International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161

Available at http://inpressco.com/category/ijcet

Research Article

Study, Comparison and Analysis of Different Types of Storage Systems

Abhay Kumar Agarwal and Agam Srivastava*

Department of Computer Science & Engineering, KNIT, Sultanpur, U.P., India

Accepted 10 Aug 2016, Available online 15 Aug 2016, **Vol.6, No.4 (Aug 2016)**

Abstract

This paper presents the study of different storage systems i.e. File System, Relational Database, distributed database, Parallel database, Object-oriented database, Object-relational database, XML based database and Data warehouse. The advantages and limitations of each type of storage system is presented in the paper. The comparison of various storage system mentioned in the paper is also presented. The comparison is presented on the basis of various parameters like I/O cost, Setup cost, Modularity, etc. Finally, the paper is concluded with the finding on the basis of comparative charts.

Keywords: *Comparison, Storage System, Database, Comparative Behaviour.*

1. Introduction

A Database often abbreviated as DB is an organised collection of data (collection of schema, tables, queries, reports, views and others objects). It is basically a collection of information organized in such a way that a computer program can quickly select desired piece of data, and the software that handles the storage, retrieval and updating of data in a computer system is called as Database Management System.

In the early phase of data and information storage, Flat File System was used for the various applications. With the passage of time, a great change has been observed in terms of Application complexities and capabilities, to deal with this, a new database system called Relational Database System came into existence. This system proves to be strong system since 1980 till date for many applications. With, the continuous growth in the application size, data and information storage leads to several other strong databases like Parallel Database, Distributed Database, Object Oriented Database, Object Relational Database, XML based Database, Data Warehouse, etc.

In this paper, we are discussing various types of databases, their advantages and limitations. The comparative analysis of various discussed databases has also been presented in the paper. The comparison has been made on the various parameters like I/O cost, Reliability, Throughput, Speed, etc.

The rest of the paper is as follows: next section presents detail about Flat File System; in section III Relational Database has been presented; section IV talks about Distributed Database; Parallel Database has been discussed in section V; the detail concept of Object Oriented Database is being presented in section VI; moving further, Object Relational Database is discussed in section VII; XML- based Database and Data Warehouse is presented in section VIII and section IX respectively. Finally, comparative analysis of each database discussed in various sections has been presented in the form of table.

2. Flat File System

A Flat File System is a system in which data is stored in an ordinary file in a computer system. A Flat File can be a binary file or a plain text file. All the information is stored in a file which can be accessed or manipulated after loading the complete file into the memory. Due to loading of complete file in the memory, each file can be accessed by only one user at a time. Usually each record is stored in a file in each line or each record is separated by some delimiters such as comma or multiple space sequences. Each record can have some fixed length and short records are appended with some special characters. Due to addition of these delimiters, certain overheads may arise which may lead to some ambiguity in the file. For Example: Records of Employee working in an organization may be created as:

```
Name,Post,Gender,DOB,DOJ,Address,Mobile
Ajay,Manager,Male,16/02/1985,15/01/2010,125/685 Civil Lines Kanpur-208001,8955215222 ;
Preeti, HR Manager,15/08/1983,16/03/2014, House No. 1578 Ring Road Mumbai-245545,9865877265 ;
Pooja, Business Head,18/08/1888,14/01/1967,H.NO. 22 worli mumbai-535124&&&&,022-2564522;
.
.
.
&&&&&&&& , &&&&&&&& , &&&&& , &&&&& , &&&&&&&&&&&&&&&,&&&&&&&;
```

Figure I: Example of Employee Record in Flat File System

*Corresponding author **Agam Srivastava** is a PG Scholar and **Abhay Kumar Agarwal** is working as Assistant Professor

It can be seen from figure I, that comma (,) is used for separating various fields in each record, semi-colon (;) for separating each record and ampersand (&) is used as padding character. Therefore, here comma, semi-colon and ampersand will never be allowed in the information.

Therefore, flat file database is simply a file which can be prepared manually and stored in a digital format for accessing in a computing environment.

A. Advantages of Flat File Database

- Simple and easy to understand.
- Can be prepared using any standard spreadsheet.
- All records in a single place.
- Suitable for small database.
- Less skill required to use flat file database.
- Less software and hardware requirement.
- User can directly handle the file.

