

OPEN SOURCE DATABASE MANAGEMENT SYSTEMS  
FOR LOW COST SOLUTIONS

by

Md. Zillur Rahman

A Thesis Presented in Partial Fulfillment  
of the Requirements for the Degree  
Master of Science  
in  
Computer Science

INDEPENDENT UNIVERSITY, BANGLADESH

June 2006

OPEN SOURCE DATABASE MANAGEMENT SYSTEMS  
FOR LOW COST SOLUTIONS

by

Md. Zillur Rahman

has been approved

May 2006

APPROVED:

Dr. Indrani Haque \_\_\_\_\_, Chairperson

Professor Dr. M. Abdus Sobhan \_\_\_\_\_, Member

Dr. Khosru Salim \_\_\_\_\_, Member

Dr. Mostofa Akbar \_\_\_\_\_, Member

Supervisory Committee

ACCEPTED:

\_\_\_\_\_  
Director, School of Engineering  
and Computer Science

## ABSTRACT

Implementation of database solutions is a costly technology for Bangladeshi the enterprises. As an alternative of high cost commercial database management system, open source database management system (OSDMS) is studied in this thesis and compared with the commercial database management systems viz. Oracle. Open source database management systems are available in the Internet at free of cost.

Among the available OSDMS, the leading OSDMS: MySQL is selected for comparison and analysis. Simultaneously, the widely used and costly RDBMS Oracle version 9i is selected for comparison. All important features for client/server and distributed solution for small and medium scale enterprises are analyzed and presented in the thesis.

In addition, ODBC connectivity implementation for both Oracle and MySQL are included. As a new concept of open source product, MySQL is still in developing stage. Features absent in MySQL but implemented in Oracle, are also discussed in the thesis.

Finally, a summarized comparison between MySQL and Oracle is presented and a cost effective database solution is recommended for small and medium scale enterprises.

## ACKNOWLEDGEMENTS

At first, the author would like to express the gratitude to the supervisor Dr. Indrani Haque for her ideas, guideline and help during the analysis and implementation phase of the thesis. The research work would never have been possible without her scholarly guidance throughout the period.

The author also would like to express heartiest thanks to Dr. Mostofa Akbar for his brilliant advice for selecting the topics and content of the thesis.

The Independent University, Bangladesh is thanked for giving the precious opportunity to do a postgraduate study and for giving access to their wide range of resources. The author thanked all the committee members, teachers and staff members of the School of Engineering and Computer Science, Independent University Bangladesh for their kind cooperation.

## TABLE OF CONTENTS

	Page
LIST OF TABLES .....	viii
LIST OF FIGURES .....	ix
CHAPTER	
1. INTRODUCTION .....	1
1.1 What is Open Source Database Management System .....	3
1.2 Motivation to use OSDMS .....	5
1.3 Problems of Small and Medium Scale Enterprises in Bangladesh .....	6
1.4 Requirement for the Solution of the Problems .....	7
1.5 Outline of the rest of the thesis .....	8
1.6 Summary .....	10
2. FEATURES REQUIRED FOR ENTERPRISES DATABASE SOLUTIONS	12
2.1 Application Development and Connectivity .....	12
2.2 Standard SQL Support and Client Utilities .....	12
2.3 Referential Integrity .....	13
2.4 Database Security .....	14
2.5 Database Transaction .....	15
2.6 Database Replication .....	16
2.7 Optimization .....	18
2.8 Clustering .....	18
2.9 Partitioning .....	19
2.10 Stored Procedure, Trigger and View .....	20
2.11 Backup & Recovery .....	23

CHAPTER	Page
2.12 Maintainability .....	23
3. FEATURES EXIST IN MYSQL AND COMPARISON WITH ORACLE ...	25
3.1 Application Development and Connectivity features .....	25
3.2 SQL Prompt interface .....	25
3.3 Data Definition Language (DDL) .....	27
3.4 Data Manipulation language .....	32
3.5 Database Constraint .....	35
3.6 Database Connectivity .....	36
3.7 Data Type .....	44
3.8 Oracle and MySQL Security .....	46
3.9 Transaction and Locking.....	60
3.10 Replication .....	63
3.11 Clustering .....	66
3.12 Optimization.....	70
3.13 Partition .....	73
3.14 Stored Procedure .....	76
3.15 Disaster Prevention and Recovery .....	78
3.16 Operating System Compatibility.....	80
3.17 Hardware Requirement, Installation and Maintainability .....	81
3.18 Support Services.....	83
3.19 Summary .....	84
4. FEATURES NOT PRESENT IN MYSQL .....	86
4.1 PL/SQL .....	86
4.2 Cursor .....	87

CHAPTER	Page
4.3 Package .....	89
4.4 Synonym .....	91
4.5 Sequence .....	91
4.6 SQL*Report .....	92
4.7 Summary .....	93
5. IMPLEMENTATION COMPARISON OF ORACLE AND MySQL .....	94
5.1 Oracle RDBMS Implementation.....	94
5.2 MySQL RDBMS Implementation .....	101
5.3 Summary .....	105
6. SUMMARY.....	108
6.1 Selection of OSDMS: MySQL.....	108
6.2 Comparative of the existing features of MySQL with Oracle .....	109
6.3 Conclusion.....	116
REFERENCES .....	120

## LIST OF TABLES

Table	Page
3.1: MySQL Data types mapping with Oracle.....	46
3.2: Grant Table .....	52
3.3: Table and Column Privilege .....	53
3.4: Process Privileges .....	53
3.5: Privileges .....	55
3.6: MySQL supported Operating System.....	81
5.1 : PMS project profile.....	100
5.2 : BBDS project profile .....	104
5.3: Oracle vs MySQL implementation.....	106



## LIST OF FIGURES

Figure	Page
3.1: Oracle SQL*Plus command Prompt Interface.....	25
3.2: MySQL Command Prompt Interface.....	27
3.3: MyODBC Architecture.....	39
3.4: Oracle Cluster .....	67
3.5: MySQL Cluster Architecture.....	69
4.1: SQL and PL/SQL implementation in Database.....	87
4.2: Query Processing.....	88

## CHAPTER 1

### INTRODUCTION

Database Management Systems have been used in Bangladesh for managing information in the field of business, management, education and research over the past 20 years. Initially single user desktop based database management systems were used. Later on, multi-user client/server system has been implemented in a number of private sectors and multinational business organizations, banks, NGOs and some government organizations as well. In the mean time, database system has become an essential tool in corporate sector to process information for planning, monitoring, decision-making and management of the business in Bangladesh.

The corporate users are using various commercial traditional database management systems such as Oracle, SQL server and DB2 etc. for their business planning and management. Along with business organizations, NGOs, banks, educational institutions, research organizations and government organizations of Bangladesh are also using database application as management tools for planning, management and decision-making purpose. In recent years, pharmaceutical companies, cellular mobile companies, garments companies, banks, insurance companies, educational institutions, research organizations are the major users of database application. The companies are spending huge amount of budget for managing their databases due to lack of proper selection without considering their requirement in regards of high cost software, hardware, and manpower. In some cases, this cost is unacceptably high for small and medium sized enterprises of Bangladesh.

As an alternative to the high cost commercial database management systems, Open Source Database Management Systems (OSDMS) may be used for database solution. Due to lack of marketing policy and lack of awareness of the clients, the OSDMSs are not used widely in the world as well as in our country though technically very closely competent to the commercial databases. The idea behind the open source is that product's source and binary codes can be found at free of cost. The Open source database management systems are available in the Internet that can be downloaded. Some of the available OSDMS are;

- MySQL
- Firebird
- PostgreSQL
- SAPDB
- SQLite
- Sleepycat/Berkeley DB
- EnterpriseDB 2005

Among them, MySQL is an unquestioned leader and world's most popular Open Source Database Management System with 3 million installations and 20 thousand downloads per day (Strandell 2003). It is also very close comparable with the widely used commercial database management systems. MySQL is perfect enough for small-scale database (Haff 2005). The license of the OSDMS usually says that the usage is free in non-commercial systems and in product that are published under the same license. The license price for the commercial systems is significantly much more lower than that of the commercial database management systems. The low-cost open source DBMS could be an attractive option for many enterprises, NGOs, GOs,

research organizations, educational institutions if their requirements are fulfilled and offered functionality and performance comparable to those commercial database systems.

At present, a number of renowned organizations in the world are using various OSDMS as per their requirement to reduce their cost. Among them, the ICT service providers and research organizations viz. Alcatel, AOL, Cox Communications, Ericsson, Google, Suzuki, NASA, Cisco and US Census Bureau etc. (MySQL 1995) are on the top of the list. Bangladeshi organizations could take this opportunity to implement the cost free database systems as an alternative to commercial systems if only the required functionalities are supported by the selected systems. The usages of OSDMS can help us to get low cost solutions in terms of low cost hardware, software and mid efficiency level manpower and with legal license which ensures to get support services in the event of any disaster of the database system.

### 1.1 What is Open Source Database Management System

Open source means practices in the production of product that promote access to their sources. It also means to have the right to make copies of the program, distribute those copies, to have access to the software's source code and also the right to make improvements to the program (OSI). The other criteria of open source programs are;

- **Free Redistribution** - The license may not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license may not require a royalty or other fee for such sale.

- **Source Code** - The program must include source code, and must allow distribution of source code as well as compiled form. If a product is not distributed with source code, there must be a well-publicized means of downloading the source code, without charge, via the Internet. The source code must be the preferred form in which a programmer would modify the program. Intermediate forms such as the output of a preprocessor or translator are not allowed.
- **Derived Works** - The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
- **Integrity of the Author's Source Code** - The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time.

The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

- **No Discrimination Against Persons or Groups** - The license must not discriminate against any person or group of persons.
- **No Discrimination Against Fields of Endeavor** - The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

- **Distribution of License** - The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
- **License Must Not Be Specific to a Product** - The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution. L
- **License Must Not Contaminate Other Software** - The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

So like Open Source Programs, Open Source Database Management System fulfills the above criteria in terms of free uses in commercial and non-commercial purposes, free downloading from Internet, modification in binary code and distribution etc.

## 1.2 Motivation to use OSDMS

In free trade economy, business is becoming very competitive in this globalised world. Proper planning and management based on information are very important factors to make the business profitable. As a developing country, Bangladeshi enterprises must depend on information for the purpose of planning, management, monitoring and decision-making. Database Management System is the only tool for storage and processing of information. Most of the small and medium scale

enterprises in our country are not as much as capable to afford high cost database solutions due to budget limitation. At the same time, we need to depend on the technology to make a sustainable business policy with minimum wastage of resources. So, it is very important for us to implement a cost effective technology to the business solutions.

Considering the limitations as well as importance, Open Source Database Management Systems can give us a cost cutting solution for development of business in the country.

### **1.3 Problems of Small and Medium Scale Enterprises in Bangladesh**

In Bangladesh, small/medium scale enterprises have handled basically operational and management information for planning and managing their businesses. Operational information is very important for development of the business. It varies from enterprise to enterprise as per the business type. As for example, production oriented enterprise needs production monitoring and stock maintenance, inventory management and marketing management system. Similarly trading company need stock maintenance, inventory and marketing management systems for maintaining the business operational information. The possible systems for business operation are;

- Production Monitoring Systems
- Product Quality Control Management System
- Warehouse Automation Systems
- Stock Maintenance System
- Inventory Management System
- Sales Information Management System

- Credit Management Systems and so on.

Similarly the Govt., NGO and National and International agencies and Business companies need management oriented information systems for better running of the organization. Some of the possible information systems are;

- Personal Management Info. System
- Payroll System
- Accounts Systems
- Human Resource Management Systems
- Inventory for Asset Management Systems etc.

#### **1.4 Requirement for the Solution of the Problems**

The basic requirements of user as well as technical point of view for Database Application System of an enterprise solution are;

##### **User point of view**

- Standard SQL supported Database Management System
- Connectivity with standard front-end language like VB, C, C++, Java and .NET languages
- Support Report generator
- Client/server facilities
- User authentication/Security
- Data conversion facility

##### **Technical point of view**

- Data storage capability



- Security
- Reliability
- Scalability
- Maintainability
- Transaction Management
- Concurrency Control
- Replication
- Backup and recovery scheme
- Easy administration
- Performance tuning etc.

## **1.5 Outline of the rest of the thesis**

Chapter wise outline for the rest of the thesis is explained below;

### **1.5.1 Chapter 2: Features required for real life enterprise database solutions**

All database features required for a solution are explained under the chapter 2. The features are listed below;

- Application development and connectivity
- Standard SQL support and client utilities
- Referential Integrity
- Database Security
- Database Transaction
- Database Replication
- Optimization

- Clustering
- Partitioning
- Stored procedure, trigger and view
- Backup and recovery
- Maintainability

### 1.5.2 Chapter 3: Features exist in MySQL and comparison with Oracle

Comparison of the existing features of MySQL with Oracle is explained in the Chapter 3. The features are listed below;

- Application Development and Connectivity features
- SQL Prompt interface
- Data Definition Language
- Data Manipulation language
- Database Constraint
- Database Connectivity
- Column Types
- Security
- Database Integrity
- Transaction and locking
- Replication
- Clustering
- Optimization
- Partitioning
- Stored Procedure, trigger and views

- Disaster Prevention and Recovery
- Operating system compatibility
- Hardware requirement, installation and maintainability
- Support services

### **1.5.3 Chapter 4: Features Not Present in MySQL**

The non-existence features of MySQL are discussed in this chapter. The advantages of these features in database solution are also covered.

### **1.5.4 Chapter 5: Implementation comparison with Oracle and MySQL**

A comparison analysis of two database systems developed with Oracle and MySQL respectively are presented in this chapter. It covers implementation details, scope of the systems, requirement specification, security implementation of Oracle and MySQL, costing, threats and status etc.

### **1.5.5 Chapter 6: Conclusion**

This chapter includes the summary of the comparative analysis of the Oracle and MySQL features and conclusion and recommendation of the study.

## **1.6 Summary**

Study proposal, objective, goal and importance of the thesis are explained in this chapter. Database requirement analysis, problems and solutions for small and medium scale enterprises of Bangladesh are presented. A list of available open source database

management systems for low cost solutions are also presented in this chapter of the thesis.

## **CHAPTER 2**

### **FEATURES REQUIRED FOR ENTERPRISES DATABASE SOLUTIONS**

All Database Applications need some basic features to fulfill the requirement of the solutions. Some of these basic features are explained below;

#### **2.1 Application Development and Connectivity**

Support for application development is the basic part of a Database Management Systems. Various front-end languages are used as application development tools that are integrated with the backend database part. The front-end language is generally used for both data insertion and data retrieval. In some cases, specialized tool is also available for data extraction, which is known as report generator. Few of the standard report generators supported by all common DBMSs are MS Data Reporter and Crystal Report etc. The application programming language is selected as per requirement of the company based on existing resources such as human, hardware, budget, timeline etc. DBMS must have connectivity support to link with the front end tools using the connectivity techniques DAO, ADO, ADO.NET, OLEDB, ODBC or JDBC etc.

#### **2.2 Standard SQL Support and Client Utilities**

Standard SQL compatibility and client utility programs are very important for a Database Management System to connect to the server to access databases and to

perform administrative tasks. In addition, other client utilities are also required to perform database operation related activities.

### 2.3 Referential Integrity

In relational database systems, referential integrity is specified between two relations and is used to maintain the consistency among the tuples of the two relations (Batini et al. 1992). Referential integrity is a system of rules that database uses to ensure that relationships between records in related tables are valid, and that the related data may not be deleted or changed accidentally. It is a relationship between primary and foreign key of two tables in a database. Referential integrity can enforce the following rules in the Database;

- Data cannot be entered in the foreign key field of the related table that doesn't exist in the primary key of the primary table.
- Records from a primary table cannot be deleted if matching records exist in a related table. For example, we can't delete an employee record from the Employees table if there are orders assigned to the employee in the Orders table.
- Primary key value cannot be changed in the primary table, if that record has related rows in the child tables. For example, we can't change an employee's ID in the Employees table if there are orders assigned to that employee in the Orders table.

Referential integrity also includes the techniques known as cascading update and cascading delete, which ensure that changes made to the linked table are reflected in

the primary table. Referential integrity with cascading options enforces the following three rules;

- If the primary key for a record in the employee table (parent table) changes, all corresponding records in the order table (child table) must be modified using a cascading update.
- If a record in the employee table is deleted, all corresponding records in the order table must be deleted using a cascading delete.

## 2.4 Database Security

Database security is very important issue in Database Management Systems to protect the data from unauthorized users. Security features can prevent the database from any destructive operation on the database. The common commercial Relational Database Management Systems have incorporated various levels of security features such as user and administrative level security etc. User authorization can prevent the unauthorized users to access the database whether it is network or single user desktop based application. Similarly, some security features can prevent from any unwanted operation in the fields. Among them, user authentication with various level of privileges, data encryption, table label security, row/record level security are the common features for the common commercial Databases Management Systems. Security is much more important in the Internet and network based distributed database application systems.

## 2.5 Database Transaction

Database transaction is a unit of interaction with a database management system or similar system that is treated in a coherent and reliable way independent of other transactions that must be either entirely completed or aborted. The ability to handle transactions allows the user to ensure that integrity of a database is maintained (Pinouts 2005). A single transaction might require several queries, each reading and/or writing information in the database. When this happens it is usually important to be sure that the database is not left with only some of the queries carried out. For example, when doing a money transfer, if the money was debited from one account, it is important that it also be credited to the depositing account. Also, transactions should not interfere with each other.

Ideally, a database system will guarantee the integrity of the data with the following ACID properties of transactions (Silberschatz *et al.* 1997);

**Atomicity** - All or none, either every part of a transaction gets completed, or none of it does at all. If everything works correctly without any errors, everything gets committed to the database. If any one part of the transaction fails, the entire transaction gets rolled back.

**Consistency** -this property ensures that the state of the database is the same when it ends as it was when it began, and that no rules or constraints of the database are violated as a result of the transaction. Any violation of rules or constraints will cause the transaction to fail.

