DATABASE MANAGEMENT SYSTEM PERFORMANCE ANALYSIS AND COMPARISON

Margesh Naik
B.E, Veer Narmad South Gujarat University, India, 2008

PROJECT

Submitted in partial satisfaction of
the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

at

CALIFORNIA STATE UNIVERSITY, SACRAMENTO

SPRING
2011

DATABASE MANAGEMENT SYSTEM PERFORMANCE ANALYSIS AND COMPARISON

A Project

by

Margesh Naik

Approved by:

______________________________________, Committee Chair
Mary J Lee, Ph.D

______________________________________, Second Reader
Senad Busovaca, Ph.D

____
Date

Student: Margesh Naik

I certify that this student has met the requirements for format contained in the University format manual, and that this Project is suitable for shelving in the Library and credit is to be awarded for the Project.

________________________________, Graduate Coordinator __________________
Nik Faroughi, Ph.D Date

Department of Computer Science

Abstract

of

DATABASE MANAGEMENT SYSTEM PERFORMANCE ANALYSIS AND COMPARISON

by

Margesh Naik

The intent of this project is to measure and analyze the performance of two widely used Database Management System technologies: SQL and Oracle, and to determine which is best suited for what types of query executions.

The main goal of the project is to produce a guide for application developers who are developing the applications where database is a considerable part and where data retrieval timing is an important factor. To achieve this goal both DBMS technologies will be tested under various conditions and the results will be analyzed.

The measurement will be done for different types of queries for both DBMS(s) by maintaining the same system environment. Measurement factors will be query execution time and memory space requirements. The constraints will be maintaining the identical hardware configurations and system load, i.e., same number of records in both DBMS(s).

_______________________________, Committee Chair
Mary J Lee, Ph.D

____
Date

ACKNOWLEDGMENTS

I would like to extend my gratitude to my project advisor Dr. Mary J Lee for guiding me throughout this project and helping me complete this project successfully. In addition, I would like to thank The Department of Computer Science at California State University for extending this opportunity for me to pursue this program and guiding me all the way to become a successful student.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Choosing a database management system (DBMS) for any application is an important decision for developers because their applications need to run efficiently. This decision should be based on the information that compares different database management systems based on various performance factors. Different applications have unique characteristics in their functionalities. Application developers need to choose appropriate technologies for the development of their product. This project will help them make a decision for the DBMS selection. The project compares two different database management systems based on some performance factors under various constraints.

The main purpose for this project is to analyze and compare the performance of two widely used database management systems. One of them is SQL and the other is Oracle. Both technologies have been implemented under various test conditions and their behaviors analyzed.

The measurement factors are Memory Requirements and Execution time for performing various types of queries. These measurements have been taken on same database schema for the both technologies. The same sets of queries have been analyzed by gradually increasing the number of records in the database. Performance impact of size increment have also been measured and compared. Performance data for both technologies are displayed on line charts for ease of comparison.

The targeted end users of this paper are application developers who work in development environments where a database plays an important role as a backend and for whom the performance is an important aspect. They can review the data of project results as well as the conclusion to determine which database management system to use for their applications.

Chapter 2

BACKGROUND

During the initial stage of this project, a literature review was done to get an understanding of SQL and Oracle. Various papers were reviewed [1], [6], [7], [9], [10], [11] to identify the performance factors to be analyzed. In the beginning, measurement factors such as Failure Behavior, Failure Recovery, Query Execution Time, Portability, Scalability, Memory Usage, Transaction Support, Multiuser Support and some others were proposed. After considering them there was a need for elimination of some of the proposed factors to make the scope of the project feasible. Review of the literature helped to finalize two performance factors: Query Execution Time and Memory Requirements for study in this project. These two factors have been considered by focusing on the needs of industry of efficient and cost effective applications.

This literature review helped in the understanding of the methods for measuring and comparing the performance of both systems. It also helped to get the understanding of enforcing constraints, such as, identical hardware configurations and system environments. Based on the literature review, decision on the design of the database schema, the numbers of records in the database, and the query types to be analyzed were made.

The deciding factors in this project have been efficient time and memory in database query applications.

Chapter 3

SQL DATABASE MANAGEMENT SYSTEM

3.1 SQL Background

SQL often referred to as Structured Query Language, is a database computer language designed for managing data in relational database management systems. The scope of SQL includes query, data insert, delete, update, schema creation and modification, and data access control.

The SQL language is sub-divided into several language elements, including:

i. Clauses: They are components of statements and queries.

ii. Expressions: They can produce either scalar values or tables consisting of columns and rows of data.

iii. Predicates: They are used to limit the effects of statements and queries, or to change program flow.

iv. Queries: They retrieve the data based on specific criteria. This is the most important element of SQL.

v. Statements: They may have a persistent effect on schemas and data, or which may control transactions or program flow.

3.2 Queries

For this project we will focus on Queries.

Queries

The most common operation in SQL is the query which is performed with the declarative SELECT statement. SELECT retrieves data from one or more tables. Standard SELECT statements have no persistent effects on the database. Some non standard implementations of SELECT can have persistent effects, such as the SELECT INTO syntax that exists in some databases. Query allows the user to describe desired data, leaving the database management system responsible for planning, optimizing and performing the physical operations necessary to produce that result.

A query includes a list of columns to be included in the final result following the SELECT keyword. An asterisk can also be used to specify that the query should return all columns of the table. The optional keywords and clauses for SELECT statement in SQL include:

i. The FROM clause which indicates the table from which data is to be retrieved.
ii. The WHERE clause includes comparison predicate, which restricts the rows returned by the query.
iii. The GROUP BY clause is used to project rows having common values into a smaller set of rows.

iv. The HAVING clause includes a predicate used to filter rows resulting from the GROUP BY clause.

v. The ORDER BY clause identifies which columns are used to sort the resulting data, and in which direction they should be stored. Without ORDER BY clause, the order of rows returned by an SQL is undefined.

3.3 Advantages of SQL

Stored Procedures

The main advantage of using SQL Server is the use of stored procedures. Stored procedures are lines of code that are called by the application. They are placed on the server, and they are pre-compiled for quicker response times. Stored procedures require the knowledge of SQL Server syntax, which is called T-SQL. The use of stored procedures also centralizes code, so troubleshooting bad database requests can be observed by a database administrator.

Scalability

The term scalability is used to describe the ability to grow when the business becomes bigger. Microsoft SQL server is quick for large and small businesses, so as the business grows, the SQL Server can handle the new volume of database requests. SQL server can handle millions of records and transactions.

Security

Security is a major issue for any site. SQL Server allows the administrator to grant access or deny access for users. The SQL Server has a specific section of the application where users are added to the permissions. SQL Server allows administrators to specify which tables and stored procedures users are able to access and query. This limits what records and user information can be queried.

Transaction Logs

Transaction logs are objects on the SQL Server that record the retrieval, update and deletion of records. There are two reasons to keep transaction logs. The first is for rollback procedures. This process is used for accidental updates or deletions. The administrator can return records back to the original data by using transaction logs. Secondly, the transaction logs can be used for security purposes. If the administrator suspects a breach of security, he can watch the transaction logs for any type of data retrieval and identify the severity of the breach.

Automatic Backup

SQL Server has an automatic backup option. The SQL Server automatically saves a copy of the database and the transaction logs on another hard drive or media like a CD-

ROM or a DVD. SQL Server also has procedures that allow the administrator to quickly restore a database when data is lost or corrupted, or the server has a hard drive crash.

3.4 Problems of SQL

Some of the problems that may occur [15] with SQL are described below.

i. The language makes it too easy to do a Cartesian join, which results in "run-away" result sets when WHERE clauses are mistyped.

ii. The grammar of SQL is perhaps unnecessarily complex, borrowing a COBOL (Common Business Oriented Language) like keyword approach, when a function-influenced syntax could result in more re-use of fewer grammar and syntax rules.

iii. Implementations are inconsistent with the standard and, usually, incompatible between vendors. In particular date and time syntax, string concatenation, NULLs and comparison case sensitivity vary from vendor to vendor.

iv. Inability to implement Recursive Processing: One of the major drawbacks of SQL is its incapability to execute recursive processing. Recursive processing is a type of computer function in which one of the steps of a program reruns the entire program. SQL lacks looping constructs that are common in other high-level programming languages, such as for and while loops. It cannot repeat actions and there is no way to define repetitive looping constructs in SQL.

Chapter 4

ORACLE DATABASE MANAGEMENT SYSTEM

4.1 ORACLE Background

The Oracle Database [16], [17], [18], [19] is an object-relational database management system (ORDBMS). The Oracle RDBMS stores data logically in the form of table-spaces and segments, such as Data Segments, Index Segments, etc. Segments, in turn, comprise one or more extents. Extents are comprised of groups of contiguous data blocks. Data blocks create the basic units of data storage.

Oracle database management tracks its computer storage with the help of information stored in the SYSTEM table-space. The SYSTEM table-space contains the data dictionary and often indexes and clusters. A data dictionary consists of a special collection of tables that contains information about all user-objects in the database.

Each Oracle instance uses a system Global Area or SGA – a shared memory area – to store its data and control information. Oracle instance allocates itself an SGA when it starts and de-allocated it at shut-down time. The information in the SGA consists of the following elements, each of which has a fixed size, established at instance startup:

i. The redo log buffer: It stores redo entries—a log of changes made to the database. The instance writes redo log buffers to the redo log as quickly and efficiently as possible. The redo log aids in instance recovery in the event of a system failure.

ii. The shared pool: This area of the SGA stores shared-memory structures such as shared SQL areas in the library cache and internal information in the data dictionary. An insufficient amount of memory allocated to the shared pool can cause performance degradation.

4.2 ORACLE Advantages

Oracle is used for almost all large applications and one of the main applications in which Oracle takes its major presence is banking. Oracle offers a powerful combination of technology and comprehensive, pre-integrated business applications, including key functionality built specifically for banks. Oracle frequently upgrades and releases new products into market.

Oracle is a database that responds very well with excellent performance in demanding environments. Oracle is a major database which along with its added features passes the ACID (A-Atomicity, C-Consistency, I-Isolation, D-Durability) test:

i. Atomicity: The the result of a transaction's execution are either all committed or all rolled back.

ii. Consistency: The database is transformed from one valid state to another valid state. Illegal transactions aren't allowed and, if an integrity constraint can't be satisfied then the transaction is rolled back.

iii. Isolation: The results of a transaction are invisible to other transactions until the transaction is complete, thus, increasing the security on data.

iv. Durability: Once committed, the results of a transaction are permanent and survive future system and media failures and thus ensuring maintenance and protection of data.

4.3 ORACLE Disadvantages

Some of the limitations of Oracle are [17], [18]:

i. Connection Takes More Memory: Each database connection takes 2-4 MB of memory on the host machine where as for SQL the amount of memory is less than $1/10^{th}$ of that.

ii. Incompatibility and Complexity: One of the major disadvantages of Oracle is inconsistency and data incompatibility in the areas of time and date syntax, string concatenation and case sensitivity. The language is complex, with a keyword approach similar in structure to COBOL (Common Business Oriented Language), with fewer syntax and grammar rules.

iii. Oracle doesn’t support Update within Join: With Oracle you can’t run a query like this:

“ update history set a.teller_id = h.teller_id

from account a, history h

where a.account_id = h.account_id ”

In Oracle, you need to use convoluted sub-select format, to achieve the same result.

iv. There is no database dump utility: There is no “block_dump” style backup program in Oracle. Space reclamation is also poor. In Oracle, tables grow even after deletion of row. A query might scan empty blocks of deleted rows.

