Encapsulation creates ADT

Friday, July 7, 2017 6:54 AM

2073 Q3 - https://www.facebook.com/groups/csitbatch2070/permalink/1502092166493068/
3. Define encapsulation? How is it used to create abstract data types?

Encapsulation means that an object contains both the data structure and the set of operations that can be used to manipulate it. Often cases, adopting encapsulation hides the implementation from the users do not necessarily have to know the detail of it.

Encapsulation is related to the concepts of abstract data types and information hiding in programming languages. Here the main idea is to define the behavior of a type of object based on the operations that can be externally applied to objects of that type. The internal structure of the object is hidden, and the object is only accessible through a number of predefined operations. Some operations may be used to create or destroy objects; other operations may update the object value and other may be used to retrieve parts of the object value or to apply some calculations to the object value.

The external users of the object are only made aware of the interface of the object, which defines the names and arguments of each operation. The implementation of the object is hidden from the external users; it includes the definition of the internal data structure of the object and the implementation of the operations that access these structures.

In object oriented - OO terminology, the interface part of each operation is called the signature, and the operation implementation is called a method. A method is invoked by sending a message to the object to execute the corresponding method.

Not all objects are meant to be stored permanently in the database. Transient objects exist in the executing program and disappear once the program terminates.

Persistent objects are stored in the database and persist after program terminates. The typical mechanism for persistence involves giving an object a unique persistent name through which it can be retrieved.

In traditional database models and systems, this concept was not applied, since it is usual to make the structure of database objects visible to users and external programs.

Relational features of SQL

Friday, July 7, 2017 6:57 AM

2073 Q5 - https://www.facebook.com/groups/csitbatch2070/permalink/1502043426497942/

5. What is object relational database? Discuss object relational features of SQL

The following are some of the object database features that have been included in SQL:

- Some type constructors have been added to specify complex objects. These include the row type, which corresponds to the tuple (or struct) constructor. An array type for specifying collections is also provided. Other collection type constructors, such as set, list, and bag constructors, were not part of the original SQL/Object specifications but were later included in the standard.
- A mechanism for specifying object identity through the use of reference type is included.
- Encapsulation of operations is provided through the mechanism of user defined types (UDTs) that may include operations as part of their declaration. These are somewhat similar to the concept of abstract data types that were developed in programming languages. In addition, the concept of user defined routines (UDRs) allows the definition of general methods (operations).
- Inheritance mechanisms are provided using the keyword UNDER.

Dis/Advantage of OODBMS

From Bhupi Page4 Unit3

Thursday, July 6, 2017 10:26 PM

Advantages of OODBMS

- Easier Design-Reflect Applications
- Modularity and Reusability
- Incremental refinement and abstraction
- Multiple inheritance
- Support for multiple version and Alternatives
- Designer can specify the structure of objects and their behavior (methods).
- Better interaction with object-oriented languages such as Java and C++
- Definition of complex and user-defined types.
- Encapsulation of operations and user-defined methods

Disadvantages of OODBMSs

There are following disadvantages of OODBMSs:

- **Lack of universal data model:** There is no universally agreed data model for an OODBMS, and most models lack a theoretical foundation. This .disadvantage is seen as a significant drawback, and is comparable to pre-relational systems.
- **Lack of experience:** In comparison to RDBMSs the use of OODBMS is still relatively limited. This means that we do not yet have the level of experience that we have with traditional systems. OODBMSs are still very much geared towards the programmer, rather than the naïve end-user. Also there is a resistance to the acceptance of the technology. While the OODBMS is limited to a small niche market, this problem will continue to exist
- **Lack of standards:** There is a general lack of standards of OODBMSs. We have already mentioned that there is not universally agreed data model. Similarly, there is no standard object-oriented query language.
- **Competition:** Perhaps one of the most significant issues that face OODBMS vendors is the competition posed by the RDBMS and the emerging ORDBMS products. These products have an established user base with significant experience available. SQL is an approved standard and the relational data model has a solid theoretical formation and relational products have many supporting tools to help .both end-users and developers.
- **Query optimization compromises encapsulations:** Query optimization requires. An understanding of the underlying implementation to access the database efficiently. However, this compromises the concept of incrassation
- **Locking at object level may impact performance** Many OODBMSs use locking as the basis for concurrency control protocol. However, if locking is applied at the object level, locking of an inheritance hierarchy may be problematic, as well as impacting performance.
- **Complexity:** The increased functionality provided by the OODBMS (such as the illusion of a single-level storage model, pointer sizzling, version management, and schema evolution--makes the system more complex than that of traditional DBMSs. In complexity leads to products that are more expensive and more difficult to use.
- **Lack of support for views:** Currently, most OODBMSs do not provide a view mechanism, which, as we have seen previously, provides many advantages such as data independence, security, reduced complexity, and customization.
- **Lack of support for security:** Currently, OODBMSs do not provide adequate security mechanisms. The user cannot grant access rights on individual objects or classes.

