CHANDIGARH ENGINEERING COLLEGE

B.TECH.(CSE 5th Sem.)

Notes Subject: Database Management System

Subject Code: ((BTCS 501-18))

Unit 6

ODBMS:

The ODBMS which is an abbreviation for object-oriented database management system is the data model in which data is stored in form of objects, which are instances of classes. These classes and objects together make an object-oriented data model.

Components of Object-Oriented Data Model:
The OODBMS is based on three major components, namely: Object structure, Object classes, and Object identity. These are explained below.

1. Object Structure:
The structure of an object refers to the properties that an object is made up of. These properties of an object are referred to as an attribute. Thus, an object is a real-world entity with certain attributes that makes up the object structure. Also, an object encapsulates the data code into a single unit which in turn provides data abstraction by hiding the implementation details from the user.

The object structure is further composed of three types of components: Messages, Methods, and Variables. These are explained below.

1. Messages –
   A message provides an interface or acts as a communication medium between an object and the outside world. A message can be of two types:
   - Read-only message: If the invoked method does not change the value of a variable, then the invoking message is said to be a read-only message.
   - Update message: If the invoked method changes the value of a variable, then the invoking message is said to be an update message.

2. Methods –
   When a message is passed then the body of code that is executed is known as a method. Whenever a method is executed, it returns a value as output. A method can be of two types:
   - Read-only method: When the value of a variable is not affected by a method, then it is known as the read-only method.
   - Update-method: When the value of a variable change by a method, then it is known as an update method.

3. Variables –
   It stores the data of an object. The data stored in the variables makes the object distinguishable from one another.

2.Object Classes:
An object which is a real-world entity is an instance of a class. Hence first we need to define a class

and then the objects are made which differ in the values they store but share the same class definition. The objects in turn correspond to various messages and variables stored in them.

ORDBMS

An object–relational database can be said to provide a middle ground between relational databases and object-oriented databases. In object–relational databases, the approach is essentially that of relational databases: the data resides in the database and is manipulated collectively with queries in a query language; at the other extreme are OODBMSes in which the database is essentially a persistent object store for software written in an object-oriented programming language, with an application programming interface API for storing and retrieving objects, and little or no specific support for querying. A database management system called an ORDBMS is a cross between an OODBMS and an RDBMS, or relational database management system.

Both of those models have advantages and disadvantages. A DBMS can benefit from certain aspects of each model's characteristics by integrating the two.

Widely acknowledged as a standard or preeminent system, ORDBMS (Object-Relational Database Management System). However, there have been initiatives to mix the advantages of relational and object-oriented databases.

Functionalities of ORDBMS:

Object-Relational Mapping (ORM): ORDBMS frequently use libraries or frameworks for object-relational mapping to connect the worlds of relational databases with object-oriented programming. These technologies offer a technique to automatically execute data persistence and retrieval and map items and their connections to relational database tables.

Complex Data Types: ORDBMSs can handle complex data types such as spatial data types, nested tables, arrays, and user-defined kinds. These kinds provide the database additional flexibility in how complicated structures are represented and handled.

Support for **inheritance and polymorphism** is one of the primary characteristics of ORDBMS. Code reuse and data modelling are made more accessible via inheritance, which enables objects to inherit properties and behaviour from their parent objects. Thanks to polymorphism, the ability to consider objects of various types as instances of a single parent type allows for flexible programming and dynamic behaviour.

Methods and Functions: ORDBMSs allow users to specify the methods and functions connected to different object types. In object-oriented programming, methods are comparable to member functions, but functions are independent processes that may manipulate object data. These functions and methods may be used on objects in the database, giving you a mechanism to integrate behavior with the actual data.

Querying Capabilities: To allow object-oriented querying, ORDBMSs often add new features to the conventional SQL language. This covers functions like object identification, type-based querying, and object relationship traversal. Additionally, some systems offer specialized query languages, such as Object-Relational SQL (O-SQL) from Oracle and Object-Relational Extensions (ORE) from IBM DB2.

Performance Considerations: ORDBMSs must balance the performance of relational databases with the adaptability of object-oriented modelling. To enable effective query execution and data access, optimizations, including indexing, caching, and query optimization methods, are used.