**Isolation** -The operations that take place within a transaction are isolated from the operations of other transactions or other database operations, and that no other



transaction or operation can have access in any way to the data that's being affected by a transaction in its intermediate state (while the transaction is taking place).

**Durability** - the guarantee that once a transaction has completed, the results of it will persist and remain accurate and stable within the database properties for each transaction. In practice, these properties are often relaxed somewhat to provide better performance.

A simple transaction is usually issued to the database system in a language like SQL in the form:

- Begin the transaction
- Execute several queries (although any updates to the database aren't actually visible to the outside world yet)
- Commit the transaction (updates become visible if the transaction is successful)

If one of the queries fails the database system may rollback either the entire transaction or just the failed query. This behaviour is dependent on the RDBMS in use and how it is set up. The transaction can also be rolled back manually at any time before the commit. Transactions are also called **LUWs** (Logical Units of Work) in some systems. So the transaction is very important property for a database management system to maintain security and quality of data.

## 2.6 Database Replication

Database replication, the copying and maintenance of data on multiple servers, has come of age. It is a very important feature in distributed database system. Changes of

data in one location is checked with the other locations and updated by the replication features (Oracle 1999).

Every major database vendor has a replication solution of one kind or another, and many number of database vendors also offers alternative methods for replicating data. A few years ago, database replication was a high-end capability that few companies considered to implement. Now, more businesses are interested in database replication for two main reasons: Businesses are becoming more geographically dispersed yet their employees still need access to a single set of coherent data, and many companies work with massive datasets that they can not quickly restore in the event of a system failure.

Database replication is different from file replication, which essentially copies files. Database replication products log selected database transactions to a set of internal replication-management tables. The software then periodically checks these tables for updated data and moves the data from the source to the target systems while guaranteeing data coherency and consistency.

Database replication helps to handle many of the problems encountered with distributed systems. Replicating to databases systems in branch offices lets branch-office users access a local copy of the data instead of accessing a central server over WAN links. Database replication products also let us transfer selected data sets to a reporting server so that we can move our processor intensive reporting processes of our main transactional database.

Database replication can also supplement disaster recovery plans by duplicating the data from a local database server to a remote database server. If the primary server fails, the applications can switch to the replicated copy of the data and continue operations. Many database replication products even have built-in tools to update the primary database with any changes that users made to the backup database while the primary database was offline.

## **2.7 Optimization**

Optimization is very important features to make the database faster. Different database has different features for optimization of its speed. The main thing is to identify the bottleneck of the system. According to its bottleneck, the system should be optimized. So, the database management systems must have supported optimization techniques to make faster operation of the database system.

## **2.8 Clustering**

Database clustering is a technology that let us a group of interconnected computers work together to serve a single database. The idea is that together they can create a fault-tolerant, high-performance scalable solution that's a low-cost alternative to high-end servers-though, so far, that claim has yet to be proven.

In mid-1990s, the major relational database vendors began experimenting with clustering as a way to make their products more efficient and fault-tolerant. Two common methods are used in the clustering technologies of the relational databases; "shared-disk" and "shared-nothing".

In shared-nothing architecture, each node in the cluster holds only one segment of the database, and the node also handles all of the computational work that corresponds to the data it stores. A master server assesses the task at hand and then parcels it out, distributing a portion of the job to each node that contains data to be processed. The task is then executed by all of the nodes in parallel, and the master server reports the result.

On the other hand, shared-disk clustering is a design that's structured around a single large data store (for example, a disk array). Each node on the cluster has equal access to all of the information in the data store. Only the processing work is divided amongst them, and not the data itself. The result is a particularly fault tolerant database. Even if one or more servers fail, all of the application data remains available to the other nodes. By comparison, if one node on a shared-nothing database crashes, all of the data stored on that node likewise goes offline, until a failover system can recover from the fault.

So, the clustering technology is very important feature to make the database systems fault tolerant.

## **2.9 Partitioning**

Very large Database (VLDB) is a rapidly growing trend in the enterprise world. The supporting time of the Database vendors is increased greatly for the client of the large database user. From this experience, the companies have developed a feature to make the large amount of data manageable by splitting into a smaller size, which is known as partition. Partitioning of large database i.e. table gives us the following advantages;

- Reduce database downtime for maintenance
- Reduce database downtime for media failure
- Improve query performance
- Better distribution of disk space
- Partition transparency

### 2.10 Stored Procedure, Trigger and View

Stored procedure is a program (or procedure), which is physically stored within the database. They are usually written in a proprietary database language like Transact-SQL for Microsoft SQL Server, PL/SQL for Oracle database, or PL/pgSQL for PostgreSQL (Silberschatz *et al.* 1997). The advantage of a stored procedure is that when it is run in response to a user request, it is run directly by the database engine, which usually runs on a separate database server and is generally faster at processing database requests. The database server has direct access to the data needs to manipulate and only needs to send the final results back to the user, doing away with the overhead of communicating potentially large amounts of interim data back and forth.

Typical uses for stored procedures include data validation which is integrated into the database structure (stored procedures used for this purpose are often called triggers), or encapsulating an API for some large or complex processing (such as manipulating a large dataset to produce a summarized result). A stored procedure that runs a (possibly complex) series of queries will often run faster as a stored procedure than if it had been implemented as, for example, a program running on a client computer which communicates with the database by submitting the SQL queries one by one,

receiving results from them, and then deciding what other queries may need to be run. By having the logic run inside the database engine, this eliminates numerous context switches and a great deal of network traffic.

Stored procedures can also simplify data management when a database is manipulated from many external programs. Embedding "business logic" in the database using stored procedures eliminates the need to duplicate the same logic in each of the programs which accesses the data. This simplifies the creation and maintenance of the programs involved, and can help avoid the data corruption that can be introduced if one or another of those external systems fails to have the validation logic upgraded properly. In some systems, stored procedures can be used to control transaction management; in others, stored procedures run inside a transaction such that transactions are effectively invisible to them.

**Database triggers** are procedures that are stored in a database and are executed automatically by the system as a side effect of a modification to the database (Silberschatz *et al.* 1997). They are very powerful tools that can be used to perform many tasks such as restricting access to specific data, perform logging, or auditing of data sets.

There are two classes of triggers; they are either "row triggers" or "statement triggers". With row triggers we can define an action for every row of a table, while statement triggers only occur once and are not dependent on the shape of the data. Each class can be of several types. There are "BEFORE triggers" and "AFTER triggers" which alter the time of execution of the trigger. There is also an "INSTEAD

OF trigger" which is a conditional trigger that will fire instead of the triggering statement.

There are typically three triggering EVENTS that cause trigger to 'fire':

- INSERT event (as a new record is being inserted into the database).
- UPDATE event (as a record is being changed).
- DELETE event (as a record is being deleted).

Databases that support triggers typically give programmers access to record variables by means of a syntax such as :OLD.cust\_name or :NEW.cust\_name. e.g. if a trigger is monitoring for changes to a salary field one could write a trigger.

```

BEFORE UPDATE ON employee_table
FOR ALL records
  IF :NEW.salary <> :OLD.salary THEN
    do something here
  END IF;
END;
```

Database triggers enable DBA's (Database Administrators) to create additional relationships between separate databases. For example, the modification of a record in one database could trigger the modification of a record in the second database.

A **view** is a read only virtual or logical table composed of the result set of a query (Silberschatz *et al.* 1997). Unlike ordinary tables in a relational database, a view is not part of the physical schema: it is a dynamic, virtual table computed or collated from data in the database. Changing the data in a table alters the data shown in the view.

Views can provide advantages over tables;

- They can create subset the data contained in a table
- They can join and simplify multiple tables into a single virtual view
- Views can act as aggregated tables, where aggregated data (sum, average etc.) are calculated and presented as part of the data
- Views can hide the complexity of data, for example a view could appear as Sales2000 or Sales2001, transparently partitioning the actual underlying table
- Views do not incur any extra storage overhead
- Dependant on the SQL engine uses views can provide extra security.

### **2.11 Backup & Recovery**

Backup and recovery is very important feature for preventing the database from any event of software or hardware failure. These features will ensure us that the loss of disk drives or deletion of files does not cause permanent loss of data. The backup is a way of achieving data and recovery is to perform restore up-to-the-minute data if losses. It also ensures to achieve and restore the control and log files into the database in case of disaster.

So backup and recovery is very important features for a corporate database solution.

### **2.12 Maintainability**

Maintainability is prerequisites for any type ICT solutions. It is directly related to the cost of the solutions. Easy maintainability of a system has greater possibility to sustain. Harder maintenance systems need more resources.



So maintenance facility is very important factor for a system to make it smooth running.

## CHAPTER 3

### FEATURES EXIST IN MYSQL AND COMPARISON WITH ORACLE

#### 3.1 Application Development and Connectivity features

Application development and connectivity features are the basic part of the Database Management System to develop a solution. All RDBMS have their own features with different formation. A comparison analysis of Commercial Database Management System i.e. Oracle and an Open Source Database Management System viz. MySQL is given below;

#### 3.2 SQL Prompt interface

##### Oracle:

Oracle has a command prompt interface for both client and administrator to connect to the Oracle databases. User name and password are needed to get the command prompt. The command prompt interface is shown below;

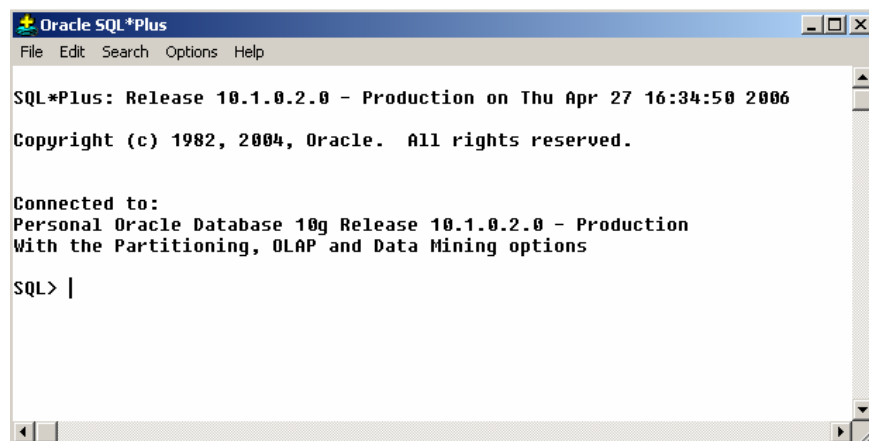


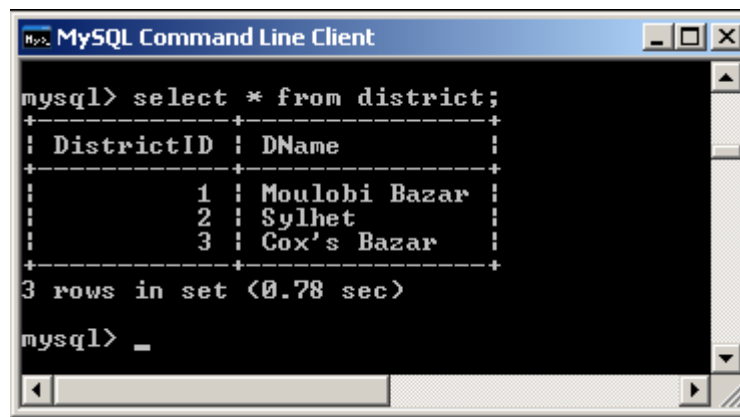
Figure 3.1: Oracle SQL\*Plus command Prompt Interface

### MySQL:

MySQL DBMS has also command prompt interface like other commercial database system to connect the Databases and run SQL statements. Using Windows/DOS command prompt mode, MySQL command along with host, user and password parameter can be passed to connect to the database server. The command is given below;

```
C:\MySQL\Bin\> mysql -h host -u user -p
Enter password: *****
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 4.0.14-
log
Type 'help;' or '\h' for help. Type '\c' to clear the
buffer.
mysql>
```

In the above MySQL command prompt interface, any SQL statement can be run to operate the database as per level of user privileges. Both administrator and user level statement can be run into the command line client interface window. The command line interface exists in both the client and server machine. One example for server connectivity and SQL statement output is shown in the next page;



```

mysql> select * from district;
+-----+-----+
| DistrictID | DName      |
+-----+-----+
| 1          | Moulobi Bazar |
| 2          | Sylhet      |
| 3          | Cox's Bazar  |
+-----+-----+
3 rows in set (0.78 sec)

mysql> _

```

**Figure 3.2: MySQL Command Prompt Interface**

### 3.3 Data Definition Language (DDL)

All SQL supported Database Management Systems support data definition language for creating and managing the databases. The basic syntax of the DDL statements is almost identical in all RDBMS. Some parameters may be different for some statement. Though there is some minor difference from one to another, but it fulfills the requirement for the small and medium scale enterprise database solutions. Some of the common Data Definition Language statements are shown below;

- Create Database Syntax
- Alter Database Syntax
- Drop Database Syntax
- Create Table Syntax
- Alter Table Syntax
- Drop Table Syntax
- Rename Table Syntax
- Create Index Syntax
- Drop Index Syntax

Details descriptions of the statements are given below;

### 3.3.1 CREATE DATABASE Syntax

CREATE DATABASE statement is used to create a database with the given name. It needs the CREATE privilege on the database to run the statement. The syntax of the statement for oracle and MySQL is shown below;

#### Oracle:

Oracle database can be created in several ways. Manually executing the CREATE DATABASE statement is one of the ways to create a database. CREATE DATABASE syntax needs several clauses such as user authentication, log files location, size, data file location, data file size, table space size etc. The syntax is;

```
CREATE DATABASE mynewdb
    USER SYS IDENTIFIED BY pz6r58
    USER SYSTEM IDENTIFIED BY y1tz5p
    LOGFILE GROUP 1
    ('/u01/oracle/oradata/mynewdb/redo01.log') SIZE
    100M,
    GROUP 2
    ('/u01/oracle/oradata/mynewdb/redo02.log') SIZE
    100M,
    GROUP 3
    ('/u01/oracle/oradata/mynewdb/redo03.log') SIZE
    100M
    MAXLOGFILES 5
    MAXLOGMEMBERS 5
    MAXLOGHISTORY 1
    MAXDATAFILES 100
    MAXINSTANCES 1
```

```

CHARACTER SET US7ASCII
NATIONAL CHARACTER SET AL16UTF16
DATAFILE
'/u01/oracle/oradata/mynewdb/system01.dbf' SIZE 325M
REUSE
EXTENT MANAGEMENT LOCAL
SYSAUX DATAFILE
'/u01/oracle/oradata/mynewdb/sysaux01.dbf' SIZE 325M
REUSE
DEFAULT TABLESPACE tbs_1
DEFAULT TEMPORARY TABLESPACE tempts1
TEMPFILE
'/u01/oracle/oradata/mynewdb/temp01.dbf'
SIZE 20M REUSE
UNDO TABLESPACE undotbs
DATAFILE
'/u01/oracle/oradata/mynewdb/undotbs01.dbf'
SIZE 200M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED;

```

### **MySQL Syntax:**

```

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
    [create_specification [, create_specification]
... ]

```

```

create_specification:
    [DEFAULT] CHARACTER SET charset_name
    | [DEFAULT] COLLATE collation_name

```

It appears that the purpose of the statement is same in both Oracle and MySQL database management Systems. The operation is comparatively easier in the MySQL database server.

### 3.3.2 ALTER DATABASE Syntax

Both Oracle and MySQL support ALTER DATABASE command. It is used to change the overall characteristics of a database. ALTER privilege is needed on the database to use the ALTER DATABASE command for security purposes in both systems. The syntax of the statement is almost same in all RDBMS except some key words (refman5.0 2006, Sheppard 2006).

### 3.3.3 DROP DATABASE Syntax

DROP DATABASE command deletes the database along with tables, views and other objects of the database. DROP privilege is needed to execute the command.

Oracle does not support the DROP DATABASE command. The MySQL syntax of the statement is shown below;

```
DROP {DATABASE | SCHEMA} [IF EXISTS] db_name
```

### 3.3.4 CREATE TABLE Syntax

CREATE TABLE command is a standard SQL statement to create a table with the given name with in the active database. All database management systems including Oracle and MySQL support the CREATE TABLE command. The Syntax of the command is also identical except some keywords, which may be seen in the manuals (refman5.0 2005, Sheppard 2006). Both Oracle and MySQL need required privileges to execute the command.

### 3.3.5 ALTER TABLE Syntax

ALTER TABLE statement allows to change the structure of an existing table. For example, we can add, modify or delete columns, create or destroy indexes, change the type of existing columns, or rename columns or the table itself. Both Oracle and MySQL support the ALTER TABLE statement. The basic syntax is almost same in all corporate database management system. Some keyword may differ from one RDBMS to another RDBMS. Basic functionality and the syntax of the ALTER DATABASE statement are same in both Oracle and MySQL (Refman5.0 2006, Sheppard 2006).

### 3.3.6 DROP TABLE Syntax

DROP TABLE command removes the table and the associated objects of the table. The syntax and functionality are exactly identical in both oracle and MySQL database management system (Sheppard 2006, Refman5.0 2006).

### 3.3.7 RENAME TABLE Syntax

This statement renames old name to new name. Both Oracle and MySQL support rename command but there is a difference between the two RDBMS. Oracle rename one table name at a time but MySQL rename multiple table name. The basic part of the syntax is almost identical in both Oracle and MySQL (Sheppard 2006, Refman5.0 2006). The rename operation is done atomically, which means that no other thread can access any of the tables while the rename is running.



### 3.3.8 CREATE INDEX Syntax

Index is an optional object that is the primary method of providing faster access to data. Oracle and MySQL both support CREATE INDEX command. Though Oracle has eight types of indexes and MySQL has single type, the basic part of the syntax is almost identical (Sheppard 2006, Refman5.0 2006). For large databases such as data warehouses, the Oracle index provides faster data retrieval. On the other hand, MySQL index is faster in small/medium type database as well as easy to implement.

### 3.3.9 DROP INDEX Syntax

DROP INDEX command deletes the index named *index\_name* from the table *tbl\_name*. Oracle and MySQL both have this command. The syntaxes are slightly differ from one to another (Sheppard 2006, Refman5.0 2006).

