Chapter 11

Advanced Database Concepts

11.1 Object Oriented Model

The Object-Oriented Model in DBMS is the data model where data is stored in the form of objects. This model is used to represent real-world entities. The data and data relationship is stored together in a single entity known as an object in the Object Oriented Model. The Object-Oriented Database Management System is built on top of Object-Oriented Model.

We can use the Object-Oriented Model in DBMS to store real-world entities. Here, we can store pictures, audio, video, and other types of data.

Components of Object-Oriented Data Model

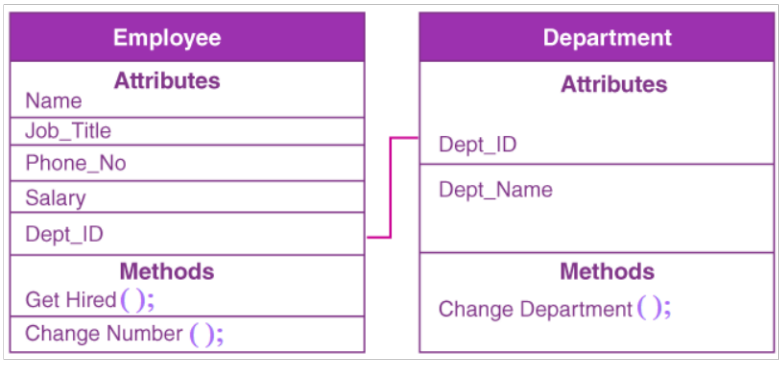
Object attribute- The objects have certain characteristics. These are known as the attributes of the object.

Object method- The object's behavior is shown using object methods.

Class- It is a collection of similar kinds of objects. It is an entity that has attributes and methods together.

Inheritance- It is the ability of the object within the class hierarchy to inherit the attributes and methods of the classes above it. A new class can be derived from an existing class, the new class has the attributes and methods described in the existing class and also has its attributes and methods. This helps in code reusability.

Example:



Advantages

- Codes can be reused due to inheritance.
- Easily understandable.
- Cost of maintenance can reduced due to reusability of attributes and functions because of inheritance.

Disadvantages:

- It is not properly developed so not accepted by users easily.

11.2 Object Relational Model (ORM)

An Object relational model is a combination of a Object oriented database model and a Relational database model. So, it supports objects, classes, inheritance etc. just like Object Oriented models and has support for data types, tabular structures etc. like Relational data model.

One of the major goals of Object relational data model is to close the gap between relational databases and the object-oriented practices frequently used in many programming languages such as C++, C#, Java etc.

Advantages of Object Relational model

The advantages of the Object Relational model are −

- **Inheritance**

The Object Relational data model allows its users to inherit objects, tables etc. so that they can extend their functionality. Inherited objects contain new attributes as well as the attributes that were inherited.

- **Complex Data Types**

Complex data types can be formed using existing data types. This is useful in Object relational data model as complex data types allow better manipulation of the data.

- **Extensibility**

The functionality of the system can be extended in Object relational data model. This can be achieved using complex data types as well as advanced concepts of object oriented model such as inheritance.

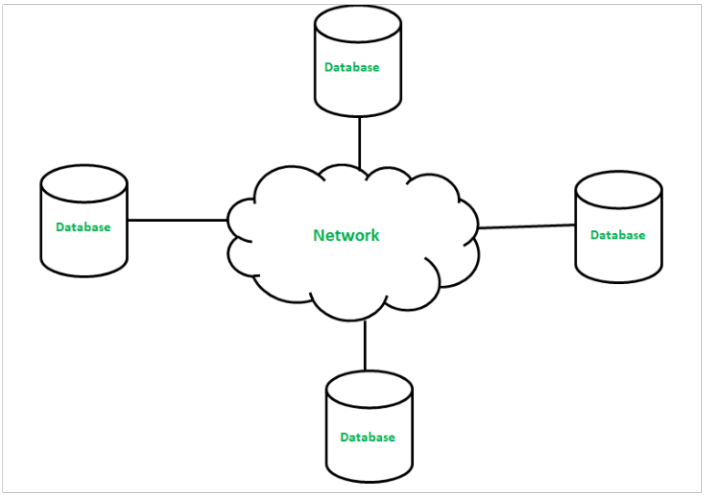
Disadvantages of Object Relational model

The object relational data model can get quite complicated and difficult to handle at times as it is a combination of the Object-oriented data model and Relational data model and utilizes the functionalities of both of them.