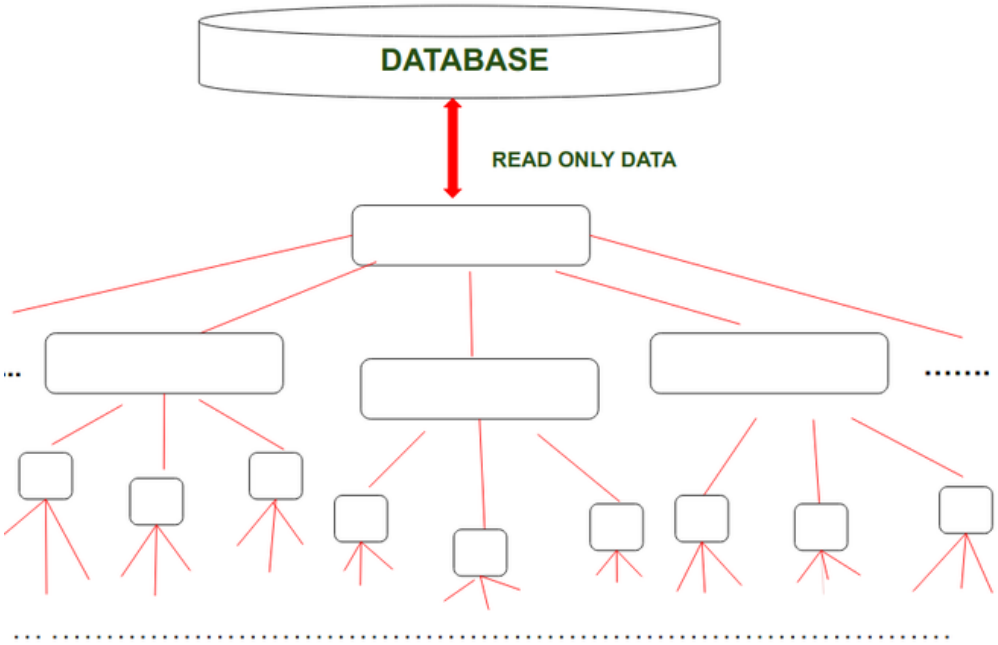
Standards and Implementations: Although there isn't a single standardized ORDBMS, several database providers have implemented capabilities that connect the relational and object-oriented

models. PostgreSQL, Oracle Database, and IBM DB2 are a few well-known database systems offering support for object-relational capabilities.

Logical Database

A Logical Database is a special type of ABAP (Advance Business Application and Programming) that is used to retrieve data from various tables and the data is interrelated to each other. Also, a logical database provides a read-only view of Data.

Structure Of Logical Database:
A Logical database uses only a hierarchical structure of tables i.e. Data is organized in a Tree-like Structure and the data is stored as records that are connected to each other through edges (Links). Logical Database contains Open SQL statements which are used to read data from the database. The logical database reads the program, stores them in the program if required, and passes them line by line to the application program.



Structure of Logical database

Features of Logical Database:
In this section, let us look at some features of a logical database:

- We can select only that type of Data that we need.
- Data Authentication is done in order to maintain security.
- Logical Database uses hierarchical Structure due to this data integrity is maintained.

Goal Of Logical Database:
The goal of Logical Database is to create well-structured tables that reflect the need of the user. The tables of the Logical database store data in a non-redundant manner and foreign keys will be used in tables so that relationships among tables and entities will be supported.

Web databases in DBMS

A web database is a system for storing and displaying information that is accessible from the Internet. It is a type of web application designed to be managed and accessed through the Internet.

Web databases are ideal for situations where the information should be shared or when it must be accessed from various locations or different devices. They are especially beneficial when the system is to be shared between locations or different devices, such as tablets, computers, and cell phones. Web databases can be used for a range of different purposes, including membership databases, client lists, inventory databases, and more.

In a web database, each field in a table has to have a defined data type, such as numbers, strings, and dates. Proper database design involves choosing the correct data type for each field to reduce memory consumption and improve performance.

The Web-based database management system is one of the essential parts of DBMS and is used to store web application data. A web-based Database management system is used to handle those databases that are having data regarding E-commerce, E-business, blogs, e-mail, and other online applications.

What are The Types of Web Databases?

Web databases can be categorised into several types based on different criteria, such as data model, location, design, and hosting. Some of the common types of web database are:

1. **Data Model Based**:

- **Hierarchical Databases**: Data is organised in a tree-like structure, with data linked based on a common point of linkage.
- **Network Databases**: Similar to hierarchical databases, child records can be associated with multiple parent records.
- **Object-Oriented Databases**: These databases are designed to work well with object-oriented programming languages such as Java, C++, and Python.
- **Relational Databases**: Data is organised into tables, rows, and columns. SQL is commonly used to query this type of database.
- **Non-Relational Databases (NoSQL)**: These databases are designed to handle unstructured data and can scale horizontally.