B. Limitations

- Difficult to access required data.
- More I/O cost (I/O cost means transfer of data from main memory to secondary memory or vice versa).
- Very Less degree of concurrency.
- Difficult to provide different levels of security.
- Duplication Problem(Redundancy) as in very large file it can be tedious to find the duplicate record.
- Hard to change format of data fields.
- Difficult to add any data field (Inflexible data design).
- Searching process takes long process time.

3. Relational Database

A Relational Database is a digital database whose organisation is based on the relational model of data, as proposed by E.F. Codd in 1970 (Codd, E.F. (1970). "A Relational Model of Data for Large Shared Data Banks".Communications of the ACM 13 (6):377-387. doi:10.1145/362384.362685).

This model organizes data into one or more tables (or "relations") of rows and columns, with a unique key for each row.

The "*relational*" part of the name comes into play because of mathematical relation. A typical relational database has anywhere from 10 to 1000 tables. Every table contains a column or column that other table can key on to assemble data from that table. One or more data or record characteristics identify with one or numerous records to form functional dependencies. These are classified as follows:

One to One: One table record identifies with another record in another table.
One to Many: One table record identifies with numerous records in another table.
Many to One: More than one table record identifies with a record of another table.
Many to Many: More than one table record relates to more than one record in another table.
Relational database uses "Select", "Project", "Join" database operations, where "Select" is used for data retrieval, "Project" identifies data attributes, and "Join" combines relations.

All tables have its unique key which is used to identify each unique record in the relational database table and referred as Primary Key. A table (referenced relation) which does not have primary key must have some foreign key whose match is there in some other relation (referenced relation).

For Example: Records of Employee working in an organization may be created as shown in the table,

Table 1 Emp Relation

Emp Code	Name	Post	Gender	DOJ	Mobile
001	Ajay	Manager	M	15/01/2010	8953215222
002	Preeti	HR. Mng	F	16/03/2014	9868877263
.					
$n_1n_2n_3$					

Therefore, we can clearly see in Table I that we can make EmpCode field as primary key which make us able to differentiate each record periodically and also make us able to avoid any duplicacy in the primary key.

A. Advantages of RDBMS

- Support for very large database.
- RDBMS have a simple view of the database that conforms to much of the data used in business.
- Data is stored, updated, deleted with the help of primary key.
- Automatic optimization of searching whenever possible.
- More efficient storage.
- Simple to delete or modify details.
- All records in other tables having a link to that entry will show the change (with the help of foreign key).
- Uses SQL (Standard Query Language).
- Better Security which can be achieved by splitting table into two or more tables and assigning different level of security to various tables.

B. Limitations

- Cost of setting up and maintaining the database is high.
- Machine performance reduces when the number of tables between which relationship to be made are large and the table themselves affect the performance in responding to the sql queries.
- Calculations becomes rigid while working with the larger data.

- Often poor support for storage of complex objects.
- Still no productive and viable coordinated backing for things such as text searching within fields.

4. Distributed Database

A distributed database is a database in which parcels of the database are put away on numerous PCs inside of a net-work. Clients have admittance to the bit of the database at their area so they can get to the information important to their undertakings without meddling with the work of others. A centralized disseminated database administration framework (DDBMS) deals with the database as though it were all put away on the same PC. The DDBMS synchronizes all the information occasionally and, in situations where various clients must get to the same information, guarantees that overhauls and erases performed on the information at one area will be naturally reflected in the information put away somewhere else (http://searchoracle.techtarget.com/definition/distributed-database).

Two procedures guarantee that the distributed databases stay cutting-edge and current: replication and duplication.

Replication: Includes utilizing of specific programming tools that searches for changes in the distributive database. Once the progressions have been identified, the replication process makes every one of the databases appear to be identical. The replication procedure can be mind boggling and tedious relying upon the size and number of the distributed databases. This procedure can likewise require a considerable measure of time and PC assets.

Duplication: Duplication has less complexity. It fundamentally recognizes one database as a master and after that copies that database. The duplication procedure is regularly done at a set time night-time. This is to guarantee that each disseminated area has the same information. In the duplication process, clients might change just the expert database. This guarantees neighbourhood information won't be overwritten. Both replication and duplication can keep the data current in all distributive locations.