11.3 Distributed Database

A distributed database is basically a type of database which consists of multiple databases that are connected with each other and are spread across different physical locations. The data that is stored in various physical locations can thus be managed independently of other physical locations. The communication between databases at different physical locations is thus done by a computer network.

This may be required when a particular database needs to be accessed by various users globally. It needs to be managed such that for the users it looks like one single database.



Types:

1. Homogeneous Database:

In a homogeneous database, all different sites store databases identically. The operating system, database management system, and the data structures used – all are the same at all sites. Hence, they're easy to manage.

2. Heterogeneous Database:

In a heterogeneous distributed database, different sites can use different schema and software that can lead to problems in query processing and transactions. Also, a particular site might be completely unaware of the other sites. Different computers may use a different operating system, different database application. They may even use different data models for the database.

Distributed Data Storage

There are two ways in which data can be stored at different sites. These are,

i) **Replication:** As the name suggests, the system stores copies of data at different sites. If an entire database is available on multiple sites, it is a fully redundant database.

ii) **Fragmentation**

In Fragmentation, the relations are fragmented, which means they are split into smaller parts. Each of the fragments is stored on a different site, where it is required. In this, the data is not replicated, and no copies are created.

The prerequisite for fragmentation is to make sure that the fragments can later be reconstructed into the original relation without losing any data.

There are two types of fragmentation,

Horizontal Fragmentation – Splitting by rows.

Vertical fragmentation – Splitting by columns.

- ✓ **Horizontal Fragmentation**

The relation schema is fragmented into group of rows, and each group is then assigned to one fragment.

- ✓ **Vertical Fragmentation**

The relation schema is fragmented into group of columns, called smaller schemas. These smaller schemas are then assigned to each fragment.

Each fragment must contain a common candidate key to guarantee a lossless join.

Advantages of Distributed Database

Better Reliability: Distributed databases offers better reliability than centralized databases. When database failure occurs in a centralized database, the system comes to a complete stop. But in the case of distributed databases, the system functions even when a failure occurs, only performance-related issues occur which are negotiable.

Modular Development: It implies that the system can be expanded by adding new computers and local data to the new site and connecting them to the distributed system without interruption.

Lower Communication Cost: Locally storing data reduces communication costs for data manipulation in distributed databases. In centralized databases, local storage is not possible.

Better Response Time: As the data is distributed efficiently in distributed databases, this provides a better response time when user queries are met locally. While in the case of centralized databases, all of the queries have to pass through the central machine which increases response time.

Disadvantages of Distributed Database

Costly Software: Maintaining a distributed database is costly because we need to ensure data transparency, coordination across multiple sites which requires costly software.

Large Overhead: Many operations on multiple sites require complex and numerous calculations, causing a lot of processing overhead.

Improper Data Distribution: If data is not properly distributed across different sites, then responsiveness to user requests is affected. This in turn increases the response time.

11.4 Data warehouse

A data warehouse is a centralized repository for storing and managing large amounts of data from various sources for analysis and reporting. It is optimized for fast querying and analysis, enabling organizations to make informed decisions by providing a single source of truth for data. Data warehouse typically involves transforming and integrating data from multiple sources into a unified, organized, and consistent format.

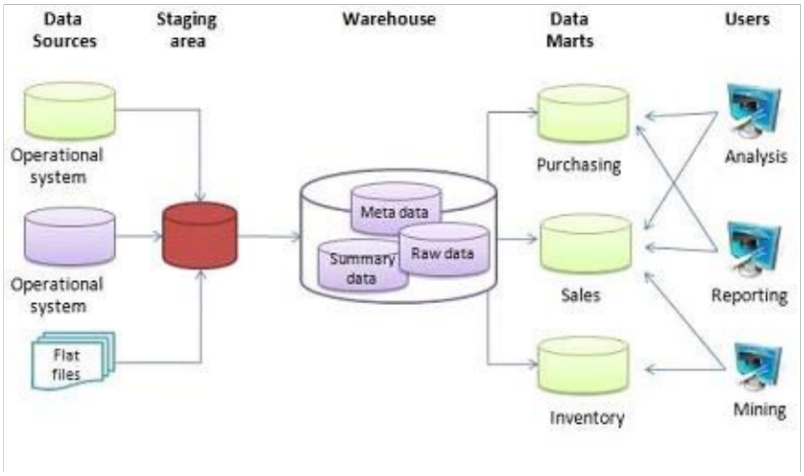


Figure: Architecture of Data -Warehouse

The above image shows a data warehouse. The key terms used in data warehouse is described below.

