REPLICAREA DATELOR ÎN MEDIILE DISTRIBUITE

Nicoleta IACOB, *Student Doctorand,*
Universitatea din Piteşti

REZUMAT: *Sistemul distribuit se modelează cel mai bine pe structura organizaţională a companiilor care, din cauza nevoilor lor de afaceri, sunt distribuite geografic. Pentru organizaţiile care se extind la nivel global, schimbul de date între bazele de date şi aplicaţii multiple a devenit foarte important. În mediile distribuite ne confruntăm cu noi probleme care nu sunt relevante în mediile centralizate, cum ar fi fragmentarea şi replicarea datelor. Proiectarea unui sistem de baze de date distribuite implică, de asemenea, luarea unor decizii cu privire la amplasarea informaţiei pe site-uri diferite într-o reţea de calculatoare. Problema alocării fragmentelor datelor pe diferite site-uri este strâns legată de replicarea datelor din baze de date distribuite. În această lucrare ne vom concentra asupra problemei alocării datelor, care dezvăluie importanţa alocării datelor în procesul proiectării bazei de date distribuite în corelaţie cu fragmentarea datelor. De asemenea, vom analiza strategiile disponibile şi metodele de alocare a datelor.*

CUVINTE CHEIE: *replicare, strategii, metode de alocare.*

1. Introducere

O **bază de date distribuită (BDD)** este o colecţie de date partajate, intercorelate logic, distribuite geografic într-o reţea de calculatoare.

Un **sistem distribuit de gestiune a bazelor de date (SGBDD)** este un sistem de programe care permite gestionarea bazei de date distribuite şi face distribuirea transparentă pentru utilizatori. Obiectivul transparenţei este de a face ca un sistem distribuit să apară ca unul centralizat. Acesta fiind denumit uneori *principiul fundamental al SGBDD.*

DATA REPLICATION IN DISTRIBUTED ENVIRONMENTS

Nicoleta IACOB, *PhD Student,*
University of Piteşti

ABSTRACT: *Distributed system model best suits on the organizational structure of companies that, because of their business needs, are geographically distributed. For the organizations that are expanding globally, the data exchange between multiple databases and applications has become more important. In distributed environments we are facing new problems that are not relevant in centralized environments, such as data fragmentation and replication. Designing a distributed database system also involves the decisions about placing the information on different sites in a computer network. The problem of data fragments allocation on different sites is closely related to data replication in distributed databases. In this paper we will concentrate on data allocation problem, revealing the data allocation importance in the process of the distributed database design in correlation with data fragmentation. In addition, we will analyze the available strategies and methods of data allocation.*

KEY-WORDS: *replication, strategies, methods of data allocation.*

1. Introduction

A **distributed database (DDB)** is a collection of shared data, logically interrelated, which are geographically distributed in a computer network.

A **distributed database management system (DDBMS)** is a set of programs that allows the management of distributed database and makes the distribution transparent to the users. The objective of transparency is to make a distributed system appear similar to a centralized system. This is usually called the *fundamental principle of DDBMS.*

2. Proiectarea bazelor de date distribuite

Proiectarea bazelor de date distribuite reprezintă o activitate complexă, care trebuie să ia în considerare mai mulţi factori, cum ar fi: cerinţele de accesare ale aplicaţiilor, restricţiile de integritate definite la nivelul întregii baze de date, configuraţia nodurilor din sistem şi configuraţia reţelei. Natura distribuită a datelor implică în afară de activităţile care se referă la proiectarea bazelor de date centralizate, următoarele: fragmentarea datelor, partiţionarea, replicarea şi alocarea fragmentelor şi a replicilor în diferite locaţii.