Chapter 5

PROJECT PROBLEM AND SOLUTION APPROACH

5.1 Purpose and Goal

For any application which has database as a backend, data retrieval time and memory usage are crucial performance factors. So choosing a right DBMS technology is an important decision for any application developer. This project is designed to help users make the right decision. Behavior of both SQL and Oracle will be analyzed under various test conditions. Measuring performance of DBMS for different types of query sets can help determine which one is better for what types of queries. Also, there will be measurements for memory usages.

To achieve this goal, there will be two systems with SQL and Oracle installed on each computer. The database schema will be designed for both DBMS. Selected types of queries will be executed and their performance in terms of time and memory usage will be analyzed. Analysis will be repeated every time when some fixed numbers of records have been inserted into both databases. There are also some constraints that will be applied for accurate measurements.

5.2 Project Requirements and Constraints

The requirement for this project will be two working computers having same hardware configuration and same operating system environment. In comparing two different systems, the main constraint of this project is to maintain the same resources and environment for both of them. Two database schemas, one for Oracle and other one for SQL were created.

5.3 Database Schema

The implementation phase started with designing database schema for analysis. As we are comparing two different technologies it is required that both of them have the same database schema. By that we can provide a consistent base for both DBMSs. Here I have designed the database of bank account. The schema for the same is given in the figure below.

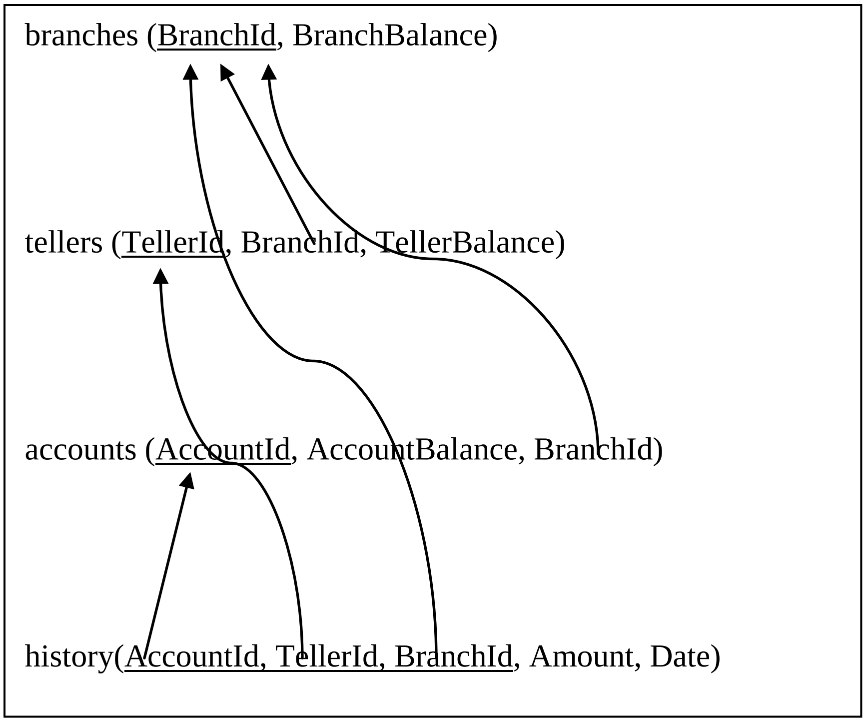

Figure 1 Database Schema Design

The above diagram shows the Database Schema design for both DBMS. Names on the left of parentheses are table names. The names within parentheses are columns of those tables. Underlines specify primary key of the table. Each arrow is for the foreign key which is being referenced from parent table. We can see there are four tables in this database.

i. BRANCHES: This table contains branch specific details. It has two columns: “BranchId” which is a primary key of the table and contains unique identification number for specific branch. Another is “BranchBalance” which specifies total amount particular branch is holding.

ii. TELLERS: This table is having details for a specific teller. It has three columns: “TellerId” which is a primary key for the table and contains unique identification number for a teller. Others are “BranchId” and “TellerBalance”. BranchId is the foreign key referenced from table Branches. It specifies the branch that a teller works for. “TellerBalance” is the total amount the teller has in his/her account.

iii. ACCOUNTS: This table contains account related information. It has three columns: “AccountId”, “AccountBalance” and “BranchId”. “AccountId” is a primary key for the table. It contains unique identification number of any account. “AccountBalance” is the current balance of the account for that “AccountId”. “BranchId” is the foreign key referencing the table Branches. It describes the branch under which the account exists.

iv. HISTORY: This table describes transaction that has occurred in the past stating the amount and date of its occurrence. It is having “AccountID”, “TellerId” and “BranchId” as its primary key. Together they all describe transaction that has occurred in the account of which branch and under which teller. Other two are “Amount” and “Date”. They specify amount of the transaction with the date on which they occurred. It should be noted that together “BranchId”, “AccountId” and “TellerId” are primary key. But individually each of them are foreign key referencing tables Branches, Accounts and Tellers accordingly.

5.4 ER-Diagram

Next step in implementation was to draw ER-Diagram based upon developed database schema. Figure drawn below shows the ER-Diagram for the same.

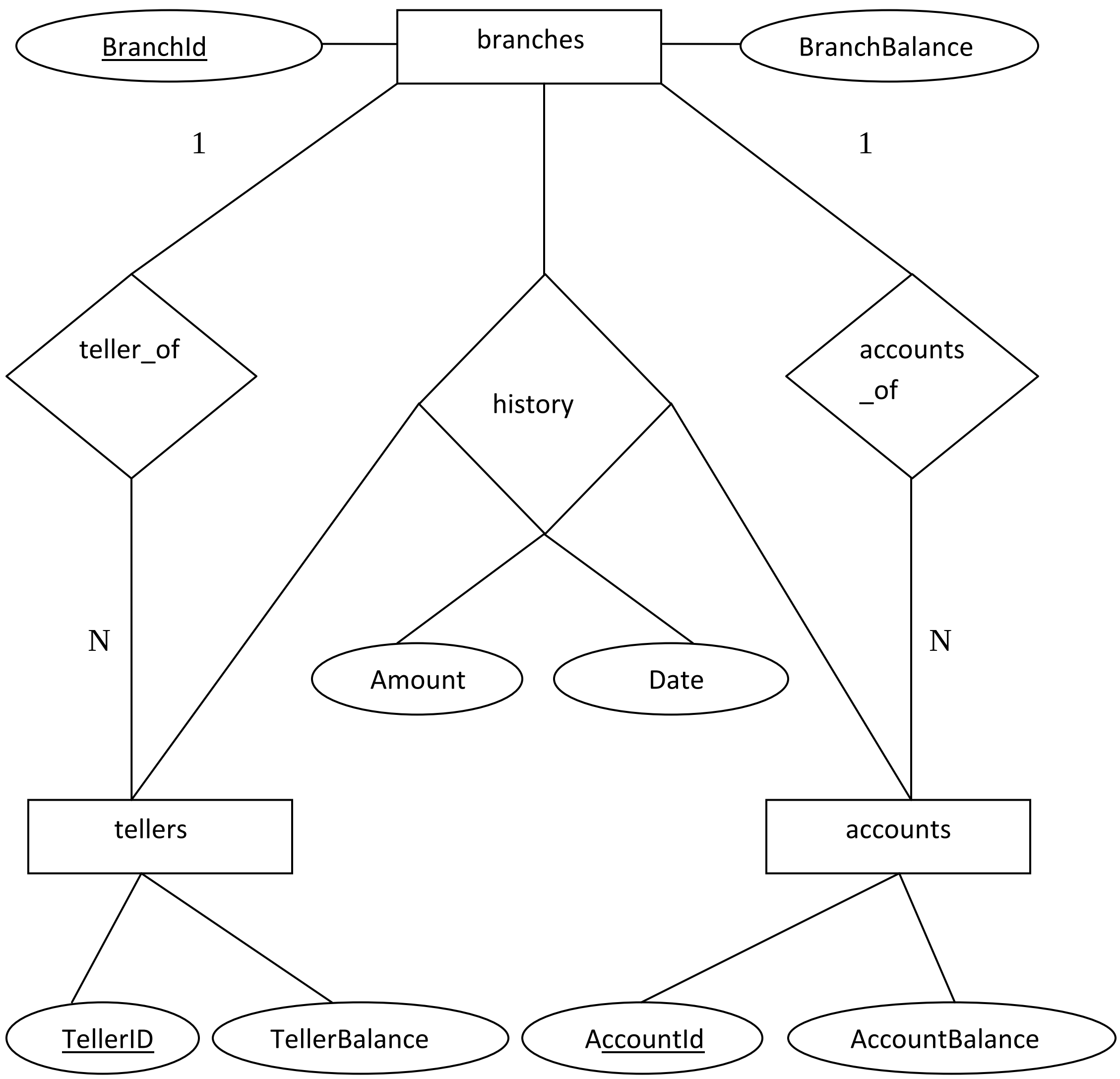

Figure 2: Database ER Diagram

In above figure, each rectangle stands for entities. Texts in them are the table names that particular square is representing. Ovals represent attributes of entities. Diamonds stands for relation between entities. Attribute names which are underlined represent primary key. Lines represent relation between connected objects. For example attributes “BranchId” and “BranchBalance” are linked to entity “Branches”. Relation “History” is ternary relation as it is connected to three different entities which are “Branches”, “Accounts” and “Tellers”. Whereas other two relations “Accounts_of” and “Teller_of” are binary relations as they are relating two entities.

The numbering on the connecting lines shows cardinality of relation. For relation between “Branches” and “Accounts”, the cardinality is 1:N. That means one branch can have many accounts but a single account can be under one branch only. Same concept applies to relation between “Branches” and “Tellers” also.

5.5 DBMS Installation and Database Creation

After designing database, the next task was to install DBMSs in two different computers. Both computers have the same hardware configuration. Oracle and SQL have been installed on them. For Oracle, the software is “Oracle Database 10g Express Edition” and for SQL, “SQL Server 2008 R2” has been installed. It was obtained for free from their vendor’s website.

After installation, the next task was to start creating a database on each system. When that was done, the memory space required for each database on both DBMSs has been measured. After successfully creating databases, data insertion was started and different types of queries have been implemented on them.

5.6 Finalizing Queries for Analysis

Before Implementation, one task that needs to be done is to choose types of queries for analysis. The research [11] that I did in the beginning of the project helped me to finalize some types of queries important for performance analysis. Among different types, five types of queries were selected. They are:

i. Insert Query: There are four Insert queries for both databases, one query for each of the four tables. They are used to insert records into the database.

ii. Select Query: Same as Insert, there are four Select queries for both databases, one for each of the four tables. It will display all the records of the table from the database.

iii. Join Query: There is one Join query that links all four tables of database and displays the result.

iv. Update Query: There are four Update queries, one for each of the four tables. Update Query can change any single row anywhere in database that satisfies given condition.

v. Delete Query: Again, four queries. One for each of the four tables. Delete query removes the entire row that satisfies the given condition.

5.7 Query Execution and Analysis

After the database was implemented and the types of queries finalized performance analysis was conducted. In this phase, one row is inserted at a time into the table. When inserted row counts reaches fifty, performance analysis is done by executing all five types of queries mentioned in the previous section. After completing analysis, again records insertion continues. When row counts reaches hundred i.e. another fifty rows have been inserted, again query performance evaluation begins. So in this project, records have been inserted in blocks of fifty rows for each table. At every row count of fifty for each table i.e. two hundred per database, query performance analysis is performed. This analysis continues until the insertion of seven hundred rows per table. That means 2800 rows per database, which means total of 9100 records. There are two similar databases so the total analysis is for 5600 rows, i.e., 18,200 records.

