Programmers and Parallelism

Milan Potdar
Computer Science Department
San Jose State University
San Jose, CA 95192
408-924-1000
milan.potdar@gmail.com

ABSTRACT

A programmer writes programs which can be defined as instruction issued to the system. These instruction can be given to the processing unit of the system or to the storage/database of the system. A programmer can make his system effective by introducing parallelism in both while writing the code or working with database. We would see how parallelism introduced in the system helps improves various performance factors.

Paper will discuss about approaches adopted by programmer to do task using parallelism while writing code and working with database. We will discuss how application having a database can run can have parallelism. We also get a brief information about a MDBMS called ESSABSE and see the parallelism aspects of that in brief.

1. INTRODUCTION

A programmer deals with writing a code to do some function and deals with database to make use to all the information. A programming language is a formal constructed language, designed to communicate instructions to a machine, thus is used to create programs in order to control the behavior of the machine or to express algorithms. There are many programming languages that are developed over time mostly used today are python, java, C/C++, .net etc.

A programmer also have a necessity to deal with database. Database is nothing but organized collection of data. We now have database management systems which are computer applications which interact with the programmer or other applications and capture and analyze data. Well know DBMSs are MySQL, Microsoft SQL, ORACLE and IBM DB2.

Now with advancement we have moved towards a multidimensional database management system called as MDBMS. MDBMS is a database management system that uses a data cube as an idea to represent multiple dimensions of data available to users. This database is optimized for data warehouse and online analytical processing application.

When we talk about parallel processing we will handle the aspects of programming and databases. How parallel processing can affect the performance to make the system better.

2. Programming Language

A programming language is a formal constructed language designed to communicate instructions to a machine, we are mostly going to talk about a computer. It is used to control the behavior of a machine and to do some specific operations. For purpose of parallel processing I would like to go ahead with a specific language.

Java is a computer programming language which is an object oriented programming language and concurrent. It is a write once run anywhere (WORA) language which means that the code runs on one platform and does not need to be recompiled to run on another. The source code is compiled to byte code and runs on JVM (Java virtual machine). James gosling, Mike Sheridan and Patrick Naughton initiated the java language project. Sun microsystem released the first public implementation as java 1.0 in 1995.

Java provides concurrency which allows us to run several programs or various parts of same program to run in parallel. Now look at it this way, if there is a time consuming task which can be performed asynchronously then to improve the performance and increase the throughput we can just execute this in parallel with other tasks.

3. Database

Databases are designed to offer an organized mechanism for storing, managing and retrieving information. They do so through the use of tables. Database tables consists of columns and rows. Each column contains a different type of attribute and each row corresponds to a single record also referred as tuples. Each table has an entity or object that is in a tabular format consisting of columns and rows. Refer figure 1.

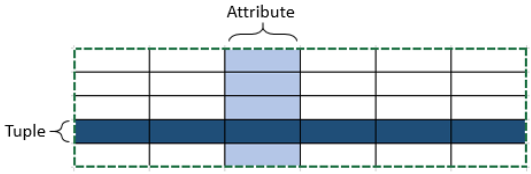


Figure 1. Relatioanl database, attribute and tuples.

SQL TERM	Relational Database Term	Description
Row	Tuple or Record	A data set representing a single term.
Column	Attribute or Field	A labeled element of a tuple.
Table	Relation	A set of columns and rows.

Table 1. SQL Terms, RDBMS terms and description.

Adding a new relationship requires creating a new database structure, a relational database thus provide a way to easily manipulate this database. Thus the concept of relational database management system came into existence. RDBMS is based on the relational model introduced by E.F. Codd.

4. Parallelism in Java

Parallel computing is done in order to achieve high speed and high performance computation. Parallel computing is to break the main task into smaller units and simultaneously execute to achieve the results. It was considered to be a difficult task due to the overhead in managing threads.

Concurrent programming in java consisted of writing threads through the java.lang.Thread class and the java.lang.Runnable interface.

4.1 Fork/Join Tasks

In parallel computing, the fork-join model is a way of setting up and executing parallel programs, in a way that executions branches off in parallel at designated points in the program to "join" at a subsequent point and resume sequential execution.

