Database Management: A Performance and Tuning Perspective

Harshit Jhaveri
Credence Analytics (I) Pvt. Ltd.
Email: harshitjhaveri {at} gmail.com

Hardik Jhaveri
DJSOE, Mumbai University

***Abstract*— Database Management Systems (DBMS) provide a mechanism for storing, organizing, retrieving and interacting with data in a database. The DBMS decides how information in the form of data is stored and retrieved. It also addresses issues concerning memory requirements, accuracy, security, consistency, maintenance and response times. DBMS Performance and Tuning is an integral aspect of database management. This paper presents various methods to tackle performance issues using effective tuning tools and techniques.**

Keywords- Database, Types of DBMS, Characteristics of DBMS, Performance, Tuning, Tools, Techniques and Methods, Security

I. INTRODUCTION

Data is a series of facts that have been gathered, processed, stored, manipulated but not organized. When this data is organized into meaningful units, it becomes information. Database is nothing but a collection of information. DBMS is the software that allows a computer to perform database functions of storing, retrieving and manipulating data.

DBMSs are specially designed collection of complex programs that enables the end-users to store, manipulate and retrieve information from the database. These systems have been optimized over time.

Application programs request DBMS to retrieve, insert, modify or delete data and thus it acts as a layer of abstraction between the application programs and the file system. Structured Query Language (SQL) is used to interact with the DBMS.

Most DBMS applications also provide reporting and query tools to view data from database in a presentable and readable manner which is easy for any lame user to interpret.

A. Types of DBMS

There are four structural types of DBMS:

- Hierarchical
- Network
- Relational
- Object-Oriented

1) Hierarchial DBMS

Hierarchical DBMS organizes data in a tree structure with fields forming the branches of the tree and records forming the nodes. There is a hierarchy of parent-child relationships. It can be difficult to navigate because of its strict one-to-many relationship. It only follows downward tree structure.

2) Network DBMS

Network DBMS demonstrates many-to-many relationships between data. This feature can make its structure complicated. A network DBMS looks more like an interconnected network of records. It may follow upward tree structure in addition to the downward tree structure. Records are easily accessed because of the linkage between records.

3) Relational DBMS

Relational DBMS or RDBMS stores data in the form of rows and columns i.e. a table. Data is stored in different tables, each having a key field that uniquely identifies each record and this key field is used to establish connection between two or multiple tables. A single database can contain information across multiple tables. It also provides for relational operators to manipulate data.

4) Object-Oriented DBMS

Objected-Oriented DBMS add database functionality to object-oriented programming languages. They bring much more than persistent storage of programming language objects. It allows users to develop products, store them as objects and manipulate or reuse existing objects to create new objects. These are however expensive to maintain. Data in the form of photograph, audio, video, graphics can be stored. The ability to create reusable objects provides for incredible multimedia capability. [3]

II. NEED AND USE

In traditional approach, information is stored in flat files which are maintained by the file system under the operating system's control. Application programs go through the file system in order to access these flat files. [1]

There are various problems with the above approach, which are listed herein below:

A. Data Redundancy

It is quite likely that data gets duplicated in multiple files, which leads to data redundancy. This would lead to unnecessary wastage of storage space.

B. Isolation of Data

It is quite possible that all related data is not stored in one file. In such cases, data is scattered across various files, which may be in different formats. This leads to additional burden of keeping track of various data stored in multiple files.

C. Data Dependency

Application programs are highly dependent on the files in which data is stored. If any changes are made to the physical file format, like addition of a data field, all corresponding application programs need to be changed accordingly.

D. Data Inconsistency

Multiple users can access and update data simultaneously in the same file. This concurrent updating of data may lead to data inconsistency.

E. Data Security

Application programs are added in an adhoc manner, due to which details of each file are easily available to every user. This leads to lack of security.

The file processing system has a number of disadvantages. This is where DBMS comes in handy.

Database Management Systems help evolving businesses to manage increasing volumes of data so as to improve customer service, control costs and accelerate product or service development. It manages concurrent access to data in a predictable, repeatable and controlled manner for multiple end-users.

