

7. Distributed Database Systems





7.1 Introduction

What is DDB?

A DDB is a collection of correlated data which are spread across a network and managed by a software called DDBMS.

Two kinds:

- (1) Distributed physically, centralized logically (general DDB)
- (2) Distributed physically, distributed logically too (FDBS)

We take the first as main topic in this course.



Features of DDBS :

- Distribution
- Correlation
- DDBMS



The advantages of DDBS:

- Local autonomy
- Good availability (because support multi copies)
- Good flexibility
- Low system cost
- High efficiency (most access processed locally, less communication comparing to centralized database system)
- Parallel process

The disadvantages of DDBS:

- Hard to integrate existing databases
- Too complex (system itself and its using, maintenance, etc. such as DDB design)



The main problems in DDBS:

Compared to centralized DBMS, the differences of DDBS are as follows:

- Query Optimization (different optimizing goal)
- Concurrency control (should consider whole network)
- Recovery mechanism (all sub-transactions must commit or abort simultaneously)

Another problem specially for DDBS:

- Data distribution



7.2 Data Distribution

7.2.1 Strategies of Data Distribution

- (1) Centralized: distributed system, but the data are still stored centralized. It is simplest, but there is not any advantage of DDB.
- (2) Partitioned: data are distributed without repetition. (no copies)
- (3) Replicated: a complete copy of DB at each site. Good for retrieval-intensive system.
- (4) Hybrid (mix of the above): an arbitrary fraction of DB at various sites. The most flexible and complex distributing method.



7.2 Data Distribution

7.2.1 Strategies of Data Distribution

- (1) Centralized: distributed system, but the data are still stored centralized. It is simplest, but there is not any advantage of DDB.
- (2) Partitioned: data are distributed without repetition. (no copies)
- (3) Replicated: a complete copy of DB at each site. Good for retrieval-intensive system.
- (4) Hybrid (mix of the above): an arbitrary fraction of DB at various sites. The most flexible and complex distributing method.



Comparison of four strategies

1

2

3

4

flexibility

complexity

Advantage of DDBS

Problems with DDBS



7.2.2 Unit of Data Distribution

- (1) According to relation(or file), that means non partition
- (2) According to fragments
 - Horizontal fragmentation: tuple partition
 - Vertical fragmentation: attribute partition
 - Mixed fragmentation: both



The criteria of fragmentation:

- (1) Completeness: every tuple or attribute must has its reflection in some fragments.
- (2) Reconstruction: should be able to reconstruct the original global relation.
- (3) Disjointness: for horizontal fragmentation.



7.2.3 Problems Caused by Data Distribution

- 1) Multi copies' consistency
- 2) Distribution consistency


Mainly the change of tuples' store location resulted by updating operation. Solution:

(1) Redistribution


After Update: Select->Move->Insert->Delete

(2) Piggybacking

Check tuple immediately while updating, if there is any inconsistency it is sent back along with ACK information and then sent to the right place.

- 
- 3) Translation of Global Queries to Fragment Queries and Selection of Physical Copies.
 - 4) Design of Database Fragments and Allocation of Fragments.

Above 1)~3) should be solved in DDBMS.
While 4) is a problem of distributed database design.

- 
- 3) Translation of Global Queries to Fragment Queries and Selection of Physical Copies.
 - 4) Design of Database Fragments and Allocation of Fragments.

