1.9.1. Two-Tier Architecture

The application is partitioned into a component that resides at the client machine, which invokes database system functionality at the server machine through query language statements. Application program interface standards like ODBC (Open Database Connectivity) and JDBC (Java Database Connectivity) are used for interaction between the client and the server.

(UPTU, 2006)

1.9.2. Three-Tier Architecture

The client machine acts as merely a front end and does not contain any direct database calls. Instead, the client end communicates with an application server, usually through a forms interface. The application server in turn communicates with a database system to access data. Three-tier applications are more appropriate for large applications, and for applications that run on the world wide web.

1.10. DATABASE MODELS

Data Modelling is a way of organising a collection of information pertaining to a system under investigation. A database model is a collection of conceptual tools for describing data, data relationships, data semantics and consistency constraints. It consists of two parts :

- A mathematical notation for describing the data and relationships.
- A set of operations used to manipulate that data.

Every database and database management system is based on a particular database model. A database model consists of rules and standards that define how data is organised in a database. It provides a strong theoretical foundation of the database structure. This theoretical base enhances the power of analysis, the ability to extract inferences and to create deductions that emerge from the new data. Different models provide deficient conceptualizations of the database and they have different outlooks and different perspective. There are four basic types of database models : Hierarchical, Network, Relational and object-oriented database model.

1.10.1. Hierarchical Database Model

The hierarchical database is the oldest form of database. It was developed by the IBM for its IMS (Information Management System) database. This data model organises the data in a tree structure, that is, each child node can have only one parent node and at the top of the structure, there is a single parentheses node. In this model, a database record is a tree that consists of one or more groupings of fields called segments, which makeup the individual nodes of the tree. All the instances of a specific record are grouped together as a record, the hierarchical model uses parent-child, that is, one-to-many relationship.

The main advantage of hierarchical database is that data access is quite predictable in structure, and therefore, both retrieval and updates can be highly optimised by a DBMS. However, as the parent-child relationship is one-to-may, it restricts a child segment to having only one parent-segment. In the hierarchical database model, the links are 'hard coded' into the data structure, that is, the link is permanently established and cannot be modified. The hard coding makes the hierarchial model rigid.

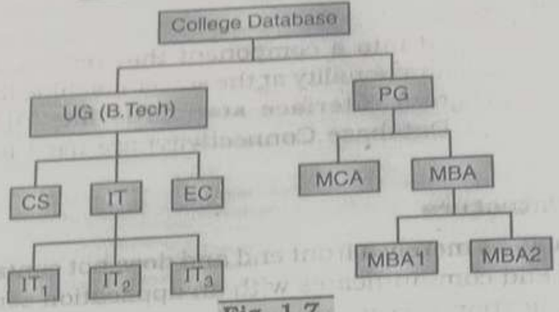


Fig. 1.7.

1.10.2. Network Database Model

The network database model was developed as an alternative to the hierarchial database. This model was formalised in 1971 by the Database Task Group (DBTG) of the conference on Data Systems Languages. The network database model expands on the hierarchial model by providing multiple paths among segments, that is, more than one parent-child relationship. Hence, this model allows having 1 : 1 (one-to-one), 1 : M (one-to-Many) and M : M (Many-to-Many) relationships.

The basic data-modelling construct in the network model is the set construct. A set consists of an owner record type, a set name and a member record type. A member record type can have that role in more than one set and as a result, multi-parent concept is supported. A networked database stores, information in data sets, which are similar to files and tables.

Although supporting multiple paths in the data structure eliminates some of the drawbacks of the hierarchical model, the network model is not very practical. The primary drawback of networked databases is that it can be quite complicating to maintain all the links.

In networks database model, all the relationships are hardwired (Pre-computed) and build into the structure of the database itself. Therefore, they are very efficient in space utilization and query execution times. However, the price for such performance is inflexibility and great difficulty of use.

