PURDUE
UNIVERSITY

CS54200: Distributed Database Systems

Distributed Database Design

23 February, 2009

Prof. Chris Clifton

Indiana
Center for
Database
Systems

Design Problem

- In the general setting:
  - Making decisions about the placement of data and programs across the sites of a computer network as well as possibly designing the network itself.
- In Distributed DBMS, this entails:
  - Placement of the distributed DBMS software; and
  - Placement of the applications that run on the database.

2

Distributed Design

- Top-Down
  - Mostly in designing systems from scratch
  - Mostly in homogeneous systems
- Bottom-Up
  - When the constituent databases already exist at a number of sites.

3

Distributed Design Issues

- Why fragment?
- How to fragment?
- How much to fragment?
- How to test correctness?
- How to allocate?
- Information requirements?

4

Fragmentation

- What is a reasonable unit of distribution?
  - Relations
    - Views are subsets of relations ➔ locality
    - Extra communication
  - Fragments of relations
    - Concurrent execution of a number of txns on the same relation
    - Views that cannot be defined on a single fragment will require extra processing
    - Semantic data control (especially integrity enforcement) more difficult

5

Types of fragmentation

- Horizontal
  - Divide tuples based upon certain properties, e.g. ranges.
- Vertical
  - Divide attributes
    - Need to replicate primary key attributes
- Hybrid
  - Alternating application of horizontal and vertical.

6

Correctness of fragmentation

- Completeness
  - Decomposition of Relation R into $R_1, R_2, \ldots R_n$ is complete if and only if each data item in R can also be found in some R_i
- Reconstruction
  - If Relation R is decomposed into $R_1, R_2, \ldots R_n$, then there should exist some operator, that R can be reconstructed from $R_1, \ldots R_n$.
- Disjointness
  - If Relation R is decomposed into $R_1, R_2, \ldots R_n$, and data item d is in R_j, then d should not be in any other fragment R_k, $k <> j$.

7

Allocation Alternatives

- Non-replicated
  - Partitioned: each fragment resides at only one site
- Replicated
  - Fully replicated
  - Partially replicated
- Rule of thumb:
  - If (read-only queries/update queries) >= 1 replication is advantageous

8

Comparison of alternatives

	Full Replication	Partial Replication	Partitioning
Query Processing	Easy	Same	Same
Directory Management	Easy or non-existant	Same	Same
Concurrency Control	Moderate	Difficult	Easy
Reliability	Very High	High	Low
Reality	Possible Application	Realistic	Possible application

9

Information Requirements

- Four categories of information are required for distributed database design:
  - *Database Information*
  - *Application Information*
  - *Communication network information*
  - *Computer system information*

10

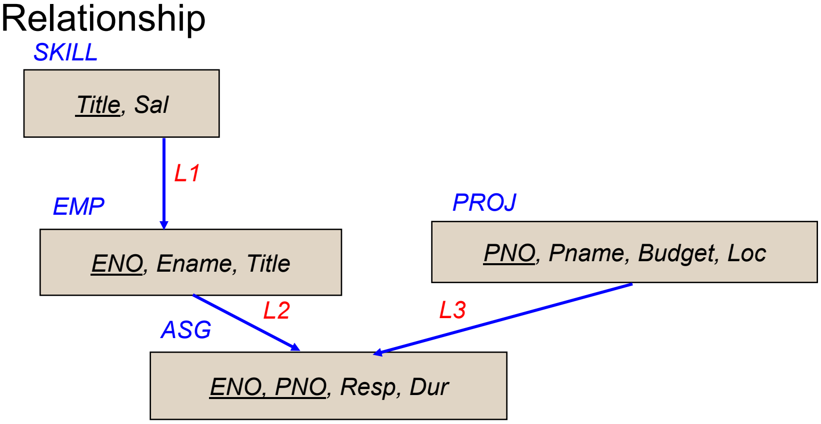
Horizontal Fragmentation

- There are two types:
  - Primary
    - Based upon values of attributes in the relation being fragmented
  - Derived
    - Based upon values of attributes of some other relation.