Indiferent de modul de abordare, în proiectarea BDD se urmăreşte să se atingă următoarele *obiective specifice*:

- *Maximizarea prelucrării datelor la nivel local* presupune ca plasarea datelor trebuie să fie cât mai aproape de aplicaţiile care le solicită. Se remarcă faptul că într-o BDD corect proiectată aproximativ 90% din volumul de date ar trebui să fie accesate la nivel local şi doar 10% ar trebui să fie accesate de la distanţă. Avantajul execuţiei locale a cererii este dat nu numai de reducerea numărului de accesări de la distanţă, dar şi de simplificarea controlului în execuţia cererii.
- *Asigurarea unui nivel ridicat de siguranţă şi disponibilitate a datelor* se poate realiza prin replicarea datelor pe diferite site-uri. În acest fel, sistemul poate utiliza o copie alternativă atunci când cea care ar fi trebuit să fie accesată în condiţii normale nu este disponibilă.
- *Prelucrarea datelor în paralel*. Distribuirea bazelor de date oferă posibilitatea de a utiliza eficient capacitatea procesorului din fiecare staţie pentru a maximiza gradul de paralelism în execuţia aplicaţiei. Creşterea numărului de operaţiuni efectuate în paralel contravine principiului care urmăreşte maximizarea numărului de prelucrări de date efectuate local. În proiectarea BDD trebuie să se asigure un raport optim între cele două tipuri de prelucrare.

2. The design of distributed database

The design of distributed database represents a complex activity, which must take into consideration a lot of factors such as: the requests used to access the applications, the integrity restrictions defined at the database level, the configurations of the nodes and of the network, identifying the applications and their access requirement of the data. The distributed nature of the data involves apart from the activities which have to do with the design of centralized database, new further data: data fragmentation, partitioning, replication and fragments allocation as well as replicas on different sites.

Irrespective of the approach DDB design seeks to reach the following *specific objectives*:

- *Maximizing local processing of data*, which implies data placement must be as close as the applications need them. It is valued that in a properly designed DDB approximately 90% of the total data should be locally accessed and only 10% percent should be accessed from remote locations. The advantage of the fully local execution of the application is given not only by the reduction of remote accesses but also by simplification of the control in application execution.
- *Enabling a high level of data security and availability*, which can be done by data replication to numerous sites. In this way, the system can use an alternative copy when the one which had to be accessed under regular circumstances is not available.
- *Parallel data processing*. Database distribution offers the possibility to use efficiently the processor's capacity on each computer to maximize the degree of parallelism in application execution. The growth of the operations done simultaneously contradicts the principle of local operations processing. In DDB design there must be a proper relation

Etapele de proiectare a BDD sunt:

- Proiectarea schemei globale;
- Proiectarea schemei fizice;
- Proiectarea fragmentării;
- Proiectarea alocării.

Tehnica de proiectare a schemelor globală şi fizică într-o bază de date distribuită este similară cu cea folosită în cazul bazei de date centralizate. Specifice proiectării BDD sunt problemele legate de *fragmentarea* şi *alocarea datelor*. Proiectarea BDD se poate face, ca şi în cazul bazei de date centralizate, folosind abordările *top-down* şi/sau *bottom-up*.

Proiectarea top-down, prezentată schematic în figura 1, are scopul de a asigura o distribuţie optimă a datelor. Ea este utilizată în faza de proiectare a bazei de date distribuite.

