL Database

1)Introduction:

MySQL is the most popular Open Source Relational SQL database management system. MySQL is one of the best RDBMS being used for developing web-based software applications.

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems.

So nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.

2)Funtional Requirements:

1 Function and Operator Reference
2 Type Conversion in Expression Evaluation
3 Operators
4 Control Flow Functions
5 String Functions
6 Numeric Functions and Operators
7 Date and Time Functions
8 What Calendar Is Used By MySQL?
9 Full-Text Search Functions
10 Cast Functions and Operators
11 XML Functions
12 Bit Functions
13 Encryption and Compression Functions
14 Information Functions
15 Spatial Analysis Functions
16 JSON Functions
17 Functions Used with Global Transaction IDs
18 MySQL Enterprise Encryption Functions
19 Miscellaneous Functions
20 Functions and Modifiers for Use with GROUP BY Clauses.

NonFuntional Requirements:

1)Security:

Access control and security within the database system itself, including the users and databases granted with access to the databases, views and stored programs in use within the databaseNetwork security of MySQL and your system. The security is related to the grants for individual users, but you may also wish to restrict MySQL so that it is available only locally on the MySQL server host, or to a limited set of other hosts.

2)Backup and Recovery:
It is important to back up your databases so that you can recover your data and be up and running again in case problems occur, such as system crashes, hardware failures, or users deleting data by mistake. Backups are also essential as a safeguard before upgrading a MySQL installation, and they can be used to transfer a MySQL installation to another system or to set up replication slave servers.

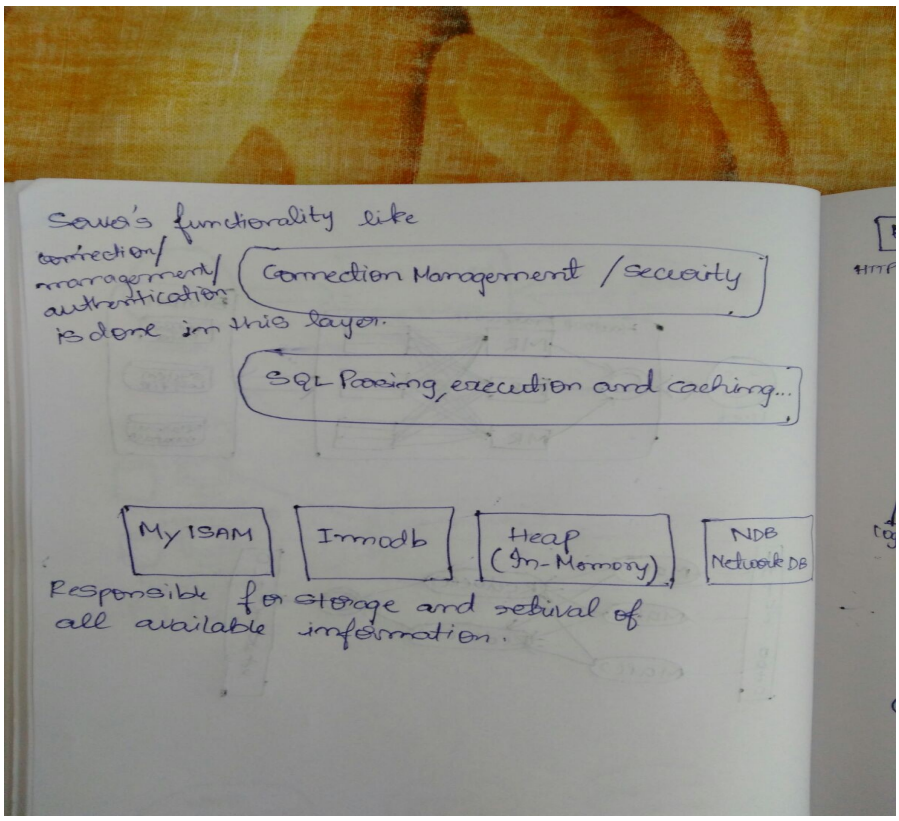
3)Optimization:
Database performance depends on several factors at the database level, such as tables, queries, and configuration settings. These software constructs result in CPU and I/O operations at the hardware level, which you must minimize and make as efficient as possible. As you work on database performance, you start by learning the high-level rules and guidelines for the software side, and measuring performance using wall-clock time. As you become an expert, you learn more about what happens internally, and start measuring things such as CPU cycles and I/O operations.

4)Replication:
Replication enables data from one MySQL database server (the master) to be copied to one or more MySQL database servers (the slaves). Replication is asynchronous by default; slaves do not need to be connected permanently to receive updates from the master. Depending on the configuration, you can replicate all databases, selected databases, or even selected tables within a database.