## 3.4 Data Manipulation language

Data Manipulation language is very important part for the client end user to insert, update, modify and relation of data. Like Oracle and MySQL RDBMS has full supported SQL statements to manipulate data. The syntax of the statements is almost same in all relational database management systems including commercial (Oracle) and Open Source Database Management Systems (MySQL). The individual discussion on the DML statements is given below;

### 3.4.1 INSERT Syntax

The INSERT statement is used to insert new rows into an existing table. The user authentication privilege is applied to execute the command. Though the syntax of the

statement is almost same in Oracle, MySQL (Sheppard 2006, Refman5.0 2006) and other RDBMS but every RDBMS has some strong features. MySQL allows INSERT with a VALUES clause to specify multiple sets of data to insert whereas Oracle and others cannot support multiple set of data.

### **3.4.2 UPDATE Syntax**

Both Oracle and MySQL have UPDATE statements to change existing values in one or more tables to new values. Unlike Oracle, MySQL UPDATE statement allows to update records in multiple tables and limit the number of rows to update using ORDER BY clause.

### **3.4.3 DELETE Syntax**

DELETE statement removes one or more rows from one or more tables. Oracle allows to remove records from one table but MySQL allows to delete records from multiple tables along with limiting rows. The basic part of the syntax is identical in both Oracle and MySQL (Sheppard 2006, Refman5.0 2006).

### **3.4.4 SELECT Syntax**

SELECT statement is virtually identical in Oracle and MySQL. The syntax is also identical (Sheppard 2006, Refman5.0 2006). It is used to retrieve rows selected from one or more tables. Oracle supports UNION, MINUS and INTERSECT operation with SELECT statement.

On the other hand MySQL only supports UNION statement and the subqueries with keywords ALL, ANY, SOME, EXISTS, NOT EXISTS but requirements of MINUS and INTERSECT keywords can be done in a customized way.

### 3.4.5 Sub-query Statement

Like Oracle, MySQL version 5.0 supports subquery in SELECT statement that is nested within another SELECT statement which returns intermediate results. It supports all SQL standard subqueries along with all standard keywords and comparison operators. One example of the subquery statement in MySQL is as follows;

```
SELECT * FROM t1 WHERE column1 = (SELECT column1 FROM t2);
```

MySQL supported keywords and operators are shown below;

- = > < >= <= <>
- IN, ANY, SOME and ALL
- EXISTS and NOT EXISTS

### 3.4.6 Outer Joins

Outer joins allow for a join predicate in WHERE clause to be satisfied even if the value of the join column in one of the tables does not exist. There are three types of outer joins: left, right and full.

Oracle supports all types of outer joins but MySQL supports only two; left and right join. The purpose of full join can be met using UNION keyword with left and right output.

### 3.4.7 TRUNCATE TABLE Syntax

TRUNCATE TABLE statement empties a table completely. Logically, this is equivalent to a DELETE statement that deletes all rows, but there are practical differences under some circumstances. It differs from DELETE FROM in the following ways;

- Truncate operations drop and re-create the table, which is much faster than deleting rows one by one.
- Truncate operations are not transaction-safe; error generated in case of active transaction or an active table lock.
- The number of deleted rows is not returned.
- As long as the table definition file *tbl\_name.frm* is valid, the table can be re-created as an empty table with TRUNCATE TABLE, even if the data or index files have become corrupted.

The syntax is same in both Oracle and MySQL which is shown below;

```
TRUNCATE TABLE tbl_name
```

### 3.5 Database Constraint

Both Oracle and MySQL support integrity constraints to keep the data unaltered from any unwanted operations. Since different type of tables transitional and non transitional exists in MySQL, it allows integrity constraint slightly different ways enforcing certain rules either with in a table or between tables. Basically constraints are used;

- to enforce rules at table level whenever a row is inserted, updated and deleted from that table.
- to prevent deletion of a table if there are dependencies from other tables

Like the commercial database management systems such as Oracle, the following constraints exist in MySQL database management system;

- NULL/NOT NULL keyword
- PRIMARY KEY
- UNIQUE key
- CHECK
- FOREIGN KEY

Unlike Oracle, MySQL does not support CASCADE option with referential integrity on update and delete operation.

### **3.6 Database Connectivity**

#### **3.6.1 Client Connectivity**

Oracle has NET8 (called SQL\*NET prior to Oracle8) Oracle's client/server middleware product that offers transparent connection from client tools to the database, or from one database to another. SQL\*Net/ Net8 works across multiple network protocols and operating systems (Gennick *et al.* 2002).

Like Oracle, MySQL has a client programs to access a MySQL server with the following parameters;

- The name of the host where the MySQL server is running

- Username
- Password

For example, the MySQL client can be started as follows from a command-line prompt:

```
shell> mysql -h host_name -u user_name -ppassword
```

Alternate forms of the -h, -u, and -p options are --host=*host\_name*, --user=*user\_name*, and --password=*password*.

MySQL client programs use default values for any connection parameter option that need not to specify;

- The default hostname is localhost.
- The default username is ODBC on Windows and your Unix login name on Unix.
- No password is supplied if -p is missing.

Thus, for a Unix user with a login name of joe, all of the following commands are equivalent:

```
shell> mysql -h localhost -u joe
shell> mysql -h localhost
shell> mysql -u joe
shell> mysql
```

Other MySQL clients behave similarly. We can specify different default values to be used when we make a connection so that we need not enter them on the command line each time invoking a client program. This can be done in a couple of ways:

- Specifying connection parameters in the [client] section of an option file.

The relevant section of the file might look like this:

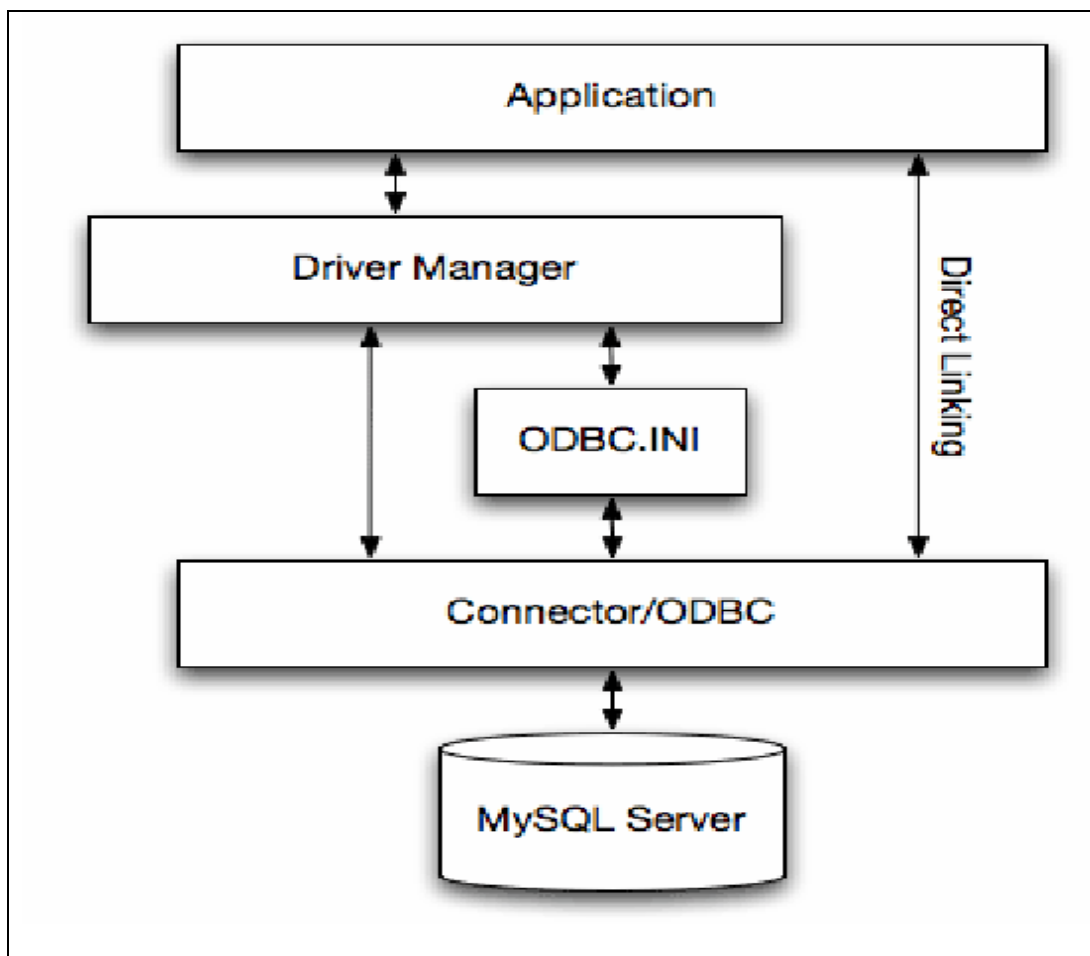
- [client]
- *host=host\_name*
- *user=user\_name*
- *password=your\_pass*
- Specifying some connection parameters using environment variables. The host can be specified for **mysql** using `MYSQL_HOST`. The MySQL username can be specified using `USER` (this is for Windows and NetWare only). The password can be specified using `MYSQL_PWD`.

### 3.6.2 ODBC Connectivity

Oracle supports OLEDB and ODBC connectivity to connect the database from any front-end programming environment under the Windows Operating system. MySQL has also designed its own ODBC connectivity to connect to the MySQL database server from different front-end application environment which is known as MyODBC connector. All types of front end programming language such as VB, C, C++ can connect MySQL server through MyODBC connector. The configuration and connectivity parameters are identical in both Oracle ODBC and MySQL's MyODBC connector. MyODBC can be used on all major platforms supported by MySQL such as;

- Windows 95, 98, Me, NT, 2000, XP, and 2003
- All Unix Operating Systems including Linux, Mac OS X, OS/2, Solaris, SunOS, SCO OpenServer etc.

The architecture (Refman5.0, 2006) of MyODBC is shown as:



**Figure 3.3: MyODBC Architecture**

MyODBC is a powerful and stable connectivity tools which is tested with different applications. The list of the tested application is given below;

- MS Access 95, 97, 2000, and 2002
- C++-Builder, Borland Builder 4
- Centura Team Developer (formerly Gupta SQL/Windows)
- ColdFusion (on Solaris and NT with service pack 5)
- Crystal Reports
- DataJunction
- Delphi



- ERwin
- MS Excel
- iHTML
- FileMaker Pro
- FoxPro
- Notes 4.5/4.6
- MS Visio Enterprise 2000
- Vision
- Visual Objects
- Visual Interdev
- SPSS
- Perl DBD-ODBC
- Paradox
- Powerbuilder
- Powerdesigner 32-bit
- MS Visual C++
- Visual Basic
- ODBC.NET through CSharp(C#), VB and C++
- Data Architect(<http://thekompany.com/products/dataarchitect/>)
- SQLExpress for Xbase++(<http://www.SQLExpress.net>)
- Open Office (<http://www.openoffice.org>)
- Star Office (<http://wwws.sun.com/software/star/staroffice/6.0/index.html>)
- G2-ODBC bridge (<http://www.gensym.com>)
- Sambar Server (<http://www.sambarserver.info>)

Presently, two versions of MyODBC drivers are available;

- MyODBC 2.50 - 32-bit ODBC driver
- MyODBC 3.51 - 32-bit ODBC driver enhanced from MyODBC 2.50

### 3.6.3 Connectivity for .NET environment

Oracle supports three connectivity tools ODBC.NET, OLEDB.NET and ODP.NET to connect the Oracle database under .NET environment. All of the technologies have some advantages and disadvantages. Among them, ODP.NET is most native to the .NET environment, bypassing the need for OLE DB or ODBC. It features high performance access to the Oracle Database, while providing accessing to advanced functionality which is not provided by OLEDB.NET or ODBC.NET. Similar to OLEDB.NET, ODBC.NET and ODP.NET can be used from any .NET language.

Similarly, MySQL Connector/.NET enables developers to easily create .NET applications that require secure, high-performance data connectivity with MySQL. It implements the required ADO.NET interfaces and integrates into ADO.NET aware tools. Developers can build applications using their choice of .NET languages; VB.NET, C++.NET and C#.NET etc. MySQL Connector/.NET is a fully managed ADO.NET driver written in pure C# language.

MySQL Connector/.NET includes full support for;

- Large-packet support for sending and receiving rows and BLOBs up to 2 gigabytes in size.

- Protocol compression which allows for compressing the data stream between the client and server.
- Support for connecting using TCP/IP sockets, named pipes, or shared memory on Windows.
- Support for connecting using TCP/IP sockets or Unix sockets on Unix.
- Support for the Open Source Mono framework developed by Novell.

### **Connector/NET Architecture**

MySQL Connector/NET comprises several classes that are used to connect to the database, execute queries and statements, and manage query results.

The following are the major classes of MySQL Connector/NET:

- **MySqlCommand:** Represents an SQL statement to execute against a MySQL database.
- **MySqlCommandBuilder:** Automatically generates single-table commands used to reconcile changes made to a DataSet with the associated MySQL database.
- **MySqlConnection:** Represents an open connection to a MySQL Server database.
- **MySqlDataAdapter:** Represents a set of data commands and a database connection that are used to fill a dataset and update a MySQL database.
- **MySqlDataReader:** Provides a means of reading a forward-only stream of rows from a MySQL database.

- **MySQLException:** The exception that is thrown when MySQL returns an error.
- **MySqlHelper:** Helper class that makes it easier to work with the provider.
- **MySqlTransaction:** Represents an SQL transaction to be made in a MySQL database.

#### 3.6.4 MySQL Connector/J and JDBC Connectivity

Oracle supports JDBC connectivity to access the Oracle database from Java environment. MySQL also provides connectivity for client applications developed in the Java programming language via a JDBC driver, which is called MySQL Connector/J. MySQL Connector/J is a JDBC-3.0 “Type 4” driver, which means that is pure Java, implements version 3.0 of the JDBC specification, and communicates directly with the MySQL server using the MySQL protocol.

Besides Oracle generic connectivity ODBC, OLEDB etc., Oracle has Transparent Gateways accesses technologies to connect the non Oracle databases from Oracle environment using their native interface. However, it is observed that Oracle has versatile connectivity tools to access database from native as well as different environment. On the other hand, MySQL has a number of tools to connect the database server. Among them, MyODBC is one of the easy configurable, secured and fastest technologies to communicate with MySQL database.

### 3.7 Data Type

Every RDBMS assigned data types for each column of the table. These data types may be native to the Database Management System or custom. Oracle database supports native data type belonging to one of six categories; Character, Number, Date, Large Object (LOB), Row and Row Identifier. Where as MySQL supports primitive data types; String, Numeric, Date and Time types. The following comparison table (OMWM 2002) is showing mapping between MySQL and Oracle data types.

#### Numeric Types

MySQL	Size	Meaning	Oracle Equivalent
TINYINT	1 Byte	Very small integer	NUMBER(3,0)
BOOL, BOOLEAN	1 Byte	0 to True and 1 for false	BOOLEAN
SMALLINT	2 Bytes	Small integer i.e boolean	NUMBER(5,0)
MEDIUMINT	3 Bytes	Medium-sized integer	NUMBER (7,0)
INT	4 Bytes	Standard integer	NUMBER (10,0)
INTEGER	4 Bytes	Large integer	NUMBER (10,0)
BIGINT	8 Bytes	Very large integer	NUMBER (19,0)
FLOAT(X<=24)	4 Bytes	Single-precision floating point number	FLOAT(0)
FLOAT(25<=X<=53)	8 Bytes	Single-precision floating point number	FLOAT(24)
DOUBLE	8 Bytes	Double-precision floating point number	FLOAT(24)
DOUBLE PRECISION	8 Bytes	Double-precision floating point number	FLOAT(24)
REAL	8 Bytes	Double-precision floating point number	FLOAT(24)
DECIMAL	M Bytes(D+2, if M<D)	Fixed point number	FLOAT(24)
NUMERIC	M Bytes(D+2, if M<D)	Numeric value	NUMBER
BIT(M)	M, 0 to 2 <sup>m</sup> -1	A bit field type (1 to 64)	NUMBER(2,0)

### Date and Time Types

MySQL	Size	Meaning	Oracle
DATE	3 Bytes	Date value in "CCYY-MM-DD" format	DATE
DATETIME	8 Bytes	Combination of date and time	DATE
TIMESTAMP	4 Bytes	Timestamp value in "CCYY-MM-DD hh:mm:ss" format	NUMBER
TIME	3 Bytes	Time value "hh:mm:ss" format	DATE
YEAR	1 Byte	Year value "CCYY" format	NUMBER

### String Types

MySQL	Size	Meaning	Oracle
CHAR(m)	M Bytes, 1<=M<=255	Fixed length non-binary string	CHAR
VARCHAR(m)	L+1 Bytes; L<=M and 1<=M<=255	Variable length non-binary string	VARCHAR2
TINYBLOB	L + 1 Bytes; L<2 <sup>8</sup>	Very small binary large object i.e. image and sound file	RAW, BLOB
BLOB	L + 2 Bytes; L<2 <sup>16</sup>	Small BLOB i.e. small image and sound files etc.	RAW, BLOB
MEDIUMBLOB	L + 3 Bytes; L < 2 <sup>24</sup>	Medium sized binary large object file	RAW, BLOB
LOB	L + 4 Bytes; L < 2 <sup>32</sup>	Large sized binary large object file	RAW, BLOB
TEXT	L + 2 Bytes; L<2 <sup>16</sup>	For small non-binary string	RAW, BLOB
MEDIUMTEXT	L + 3 Bytes; L < 2 <sup>24</sup>	Medium sized non-binary string	RAW, BLOB
LONGTEXT	L + 4 Bytes; L < 2 <sup>32</sup>	Large non-binary string	RAW, BLOB
ENUM (VALUE1, VALUE2, ...)	1 or 2 Bytes; (65535 values max)	An enumeration; each column value may be assigned one enumeration member	
SET (VALUE1, VALUE2, ...)	1, 2, 3, 4 or 8 Bytes; (64 me- mbers max.)	A set; each column value may be assigned zero or more set numbers.	