2. **Location-Based**:

- **Centralised Database**: The database is stored and maintained in a single location.
- **Distributed Database**: The database is spread across different physical locations for improved performance and reliability.

3. **Design Based**:

- **Operational (OLTP) Database**: These databases are designed for transaction-oriented applications.
- **Analytical (OLAP) Database**: These databases are designed for data analysis and reporting.

4. **Hosting Based**:

- **On-Premises Database**: The database is hosted on the company's own servers.
- **Cloud Database**: The database is hosted on a cloud service provider's platform.

Here are various benefits that come through the use of web-based DBMS are:

- Provides simplicity
- Web-DBMS is Platform independence
- Provides Graphical User Interface (GUI)
- Standardization
- Provides Cross-platform support
- Facilitates transparent network access
- Scalability
- Innovation

Distributed Database System

A distributed database is basically a database that is not limited to one system, it is spread over different sites, i.e, on multiple computers or over a network of computers. A distributed database system is located on various sites that don't share physical components. 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 database 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. Hence, translations are required for different sites to communicate.

Distributed Data Storage:
There are 2 ways in which data can be stored on different sites. These are:

1. Replication –
In this approach, the entire relationship is stored redundantly at 2 or more sites. If the entire database is available at all sites, it is a fully redundant database. Hence, in replication, systems maintain copies of data.

This is advantageous as it increases the availability of data at different sites. Also, now query requests can be processed in parallel.
However, it has certain disadvantages as well. Data needs to be constantly updated. Any change made at one site needs to be recorded at every site that relation is stored or else it may lead to inconsistency. This is a lot of overhead. Also, concurrency control becomes way more complex as concurrent access now needs to be checked over a number of sites.

2. Fragmentation –
In this approach, the relations are fragmented (i.e., they're divided into smaller parts) and each of the

fragments is stored in different sites where they're required. 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). Fragmentation is advantageous as it doesn't create copies of data, consistency is not a problem.

Fragmentation of relations can be done in two ways:

- **Horizontal fragmentation – Splitting by rows –**
  The relation is fragmented into groups of tuples so that each tuple is assigned to at least one fragment.
- **Vertical fragmentation – Splitting by columns –**
  The schema of the relation is divided into smaller schemas. Each fragment must contain a common candidate key so as to ensure a lossless join.

In certain cases, an approach that is hybrid of fragmentation and replication is used.

Applications of Distributed Database:

- It is used in Corporate Management Information System.
- It is used in multimedia applications.
- Used in Military's control system, Hotel chains etc.
- It is also used in manufacturing control system.

A distributed database system is a type of database management system that stores data across multiple computers or sites that are connected by a network. In a distributed database system, each site has its own database, and the databases are connected to each other to form a single, integrated system.

The main advantage of a distributed database system is that it can provide higher availability and reliability than a centralized database system. Because the data is stored across multiple sites, the system can continue to function even if one or more sites fail. In addition, a distributed database system can provide better performance by distributing the data and processing load across multiple sites.

There are several different architectures for distributed database systems, including:

Client-server architecture: In this architecture, clients connect to a central server, which manages the distributed database system. The server is responsible for coordinating transactions, managing data storage, and providing access control.

Peer-to-peer architecture: In this architecture, each site in the distributed database system is connected to all other sites. Each site is responsible for managing its own data and coordinating transactions with other sites.

Federated architecture: In this architecture, each site in the distributed database system maintains its own independent database, but the databases are integrated through a middleware layer that provides a common interface for accessing and querying the data.

Distributed database systems can be used in a variety of applications, including e-commerce, financial services, and telecommunications. However, designing and managing a distributed database system can be complex and requires careful consideration of factors such as data distribution, replication, and consistency.

Advantages of Distributed Database System:

1) There is fast data processing as several sites participate in request processing.
2) Reliability and availability of this system is high.
3) It possess reduced operating cost.
4) It is easier to expand the system by adding more sites.
5) It has improved sharing ability and local autonomy.

Disadvantages of Distributed Database System :

1) The system becomes complex to manage and control.
2) The security issues must be carefully managed.
3) The system require deadlock handling during the transaction processing otherwise the entire system may be in inconsistent state.
4) There is need of some standardization for processing of distributed database system.