A. Advantages

- Increase reliability and availability
- Local autonomy or site autonomy — a department can control the data about them (as they are the ones acquainted with it)
- Management of distributed data with various levels of transparency such as network transparency, fragmentation transparency, replication transparency, etc.
- Easier expansion
- Reflects organizational structure — database fragments potentially stored within the departments they relate to
- Protection of valuable data — in the event that there was ever a catastrophic event, for example, a fire, the majority of the information would not be in one spot, but rather distributed in various locations
- Improved performance — data is located near the site of greatest demand, and the database systems themselves are parallelized, permitting load on the databases to be balanced among servers. (A high load on one module of the database won't affect other modules of the database in a distributed database)
- Economics — it may cost less to create a network of smaller computers with the power of a single large computer
- Modularity — systems can be modified, added and removed from the distributed database without affecting other modules (systems)
- Reliable transactions - because of replication of the database.
- Continuous operation, regardless of the fact that a few nodes go offline (depending on design)
- Hardware, operating-system, network, fragmentation, DBMS, replication and location independence
- Single-site failure does not affect performance of system.
- Distributed query processing can improve performance

B. Limitations

- *Complexity: DBA'* might need to do additional work to guarantee that the distributed nature of the system is transparent. Additional work should likewise be done to keep up various disparate systems, rather than one major one. Additional database outline work should likewise be done to represent the disengaged way of the database — for instance, joins turn out to be restrictively costly when performed over multiple systems.
- *Economics:* Increased complexity and a broader infra-structure implies additional work expenses.
- *Security:* Remote database parts(fragments) must be secured, and they are not concentrated so the remote locales must be secured also. The base should likewise be secured (for instance, by encoding the network links between remote sites).
- *Difficult to maintain integrity:* but in a distributed database enforcing integrity over a network might require a lot of the network's assets to be feasible
- *Lack of standards:* there are no tools or approaches yet to offer users some assistance with converting a centralized DBMS into a distributed DBMS.
- *Inexperience:* distributed databases are hard to work with, and in such a youthful field there is not much readily available experience in "appropriate" practice
- *Database design more complex:* In addition to conventional database design challenges, the

design of a distributed database needs to consider fragmentation of data, allocation of fragments to particular destinations and data replication.
- Operating system should support distributed environment
- Concurrency Control represents a noteworthy issue. It can be solved by locking and timestamping.
- Additional software is required

5. Parallel Database

A Parallel Database system tries to enhance performance through parallelization of different operations, for example, loading data, building indexes and assessing queries. Although data might be put away in a distributed style, the distribution is administered exclusively by performance considerations (Raghu Ramakrishnan, Database Management System, Third Edition, Chapter 22, Section 22.1.).

Parallel databases enhance processing and input/output speeds by utilizing multiple CPUs and disks in parallel. Centralized and client–server database systems are not sufficiently capable to handle such applications. In parallel processing, numerous operations are performed at the same time, instead of serial preparing, in which the computational steps are performed sequentially. Parallel databases can be generally divided into two groups, the first group of architecture is multi-processor architecture, the alternatives of which are the followings:

Shared Memory Architecture: Where multiple processors share the main memory space.

Shared Disk Architecture: Where every node has its own main memory, however all nodes share mass storage, usually a storage area network. In practice, every node usually has multiple processors.

Shared nothing Architecture: Where each node has its own mass storage as well as main memory.

A. Advantages

- *Speed:* The main advantage to parallel databases is speed. The server breaks up a user database request into parts and dispatches each part to a separate computer. They work on the parts simultaneously and merge the results, passing them back to the user. This speeds up most data requests, allowing faster access to very large databases.
- *Reliability:* A parallel database appropriately arranged can keep on working in spite of the failure of any system in the cluster. The database server detects that a particular system is not responding and reroutes its work to the remaining systems.
- *Capacity:* As more users request access to the database, the computer administrators add more computers to the parallel server, boosting its overall capacity.
- Useful to the application to query extremely large databases and to process an extremely large number of transactions rate (in order of thousands of transactions per second).
- Greater Flexibility

B. Limitations

- More Start-Up cost.
- Complexity increases due to large number of resources.
- Huge Number of resources are required to support parallelism.