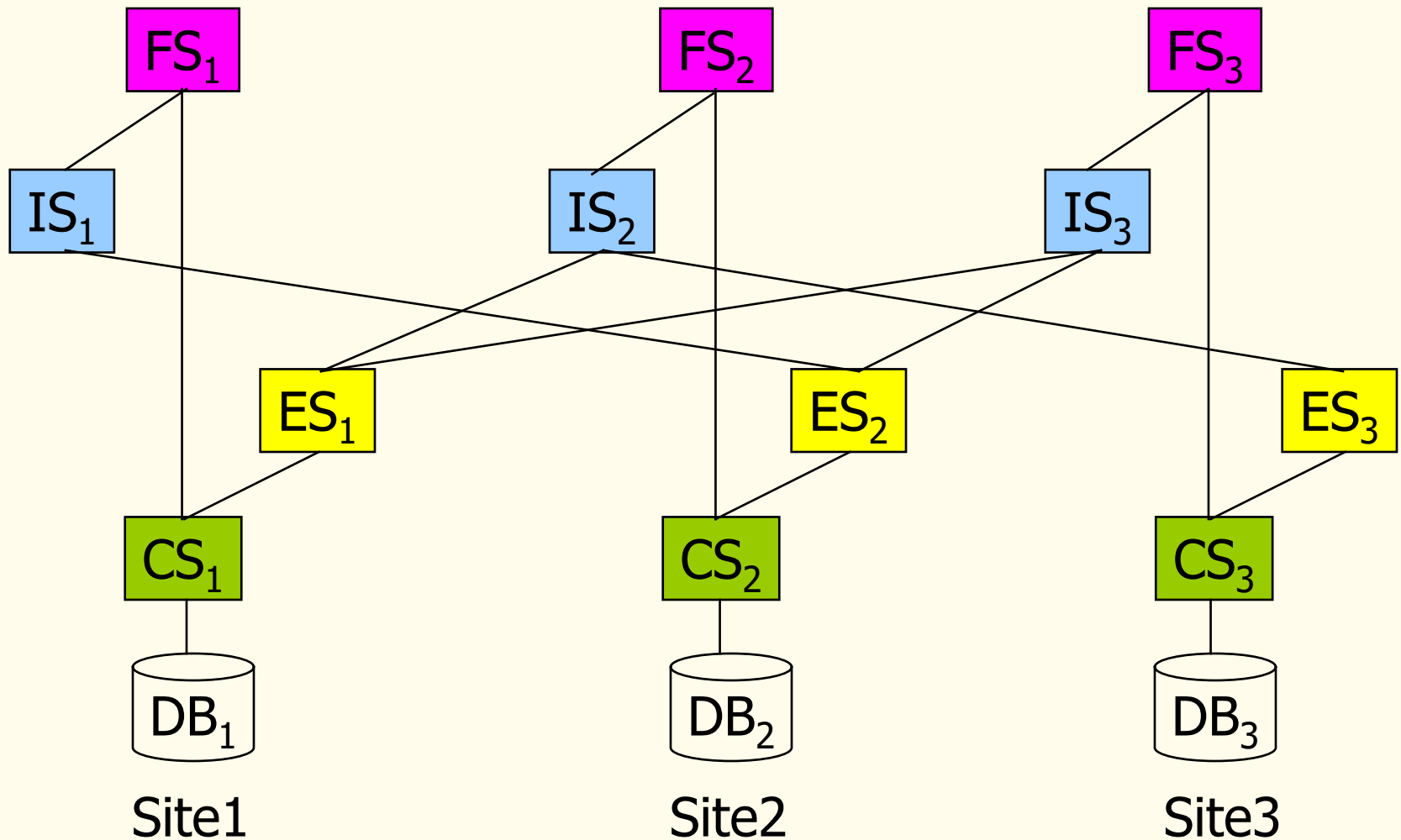
Above 1)~3) should be solved in DDBMS.
While 4) is a problem of distributed database design.




7.3 Federated Database

- In practical applications, there are strong requirements for solving the integration of multi existing, distributed and heterogeneous databases.
- The database system in which every member is autonomic and collaborate each other based on negotiation --- federated database system.
- No global schema in federated database system, every federated member keeps its own data schema.
- The members negotiate each other to decide respective input/output schema, then, the data sharing relations between each other are established.

The schema structure in federated database System



- 
- $FS_i = CS_i + IS_i$
 - FS_i is all of the data available for the users on $site_i$.
 - IS_i is gained through the negotiation with ES_j of other sites ($j \neq i$).
 - User's query on $FS_i \Rightarrow$ the sub-queries on CS_i and $IS_i \Rightarrow$ the sub-queries on corresponding ES_j .
 - The results gained from $ES_j \Rightarrow$ the result forms of corresponding IS_i , and combined with the results get from the sub-queries on CS_i , then synthesized to the eventual result form of FS_i .



7.4 Query Optimization in DDBMS

- Optimization goal: minimize the transmission cost on network
- Algebra optimization
- Translation of global queries to fragment queries and selection of physical copies
- Query Decomposition
- Global query plan



An example of global query optimization

R1

R2

Site1

Site2

```
Select *  
From R1,R2  
Where R1.a = R2.b;
```

Global query optimization may get an execution plan based on cost estimation, such as:

- (1) send R2 to site1, R'
- (2) execute on site1:

```
Select *  
From R1, R'  
Where R1.a = R'.b;
```



7.5 Recovery Mechanism in DDBMS

- The basic principle is the same as that in centralized DBMS
- **Distributed transactions** : the key of distributed transaction management is how to assure all sub-transactions either commit together or abort together.
- Realize the sub-transactions' harmony with each other relies on communication, while the communication is not reliable.
- Two phase commitment protocol
- Combination of failures



7.6 Concurrency Control in DDBMS

- The basic principle is the same as that in centralized DBMS, demand concurrent transactions to be scheduled serializably
- Because of multi copies, need locking globally
- Communication overhead
- Global deadlock