11

Primary Horizontal Fragmentation

- Database Information
  - Relationship



  - Cardinality of each relation, card(R)

12

PHF-Information Requirements

- Application Information
  - Simple predicates: Given $R[A_1, A_2, \ldots, A_n]$, a simple predicate p_j is:
    - P_j: A_i θ $Value$
    - where θ is a comparison operator, Value is from the domain of attribute A_i
  - Minterm predicates: Given R and $P_r=\{p_1, p_2, \ldots p_m\}$, define $M=\{m_1, m_2, \ldots, m_z\}$ as

$$M = \{m_i \mid m_i = \wedge_{pj \in \Pr} p_j^*\}, 1 \le i \le z$$

where $p_j^* = p_j$ or $NOT(p_j)$.

13

PHF – Information Requirements

- Examples
  - PNAME = “Maintenance” AND BUDGET <= 200000
  - NOT(PNAME=“Maintenance”) AND BUDGET <= 200000
  - PNAME = “Maintenance” AND NOT(BUDGET <=200000)
  - NOT(PNAME=“Maintenance”) AND NOT(Budget<=200000)

14

PHF-Information Req.

- Application Information
  - Minterm selectivities: $sel(m_i)$
    - The number of tuples of the relation that would be accessed by a user query which is specified according to a given minterm predicate m_i.
  - Access frequencies: $acc(q_i)$
    - The frequency with which a query q_i is accessed
    - Access frequency of a minterm predicate can also be defined.

15

Primary Horizontal Frag.

- Definition: $R_j = \sigma_{Fj}(R), 1 \leq j \leq w$
  - Where F_j is a selection formula, which is (preferably) a minterm predicate.
- Therefore,
  - A horizontal fragment, R_i of relation R consists of all the tuples of R which satisfy a minterm predicate m_i ➔
  - Given a minterm of predicates M, there are as many horizontal fragments of relation R as there are minterm predicates
  - Set of horizontal fragments also referred to as minterm fragments.

16

PHF - Algorithm

- GIVEN: A relation R, the set of simple predicates P_r
- OUTPUT: The set of fragments of $R = \{R_1, \ldots, R_w\}$ which obey the fragmentation rules.
- Preliminaries:
  - P_r should be complete
  - P_r should be minimal

17

Completeness of Simple Predicates

- A set of simple predicates P_r is said to be *complete* iff the accesses to the tuples of the minterm fragments defined on P_r requires that two tuples of the same minterm fragment have the same probability of being accessed by the application.
- Example:
  - Assume PROJ[PNO, PNAME, BUDGET, LOC] has two applications defined on it.
  - Find the budgets of projects at each location. (1)
  - Find projects with budgets less than $200000. (2)

18

Completeness of Simple Predicates

- According to (1),
  - P_r = {LOC="Montreal", LOC="New York", LOC="Paris"}
- Which is not complete with respect to (2).
- Modify
  - P_r = {LOC="Montreal", LOC="New York", LOC="Paris",BUDGET<=200000, BUDGET >200000}
- Which is complete.

19

Minimality of Simple Predicates

- If a predicate influences how fragmentation is performed, (I.e. causes a fragment f to be further fragmented into, say fi and fj) then there should be at least one application that accesses fi and fj differently.
- In other words, the simple predicate should be relevant in determining a fragmentation.
- If all the predicates of a set Pr are relevant, then Pr is minimal. $\frac{acc(m_i)}{card(f_i)} \neq \frac{acc(m_j)}{card(f_j)}$

20

COM-MIN Algorithm

- Given: a relation *R* and a set of simple predicates P_r.
- Output: a complete and minimal set of simple predicates P_r' for P_r.

- *Rule 1:* a relation or fragment is partitioned into at least two parts which are accessed differently by at least one application.

21

PHORIZONTAL Algorithm

- Makes use of COM_MIN to perform fragmentation.
- Input: a relation *R* and a set of simple predicates P_r
- Output: a set of minterm predicates *M* according to which *R* is to be fragmented.
  1. $P_r' \leftarrow COM\_MIN(R, P_r)$
  2. Determine the set *M* of minterm predicates
  3. Determine the set *I* of implications among p_i from P_r.
  4. Eliminate the contradictory minterms from *M*