6. Object Oriented Database

Object oriented databases are managed by a software called as Object Database Management Systems (ODBMS). Object databases store objects rather than data such as integers, strings or real numbers. Objects are used in object oriented languages such as Smalltalk, C++, Java, and others. Objects basically consist of the following:

Attributes - Attributes are data which defines the characteristics of an object. This data may be simple such as integers, strings, and real numbers or it may be a reference to a complex object.

Methods - Methods define the behavior of an object and are what was formally called procedures or functions.

Therefore, objects contain both executable code and data. There are different characteristics of objects, for example, whether methods or data can be gotten to from outside the objects. We don't consider this here, to keep the definition simple and to apply it to what an object database is. One other term worth mentioning is classes. Classes are used in object oriented programming to define the data and methods the object will contain. The class is like a template to the object. The class does not itself contain data or methods but defines the data and methods contained in the object. The class is used to create (instantiate) the object. Classes may be used in object databases to recreate parts of the object that may not actually be stored in the database. Methods may not be stored in the database and may be recreated by using a class (http://www.comptechdoc.org/independent/database/basicdb/dataobject.html). Two basic methods are used to store objects by different database vendors.

- Each object has a unique ID and is defined as a subclass of a base class, using inheritance to determine attributes.
- Virtual memory mapping is used for object storage and management.

A. Advantages

- *Reduced Maintenance:* Because most of the processes within the system are encapsulated, the

behaviors may be reused and incorporated into new behaviors.

- *Real-World Modelling:* Object-oriented system tend to model the real world in a more complete fashion than do traditional methods. Objects are organized into classes of objects, and objects are associated with behaviors. The model is based on objects, rather than on data and processing.
- *High Code Reusability:* At the point when new object is made, it will consequently acquire the properties of the class from which it was generated. The new object will likewise acquire the data and behaviors from all super classes in which it participates.
- *Improved Reliability and Flexibility:* Object-situated system guarantee to be much more reliable than convectional systems, basically on the grounds that new behaviors can be "built" from existing objects. Because objects can be progressively called and accessed, new objects might be made whenever required. The new objects may inherit data attributes from one, or numerous different objects. Behaviors might be acquired from super-classes, and novel behaviors might be included without affecting existing systems working.

B. Limitations

- *Lack of universal data model:* There is no universally agreed data model for an OODBMS, and most models lack a theoretical foundation. This disadvantage is seen as a significant drawback, and is comparable to pre-relational systems.
- *Competition:* Maybe a standout amongst the most noteworthy issues that face OODBMS vendors is the competition faced by the RDBMS and the emerging ORDBMS products
- Query optimization bargains encapsulations: Query optimization requires a comprehension of the basic usage to access the database efficiently.
- Locking at object level might affect performance: Many OODBMSs use locking as the basis for concurrency control protocol. In any case, if locking is applied at the object level, locking of an inheritance hierarchy might be risky, and in addition affecting execution.
- Lack of support for views: Presently, most OODBMSs do not provide a view mechanism, which, as we have seen previously, provides many advantages such as data independence, security, reduced complexity, and customization.
- Lack of support for security: Currently, OODBMSs do not to give satisfactory security mechanisms. The user can't grant rights on individual objects or classes.
- Most organizational data are in either file or traditional database format. Converting those data to an ODBMS is also extremely expensive.
- ODBMS don't give effective query and reporting tools. The application software engineer must code query and reporting capabilities into his or her programs which is difficult and costly.

7. Object Relational Database

An Object-Relational Database (ORD), or Object-Relational Database Management System (ORDBMS), is a database management system (DBMS) similar to a relational database, but with an object-oriented database model (objects, classes and inheritance are directly supported in database schemas and in the query language). In addition, like pure relational systems, it supports extension of the data model with custom data-types and methods (Imed Bouchrika, First Edition, Chapter 7, Learn Database Systems with Implementation and Examples).

The main benefit of this type of database lies in the fact that the software to convert the object data between a RDBMS format and object database format is provided. Therefore, it is not necessary for programmers to write code to convert between the two formats and database access is easy from an object oriented computer language.