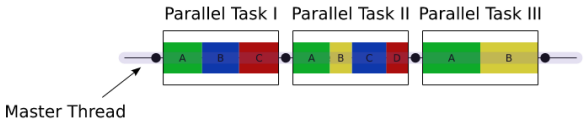


Figure 2. Sequential Execution[3]

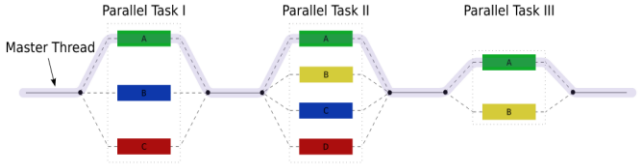


Figure 3. Fork-Join Execution[3]

Above is the illustration of a fork-join paradigm. The three region of the program permits parallel execution of the variously colored blocks.

Some task follow algorithms that requires tasks to create subtasks and communication with each other to complete. Those are the "divide and conquer" algorithm, also called as "map and reduce". The above concept can be understood with the help of example. Consider a huge array of integer in order to computer the sum.

1. Split the array in smaller portions where concurrent threads compute partial sums.
2. This partial sum can now be added to compute the total sum.

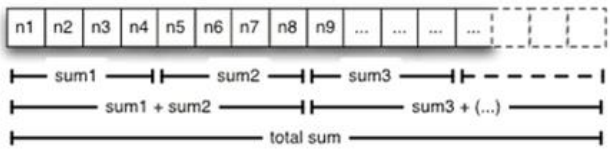


Figure 4. Partial Sums over an Array of Integers

The fork/join framework is added to java.util.concurrent has helped in dealing with the concurrency and later helped in parallelism in Java SE 7.

Fork/Join Framework adds two main classes to java.util.concurrent package which are listed below:

1. ForkJoinPool
2. ForkJoinTask

ForkJoinPool: Its an executor class which exeutes ForkJoinTask instances. The main principle is "Work Stealing", it attempts to find and execute subtasks, which were already created and are in active state currently. It is beenficial to use this over other threads as its very light weight when compared to java Threads.

ForkJoinTask when onces started, it starts subtasks and wiat for it to finish its execution.The executors responisbilty is to assign the subtasks to another thread with in the Thread pool, which is currently waiting for some task to be completed. The active threads in the thread pool try to execute some other subtask created by other tasks. Based on the number of processors available, the ForkJoinPool tries to work with that level of parallelism by maintaining adequate active threads at any given point of time.

Two main methods to be mentioned here are:

1. Fork(): Decides asynchronous execution of the forkJoinTask. Results in creation of new task.
2. Join(): This resturn the output of the computation after completion. This allows a task to wait for the completion of another task.

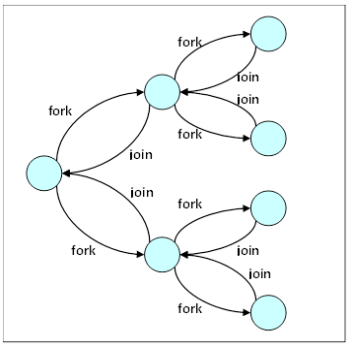


Figure 5. Fork/Join Model

Informal test of execution time and speed up was conducted by oracle to find out the performance of single thread execution and fork/join execution time and the speed up fro a program to count the occurance of words in a document. The reuslt of the test is given in the table below:

Number of Cores	Single-Thread Execution Time (ms)	Fork/Join Execution Time (ms)	Speedup
2	18798	11026	1.704879376
4	19473	8329	2.337975747
8	18911	4208	4.494058935
12	19410	2876	6.748956885

Table 2. Performance Comparision

As we can see a near- linear speedup in the number of cores with minimal effrost,because the fork/join framework takes care of maximizing parallelism.

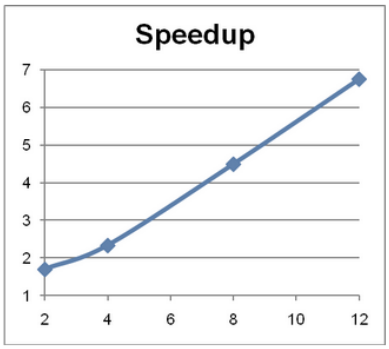


Figure 6 : Speedup (vertical Axis) to number of cores (Horizontal Axis)

4.2 Threads

Java is a multithreaded programming language which means we can develop multithreaded programs using java. It contains 2 or more parts that can run concurrently and each part can handle different task at the same time making optimal use of the available resources especially when computer has multiple CPUs.