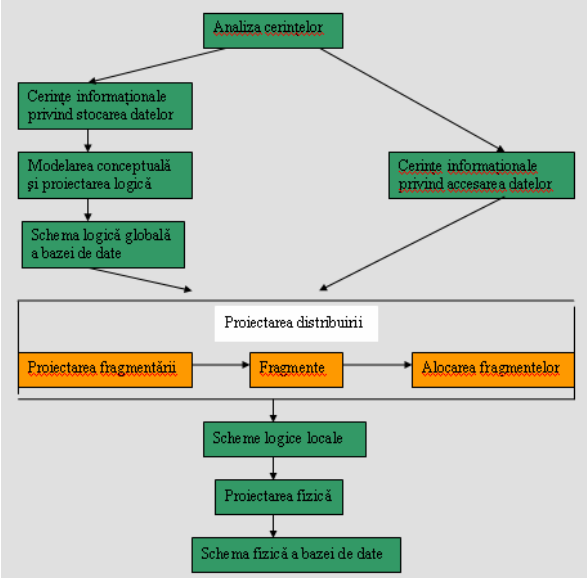

Figura 1. Proiectarea top-down a bazelor de date distribuite

Proiectarea top-down începe cu identificarea şi analiza cerinţelor informaţionale ale sistemului care va utiliza baza de date distribuită şi cu identificarea cerinţelor de stocare a datelor şi a modelului de acces la date. În primul rând, ele sunt translatate într-un set de entităţi cu atribute specifice şi relaţii între entităţi, care sunt apoi transformate într-un set de tabele cu legături între ele, care va forma schema globală a

between the two types of processing.

DDB design steps are:

- Global design scheme;
- Physical design scheme;
- Fragmentation design;
- Allocation design.

Technique design approach of the global and physical schemes in a distributed database is similar to that used for a centralized database. The issues of *fragmentation* and *data allocation* are specific to distributed databases design. The design of DDB can be done, as in the case of centralized database, using the *top-down* and/or *bottom-up* approach.

Top-down design shown schematically in Figure 1, aims to assure an optimal distribution of data and is used in the distributed database design phase.

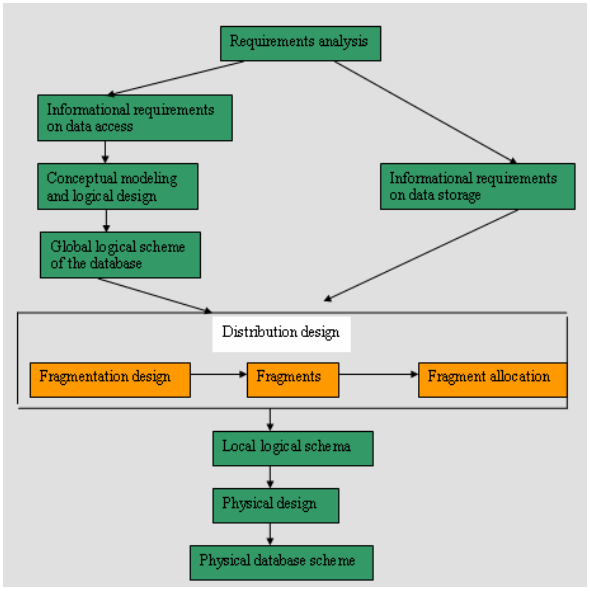

Figure 1. Top-down design of distributed databases

The top-down design begins with the identification and analysis of information requirements of the system which will use the distributed database and with the identification of data storage requirements and data access model. Firstly, they are translated in a set of entities with specific attributes and relationships between entities, which are then transformed into a set of tables with links between them that will form

datelor.

Până în acest punct, procesul de proiectare este identic cu cel pentru proiectarea bazelor de date centralizate, însă implicaţiile legate de natura distribuită a datelor nu sunt luate în considerare. Schema logică globală şi cerinţele de accesare a datelor reprezintă intrări în etapa de proiectare a distribuirii datelor. Obiectivul acestei etape este proiectarea schemelor logice locale, prin distribuirea tabelelor pe nodurile din reţea.

Cele două activităţi, fragmentarea şi alocarea datelor sunt abordate independent, rezultatele fragmentării reprezentând intrarea în faza de alocare a datelor. Ambele activităţi au aceleaşi intrări, diferenţa dintre ele constând în faptul că fragmentarea porneşte de la relaţii globale, în timp ce alocarea datelor ia în considerare şi rezultatele fragmentării. Ambele ţin seama de cerinţele de acces la date, dar fiecare dintre ele ignoră constrângerile rezultate din deciziile de proiectare. Definirea fragmentelor şi alocarea acestora pe site-uri îşi propun să realizeze, pe cât posibil, maximizarea numărului de tranzacţii în regim local.