ADDITIONAL:

Features of object oriented (O-O) Databases

- Object oriented databases store persistent objects permanently on secondary storage, and allow the sharing of these objects among multiple programs and applications.
- Object oriented databases provide a unique system-generated object identifier for each object. Object oriented databases maintain a direct correspondence between real-world and database objects so that objects don't lose their integrity and identify and can be easily be identified and operated upon.
- In Object Oriented databases, objects can be very complex in order to contain all significant information that may be required to describe the object completely.
- Object oriented databases allow us to store both the class and state of an object between programs. They take the responsibility for maintaining the links between stored object behavior and state away from the programmer, and manage objects outside of programs with their public and private elements intact. They also simplify the whole process of rendering objects persistent by performing such tasks invisibly

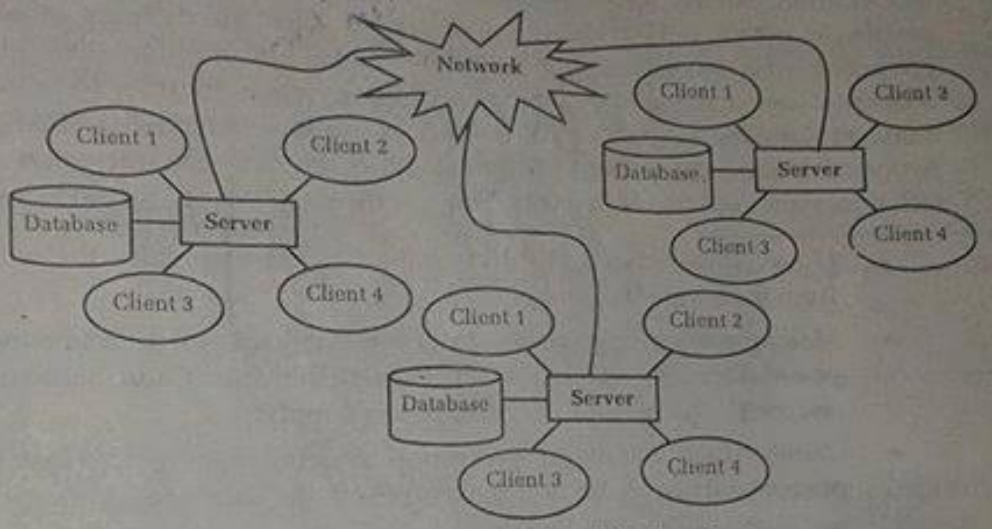
potential advantage for distributed database.

Thursday, July 6, 2017 10:36 PM

2071 Q8 - https://www.facebook.com/groups/csitbatch2070/permalink/1502048586497426/
8.) Describe the main reasons for the potential advantage for distributed database. What additional functions does it have over centralized DBMS?