When a query is executed, the results are checked for correctness. For each query, execution time or memory usages are measured depending upon query type. All the results of the measurements are described in next chapter.

Chapter 6

PROJECT RESULTS

This chapter discusses the results of the performance measurements that have been obtained during performance analysis.

6.1 Insert Query Results

As discussed in the previous chapter there are four different insert queries per database – one for each database table. Sample queries for inserting one row in each table are:

i. For Table “BRANCHES” : insert into branches(BranchId,BranchBalance) values('b0001',1000001)
ii. For Table “TELLERS”: insert into tellers(TellerId,BranchId,TellerBalance) values(‘t0100’,’b0100’,’10000100)
iii. For Table “ACCOUNTS”: insert into accounts(AccountId,AccountBalance,BranchId) values('a0700',100600,'b0700')
iv. For Table “HISTORY”: insert into history(AccountId,TellerId,BranchId,Amount,Date1) values ('a0700','t0700','b0700',1600,'02/Mar/11')

Below is a line chart describing Insert Query behavior of the both technologies as number of rows increases. Data of History table have been selected for the presentation purpose. Actual data of results are displayed in Appendix. There are considerable differences in execution time between SQL and Oracle. SQL takes more time to execute a query than Oracle. We can see that in the chart that for SQL, there are peaks in execution time due to additional load. At the same time, Oracle is able to absorb the additional load successfully without impacting the performance in most of the cases.

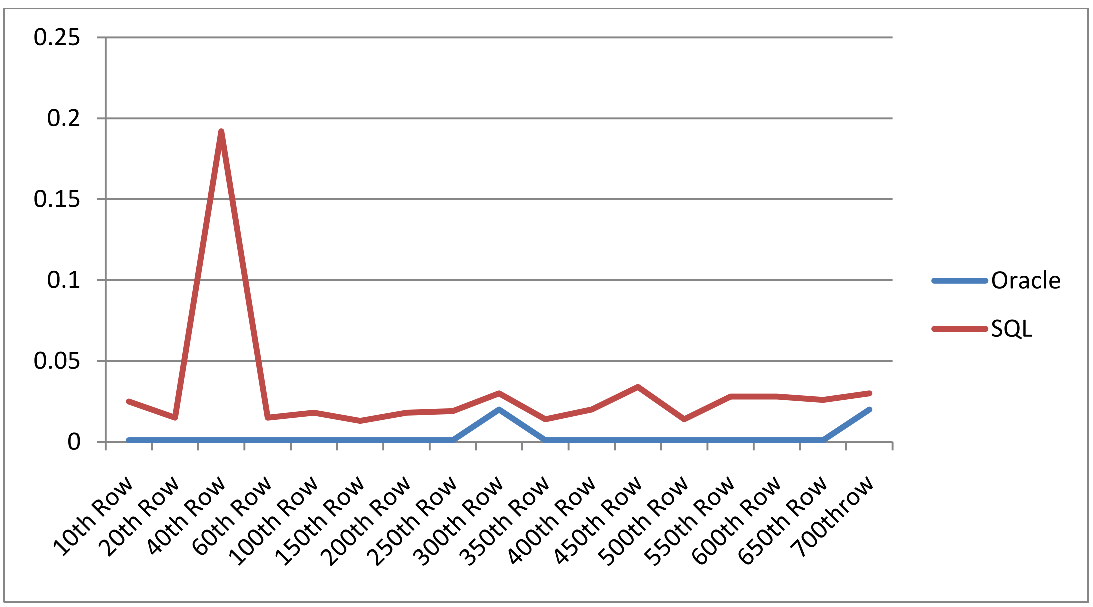

Figure 3: Insert Query Behavior Chart

6.2 Select Query Results

As discussed before Select query displays the results obtain from the table. Like Insert query, there are four select queries per database for each table. Sample select queries for each table are given below.

i. For Table “BRANCHES” : select * from branches
ii. For Table “TELLERS”: select * from tellers
iii. For Table “ACCOUNTS”: select * from accounts
iv. For Table “HISTORY”: select * from history

Below is a line chart showing Select query behavior of the both technologies as number of rows increases. Actual measurements are given in Appendix. Data of History table have been selected for the presentation purposes. From this chart, it can be seen that for Select query, Oracle takes considerably less time than SQL. Peaks in the lines show the performance impact due to additional loads on both systems.

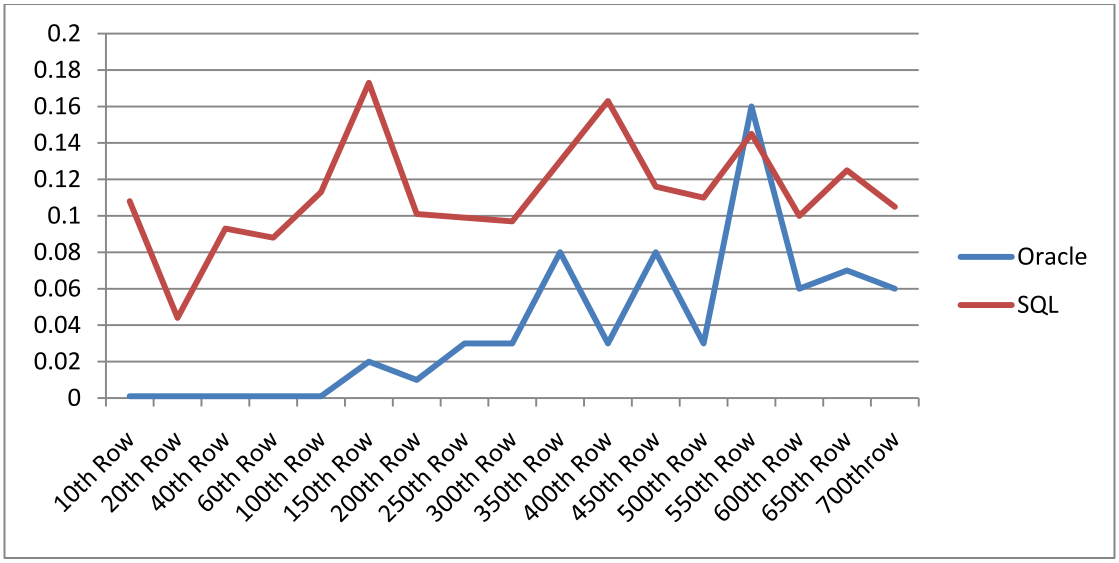

Figure 4: Select Query Behavior Chart

6.3 Join Query Results

Join query displays selected records from all four tables of a database which satisfies the condition in the given query. There is just a single join query per database in this project. Sample query for both DBMS is:

select * from accounts a, branches b, tellers t, history h where

h.AccountId=a.AccountId and h.TellerId=t.TellerId and h.BranchId=b.BranchId;

Below is a line chart describing Join query behavior of both technologies as number of rows increases. Data is provided in the Appendix. Data from History table is provided for presentation purposes. As before, Oracle requires less time than SQL for join queried.

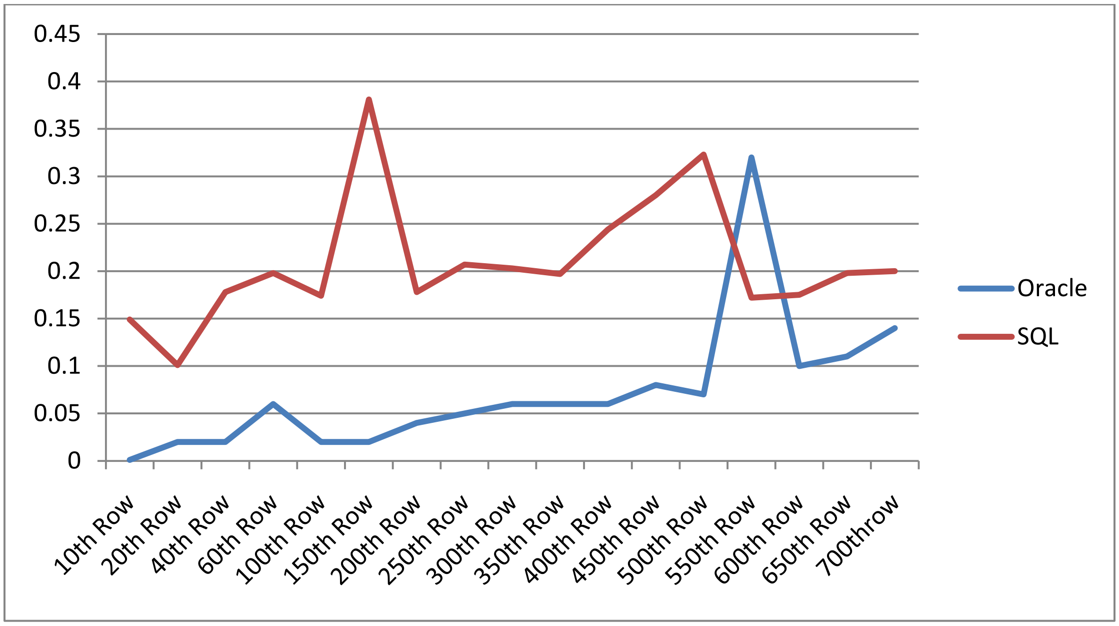

Figure 5: Join Query Behavior Chart

6.4 Update Query Results

Update query updates any single row of the table if the row satisfies the given condition of the query. There are four update queries for a database, separate for each table. Below are the sample queries for each table.

i. For Table “BRANCHES” : update bank.dbo.branches SET BranchBalance=99 where BranchId='b0650'

ii. For Table “TELLERS” : update bank.dbo.tellers SET TellerBalance=99 where TellerId='t0650'

iii. For Table “ACCOUNTS”: update bank.dbo.accounts set AccountBalance=99 where AccountId='a0650'

iv. For Table “HISTORY”: update bank.dbo.history set Amount=99 where AccountId='a0650' and TellerId='t0650' and BranchId='b0650'

Below is a line chart describing Update Query behavior of the both technologies as number of rows increases. Selected data of HISTORY table is presented below. It can be seen from the chart that the performance of the both systems are almost same in normal situations. When there is an additional load on the system, performance of SQL is impacted more negatively than for Oracle in some cases. But still the overall performance is same.