By definition multitasking is when multiple processes share common processing resources such as a CPU. Multithreading extends the idea of multitasking into applications where you can subdivide specific operations within a single application into individual threads. Each of the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application.

Multithreading enables you to write in a way where multiple activities can proceed concurrently in the same program.

4.2.1 Life Cycle of Thread

A thread can go through 5 states in its life time, but we can consider one more state in the life cycle of threads. All 6 are are follows:

1. NEW
2. WAITING
3. RUNNABLE
4. TIMED WAITING
5. TERMINATED
6. NON RUNNABLE(BLOCKED)

4.2.1.1 NEW

The thread is in new state if an we create an instance of the thread class but before invocation of start() method.

4.2.1.2 WAITING

Sometimes, a thread transitions to the waiting state while the thread waits for another thread to perform a task. A thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing.

4.2.1.3 RUNNABLE

The Thread is in runnable state after invocation of start() method, but the thread scheduler had not sleected it to be the runnig thread.

4.2.1.4 TIMED WAITING

A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs.

4.2.1.5 TERMINATED

A runnable thread enters the terminated state when it completes its task or otherwise terminates. Here run () exits.

4.2.1.6 NON RUNNABLE

This is the state when the thread is still alive, but is currently not eligible to run.

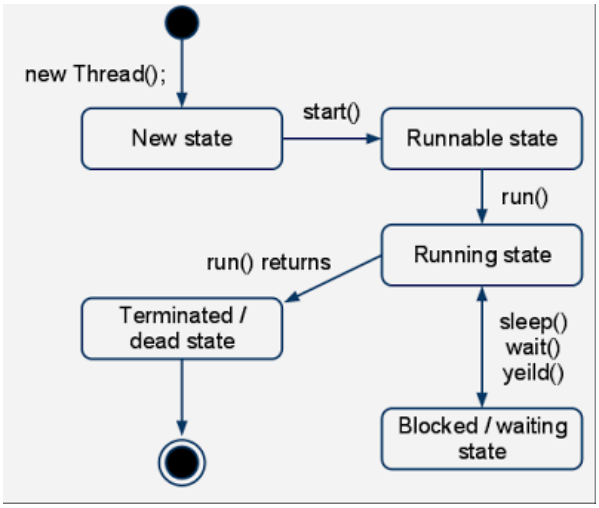


Figure 7: Life Cycle of Thread

4.2.2 Thread class, Constructor, Runnable Interface

A thread can be created by extending Thread class or and by implementing Runnable Interface. It provides constructors and methods to create and perform operations on a thread.

Three is a start() method of thread class that is used to start a newly created thread. It moves the thread from new state to runnable state.

Runnable interface: It should be implemented by any class whose instances are intended to be executed by a thread. It only has one methods named run().

1. public void run(): it is used to perform action for a thread.

4.2.2.1 CONSTRUCTORS

1. Thread()
2. Thread(String name)
3. Thread(Runnable r)
4. Thread(Runnable r, String name)

4.2.2.2 METHODS

1. ***public void run():*** is used to perform action for a thread.
2. ***public void start():*** starts the execution of the thread.JVM calls the run() method on the thread.
3. ***public void sleep(long miliseconds):*** Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds.
4. ***public void join():*** waits for a thread to die.
5. ***public void join(long miliseconds):*** waits for a thread to die for the specified miliseconds.
6. ***public int getPriority():*** returns the priority of the thread.
7. ***public int setPriority(int priority):*** changes the priority of the thread.
8. ***public String getName():*** returns the name of the thread.
9. ***public void setName(String name):*** changes the name of the thread.
10. ***public Thread currentThread():*** returns the reference of currently executing thread.
11. ***public int getId():*** returns the id of the thread.
12. ***public Thread.State getState():*** returns the state of the thread.
13. ***public boolean isAlive():*** tests if the thread is alive.
14. ***public void yield():*** causes the currently executing thread object to temporarily pause and allow other threads to execute.
15. ***public void suspend():*** is used to suspend the thread(depricated).
16. ***public void resume():*** is used to resume the suspended thread(depricated).
17. ***public void stop():*** is used to stop the thread(depricated).
18. ***public boolean isDaemon():*** tests if the thread is a daemon thread.
19. ***public void setDaemon(boolean b):*** marks the thread as daemon or user thread.
20. ***public void interrupt():*** interrupts the thread.
21. ***public boolean isInterrupted():*** tests if the thread has been interrupted.
22. ***public static boolean interrupted():*** tests if the current thread has been interrupted.