Ultima etapă a procesului de proiectare este proiectarea fizică. Obiectivul acestei etape constă în maparea fiecărei scheme logice locale, rezultată în urma proiectării sistemului distribuit în schemele fizice de stocare, în funcţie de facilităţile oferite de software-ul SGBD-ului de pe fiecare nod şi de cerinţele de performanţă formulate. Conţinutul acestei etape nu este diferit faţă de etapele implicate în proiectarea unei baze de date centralizate.

Proiectarea fragmentării. Înainte de a distribui datele, proiectarea fragmentării datelor determină modul în care relaţile globale vor fi divizate în unităţi logice care vor fi distribuite.

Există câteva *reguli elementare* care trebuie respectate în definirea fragmentelor:

- Condiţia de *completitudine*: întreaga relaţie globală trebuie să fie împărţită în fragmente;
- Condiţia de *reconstrucţie*: în orice

the global schema of the data.

Until this point the design process is identical to the centralized database design, but the implications of the distributed nature of the data are not taken into account. Logical global scheme and data access requirements represent inputs in the design phase of data distribution. The objective of this phase is the logical design of local schemes, by distributing tables on system nodes.

The two activities, fragmentation and data allocation are addressed independently, fragmentation results representing the input in data allocation phase. Both activities have the same inputs, the difference between them being the fact that fragmentation starts from global relationships, while the data allocation takes into account fragmentation results. Both take into account the data access requirements, but each of them ignore the way in which requirements are comprised by the other design decisions. The defining of fragments and their allocation on sites aims to achieve, as much as possible, access to the local data references.

The last step of the design process is the physical design. The objective of this phase consists in mapping each local scheme, resulted from the distributed system design phase in physical storage schemes, depending on the facilities offered by the DBMS software on each node and performance requirements. The steps of this phase are no different from the steps involved in the centralized database design.

Designing fragmentation. Prior to distribute data, data fragmentation design determines how the global relationship will be divided into logical units that will be distributed.

There are some *basic rules* to be followed in defining the fragments:

- The condition of *completeness*: the entire global relationship should be partitioned into fragments;
- The condition of *reconstruction*: in any event it must be possible to

caz trebuie să fie posibil să putem reconstrui relaţia globală din fragmentele sale;

- Condiţia de *disjuncţie*: fragmentele trebuie să fie disjuncte, astfel încât replicarea datelor să poată fi controlată în mod explicit la nivelul alocării datelor.

În *partiţionarea orizontală*, un tabel relaţional poate fi împărţit astfel încât unele linii ale tabelei bazei de date sunt situate într-un site, alte linii cu înregistrări sunt situate într-un alt site, şi aşa mai departe. În *partiţionarea verticală*, coloanele unui tabel sunt împărţite între mai multe site-uri în reţea. Acestă partiţionare are sens atunci când datele din site-uri diferite sunt implicate într-o tranzacţie.

Proiectarea alocării şi replicarea datelor. Alocarea fragmentelor este strâns legată de replicarea datelor din BDD. În faza de alocare a datelor proiectantul trebuie să decidă dacă fragmentele BDD vor fi replicate şi, de asemenea, gradul lor de replicare.

Replicarea este un proces care constă în realizarea şi distribuirea de copii ale datelor şi, în plus, permite ca modificările efectuate să fie propagate în mod consistent la copiile corespunzătoare. Distribuirea acestor replici are ca scop procesarea datelor la nivel local.

Procesul de replicare sporeşte securitatea sistemului şi îmbunătăţeşte viteza operaţiunilor de procesare de date. În afară de aceasta, aplicaţia poate funcţiona chiar şi dacă un server local ar eşua, dar alte servere cu baze de date replicate rămân accesibile. În cazul replicării datelor, este mai dificil să se asigure menţinerea consistenţei acestora pentru că actualizarea unui fragment dintr-o bază de date trebuie să fie propagată la toate copiile fragmentului din bazele de date replicate. Decizia replicării unui fragment este influenţată de raportul dintre numărul de procese de scriere şi citire ale fragmentului.