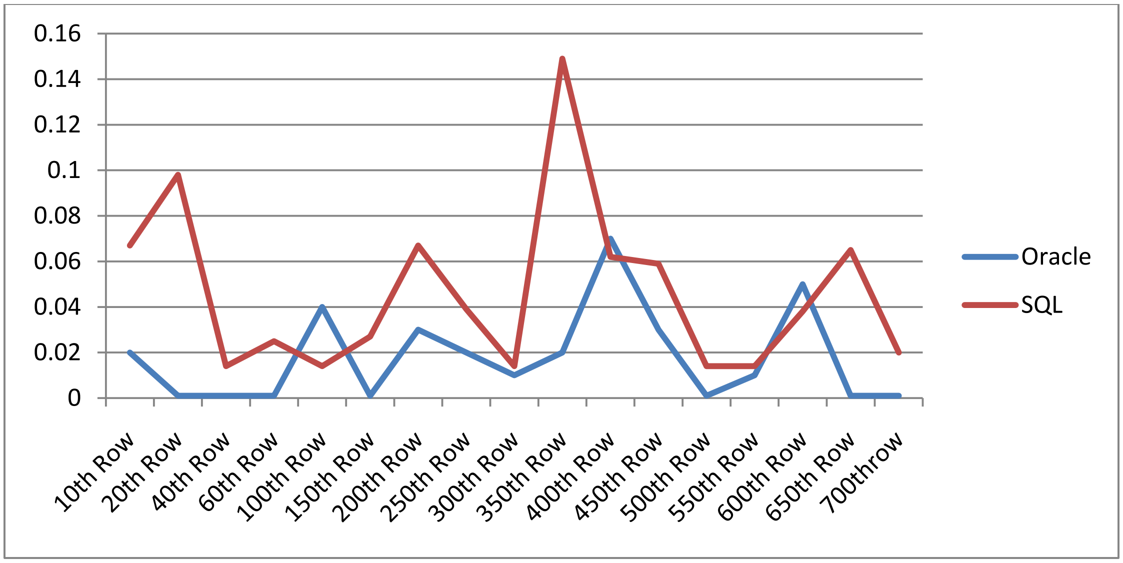

Figure 6: Update Query Behavior Chart

6.5 Delete Query Results

Delete query removes single row which satisfies enforced condition. There are four delete queries for four databases. The sample delete queries for each table are given below.

i. For table “BRANCHES”: delete from branches where BranchId='b0700'

ii. For table “TELLERS”: delete from tellers where TellerId='t0700'

iii. For table “ACCOUNTS”: delete from accounts where AccountId='a0700'

iv. For table “HISTORY”: delete from history where AccountId='a0700' and TellerId='t0700' and BranchId='b0700'

Below is a line chart describing Delete Query behavior of the both technologies as number of rows increases. Selected data of HISTORY table are presented. It can be seen that for Delete query, there is not much difference in performance for most of the cases. Peaks in the chart are performance impact of additional load. It can be seen at some point that Oracle is affected negatively by an increased load than SQL.

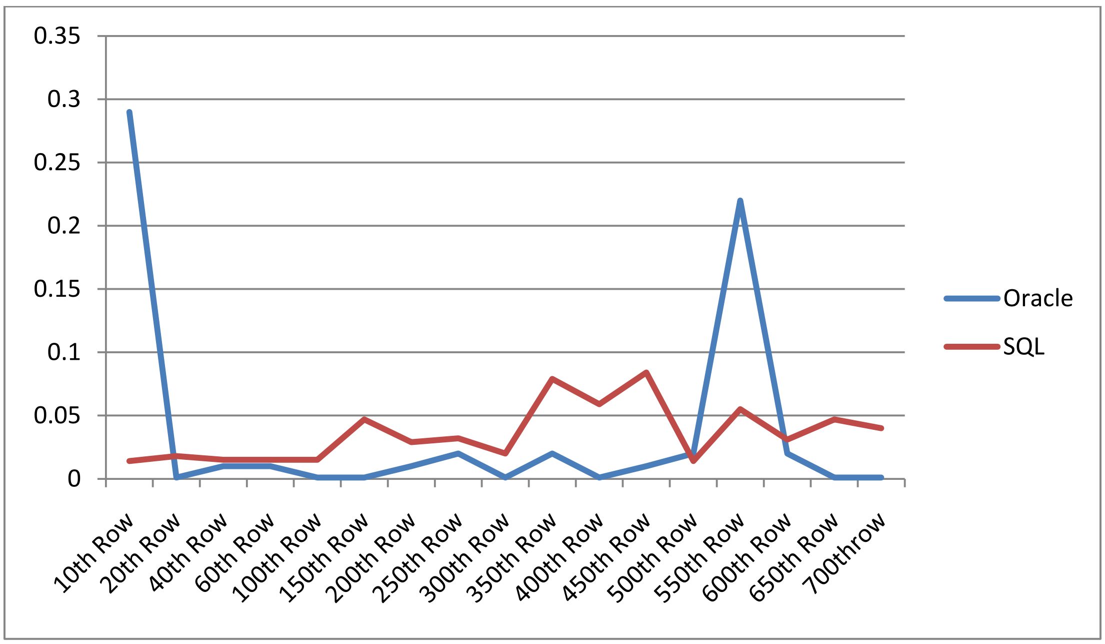

Figure 7: Delete Query Behavior Chart

6.6 Memory Usage

This section discusses the memory requirements for database tables as the number of rows increases. For SQL there is a built-in tool to measure the memory usage. For Oracle, in addition to the built-in tool, queries were used to get the memory usages. Data for actual memory usage for both SQL and Oracle are given in the tables in Appendix. Below is a chart describing Memory Usage of the both technologies as number of rows increases. Memory usage for Oracle is constant while for SQL it increases in proportion to number of rows in table. Still memory usage for Oracle is very high than for SQL.

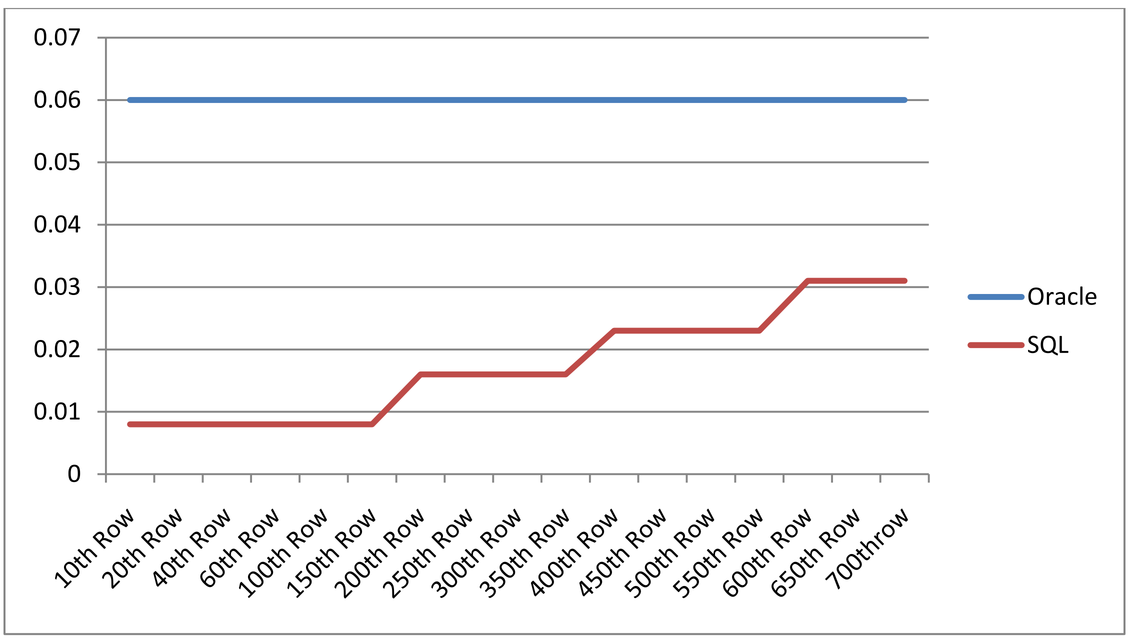

Figure 8: Memory Usage Behavior Chart

Chapter 7

COMPARISON OF SQL AND ORACLE

This chapter discusses the results obtained and analyzes which DBMS is appropriate for what types of queries and for memory requirements.

7.1 Insert Query Analysis

After reviewing tables in section 5.1 one can say that Oracle provides faster response time than SQL in case of Insert Query. For all four tables of database average query response time for Oracle is far better than SQL for the same Query. Table below describes average response time for all four tables for both DBMS. There is also a chart available on next page for the same data.

	Oracle	SQL
BRANCHES	7.7 ms	34.8 ms
TELLERS	2.1 ms	19.6 ms
ACCOUNTS	2.1 ms	25.6 ms
HISTORY	3.2 ms	29.0 ms

Table 1: Average Execution Time for Insert Query

Figure below specifies the column chart for the same data in the table above.

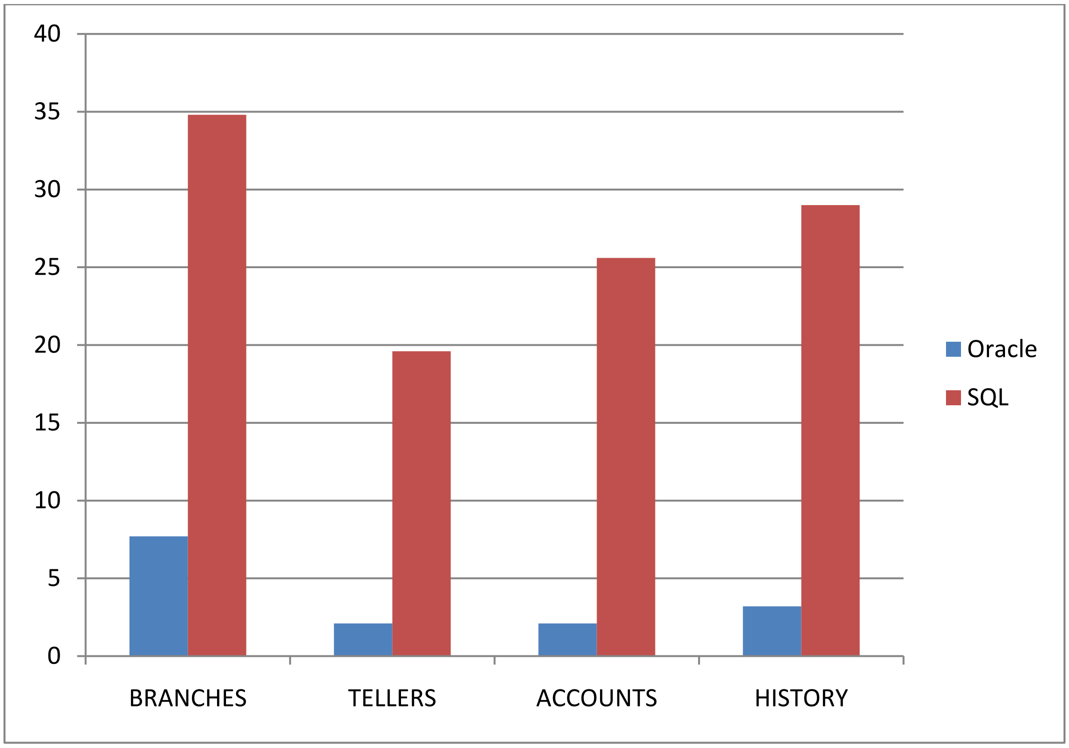

Figure 9: Average Execution Time for Insert Query

7.2 Select Query Analysis

Reviewing tables in section5.2 gives an idea that in select also Oracle gives better performance than SQL. Again for all four tables Oracle has given better performance than SQL. Table below compares the average response time for both technologies for same select query. There is also a chart available on the next page for the same data.

	Oracle	SQL
BRANCHES	22.4 ms	120.1 ms
TELLERS	27.9 ms	93.0 ms
ACCOUNTS	20.3 ms	101.1 ms
HISTORY	39.1 ms	112.3 ms

Table 2: Average Execution Time for Select Query

Figure below describes the column chart for the same data available in the above table.