### Spatial Data types for geo-information.

MySQL Type Name	Meaning
GEOMETRY	A spatial value of any type
POINT	A point (a pair of X,Y coordinates)
LINestring	A curve (one or more POINT values)
POLYGON	A polygon
GEOMETRYCOLLECTION	A collection of GEOMETRY values
MULTILINESTRING	A collection of LINestring values
MULTIPOINT	A collection of POINT values
MULTIPOLYGON	A collection of POLYGON values

**Table 3.1: MySQL Data types mapping with Oracle**

### 3.8 Oracle and MySQL Security

Oracle database security hallmarks are confidentiality, integrity, and availability. Who should have the right to access data? What portion of all the data should a particular user be able to access? What operations should an authorized user be able to perform on the data? Can authorized users access valid data when necessary? What authorization or permission is given to a user, program, or process to access an object or set of objects. The type of data access granted to a user can be read-only, or read/write. Privileges specify the type of Data Manipulation Language (DML) operations that the user can perform upon data.

Oracle uses schemas and security domains to control access to data and to restrict the use of various database resources. Some of the intrinsic security mechanisms of the Oracle database are;

- Oracle Identity Management
- Integrity
- Authentication and Access Controls in Oracle

- Privileges
- Roles
- Auditing
- Views, Stored Program Units, Triggers
- Data Encryption
- High Availability
- Proxy Authentication in Oracle
- Application Context in Oracle

Oracle uses a decentralized security system, where users are themselves responsible for granting access rights for the objects they own to the other users (SSIL 1992). Before doing this, the user will need the relevant privileges that allow activities such as connection to the database and creation of objects. When a new user is created by the Database Administrator, a number of system privileges viz. CREATE SESSION, CREATE TABLE, CREATE VIEW AND CREATE USER etc. may be granted to define the kinds of database activities to the users.

MySQL uses Access Control Lists (ACLs) based security for all connections, queries, and other operations that users can attempt to perform. There is also support for SSL (Secured Socket Layer)-encrypted connections between MySQL clients and servers (refman5.0, 2006). MySQL suggested to follow the guideline whenever running;

- Not to give anyone (except MySQL root accounts) access to the user table in the mysql database. The encrypted password is the real password in MySQL. Anyone who knows the password that is listed in the user table and has access to the host listed for the account can easily log in as that user.



- Not to grant more privileges than necessary. Never grant privileges to all hosts.
- Not to store any plain-text passwords in database. If the computer becomes compromised, the intruder can take the full list of passwords and use them. Instead, use MD5(), SHA1(), or some other one-way hashing function.
- Not to choose passwords from dictionaries.
- To invest in a firewall. This protects from at least 50% of all types of exploits in any software. Put MySQL behind the firewall or in a demilitarized zone (DMZ).

Like Oracle, user name and password is needed to connect to MySQL server. The password is not transmitted in clear text over the connection. Password handling during the client connection sequence is very secure due to uses of encryption algorithm.

All other information is transferred as text, and can be read by anyone who is able to watch the connection. If the connection between the client and the server goes through an untrusted network, compressed protocol may be used to make traffic much more difficult to decipher. MySQL's internal SSL support may also be used to make the connection more secure. Alternatively, SSH can be used to get an encrypted TCP/IP connection between a MySQL server and a MySQL client.

So, the MySQL security feature is as strong as Oracle because both used almost identical technology and same SSL/SSH encryption technology for the user password.

### 3.8.1 Privilege System

A privilege is a right to execute an SQL statement or to access another user's object. In Oracle, there are two types of privileges: system privileges and object privileges. Oracle system privileges allow users to perform a particular system wide action or a particular action on a particular type of schema object. For example, the privileges to create a tablespace or to delete the rows of any table in the database are system privileges. Many system privileges are available only to administrators and application developers because the privileges are very powerful.

Object privileges in Oracle allow users to perform a particular action on a specific schema object. For example, the privilege to delete rows of a specific table is an object privilege. Schema object privileges for tables allow table security at the level of data manipulation language (DML) and data definition language (DDL) operations. For example, an administrator can grant individual users the privileges to use the DML operations DELETE, INSERT, SELECT, and UPDATE on a table or view, or to use the ALTER, INDEX, and REFERENCES privileges to perform DDL operations on a table.

Privileges can be specified at the column level. It is possible to restrict a user's INSERT and UPDATE privileges for a table to individual columns of the table. Likewise, privileges can be specified at the row level. It is possible to restrict a user's SELECT, INSERT, UPDATE, and DELETE privileges for a table to specific rows of the table.

As a general rule, object privileges can only be granted by the object owner. However, an owner can also specify that a particular user has the right to grant a

privilege to others. The full range of privileges for any action on any object in the schema is typically granted by default to the administrator. Even this complete set can be delegated by the administrator to other users, that is, selectively granted to or revoked from any user or group. In particular, an administrator can grant an application developer or DBA the privilege to GRANT ANY OBJECT PRIVILEGE. Having this privilege can make it easier for the developer to do the security configuration tasks he/she faces, and aid the DBA in resolving access control problems as they arise.

MySQL privilege system is to authenticate a user connecting from a given host, and to associate that user with privileges on a database such as SELECT, INSERT, UPDATE, and DELETE. Additional functionality includes the ability to have anonymous users and to grant privileges for MySQL-specific functions such as LOAD DATA INFILE and administrative operations.

The MySQL privilege system ensures that all users may perform only the operations allowed to them. As a user, when we connect to a MySQL server, our identity is determined by the host from which we connect and the username we specify. When we issue requests after connecting, the system grants privileges according to our identity and what we want to do.

MySQL considers both hostname and username in identifying the user because there is little reason to assume that a given username belongs to the same person everywhere on the Internet. For example, the user joe who connects from office.com need not be the same person as the user joe who connects from elsewhere.com. MySQL handles this by allowing us to distinguish users on different hosts that happen

to have the same name, It can grant one set of privileges for connections by joe from office.com, and a different set of privileges for connections by joe from elsewhere.com.

MySQL access control involves two stages:

- Stage 1: The server checks whether it should allow us to connect.
- Stage 2: Assuming that we can connect, the server checks each statement we issue to see whether we have sufficient privileges to perform it. For example, if we try to select rows from a table in a database or drop a table from the database, the server verifies that we have the SELECT privilege for the table or the DROP privilege for the database.

The server stores privilege information in the grant tables (Table 3.2) of the mysql database (that is, in the database named mysql). The MySQL server reads the contents of these tables into memory when it starts and re-reads them under the change privileges circumstances. Access-control decisions are based on the in-memory copies of the grant tables. Normally, manipulation of the contents of the grant table are indirectly done by using the GRANT and REVOKE statements to set up accounts and control the privileges available to each one. The syntax of the statements are as follows;

```
mysql> GRANT all on mysql.* to username;
```

```
mysql> REVOKE grant all on MySQL.* to username;
```

The underlying structure of the grant tables and how the server uses their contents when interacting with clients are shown here. The server uses the user, db, and host

tables in the mysql database at both stages of access control. The columns in these grant tables are shown here:

Table Name	User	Db	Host
<b>Scope columns</b>	Host	Host	Host
	User	Db	Db
	Password	User	
<b>Privilege columns</b>	Select_priv	Select_priv	Select_priv
	Insert_priv	Insert_priv	Insert_priv
	Update_priv	Update_priv	Update_priv
	Delete_priv	Delete_priv	Delete_priv
	Index_priv	Index_priv	Index_priv
	Alter_priv	Alter_priv	Alter_priv
	Create_priv	Create_priv	Create_priv
	Drop_priv	Drop_priv	Drop_priv
	Grant_priv	Grant_priv	Grant_priv
	Create_view_priv	Create_view_priv	Create_view_priv
	Show_view_priv	Show_view_priv	Show_view_priv
	Create_routine_priv	Create_routine_priv	
	Alter_routine_priv	Alter_routine_priv	
	References_priv	References_priv	
	Reload_priv		
	Shutdown_priv		
	Process_priv		
	File_priv		
	Show_db_priv		
	Super_priv		
	Create_tmp_table_priv	Create_tmp_table_priv	Create_tmp_table_priv
Lock_table_priv	Lock_table_priv	Lock_table_priv	
Execute_priv			
Repl_slave_priv			
Repl_client_priv			
<b>Security columns</b>	ssl_type		
	ssl_cipher		
	x509_issuer		
	x509_subject		
<b>Resource control columns</b>	max_questions		
	max_connections		
	max_user_connections		

**Table 3.2: Grant Table**

During the second stage of access control, the server performs request verification to make sure that each client has sufficient privileges for each request that it issues. In addition to the user, db, and host grant tables, the server may also consult the tables\_priv and columns\_priv tables for requests that involve tables. The tables\_priv and columns\_priv tables provide finer privilege control at the table and column levels. They have the following columns:

Table Name	Tables_priv	Columns_priv
Scope Columns	Host	
	Db	
	User	
	Table_name	
		Column_name
Privilege columns	Table_priv	Column_priv
	Column_priv	
Other columns	Timestamp	Timestamp
	Grantor	

**Table 3.3: Table and Column Privilege**

For verification of requests that involve stored routines, the server may consult the procs\_priv table. This table has the following columns:

Table Name	Procs_priv
	Host
	User
	Routine_name
	Routine_type
<b>Privilege columns</b>	Proc_priv
<b>Other columns</b>	Timestamp
	Grantor

**Table 3.4: Process Privileges**

Information about account privileges is stored in the user, db, host, tables\_priv, columns\_priv, and procs\_priv tables in the mysql database. The MySQL server reads the contents of these tables into memory when it starts and re-reads them under the

circumstances. Access control decisions are based on the in-memory copies of the grant tables. The names used in the GRANT and REVOKE statements to refer to privileges are shown in the following table, along with the column name associated with each privilege in the grant tables and the context in which the privilege applies.

<b>Privilege</b>	<b>Column</b>	<b>Context</b>
CREATE	Create_priv	databases, tables, or indexes
DROP	Drop_priv	databases or tables
GRANT OPTION	Grant_priv	databases or tables
REFERENCES	References_priv	databases or tables
ALTER	Alter_priv	tables
DELETE	Delete_priv	tables
INDEX	Index_priv	tables
INSERT	Insert_priv	tables
SELECT	Select_priv	tables
UPDATE	Update_priv	tables
CREATE VIEW	Create_view_priv	views
SHOW VIEW	Show_view_priv	Views
ALTER ROUTINE	Alter_routine_priv	stored routines
CREATE ROUTINE	Create_routine_priv	stored routines
EXECUTE	Execute_priv	stored routines
FILE	File_priv	file access on server host
CREATE TEMPORARY TABLES	Create_temporary_tables_priv	server administration
LOCK TABLES	Lock_tables_priv	server administration
CREATE USER	Create_user_priv	server administration
PROCESS	Process_priv	server administration
RELOAD	Reload_priv	server administration

REPLICATION CLIENT	Replication_client_priv	server administration
REPLICATION SLAVE	Replication_slave_priv	server administration
SHOW DATABASES	Show_databases_priv	server administration
SHUTDOWN	Shutdown_priv	server administration
SUPER	Super_priv	server administration

**Table 3.5: Privileges**

### 3.8.2 MySQL User Name and Password

Both Oracle and MySQL user account is defined in terms of a username. In addition, MySQL account is also associated with client host or hosts from which the user can connect to the server. The account also has a password. There are several distinctions between the way usernames and passwords used by MySQL and the way they are used by the operating system.

### 3.8.3 Adding New user

In Oracle, the user is created by issuing the SQL statement CREATE USER with necessary parameters. Initially the user has no privileges. GRANT command is used to grant roles and privileges to the users. The Syntax of the statements are shown below;

```
CREATE USER user IDENTIFIED BY password
    [DEFAULT TABLESPACE tablespace]
    [TEMPORARY TABLESPACE tablespace]
    [QUOTA {integer {k | m} | UNLIMITED} ON
    tablespace]
    [PROFILE profile]
    | PASSWORD EXPIRE
```



```
| ACCOUNT {LOCK | UNLOCK} . . . . .
```

On the other hand, MySQL user accounts can be created in two ways; (1) by using GRANT statements, (2) by manipulating the MySQL grant tables directly. The suggested method is to use GRANT statements because of concise and less errorprone. Another option for creating accounts is to use one of several available third-party programs that offer capabilities for MySQL account administration. phpMyAdmin is one such program. The following examples show how to use the mysql client program to set up new users. These examples assume that privileges are set up according to the defaults. This means that to make changes, we must connect to the MySQL server as the MySQL root user, and the root account must have the INSERT privilege for the MySQL database and the RELOAD administrative privilege.

First, using the MySQL program to connect to the server as the MySQL root user:

```
shell> mysql --user=root mysql
```

If a password is assigned to the root account, then need to supply the password with --password or -p option for this mysql command.

After connecting to the server as root, accounts can be created. GRANT command can be used to create a new account with privileges.

```
mysql> GRANT ALL PRIVILEGES ON *.* TO
      'monty'@'localhost' IDENTIFIED BY 'some_pass' WITH
      GRANT OPTION;
```

```
mysql> GRANT ALL PRIVILEGES ON *.* TO
    'monty'@'%' IDENTIFIED BY 'some_pass' ->WITH GRANT
    OPTION;
mysql> GRANT RELOAD,PROCESS ON *.* TO
    'admin'@'localhost';
mysql> GRANT USAGE ON *.* TO 'dummy'@'localhost';
```

As an alternative to GRANT, INSERT statement may be used to add a new user with required privileges using FLUSH PRIVILEGES command. Some example is given below;

```
shell> mysql --user=root mysql
mysql> INSERT INTO user
    -> VALUES('localhost','monty',PASSWORD('some_pass')),
    ->
    'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');
mysql> INSERT INTO user
    VALUES('%','monty',PASSWORD('some_pass'),
    'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y',
    'Y');
mysql> INSERT INTO user SET
Host='localhost',User='admin',
    Reload_priv='Y', Process_priv='Y';
mysql> INSERT INTO user (Host,User>Password)
    VALUES('localhost','dummy','');
mysql> FLUSH PRIVILEGES;
```

The reason for using FLUSH PRIVILEGES after adding is to tell the server to re-read the grant tables. Otherwise, the changes go unnoticed until restart of the server. Similarly the reason for using the PASSWORD() function with INSERT is to encrypt the password. The 'Y' values enable privileges for the accounts.

The next examples create three accounts and give them access to specific databases. Each of them has a username of custom and password of obscure. To create the accounts with GRANT, use the following statements:

```
shell> mysql --user=root mysql
mysql> GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP TO
      'custom'@'whitehouse.gov' IDENTIFIED BY
      'obscure';
mysql> GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP ON
      customer.* TO 'custom'@'server.domain'
      IDENTIFIED BY 'obscure';
```

We can also give a specific user access from all machines in a given domain (for example, mydomain.com) issuing a GRANT statement that uses the ‘%’ wildcard character in the host part of the account name:

```
mysql> GRANT ... ON *.* TO 'myname'@'%.mydomain.com'
      IDENTIFIED BY 'mypass';
```

### 3.8.4 Limiting Account Resources

Like Oracle, MySQL 5.0, server resources can be controlled for individual accounts using max\_user\_connections system variable. The resources are;

- The number of queries that an account can issue per hour
- The number of updates that an account can issue per hour
- The number of times an account can connect to the server per hour

To set resource limits with a GRANT statement, WITH clause is used that names each resource to be limited and a per-hour count indicating the limit value. For example, the statement to create a new account that can access the customer database, but only in a limited fashion.

```
mysql> GRANT ALL ON customer.* TO 'francis'@'localhost'
-> IDENTIFIED BY 'frank'
-> WITH MAX_QUERIES_PER_HOUR 20
-> MAX_UPDATES_PER_HOUR 10
-> MAX_CONNECTIONS_PER_HOUR 5
-> MAX_USER_CONNECTIONS 2;
```

### 3.8.5 Assigning Account Passwords

Both Oracle and MySQL have their own provisions to assign and re-assign user password for the security purpose. Oracle uses ALTER USER statement but MySQL has several ways to assign password for a user account from command line using mysqladmin command, set password statement, GRANT USAGE statement, INSERT and UPDATE statement etc. Examples are showing below;

```
mysql> mysqladmin -u user_name -h host_name password
"newpwd"
```

```
mysql> SET PASSWORD FOR 'jeffrey'@'%' =
PASSWORD('biscuit');
```

```
mysql> SET PASSWORD = PASSWORD('biscuit');
```

```
mysql> GRANT USAGE ON *.* TO 'jeffrey'@'%' IDENTIFIED BY
'biscuit';
```

```
shell> mysql -u root mysql
```

```
mysql> INSERT INTO user (Host,User>Password)
-> VALUES('%','jeffrey',PASSWORD('biscuit'));
```

```
mysql> FLUSH PRIVILEGES;
```

```
shell> mysql -u root mysql
```

```
mysql> UPDATE user SET Password = PASSWORD('bagel')
      -> WHERE Host = '%' AND User = 'francis';
```

```
mysql> FLUSH PRIVILEGES;
```

```
shell> mysql -u root mysql
```

```
mysql> INSERT INTO user (Host,User,Password)
      VALUES('%','jeffrey','biscuit');
```

```
mysql> FLUSH PRIVILEGES;
```

### 3.8.6 Removing User Accounts from MySQL

Both Oracle and MySQL has DROP USER statement to remove an user account the syntax of which is as follows;

```
DROP USER
```

### 3.9 Transaction and Locking

A transaction is a logical unit of work that comprises one or more SQL statements run by a single user. According to the ANSI/ISO SQL standard, with which Oracle is compatible. An Oracle transaction begins with the user's first executable SQL statement. Transactions let users guarantee consistent changes to data, as long as the SQL statements within a transaction are grouped logically. A transaction should consist of all of the necessary parts for one logical unit of work—no more and no less. Data in all referenced tables are in a consistent state before the transaction begins and after it ends. Transactions should consist of only the SQL statements that make one consistent change to the data.