The basic goal for the ORD is to bridge the gap between relational databases and the object-oriented modelling techniques used in programming languages such as Java, C++, Visual Basic .NET or C#. However, a more popular alternative for achieving such a bridge is to use a standard relational database system with some form of Object-Relational Mapping (ORM) software. Whereas traditional RDBMS or SQL-DBMS products focused on the efficient management of data drawn from a limited set of data-types (defined by the relevant language standards), an object-relational DBMS allows software developers to integrate their own types and the methods that apply to them into the DBMS.

A. Advantages

- *Reuse and Sharing:* The primary focal points of broadening the Relational data model originate from reuse and sharing. Reuse originates from the ability to extend the DBMS server to perform standard functionality centrally, rather to code in every application.
- *Increased Productivity:* ORDBMS gives increased productivity both to the designer and for the, end user.
- *Use of experience in developing RDBMS:* It preserves the noteworthy collection of knowledge and experience that has gone into creating relational applications i.e. in the event that the new functionality is composed suitably, this methodology ought to permit us to exploit the new expansions in a developmental route without losing the advantages of current database features and functionality.

B. Limitations

- The ORDBMS is more complex and thus has increased costs.

- Pure object-oriented database engineers are unhappy with the object-relational terminology which is based on the relational model and not on object-oriented software engineering concepts.
- ORDBMS engineers are data focused while OODB engineers have models which attempt to mirror the real-world (data & behaviour).
- There are defenders of the relational approach that trust the crucial simplicity and virtue of the relational model are lost with these sorts of expansion.
- This conceivably overlooks what's really important of object orientation, highlighting the huge semantic gap between these two technologies.

8. XML Based Database

A XML database is a data persistence software system that permits data to be specified, and sometimes stored, in XML format. These data can be queried, transformed, exported and returned back to a calling system. XML databases are a kind of document oriented databases which are in turn a category of NoSQL database (which means Not (just) SQL). (https://en.wikipedia.org/wiki/XML_database).

It is used to store the enormous amount of information in the XML format. As the utilization of XML is expanding in each field, it is required to have the secured spot to store the XML documents. The data stored in the database can be queried using XQuery, serialized, and exported into desired format. XML databases have picked up in notoriety in the recent years as database vendors, such as, Oracle have included these abilities into baseline of numerous RDBMS.

Steve O'Connell gives one purpose for the the use of XML in databases: the increasingly common use of XML for data transport, which has implied that "information is extracted from databases and put into XML documents and vice-versa". It might demonstrate more productive (in terms of conversion costs) and simpler to store the data in XML format. In content based applications, the ability of the native XML database also minimizes the requirement for extraction or passage of metadata to bolster searching and navigation. There are two noteworthy sorts of XML databases:

- XML- enabled
- Native XML (NXD)

XML- Enabled Database

XML enabled database is nothing but the extension provided for the conversion of XML archive (documents). This is relational database, where data are put away in tables comprising of rows and columns. The tables contain set of records, which thus consist of fields.

Native XML Database

Native XML database is based on the container instead of table organization. It can store large amount of XML documents and data. Native XML database is queried by the XPath-expressions.

Native XML database has advantage over the XML-enabled database. It is exceedingly skilled to store, query and maintain the XML archive(document) than XML-enabled database (http://www. tutorialspoint. com/ xml/xml_databases.htm).

A. Advantages

- An enterprise may have a lot of XML in an existing standard format
- Data may need to be exposed or ingested as XML, so using another format such as relational forces double-modelling of the data
- XML is very well suited to sparse data, deeply nested data and mixed content (such as text with embedded mark-up tags)
- XML is human readable whereas relational tables require expertise to access
- Metadata is often available as XML
- Semantic web data is available as RDF/XML

B. Limitations

- XML databases runs slower
- XML searches are also slow because XML documents are built into databases via document trees, and search must go through all branches of the tree before completing unless the search node is written to look for all related nodes and only search-related nodes.
- XPath is not very easy to learn. Whereas SQL is well known by many, not so many developers and database manager masters XPath.
- XML database do not have referential integrity to ensure that data stays where it was placed for storage, which can cause data references to be lost.
- It requires the entire data set to be loaded into the database before it can be viewed, so it cannot be checked in part without loading the entire database.