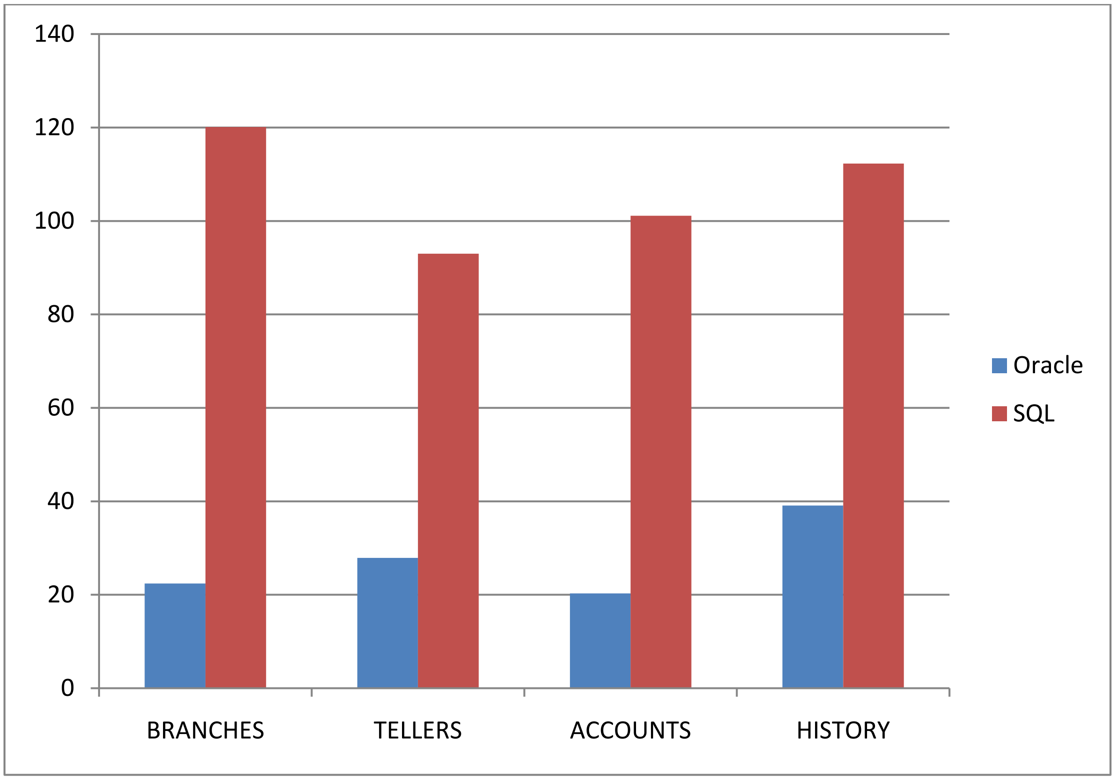

Figure 10: Average Execution Time for Select Query

7.3 Join Query Analysis

Again after going through section 6.3, it is proved that Oracle is better responsive than SQL in terms of time. The join query that has been used for analysis relates all four tables and returns records satisfying conditions. Below is a comparison of average response time for both Oracle and SQL. There also a chart available for the same data.

	Oracle	SQL
Response Time	72.4 ms	209.3 ms

Table 3: Average Execution Time for Join Query

Figure below describes the chart for the data available in the above table.

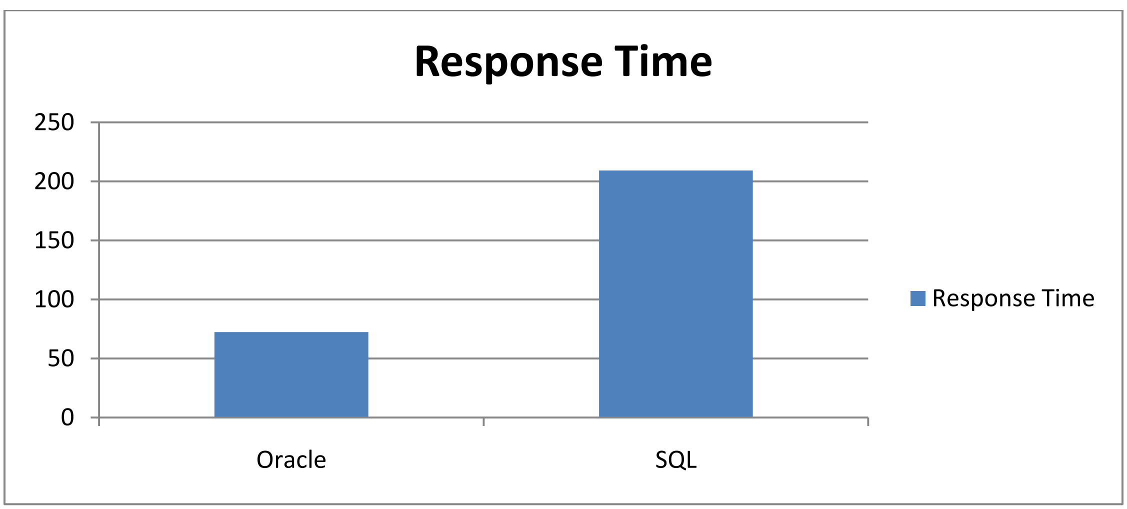

Figure 11: Average Execution Time for Join Query

7.4 Update Query Analysis

Unlike previous three types of queries for Update query, performance results for both DBMS were two sided. For one table SQL gave better performance where as for others Oracle won the race. Difference in performance is not as huge as we have seen before for all tables. Below is the table describing fact. There is also a chart available for the same data.

	Oracle	SQL
BRANCHES	16.6 ms	23.4 ms
TELLERS	47.1 ms	18.3 ms
ACCOUNTS	11.2 ms	27.0 ms
HISTORY	18.1 ms	35.6 ms

Table 4: Average Execution Time for Update Query

Figure below describes the column chart for the data available in the above table.

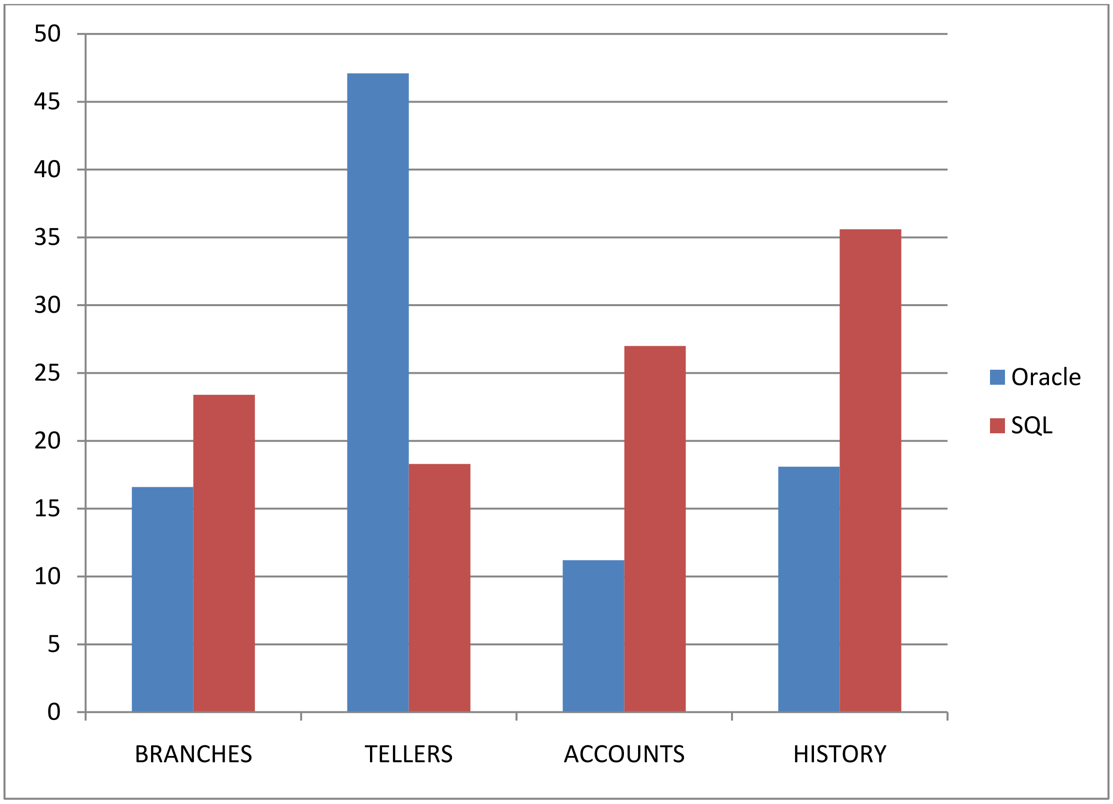

Figure 12: Average Execution Time for Update Query

7.5 Delete Query Analysis

For delete query also the results were mixed. For some tables Oracle was better while for some other SQL was impressive. And for some they both reacted almost same. The reason for such behavior is that in situation of applying an additional load, SQL performs better and Oracle is impacted considerable. In normal situation i.e. without any additional load, Oracle performs well. So it can be stated that for delete query, both technologies are almost same. Below is the table, comparing response time for both SQL and Oracle. There is also a chart available on next page for the same data.

	Oracle	SQL
BRANCHES	1.4 ms	32.2 ms
TELLERS	167.4 ms	28.5 ms
ACCOUNTS	34.4 ms	66.7 ms
HISTORY	37.4 ms	36.1 ms

Table 5: Average Execution Time for Delete Query

Figure below describes the column chart for the data available in the above table.

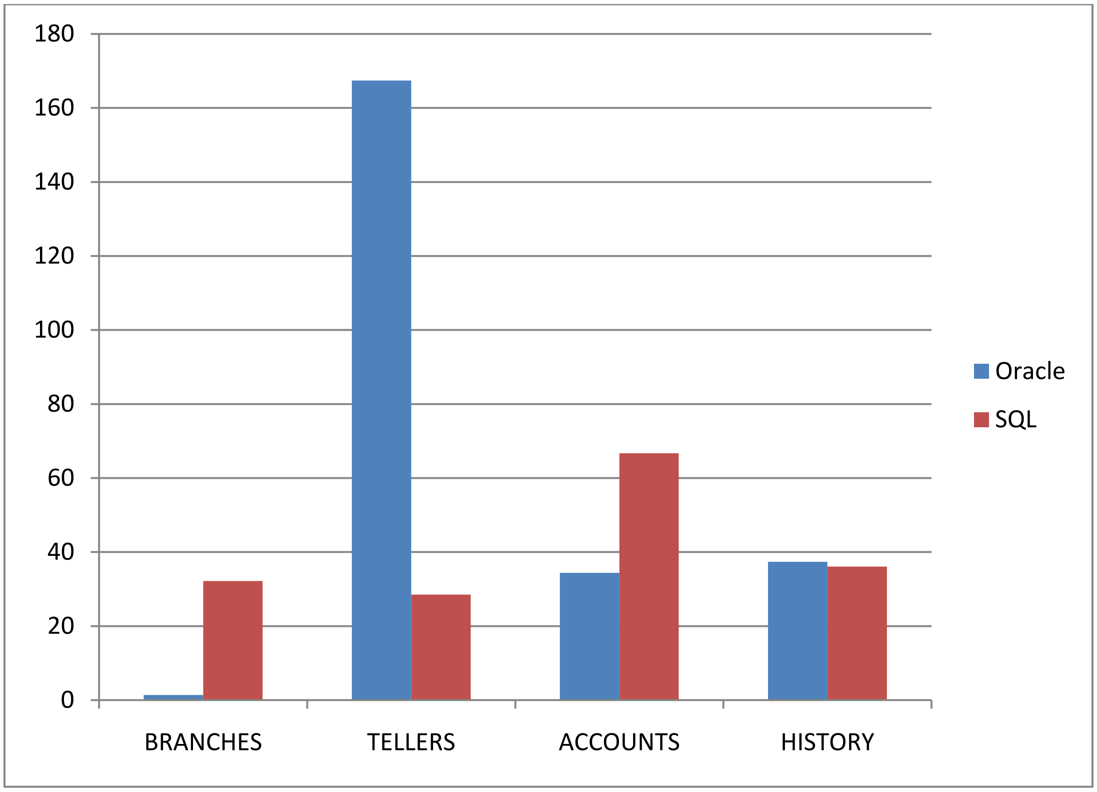

Figure 13: Average Execution Time for Delete Query

7.6 Memory Usage Analysis

For memory usage, simply the winner is SQL. One can easily tell that after going through all the tables of Appendix. For Oracle, every table requires space of 0.06 MB. While at the same time in SQL it begins with 0.008 MB and goes till .031 MB. That's still far less than what oracle is using. So it can be stated that SQL is more memory efficient. Below are a table and a chart which are comparing the average memory usage of both DBMS.