A DBMS can make the same data available to multiple applications. It enables sharing of data and reduces data redundancy. A DBMS creates backup data copies for disaster recovery. Data can be quickly recovered and operations restored after a calamity such as a fire, a data management error or hardware failure.

Managing huge volumes of data deluge becomes increasingly difficult. It is a powerful tool used to store data, secure it, protect it and make it quickly available to people who need it. It enables a business to squeeze more value from the data it collects for improved decision-making. DBMS keeps track of all rules implicit in the database and helps to maintain data integrity.

III. CHARACTERISTICS

DBMS has the following characteristics.

A. Data Integrity

The template is designed so that author affiliations are not repeated each time for multiple authors of the same affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization). This template was designed for two affiliations.

B. ACID Properties

DBMS follows the concepts of ACID properties, which stands for Atomicity, Consistency, Isolation and Durability. These concepts are applied on transactions, which manipulate data in database. ACID properties maintains database in healthy state in multi-transactional environment and in case of failure.

C. Less Redundancy

DBMS follows rules of normalization, which is to organize data to minimize redundancy. Following normalization, which itself is a mathematically rich and scientific process, make the entire database to contain as less redundancy as possible.

D. Consistency

Consistency is a state where every relation in database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state.

E. Relation based Tables

DBMS allows entities and relations among them to form as tables. This eases the concept of data saving. A user can understand the architecture of database just by looking at table names etc.

F. Query Language

DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and different filtering options, as he or she wants.

G. Multi User And Concurrent Access

DBMS supports multi-user environment and allows for access and manipulation of data in parallel. There are restrictions on transactions when they attempt to handle same data item, but users are always unaware of them. Sharing of the data takes place amongst the different type of the users and the applications.

H. Multiple Views

DBMS offers multiples views for different users. This enables user to have a concentrate view of database according to their requirements.

I. Security

DBMS have provisions for controlling access to data and to prevent unauthorized access and update. Only authorized users are allowed access to the database. It offers methods to impose constraints while entering data into database and retrieving data at later stage. It also offers many different levels of security

features, which enables multiple users to have different view with different features. [1]

IV. PERFORMANCE AND TUNING

A. What is it ?

The DBMS supplies information to those requesting it. DBMS performance refers to the rate at which the DBMS provides information to the user requesting it. Few factors influence database performance – resources, throughput, workload, optimization & contention.

a) Resouces : The hardware and software tools used by the DBMS are known as resources. Examples are memory, disk, etc.

b) Throughput : It defines the overall capability of the DBMS to process data. It is a composite of I/O speed, CPU speed, parallel capabilities of the machine, and the efficiency of the operating system and system software.

c) Workload : It refers to the extent of demand of information at any given time. It is a combination of online transactions, batch jobs, ad hoc queries, business intelligence queries and analysis, utilities, and system commands directed through the DBMS at any given time. Workload can fluctuate drastically from time to time.

d) Optimization : All types of systems can be optimized, but relational database systems are unique in that query optimization is primarily accomplished internal to the DBMS. Ensuring that you have provided up-to-date and accurate database statistics for the query optimizer is of the utmost importance in achieving optimal SQL queries. Keep in mind, though, that there are other factors that need to be optimized (database parameters, system parameters, etc.) to enable the relational optimizer to create the most efficient access paths. And there are optimization aspects that are outside the scope and control of the relational optimizer, too, such as efficient script coding, proper application design, coding efficient utility options, and so on.

e) Contention : When the demand (workload) for a particular resource is high, contention can result. Contention is the condition in which two or more components of the workload are attempting to use a single resource in a conflicting way (for example, dual updates to the same piece of data). When one program tries to read data that is in the process of being changed by another, the DBMS must prohibit access until the modification is complete in order to ensure data integrity. The DBMS uses a locking mechanism to enable multiple, concurrent users to access and modify data in the database. Using locks the DBMS automatically guarantees the integrity of data. The DBMS locking strategies permit multiple users from multiple environments to access and modify data in the database at the same time. As contention increases, locking ensues and throughput decreases.

Database performance can thus be defined as the optimization of resource use to increase throughput and minimize contention, enabling the largest possible workload to be processed.