Operational System: An operational system is a method used in data warehousing to refer to a system that is used to process the day-to-day transactions of an organization.

Flat Files: A Flat file system is a system of files in which transactional data is stored, and every file in the system must have a different name.

Meta Data: A set of data that defines and gives information about other data. Meta Data summarizes necessary information about data, which can make finding and work with particular instances of data more accessible. For example, author, data build, data changed, and file size is examples of very basic document metadata.

Metadata is used to direct a query to the most appropriate data source.

We must clean and process your operational information before putting it into the warehouse.

Staging area: simplifies data cleansing and consolidation for operational methods coming from multiple source systems, especially for enterprise data warehouses where all relevant data of an enterprise is consolidated.

Data Warehouse Staging Area is a temporary location where a record from source systems is copied.

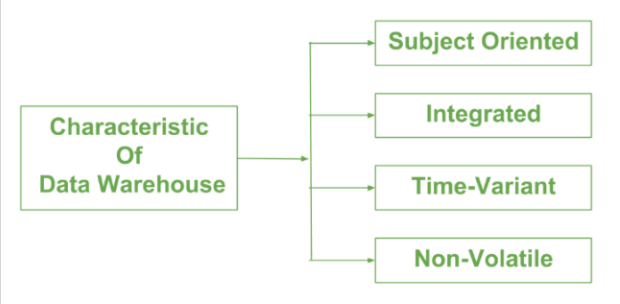
Data mart: We may want to customize our warehouse's architecture for multiple groups within our organization. We can do this by adding data marts.

A data mart is a segment of a data warehouse that can provide information for reporting and analysis on a section, unit, department, or operation in the company, e.g., sales, payroll, production, etc.

Need for Data Warehouse

An ordinary Database can store MBs to GBs of data and that too for a specific purpose. For storing data of TB size, the storage shifted to Data Warehouse. Besides this, a transactional database doesn't offer itself to analytics. To effectively perform analytics, an organization keeps a central Data Warehouse to closely study its business by organizing, understanding, and using its historic data for taking strategic decisions and analyzing trends.

Characteristics of Data Warehouse



Subject-oriented: A data warehouse typically provides information on a topic (such as a sales inventory or supply chain) rather than company operations.

Time-variant: Time variant keys (e.g., for the date, month, time) are typically present.

Integrated: A data warehouse combines data from various sources. These may include a cloud, relational databases, flat files, structured and semi-structured data, metadata, and master data. The sources are combined in a manner that's consistent, relatable, and ideally certifiable, providing a business with confidence in the data's quality.

Persistent and non-volatile: Prior data isn't deleted when new data is added. unctions of Data warehouse: It works as a collection of data and here is organized by various communities that endures the features to recover the data functions. It has stocked facts about the tables which have high transaction levels which are observed so as to define the data warehousing techniques and major functions which are involved in this are mentioned below:

Functions of data warehouse

Data Consolidation: The process of combining multiple data sources into a single data repository in a data warehouse. This ensures a consistent and accurate view of the data.

Data Cleaning: The process of identifying and removing errors, inconsistencies, and irrelevant data from the data sources before they are integrated into the data warehouse. This helps ensure the data is accurate and trustworthy.

Data Integration: The process of combining data from multiple sources into a single, unified data repository in a data warehouse. This involves transforming the data into a consistent format and resolving any conflicts or discrepancies between the data sources. Data integration is an essential step in the data warehousing process to ensure that the data is accurate and usable for analysis. Data from multiple sources can be integrated into a single data repository for analysis.

Data Storage: A data warehouse can store large amounts of historical data and make it easily accessible for analysis.

Data Transformation: Data can be transformed and cleaned to remove inconsistencies, duplicate data, or irrelevant information.