	Oracle	SQL
BRANCHES	0.06 MB	0.011 MB
TELLERS	0.06 MB	0.013 MB
ACCOUNTS	0.06 MB	0.013 MB
HISTORY	0.06 MB	0.017 MB

Table 6: Average Memory Usage for both DBMS

Figure below describes the column chart for the data available in the above table.

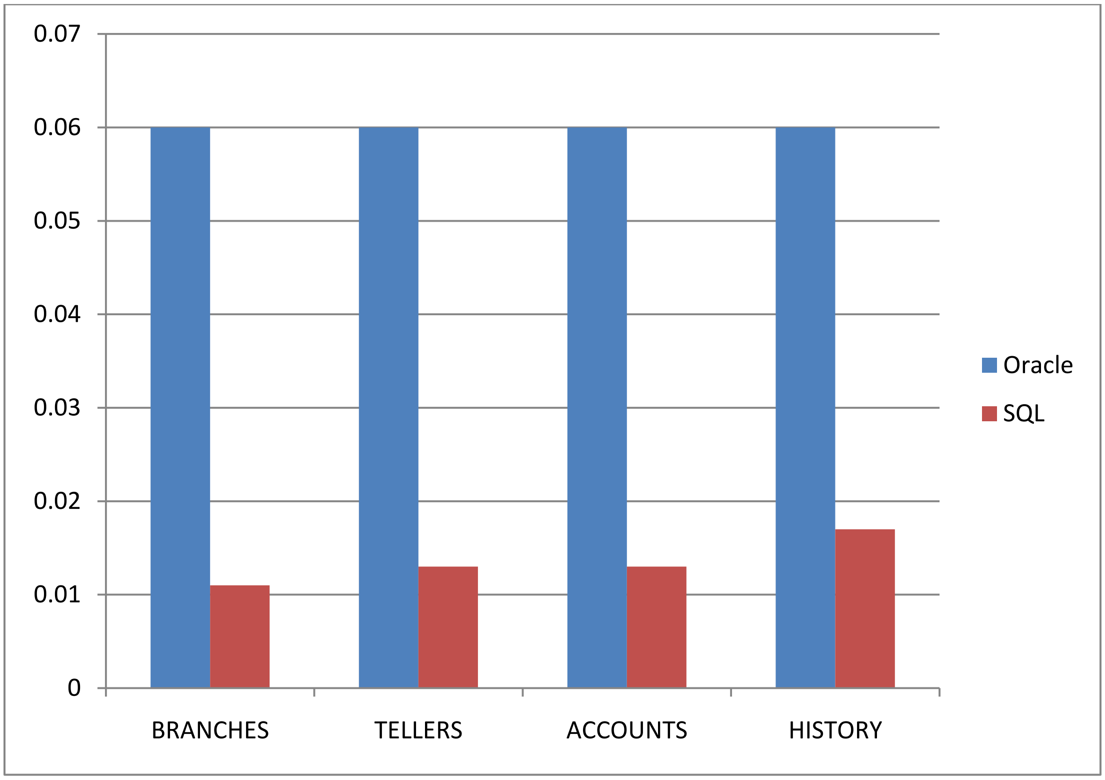

Figure 14: Average Memory Usage for both DBMS

Chapter 8

CONCLUSION

In analyzing the results given in the tables, it’s obvious that SQL is far better than Oracle in terms of memory requirements. On average Oracle tables require around five times more memory that what SQL tables need. So for the applications in which memory usage is a concern, SQL is a better choice over Oracle based on performance analysis for this project.

Both SQL and Oracle keep aside a chunk of memory for records in database tables. The reason that Oracle requires more memory than SQL is that for Oracle, the size of that chunk is 0.06 MB. So every time when there is a need for additional memory for database tables, Oracle occupies a new memory chunk of the same size. SQL also works the same way but the memory chunk that it asks for is much smaller, 0.008 MB which is almost 10 times less than what Oracle requires. So every time one memory slot fills up and new one is needed, Oracle will occupy 10 times more memory than SQL. The resulting size of an Oracle database will be much larger than that for a SQL database.

In the case of query execution time, though, results were balanced for some query type, Oracle performs well in most of the cases with huge timing differences. For Insert Select and Join queries there is no comparison between them as Oracle’s execution time is far better than SQL. For Delete query type SQL is somewhat better than Oracle for some tables. But still overall performance is nearly same. Same for Update query. The results were balanced. But if one tries to see the big picture, Oracle is better in terms of query execution time. So for time critical applications, Oracle can be a wise selection over SQL.

Another performance impact that one can observe in the line charts of chapter 6 is that whenever there is an additional load is added to the system, query execution time for SQL increases multiple times. Under similar conditions, Oracle also slows down but the impact is not as major as for SQL in most of the cases. So one can say that Oracle is pretty good absorbing unexpected loads than SQL. The average execution time for SQL is also considerably higher than for Oracle due to the slowdown of SQL in such cases. That is why one can see some huge differences between column charts of SQL and Oracle.

For this project, two DBMSs have been analyzed, but there are other products are available in the market. One can select them for same types of comparisons. Even one can carry out analysis based on other parameters like security, failure behavior, transaction property, and portability.

APPENDIX

Measurement Results of Performance Analysis

This section contains data obtained for each table for both databases. Line charts in Chapter 6 and column charts in Chapter 7 are based on these results only.

There are two columns in each table. First column describes the numbers of rows in the table for which the analysis has been made. Second column stands for actual measurements that have been made during the project. There are a total of sixteen measurements for each table. Measurements were taken after every block of fifty rows had been inserted into the database.

Tables shown below specify the results of both DBMS for Insert query.

Insert Query Results for table "BRANCHES" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.020 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.001 Seconds
200th Row	0.001 Seconds
250th Row	0.001 Seconds
300th Row	0.020 Seconds
350th Row	0.020 Seconds
400th Row	0.020 Seconds
450th Row	0.001 Seconds
500th Row	0.001 Seconds
550th Row	0.001 Seconds
600th Row	0.001 Seconds
650th Row	0.020 Seconds
700th Row	0.001 Seconds

Insert Query Results for table "BRANCHES" in SQL:

Number of Rows	Query Execution Time
20th Row	0.024 Seconds
40th Row	0.014 Seconds
60th Row	0.016 Seconds
100th Row	0.016 Seconds
150th Row	0.030 Seconds
200th Row	0.014 Seconds
250th Row	0.064 Seconds
300th Row	0.045 Seconds
350th Row	0.058 Seconds
400th Row	0.013 Seconds
450th Row	0.110 Seconds
500th Row	0.018 Seconds
550th Row	0.098 Seconds
600th Row	0.013 Seconds
650th Row	0.014 Seconds
700th Row	0.045 Seconds

Insert Query Results for table "TELLERS" in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.001 Seconds
40^{th} Row	0.001 Seconds
60^{th} Row	0.020 Seconds
100^{th} Row	0.001 Seconds
150^{th} Row	0.001 Seconds
200^{th} Row	0.001 Seconds
250^{th} Row	0.001 Seconds
300^{th} Row	0.001 Seconds
350^{th} Row	0.001 Seconds
400^{th} Row	0.001 Seconds
450^{th} Row	0.001 Seconds
500^{th} Row	0.001 Seconds
550^{th} Row	0.001 Seconds
600^{th} Row	0.001 Seconds
650^{th} Row	0.001 Seconds
700^{th} Row	0.001 Seconds

Insert Query Results for table "TELLERS" in SQL

Number of Rows	Query Execution Time
20^{th} Row	0.013 Seconds
40^{th} Row	0.021 Seconds
60^{th} Row	0.015 Seconds
100^{th} Row	0.019 Seconds
150^{th} Row	0.013 Seconds
200^{th} Row	0.015 Seconds
250^{th} Row	0.015 Seconds
300^{th} Row	0.027 Seconds
350^{th} Row	0.014 Seconds
400^{th} Row	0.013 Seconds
450^{th} Row	0.031 Seconds
500^{th} Row	0.014 Seconds
550^{th} Row	0.013 Seconds
600^{th} Row	0.035 Seconds
650^{th} Row	0.013 Seconds
700^{th} Row	0.025 Seconds

Insert Query Results for table "ACCOUNTS" in Oracle

Number of Rows	Query Execution Time
20^{th} Row	0.001 Seconds
40^{th} Row	0.001 Seconds
60^{th} Row	0.001 Seconds
100^{th} Row	0.001 Seconds
150^{th} Row	0.001 Seconds
200^{th} Row	0.001 Seconds
250^{th} Row	0.001 Seconds
300^{th} Row	0.001 Seconds
350^{th} Row	0.001 Seconds
400^{th} Row	0.001 Seconds
450^{th} Row	0.001 Seconds
500^{th} Row	0.001 Seconds
550^{th} Row	0.020 Seconds
600^{th} Row	0.001 Seconds
650^{th} Row	0.001 Seconds
700^{th} Row	0.001 Seconds

Insert Query Results for table "ACCOUNTS" in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.014 Seconds
40^{th} Row	0.023 Seconds
60^{th} Row	0.018 Seconds
100^{th} Row	0.016 Seconds
150^{th} Row	0.021 Seconds
200^{th} Row	0.016 Seconds
250^{th} Row	0.020 Seconds
300^{th} Row	0.057 Seconds
350^{th} Row	0.014 Seconds
400^{th} Row	0.013 Seconds
450^{th} Row	0.043 Seconds
500^{th} Row	0.037 Seconds
550^{th} Row	0.039 Seconds
600^{th} Row	0.042 Seconds
650^{th} Row	0.018 Seconds
700^{th} Row	0.025 Seconds

Insert Query Results for table "HISTORY" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.001 Seconds
200th Row	0.001 Seconds
250th Row	0.001 Seconds
300th Row	0.020 Seconds
350th Row	0.001 Seconds
400th Row	0.001 Seconds
450th Row	0.001 Seconds
500th Row	0.001 Seconds
550th Row	0.001 Seconds
600th Row	0.001 Seconds
650th Row	0.001 Seconds
700th Row	0.020 Seconds

Insert Query Results for table "HISTORY" in SQL:

Number of Rows	Query Execution Time
20th Row	0.015 Seconds
40th Row	0.192 Seconds
60th Row	0.015 Seconds
100th Row	0.018 Seconds
150th Row	0.013 Seconds
200th Row	0.018 Seconds
250th Row	0.019 Seconds
300th Row	0.030 Seconds
350th Row	0.014 Seconds
400th Row	0.020 Seconds
450th Row	0.034 Seconds
500th Row	0.014 Seconds
550th Row	0.028 Seconds
600th Row	0.028 Seconds
650th Row	0.026 Seconds
700th Row	0.030 Seconds

Tables given below specify the results of Select query for both DBMS.