When a transaction begins, Oracle assigns the transaction to an available undo tablespace to record the rollback entries for the new transaction. A transaction ends when any of the following occurs:

- A user issues a COMMIT or ROLLBACK statement without a SAVEPOINT clause.
- A user runs a DDL statement such as CREATE, DROP, RENAME, or ALTER. If the current transaction contains any DML statements, Oracle first commits the transaction, and then runs and commits the DDL statement as a new, single statement transaction.
- A user disconnects from Oracle. The current transaction is committed.
- A user process terminates abnormally. The current transaction is rolled back.

After one transaction ends, the next executable SQL statement automatically starts the following transaction.

Like Oracle, MySQL supports same technology and uses same statements. MySQL supports local transactions (within a given client connection) through statements such as;

- SET AUTOCOMMIT
- START TRANSACTION
- COMMIT and
- ROLLBACK

MySQL version 5.0 and higher can participate in distributed transactions as well. The example of transaction statement is given below;

```
START TRANSACTION;

SELECT @A:=SUM(salary) FROM table1 WHERE type=1;

UPDATE table2 SET summary=@A WHERE type=1;

COMMIT;
```

MySQL InnoDB supports the SQL statements SAVEPOINT and ROLLBACK TO SAVEPOINT. Starting from MySQL 5.0.3, RELEASE SAVEPOINT and the optional WORK keyword for ROLLBACK are supported as well. The SAVEPOINT statement sets a named transaction savepoint with a name of *identifier*. If the current transaction has a savepoint with the same name, the old savepoint is deleted and a new one is set. Example is shown below;

```
SAVEPOINT identifier

ROLLBACK [WORK] TO SAVEPOINT identifier

RELEASE SAVEPOINT identifier
```

### **Locking**

In general, multiuser databases use some form of data locking to solve the problems associated with data concurrency, consistency, and integrity. Locks are mechanisms that prevent destructive interaction between transactions accessing the same resource.

Resources include two general types of objects:

- User objects, such as tables and rows (structures and data)

- System objects not visible to users, such as shared data structures in the memory and data dictionary rows

In all cases, Oracle automatically obtains necessary locks when executing SQL statements, so users need not be concerned with such details. Oracle automatically uses the lowest applicable level of restrictiveness to provide the highest degree of data concurrency yet also provide fail-safe data integrity. Oracle also allows the user to lock data manually.

MySQL LOCK TABLES locks tables for the current thread. If any of the tables are locked by other threads, it blocks until all locks can be acquired. UNLOCK TABLES releases any locks held by the current thread. A table lock protects only against inappropriate reads or writes by other clients. The client holding the lock, even a read lock, can perform table-level operations such as DROP TABLE. The main reasons to use LOCK TABLES are for emulating transactions or to get more speed when updating tables. The example is shown as;

```
mysql> LOCK TABLE t WRITE, t AS t1 WRITE;
mysql> INSERT INTO t SELECT * FROM t;
ERROR 1100: Table 't' was not locked with LOCK TABLES
mysql> INSERT INTO t SELECT * FROM t AS t1;
```

### 3.10 Replication

Replication is the process of creating and maintaining replica versions of database objects (e.g. tables) in a distributed database system. Replication can improve



performance and increase availability of applications because alternate data access options become available (OraReplFaq 2002). For example, users can access a local database rather than a remote server to minimize network traffic. Furthermore, the application can continue to function if parts of the distributed database are down as replicas of the data might still be accessible.

Replication allows us to take one database, make an exact copy of it on another server, and set one of them as slave to take all its updates from the other as master. The slave reads the master's binary logs, which store all statements that change a database, and repeats these on its database, keeping the two in exact synchronization. Since a replicating database simply repeats statements, the databases are not necessarily exactly in sync, and advanced users can take advantage of this.

Oracle supports two modes; snapshot and multimaster replication depending on the reasons of replication deployment. In some cases hybrid, a combination of two is useful. Snapshots are known as more descriptively "Materialized views" (MVs). An MV is an instantiation of a query and is updated via an individual batch update process known as "Refresh". In snapshot replication environment, one database holds the master and the other database contains the snapshots. Snapshots may be read-only, in which data flows only from the master to snapshots or updatable. Oracle propagates the changes made to data in updatable snapshots back to the snapshots master site. If other snapshots of the master exist, the changes are applied to them as well. In multimaster replication, replicated objects are peers and each of them is subject to DML operations. Each database must contain an identical copy of the object; subsets

based on queries are not supported. Data changes from each master are then continuously replicated from each of the other masters.

On the other hand, MySQL support for one-way, asynchronous replication, in which one server act as the master, while one or more other servers act as slaves. The master server writes updates to its binary log files, and maintains an index of the files to keep track of log rotation. These logs serve as records of updates to be sent to any slave servers. When a slave connects to the master, it informs the master of the position up to which the slave read in the logs at the last successful update. The slave receives any updates that have taken place since that time, and then blocks and waits for the master to notify it of new updates. A slave server can itself serve as a master if we want to set up chained replication servers.

One-way replication has benefits for robustness, speed, and system administration;

- Robustness is increased with a master/slave setup. In the event of problems with the master, we can switch to the slave as a backup.
- Better response time for clients can be achieved by splitting the load for processing client queries between the master and slave servers. SELECT queries may be sent to the slave to reduce the query processing load of the master. Statements that modify data should still be sent to the master so that the master and slave do not get out of sync. This load-balancing strategy is effective if non-updating queries dominate, but that is the normal case.

- Another benefit of using replication is that we can perform backups using a slave server without disturbing the master. The master continues to process updates while the backup is being made.

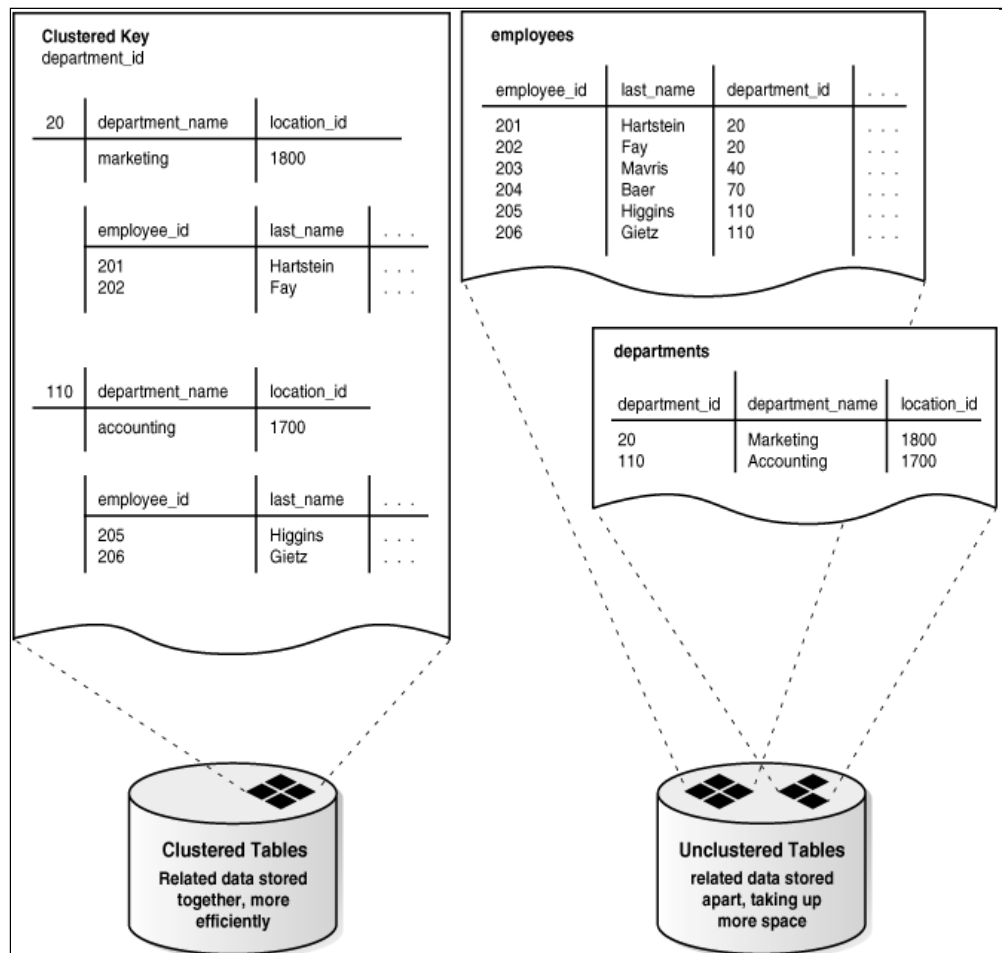
There are several ways to implement the replication process in MySQL. One way to copy the master's data to the slave is using the `LOAD DATA FROM MASTER` statement. Due to some limitations, the recommend way to use `LOAD DATA FROM MASTER` only if the dataset on the master is relatively small, or if a prolonged read lock on the master is acceptable. Although the actual speed of `LOAD DATA FROM MASTER` may vary from system to system, a good rule of thumb for how long it takes is 1 second per 1MB of data.

Replication is a useful administrative feature of MySQL. It is an excellent way to be assured of good regular backups of databases. There are number of options and SQL statements available for replication. For active and large systems, more than one slave server could be setup for added protection of data. The configuration and concepts are the same for multiple slaves as it is for one slave.

### **3.11 Clustering**

A cluster provides an optional method of storing table data. A cluster is made up of a group of tables that share the same data blocks. The tables are grouped together because they share common columns and are often used together. For example, the employee and department table share the `department_Id` column. When we cluster the employee and department tables, Oracle Database physically stores all rows for each department from both the employee and department tables in the same data blocks .

The following figure shows what happens when we cluster the employees and departments tables:



**Figure 3.4: Oracle Cluster**

Because clusters store related rows of different tables together in the same data blocks, properly used clusters offer these benefits:

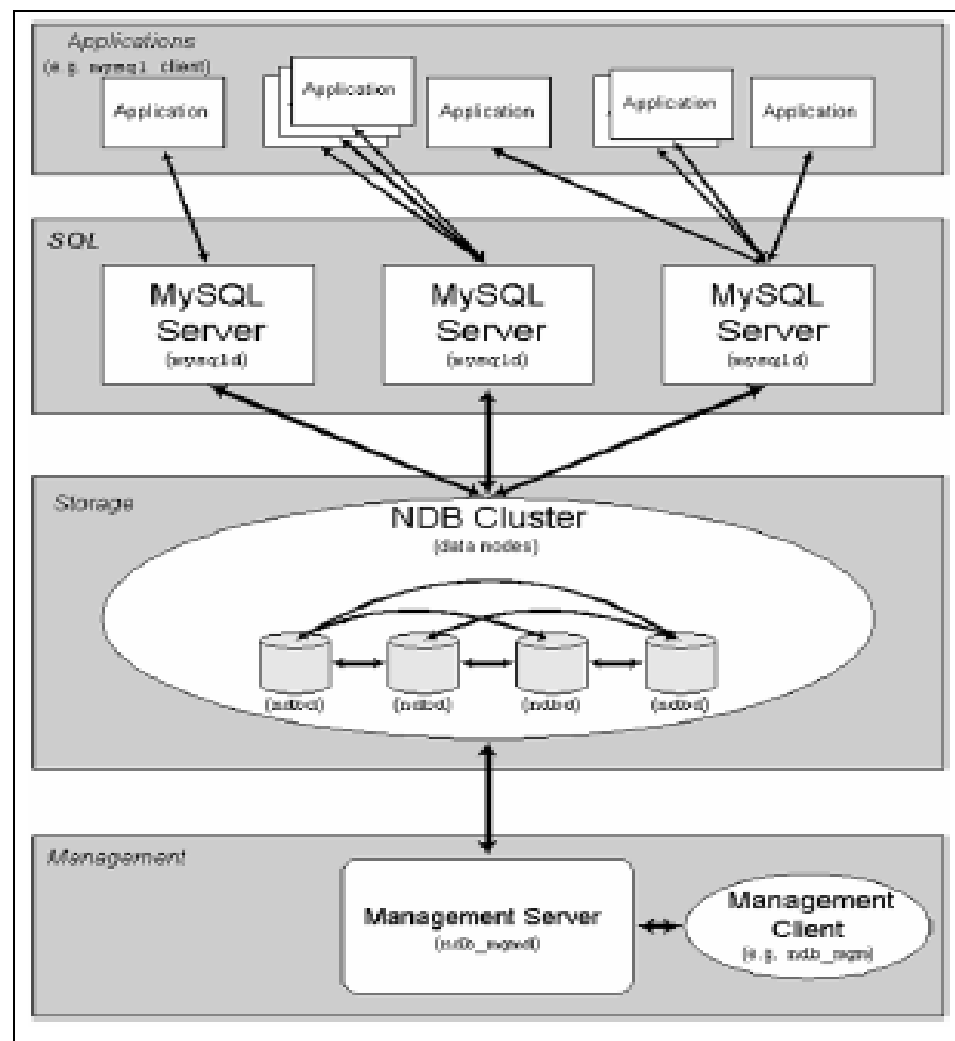
- Disk I/O is reduced for joins of clustered tables.
- Access time improves for joins of clustered tables.
- In a cluster, a **cluster key value** is the value of the cluster key columns for a particular row. Each cluster key value is stored only once each in the cluster and the cluster index, no matter how many rows of different tables contain the

value. Therefore, less storage is required to store related table and index data in a cluster than is necessary in nonclustered table format. For example, in figure 3.4, notice how each cluster key (each `department_id`) is stored just once for many rows that contain the same value in both the employees and departments tables.

Oracle implemented Real Application Clustering (RAC) which uses disk-share technology. The RAC benefits from a dramatic reduction in the cost of hardware components, which makes it more economically viable for even mid sized companies to build cluster (Niccolai 2001).

On the other hand, MySQL Cluster is a technology that enables clustering of in-memory databases in a share-nothing system. The share-nothing architecture allows the system to work with very inexpensive hardware, and without any specific requirements on hardware or software. It also does not have any single point of failure because each component has its own memory and disk. MySQL cluster integrates the standard MySQL server with an in-memory clustered storage engine called NDB. The term NDB refers to the part of the setup that is specific to the storage engine, whereas “MySQL Cluster” refers to the combination of MySQL and the NDB storage engine.

MySQL Cluster consists of a set of computers, each running a number of processes including MySQL servers, data nodes for NDB Cluster, management servers, and (possibly) specialized data access programs. The relationship of these components in a cluster is shown here.



**Figure 3.5: MySQL Cluster Architecture**

All these programs work together to form a MySQL cluster. When data is stored in the NDB Cluster storage engine, the tables are stored in the data nodes. Such tables are directly accessible from all other MySQL servers in the cluster. Thus, in a payroll application storing data in a cluster, if one application updates the salary of an employee, all other MySQL servers that query this data can see this change immediately.

The data stored in the data nodes for MySQL Cluster can be mirrored; the cluster can handle failures of individual data nodes with no other impact than that a small number of transactions are aborted due to losing the transaction state. Because transactional applications are expected to handle transaction failure, this should not be a source of problems. MySQL Clustered data management maintains high availability, high performance and scalability.

### 3.12 Optimization

Optimization is very important task for a database system to make it faster. It is a complicated task because it ultimately requires understanding of the whole system. While it may be possible to do some local optimizations with small knowledge of the system or application, the more optimal want the system to become the more knowledgeable about it.

The most important part for getting a system faster is of course the basic design. We also need to know what kinds of things our system will be doing, and what our bottlenecks are. The most common bottlenecks are: Disk seeks. It takes time for the disk to find a piece of data. Some other important tasks are;

- Disk reading/writing. When the disk is at the correct position we need to read the data. With modern disks in 1999, one disk delivers something like 10-20Mb/s. This is easier to optimize than seeks because we can read in parallel from multiple disks.
- CPU cycles. When we have the data in main memory (or if it already were there) we need to process it to get our result. Having small tables

compared to the memory is the most common limiting factor. But then, with small tables speed is usually not the problem.

- Memory bandwidth. When the CPU needs more data than can fit in the CPU cache the main memory bandwidth becomes a bottleneck. This is an uncommon bottleneck for most systems, but one should be aware of it.
- Design Limitations
- Portability: Portability
- Internal use:
- Benchmarks

Oracle has a number of tools for monitoring, diagnostic problems and performance tuning, oversee and optimization of database. Among them, advisors are powerful tools for database management. They provide specific advice on how to address the key database management challenges, covering a wide range of areas including space, performance, and undo management. In general, advisors produce more comprehensive recommendations than alerts. This is because alert generation is intended to be low cost and have minimal impact on performance, whereas advisors are user-invoked, consume more resources, and perform more detailed analysis. This, along with the what-if capability of some advisors, provides vital information for tuning that cannot be procured from any other source.

Similarly, MySQL supports different optimization ways for different part of the database management system. Areas of optimization in MySQL database are as follows;

- Optimizing SELECT AND Other statement



- Optimizing Queries with EXPLAIN
- Estimating Query Performance
- Speed of SELECT Queries
- WHERE Clause Optimization
- Range Optimization
- Index Merge Optimization
- IS NULL Optimization
- DISTINCT Optimization
- LEFT JOIN and RIGHT JOIN Optimization
- Nested Join Optimization
- Outer Join Simplification
- ORDER BY Optimization
- GROUP BY Optimization
- LIMIT Optimization
- How to Avoid Table Scans
- Speed of INSERT Statements
- Speed of UPDATE Statements
- Speed of DELETE Statements
- Other Optimization Tips
- Locking Issues
- Locking Methods
- Table Locking Issues
- Concurrent Inserts
- Optimizing Database Structure

- Design Choices
- Making data as small as possible
- Column Indexes
- Multiple-Column Indexes
- How MySQL Uses Indexes
- The MyISAM Key Cache
- MyISAM Index Statistics Collection
- How MySQL Opens and Closes Tables
- Drawbacks to Creating Many Tables in the Same Database
- Optimizing the MySQL Server
  - System Factors and Startup Parameter Tuning
  - Tuning Server Parameters
  - Controlling Query Optimizer Performance
  - How Compiling and Linking Affects the Speed of MySQL
  - Memory uses of MySQL
  - Uses of DNS in MySQL
- Disk Issues : Using Symbolic Links

### 3.13 Partition

Partitioning addresses the key issues in supporting very large tables and indexes by letting us decompose them into smaller and more manageable pieces called partitions. SQL queries and DML statements do not need to be modified in order to access partitioned tables. However, after partitions are defined, DDL statements can access and manipulate individual partitions rather than entire tables or indexes. This is how

partitioning can simplify the manageability of large database objects. Also, partitioning is entirely transparent to applications (Cyran 2003).