Database tuning comprises of a group of activities used to optimize and regulate the performance of a database. It refers to DBA of the database files, the database management system (DBMS), as well as the hardware and operating system on which the database is hosted. The goal of database tuning is to maximize the application of system resources in an attempt to execute transactions as efficiently and quickly as possible. The large majority of DBMS are designed with efficiency in mind; however, it is possible to enhance a database's performance via custom settings and DBAs.

The tuning of a database management system centers on the DBA of memory and processing resources of the computer running the DBMS. This can involve setting the recovery interval of the DMBS, establishing the level of concurrency control, and assigning which network protocols are used to communicate throughout the database. Memory utilized by the DBMS is allocated for data, execution procedures, procedure cache, and work space. Since it is faster to directly access data in memory than data on storage, it is possible to decrease the average access time of database transactions by maintaining a decent sized data cache.

Database performance can also be improved by using the cache to store execution procedures as they would not need to be recompiled with every transaction. By assigning processing resources to specific functions and activities, it is also possible to improve the concurrency of the system. Database concurrency controls ensure that transactions occur in an ordered fashion. The main job of these controls is to protect transactions issued by different users/applications from the effects of each other. [12]

B. Why is it needed and its importance?

DBMS is used to store and retrieve data. With the huge advancement of technology and the computing powers being doubled probably a few hundred times, we are now not satisfied with just being able to retrieve stored data; we want to retrieve data faster, we want our operations to complete in the least amount of time possible. [8]

Queries must be tuned to achieve optimal performance; some of the common ways of tuning database systems include creating appropriate indexes, updating statistics, rebuilding indexes to reduce fragmentation. However, sometimes we tend to forget that each database system is just another piece of code running on a bunch of dumb hardware; unless we design our databases properly, unless we write efficient code, we will never be able to achieve the speeds we desire.

Performance tuning requires a different, although related, method to the initial configuration of a system. Configuring a system involves allocating resources in an ordered manner so

that the initial system configuration is functional. Tuning is driven by identifying the most significant bottleneck and making the appropriate changes to reduce or eliminate the effect of that bottleneck. Usually, tuning is performed reactively, either while the system is preproduction or after it is live. [10]

C. Methods

Effective data collection and analysis is essential for identifying and correcting performance problems. DBMS provide a number of tools that allow a performance specialist to gather information regarding database performance. In addition to gathering data, it also provides tools to monitor performance, diagnose problems, and tune applications. [7]

a) Regular Monitoring :

A number of performance statistics are examined at regular intervals to identify whether the system behavior and resource usage has changed. Monitoring does not necessarily result in DBA changes to the system, unless the monitoring exposes a serious problem that is developing. In some situations, experienced performance specialists can identify potential problems through statistics alone.

Experimenting with or tweaking a system when there is no apparent performance degradation can be a dangerous activity, resulting in unnecessary performance drops. Tweaking a system should be considered reactive tuning, and the steps for reactive tuning should be followed.

Monitoring is usually part of a larger capacity planning exercise, where resource consumption is examined to see changes in the way the application is being used, and the way the application is using the database and host resources.

b) Analysis and Testing :

This requires the performance specialist to plan and implement experiments and gather and assess statistics to improve system performance. This could however, be risky in implementing in real life scenario. Analysis may allow establishing upper and lower run time bounds and thereby choosing the solution that provides the greatest reduction in cost. Experimentation allows testing for expected loads and unique circumstances in accord with the transitory nature of today's technology and business environment.

In order to optimize and make efficient use of memory, some of today's DBMS make use of compression to fit as much data into memory as possible. Compression consumes computational resources therefore using experimentation the DBA may be able to statistically measure and assess the application of compression on the system.

c) Bottleneck Elimination :

Tuning usually implies fixing a performance problem. However, tuning should be part of the life cycle of an application-through the analysis, design, coding, production, and maintenance stages. Oftentimes, the tuning phase is left until the system is in production. At this time, tuning becomes a reactive fire-fighting exercise, where the most important bottleneck is identified and fixed.