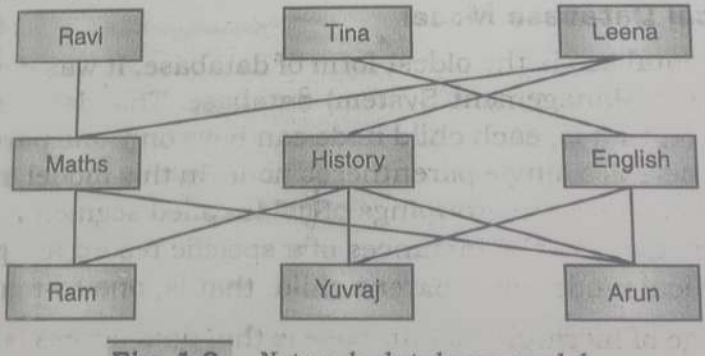


Fig. 1.8. *Network database model.*

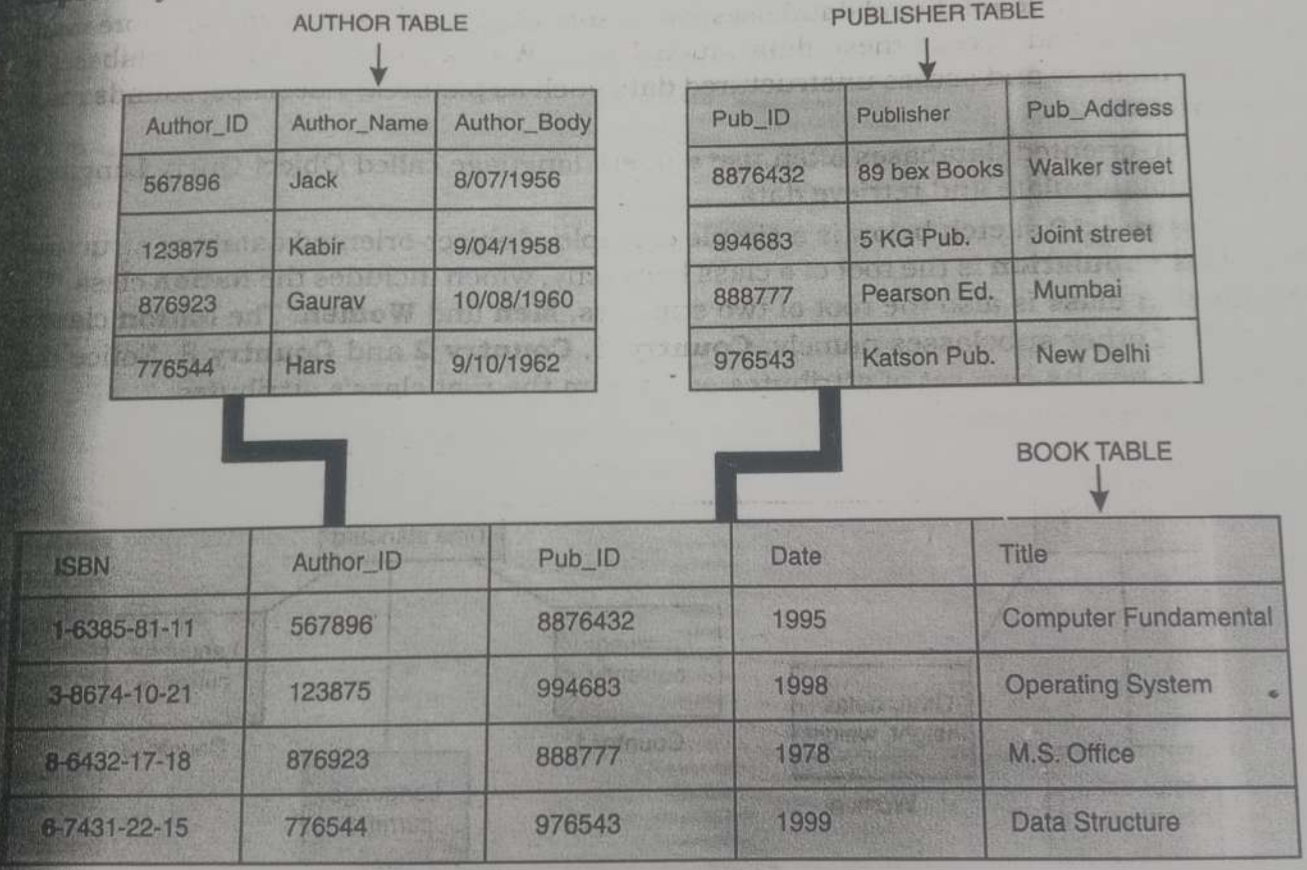
1.10.3. Relational Database Model

The hierarchical and network database model, although more flexible than tradition file systems, were still not flexible enough. The limitations of these database systems led to the development of relational database management system (RDBMS). The key difference between previous database models and relational database model is in terms of flexibility.

relatively easy and quick to create a new database structure and amend existing structures in relational systems as compared to reconstructing a network database which is complex and time consuming task.

Relational database management system is based on the relational model developed by E.F. Codd. A relational database represents all datas in the database as simple two dimensional tables called relations that are the logical equivalent of files.

Each row (record) of a relational table, called tuple, represents a data Entity with columns of the table representing attributes (fields). The allowable values for these attributes are called the domain, indicating that values can be placed in each of the columns of the relational table. Each row in a relational table must have a unique primary keys and typically, it has some secondary keys, these secondary keys correspond with primary keys in other tables, *i.e.*, relate all the tables. For example, in Fig. 1.9 mentioned below, along with ISBN as the primary key, the Book table has secondary keys AUTHOR_ID and PUB_ID. In turn, these keys serve as primary keys for the AUTHOR and PUBLISHER tables, respectively.



Author_ID	Author_Name	Author_Body
567896	Jack	8/07/1956
123875	Kabir	9/04/1958
876923	Gaurav	10/08/1960
776544	Hars	9/10/1962

Pub_ID	Publisher	Pub_Address
8876432	89 bex Books	Walker street
994683	5 KG Pub.	Joint street
888777	Pearson Ed.	Mumbai
976543	Katson Pub.	New Delhi

ISBN	Author_ID	Pub_ID	Date	Title
1-6385-81-11	567896	8876432	1995	Computer Fundamental
3-8674-10-21	123875	994683	1998	Operating System
8-6432-17-18	876923	888777	1978	M.S. Office
6-7431-22-15	776544	976543	1999	Data Structure

Fig. 1.9. *Relational Database Model.*

In a relational database, three basic operations are used to develop useful sets of data : selection, Projection and join. The selection operation retrieves certain records from a table based on the user specified criteria. The projection operation extracts fields from a table. Permitting the user to create new tables that contain only the required information. The join operation combines the data from the two tables based on a common column, providing the user with more information than is available in individual tables. Together these three operations are called relational algebra. The relational databases provides flexibility that allows charges to the database structure to be easily accommodated. The relational database system is truly a mathematically complete data model. Relational model facilities multiple views, that is, it is easy to present different user with different views of the same database. It also **in corporates concurrency** control, which guarantees the correct execution of concurrent queries.

1.10.4. Object-Oriented Database Model

The relational database model has been successfully in a wide variety of application areas. However it does not easily support the distribution of one database across a number of servers, which is the natural node for the internet. Due to these reasons, object-oriented database management systems (ODBMS) was developed. In these databases, the users can define own data access methods, the representation of data and the method of manipulating it. The objected-oriented database model is a relatively new database model and provides an outlook for the future database models. An object-oriented databases stores and maintains objects. An object is an item that can contain both data and the procedure that manipulation the data.

As compared to relational databases, object-oriented database system can store more types of data and access these data much faster. With an object-oriented database, a system can store and access unstructured data such as pictures, videoclips, sounds more efficiently.

Object-oriented databases often use a query language called Object Query Language (OQL) to manipulate and retrieve data.

In figure 1.10 sketch below is a simple example of object-oriented database structure. The class **population** is the root of a class hierarchy, which includes the **Nation** class. The **Population class** is also the root of two subclass, **Men** and **Women**. The **Nation** class is the root of other subclasses namely, **Country 1**, **Country 2** and **Country 3**. Notice that each class has its own set of attributes apart from the root class's attributes.