Each partition of a table or index must have the same logical attributes, such as column names, data types, and constraints, but each partition can have separate physical attributes such as `pctfree`, `pctused`, and tablespaces. Partitioning is useful for many different types of applications, particularly applications that manage large volumes of data. OLTP systems often benefit from improvements in manageability and availability, while data warehousing systems benefit from performance and manageability. Oracle provides the following partitioning methods:

- Range Partitioning
- List Partitioning
- Hash Partitioning
- Composite Partitioning

In MySQL, the user-selected rule by which the division of data is accomplished is known as a partitioning function, which can be the modulus, simple matching against a set of ranges or value lists, an internal hashing function, or a linear hashing function. The function is selected according to the partitioning type specified by the user, and takes as its parameter the value of a user-supplied expression. This expression can be either an integer column value, or a function acting on one or more column values and returning an integer. The value of this expression is passed to the partitioning function, which returns an integer value representing the number of the partition in which that particular record should be stored. This function must be non-constant and non-random. It may not contain any queries, but may use virtually any SQL

expression that is valid in MySQL, so long as that expression returns a positive integer less than MAXVALUE (the greatest possible positive integer). This is known as horizontal partitioning that is, different rows of a table may be assigned to different physical partitions.

MySQL server has SHOW VARIABLES command to examine that the server supports partitioning or not such as;

```
mysql> SHOW VARIABLES LIKE '%partition%';
```

```
+-----+-----+
| Variable_name      | Value |
+-----+-----+
| have_partitioning  | YES   |
+-----+-----+
1 row in set (0.00 sec)
```

If output "have\_partitioning with YES" indicated the MySQL server supports partitioning. The following example shows how to create a table that is partitioned by hash into 6 partitions and which uses the InnoDB storage engine:

```
CREATE TABLE ti (id INT, amount DECIMAL(7,2), tr_date
DATE)
ENGINE=INNODB
PARTITION BY HASH( MONTH(tr_date) )
PARTITIONS 6;
```

Some of the advantages of partitioning include;

- Being able to store more data in one table than can be held on a single disk or filesystem partition.

- Data that loses its usefulness can often be easily be removed from the table by dropping the partition containing only that data. Conversely, the process of adding new data can in some cases be greatly facilitated by adding a new partition specifically for that data.

### 3.14 Stored Procedure

A stored procedure is a set of SQL statements that can be stored in the server. Once this has been done, clients don't need to keep reissuing the individual statements but can refer to the stored procedure instead.

Stored routines (procedures and functions) are implemented in MySQL 5.0. MySQL follows the SQL:2003 syntax for stored routines. The MySQL implementation of stored routines is still in progress. The purpose and syntax of creating stored procedure are identical in both Oracle and MySQL.

A stored routine is either a procedure or a function are created with CREATE PROCEDURE and CREATE FUNCTION statements. A procedure is invoked using a CALL statement, and can only pass back values using output variables. A function can be called from inside a statement just like any other function (that is, by invoking the function's name), and can return a scalar value. Stored routines may call other stored routines. As of MySQL 5.0.1, a stored procedure or function is associated with a particular database. The example of stored procedure is given below;

```
mysql> delimiter //
mysql> CREATE PROCEDURE simpleproc (OUT param1 INT)
-> BEGIN
```

```
-> SELECT COUNT(*) INTO param1 FROM t;
```

```
-> END;
```

```
-> //
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> CALL simpleproc(@a);
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> SELECT @a;
```

```
+-----+
```

```
| @a |
```

```
+-----+
```

```
| 3 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

### **Trigger**

Oracle triggers are procedures written in PL/SQL, Java, or C that run (fire) implicitly whenever a table or view is modified or when some user actions or database system actions occur. Triggers supplement the standard capabilities of Oracle to provide a highly customized database management system ([http://www.dba-oracle.com/art\\_proc.htm](http://www.dba-oracle.com/art_proc.htm)). For example, a trigger can restrict DML operations against a table to those issued during regular business hours.

In MySQL, trigger is included beginning with MySQL 5.0.2. For example, the following statements create a table and an INSERT trigger. The trigger sums the values inserted into one of the table's columns:

```
mysql> CREATE TABLE account (acct_num INT, amount
-> DECIMAL(10,2));

Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TRIGGER ins_sum BEFORE INSERT ON account
-> FOR EACH ROW SET @sum = @sum + NEW.amount;

Query OK, 0 rows affected (0.06 sec)
```

### Views

MySQL 5.0 server supports views. A views is a virtual table created using CREATE VIEW command from one or more tables. Oracle and MySQL views are identical.

One example of views is as follows;

```
mysql> CREATE VIEW v AS SELECT qty, price, qty*price AS
-> value FROM t;

mysql> SELECT * FROM v;
+-----+-----+-----+
| qty | price | value |
+-----+-----+-----+
| 3 | 50 | 150 |
+-----+-----+-----+
```

### **3.15 Disaster Prevention and Recovery**

Oracle backup and recovery is generally on the physical backup of database files, which permit the full reconstruction of database. The files protected by the backup and recovery facilities built into Enterprise Manager include data files, control files, server parameter files (SPFILEs), and archived redo log files. With these database can be reconstructed. The backup mechanisms that work at the physical level protect against damage at the file level, such as the accidental deletion of a data-file or the

failure of a disk drive. Logical-level backups, such as exporting database objects like tables or tablespaces, are a useful supplement to physical backups for some purposes though cannot protect the entire database.

Oracle's flashback features provide a range of physical and logical data recovery tools as efficient, easy-to-use alternatives to physical and logical backups. Oracle has two flashback features that operate at a logical level: Oracle Flashback Table, which lets revert a table to its contents at a time in the recent past; and Oracle Flashback Drop, which lets rescue dropped database tables. Neither requires advance preparation such as creating logical-level exports to allow for retrieval of lost data, and both can be used while database is available.

Oracle Enterprise Manager's physical backup and recovery features are built on Oracle's Recovery Manager (RMAN) command-line client. Enterprise Manager carries out its backup and recovery tasks by composing RMAN commands and sending them to the RMAN client. Enterprise Manager makes available much of the functionality of RMAN, as well as providing wizards and automatic strategies to simplify and further automate implementing RMAN-based backup and recovery.

MySQL has strong disaster prevention and recovery provision like other commercial Database Management Systems. It supports several ways to make backup of tables and database. There are 2 ways to make SQL-level backup;

```
SELECT INTO ... OUTFILE or
BACKUP TABLE command
```



The other ways to backup full database are to use the mysqldump program, mysqlhotcopy, mysqlsnapshot and innodb2myisam script (FNAL 2005). The statements are shown below;

```
mysql> mysqldump --tab=/path/to/some/dir --opt db_name
```

```
mysql> mysqlhotcopy db_name /path/to/some/dir
```

The binary log functions of MySQL provide adequate means of incrementally backing up data with very little overhead and a careful use of mysqldump and flush logs can limit the downtime of the database to once a week for a short time depending on the size of data.

MySQL also allows to recover data after several types of crashes in several ways.

Some of the situations where data can be recovered are as follows;

- Operating system crash
- Power failure
- File-system crash
- Hardware problem (hard drive, motherboard, and so forth)

MySQL has provisions to restore database as completely as possible, as quickly as possible with minimum overhead than other commercial database management systems.

### 3.16 Operating Systems Compatibility

Oracle support all major Operating systems such as Windows all versions, all SCO UNIX including Linux, IBM AIX, Informix etc. But all type of hardware is not supported. So, it is jointly dependent on operating systems and hardware.

Similarly, as an open source database management system, MySQL support almost all available operating systems of the world. It runs on many different combinations of operating systems and hardware platforms. It divides these into three tiers:

- Fully Supported (FS)
- Conditionally Supported (CS)
- Limited Support (LS)

The details supporting platforms for MySQL server is listed below

MySQL Supported Operating System	Support Type
Red Hat Linux	FS
SuSELinux	FS
Fedora Linux	FS
Debian GNU/Linux	FS
Mandrake Linux	LS
Monta Vista Linux	LS
Microsoft Windows (all version)	FS/LS/CS
Sun Solaris	FS
Apple Mac OS	FS
FreeBSD/Open BSD/NetBSD	FS/CS/LS
HP-UX 11.00	FS
IBM AIX	FS
Novell Netware 6.0	FS
SCO Unix	FS/CS

**Table 3.6: MySQL supported Operating System**

### 3.17 Hardware Requirement, Installation and Maintainability

In comparison with Oracle and other commercial database management systems, MySQL needs very minimum resources. 32 bit Pentium 4 processor with minimum 32 MB RAM and 60 MB disk space are general requirement for installation. 128 MB RAM will give optimum performance. Where as Oracle version 9i needs minimum 512 MB RAM and 3 GB disk space. At the same time, it supports dual processor

server for windows operating system and multiprocessor for UNIX operating system i.e. SCO OpenServer 6.0.x (Refman5.0 2006).

MySQL distribution software is available in both installable and binary format for different platforms. Customization is possible in the binary file without any consent of MySQL AB; the owner company. Installation is easier than Oracle and other proprietary RDBMSs, because it has exceptional quick-start capability with less than 15 minutes average time from software download to installation completion. This rule is true for all of the platforms such as Microsoft Windows, Linux, Macintosh, or UNIX. Once installed, self-management features like automatic space expansion, auto-restart, and dynamic configuration changes take much of the burden off already overworked database administrators. The processes and options of the installation are unique for both client and server. No additional tools are required to configure to connect the database server unlike Oracle's Net8 Listener, SQL\*Plus and TNS configurations etc. Oracle has different installations for server and client but MySQL has same installation for both server and client.

MySQL also provides a complete suite of graphical management and migration tools that allow a DBA to manage, troubleshoot, and control the operation of many MySQL servers from any single workstation. Some of the tools are described in the next page;

- MySQL Administrator: This tool makes it possible for administrators to set up, evaluate, and tune their MySQL database server. This is intended as a replacement for mysqladmin.

- MySQL Query Browser: Provides database developers and operators with a graphical database operation interface. It is especially useful for seeing multiple query plans and result sets in a single user interface.
- Configuration Wizard: Administrators can choose from a predefined list of optimal settings, or create their own.
- MySQL System Tray: Provides Windows-based administrators a single view of their MySQL instance, including the ability to start and stop their database servers.

Unlike other commercial database management systems such as Oracle, MySQL does not need to work with server machine for administration purposes. Many third party software vendor tools are also available for MySQL that handle tasks ranging from data design and ETL, to complete database administration, job management, and performance monitoring.

So, it appears that installation and maintainability is much easier than Oracle as well as other commercial database management systems.

### **3.18 Support Services**

As a high cost commercial database management system; Oracle has support center. Oracle Corporation provides all types of post installation support as per contract of the license.

Similarly, MySQL has a Customer Support Center (CSC) accessible at <https://network.mysql.com> is a web application used by MySQL support engineers to manage and communicate with their customers on the incidents they file. The CSC

lets them easily keep track of all their support related issues. It keeps all related messages, files, and other details grouped together on an issue-by-issue basis. It also provides individual preferences, customer statistics, issue prioritization, time tracking, and many other very practical features. The CSC has built-in features to help users receive timely replies from support engineers. It facilitates sharing users issue among different technical specialists.

In addition to that there are also some community support forums available as an Open Source Database product. User can get any type of helps from the forums whenever needed.

### **3.19 Summary**

It is found from the above discussion that most of the necessary features required for small and medium scale enterprise database solution exist in both Oracle and MySQL. Most of the MySQL features are very easy to use in the database solutions.

Application development SQL statements under DDL and DML are almost identical in the Oracle and MySQL. Security features in MySQL are as strong as Oracle and easy to implement in various ways. MySQL security feature includes integrity constraints, user authentication, access privilege, host identification and encrypted password technology. Connectivity features for MySQL RDBMS are also comparable with Oracle.

MySQL column type are mapped to Oracle column type. MySQL column types: signed/unsigned integers 1, 2, 3, 4, and 8 bytes long, FLOAT, DOUBLE, CHAR,

VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, and OpenGIS spatial types.

Transaction, locking, replication, clustering and optimization features are also available in MySQL database server for distributed application to make the system faster. The implementation procedure and technology may be different but much easier than other commercial database management systems viz. Oracle.

Stored procedure, trigger, views are available in the MySQL version 5 and above. The implementation procedures are exactly same with Oracle. Disaster prevention, backup and recover technology are very flexible like Oracle. Server tools are available for the recovery process.

Unlike Oracle, MySQL does not need high cost hardware configuration. Maintainability is also easier than Oracle. The support service is available in various ways through MySQL CSC and user forum. So, MySQL may be a chosen database management system for an enterprise solution as cost cutting technology.

## CHAPTER 4

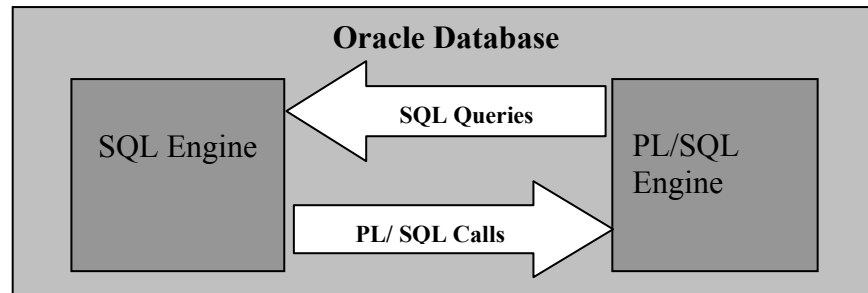
### FEATURES NOT PRESENT IN MYSQL

As a new concept, open source database management systems are still in improving stage. Most of the important features of database systems exist in MySQL OSDMS are already discussed in chapter 3. Besides, still there are some features are not implemented in MySQL. The purpose of the missing features some times can be metup by substituting ways. Anyway, some of the missing tools in MySQL are PL/SQL, Cursor, Synonym, Package and Sequence etc. Although they are very important in database application but developer can fulfill their needs in the database solutions using other existing features. Detailed descriptions of the missing features are given below;

#### 4.1 PL/SQL

PL/SQL is exclusively an Oracle's procedural language (PL) superset of structured query language (SQL) but not exists in other commercial language. It is an extension of SQL, incorporating many design features of language. It allows data manipulation and query statements of SQL to be included within block structured and procedural units of code. PL/SQL can encode our business rules through the creation of stored procedure and packages to trigger database events to occur or to add programming logic to the execution of SQL commands.

PL/SQL is proprietary procedural language that Oracle developed as an extension of SQL. Like SQL, it runs inside the database server and tightly coupled with SQL (Gennick *et al* 2002) shown in following figure.



**Figure 4.1: SQL and PL/SQL implementation in Database**

PL/SQL supports all standard built-in SQL functions and data types. PL/SQL code is grouped into structures called block. A PL/SQL block is a programming unit consisting of the following elements;

- Variable declaration
- Program code
- Exception handling code

Thus, PL/SQL combines the data manipulating power of SQL with the data processing power of procedural languages.

## 4.2 Cursor

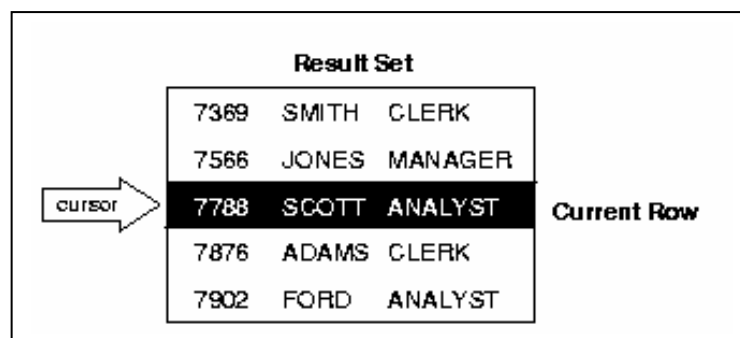
Oracle uses work areas to execute SQL statements and store processing information. A PL/SQL construct called a cursor lets us name a work area and access its stored information. There are two kinds of cursors: implicit and explicit. PL/SQL implicitly declares a cursor for all SQL data manipulation statements including queries that



return only one row. For queries that return more than one row, we can explicitly declare a cursor to process the rows individually. An example follows:

```
DECLARE
    CURSOR c1 IS
        SELECT empno, ename, job FROM emp WHERE deptno= 20;
```

The set of rows returned by a multi-row query is called the result set. Its size is the number of rows that meet our search criteria. The following figure shows, an explicit cursor "points" to the current row in the result set. This allows our program to process the rows one at a time.



**Figure 4.2: Query Processing**

Multi-row query processing is somewhat like file processing. For example, a COBOL program opens a file, processes records, then closes the file. Likewise, a PL/SQL program opens a cursor, processes rows returned by a query, then closes the cursor. Just as a file pointer marks the current position in an open file, a cursor marks the current position in a result set.

`OPEN`, `FETCH`, and `CLOSE` statements are used to control a cursor. The `OPEN` statement executes the query associated with the cursor, identifies the result set, and positions

the cursor before the first row. The `FETCH` statement retrieves the current row and advances the cursor to the next row. When the last row has been processed, the `CLOSE` statement disables the cursor.

### 4.3 Package

An Oracle package serves as a way to group procedures and functions into common groups typically based upon their functionality. A package has two sections: the specification that contains a definition of any objects that can be referenced outside of the package, and a body that contains the implementation of the objects. The specification section must be declared first:

```
PACKAGE package_name
IS
    {variable and type declarations }
    {cursor declarations}
    [module specifications]
END {package_name};
```

For example:

```
CREATE OR REPLACE PACKAGE Employee_pkg AS
    PROCEDURE GetEmployeeName (i_empno IN NUMBER, o_ename
    OUT VARCHAR2) ;
END Employee_pkg;
```

This defines a package called `Employee_pkg` that contains a single stored procedure called `GetEmployeeAge`.

The package body contains the actual implementation of the procedures within the package. This effectively allows us to hide procedures inside the package by not declaring them in the package specification:

```

PACKAGE BODY package_name
IS
    {variable and type declarations}
    {cursor specifications - SELECT statements}
    [module specifications]
BEGIN
    [procedure bodies]
END {package_name};

```

The specification for our Employee\_pkg could look like this:

```

CREATE OR REPLACE PACKAGE BODY Employee_pkg
AS
    PROCEDURE GetEmployeeName(i_empno IN NUMBER,
                              o_ename OUT VARCHAR2)
    IS
    BEGIN
        SELECT ename
        INTO o_ename
        FROM emp
        WHERE empno = i_empno;
    END GetEmployeeName;
END Employee_pkg;

```

#### 4.4 Synonym

Synonyms are a very powerful feature of Oracle and other SQL-compliant relational database systems. They are used as database shorthand. They make it possible to shorten the specification of long or complex object names. This is especially useful for shared tables or views. In addition, the use of DATABASE LINKS in synonyms allows transparent access to other databases on other nodes or even other entire systems halfway around the globe.

Synonyms are created using the CREATE SYNONYM command. Synonyms can be either PRIVATE (the default) or PUBLIC. Private synonyms can only be used by the user who create them. GRANT or one of the special ROLES must be assigned to user before creating synonyms. Only users with appropriate privileges (usually DBAs) can create PUBLIC synonyms. Since only the owner can use them, PRIVATE synonyms are more secure.

#### 4.5 Sequence

A sequence is an Oracle object used to deliver a series of unique number. Sequences are transaction independent. Each time of access a sequence, the value of that sequence is incremented/decremented by a predetermined amount. Committing or rolling back a transaction does not affect changes to a sequence. Sequences are stored in the database. Sequences are most often used to generate unique or primary keys for tables.

## 4.6 SQL\*Report

SQL\*ReportWriter is a general purpose tool for developing and executing reports, specially designed for application developers who know the SQL language. Information is entered into SQL\*ReportWriter through its fill-in-the-form interface. This interface allows to navigate quickly and easily throughout the report definition screens. A SQL\*ReportWriter report is made up of the following objects:

- **Queries** define the data to be retrieved from the database.
- **Fields** represent column expressions and report calculations from SELECT statements and describe how each is to be displayed.
- **Groups** contain sets of fields. Groups are used to describe each section or subsection in the report and its relationships, as well as to dictate control breaks for subtotaling purposes.
- **Summaries** are similar to fields, except they display subtotals and grandtotals rather than data directly from a query.
- **Texts** contain fields, summaries, and parameter references, combined with literal strings, such as titles, and define the final report format.
- The **Report** object defines the page size, margins, parameter form text, comments, security, and history of the report.
- **Parameters** contain literal values that can be supplied at runtime to control the behavior of the report. Data can be varied, routed of the output, and national language settings can be selected.

Every report contains at least one query, one group, and one field. Each object in a report has a set of attributes, or "settings", which provides information about the object. For example, fields have settings for Width and Display Format, among others. Most objects are initially created by default, and, in many cases, the default settings are sufficient. Simple forms are provided to edit the defaults and to create new objects, such as subtotals and grandtotals. Every SQL\*ReportWriter object (except Report) is owned by another object. For example, a group owns all of the fields within it. An object is not shared with another object; it has one, and only one, owner.

#### 4.7 Summary

PL/SQL, cursor, package and SQL\*Report are exclusively Oracle's own features. These are the extension of SQL in Oracle and mostly associated with the Oracle front end tool Developer/2000 (presently Developer 6i). The other renown and popular commercial database servers also do not have such programming tools. Without these features, the other existing database servers are meeting-up the requirement of database solution in substituting ways using front end programming language. MySQL supports C and C++ language function which may be the replacement of Oracle's PL/SQL. So, this is not the major disadvantage of MySQL database for mid market database solutions.

Similarly instead of the Oracle synonym and sequence, MySQL has same features such as MERGE tables that can specify a single table to act as a single table synonym or alias and more powerful Auto-Increment column type to generate sequence number value at the initial value within a group which Oracle sequence can not do.

## CHAPTER 5

### IMPLEMENTATION COMPARISON OF ORACLE AND MySQL

Oracle is one of the widely used and reliable database management systems of the world. At the same time, it is the most costly database system among the commercial database management systems. High cost hardware and efficient manpower are prerequisite to implement the Oracle database system.

On the other hand, MySQL is an open source database management systems. License price is free of cost. It is also one of the popular database management systems of the world. At present, MySQL has become a comparable RDBMS with the commercial database management systems. It is already proved that MySQL can be appropriate database management system for a mid market solution. Bangladeshi companies can implement MySQL database management systems as a cost cutting solution if the requirement is fulfilled.

However, a comparative analysis of Oracle and MySQL database application is given below;

#### 5.1 Oracle RDBMS Implementation

Progress Monitoring System (PMS) was developed for monitoring the progress of 50 on-going infrastructure development projects under the Local Government Engineering Department (LGED), Government of Bangladesh. The development projects include the activities; earth works, development of rural roads, bridges,

culverts, ghats, development of markets, drainages, school buildings, union parishad buildings, cyclone centers etc. The PMS deals with information of the rural infrastructure development engineering schemes of 50 on-going LGED projects all over Bangladesh. Project and scheme wise monthly progress data such as project identification, scheme identification, scheme location, scheme status, physical progress, financial progress, constraints and remarks etc were included in the systems. The output of the system was to produce project wise monthly/quarterly reports, government agency prescribed reports, department (LGED) and donor required reports. I worked with the project as one of the developer of the team in the year 2000.

### **5.1.1 Scope of the PMS**

PMS is a client/server database application system used for monitoring the progress of the 50 on-going projects under LGED. Oracle and VBA programming with MS Access was used as backend and front-end tools respectively. The feasibility study, requirement analysis, system analysis and design of the system were done by an international expert.

However, the PMS includes static and dynamic information from 50 ongoing infrastructure development projects under 64 districts in Bangladesh. Static information covers scheme identification such as project name, implementation area, scheme identification as per location (district, thana, union and village) of the schemes and dynamic data covers activity wise monthly physical and financial progress etc.



PMS consists of two versions. One is District PMS and the other is Head Quarter PMS/Central PMS. LGED District offices used the district PMS for entering information received from respective thana offices weekly or monthly. After completion of weekly and monthly data entry, district office send this monthly data to the respective project head offices located in LGED HQ, Dhaka to import into the central PMS. Respective Project receives their progress data from field (District) each month to append into the central PMS and produce required monthly and quarterly reports.

### 5.1.2 System Configuration of PMS

The following configuration was implemented for smooth running of the PMS.

#### **Software used:**

- Oracle 8i as backend server
- VBA Programming with MS Access as front end programming
- Windows NT 4.0 as OS

#### **Hardware used:**

- 1 Mid level servers (for database server)
- 1 Mid level server for backup server

#### **Manpower:**

- 1 MIS Advisor (International)
- 2 Developers
- 1 Database Administrator (DBA)

**Database features used:**

- Tables with referential integrity (more than 48 tables)
- Views (around 50 views)
- Trigger
- Stored Procedure
- ODBC connectivity to connect the tables, views with the Front end

**5.1.3 Justification of Tools selection**

PMS is a continuous updateable database system. Information coverage area of the system is too big. The rural infrastructure development schemes from all over Bangladesh were included under PMS. On an average 5000 records are added in to the progress table of the system in every month. Only the progress table is updated monthly and the others (static) among 48 tables are updated occasionally. After long run, the database size will be several gigabyte and table (progress data table) size will also be several hundred megabyte because around 75,000 records are added into the progress table yearly. In addition to that at least 100 concurrent users from 50 projects will get access to the central database system at a time. Considering the situation, the tools were selected with following justification.

**RDBMS Selection: Oracle 8i**

As client/server application was suggested to implement the PMS with several gigabyte size database in the long run operation (10 to 15 years) and 100 concurrent users access at a time, the Oracle version 8i (currently available version in the year 2000) was selected to purchase for the PMS data management. At the same time

database speed, performance, reliability in terms of security, disaster management i.e. backup and recovery etc. were considered. A proto type was also implemented with dummy records to check the requirement before procurement of the Oracle RDBMS. Oracle 8i was found best-suited database system for the PMS after necessary pre-testing implementation.

### **Front end selection: VBA with MS Access 97**

As front-end is the media to interact with the users and the backend database. It cannot influence database properties. Any front can be selected as per availability of the developers and connectivity compatibility with the backend database. VBA programming with MS Access was selected for PMS because most of the users (LGED engineers) were experienced with MS Access. They can process MS Access data after importing from Oracle database as per their requirements. Moreover, MS Access 97 was already purchased with valid license and used with other database operations.

### **Hardware and Operating Systems**

As Oracle 8i was selected for the client/server database application, Mid level server (HP brand) and Windows NT 4 was selected respectively for hardware and OS as per the configuration of the Oracle database.

#### **5.1.4 Security Implementation**

Two layer of security (database level and front end level) were implemented in the PMS system. Front end programming interface control the project level users access

to the database. User of one project cannot access data of another project. During installation of the front end interface, the project name was selected for operation.

In backend database, two levels of security were implemented in the system which are described below;

**User Level Security** – Only authorized users got access to the database with privileges (INSERT, UPDATE, DELETE & SELECT etc.) granted to him/her. Database was kept safe from unauthorized users as well as any destructive operation.

Four types of users were implemented in the systems;

- LGED management. They are allowed to view data and to make reports for all projects. They are not allowed to change any data.
- PIO power user. Allowed to view and change all data of their own project. This includes the project configuration, the static data and the dynamic data. Also allowed to import district data for their project.
- PIO user. Allowed to view all data of their own project. Allowed to change the dynamic project data.
- MIS administrator. Allowed to change all data of all projects. It is assumed that the MIS administrators treat all data confidentially, and only change project data when assisting the PIOs or in case of emergencies.

The user authorization would be enforced by Oracle. This means that even when a user without the proper access rights succeeds in opening the access forms or tables, Oracle still blocks unauthorized access to the data.

**Column Level Security** –Users were granted limited resources such as selected columns of the views. Tables were not linked with the MS Access due to ODBC

limitations for the Oracle. Some user were allowed to work the master data (Static data-Union, Upazilla, Scheme activity etc.) and some users were allowed to work with progress data and some were not allowed to execute INSERT, UPDATE and DELETE operation but allowed to execute the SELECT statement etc.

### 5.1.5 PMS development Profile (Duration, team and budget)

As a European Commission funded project implemented by LGED, the development team consists of international and national experts with proper budget to institutionalize with LGED. The details profile including approximate budget of the project is shown below;

Sl. No.	Activity/Description	Budget/Time	Remarks
1.	Project Duration	18 months	(development, piloting & implementation)
2.	Project Budget (in taka) <ul style="list-style-type: none"> <li>▪ Hardware (server, Laptop, Desktop and others)</li> <li>▪ Software</li> <li>▪ Database (initially 100 user Oracle license)</li> <li>▪ Manpower (including 1 International and 3 national team members)</li> </ul>	10,000,000/- 1,000,000/- 500,000/- 3,000,000/- 5,500,000/-	(Approximate)
3.	Manpower <ul style="list-style-type: none"> <li>▪ MIS Advisor (International Expert)</li> <li>▪ DBA (National)</li> <li>▪ Developer (National)</li> </ul>	1 person 1 person 2 person	

**Table 5.1 : PMS project profile**

### 5.1.6 Project threat

During the development of the PMS, the project was interrupted for 2 months due to some unavoidable circumstances with the donor agency. Government could not continue due to unavailability of fund. At that time 2 members of the team had left the project. Another couple of month was delayed to reappoint the developers.

## 5.2 MySQL RDBMS Implementation

Biodiversity Baseline Database System (BBDS) has been developed for maintaining information regarding biodiversity of coastal areas under the UNDP funded Coastal and Wetland Biodiversity Management Project, Department of Environment, Government of Bangladesh. The system covers information of biological diversity of agriculture, horticulture, plant, fisheries, wildlife as well as ecotourism of the coastal zone in Bangladesh.

### 5.2.1 Scope of the BBDS

BBDS is a client/server application software used to maintain information for biodiversity baseline survey of the Coastal and Wetland Biodiversity Management Project at Cox's Bazaar and Hakaluki Haor area. Visual Basic Programming with MS Access 2000 used as front end programming interface and MySQL as backend database server for the system. ODBC connectivity for MySQL i.e. MyODBC is used as connectivity tool to connect the database with the front end.

Along with database security, necessary integrity constraints (primary and foreign key) are maintained in BBDS to strengthen the data security and quality. Total 42

tables are implemented in the system. Among them 6 tables are used to store dynamic data that will be updated regularly. Several millions records will be accumulated by the next 5 years of operation. The estimated size of the database will be less than 1000 MB and table size will be few hundred MB with several million records per tables. Since database speed is not affected for implementing functions and procedures in the front-end programming level, stored procedure and trigger are avoided due to complexity. A prototype is tested with dummy records before implementation of the system.

### 5.2.2 System Configuration of PMS

The configuration of the BBDS is as follows;

#### **Software Requirement:**

- VBA Programming with MS Access as front end programming
- MySQL 4.1 as back end server
- Windows 2000 Professional

#### **Hardware Requirement:**

- 1 Pentium 4 PC (for database server)

#### **Manpower:**

- 1 Database Specialist
- 1 Developer for both backend and front end development

### 5.2.3 Justification of Tools selection

#### **RDBMS Selection: MySQL 4.1**

As per the requirement of the solution, BBDS is a multiuser client/server database system. No budget line is allocated for database implementation. As an alternative, free of cost open source database management system is considered for the solution. MySQL is selected as a leading OSDMS, which fulfills the requirements of the solution. Moreover, MySQL will run on the normal hardware (Desktop PC) and need not purchased high cost server. The implementation is also easier than others.

#### **Front end Selection: VBA MS Access 2000**

Budget limitation and technicality are considered for the tools selection. VBA programming with MS Access 2000 is selected as because MySQL supports MS Access 2000 VBA programming with ODBC connectivity viz. MyODBC which is available in the project and easy for the users for further development and data processing. No additional budget is required for the front end. Moreover, front end does not influence in the performance of the system.

### 5.2.4 Security Implementation

Two levels of security are implemented in the system which is described below;

- **User Level Security** – Only authorized users get access to the database with necessary privileges INSERT, UPDATE, DELETE & SELECT etc. granted to him/her. Database is kept safe from unauthorized users as well as any destructive operation
- **Normal User** - users are given only SELECT privileges to view the data



- **Power user** - they are given INSERT, UPDATE and SELECT privileges to view the data/reports
- **Admin User** - are given all privileges of root user for all types of operations.
- **Resource Limited Security** –Users are granted limited resources with restriction. As a result, number of access to the server per hour is limited to the user and limited number of queries can be run per hour as per the privileges of the users

### 5.2.5 BBDS development team and budget

It is a two-member development team and cost cutting system. The hardware and database budget is almost nil which is shown below;

Sl. No.	Activity/Description	Budget/Time	Remarks
1.	Project Duration	24 months	Including implementation
2.	Project Budget (in taka) <ul style="list-style-type: none"> <li>▪ Hardware (3 Desktop PCs, Laptop, Desktop and others)</li> <li>▪ Software</li> <li>▪ Database</li> <li>▪ Manpower</li> </ul>	2500,000/- 250,000/- 250,000/- Nil 2,000,000/-	(Approximate)
3.	Manpower <ul style="list-style-type: none"> <li>▪ Database Specialist</li> <li>▪ Developer (National)</li> </ul>	1 person 1 person	

**Table 5.2 : BBDS project profile**

### 5.2.6 Project threat

As ease of installation, development and maintenance, the threats to MySQL implementation is very minimum than others commercial database management

systems. No interruption such as data corruption, data loss and server down etc. was found yet

### 5.3 Summary

It is found from the above discussion and as a member of the development team in both the projects that the two systems are almost same. Both have fulfilled the requirements of the organizations. Almost same features are implemented in both the systems such as tables, views, integrity constraint and security etc. Both databases have maintained integrity constraints to prevent the data from any destructive operation. In addition to that two layer security; user authentication and column level privileges are implemented in both the systems. Tables and views are also implemented in Oracle and MySQL database systems. Stored procedure and trigger are used in the Oracle systems. But front end function and procedure are used in the MySQL system instead of trigger and stored procedure.

The main dissimilarity is in costing and level of complexity. Oracle needs higher cost for implementation and more complex to manage the security. One additional expert worked as DBA to develop and maintain the database backend component. The costing of which is around 7 lacs (approx.). In addition the Oracle software for 100 users was also very expensive amounting Tk. 30 lacs (approx.).

On the other hand, BBDS project is being implemented by the cost 25 lacs taka (approx.). The MySQL RDBMS license is totally free of cost instead of 30 lacs, no additional developer is appointed as DBA because of ease of use and no high-configured server is needed for small/medium scale enterprise solutions. Anyway the detail comparison table is shown below;

Sl. No.	Activity/Description	PMS	BBDS
1.	Estimated Database Size	1 GB	1 GB
2.	Estimated Total Records	5 Millions	5 Millions