9. Data Warehouse

A Data Warehouse is a centralized repository that stores data from different data sources and transforms them into a typical, multidimensional data model for efficient querying and analysis.

Different individuals have different definitions for a Data Warehouse. The most intense definition originated from Bill Inmon who give the accompanying: *"A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process".*

a) Subject Oriented: A DW can be used to break down a specific branch of knowledge. For Example, "sales" can be a specific subject.

b) *Integrated:* A DW incorporates information from various data sources. For Example, source A and source B might have distinctive methods for recognizing an item, but in a data warehouse, there will be just a solitary method for distinguishing an item.
c) *Time –Variant:* Historical information is kept in a data warehouse. For Example, one can recover information from 3 months, 6 months, 12 months or even more established information from a warehouse. This agreement with a transaction system, where regularly just the latest information is kept. For Example, a transaction system might hold all addresses as associated with a customer.
d) *Non-volatile:* Once the data is in the DW, it won't change. So, historical data in a data warehouse should never be altered (http://www.1keydata.com/datawarehousing/data-warehouse-definition.html).

An operational database experiences continuous changes every day by virtue of the transactions that take place. Assume a business official needs to break down past feedbacks on any data such as an item, a supplier, or any customer data, then the official will have no data available to examine in light of the fact that the past information has been upgraded because of transactions.

A data warehouses provides us generalized and consolidated data in multidimensional view. Along with generalized and consolidated view of data, a data warehouse also provide us Online Analytical Processing (OLAP) tools. These tools help us in intelligent and effective investigation of data in a multidimensional space. This examination results in data generalization and data mining.

Data mining functions such as, association, clustering, classification, prediction can be integrated with OLAP operations to upgrade the interactive mining of knowledge at different level of abstraction. That is the reason data warehouse has now turned into an essential platform for data analysis and online analytical processing.

Understanding a Data Warehouse

- A data warehouse is a database, which is kept sepa-rate from the organization's operational database.
- A data warehouse helps executives to organize, understand, and use their data to take strategic decisions.
- Data warehouse systems help in the integration of diversity of application systems.
- It possesses consolidated historical data, which helps the organization to analyse its business.
- A data warehouse system helps in consolidated historical data analysis.
- An operational database query allows to read and modify operations, while an OLAP query needs only read only access of stored data.
- Why a Data Warehouse is Separated from Operational Databases
- A data warehouses is kept separate from operational databases due to the following reasons:
- An operational database is constructed for well-known tasks and workloads such as searching particular records, indexing, etc. In contract, data warehouse queries are often complex and they present a general form of data.
- Operational databases support concurrent processing of multiple transactions. Concurrency control and recovery mechanisms are required for operational databases to ensure robustness and consistency of the database.
- There is no frequent updating done in a data warehouse.
- An operational database maintains current data. On the other hand, a data warehouse maintains historical data.

A. Advantage

- A data warehouse delivers enhanced business Intelligence by providing historical facts and data which can be used for taking business decision.
- It saves time as business users can quickly access critical data from a number of sources (all in one place) and does not waste time in retrieving data from multiple sources.
- It also enhances data quality and consistency of data as DW implementation includes the conversion of data from numerous source systems into a common format. Since each data from the various departments is standardized, each department will produce results that are in line with all other department.
- It provides Historical Intelligence as it stores a large amount of historical data so that we can analyse different time periods and trends in order to make future predictions.
- It reduces cost to access historical data as all the data is available at one place only.
- It also supports ad hoc reporting and inquiry.
- It removes informational processing load from transaction-oriented databases.

B. Limitations

- Data Warehouse is high maintenance systems and results in high maintenance cost.
- Sometimes concealed issue associated with the source systems feeding the data warehouse might be identified after years of being undetected.
- Complexity of Integration is the extreme assignment as an organization must invest a lot of time deciding how well the different diverse data warehousing tools can be coordinated into the overall solution that is required.