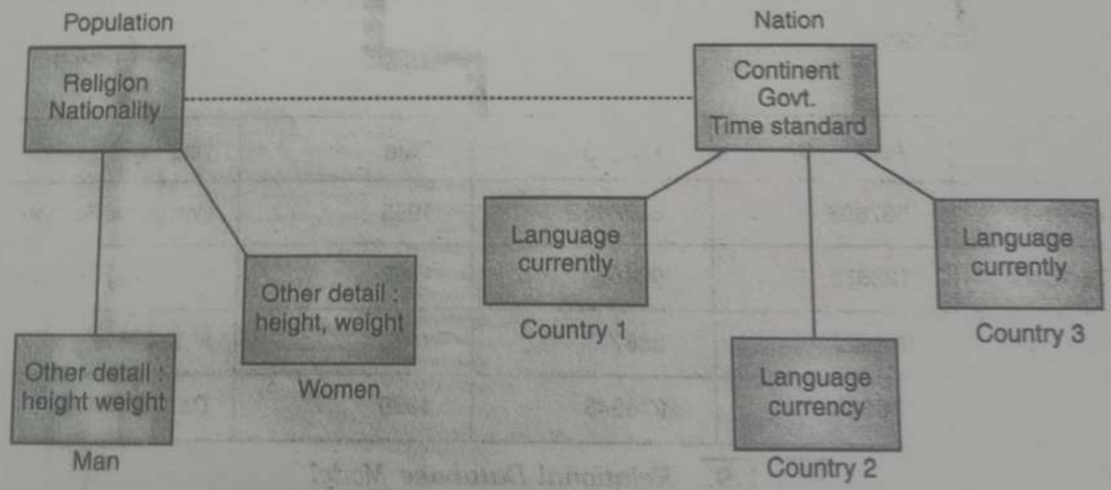


Fig. 1.10. *Object-Oriented database model.*

1.11. SCHEMA AND INSTANCES

(UPTU, 2005-06

"The collection of information stored in the database at a particular moments is called an *instance of the database*."

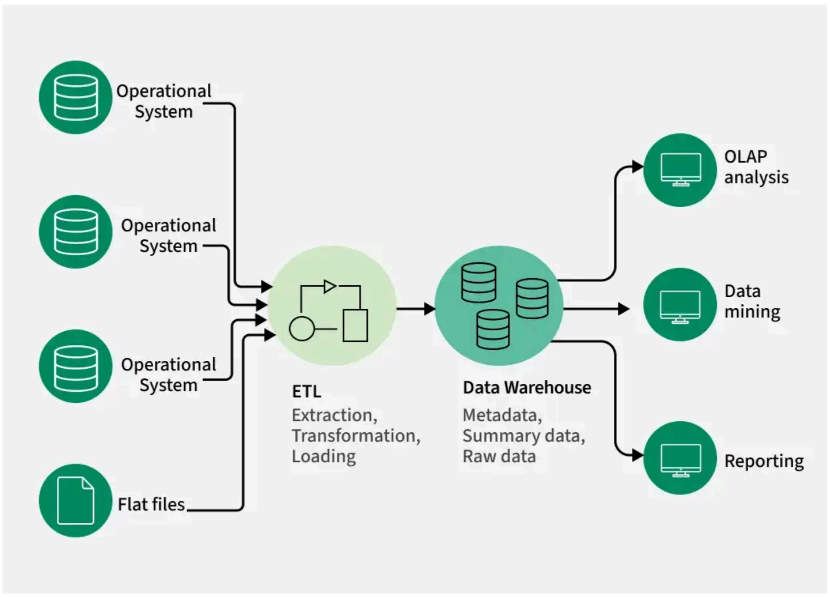
"The overall design of the database is called the *database schema*."

Data Warehousing

Last Updated : 27 Jan, 2025

A data warehouse is a centralized system used for storing and managing large volumes of data from various sources. It is designed to help businesses analyze historical data and make informed decisions. Data from different operational systems is collected, cleaned, and stored in a structured way, enabling efficient querying and reporting.

- Goal is to produce statistical results that may help in decision-making.
- Ensures fast data retrieval even with the vast datasets.



Data Warehouse Architecture

Need for Data Warehousing

1. Handling Large Volumes of Data: Traditional databases can only store a limited amount of data (MBs to GBs), whereas a data warehouse is designed to handle much larger datasets (TBs), allowing businesses to store and manage massive amounts of historical data.

2. Enhanced Analytics: Transactional databases are not optimized for analytical purposes. A data warehouse is built specifically for data analysis, enabling businesses to perform complex queries and gain insights from historical data.

3. Centralized Data Storage: A data warehouse acts as a central repository for all organizational data, helping businesses to integrate data from multiple sources and have a unified view of their operations for better decision-making.

4. Trend Analysis: By storing historical data, a data warehouse allows businesses to analyze trends over time, enabling them to make strategic decisions based on past performance and predict future outcomes.

5. Support for Business Intelligence: Data warehouses support business intelligence tools and reporting systems, providing decision-makers with easy access to critical information, which enhances operational efficiency and supports data-driven strategies.

Components of Data Warehouse

The main components of a data warehouse include:

- **Data Sources**: These are the various operational systems, databases, and external data feeds that provide raw data to be stored in the warehouse.

- **ETL (Extract, Transform, Load) Process**: The ETL process is responsible for extracting data from different sources, transforming it into a suitable format, and loading it into the data warehouse.

- **Data Warehouse Database**: This is the central repository where cleaned and transformed data is stored. It is typically organized in a multidimensional format for efficient querying and reporting.

- **Metadata**: Metadata describes the structure, source, and usage of data within the warehouse, making it easier for users and systems to understand and work with the data.

- **Data Marts**: These are smaller, more focused data repositories derived from the data warehouse, designed to meet the needs of

Open In App

specific business departments or functions.

- **OLAP (Online Analytical Processing) Tools**: OLAP tools allow users to analyze data in multiple dimensions, providing deeper insights and supporting complex analytical queries.

- **End-User Access Tools**: These are reporting and analysis tools, such as dashboards or Business Intelligence (BI) tools, that enable business users to query the data warehouse and generate reports.