În ceea ce priveşte replicarea, o bază de date poate fi:

- *nereplicată* - atunci când

reconstruct the global relationship from its fragments;

- The condition of *disjunction*: the fragments must be disjunctive, so that data replication can be controlled explicitly at data allocation level.

In *horizontal partitioning*, a relational table can be split up so that some rows of a database tables are located at one site, other rows with records are located at another site, and so on. In *vertical partitioning*, the columns of a table are divided up among several sites on the network. This arrangement can make sense when different sites are responsible for processing different functions involving an entity.

Design of data allocation and data replication. The allocation of fragments is closely related to the replication of data from DDB. In the data allocation phase the designer must decide whether DDB fragments will be replicated and also their degree of replication.

Replication is a process that consists in making and distributing copies of target data and in addition, allows changes committed in the sources to be consistently propagated to the appropriate copies. Distribution of these replicas must ensure that the data processing is done locally.

The replication process increases the security of the system and improves the speed of data operations. Apart from this, application can work even if a local server would fail, but other servers with replicated databases remain accessible. In terms of data replication, maintaining consistency of data in the database is more difficult to ensure because a fragment update must be propagated to all copies of the fragment. The decision of a fragment replication is influenced by the report between the number of database reads and database writes for a fragment.

In terms of replication, a database can be:

- *not replicated* - when the system has a single copy that resides only on a

sistemul are o singură bază, care se află doar la un nod;
- *parţial replicată* - atunci când există date care sunt replicate şi date care nu sunt replicate;
- *complet replicată* - când întreaga bază de date este pe deplin replicată pe una sau mai multe noduri ale rețelei.

Tehnicile de replicare şi fragmentare pot fi aplicate succesiv. Astfel, un fragment poate fi replicat şi unitățile replicate pot fi fragmentate.

Există două abordări pentru a controla replicarea: *controlul replicării sincrone* şi *controlul replicării asincrone.* În replicarea sincronă, replicile sunt sincronizate în orice moment. În această abordare, o tranzacție poate accesa orice copie deoarece datele accesate sunt aceleaşi în toate celelalte copii. Evident, este fizic imposibilă schimbarea valorilor din toate copiile unui element de date în acelaşi timp.

Prin urmare, pentru a păstra consistența copiilor, în timp ce o copie a unor date este în curs de modificare, algoritmul de control al replicării va ascunde valorile altor copii care nu sunt sincronizate cu aceasta (de exemplu, prin blocarea acestora). Cu alte cuvinte, în nici o tranzacție nu vom putea să vedem vreodată valori diferite pentru diferite copii ale aceloraşi date. În replicarea asincronă, spre deosebire de replicarea sincronă, replicile nu sunt ținute în sincronizare tot timpul. Două sau mai multe replici ale aceloraşi date pot avea valori diferite uneori şi într-o tranzacție se pot vedea aceste valori diferite. Aceste lucru este acceptabil pentru unele aplicații care nu necesită o updatare în timp real, cum ar fi cele pentru depozit.

Definirea problemei alocării

Problema alocării fragmentelor poate fi tratată ca o problemă de optimizare a plasării fiecărui fragment. Distribuția eficientă a fragmentelor necesită un echilibru între costurile (de stocare, prelucrare şi transmisie a datelor), performanțe (în special

node;
- *partially replicated* - when there are data that are replicated and data that are not replicated;
- *fully replicated* - when the entire database is fully replicated on one or more nodes of the network.

Replication and fragmentation techniques can be applied successively on the same relationship. Thus, a fragment can be replicated and replicated units can be fragmented.

There are two approaches to replication control: *synchronous replication control* and *asynchronous replication control*. In synchronous replication, replicas are kept in sync at all times. In this approach, a transaction can access any copy of the data item with the assurance that the data item is accessing has the same value as all its other copies. Obviously, it is physically impossible to change the values of all copies of a data item at exactly the same time.