Usually, the purpose for tuning is to reduce resource consumption or to reduce the elapsed time for an operation to complete. Either way, the goal is to improve the effective use of a particular resource. In general, performance problems are caused by the over-use of a particular resource. That resource is the bottleneck in the system. There are a number of distinct phases in identifying the bottleneck and the potential fixes.

d) Distributed Tuning Tools :

Organizations need to develop formalized database tuning best practices. Citing the complexity and diverse skill sets, it appears a community-based or open source development approach may be appropriate. It is also understood that security and information assurance must be foremost in every DBA's mind and an open-source community can cull the expertise of security experts and also assess performance tuning changes from a security standpoint.

e) I/O Tuning :

Input/output (I/O) tuning is another major component of DBMS tuning. I/O tuning mainly deals with database transaction logs. Database transaction logs are files that are associated with temporary work spaces as well as both table and index file storage. Transaction logs and temporary spaces are heavy consumers of I/O, and affect performance for all users of the database. Placing them appropriately is crucial. The main goal of I/O tuning a database is to optimize and balance the read and write transactions of the system in order to achieve an increased speed in database transactions and a decreased database access time.

f) RAID :

Another method of ensuring that a database is fast and reliable is the use of RAID in the creation of the database. RAID stands for Redundant Array of Independent Disks.

RAID is superior to a single disk in the sense that if data is stored on one disk, the entire database is completely reliant on that one disk; if it were to fail, the database would not exist anymore. Another drawback to having it on a single disk is the read/write time. One hard disk can only be so fast. If there is a lot of I/O data being processed, it can be a lengthy process. One thing that RAID does is, it divides and replicates the data onto several independent disks.

If we had a RAID 6 array with 4 drives, the tolerance for failure is 2 disks. This means that if 2 hard drives fail

completely, the database will still function perfectly. On top of failure tolerance, another great upside to using RAID is tasks are performed faster. Reading is faster, and writing is faster because instead of one disk trying to find all the data, the task is broken down into parts, and each hard disk does part of the job.

Performance tuning is an overwhelming task and as a result is driving the need for increased assistance and automatic self tuning. In an optimal scenario, the move towards autonomic or self tuning DBMS algorithms may allow administrators to specify their expectations or goals allowing the system to dynamically reconfigure. This goal oriented approach may allow the DBA to specify goals in terms of response times for classes of transactions. This of course requires that the system first identify bottlenecks, constraints and solutions.

D. Tools

We shall discuss this section with respect to Oracle DBMS. Oracle provides the following tools to assist with performance monitoring and tuning:

- ADDM (Automated Database Diagnostics Monitor) (cost option)
- Stats pack (Free)
- TKPROF (Free)
- Oracle Enterprise Manager (cost option)

1) ADDM

In Oracle, ADDM stands for Automatic Database Diagnostic Monitor. The ADDM analyzes data in the Automatic Workload Repository (AWR) to identify potential performance bottlenecks. For each of the identified issues, it locates the root cause and provides recommendations for correcting the problem. An ADDM analysis task is performed and its findings and recommendations stored in the database every time an AWR snapshot is taken provided the STATISTICS_LEVEL parameter is set to TYPICAL or ALL. [2]

The ADDM analysis includes the following:

- CPU load
- Memory usage
- I/O usage
- Resource intensive SQL
- Resource intensive PL/SQL and Java
- RAC issues
- Application issues
- Database configuration issues
- Concurrency issues
- Object contention

The simplest way to generate this report is through the Oracle Enterprise Manager tool (OEM); however, there are times when this tool is not available. On those occasions, the DBA user can generate the report from the machine hosting Oracle, as described in the steps below.

To invoke ADDM analysis, simply follow the below mentioned steps:

- Log in to the machine hosting Oracle.
- Run the addmrpt.sql script at the SQLPlus prompt: @$ORACLE_HOME/rdbms/admin/ addmrpt.sql
- When you are asked for a system password, type it in and press Enter.
- Specify a begin_snap from the list and press Enter.
- Specify the end_snap from the list and press Enter.
- Enter a report name or accept the default name when prompted to specify the report name.
- Issue the quit command to exit from SQLPlus.

The report is generated at the path where the SQLPlus command was issued.

2) Statspack

Stats pack is a set of performance monitoring and reporting utilities provided by Oracle. Although AWR and ADDM provide better statistics than Statspack, users that do not have access to licensed version of Enterprise Manager should be availing this utility.