Characteristics of Data Warehousing

Data warehousing is essential for modern data management, providing a strong foundation for organizations to consolidate and analyze data strategically. Its distinguishing features empower businesses with the tools to make informed decisions and extract valuable insights from their data.

- **Centralized Data Repository:** Data warehousing provides a centralized repository for all enterprise data from various sources, such as transactional databases, operational systems, and external sources. This enables organizations to have a comprehensive view of their data, which can help in making informed business decisions.

- **Data Integration:** Data warehousing integrates data from different sources into a single, unified view, which can help in eliminating data silos and reducing data inconsistencies.

- **Historical Data Storage:** Data warehousing stores historical data, which enables organizations to analyze data trends over time. This can help in identifying patterns and anomalies in the data, which can be used to improve business performance.

- **Query and Analysis:** Data warehousing provides powerful query and analysis capabilities that enable users to explore and analyze data in different ways. This can help in identifying patterns and trends, and can also help in making informed business decisions.

- **Data Transformation:** Data warehousing includes a process of data transformation, which involves cleaning, filtering, and formatting data from various sources to make it consistent and usable. This can help in improving data quality and reducing data inconsistencies.
- **Data Mining:** Data warehousing provides data mining capabilities, which enable organizations to discover hidden patterns and relationships in their data. This can help in identifying new opportunities, predicting future trends, and mitigating risks.
- **Data Security:** Data warehousing provides robust data security features, such as access controls, data encryption, and data backups, which ensure that the data is secure and protected from unauthorized access.

Types of Data Warehouses

The different types of Data Warehouses are:

1. **Enterprise Data Warehouse (EDW)**: A centralized warehouse that stores data from across the organization for analysis and reporting.
2. **Operational Data Store (ODS)**: Stores real-time operational data used for day-to-day operations, not for deep analytics.
3. **Data Mart**: A subset of a data warehouse, focusing on a specific business area or department.
4. **Cloud Data Warehouse**: A data warehouse hosted in the cloud, offering scalability and flexibility.
5. **Big Data Warehouse**: Designed to store vast amounts of unstructured and structured data for big data analysis.
6. **Virtual Data Warehouse**: Provides access to data from multiple sources without physically storing it.
7. **Hybrid Data Warehouse**: Combines on-premises and cloud-based storage to offer flexibility.
8. **Real-time Data Warehouse**: Designed to handle real-time data streaming and analysis for immediate insights.

Data Warehouse vs DBMS

Database	**Data Warehouse**
A common Database is based on operational or transactional processing. Each operation is an indivisible transaction.	A data Warehouse is based on analytical processing.
Generally, a Database stores current and up-to-date data which is used for daily operations.	A Data Warehouse maintains historical data over time. Historical data is the data kept over years and can used for trend analysis, make future predictions and decision support.
A database is generally application specific. **Example** – A database stores related data, such as the student details in a school.	A Data Warehouse is integrated generally at the organization level, by combining data from different databases. **Example** – A data warehouse integrates the data from one or more databases , so that analysis can be done to get results , such as the best performing school in a city.
Constructing a Database is not so expensive.	Constructing a Data Warehouse can be expensive.

Issues Occur while Building the Warehouse

When and how to gather data?

In a source-driven architecture for gathering data, the data sources transmit new information, eit

lly (as transaction processing

takes place), or periodically (nightly, for example). In a destination-driven architecture, the data warehouse periodically sends requests for new data to the sources. Unless updates at the sources are replicated at the warehouse via two phase commit, the warehouse will never be quite up to-date with the sources. Two-phase commit is usually far too expensive to be an option, so data warehouses typically have slightly out-of-date data. That, however, is usually not a problem for decision-support systems.

What schema to use?

Data sources that have been constructed independently are likely to have different schemas. In fact, they may even use different data models. Part of the task of a warehouse is to perform schema integration, and to convert data to the integrated schema before they are stored. As a result, the data stored in the warehouse are not just a copy of the data at the sources. Instead, they can be thought of as a materialized view of the data at the sources.

Data transformation and cleansing?

The task of correcting and preprocessing data is called data cleansing. Data sources often deliver data with numerous minor inconsistencies, which can be corrected. For example, names are often misspelled, and addresses may have street, area, or city names misspelled, or postal codes entered incorrectly. These can be corrected to a reasonable extent by consulting a database of street names and postal codes in each city. The approximate matching of data required for this task is referred to as fuzzy lookup.

How to propagate update?

Updates on relations at the data sources must be propagated to the data warehouse. If the relations at the data warehouse are exactly the same as those at the data source, the propagation is straightforward. If they are not, the problem of propagating updates is basically the view-maintenance problem.

What data to summarize?

The raw data generated by a transaction-processing system may be too large to store online. However, we can answer many queries by maintaining just summary data obtained by aggregation on a relation, rather than maintaining the entire relation. For example, instead of storing data about every sale of clothing, we can store total sales of clothing by item name and category.

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.

Advantages of Data Warehousing

- **Intelligent Decision-Making:** With centralized data in warehouses, decisions may be made more quickly and intelligently.
- **Business Intelligence:** Provides strong operational insights through business intelligence.
- **Data Quality:** Guarantees data quality and consistency for trustworthy reporting.
- **Scalability:** Capable of managing massive data volumes and expanding to meet changing requirements.
- **Effective Queries:** Fast and effective data retrieval is made possible by an optimized structure.
- **Cost reductions:** Data warehousing can result in cost savings over time by reducing data mana[illegible]ocedures and increasing

overall efficiency, even when there are setup costs initially.

- **Data security:** Data warehouses employ security protocols to safeguard confidential information, guaranteeing that only authorized personnel are granted access to certain data.
- **Faster Queries:** The data warehouse is designed to handle large queries that's why it runs queries faster than the database..
- **Historical Insight:** The warehouse stores all your historical data which contains details about the business so that one can analyze it at any time and extract insights from it.