Therefore, to provide a consistent view across all copies, while a data item copy is being changed, the replication control algorithm has to hide the values of the other copies that are out of sync with it (e.g., by locking them). In other words, no transaction will ever be able to see different values for different copies of the same data item. In asynchronous replication, as opposed to synchronous replication, the replicas are not kept in sync at all times. Two or more replicas of the same data item can have different values sometimes, and any transaction can see these different values. This happens to be acceptable in some applications, such as the warehouse and point-of-sales database copies.

Allocation problem definition

The problem of fragment allocation can be treated as an optimization problem of placement of each fragment. Efficient distribution of fragments requires a balance between costs (storage, processing and transmission of data), performance (especially response time) and data

timpul de răspuns) şi restricţii de distribuire a datelor. Să presupunem că există o mulţime de n fragmente F = {F1, F2, ... , Fn} exploatate de o mulţime de k aplicaţii A = {A1, A2, ..., Ak} pe o mulţime m de staţii S = {S1, S2, ..., Sm}. Problema alocării se reduce la găsirea distribuţiei optime a lui F pe S.

Soluţia optimă de distribuire poate fi definită prin două *obiective:*

1. *Minimizarea costurilor.* Costul total se calculează prin însumarea costurilor de comunicaţie (aferente transmisiei mesajelor şi datelor), a costurile de prelucrare a operaţiilor de integrare şi actualizare a fiecărui fragment de pe fiecare nod (aferent utilizării procesorului şi operaţiilor de intrare/ieşire) şi a costurilor cu stocarea fiecărui fragment pe fiecare nod;
2. *Maximizarea performanţelor.* Performanţele sunt măsurate, în special, de timpii de răspuns aferenţi operaţiunilor de citire/scriere a datelor. Timpul de răspuns al unei tranzacţii se calculează prin însumarea următoarele elemente: timpul de transmisie prin reţea, timpul de citire/scriere locală a datelor.

3. Strategii de alocare a datelor

După fragmentarea schemei globale a bazei de date, următorul pas este alocarea de fragmente. Există patru strategii de alocare a datelor aplicabile într-o bază de date relaţională distribuită:

1. Centralizată, în care toate datele vor fi amplasate pe un singur nod, numai utilizatorii fiind distribuiţi în reţea, şi este gestionată de un singur SGBD. În acest caz, dezavantajele sunt în special, costurile ridicate de comunicaţie şi o fiabilitate şi disponibilitate foarte scăzute, deoarece orice eroare care blochează accesul la baza de date întrerupe întreaga activitate în reţea.

2. Fragmentată (partiţionată), de asemenea, este numită şi **distribuire pură**,

distribution restrictions. Suppose there is a set of n fragments F = {F1, F2, ... Fn} exploited by a set of k applications A = {A1, A2, ..., Ak} on a set of m stations S = {S1, S2, ..., Sm}. The allocation problem reduces in finding the optimal distribution of F on S.

The optimal solution for the distribution can be defined by two *objectives*:

1. *Minimize costs*. The total cost is calculated by adding the costs of communication (transmission of messages and related data), the costs of integrating and updating operations of each fragment on each node (CPU utilization and operations for input / output) and storage costs of each fragment on each node;
2. *Maximizing performance*. Performances are measured especially by system response times when dealing with read/write data processing operations. Response time of a transaction is calculated by summing the following elements: the network transmission time and read/write operations processing time.

3. Data allocation strategies

After fragmentation of the global scheme of the database, the next step is the allocation of fragments. There are four data allocation strategies applicable in a distributed relational database:

1. **Centralized**, in which all data will be located on a single node, only users being distributed in the network, and is managed by a single DBMS. In this case, the disadvantages are especially high costs of communication and a very low reliability and availability, since any error that blocks access to the database interrupts the whole network activity.
2. **Fragmentation (partition)**, also called **pure distribution**, involves

implică partiţionarea bazei de date în fragmente disjuncte şi alocarea fiecărui fragment pe un singur nod. Avantajele acestei variante sunt costurile reduse cu stocarea şi costurile de comunicaţie. În schimb, disponibilitatea şi fiabilitatea datelor sunt scăzute, deşi acestea sunt mai bune decât în cazul centralizării.