The best feature of STATSPACK is that it stores Oracle performance information in a set of 25 tables that can be used to develop historical trends. By interrogating these tables, Oracle professionals can gain tremendous insight into the relative performance of their databases.

The STATSPACK schema contains several control tables. The stats$parameter tables controls the thresholds for collection of detailed information, and a table called stats$level_description provides information regarding the level of detail collected with a snapshot.

Within a STATSPACK installation, the stats$sql_summary table will grow very rapidly because STATSPACK will extract SQL from the library cache every time a snapshot is executed. Hence, the Oracle administrator has to be careful to set the appropriate threshold values for stats$sql_summary data collection to ensure that the database doesn't run wild, consuming multiple megabytes of information every day.

The main anchor for STATSPACK is the table called stats$snapshot. This table contains the snapshot ID for all of the subordinate tables and the snap_time indicating when the snapshot was taken. Oracle also implements all of the subordinate tables with referential integrity, using the on cascade delete option.

This means that the stats$snapshot table can be deleted in order to delete rows from all of the subordinate tables after they have passed their useful lives within the database.

The simplest interactive way to take a snapshot is to login to SQL*Plus as the PERFSTAT user and run the procedure STATSPACK.SNAP. For example:

```
SQL> CONNECT perfstat/my_perfstat_password
SQL> EXECUTE statspack.snap;
```

Taking such a snapshot stores the current values for the performance statistics in the Statspack tables. This snapshot can be used as a baseline for comparison with another snapshot taken at a later time.

For better performance analysis, set the initialization parameter TIMED_STATISTICS to TRUE. Statspack will then include important timing information in the data it collects. You can change the TIMED_STATISTICS parameter dynamically by using the ALTER SYSTEM statement. Timing data is important and is usually required by Oracle support to diagnose performance problems.

3) TKPROF

TKPROF is an Oracle database utility used to format SQL Trace output into human readable format. The TKProf executable is located in the ORACLE HOME/bin directory. The TKPROF program converts Oracle trace files into a more readable form. If you have a problem query, you can user TKPROF to get more information. [6]

TKPROF is a program that you invoke at the operating system command prompt in order to reformat the trace file into a format that is much easier to comprehend. Each SQL statement is displayed in the report, along with counts of how many times it was parsed, executed, and fetched. CPU time, elapsed time, logical reads, physical reads, and rows processed are also reported, along with information about recursion level and misses in the library cache. TKPROF can also optionally include the execution plan for each SQL statement in the report, along with counts of how many rows were processed at each step of the execution plan.

The SQL statements can be listed in a TKPROF report in the order of how much resource they used, if desired. Also, recursive SQL statements issued by the SYS user to manage the data dictionary can be included or excluded, and TKPROF can write SQL statements from the traced session into a spool file.

Follow below mentioned steps to avail TKPROF utility:

- **Step 1**
  Set initialization parameters for trace file management. Parameters like TIMED_STATISTICS, USER_DUMP_DEST, MAX_DUMP_FILE_SIZE. TIMED_STATISTICS enables and disables the collection of timed statistics, such as CPU and elapsed times, by the SQL Trace facility, as well as the collection of various statistics in the dynamic performance tables. USER_DUMP_DEST must fully specify the destination for the trace file according to the conventions of the operating system. When the SQL Trace facility is enabled at the instance level, every call to the server produces a text line in a file in the operating system's file format. The maximum size of these files (in operating system blocks) is limited by the MAX_DUMP_FILE_SIZE parameter.
- **Step 2**
  Enable the SQL Trace facility for the desired session, and run the application. This step produces a trace file containing statistics for the SQL statements issued by the application. Enable the SQL Trace facility for the session using one of the following commands:
  DBMS_SESSION.SET_SQL_TRACE procedure;
  OR
  ALTER SESSION SET SQL_TRACE = TRUE;
- **Step 3**
  Run `TKPROF` to translate the trace file created in Step 2 step into a readable output file. This step can optionally create a SQL script that can be used to store the statistics in a database.

TKPROF accepts as input a trace file produced by the SQL Trace facility, and it produces a formatted output file. It can also be used to generate execution plans. After the SQL Trace facility has generated a number of trace files, you can:

- Run TKPROF on each individual trace file, producing a number of formatted output files, one for each session.
- Concatenate the trace files, and then run TKPROF on the result to produce a formatted output file for the entire instance.