Disadvantages of Data Warehousing

- **Cost:** Building a data warehouse can be expensive, requiring significant investments in hardware, software, and personnel.
- **Complexity:** Data warehousing can be complex, and businesses may need to hire specialized personnel to manage the system.
- **Time-consuming:** Building a data warehouse can take a significant amount of time, requiring businesses to be patient and committed to the process.
- **Data integration challenges:** Data from different sources can be challenging to integrate, requiring significant effort to ensure consistency and accuracy.
- **Data security:** Data warehousing can pose data security risks, and businesses must take measures to protect sensitive data from unauthorized access or breaches.

Next Article

How to store a password in database?

Definition and Overview of ODBMS

Last Updated : 11 May, 2023

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.

Example –

```
class CLERK

  { //variables
     char name;
     string address;
     int id;
     int salary;

    //Messages
     char get_name();
     string get_address();
```

```
      int annual_salary();
   };
```

In the above example, we can see, CLERK is a class that holds the object variables and messages.

An OODBMS also supports inheritance in an extensive manner as in a database there may be many classes with similar methods, variables and messages. Thus, the concept of the class hierarchy is maintained to depict the similarities among various classes.

The concept of encapsulation that is the data or information hiding is also supported by an object-oriented data model. And this data model also provides the facility of abstract data types apart from the built-in data types like char, int, float. ADT's are the user-defined data types that hold the values within them and can also have methods attached to them.

Thus, OODBMS provides numerous facilities to its users, both built-in and user-defined. It incorporates the properties of an object-oriented data model with a database management system, and supports the concept of programming paradigms like classes and objects along with the support for other concepts like encapsulation, inheritance, and the user-defined ADT's (abstract data types).

ODBMS 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.

An ODBMS stores and manages data as objects, and provides mechanisms for querying, manipulating, and retrieving the data. In an ODBMS, 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 ODBMS, 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 ODBMS also support standard SQL for querying the data.

ODBMS have several advantages over traditional relational databases. One of the main advantages is that they provide a natural way to represent complex data structures and relationships. Since the data is represented using objects, it can be easier to model real-world entities in the database. Additionally, ODBMS can provide better performance and scalability for applications that require a large number of small, complex transactions.

However, there are also some disadvantages to using an ODBMS. One of the main disadvantages is that they can be more complex and harder to use than traditional relational databases. Additionally, ODBMS may not be as widely used and supported as traditional relational databases, which can make it harder to find expertise and support. Finally, some applications may not require the advanced features and performance provided by an ODBMS, and may be better suited for a simpler database solution

Features of ODBMS:

Object-oriented data model: ODBMS uses an object-oriented data model to store and manage data. This allows developers to work with data in a more natural way, as objects are similar to the objects in the programming language they are using.

Complex data types: ODBMS supports complex data types such as arrays, lists, sets, and graphs, allowing developers to store and manage complex data structures in the database.

Automatic schema management: ODBMS automatically manages the schema of the database, as th[illegible]s defined by the classes and objects in the application code. This eliminates the need for a separate schema definition language and simplifies the development process.

High performance: ODBMS can provide high performance, especially for applications that require complex data access patterns, as objects can be retrieved with a single query.

Data integrity: ODBMS provides strong data integrity, as the relationships between objects are maintained by the database. This ensures that data remains consistent and correct, even in complex applications.

Concurrency control: ODBMS provides concurrency control mechanisms that ensure that multiple users can access and modify the same data without conflicts.

Scalability: ODBMS can scale horizontally by adding more servers to the database cluster, allowing it to handle large volumes of data.

Support for transactions: ODBMS supports transactions, which ensure that multiple operations on the database are atomic and consistent.

Advantages:

Supports Complex Data Structures: ODBMS 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: ODBMS provides improved performance compared to traditional relational databases for complex data models. ODBMS can reduce the amount of mapping and translation required between the programming language and the database, which can improve performance.

Reduced Development Time: ODBMS 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: ODBMS 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: ODBMS can scale horizontally and vertically, which means it can handle larger volumes of data and can support more users.

Disadvantages:

Limited Adoption: ODBMS 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: ODBMS lacks standardization, which means that different vendors may implement different features and functionality.

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

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

Scalability Challenges: ODBMS 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.

Next Article

Double not (!!) operator in PHP

Similar Reads

Difference Between OODBMS and ORDBMS

Difference Between OODBMS and ORDBMS

Last Updated : 23 Sep, 2024

Many different models exist in the field of database management which describes the way how data is to be stored and placed. Some of the models that can be compared easily are Object-Oriented Database Management Systems (OODBMS) and Object-Relational Database Management Systems (ORDBMS). It is important to understand the difference between these two to know how to use them appropriately for the right data-oriented applications. OODBMS includes object-oriented concepts with the database system itself, on the other hand, ORDBMS is an enhancement of the basic relational database system which provides object-oriented characteristics as well as maintains relational characteristics.

OODBMS

The object-oriented database system is an extension of an object-oriented programming language that includes DBMS functions such as persistent objects, integrity constraints, failure recovery, transaction management, and query processing. These systems feature object description language (ODL) for database structure creation and object query language (OQL) for database querying. Some examples of OODBMS are ObjectStore, Objectivity/DB, GemStone, db4o, Giga Base, and Zope object database.

Advantages of OODBMS

- **Complex Data Representation:** This is because, through OODBMS, hard dependencies can be well modeled to capture flexibility such as inheritance, encapsulation, and polymorphism.

- **Direct Mapping:** Here, objects of OODBMS match with the objects in programming languages, and so there is less difference between them, which is known as impedance.
- **Reusability:** Lectures such as inheritance make it possible to use existing code thereby shortening development time.

Disadvantages of OODBMS

- **Complexity:** The key working principle in an OODBMS is that the OODBMS is an implementation level of OOD, therefore, is more complex than traditional relational databases.
- **Limited Query Support:** OODBMS does not support queries such as SQL meaning there is little flexibility when it comes to querying.
- **Lack of Standardization:** OODBMS do not contain such standards as is the case with relational databases hence making the interfaces across different plates forms to be inconsistent.

ORDBMS

An object-relational database system is a relational database system that has been extended to incorporate object-oriented characteristics. Database schemas and the query language natively support objects, classes, and inheritance. Furthermore, it permits data model expansion with new data types and procedures, exactly like pure relational systems. Oracle, DB2, Informix, PostgreSQL (UC Berkeley research project), etc. are some of the ORDBMSs.