5. Parallelism in Database

Parallel database systems seeks to improve performance through parallelization of various operation like loading data, building indexes and evaluating queries. Parallel database improves processing and I/O speeds by using multiple CPUs and disks in parallel.

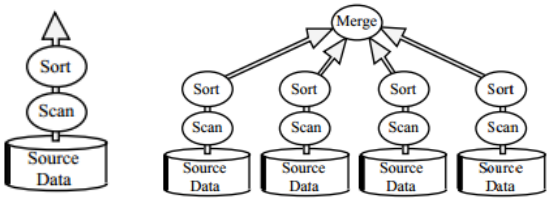
An ideal parallel system demonstrates 2 key properties as listed below:

1. **Linear speedup**: Twice as much hardware can perform the task in half the elapsed time.
2. **Linear scaleup**: Twice as much hardware can perform twice as large a task in the same elapsed time.

In parallel processing, many operations are performed simultaneously with respect to serial processing where computation is sequential. It can be divide into two groups.

1. ***Based on multiprocessor architecture which has following***:
   a. ***Shared memory architecture***: All processors share direct access to a common global memory and to all disk.
   b. ***Shared disk architecture:*** Each processor has a private memory but has direct access to all disks.
   c. ***Shared nothing architecture***: Each memory and disk is owned by some processor that acts as a server for that data. Mass storage in such an architecture is distributed among the processors by connecting one or more disks.
2. ***Hybrid Architecture***
   a. Non-Uniform Memory Architecture
   b. Cluster

Relational queries are ideally suited to parallel execution. Here user apply uniform operation to uniform streams of data. Each operation produces a new relation that can be composed into highly parallel dataflow graphs. By streaming the output of one operation into the input of another operator, the two operators can work in series giving pipelined parallelism. See figure (A)



(A) (B)

By partitioning the input data among multiple processors and memories, an operator can often be split into many independent operators each working on a part of the data. This partitioned data and execution gives partitioned parallelism. See figure (B).

5.1 Relational Database

We will be tackling all the concepts below with regards to the oracle relational database. It was designed to take advantage of the parallel architecture. Databases are used by a large number of users concurrently. A SQL query can be paralleled to achieve higher speed and throughput y using multiple processors. SQL query aims at extracting or updating target data, which is a set of rows and columns based on the constraints. SQL query is divided into database operations can be called as sub operations like SELECT, JOIN, GROUP, SORT, PROJECTION etc. Thus making it best for parallel execution. Thus making RDBMS system ideal for implementations of parallel processing.

One component that plays a huge role in attaining parallelism is the query optimizer. This selects sequence of inputs, joins, and scans to give output. Query optimizer is aware of underlying hardware architecture and is the one which select the path for

parallel execution. Parallel execution will not always increase performance. The scenarios in which it will improve performance in following case:

1. Create Large Indexes,
2. Partitioned index scans
3. Bulk INSERTs, DELETEs and UPDATEs,
4. Aggregation and copying,
5. Large table scans and joins

5.1.1 GRANULE

The basic unit of work in parallelism is a called a granule. Oracle Database divides the operation being parallelized (for example, a table scan, table update, or index creation) into granules. Parallel execution processes execute the operation one granule at a time. The number of granules and their size correlates with the degree of parallelism (DOP).

Types:

1. Block Range Granules.
2. Partition Granules.

5.1.2 PARTITIONING

Oracle provides portioning feature to enhance data access and improve overall application performance significantly. This holds true when applications needs to access table with huge table and indexes. It offer 4 methods for partitioning. They are listed below:

1. Range Partitioning: Range partitioning maps data to partitions based on ranges of partition key values that you establish for each partition. It is the most common type of partitioning and is often used with dates. For example, you might want to partition sales data into monthly partitions.
2. Hash Partitioning: Hash partitioning maps data to partitions based on a hashing algorithm that Oracle applies to a partitioning key that you identify. The hashing algorithm evenly distributes rows among partitions, giving partitions approximately the same size. Hash partitioning is the ideal method for distributing data evenly across devices.
3. List partitioning: List partitioning enables you to explicitly control how rows map to partitions. The advantage of list partitioning is that you can group and organize unordered and unrelated sets of data in a natural way.
4. Composite partitioning: Composite partitioning combines range and hash or list partitioning.