TKPROF does not report COMMITs and ROLLBACKs that are recorded in the trace file.

- **Step 4**
  Interpret the output file created in Step 3.
  While `TKPROF` provides a very useful analysis, the most accurate measure of efficiency is the actual performance of the application in question. At the end of the `TKPROF` output is a summary of the work done in the database engine by the process during the period that the trace was running.
- **Step 5**
  Optionally, run the SQL script produced in Step 3 to store the statistics in the database.
  You might want to keep a history of the statistics generated by the SQL Trace facility for an application, and compare them over time. TKPROF can generate a SQL script that creates a table and inserts rows of statistics into it. This script contains:
  - A CREATE TABLE statement that creates an output table named TKPROF_TABLE.
  - INSERT statements that add rows of statistics, one for each traced SQL statement, to the TKPROF_TABLE.

  After running TKPROF, you can run this script to store the statistics in the database.

4) Oracle Enterprise Manager

Oracle Enterprise Manager is a system management tool that provides an integrated solution for centrally managing your heterogeneous environment. Combining a graphical console, Oracle Management Servers, Oracle Intelligent Agents, common services, and administrative tools, Oracle Enterprise Manager provides a comprehensive systems management platform for managing Oracle products. [11]

From the client interface, the Oracle Enterprise Manager Console, you can perform the following tasks:

- Administer the complete Oracle environment, including databases, iAS servers, applications, and services.
- Diagnose, modify, and tune multiple databases.
- Schedule tasks on multiple systems at varying time intervals.
- Monitor database conditions throughout the network.
- Administer multiple network nodes and services from many locations.
- Share tasks with other administrators.
- Group related targets together to facilitate administration tasks.
- Launch integrated Oracle and third-party tools.
- Customize the display of an Enterprise Manager administrator.

Ensure the dbconsole process is running on the server to access Oracle Enterprise Manager using the following steps:

Step 1

Point your Web browser to the following URL:

http://hostname:portnumber/em

If the database is up, Enterprise Manager displays the Database Control Login page. If the database is down and needs to be re-started, Enterprise Manager displays the Startup/Shutdown and Perform Recovery page. If this is the case, click Startup/Shutdown and enter the host and target database login usernames and passwords. For the database user and password, use SYS and the password you specified during installation.

Step 2

Click OK to start the database. In the Confirmation screen, click YES to start the database in open mode.

Step 3

Log in to the database using a username that is authorized to access the Database Control. This initially could be SYS or SYSTEM. Use the password you specified for the account during the database installation. Enterprise Manager displays the Database Home page. The property pages across the top of the page enable you to access performance, administration, and maintenance pages for managing your database. You can then update the details as required.

E. Techniques

In this section, we will discuss the various techniques available for improving performance of a SQL query. [5]

1. To select specific column names in SELECT statement instead of "SELECT * FROM".
2. Try to use UNION ALL instead of UNION.
3. Use UPPER or LOWER on either sides of a conditional statement whenever a static value is used in left hand side of condition e.g. WHERE, HAVING clauses.
4. In conditional statements involving date column, on RHS use TRUNC with date column name and on LHS include TO_DATE.
5. In conditional statements, ensure data types of columns on either side are same.
6. Use HAVING clause only to filter rows i.e. avoid using it for simple conditions.
   E.g. Instead of:
   select deptno,count(deptno) from dept group by deptno having deptno!=1 and deptno!=2
   Use:
   select deptno,count(deptno) from dept where deptno!=1 and deptno!=2 group by deptno.
7. Try to minimize no. of sub query blocks in a query
   E.g. Instead of:
   select name from emp where salary=(select max(salary) from emp_det) and age=(select max(age) from emp_det) and lower(dept)='electronics'
   Use:
   select name from emp where (salary,age)=(select max(salary),max(age) from emp_det) and lower(dept)='electronics'
8. Use operators EXISTS, IN and table joins appropriately in a query
   - Usually IN has the slowest performance
   - IN is efficient when most of the filter criteria are in the sub-query
   - EXISTS is efficient when most of the filter criteria is in the main query