Data Analysis: Data can be analyzed and visualized in various ways to gain insights and make informed decisions.

Data Reporting: A data warehouse can provide various reports and dashboards for different departments and stakeholders.

Data Mining: Data can be mined for patterns and trends to support decision-making and strategic planning.

Performance Optimization: Data warehouse systems are optimized for fast querying and analysis, providing quick access to data.

Example Applications of Data Warehousing

Data Warehousing can be applied anywhere where we have a huge amount of data and we want to see statistical results that help in decision making.

Social Media Websites: The social networking websites like Facebook, Twitter, Linkedin, etc. are based on analyzing large data sets. These sites gather data related to members, groups, locations, etc., and store it in a single central repository. Being a large amount of data, Data Warehouse is needed for implementing the same.

Banking: Most of the banks these days use warehouses to see the spending patterns of account/cardholders. They use this to provide them with special offers, deals, etc.

Government: Government uses a data warehouse to store and analyze tax payments which are used to detect tax thefts.

Old questions solutions from this chapter

Write the short notes on:

- **Object Relational Model (ORM) [PU:2014 spring] [PU:2019 fall]**
- **Distributed Database [2015 fall] [PU:2016 spring] [PU:2017 fall][PU:2022 fall]**
- **Data warehouse [PU:2015 spring]**

(Already solved: see from above contents)

OODBMS (Object Oriented Database Management System) [PU:2011 fall]

OODBMS stands for Object-Oriented Database Management System, which is a type of database management system that is designed to store and manage object-oriented data. Object-oriented data is data that is represented using objects, which encapsulate data and behavior into a single entity.

It stores and manages data as objects, and provides mechanisms for querying, manipulating, and retrieving the data.

In an OODBMS, the data is typically stored in the form of classes and objects, which can be related to each other using inheritance and association relationships.

In an OODBMS, the data is managed using an object-oriented programming language or a specialized query language designed for object-oriented databases. Some of the popular object-oriented database languages include Smalltalk, Java, and C++. Some OODBMS also support standard SQL for querying the data.

Advantages of OODBMS:

Supports Complex Data Structures: It is designed to handle complex data structures, such as inheritance, polymorphism, and encapsulation. This makes it easier to work with complex data models in an object-oriented programming environment.

Improved Performance: It provides improved performance compared to traditional relational databases for complex data models. It can reduce the amount of mapping and translation required between the programming language and the database, which can improve performance.

Reduced Development Time: It can reduce development time since it eliminates the need to map objects to tables and allows developers to work directly with objects in the database.

Supports Rich Data Types: It supports rich data types, such as audio, video, images, and spatial data, which can be challenging to store and retrieve in traditional relational databases.

Scalability: It can scale horizontally and vertically, which means it can handle larger volumes of data and can support more users.

Disadvantages:

Limited Adoption: It is not as widely adopted as traditional relational databases, which means it may be more challenging to find developers with experience working with ODBMS.

Lack of Standardization: It lacks standardization, which means that different vendors may implement different features and functionality.

Cost: It can be more expensive than traditional relational databases since it requires specialized software and hardware.

Integration with Other Systems: It can be challenging to integrate with other systems, such as business intelligence tools and reporting software.

Scalability Challenges: It may face scalability challenges due to the complexity of the data models it supports, which can make it challenging to partition data across multiple nodes.

What is OODBMS? Discuss Distributed database along with necessary diagram.

[PU:2012 fall]

What do you mean by object-oriented model. Write advantages and disadvantages of distributed databases. [PU:2013 spring]

Distributed Database Model [PU:2011 spring] [PU:2013 fall]

Advantages of Distributed database processing [2011 fall]

(Already solved)

What are the major characteristics of a distributed database system? [PU:2010 Spring]

A DDBMS has the following characteristics in common:

- ✓ **It handles a collection of logically related shared data.**

  In distributed database, each site has some data that are same or similar like other sites data. These data can be accessed by any request initiated at any server that is part of the application. Hence, they are shared.

- ✓ **The data are normally split into fragments**.

The data of an entity set is usually divided into many fragments (horizontal, vertical, hybrid) based on the application that accesses the data. This is done based on several factors.

- ✓ **The data fragments may also be replicated (duplicated).**

  Some of the data fragments may be duplicated and maintained in several sites. That is, the data available at site A may also be made available at site B.