Advantages of ORDBMS

- **SQL Support:** While ORDBMS supports the structures of the SQL query language, they are quite flexible when tested against query language make it easier to handle complex queries unlike OODBMS.
- **Scalability:** It is easier and can accommodate large data sets as compared to OODBMS due to the fact that it retains the structure of relational databases.

- **Interoperability:** ORDBMS appears to offer better compatibility with the conventional RDMS thus making it easy to migrate from the conventional normal database to more OOD respectively.

Disadvantages of ORDBMS

- **Complexity in Implementation:** Although many object-oriented notions are supported by ORDBMS their representation using SQL can be problematic.
- **Overhead:** The combination in a hybrid approach causes a certain overhead which does have its influence in cases where only object-oriented or tabular representations are needed.

Difference Between OODBMS and ORDBMS

OODBMS	ORDBMS
It stands for Object Oriented Database Management System.	It stands for Object Relational Database Management System.
Object-oriented databases, like Object Oriented Programming, represent data in the form of objects and classes.	An object-relational database is one that is based on both the relational and object-oriented database models.
OODBMSs support ODL/OQL.	ORDBMS adds object-oriented functionalities to SQL.
Every object-oriented system has a different set of constraints that it can accommodate.	Keys, entity integrity, and referential integrity are constraints of an object-oriented database.
The efficiency of query processing is low.	Processing of queries is quite effective.

Conclusion

OODBMS and ORDBMS both are used for different requirements and applications are well suited for them. OODBMS is used for the system which needs objects to be mapped directly and when the structure of the data is complex. It is best for projects that require ORDBMS and benefits of relational databases but with little of OOP. Therefore, the choice between these two systems depends on certain aspects of the application, for example, need for query capabilities or object orientation.

Next Article

Advantages and Disadvantages of an ER-Model

Distributed Database System

Last Updated : 19 Sep, 2023

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.

Next Article

Functions of Distributed Database System

Similar Reads

Functions of Distributed Database System

Distributed database systems play an important role in modern data management by distributing data across multiple nodes. This article...

15+ min read

Concurrency Control in DBMS

Open In App

Introduction to Data Mining

Last Updated : 17 Apr, 2023

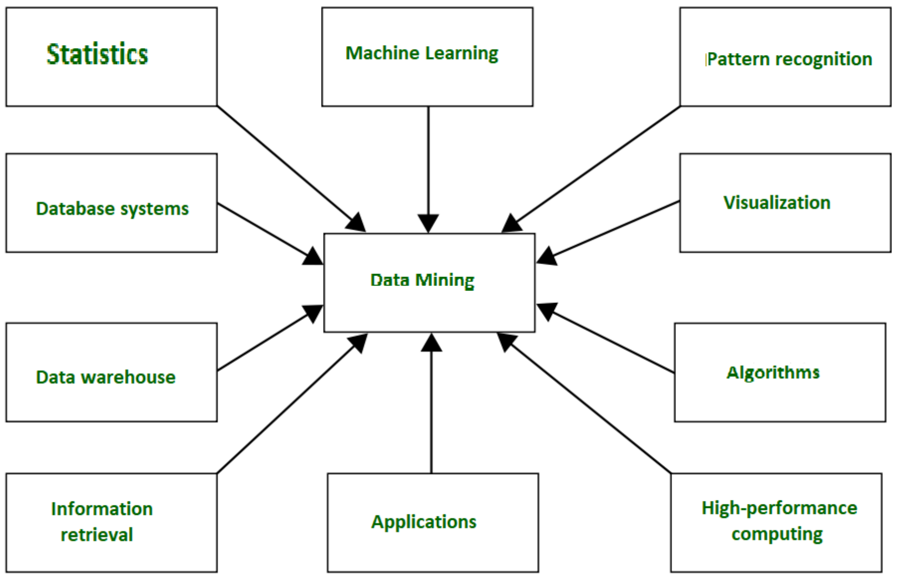
Data mining is the process of extracting useful information from large sets of data. It involves using various techniques from statistics, machine learning, and database systems to identify patterns, relationships, and trends in the data. This information can then be used to make data-driven decisions, solve business problems, and uncover hidden insights. Applications of data mining include customer profiling and segmentation, market basket analysis, anomaly detection, and predictive modeling. Data mining tools and technologies are widely used in various industries, including finance, healthcare, retail, and telecommunications.

In general terms, "**Mining**" is the process of extraction of some valuable material from the earth e.g. coal mining, diamond mining, etc. In the context of computer science, "**Data Mining**" can be referred to as **knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging**. It is basically the process carried out for the extraction of useful information from a bulk of data or data warehouses. One can see that the term itself is a little confusing. In the case of coal or diamond mining, the result of the extraction process is coal or diamond. But in the case of Data Mining, the result of the extraction process is not data!! Instead, data mining results are the patterns and knowledge that we gain at the end of the extraction process. In that sense, we can think of Data Mining as a step in the process of Knowledge Discovery or Knowledge Extraction.

Gregory Piatetsky-Shapiro coined the term **"Knowledge Discovery in Databases"** in 1989. However, the term **'data mining'** became more popular in the business and press communities. Currently, Data Mining and Knowledge Discovery are used interchangeably.

Nowadays, data mining is used in almost all places where a large amount of data is stored and processed. For example, banks typically use 'data mining' to find out their prospective customers who could be interested in credit cards, personal loans, or insurance as well. Since banks have the transaction details and detailed profiles of their customers, they analyze all this data and try to find out patterns that help them predict that certain customers could be interested in personal loans, etc.

Main Purpose of Data Mining



Data Mining

Basically, Data mining has been integrated with many other techniques from other domains such as **statistics, machine learning, pattern recognition, database and data warehouse systems, information retrieval, visualization,** etc. to gather more information about the data and to helps predict hidden patterns, future trends, and behaviors and allows businesses to make decisions.

Technically, data mining is the computational process of analyzing data from different perspectives, dimensions, angles and categorizing/summarizing it into meaningful information.