3. Replicată complet, presupune existenţa unei copii complete a bazei de date pe toate nodurile din sistem. Orice schimbare în site trebuie să se reflecte în toate copiile site-ului. Principalele ei avantaje sunt siguranţa, disponibilitatea şi performanţele pentru operaţiile de interogare, dar costurile de depozitare sunt foarte ridicate, la fel sunt şi costurile de actualizare.

4. Replicată parţial (selectiv), este o combinaţie între partiţionare, replicare şi centralizare, în scopul de a cumula cât mai mult posibil avantajele fiecărei strategii şi să elimine dezavantajele. Aceasta implică partiţionarea bazei de date în *fragmente critice* şi *necritice*. Fragmentele necritice vor fi stocate pe un singur server, în timp ce fragmentele critice vor fi replicate pe mai multe noduri, în funcţie de cerinţele de disponibilitate şi performanţă ale sistemului. Datorită flexibilităţii, această strategie este de departe cea mai folosită în practică.

În tabelul 2 de mai jos, vom prezenta o comparaţie a acestor strategii de alocare a datelor într-o bază de date distribuită.

partitioning the database into disjoint fragments and allocation of each fragment on a single node. The advantages of this alternative are reduced storage and communication costs. Instead, data availability and reliability are low, although they are better than with centralization.

3. **Full replication,** involves the existence of a complete copy of the database on all nodes in the system. Any change in the site should be reflected in all copies of the site. Its main advantages are safety, availability and performance for query operations, but costs of storage are very high, so are the costs of update.
4. **Partial replication (Selective),** is a combination of partitioning, replication, and centralization in order to cumulate as much as possible the advantages of each approach and eliminate the disadvantages. It involves partitioning the database into critical and uncritical fragments. Uncritical fragments will be stored on a single server, while critical fragments will be replicated on multiple nodes, depending on availability and performance requirements of the system. Thanks to flexibility, this strategy is by far the most used in practice.

In Table 2 below, we present a comparison of these data allocation strategies in a distributed database.

	Centralized database	*Fragmented database*	*Full replication*	*Selective replication*
Data consistency	High	Medium	Low	Low
Storage costs	Low	Low	High	Medium
Reliability and availability	Low	Low for item, high for system	High	Medium
Speed of actualization operations	Medium	High	Low	Low
Communication costs	High	Low	High	Low

4. Metode de alocare a datelor

În proiectarea alocării datelor, următoarea regulă trebuie respectată: datele trebuie să fie plasate cât mai aproape de locul în care vor fi utilizate. În practică, metodele utilizate pentru alocarea datelor sunt:

- **Metoda de determinare a alocării neredundante**, de asemenea, numită şi **metoda celei mai bune alegeri**, constă în evaluarea fiecărei posibile alocări şi alegerea unui singur nod pentru fiecare fragment, respectiv a nodului cu beneficiile cele mai mari. Beneficiile vor fi calculate luând în considerare toate operaţiile de actualizare şi interogare. Metoda oferă o soluţie care elimină posibilitatea de a plasa un fragment la un anumit nod dacă un fragment este deja alocat acelui nod.
- **Metoda alocării redundante pe toate nodurile profitabile** implică selectarea tuturor nodurilor pentru care beneficiul alocării unei copii a fragmentului este mai mare decât costul alocării.
- **Metoda replicării progresive** implică construcţia iniţială a unei soluţii neredundante şi apoi introduce progresiv copii replicate începând cu nodul cele mai profitabil. În primul rând, fiecare fragment va fi alocat pe un nod pe baza profitului maxim. Decizia următoare va lua în considerare nodul pe care a fost plasat în prima fază un fragment şi profitul maxim pentru nodurile rămase. Această procedură va continua succesiv, fiind realizată o singură alocare în fiecare etapă şi se va încheia atunci când nici o replicare posibilă nu mai este profitabilă.