Distributed Database

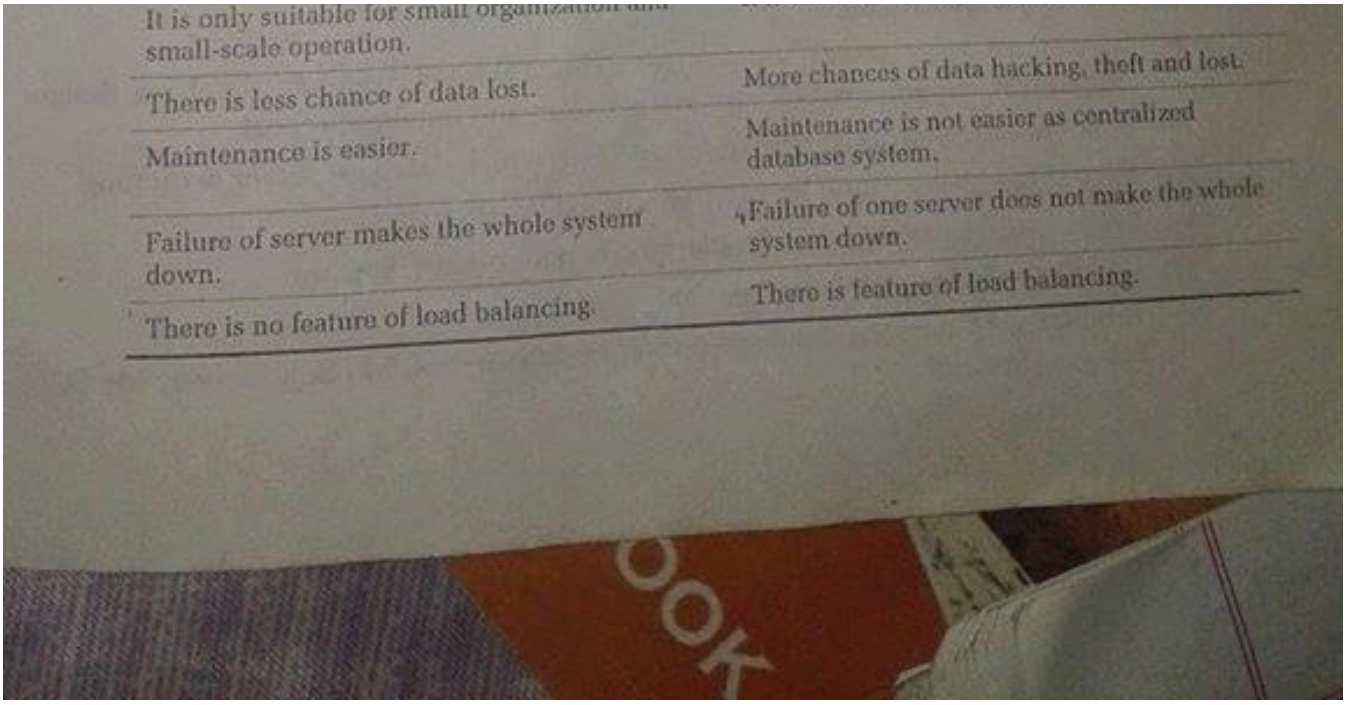
This is a complex type of database system. It is a collection of multiple logically interrelated databases which are distributed in many geographical locations. Data communication is harder because they should communicate through different system to different platforms. It has problem of control of security, increased problem of control of database and need more computer resources. It is much suitable for such types of organizations, which are spread all over with similar objectives and similar data sets. In other words, it is the collection of different centralized database systems in different locations.



Distributed Database System

Difference between Centralized and Distributed Database System

Centralized Database System	Distributed Database System
Centralized database system is simple type.	Distributed Database System is complex type.
They are located on particular location.	They are located in many geographical locations.
It does not contain servers in several locations.	It contains servers in several locations.
It is only suitable for small organization and small-scale operation.	It is suitable for large organizations.
There is less chance of data lost.	More chances of data hacking, theft and lost.



It is only suitable for small organization and small-scale operation.	
There is less chance of data lost.	More chances of data hacking, theft and lost.
Maintenance is easier.	Maintenance is not easier as centralized database system.
Failure of server makes the whole system down.	Failure of one server does not make the whole system down.
There is no feature of load balancing.	There is feature of load balancing.