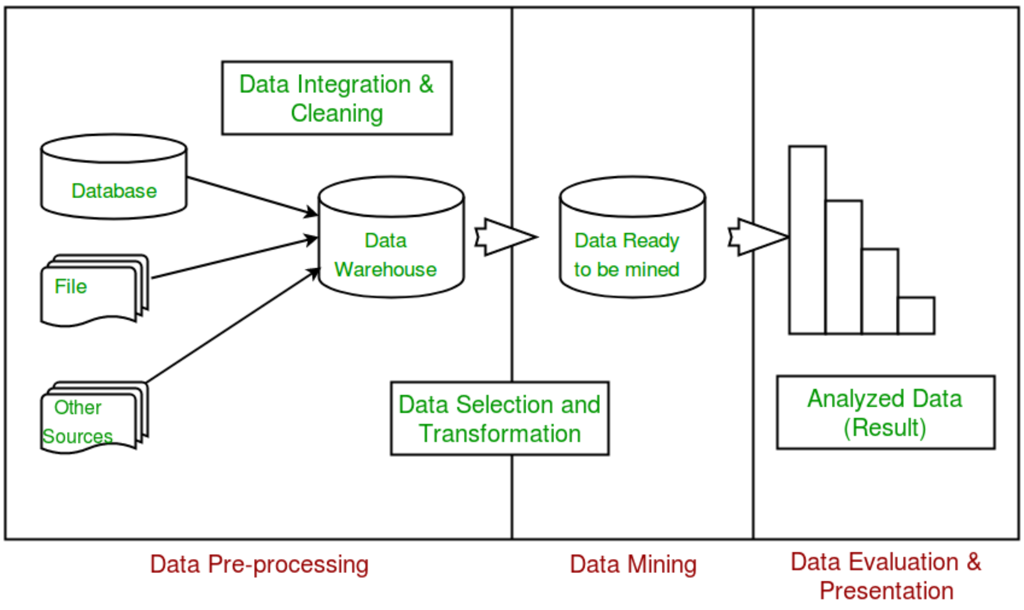
Data Mining can be applied to any type of data e.g. **Data Warehouses, Transactional Databases, Relational Databases, Multimedia Databases, Spatial Databases, Time-series Databases, World Wide Web.**

Open In App

Data Mining as a Whole Process

The whole process of Data Mining consists of three main phases:

1. Data Pre-processing – Data cleaning, integration, selection, and transformation takes place
2. Data Extraction – Occurrence of exact data mining
3. Data Evaluation and Presentation – Analyzing and presenting results



In future articles, we will cover the details of each of these phases.

Applications of Data Mining

1. Financial Analysis
2. Biological Analysis
3. Scientific Analysis
4. Intrusion Detection
5. Fraud Detection
6. Research Analysis

Benefits of Data Mining

1. **Improved decision-making**: Data mining can provide valuable insights that can help organizations make better decisions by identifying patterns and trends in large data sets.
2. **Increased efficiency:** Data mining can automate repetitive and time-consuming tasks, such as data cleaning and preparation, which can

Open In App

help organizations save time and resources.

3. **Enhanced competitiveness:** Data mining can help organizations gain a competitive edge by uncovering new business opportunities and identifying areas for improvement.
4. **Improved customer service:** Data mining can help organizations better understand their customers and tailor their products and services to meet their needs.
5. **Fraud detection:** Data mining can be used to identify fraudulent activities by detecting unusual patterns and anomalies in data.
6. **Predictive modeling:** Data mining can be used to build models that can predict future events and trends, which can be used to make proactive decisions.
7. **New product development:** Data mining can be used to identify new product opportunities by analyzing customer purchase patterns and preferences.
8. **Risk management:** Data mining can be used to identify potential risks by analyzing data on customer behavior, market conditions, and other factors.

Real-Life Examples of Data Mining

Market Basket Analysis: It is a technique that gives the careful study of purchases done by a customer in a supermarket. The concept is basically applied to identify the items that are bought together by a customer. Say, if a person buys bread, what are the chances that he/she will also purchase butter? This analysis helps in promoting offers and deals by the companies. The same is done with the help of data mining.

Protein Folding: It is a technique that carefully studies biological cells and predicts the protein interactions and functionality within biological cells. Applications of this research include determining **causes and possible cures for Alzheimer's, Parkinson's,** and cancer caused by Protein misfolding.

Fraud Detection: Nowadays, in this land of cell phones, we can use data mining to analyze cell phone activities for comparing suspicious phone activity. This can help us to detect calls made on cloned phones. Similarly, with credit cards, comparing purchases with historical purchases can detect activity with stolen cards.

Data mining also has many successful applications, such as business intelligence, Web search, bioinformatics, health informatics, finance, digital libraries, and digital governments.

Next Article
Data Mining Tutorial

Similar Reads

Data Mining Techniques
Data mining refers to extracting or mining knowledge from large amounts of data. In other words, Data mining is the science, art, and technology o…
15+ min read

Data Mining Tutorial
Data Mining Tutorial covers basic and advanced topics, this is designed for beginner and experienced working professionals too. This Data…
15+ min read

Different Types of Data in Data Mining
Introduction : In general terms, "Mining" is the process of extraction. In the context of computer science, Data Mining can be referred to as…
15+ min read

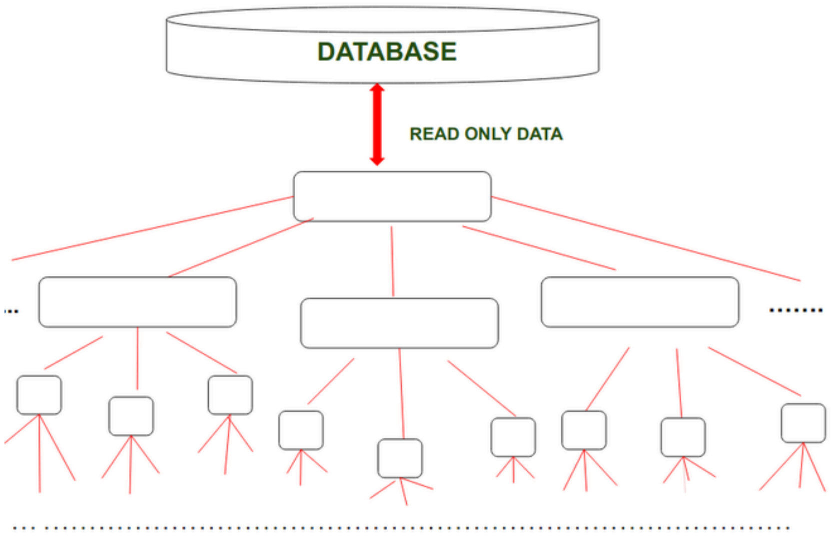
Privacy, security and social impacts of Data Mining
Data Mining is to intelligently discover useful information from large amounts of data to solve real-life problems. It is a combination of two…

Logical Database

Last Updated : 15 Mar, 2021

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.