   E.g. Instead of:
   select product_id,product_name from product where product_id IN (select product_id from orders)
   Use:
   *select product_id,product_name from product p where EXISTS (select * from orders o where o.product_id=p.product_id)*
9. Try to avoid writing a SQL query using multiple joins that include outer joins, cross joins and other complex sub-queries. It reduces the choices for Optimizer to decide the join order and join type.
10. Try to remove cursors from the query and use set-based query; set-based query is more efficient than cursor-based. If there is a need to use cursor than avoid dynamic cursors as it tends to limit the choice of plans available to the query optimizer.
11. Creation and use of indexes should be done in the most effective way. Indexes can help to reduce data retrieval

time but can have an adverse effect on DML operations, which may degrade query performance.

12. Understand the data, its type and how queries are written to retrieve data before an index can be created on it. If you understand the behavior of data thoroughly, it will help you to decide which column should have either a clustered index or a non-clustered index. If a clustered index is not on a unique column, then SQL Server will maintain uniqueness by adding a unique identifier to every duplicate key, which leads to overhead. To avoid this type of overhead, choose the column correctly or make appropriate changes.
13. Order or position of a column in an index also plays a vital role to improve SQL query performance. An index can help to improve the SQL query performance if the criterion of the query matches the columns that are left most in the index key. As a best practice, most selective columns should be placed leftmost in the key of a non-clustered index.
14. Dropping unused indexes can help to speed up data modifications without affecting data retrieval.
15. Provide an appropriate WHERE clause in SELECT statement, so that only the required rows is returned by the query.
16. Fire DELETE or UPDATE statements in small batches. If the transaction gets killed for whatever reason, it only has a small number of rows to roll back, so the database returns online much quicker. Second, while the smaller batches are committing to disk, others can sneak in and do some work, so concurrency is greatly enhanced.
17. Use DECODE to avoid the scanning of same rows or joining the same table repetitively. DECODE can also be made used in place of GROUP BY or ORDER BY clause.
    E.g. Instead of:
    select emp_id from employee where emp_name like '%ABC%' and location='INDIA'
    Use:
    select decode(location,'INDIA',emp_id,null) emp_id from employee where emp_name like '%ABC%'
18. Avoid use of sub-queries, if the same result can be achieved using joins.
19. Avoid the insertion of NULL values in the fixed-length (char) field, since NULL takes the same space as desired input value for that field. In case of requirement of NULL, use variable-length (varchar) field that takes less space for NULL.
20. All columns involved in indexes should appear on WHERE and JOIN clauses on the same sequence that they appear on index.
21. Verify if a critical query gains performance by turning it into a stored procedure.
22. The decreasing performance order of operators is:
    = (faster)>, >=, <, <=LIKE<> (slower)
23. When using LIKE operator, try to leave the wildcards on the right side of the VARCHAR.
24. Always avoid using functions on your queries. E.g. use LIKE operator instead of SUBSTRING.
25. A Query with all operations on the WHERE clause connected by ANDs is processed from the left to right. So, if a operation returns false, all other operations in the right side of it are ignored, because they can't change the AND result anyway. It is better than to start your WHERE clause with the operations that returns false most of the time.
26. To delete all rows on a table, use TRUNCATE TABLE statement instead of DELETE.

V. SECURITY

Database security deals with all various aspects of protecting the database content, its owners, and its users. Database access control deals with controlling who (a person or a certain computer program) is allowed to access what information in the database. The information may comprise specific database objects (e.g., record types, specific records, data structures), certain computations over certain objects (e.g., query types, or specific queries), or utilizing specific access paths to the former (e.g., using specific indexes or other data structures to access information). Database access controls are set by special authorized (by the database owner) personnel that use dedicated protected security DBMS interfaces. [4]

Database security concerns the use of a broad range of information security controls to protect databases (including data, applications, etc.) against compromises of their confidentiality, integrity and availability. It involves varied types or categories of controls, such as technical, procedural/administrative and physical. Database security is a specialist topic within the broader realms of computer security, information security and risk management.