5)Partitioning:
The SQL standard does not provide much in the way of guidance regarding the physical aspects of data storage. The SQL language itself is intended to work independently of any data structures or media underlying the schemas, tables, rows, or columns with which it works. Nonetheless, most advanced database management systems have evolved some means of determining the physical location to be used for storing specific pieces of data in terms of the file system, hardware or even both.

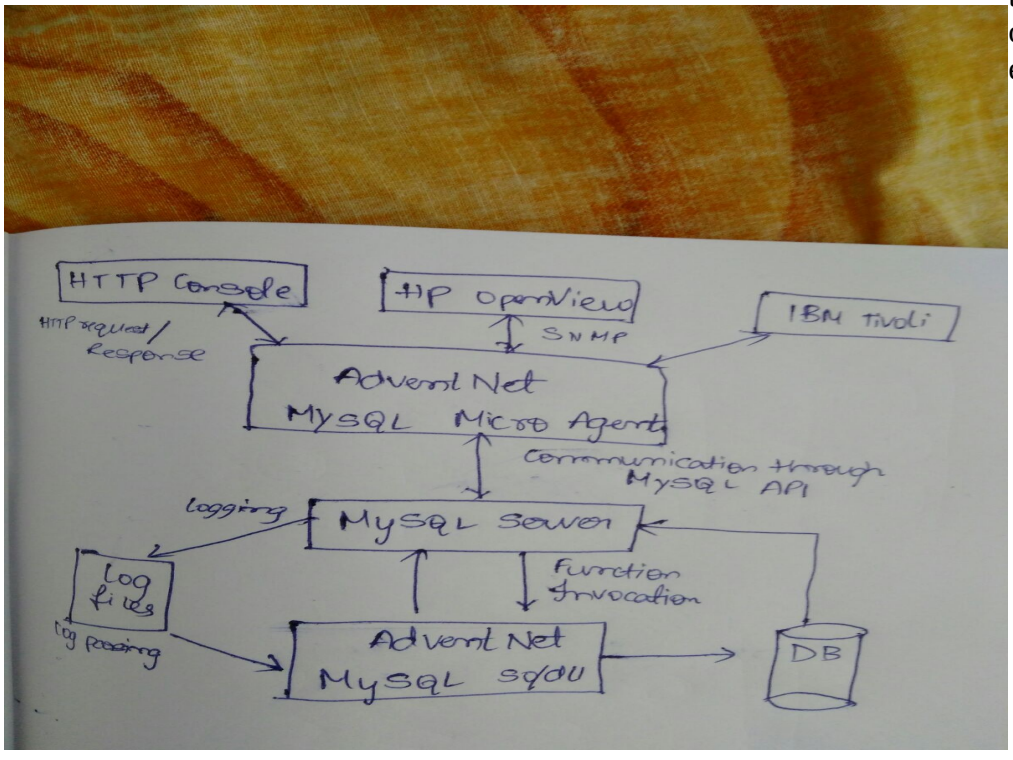
3)Architecture of MySQL Database:

The MySQL pluggable storage engine architecture enables a database professional to select a specialized storage engine for a particular application need while being completely shielded from the need to manage any specific application coding requirements.



The MySQL server architecture isolates the application programmer and DBA from all of the low-level implementation details at the storage level, providing a consistent and easy application model and API. Thus, although there are different capabilities across different storage engines, the application is shielded from these differences.The pluggable storage engine architecture provides a standard set of management and support services that are common among all underlying storage engines. The storage engines themselves are the components of the database server that actually perform actions on the underlying data that is maintained at the physical server level.

This efficient and modular architecture provides huge benefits for those wishing to specifically target a particular application need—such as data warehousing, transaction processing, or high availability situations—while enjoying the advantage of utilizing a set of interfaces and services that are independent of any one storage engine.



Work Flow of MySQL Database:

1 - This application will have several groups and these groups , several users .

2 - Each request would have its flow , multiple requests can have the same flow .

3 - The " admin " can create multiple workflows , with up to 5 steps between groups and individuals

4 - When the administrator chooses a group , not a specific user , someone would be drawn from this group .

5 - The approvals follow the order , ie , the next can only approve if the previous one has already approved .

My Design For MySQL Database:

MySQL Database's workflow is good in terms of quality, speed, security and privacy. All the components, which it uses currently, are responsible for making it what it is currently now. Many developers uses it. However, there are certain things that I would tend to improve in MySQL Database:

1)Improve performance of a Fact Table.
2)Improve the performance of a query that selects on a low cardinality column in MySQL.
3)Partitioning MySQL for “expired” transactions to improve performance.
4)Understanding multiple indexes with MySQL (and performance).
5)Improve query performance when selecting almost all rows with many “group by” columns.