Tasks Of Logical Database:
Below is some important task of Logical Database:

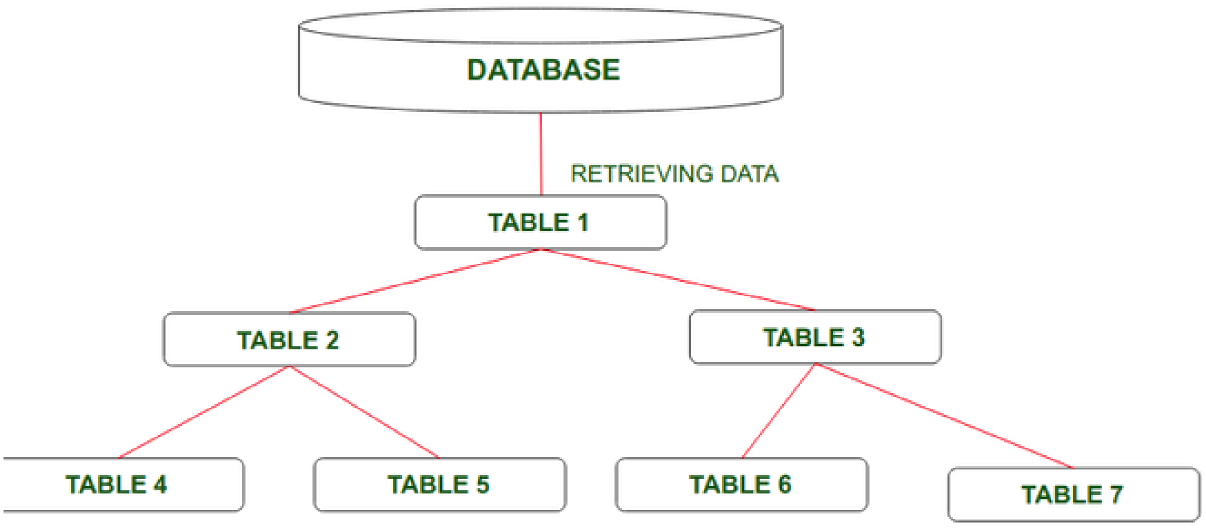
- With the help of the Logical database, we will read the same data from multiple programs.
- A logical database defines the same user interface for multiple programs.
- Logical Database ensures the Authorization checks for the centralized sensitive database.
- With the help of a Logical Database, Performance is improved. Like in Logical Database we will use joins instead of multiple SELECT statements, which will improve response time and this will increase the Performance of Logical Database.

Data View Of Logical Database:
Logical Database provides a particular view of Logical Database tables. A logical database is appropriately used when the structure of the Database is Large. It is convenient to use flow i.e

- SELECT
- READ
- PROCESS
- DISPLAY

In order to work with databases efficiently. The data of the Logical Database is hierarchical in nature. The tables are linked to each other in a Foreign Key relationship.

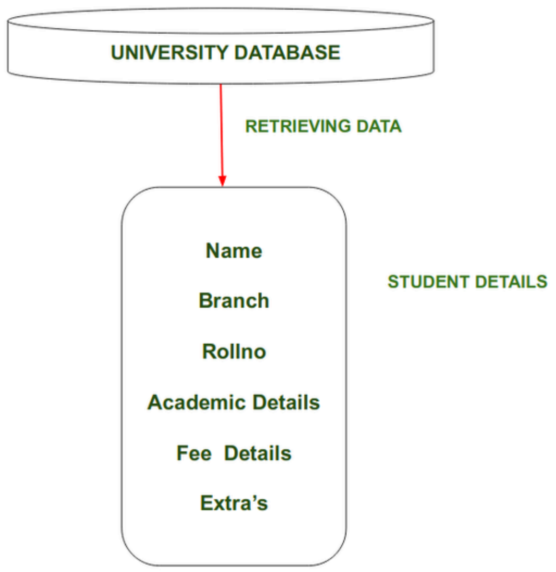
Diagrammatically, the Data View of Logical Database is shown as:



Points To Remember:

- Tables must have Foreign Key Relationship.
- A logical Database consists of logically related tables that are arranged in a hierarchical manner used for reading or retrieving Data.
- Logical Database consist of three main elements:
    - Structure of Database
    - Selections of Data from Database
    - Database Program
- If we want to improve the access time on data, then we use VIEWS in Logical Database.

Example:
Suppose in a University or College, a HOD wants to get information about a specific student. So for that, he firstly retrieves the data about its batch and Branch from a large amount of Data, and he will easily get information about the required Student but didn't alter the information about it.



Advantages Of Logical Database:

Let us look at some advantages of the logical database:

- In a Logical database, we can select meaningful data from a large amount of data.
- Logical Database consists of Central Authorization which checks for Database Accesses is Authenticated or not.
- In this Coding, the part is less required to retrieve data from the database as compared to Other Databases.
- Access performance of reading data from the hierarchical structure of the Database is good.
- Easy to understand user interfaces.
- Logical Database firstly check functions which further check that user input is complete, correct, and plausible.

Disadvantages Of Logical Database:

This section shows the disadvantages of the logical database:

- Logical Database takes more time when the required data is at the last because if that table which is required at the lowest level then firstly all upper-level tables should be read which takes more time and this slows down the performance.
- In Logical Database ENDGET command doesn't exist due to this the code block associated with an event ends with the next event statement.