Concluzii

Obiectivul alocării datelor este de a găsi soluţia optimă în vederea minimizării costurilor şi maximizării performanţelor. Atingerea acestui obiectiv necesită definirea atentă a strategiei de alocare. Strategia de alocare implică analiza cerinţelor privind

4. Methods of data allocation

In designing data allocation, the following rule must be respected: the data must be placed closer to the place that will be used. In practice the methods used for data allocation are:

- **Method of determination of non redundant allocation** also called the best choices method consists in evaluating each possible allocation and choosing a single node for each fragment. The benefits will be calculated by taking into consideration all query and update operations. The method offers a solution that eliminates the possibility of placing a fragment at a particular node if a fragment is already assigned to that node.
- **The redundant allocation method on all profitable nodes** involves the selection of all nodes for which benefit of a copy allocation of the fragment is greater than the cost allocation.
- **Progressive replication method** involves initial construction of a non redundant solution and then progressively introduces the replicated copies from the most profitable node. First, each fragment will be allocated on a node based on the maximum profit. The next decision will consider the node on which was placed in the first phase a fragment and the maximum profit for the remaining nodes. This procedure will continue successively being performed one allocation in each phase and it will end when no possible replication is profitable.

Conclusions

The data allocation goal is to find optimal solution in the problem of fragments placement in terms of minimizing costs and maximizing performance. Achieving this objective requires careful definition of the allocation strategy. The allocation strategy involves the analysis of the requirements

consistența datelor, ponderea operațiilor de de citire/scriere în raport cu timpul de răspuns al sistemului. În funcție de aceste rezultate, putem găsi o combinație optimă între alocarea datelor şi replicarea datelor. Presupunând că replicile sunt mutual consistente, replicarea îmbunătățeşte disponibilitatea unei tranzacții deoarece poate citi oricare dintre copii. În plus, replicarea oferă mai multă fiabilitate, minimizează şansele de pierdere totală de date, şi îmbunătățeşte recuperarea datelor.

regarding data consistency, read/write operations percent in relation with system response time. Depending on these results, we can find the optimal combination between data distribution and data replication. Assuming replicas are mutually consistent, replication improves availability since a transaction can read any of the copies. Furthermore, replication provides more reliability, minimizes the chance of total data loss, and greatly improves disaster recovery.

Bibliografie

1. Ozsu M.T., Valduriez P., *Principles of Distributed Database Systems*, Second Edition, Prentice Hall International, 1998.
2. Elmasri R., Shamkant B., *Fundamentals of Database Systems*, Fifth Edition, Addison Wesley, 2006.
3. Connolly T., Begg C., *Database Systems. A practical approach to design, implementation and management*, Fourth Edition, Addison Wesley, 2005.
4. Date C. J., *An introduction to database systems*, Eight edition, Pearson Education , 2004.
5. Silberschatz A., Korth H.F., Sudarshan S., *Database System Concepts*, Sixth Edition, McGraw-Hill, 2010.
6. Beynon-Davies P., *Database systems*, 3rd Edition, Palgrave-Macmillan, 2004.

Bibliography

1. Ozsu M.T., Valduriez P., *Principles of Distributed Database Systems*, Second Edition, Prentice Hall International, 1998.
2. Elmasri R., Shamkant B., *Fundamentals of Database Systems*, Fifth Edition, Addison Wesley, 2006.
3. Connolly T., Begg C., *Database Systems. A practical approach to design, implementation and management*, Fourth Edition, Addison Wesley, 2005.
4. Date C. J., *An introduction to database systems*, Eight edition, Pearson Education , 2004.
5. Silberschatz A., Korth H.F., Sudarshan S., *Database System Concepts*, Sixth Edition, McGraw-Hill, 2010.
6. Beynon-Davies P., *Database systems*, 3rd Edition, Palgrave-Macmillan, 2004.