5.1.3 PARALLEL QUERY

SQL query can be divided into sub queries and can be given dedicated separate processors for each sub part. We can use them for performing full-table scans on long tables. Instead of having a single query server to manage the I/O of the table, parallel queries will allow to dedicate many processes to simultaneously access the data.

To get best results, the table should be partitioned onto separate disk devices.

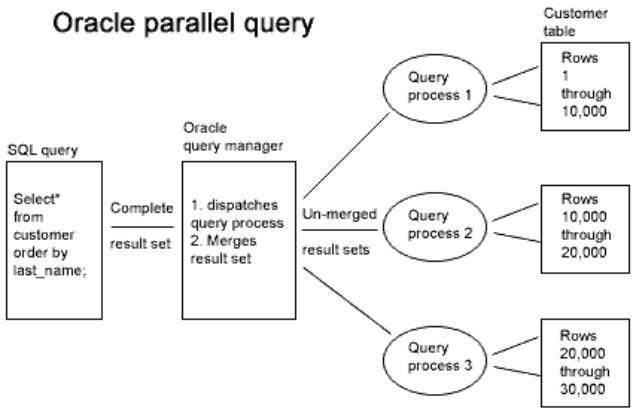


Figure 8. Oracle Parallel Query[1]

The above diagram shows how oracle system will process parallel query. This query can be a sort operation in which case the process will look like something depicted below when parallelism is applied while execution.

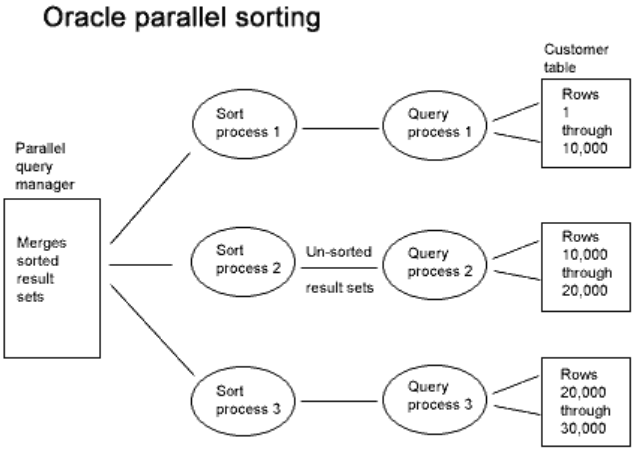


Figure 9. Oracle Parallel Sorting[1]

To invoke a parallel query the most importanat step is that the execution plan of the query specifies a full –table scan. User can specify the upper limit on the number of simultaneous processes and PARALLEL hints can be embedded into the SQL to specify the number of processes.

There can be various parameters that can have direct impact on parallel queries. They are as follows:

1. **sort_area_size**: The higher the value, the more memory available for individual sorts on each parallel process. Note that the sort_area_size parameter allocates memory for every query on the system that invokes a sort.
2. **parallel_min_servers**: This value specifies the minimum number of query servers that will be active on the instance
3. **parallel_max_servers**: This value specifies the maximum number of query servers allowed on the instance.

5.1.4 DECISION TREE OF QUERY PARALLELISM

Step 1: Query is parsed.

Step 2: Database generates a plan (no parallelism)

Step 3: Estimates the execution time

Step 4: Execution time is more than Threshold value time, generate parallel plans.

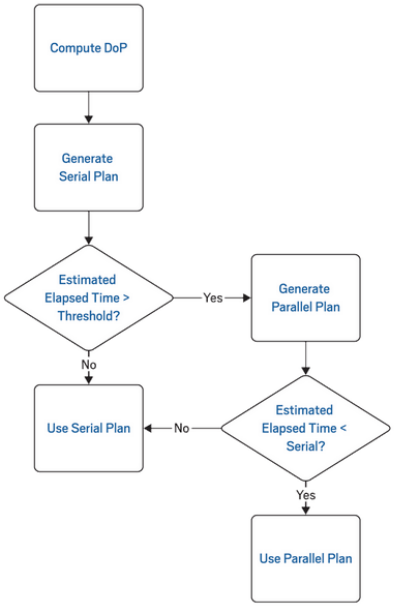


Figure 10. Decision Tree Query Parallelism[1]

5.2 MULTIDIMENSIONAL DATABASE MANAGEMENT SYSTEM(MDBMS)

Under this section we will talk about a new field and type of database management system called as ESSBASE. It is a multidimensiona database system that provies multidimensional database platform upon which we can build analytic applications. Essbase stands for "Extended Spread Sheet DataBASE." The database researcher E. F. Codd coined the term "on-line analytical processing(OLAP)in a white paper which have set out tweleve rules for analytic systems.