Table 2 Comparative Chart of the various types of storage systems

S.No	Parameters	Flat File	Relational	Distributed	Parallel	Object Oriented	Object Relational	XML based	Data Warehouse
1	**I/O Cost**	High	Moderate	High	High	Moderate	Moderate	Very Low	Very High
2	**Setup Cost**	Low	Moderate	High	High	High	High	Very Low	Very High
3	**Modularity**	Very Low	Low	High	High	High	High	Low	Very High
4	**Concurrency**	Very Low	Moderate	High	High	High	High	High	Very High
5	**Design Complexity**	Very Low	Low	High	High	Low	Low	High	Very High
6	**Security**	Very Low	Low	High	High	Low	Moderate	Low	Very High
7	**Flexibility**	Very Low	Low	Moderate	High	High	High	Low	Very High
8	**Maintenance**	Low	Low	High	High	Low	Moderate	High	Very High
9	**Reliability**	Low	Moderate	High	High	High	High	Low	Very High
10	**Reuse and Sharing**	Very Low	Low	Moderate	High	High	High	Low	Very High
11	**Intelligence**	None	Very Low	Moderate	Moderate	Low	Low	Moderate	Very High
12	**Throughput**	Very Low	Low	High	High	Moderate	High	Low	Very High
13	**Speed**	Very Low	Low	Moderate	High	Moderate	Moderate	Low	Very High

10. Comparative Analysis of Various Databases with Reference To Various Parameters

This section presents the comparative chart of the various types of storage systems as shown in table II. The comparison has been made on the various parameters like, I/O cost, Setup cost, Modularity, etc. For each parameter, the comparison has been done the term Very Low, Low, Moderate, High and Very High. For the parameters I/O cost, Setup cost, Design Complexity and Maintenance, the storage system with the value low for the specified parameters means the storage system is better than other while for rest of the others the meaning is vice-versa.

Conclusion

In this paper, we have discussed eight different types of storage systems which are Flat file system, RDBMS, Distributed DBMS, Parallel DBMS, Object-oriented DBMS, Object-oriented DBMS, Object-relational DBMS, XML-based system and Data warehouse; along with their advantages and disadvantages in the very first part.

In the second part, we have shown a comparative table of all discussed databases with some crucial factors. Comparative table shows how one database is similar to other and how different from other with respect to specific parameters.

We have seen that the overall throughput in case of File System is very low and in case of Relational and XML based database is very low. Distributed System and Parallel Databases provide much higher throughput. Object Relational database provide more through rate than Object oriented database; which in case of Data Warehouse is very much high.

Also, speed is very high in case of Data Warehouse and will give the best speed for large application processing than any of discussed database.

Resource Sharing and reusing capability is more seen in Data Warehouse and Parallel, Object Oriented and Object Relational database.

Design Complexity is very low in case of File System, low in Relational, Object Oriented and Object Relational, whereas it is High in Parallel, Distributed, XML based databases. And Very High in case of Data Warehouse.

Security can well have achieved in case of Distributed, Parallel and in Data warehouse. Also, it can be achieved in Relational at various level and XML based databases.

In terms of Concurrency, only Flat File System gives the worst scenario otherwise all of them give satisfactory performance, which in case of Data warehouse is very High.

In a nut-shell, we can say that Data warehouse is the most efficient, reliable, flexible, secure database which is most suitable for today's large and heavy application instead of involving high cost and maintenance. XML based databases are also capable of serving many of the needs of current customer but due to its slow speed it somewhere lacks down. But many Database vendors are using the advantages of XML and incorporating into the standard system so to utilize the power of XML.

And for medium and small application we can choose any of them except Flat File System due to its so many limitations. So, one must decide the database according to its need and application.

References

Codd, E.F. (1970). A Relational Model of Data for Large Shared Data Banks, Communications of the ACM 13 (6):377- 387. doi:10.1145/362384.362685

http://searchoracle.techtarget.com/definition/distributed-database

Raghu Ramakrishnan, Database Management System, Third Edition, Chapter 22, Section 22.1.

http://www.comptechdoc.org/independent/database/basicdb/dataobject.html

Author Imed Bouchrika, First Edition, Chapter 7,Learn Database Systems with Implementation and Examples,

https://en.wikipedia.org/wiki/XML_database

http://www.tutorialspoint.com/xml/xml_databases.htm

http://www.1keydata.com/datawarehousing/data-warehouse-definition.html