Select Query Results for table “BRANCHES” in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.020 Seconds
200th Row	0.001 Seconds
250th Row	0.010 Seconds
300th Row	0.020 Seconds
350th Row	0.010 Seconds
400th Row	0.020 Seconds
450th Row	0.006 Seconds
500th Row	0.030 Seconds
550th Row	0.050 Seconds
600th Row	0.080 Seconds
650th Row	0.030 Seconds
700th Row	0.080 Seconds

Select Query Results for table “BRANCHES” in SQL:

Number of Rows	Query Execution Time
20th Row	0.072 Seconds
40th Row	0.094 Seconds
60th Row	0.105 Seconds
100th Row	0.137 Seconds
150th Row	0.145 Seconds
200th Row	0.139 Seconds
250th Row	0.084 Seconds
300th Row	0.150 Seconds
350th Row	0.262 Seconds
400th Row	0.165 Seconds
450th Row	0.099 Seconds
500th Row	0.114 Seconds
550th Row	0.075 Seconds
600th Row	0.080 Seconds
650th Row	0.111 Seconds
700th Row	0.080 Seconds

Select Query Results for table “TELLERS” in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.020 Seconds
200th Row	0.020 Seconds
250th Row	0.020 Seconds
300th Row	0.010 Seconds
350th Row	0.060 Seconds
400th Row	0.050 Seconds
450th Row	0.030 Seconds
500th Row	0.030 Seconds
550th Row	0.030 Seconds
600th Row	0.090 Seconds
650th Row	0.050 Seconds
700th Row	0.040 Seconds

Select Query Results for table “TELLERS” in SQL:

Number of Rows	Query Execution Time
20th Row	0.063 Seconds
40th Row	0.072 Seconds
60th Row	0.072 Seconds
100th Row	0.086 Seconds
150th Row	0.146 Seconds
200th Row	0.080 Seconds
250th Row	0.111 Seconds
300th Row	0.092 Seconds
350th Row	0.086 Seconds
400th Row	0.132 Seconds
450th Row	0.121 Seconds
500th Row	0.076 Seconds
550th Row	0.074 Seconds
600th Row	0.090 Seconds
650th Row	0.103 Seconds
700th Row	0.100 Seconds

Select Query Results for table "ACCOUNTS" in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.001 Seconds
40^{th} Row	0.001 Seconds
60^{th} Row	0.001 Seconds
100^{th} Row	0.001 Seconds
150^{th} Row	0.020 Seconds
200^{th} Row	0.020 Seconds
250^{th} Row	0.020 Seconds
300^{th} Row	0.020 Seconds
350^{th} Row	0.030 Seconds
400^{th} Row	0.030 Seconds
450^{th} Row	0.020 Seconds
500^{th} Row	0.040 Seconds
550^{th} Row	0.050 Seconds
600^{th} Row	0.030 Seconds
650^{th} Row	0.030 Seconds
700^{th} Row	0.030 Seconds

Select Query Results for table "ACCOUNTS" in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.081 Seconds
40^{th} Row	0.073 Seconds
60^{th} Row	0.087 Seconds
100^{th} Row	0.132 Seconds
150^{th} Row	0.128 Seconds
200^{th} Row	0.080 Seconds
250^{th} Row	0.111 Seconds
300^{th} Row	0.094 Seconds
350^{th} Row	0.090 Seconds
400^{th} Row	0.083 Seconds
450^{th} Row	0.100 Seconds
500^{th} Row	0.077 Seconds
550^{th} Row	0.121 Seconds
600^{th} Row	0.075 Seconds
650^{th} Row	0.137 Seconds
700^{th} Row	0.124 Seconds

Select Query Results for table "HISTORY" in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.001 Seconds
40^{th} Row	0.001 Seconds
60^{th} Row	0.001 Seconds
100^{th} Row	0.001 Seconds
150^{th} Row	0.020 Seconds
200^{th} Row	0.010 Seconds
250^{th} Row	0.030 Seconds
300^{th} Row	0.030 Seconds
350^{th} Row	0.080 Seconds
400^{th} Row	0.030 Seconds
450^{th} Row	0.080 Seconds
500^{th} Row	0.030 Seconds
550^{th} Row	0.160 Seconds
600^{th} Row	0.060 Seconds
650^{th} Row	0.070 Seconds
700^{th} Row	0.060 Seconds

Select Query Results for table "HISTORY" in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.044 Seconds
40^{th} Row	0.093 Seconds
60^{th} Row	0.088 Seconds
100^{th} Row	0.113 Seconds
150^{th} Row	0.173 Seconds
200^{th} Row	0.101 Seconds
250^{th} Row	0.099 Seconds
300^{th} Row	0.097 Seconds
350^{th} Row	0.130 Seconds
400^{th} Row	0.163 Seconds
450^{th} Row	0.116 Seconds
500^{th} Row	0.110 Seconds
550^{th} Row	0.145 Seconds
600^{th} Row	0.100 Seconds
650^{th} Row	0.125 Seconds
700^{th} Row	0.105 Seconds

Tables shown below are the results for the Join query for both databases.

Join Query Results for Oracle database:

Number of Rows	Query Execution Time
20^{th} Row	0.020 Seconds
40^{th} Row	0.020 Seconds
60^{th} Row	0.060 Seconds
100^{th} Row	0.020 Seconds
150^{th} Row	0.020 Seconds
200^{th} Row	0.040 Seconds
250^{th} Row	0.050 Seconds
300^{th} Row	0.060 Seconds
350^{th} Row	0.060 Seconds
400^{th} Row	0.060 Seconds
450^{th} Row	0.080 Seconds
500^{th} Row	0.070 Seconds
550^{th} Row	0.320 Seconds
600^{th} Row	0.100 Seconds
650^{th} Row	0.110 Seconds
700^{th} Row	0.140 Seconds

Join Query Results for SQL database:

Number of Rows	Query Execution Time
20^{th} Row	0.101 Seconds
40^{th} Row	0.178 Seconds
60^{th} Row	0.198 Seconds
100^{th} Row	0.174 Seconds
150^{th} Row	0.381 Seconds
200^{th} Row	0.178 Seconds
250^{th} Row	0.207 Seconds
300^{th} Row	0.203 Seconds
350^{th} Row	0.197 Seconds
400^{th} Row	0.244 Seconds
450^{th} Row	0.280 Seconds
500^{th} Row	0.323 Seconds
550^{th} Row	0.172 Seconds
600^{th} Row	0.175 Seconds
650^{th} Row	0.198 Seconds
700^{th} Row	0.200 Seconds

Tables given below are the performance measurements for Update query.

Update Query Results for table “BRANCHES” in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.020 Seconds
100th Row	0.001 Seconds
150th Row	0.001 Seconds
200th Row	0.001 Seconds
250th Row	0.001 Seconds
300th Row	0.020 Seconds
350th Row	0.001 Seconds
400th Row	0.001 Seconds
450th Row	0.001 Seconds
500th Row	0.001 Seconds
550th Row	0.030 Seconds
600th Row	0.001 Seconds
650th Row	0.010 Seconds
700th Row	0.001 Seconds

Update Query Results for table “BRANCHES” in SQL:

Number of Rows	Query Execution Time
20th Row	0.035 Seconds
40th Row	0.016 Seconds
60th Row	0.013 Seconds
100th Row	0.015 Seconds
150th Row	0.032 Seconds
200th Row	0.053 Seconds
250th Row	0.013 Seconds
300th Row	0.042 Seconds
350th Row	0.034 Seconds
400th Row	0.019 Seconds
450th Row	0.018 Seconds
500th Row	0.014 Seconds
550th Row	0.021 Seconds
600th Row	0.013 Seconds
650th Row	0.026 Seconds
700th Row	0.020 Seconds

Update Query Results for table "TELLERS" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.020 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.001 Seconds
200th Row	0.001 Seconds
250th Row	0.030 Seconds
300th Row	0.020 Seconds
350th Row	0.030 Seconds
400th Row	0.001 Seconds
450th Row	0.001 Seconds
500th Row	0.001 Seconds
550th Row	0.650 Seconds
600th Row	0.001 Seconds
650th Row	0.020 Seconds
700th Row	0.001 Seconds

Update Query Results for table "TELLERS" in SQL:

Number of Rows	Query Execution Time
20th Row	0.014 Seconds
40th Row	0.021 Seconds
60th Row	0.017 Seconds
100th Row	0.024 Seconds
150th Row	0.014 Seconds
200th Row	0.013 Seconds
250th Row	0.014 Seconds
300th Row	0.045 Seconds
350th Row	0.025 Seconds
400th Row	0.025 Seconds
450th Row	0.014 Seconds
500th Row	0.014 Seconds
550th Row	0.013 Seconds
600th Row	0.022 Seconds
650th Row	0.012 Seconds
700th Row	0.014 Seconds

Update Query Results for table "ACCOUNTS" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.030 Seconds
200th Row	0.020 Seconds
250th Row	0.001 Seconds
300th Row	0.010 Seconds
350th Row	0.010 Seconds
400th Row	0.001 Seconds
450th Row	0.001 Seconds
500th Row	0.001 Seconds
550th Row	0.080 Seconds
600th Row	0.001 Seconds
650th Row	0.001 Seconds
700th Row	0.001 Seconds

Update Query Results for table "ACCOUNTS" in SQL:

Number of Rows	Query Execution Time
20th Row	0.018 Seconds
40th Row	0.014 Seconds
60th Row	0.014 Seconds
100th Row	0.015 Seconds
150th Row	0.014 Seconds
200th Row	0.015 Seconds
250th Row	0.014 Seconds
300th Row	0.043 Seconds
350th Row	0.065 Seconds
400th Row	0.028 Seconds
450th Row	0.019 Seconds
500th Row	0.033 Seconds
550th Row	0.022 Seconds
600th Row	0.013 Seconds
650th Row	0.051 Seconds
700th Row	0.045 Seconds

Update Query Results for table "HISTORY" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.040 Seconds
150th Row	0.001 Seconds
200th Row	0.030 Seconds
250th Row	0.020 Seconds
300th Row	0.010 Seconds
350th Row	0.020 Seconds
400th Row	0.070 Seconds
450th Row	0.030 Seconds
500th Row	0.001 Seconds
550th Row	0.010 Seconds
600th Row	0.050 Seconds
650th Row	0.001 Seconds
700th Row	0.001 Seconds

Update Query Results for table "HISTORY" in SQL:

Number of Rows	Query Execution Time
20th Row	0.098 Seconds
40th Row	0.014 Seconds
60th Row	0.025 Seconds
100th Row	0.014 Seconds
150th Row	0.027 Seconds
200th Row	0.067 Seconds
250th Row	0.039 Seconds
300th Row	0.014 Seconds
350th Row	0.149 Seconds
400th Row	0.062 Seconds
450th Row	0.059 Seconds
500th Row	0.014 Seconds
550th Row	0.014 Seconds
600th Row	0.038 Seconds
650th Row	0.065 Seconds
700th Row	0.020 Seconds

Tables given below are performance measurements for the Delete queries on both DBMS. Delete Query Results for table "BRANCHES" in Oracle:

Number of Rows	Query Execution Time
20th Row	0.001 Seconds
40th Row	0.001 Seconds
60th Row	0.001 Seconds
100th Row	0.001 Seconds
150th Row	0.010 Seconds
200th Row	0.001 Seconds
250th Row	0.020 Seconds
300th Row	0.001 Seconds
350th Row	0.030 Seconds
400th Row	0.001 Seconds
450th Row	0.001 Seconds
500th Row	0.020 Seconds
550th Row	0.090 Seconds
600th Row	0.001 Seconds
650th Row	0.010 Seconds
700th Row	0.001 Seconds

Delete Query Results for table "BRANCHES" in SQL:

Number of Rows	Query Execution Time
20th Row	0.020 Seconds
40th Row	0.020 Seconds
60th Row	0.014 Seconds
100th Row	0.064 Seconds
150th Row	0.015 Seconds
200th Row	0.055 Seconds
250th Row	0.048 Seconds
300th Row	0.031 Seconds
350th Row	0.042 Seconds
400th Row	0.051 Seconds
450th Row	0.020 Seconds
500th Row	0.028 Seconds
550th Row	0.014 Seconds
600th Row	0.059 Seconds
650th Row	0.028 Seconds
700th Row	0.020 Seconds

Delete Query Results for table “TELLERS” in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.040 Seconds
40^{th} Row	0.040 Seconds
60^{th} Row	0.020 Seconds
100^{th} Row	0.050 Seconds
150^{th} Row	0.020 Seconds
200^{th} Row	0.001 Seconds
250^{th} Row	0.001 Seconds
300^{th} Row	0.001 Seconds
350^{th} Row	0.070 Seconds
400^{th} Row	0.030 Seconds
450^{th} Row	0.040 Seconds
500^{th} Row	0.001 Seconds
550^{th} Row	2.380 Seconds
600^{th} Row	0.070 Seconds
650^{th} Row	0.001 Seconds
700^{th} Row	0.020 Seconds

Delete Query Results for table “TELLERS” in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.050 Seconds
40^{th} Row	0.029 Seconds
60^{th} Row	0.034 Seconds
100^{th} Row	0.019 Seconds
150^{th} Row	0.027 Seconds
200^{th} Row	0.030 Seconds
250^{th} Row	0.051 Seconds
300^{th} Row	0.017 Seconds
350^{th} Row	0.014 Seconds
400^{th} Row	0.029 Seconds
450^{th} Row	0.028 Seconds
500^{th} Row	0.031 Seconds
550^{th} Row	0.028 Seconds
600^{th} Row	0.030 Seconds
650^{th} Row	0.026 Seconds
700^{th} Row	0.025 Seconds

Delete Query Results for table “ACCOUNTS” in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.020 Seconds
40^{th} Row	0.020 Seconds
60^{th} Row	0.030 Seconds
100^{th} Row	0.020 Seconds
150^{th} Row	0.010 Seconds
200^{th} Row	0.010 Seconds
250^{th} Row	0.020 Seconds
300^{th} Row	0.010 Seconds
350^{th} Row	0.001 Seconds
400^{th} Row	0.001 Seconds
450^{th} Row	0.010 Seconds
500^{th} Row	0.001 Seconds
550^{th} Row	0.360 Seconds
600^{th} Row	0.020 Seconds
650^{th} Row	0.020 Seconds
700^{th} Row	0.001 Seconds

Delete Query Results for table “ACCOUNTS” in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.068 Seconds
40^{th} Row	0.015 Seconds
60^{th} Row	0.016 Seconds
100^{th} Row	0.015 Seconds
150^{th} Row	0.068 Seconds
200^{th} Row	0.060 Seconds
250^{th} Row	0.014 Seconds
300^{th} Row	0.028 Seconds
350^{th} Row	0.045 Seconds
400^{th} Row	0.032 Seconds
450^{th} Row	0.484 Seconds
500^{th} Row	0.014 Seconds
550^{th} Row	0.014 Seconds
600^{th} Row	0.128 Seconds
650^{th} Row	0.053 Seconds
700^{th} Row	0.028 Seconds

Delete Query Results for table “HISTORY” in Oracle:

Number of Rows	Query Execution Time
20^{th} Row	0.001 Seconds
40^{th} Row	0.010 Seconds
60^{th} Row	0.010 Seconds
100^{th} Row	0.001 Seconds
150^{th} Row	0.001 Seconds
200^{th} Row	0.010 Seconds
250^{th} Row	0.020 Seconds
300^{th} Row	0.001 Seconds
350^{th} Row	0.020 Seconds
400^{th} Row	0.001 Seconds
450^{th} Row	0.010 Seconds
500^{th} Row	0.020 Seconds
550^{th} Row	0.220 Seconds
600^{th} Row	0.020 Seconds
650^{th} Row	0.001 Seconds
700^{th} Row	0.001 Seconds

Delete Query Results for table “HISTORY” in SQL:

Number of Rows	Query Execution Time
20^{th} Row	0.018 Seconds
40^{th} Row	0.015 Seconds
60^{th} Row	0.015 Seconds
100^{th} Row	0.015 Seconds
150^{th} Row	0.047 Seconds
200^{th} Row	0.029 Seconds
250^{th} Row	0.032 Seconds
300^{th} Row	0.020 Seconds
350^{th} Row	0.079 Seconds
400^{th} Row	0.059 Seconds
450^{th} Row	0.084 Seconds
500^{th} Row	0.014 Seconds
550^{th} Row	0.055 Seconds
600^{th} Row	0.031 Seconds
650^{th} Row	0.047 Seconds
700^{th} Row	0.040 Seconds

Memory usage for table “BRANCHES” in Oracle:

Number of Rows	Memory Usage
20^{th} Row	0.06 MB
40^{th} Row	0.06 MB
60^{th} Row	0.06 MB
100^{th} Row	0.06 MB
150^{th} Row	0.06 MB
200^{th} Row	0.06 MB
250^{th} Row	0.06 MB
300^{th} Row	0.06 MB
350^{th} Row	0.06 MB
400^{th} Row	0.06 MB
450^{th} Row	0.06 MB
500^{th} Row	0.06 MB
550^{th} Row	0.06 MB
600^{th} Row	0.06 MB
650^{th} Row	0.06 MB
700^{th} Row	0.06 MB

Memory usage for table “BRANCHES” in SQL:

Number of Rows	Memory Usage
20^{th} Row	0.008 MB
40^{th} Row	0.008 MB
60^{th} Row	0.008 MB
100^{th} Row	0.008 MB
150^{th} Row	0.008 MB
200^{th} Row	0.008 MB
250^{th} Row	0.008 MB
300^{th} Row	0.008 MB
350^{th} Row	0.008 MB
400^{th} Row	0.016 MB
450^{th} Row	0.016 MB
500^{th} Row	0.016 MB
550^{th} Row	0.016 MB
600^{th} Row	0.016 MB
650^{th} Row	0.016 MB
700^{th} Row	0.016 MB

Memory usage for table “TELLERS” in Oracle:

Number of Rows	Memory Usage
20th Row	0.06 MB
40th Row	0.06 MB
60th Row	0.06 MB
100th Row	0.06 MB
150th Row	0.06 MB
200th Row	0.06 MB
250th Row	0.06 MB
300th Row	0.06 MB
350th Row	0.06 MB
400th Row	0.06 MB
450th Row	0.06 MB
500th Row	0.06 MB
550th Row	0.06 MB
600th Row	0.06 MB
650th Row	0.06 MB
700th Row	0.06 MB

Memory usage for table “TELLERS” in SQL:

Number of Rows	Memory Usage
20th Row	0.008 MB
40th Row	0.008 MB
60th Row	0.008 MB
100th Row	0.008 MB
150th Row	0.008 MB
200th Row	0.008 MB
250th Row	0.008 MB
300th Row	0.016 MB
350th Row	0.016 MB
400th Row	0.016 MB
450th Row	0.016 MB
500th Row	0.016 MB
550th Row	0.016 MB
600th Row	0.023 MB
650th Row	0.023 MB
700th Row	0.023 MB

Memory usage for table “ACCOUNTS” in Oracle:

Number of Rows	Memory Usage
20th Row	0.06 MB
40th Row	0.06 MB
60th Row	0.06 MB
100th Row	0.06 MB
150th Row	0.06 MB
200th Row	0.06 MB
250th Row	0.06 MB
300th Row	0.06 MB
350th Row	0.06 MB
400th Row	0.06 MB
450th Row	0.06 MB
500th Row	0.06 MB
550th Row	0.06 MB
600th Row	0.06 MB
650th Row	0.06 MB
700th Row	0.06 MB

Memory usage for table “ACCOUNTS” in SQL:

Number of Rows	Memory Usage
20th Row	0.008 MB
40th Row	0.008 MB
60th Row	0.008 MB
100th Row	0.008 MB
150th Row	0.008 MB
200th Row	0.008 MB
250th Row	0.008 MB
300th Row	0.016 MB
350th Row	0.016 MB
400th Row	0.016 MB
450th Row	0.016 MB
500th Row	0.016 MB
550th Row	0.016 MB
600th Row	0.023 MB
650th Row	0.023 MB
700th Row	0.023 MB

Memory usage for table “HISTORY” in Oracle:

Number of Rows	Memory Usage
20th Row	0.06 MB
40th Row	0.06 MB
60th Row	0.06 MB
100th Row	0.06 MB
150th Row	0.06 MB
200th Row	0.06 MB
250th Row	0.06 MB
300th Row	0.06 MB
350th Row	0.06 MB
400th Row	0.06 MB
450th Row	0.06 MB
500th Row	0.06 MB
550th Row	0.06 MB
600th Row	0.06 MB
650th Row	0.06 MB
700th Row	0.06 MB

Memory usage for table “HISTORY” in SQL:

Number of Rows	Memory Usage
20th Row	0.008 MB
40th Row	0.008 MB
60th Row	0.008 MB
100th Row	0.008 MB
150th Row	0.008 MB
200th Row	0.016 MB
250th Row	0.016 MB
300th Row	0.016 MB
350th Row	0.016 MB
400th Row	0.023 MB
450th Row	0.023 MB
500th Row	0.023 MB
550th Row	0.023 MB
600th Row	0.031 MB
650th Row	0.031 MB
700th Row	0.031 MB

BIBLIOGRAPHY

[1] http://reference.kfupm.edu.sa/content/d/e/design_and_implementation_of_a_dbms_perf_2133961.pdf

[2] Technical White Paper, “MySQL Performance Benchmarks”, 2006

[3] Bitton, D., DeWitt, D.J., Turbyfill, C.: Benchmarking Database Systems A Systematic Approach. VLDB 1983: 8-19.

[4] A.Ailamaki. “Database Architecture for New Hardware.” Tutorial. In *Proc. VLDB*, 2004.

[5] P. A. Boncz, S. Manegold, and M. L. Kersten. “Database Architecture Optimized for the New Bottleneck: Memory Access.” In *Proc. VLDB*, 1999.

[6] http://informingscience.org/proceedings/InSITE2006/IISITLee172.pdf

[7] http://www.theserverside.com/news/1364725/Tips-on-Performance-Testing-and-Optimization

[8] http://docsdrive.com/pdfs/medwelljournals/jeasci/2008/155-160.pdf

[9] http://www.dbd.puc-rio.br/depto_informatica/04_40_macedo.pdf

[10] http://eden.dei.uc.pt/~henrique/papers/PRDC11_cr.pdf

[11] http://pages.cs.wisc.edu/~dewitt/includes/benchmarking/sigmod84.pdf

[12] http://dev.mysql.com/doc/refman/5.1/en/pluggable-storage-overview.html

[13] http://www.techotopia.com/index.php/MySQL_Database_Architecture

[14] http://www.ehow.com/about_5387852_advantages-microsoft-sql.html

[15] http://www.simple-talk.com/sql/database-administration/ten-common-database-design-mistakes/

[16] http://www.learn.geekinterview.com/database/oracle/advantages-of-using-oracle.html

[17] http://www.ehow.com/list_6309384_advantages-disadvantages-oracle-sql.html

[18] http://www.rocket99.com/techref/8690.html

[19] http://www.wikipedia.org/