- ✓ **The sites are linked by communication network.**

  A distributed database is a database that is spread over multiple sites and linked through communication networks.

- ✓ **The data at each site is controlled by the local DBMS.**

  The site on which the data stored is the owner of the data. And, this data is handled by the DBMS software installed in that site.

- ✓ **The local requests must be handled by local DBMS autonomously.**

As said above, the local DBMS can answer autonomously the queries that are requesting the data stored in that site only. There is no need to consult with other sites.

- ✓ **DBMS at each site must participate in at least one global application.**

  The applications that are accessing the data in a distributed database may access local data as well as global data. If the DBMS is accessing the global data, we would say that as the distributed database. Hence, we need all of our sites to participate in at least one global data access.

Data warehouse vs Data Mining

Data warehouse	Data Mining
A data warehouse is a database system that is designed for analytical analysis instead of transactional work.	Data mining is the process of analyzing data patterns.
Data is stored periodically.	Data is analyzed repeatedly.
Data warehouse is entirely carried out by the engineers.	Data mining is carried out by business users with the help of engineers.
Data warehouse is the process of extracting and storing data that allow easier reporting.	Data mining uses pattern recognition techniques to identify patterns.
Subject-oriented, integrated, time-varying and non-volatile constitute data warehouses.	AI, statistics, databases, and machine learning systems are all used in data mining technologies.
In the data warehouse, there is a high possibility that the data required for analysis by the company may not be integrated into the warehouse. It can simply lead to loss of data.	The data mining techniques are not 100 percent accurate. It may lead to serious consequences in a certain condition.
When a data warehouse is connected with operational business systems like CRM (Customer Relationship Management) systems, it adds value.	Data mining aids in the creation of suggestive patterns of key parameters. Customer purchasing behavior, items, and sales are examples. As a result, businesses will be able to make the required adjustments to their operations and production.

What are data fragmentations? State the various fragmentation with examples. [PU:2019 fall]

Fragmentation is a process of dividing the whole or full database into various subtables or sub relations. The small pieces or sub relations or subtables are called fragments. These fragments are called logical data units and are stored at various sites. It must be made sure that the fragments are such that they can be used to reconstruct the original relation (i.e, there isn't any loss of data). Reconstruction of original data can be done using UNION or JOIN operations. Database fragmentation is of three types:

1. Horizontal fragmentation
2. Vertical fragmentation
3. Mixed or Hybrid fragmentation

1. Horizontal fragmentation

Horizontal fragmentation refers to the process of dividing a table horizontally by assigning each row (or a group of rows) of relation to one or more fragments. These fragments can then be assigned to different sites in the distributed system. Some of the rows or tuples of the table are placed in one system and the rest are placed in other systems. The rows that belong to the horizontal fragments are specified by a condition on one or more attributes of the relation. In relational algebra horizontal fragmentation on table T, can be represented as follows:

$\sigma_p(T)$

where, σ is relational algebra operator for selection.

p is the condition satisfied by a horizontal fragment

Note that a union operation can be performed on the fragments to construct table T. Such a fragment containing all the rows of table T is called a complete horizontal fragment.

For example, consider an EMPLOYEE table (T) :

Empno	Ename	Address	Salary	Dept
101	A	Kathmandu	3000	1
102	B	Butwal	4000	1
103	C	Pokhara	5500	2
104	D	Chitwan	5000	2
105	E	Bhaktapur	2000	2

This EMPLOYEE table can be divided into different fragments like:

EMP1 = $\sigma_{Dept = 1}$ (EMPLOYEE)

EMP2 = $\sigma_{Dept = }$ 2(EMPLOYEE)

These two fragments are: T1 fragment of Dept = 1

Empno	Ename	Address	Salary	Dept
101	A	Kathmandu	3000	1
102	B	Butwal	4000	1

Similarly, the T2 fragment on the basis of Dept = 2 will be:

Empno	Ename	Address	Salary	Dept
103	C	Pokhara	5500	2
104	D	Chitwan	5000	2
105	E	Bhaktapur	2000	2