Paralell processign can be employed at many places in this multideminsional database. It basically froms a cube to show the multideminsion. Parellelism here can be applied to data load, data export, calc script, restructure.

As a beginner to understand this new MDBMS we will just see the Calc Script parallelism.

5.2.1 CALCULATION SCRIPT

A calculations script, which contains a series of calculation commands, equations, and formulas that allows to define calculations other than those defined by the database outline.

In a calculation script, you can perform a default calculation or a calculation of your choosing. The following tasks can be performed using calculation scripts:

1. Calculate subset of a database.
2. Caluculate the order of the dense and sparse dimensions .
3. Perform complex calculation in a specific order or perform operatiosn that needs multiple iterations through data.
4. Currency conversions and many more.

5.2.2 CALCULATION COMMANDS:

Calculation	**Command**
The entire database, based on the outline	CALC ALL
A specified dimension or dimensions	CALC DIM
All members tagged as two-pass on the dimension tagged as accounts	CALC TWOPASS
The formula applied to a member in the database outline, where *membername* is the name of the member to which the formula is applied	*membername*
All members tagged as Average on the dimension tagged as accounts	CALC AVERAGE
All members tagged as First on the dimension tagged as accounts	CALC FIRST
All members tagged as Last on the dimension tagged as accounts	CALC LAST
Currency conversions	CCONV

Table 3: CalcScript Commands

Essbase is capable to do multithreaded calculations. The Essbase clock storage calculation engine can do this operation using multi-threading and can be configured using handful of Essbase configuration commands and calculation commands.

When we have multiple processor cores with the cycle to spare, turn on the parallel calculation by inserting ***"SETCALCPARALLEL n;"*** into the calculation script where n is the number of threads to use.

Other calculation and configuration commands helps manage parallel calculations is:

"MAXACTIVEUPDATETRANSACTION n"

This dictates the total number of transaction that can be concurrently updating data. If this value is less than your CALCPARALLEL setting then one or more of your update threads from the calculation will be waiting for others to complete.

The ***"SET CALCTASKDIMS n;"*** calculation script command specifies the number of dimensions to use for determining the calculation tasks. By default, Essbase uses the last sparse dimension to determine how it will divide up calculations among multiple threads. If the last dimension is not a consolidating

dimension, you will find that the number of parallel calculations will be low. By using the CALCTASKDIMS calculation setting, you can increase the number of dimensions to use in this calculation.

6. SUMMARY/CONCLUSION

Thus we see from the above text that parallelism is very important and a way to increase the performance of the system. It helps to speed up and make the system works more efficiently. A programmer can do effective programming if he understand the basic concepts of parallelism in the domain of his work. A programmer usually deals with both the coding and the database aspects of the application. Thus knowing parallelism in the system will only help to be better. MDBMS is a new system, it's a BI applications which has a lot of scope of enhancement and parallelism. WE have covered the fork/join concepts of java and multithreading which is the most important concept in JAVA programming language. Then we saw how parallelism can be done in database. We have also seen that at some point parallelism should not be done as it may decrease the system performance. Thus knowing when to use it as a part of the application will play an important role.

7. SUMMARY/CONCLUSION

Thus we see from the above text that parallelism is very important and a way to increase the performance of the system. It helps to speed up and make the system works more efficiently. A programmer can do effective programming if he understand the basic concepts of parallelism in the domain of his work. A programmer usually deals with both the coding and the database aspects of the application. Thus knowing parallelism in the system will only help to be better. MDBMS is a new system, it's a BI applications which has a lot of scope of enhancement and parallelism. WE have covered the fork/join concepts of java and multithreading which is the most important concept in JAVA programming language. Then we saw how parallelism can be done in database. We have also seen that at some point parallelism should not be done as it may decrease the system performance. Thus knowing when to use it as a part of the application will play an important role.

8. REFERENCES

[1] ORACLE documentation, www.docs.oracle.com

[2] Techopedia

[3] www.dba-oracle.com

[4] WIKIPEDIA

[5] www.developers.com