Traditionally databases have been largely secured against hackers through network security measures such as firewalls, and network-based detection systems. While network security controls remain valuable in this regard, securing the database systems themselves, and the programs/functions and data within them, has arguably become more critical as networks are increasingly opened to wider access, in particular access from the Internet.

Change and access logging records, who accessed which attributes, what was changed, and when it was changed. Logging services allow for a forensic database audit later by keeping a record of access occurrences and changes. Sometimes application-level code is used to record changes rather than leaving this to the database. Monitoring can be set up to attempt to detect security breaches.

A. Importance of Security in Database Environment

Database security is the protection of the database against intentional and unintentional threats that may be computer-based or non-computer-based. Database security is the business of the entire organization as all people use the data held in the organization's database and any loss or corruption to data would affect the day-to-day operation of the organization and the performance of the people. Therefore, database security encompasses hardware, software, infrastructure, people and data of the organization.

Now there is greater emphasis on database security than in the past as the amount of data stored in corporate database is increasing and people are depending more on the corporate data for decision-making, customer service management, supply chain management and so on. Any loss or unavailability to the corporate data will cripple today's organization and will seriously affect its performance. Now the unavailability of the database for even a few minutes could result in serious losses to the organization.

There are four key issues in the security of databases just as with all security systems:

a) Availability

Data needs to be made available at all necessary times and only to authorize users. Provisions must be made to figure out who has access to what and who has accessed what data.

b) Authenticity

DBMS must be able to confirm that users accessing the system are who they say they are. It needs to ensure that the data has been edited by an authorized source. It needs to verify that all report requests are from authorized users only. It needs to verify that any outbound data is going to the expected receiver.

c) Integrity

DBMS needs to verify that all input data is accurate and verifiable. It needs to ensure that data is following the correct work flow rules for the requirement of the corporation using it. It needs to be able to report on all data changes and who authored them to ensure compliance with laid down rules and privacy laws.

d) Confidentiality

DBMS needs to ensure that confidential data is only available to correct people. It has to ensure that entire database is secured from external and internal system breaches. It needs to provide for reporting on who has accessed what data and what they have done with it. Mission critical and Legal sensitive data must be highly secure at the potential risk of lost business and litigation. [9]

B. Security Levels

To protect the database, we must take security measures at several levels, which are mentioned herein below:

• Physical: The sites containing the computer systems must be secured against armed or surreptitious entry by intruders.

• Human: Users must be authorized carefully to reduce the chance of any such user giving access to an intruder in exchange for a bribe or other favors.

•Operating System: No matter how secure the database system is, weakness in operating system security may serve as a means of unauthorized access to the database.

• Network: Since almost all database systems allow remote access through terminals or networks, software-level security within the network software is as important as physical security, both on the Internet and in networks private to an enterprise.

• Database System: Some database-system users may be authorized to access only a limited portion of the database. Other users may be allowed to issue queries, but may be forbidden to modify the data. It is responsibility of the database system to ensure that these authorization restrictions are not violated.

Security at all these levels must be maintained if database security is to be ensured. A weakness at a low level of security (physical or human) allows circumvention of strict high level (database) security measures.

VI. Acknowledgment

We would like to thank our parents Mr. Ketan Jhaveri and Mrs. Bina Jhaveri for their love, support and encouragement during all times and for always believing in us.
Special acknowledgement to our grand-parents Mr. Kiritbhai Doshi and Mrs. Charuben Doshi.

VII. References

[1] Database System Concepts (Korth, Silberschatz, Sudarshan)
[2] Oracle Docs 11g - Performance Tuning Guide
[3] www.revsys.com/
[4] Oracle Database 11g Performance Tuning Recipes: A Problem-solution Approach. Sam Alapati Darl Kuhn Bill Padfield
[5] www.dba-oracle.com/plsql/
[6] www.dba-oracle.com/art_statspack/
[7] www.codeproject.com/Database-performance-optimization-part-Indexing
[8] www.codeproject.com/Top-steps-to-optimize-data-access-in-SQL-Server
[9] www.cse.psu.edu/
[10] Oracle Database 11g Release 2 Performance Tuning Tips & Techniques.
[11] Database Mirroring Best Practices and Performance Considerations.
[12] Beginning Performance Tuning: Active Session History. Arup Nanda