Now, here it is possible to get back T as T = T1 ∪ T2

2. Vertical Fragmentation

Vertical fragmentation refers to the process of decomposing a table vertically by attributes or columns. In this fragmentation, some of the attributes are stored in one system and the rest are stored in other systems. This is because each site may not need all columns of a table. In order to take care of restoration, each fragment must contain the primary key field(s) in a table. The fragmentation should be in such a manner that we can rebuild a table from the fragment by taking the natural JOIN operation and to make it possible we need to include a special attribute called Tuple-id to the schema. For this purpose, a user can use any super key. And by this, the tuples or rows can be linked together. The projection is as follows:

$\pi_{a1, a2,..., an}(T)$

where, π is relational algebra operator

a1…., an are the attributes of T

T is the table (relation)

For example, for the EMPLOYEE table we have T1 as:

Empno	Ename	Address	Tuple_id
101	A	Kathmandu	1
102	B	Butwal	2
103	C	Pokhara	3
104	D	Chitwan	4
105	E	Bhaktapur	5

For the second. sub table of relation after vertical fragmentation is given as follows:

Salary	Dept	Tuple_id
3000	1	1
4000	1	2
5500	2	3
5000	2	4
2000	2	5

This is T2 and to get back to the original T, we join these two fragments T1 and T2 as (T1 ⋈ T2)

3. Mixed or Hybrid fragmentation

The combination of vertical fragmentation of a table followed by further horizontal fragmentation of some fragments is called mixed or hybrid fragmentation. For defining this type of fragmentation we use the SELECT and the PROJECT operations of relational algebra. In some situations, the horizontal and the vertical fragmentation isn't enough to distribute data for some applications and in that conditions, we need a fragmentation called a mixed fragmentation.

Mixed fragmentation can be done in two different ways:

- ✓ The first method is to first create a set or group of horizontal fragments and then create vertical fragments from one or more of the horizontal fragments.
- ✓ The second method is to first create a set or group of vertical fragments and then create horizontal fragments from one or more of the vertical fragments.

The original relation can be obtained by the combination of JOIN and UNION operations which is given as follows:

$\sigma_P(\pi_{a1, a2..,an}(T))$

$\pi_{a1,a2....,an}(\sigma_p(T))$

For example, for our EMPLOYEE table, below is the implementation of mixed fragmentation is

$\pi_{Ename, Address}(\sigma_{Empno < 104}(EMPLOYEE))$

The result of this fragmentation is:

Ename	Address
A	Kathmandu
B	Butwal
C	Pokhara

Under which condition It will be beneficial to have replication or fragmentation of data? Explain with suitable example. [PU:2014 fall]

Replication and fragmentation of data are two strategies used in database design and distributed systems to improve data availability, reliability, and performance. The choice between these strategies depends on the specific requirements and constraints of the system. Let's explore both strategies and the conditions under which each is beneficial:

Replication of Data:

Replication involves creating multiple copies of the same data and distributing them across different nodes or locations within a distributed system.

Benefits of replication:

Improved fault tolerance: If one copy of the data becomes unavailable due to a node failure, other copies can still be accessed.

Enhanced data availability: Data can be read from the nearest or most available copy, reducing latency for read operations.

Load balancing: Distributing read requests among replicated copies can distribute the query load and improve system performance.

Example: Consider an e-commerce website where product information is frequently accessed by users. To improve user experience, product details (e.g., name, price, description) can be replicated across multiple data centers. If one data center experiences a network issue or hardware failure, users can still access product information from other data centers, ensuring high availability and low latency.

Fragmentation of Data:

- ✓ Fragmentation involves dividing a database into smaller, more manageable pieces called fragments. There are two main types of fragmentation: horizontal and vertical.
- ✓ Horizontal fragmentation divides data into rows, typically based on some condition or attribute value.
- ✓ Vertical fragmentation divides data into columns, often splitting them into related sets of attributes.

Benefits of fragmentation:

Improved scalability: Data can be distributed across multiple nodes, allowing for parallel processing and accommodating a larger volume of data.

Enhanced security: Sensitive data can be fragmented and stored separately from non-sensitive data, reducing the risk of data breaches.

Efficient data storage: Only relevant fragments need to be stored on each node, optimizing storage utilization.

Example: In a social media platform, user data can be horizontally fragmented based on geographical regions. Users from North America might have their data stored in one fragment, while users from Europe have their data in another. This fragmentation enables the system to scale more effectively and improves response times for regional users.