22

PHF - Example

- Two candidate relations: PAY and PROJ.
- Fragmentation of relation PAY
  - Application: check the salary info and determine raise.
  - Employee records kept at two sites ➔ application run at two sites
  - Simple predicates
    - p_1 : SAL <= 30000
    - p_2 : SAL > 30000
    - $P_r = \{p_1, p_2\}$ which is complete and minimal $P_r' = P_r$
  - Minterm predicates
    - m_1 : (SAL <= 30000)
    - m_2 : NOT(SAL <= 30000) = (SAL>30000)

23

PHF Example

PAY_1

TITLE	SAL
Mech. Eng.	27000
Programmer	24000

PAY_2

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000

24

Fragmentation of PROJ

- Applications:
  - Find the name and budget of projects given their no. – issued at three sites
  - Access project information according to budget
    - One site accesses <=200000 another accesses > 200000
- Simple Predicates
  - For application 1:
    - p_1 : LOC = "Montreal"
    - p_2 : LOC = "New York"
    - p_3 : LOC = "Paris"
  - For application 2:
    - P_4: BUDGET <= 200000
    - P_5: BUDGET > 200000
  - $P_r = P_r' = \{p_1, p_2, p_3, p_4, p_5\}$

25

PHF Example

- Fragmentation of PROJ contd:
  - Minterm fragments left after elimination
  - m_1: (LOC = "Montreal") AND (BUDGET <=200000)
  - m_2: (LOC = "Montreal") AND (BUDGET>200000)
  - m_3: (LOC = "New York") AND (BUDGET <=200000)
  - m_4: (LOC = "New York") AND (BUDGET >200000)
  - m_5: (LOC = "Paris") AND (BUDGET <=200000)
  - m_6: (LOC = "Paris") AND (BUDGET >200000)

26

PHF -- Example

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instr.	150000	Montreal

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P2	Database Develop.	135000	New York

$PROJ_4$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	250000	New York

$PROJ_6$

PNO	PNAME	BUDGET	LOC
P4	Maint.	310000	Paris

27

PHF – Correctness

- Completeness
  - Since P_r' is complete and minimal, the selection predicates are complete
- Reconstruction
  - If relation R is fragmented into $F_R=\{R_1, R_2, \ldots R_r\}$

$$R = \bigcup_{\forall R_i \in F_R} R_i$$

- Disjointness
  - Minterm predicates that form the basis of fragmentation should be mutually exclusive.

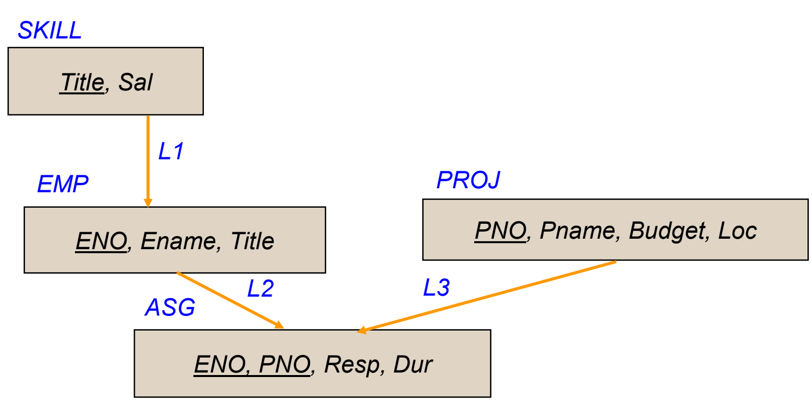
28

Derived Horizontal Fragmentation

- Defined on a member relation of a link according to a selection operation specified on its owner.
  - Each link is an equijoin
  - Equijoin can be implemented by means of semijoins.

30

Derived Horizontal Fragmentation



31

DHF -- Definition

- Given a link L where owner(L)=S and member(L) = R, the derived horizontal fragments of R are defined as

$$R_i = R \ltimes_F S_i, 1 \leq i \leq w$$

where w is the maximum number of fragments that will be defined on R and $S_i = \sigma_{F_i}(S)$ where Fi is the formula according to which the primary horizontal fragment Si is defined.

32

DHF -- Example

- Given link L1 where owner(L1)=SKILL and member(L1)=EMP

$$EMP_1 = EMP \ltimes SKILL_1$$
$$EMP_2 = EMP \ltimes SKILL_2$$

where

$$SKILL_1 = \sigma_{SAL \leq 30000}(SKILL)$$

$$SKILL_2 = \sigma_{SAL > 30000}(SKILL)$$

33

DHF – Example

EMP1

ENO	ENAME	TITLE
E3	B. Lee	Mech. Eng.
E4	J. Miller	Programmer
E7	R. Davis	Mech. Engr.