3.	Total Tables	48	42
4.	Views	45	Front-end query
4.	Application Type	Client/server	Client/server
5.	Concurrent users	100 users	unlimited users
6.	Database used	Oracle 8i	MySQL 4.1
7.	Front end programming	VBA with MS Access 97	VBA with MS Access 97
8.	Connectivity	ODBC	MyODBC
9.	Project duration	18 months	24 months
9.	Budget Allocation (taka)	10 millions	2.5 millions
10.	Database Cost	3.0 million	nil
11.	Manpower	4 persons	2 persons
12.	Complexity	high	low
13.	Error reported	ODBC error found	ODBC error not found yet.
14.	Security implementation	same in both as discussed.	same in both as discussed.
15.	Performance	good	Not significantly differ with Oracle
16.	Installation & configuration	Complex	Easier than Oracle

**Table 5.3: Oracle vs MySQL implementation**

Though Oracle is a high cost database management system, lot of features such as indexing, PL/SQL, cursor, optimization, big database handling capability i.e. data warehousing etc. are still un-comparable to other database management systems. On the other Open source databases are still in the improving stage. So, for some situation, no alternative to the Oracle is available in the world.

However, it is possible to implement a database solution by 30%-50% total cost of commercial database systems. So, considering the requirement of the organization, MySQL may be the best solution for small and medium scale enterprises of Bangladesh.

## CHAPTER 6

### SUMMARY

Open Source software is becoming attractive issue in the ICT field of the world. Open Source software is offered with free license and source code is available in the internet for down loading without any cost. Many renowned international business corporations are choosing open source database systems as new installation as well as migrating from the proprietary database systems for reducing their ICT budget. A number of Open Source Database Management Systems are available in the world. It is found from an internet survey that the application of open source database systems has increased 30 percent in the years 2003 where as only 6 % for proprietary commercial database systems (Cox 2004).

In this situation, most of the business organizations in Bangladesh are using different commercial database management systems to maintain their information for the purpose of planning and decision-making without any requirement assessment. Some individual organizations avoid this data management technology due to high implementation and management cost despite realizing its importance. Like the international companies, Bangladeshi enterprises can adopt low cost open source database systems as an alternative solution.

#### 6.1 Selection of OSDMS: MySQL

Selection of database management systems is not based on the heritage but on the existence of solutions required features, performance, reliability, ease of use and

finally cost meeting capabilities etc. MySQL is one of the most popular open source free database system which offers both cost free and commercial licenses. According to the CD Times magazine's deployed databases polls, MySQL is the 3rd ranked (33% users) database in the world among polled databases SQL server, Oracle, DB2 and others. (FNAL, 2004).

MySQL is already chosen by many worlds' renowned companies as new implementation as well as migrating from the commercial database systems for cost reducing purpose (devx, 2005). It is shown that MySQL can be maintained by 30% of the commercial database budget per year (GCN 2005). According to Haff G., Illuminata analyst, CRM Daily, MySQL is perfect enough for small and medium scale databases (Haff 2005). MySQL is available in the Internet as binary and executable form. Developers can download, modify and compile the source code on the target server.

## **6.2 Comparative of the existing features of MySQL with Oracle**

As Oracle is the most widely used, reliable, high performance and costly RDBMS in the world, MySQL is compared in the perspective of small and medium scale enterprise solution. All features required in small/medium scale enterprise solutions are compared with oracle and MySQL. Anyway, the detail findings of the comparison are presented below;

### **6.2.1 SQL Statements: DDL, DML and Subquery**

Like Oracle, MySQL supports standard SQL statements. The statements for DDL and DML are identical. In some cases, MySQL statements are easier than oracle but fulfill

the requirements. The syntaxes are also same except some keyword. MySQL supports sub-query with Oracle's supported keywords except MINUS and INTERSECT keywords. These can be overcome in alternative ways.

### **6.2.2 Tables**

As MySQL supports different types of tables; transitional (MyISAM, MERGE, ISAM and HEAP) and non transitional (InnoDB and BDB) against one type of Oracle table, MySQL table capabilities are more powerful and speedy. MySQL supports maximum 3398 columns per table with 2 GB size but Oracle support 1000 and both up to 32 columns in index key (FNAL 2005). MySQL has proven to be fast at handling links among billions of rows of data in several virtual tables.

Both Oracle and MySQL support temporary tables but Oracle data is recoverable from the temporary tables in the event of database failure. On the other hand Oracle supports external tables to allow users to query data in flat files as if the data was in the database tables.

### **6.2.3 Indexes**

MySQL supports one type of index implementation with different access paths using - various table types where as Oracle supports eight types of index which helps faster data retrieval especially for the large databases such as data warehouses. For small and medium scale databases MySQL index is more enough for faster access to data.

#### **6.2.4 Database Constraint**

Like Oracle, MySQL supports all integrity constraints such as PRIMARY key, FOREIGN key, UNIQUE key, NULL/NOT NULL and CHECK options except CASCADE option.

#### **6.2.5 Database Connectivity**

MySQL and Oracle both have their own connectivity tools to connect to the database server with the front end interface. MySQL supports ODBC connectivity; MyODBC supports all programming languages, Connector/NET for .NET programming and connector/J for Java programming language.

#### **6.2.6 Data Types**

Oracle supports six types of native data type. MySQL supports standard primitive data types; String, Numeric, date and Time types. MySQL design data types to store data as quickly as possible because it does not validate data content and truncate values that exceed the column precision.

#### **6.2.7 Sequence and Auto-Increment**

MySQL uses an AUTO-INCREMENT data type when creating a table. The ability to start and increment sequence number value at the initial value within a group is a useful MySQL feature and is not available in Oracle. But Oracle uses SEQUENCE objects to generate sequence number.



### 6.2.8 Security

MySQL uses the user name, password and host to lookup the user's privileges in the system tables. The user table stores database-level privileges by user, and two tables maintain object-level privileges to restrict access to tables and columns. An administrator creates users by issuing GRANT statements, or by inserting values directly into the user table.

Unlike Oracle, MySQL does not use roles or groups to grant and revoke privileges to multiple users in individual statements. The absence of database roles is the major difference of MySQL with Oracle. Without the ability to group users into roles, the database administrator will have to rely on other methods to track which users should have which permissions to each database object.

### 6.2.9 Transaction

Oracle Database only supports one type of table within the database, all data inserts, updates, and deletes for Oracle tables are transactional. Oracle takes the approach that every insert, update, and delete statement is part of transaction by default, and supports all transactional capabilities including save points. In addition, Oracle Database provides for autonomous transactions. Autonomous transactions are a transaction within a transaction, and are useful for logging or debugging purposes. Inserts, updates, and deletes within an autonomous transaction are independent of any open transaction.

MySQL views transactional tables as a significant performance hit, and only recommends using transaction tables if necessary. If an application requires transactional capabilities, a table type that supports transactions, either InnoDB or

BDB, is required; the default MyISAM table type does not support transactions. Both non-transactional and transactional tables can exist in the same database, and a single query can select from or modify both types of tables. MySQL takes an alternate approach that nothing is part of a transaction by default, and the database engine automatically commits all inserts, updates, and deletes unless the auto-commit functionality is disabled. MySQL5.0 and higher support the COMMIT, ROLLBACK, SAVEPOINT and ROLL BACK TO SAVEPOINT statements.

Applications that depend on data integrity should never consider using non-transactional tables. In addition, the benefits of row-level locking and read consistency associated with all Oracle9i Database tables and InnoDB or BDB tables in MySQL ensure that the query processes data as it existed when the query started.

### **6.2.10 Replication**

Oracle Databases create materialized views as replicas of tables residing in the master database. Because the materialized views are updateable where the changes propagate to the master database, Oracle9i Database supports two-way replication.

MySQL supports one-way replication from the master to the replica by applying transaction log files to the replica database. If the database replicates transactional tables, utilize the hot backup utility to transfer the log files containing the changes to apply to the replica, or shutdown the database before copying the log files. Replication is a powerful tool for creating copies of databases to minimize the load of the master database, where the replication facility updates the replica as needed. Both Oracle9i Database and MySQL support one-way replication.

### 6.2.11 Clustering

Both Oracle and MySQL support database cluster with different ways. MySQL enables clustering of in-memory databases in a share-nothing system. The share-nothing architecture allows the system to work with very inexpensive hardware and without any specific requirements on hardware or software. It also does not have any single point of failure because each component has its own memory and disk.

### 6.2.12 Partitioning

MySQL supports user-selected rule by which the division of data is accomplished is known as a *partitioning function*, which can be the modulus, simple matching against a set of ranges or value lists, an internal hashing function, or a linear hashing function. The function is selected according to the partitioning type specified by the user, and takes as its parameter the value of a user-supplied expression.

Where as Oracle supports four partitioning methods; Range Partitioning, List Partitioning, Hash Partitioning and Composite Partitioning.

### 6.2.13 Optimization

Both Oracle and MySQL have number of optimization tools to make faster of the database. Oracle has a number of tools for monitoring, diagnostic problems and performance tuning, oversee and optimization of database. Advisor is one of the powerful tools to tune the Oracle database. Similarly MySQL has a number analysis

tools and ways to detect the performance in different part of the database such as SELECT statement, sub-query, Query speed optimization, range optimization etc.

#### **6.2.14 Stored Procedure and trigger**

Oracle supports stored procedures written in PL/SQL as well as other languages such as Java. MySQL version 5.0 and higher also supports stored procedure which follow SQL:2003 syntax for writing the routine. In addition to that C and C++ language are also supported by the MySQL 5.0 .

#### **6.2.15 Backup and Recovery**

Oracle Database provides for both hot and cold backups, and includes the Recovery Manager (RMAN) utility to facilitate the backup process.

Since MySQL stores data in operating system files, administrators perform cold backups by simply copying the files. MySQL also supports hot backups of InnoDB tables.

#### **6.2.16 Platform support**

Oracle support all major Operating systems such Windows all versions, all SCO UNIX including Linux, IBM AIX, Informix etc. Similarly, MySQL supports almost all available operating system in the world and with combination of operating systems.

### **6.2.17 Hardware Requirement, Installation and Maintainability**

Hardware requirement for MySQL is very low; minimum 32 MB RAM with 60 MB hard disk space. Installation is very easy, quick and unique for both client and server, which needs less than 15 minutes time from software download to installation.

It is a high-performance, relatively simple database system. From the beginning, MySQL has typically been configured, monitored, and managed from the command line. No additional tools need to be configured after installation of the database unlike Oracle's Net8 listner and SQL\*Plus etc. In addition, several MySQL graphical interfaces are available for easy maintenance of the system from any single workstation.

### **6.2.18 Vendor's support**

As commercial database software, customer support is a business policy for Oracle Corporation.

On the other hand, though MySQL is an OSDMS, it has two types of license; one is free and the other is general public license, for which MySQL AB provides customer support services through their Customer Support Center (CSC) accessible to all users in all time. In addition, open source community supports are also available in all time.

## **6.3 Conclusion**

It is found from the analysis of the thesis that MySQL is a reliable, scalable, full featured and high performance open source database management system. The

required database features for small and medium scale enterprise solution exist in the MySQL database system. All of the MySQL features are very closely comparable to other commercial database management systems viz. Oracle. In addition, it is very easy to use and manage which means database administrators do not waste time in troubleshooting performance and downtime issues. MySQL provides rock-solid reliability and high availability options from high speed master/slave replication, shared-nothing clustering and robustness transaction. It also supports strong data protection with exceptional security provisions using user authentication, privileges, SSH and SSL encryption technology and backup-recovery policies. MySQL provided comprehensive application development tools including connectors and drivers (ODBC, JDBC etc.) for all types of front end programming language and provided 24 hours technical support services by the owned company MySQL AB. At present, web database and database driven web site design are very attractive technology in the world. MySQL is the most widely used relational database for web application development with the server side scripting language PHP.

The implementation cost of MySQL is far lower than that of commercial database system. The software license fee is free of cost. Hardware requirement is very minimal. Maintenance cost is marginal. MySQL database can be implemented for medium scale enterprise solution by the 30% of the commercial database cost.

Considering the above, Bangladeshi enterprises especially Government Organizations, research organizations, educational institutions, trading companies, service oriented enterprise, production oriented industries etc. can easily select

MySQL database management system as a cost cutting solution to manage their business information if the requirement is fulfilled.

The private sector corporate can use MySQL in their business solutions such as inventory management, trading, production monitoring, supply and stock management, sale maintenance etc. instead of high cost proprietary RDBMS. MySQL can also be used in the NGOs in their management, operational and research activities for the purpose planning, monitoring and decision making purpose.

The Government of Bangladesh may implement MySQL database system in the Ministries, Directories, Departments and Institutions for accounting, personal management, payroll, inventory, monitoring & evaluation, tendering process, recruitment process, election process, people's ID card, passport, driving license, land inventory, tax and revenue sector, roads and transport sector inventory, hospitals automation project and research project etc. as cost reducing solutions. MySQL can also be used effectively in the e-governance projects with low cost infrastructure that will be more sustainable than the implementation of the other costly database management systems because of easy maintenance with minimum cost. This open source technology i.e. MySQL RDBMS helps the Government to implement a sustainable e-governance strategy in the country with minimum expenses of foreign currency which will reduce the wastage of government resources, increase the efficiency and transparency in the Government activities and so on. Ultimately, it will effect on the development process of the country dramatically.

Similarly educational institutions of Bangladesh such as schools, collages and universities can implement MySQL RDBMS in their operational sections such as

student information system, library system, payment system, accounting and personal management system etc. as a cost cutting solutions. In addition to that MySQL can also be used as a teaching tool in the academic purpose as well as research purposes which will be much more effective for implementation of that type of cost reducing technology in the private and public sector of the country.



## REFERENCES

- [1] Batini, C., Ceri, S., Navathe, S. B. 1992 Conceptual Database Design, “*An Entity-Relationship Approach*”
- [2] Baumgartel, P. 2002 "*Oracle Replication: An Introduction*", [www.nyoug.org/200212baumgartel.pdf](http://www.nyoug.org/200212baumgartel.pdf)
- [3] Cox, J. 2004 “*Open Source Database improvement Grow*”, Network World Research Center, <http://www.networkworld.com> (accessed on May 06, 2006)
- [4] Devx, 2004, "*PostgreSQL vs MySQL vs Commercial Databases: It's All About What You Need*"<http://www.devx.com/dbzone/Article/20743> (accessed on April 04, 2006)
- [5] Fermi, 2004 “*MySQL General Information*”, Fermi National Accelerator Laboratory, <http://www-css.fnal.gov/dsg/external/freeware/mysql-vs-pgsql.html> (accessed on March 27, 2006)
- [6] Gennick J., Dieter M. C., Linker G. 2002, Oracle 8i DBA Bible
- [7] Gogo, 2005 *MySQL Disaster Recovery*, [http://code.gogo.co.nz/dev-2-dev /mysql\\_disaster\\_recovery.html](http://code.gogo.co.nz/dev-2-dev/mysql_disaster_recovery.html)
- [8] Haff, G., CRM Daily, <http://www.crm-daily.com> (accessed on December 25, 2005).
- [9] Jackson J. 2005 Government Computer News:GCN.com, “*Beyond the Database Giants*”, [http://www.gcn/24\\_8/tech-reports/35512-1.html](http://www.gcn/24_8/tech-reports/35512-1.html) (accessed on December 28, 2005).
- [10] Matthew,H., eWEEK.com, “*Open-Source Databases Hike Enterprise Appeal*”, <http://eweek.com/>
- [11] MySQL, 1995 "*The world's most popular open source database* ", MySQL AB, <http://www.mysql.com>
- [12] Niccolai, J. 2001 Info World, “*OpenWorld:Oracle touts database Clustering Software*” <http://www.infoworld.com/articles/hn/xml/011204hnclusers.html> (accessed on may 07, 2006)
- [13] Open Source Initiative (OSI), <http://www.opensource.org/docs/definition.php>
- [14] Oracle Connectivity in .NET, <http://www.c-sharpcorner.com/Code/2003/Dec/OracleConnectivityInNet.asp>
- [15] Oracle Corporation 1999, “*Oracle 8i Replication Release 2 (8.1.6)*”,

[http://download-west.oracle.com/docs/cd/A87860\\_01/doc/index.htm](http://download-west.oracle.com/docs/cd/A87860_01/doc/index.htm)

- [16] ODDL , 1995 “*Oracle Database Documentation Library*”, Oracle Corporation, <http://www.stanford.edu/dept/itss/docs/oracle/10g/index.htm>
- [17] OMWM 2002, Oracle Migration Workbench Reference Guide for MySQL 3.22, 3.23 Migrations, Release 9.2.0 for Microsoft Windows 98 /2000, Microsoft Windows NT and Red Hat Linux 6.2, Part Number A97249 -01, Oracle Corporation., <http://mis057.mis.udel.edu/Oracle9i/win.920/a97249/ch1.htm#1026332>
- [18] OraReplFaq, 2002 Oracle Replication FAQ, “*KeepTool, a tool for Oracle Databases*” Oracle Faq's, <http://www.oraFAQ.com/faqrepl.htm#TopOfFile>
- [19] Cyran, M. 2003 “*Oracle Database Concepts, 10g Release 1 (10.1), Part Number B10743-01*”, Oracle Corporation 1993, 2003 <http://www.stanford.edu/dept/itss/docs/oracle/10g/server.101/b10743/schema.htm#CNCPT608>
- [20] Paul N. R., 2005, OSNews.com, “*Selecting a Mid-market Database*”, <http://osnews.com/story.-php> (accessed on December 12, 2005).
- [21] Petri G., TUSC, “*A Comparison of Oracle and MySQL*”, [http://www.ioug.org/client\\_files/members/select\\_pdf/05q1/003\\_OracleMySQL.pdf](http://www.ioug.org/client_files/members/select_pdf/05q1/003_OracleMySQL.pdf)
- [22] Pin-outs, 2005, “*Transaction Information*” <http://www.pin-outs.com/transaction.html> (accessed on December 29, 2005)
- [23] Refman5.0, 2006 “*MySQL Reference Manual 5.0*”, MySQL AB <http://dev.mysql.com/doc/refman/5.0/en.a4.pdf>
- [24] Sheppard, S. 2006 “*Command line reference for Oracle*”, SS64.com <http://www.ss64.com/ora/index>
- [25] Silberschatz A., Korth F. H., Sudarshan S. 1997, “*Database System Concepts*”, Third Edition
- [26] SSIL, 1992 “*Introduction to Oracle:SQL, SQL\*PLUS and PL/SQL*”, SQL Star International Limited, India
- [25] Strendell T. 2003, “*Open Source Database Systems: Systems Study, Performance and Scalability*”, Helsinki
- [27] Wiedman, B. 2006 “*Database Security (Common-sense Principles)*”. GovernmentSecurity.org, <http://governmentsecurity.org/articles/databsesecuritycommensense.html> (accessed on January 01, 2006)