EMP2

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E8	J. Jones	Syst. Anal.

34

DHF – Correctness

- Completeness
  - Referential Integrity
  - Let *R* be the member relation of a link whose owner is relation *S* which is fragmented as *Fs={S1, S2, …, Sn}*. Furthermore, let *A* be the join attribute between R and *S*. Then, for each tuple *t* of *R*, there should be a tuple *t'* of *S* such that *t[A]=t'[A]*
- Reconstruction
  - Same as primary HF
- Disjointness
  - Simple join graphs between the owner and member fragments

35

Vertical Fragmentation

- Has been studied within the centralized context
  - Design methodology
  - Physical clustering
- More difficult than horizontal, because more alternatives exist. Two approaches:
  - Grouping
    - Attributes to fragments
  - Splitting
    - relation to fragments

36

Vertical Fragmentation

- Overlapping Fragments
  - Grouping
- Non-overlapping Fragments
  - Splitting
- We do not consider the replicated key attributes to be overlapping.
- Advantage:
  - Easier to enforce functional dependencies

37

VF – Information Requirements

- Application Information
  - Attribute affinities
    - A measure that indicates how closely related the attributes are
    - This is obtained from more primitive usage data
  - Attribute usage values
    - Given a set of queries $Q=\{q_1, q_2, \ldots, q_k\}$ that will run on the relation $R[A_1, A_2, \ldots, A_n]$,
    - Use(q_i, A_j) = 1 if A_j is referenced by qi, 0 otherwise
    - Use$(qi,\cdot)$ can be defined accordingly

38

VF – Definition of use(qi,Aj)

- Consider the following 4 queries for PROJ

```
SELECT BUDGET
FROM PROJ
WHERE PNO=Value
```

```
SELECT PNAME, BUDGET
FROM PROJ
```

```
SELECT PNAME
FROM PROJ
WHERE LOC=Value
```

```
SELECT SUM(BUDGET)
FROM PROJ
WHERE LOC=Value
```

- Let A1=PNO, A2=PNAME, A3=BUDGET, A4=LOC

	A_1	A_2	A_3	A_4
q_1	1	0	1	0
q_2	0	1	1	0
q_3	0	1	0	1
q_4	0	0	1	1

39

VF – Affinity Measure aff(Ai,Aj)

- The attribute affinity measure between two attributes Ai and Aj of a relation R with respect to the set of applications Q={q1, q2, …, qk} is defined as follows:

$$aff(A_i, A_j) = \sum_{\text{all queries that access Ai and Aj}} (\text{query access})$$

$$\text{query access} = \sum_{allsites} \text{access freq of a query} * \frac{\text{access}}{\text{execution}}$$

40

VF – Clustering Algorithm

- Take the attribute affinity matrix AA and reorganize the attribute orders to form clusters where the attributes in each cluster have high affinity for each other
- Bond Energy Algorithm (BEA) has been used for clustering of attributes. This algorithm finds clustering such that the global affinity measure

$$AM = \sum_i \sum_j (\text{affinity of } A_i \text{ and } A_j \text{ with thei r neighbors })$$

is maximized.

41

Bond Energy Algorithm

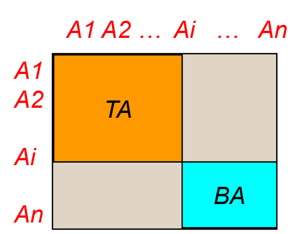
- Input: the AA matrix
- Output: the clustered affinity matrix CA (a perturbation of AA)

1. Initialization: Place and fix one of the columns of AA in CA
2. Iteration: Place the remaining n-I columns in the remaining I+1 positions in the CA matrix. For each column, chose the placement that makes the most contribution to the global affinity measure.
3. Row Order: Order the rows according to the columns.

42

VF Algorithm

- How can you divide a set of clustered attributes $\{A_1, A_2, \ldots, A_n\}$ into two (or more) sets $\{A_1, \ldots, A_i\}$ and $\{A_{i+1}, \ldots, A_n\}$ such that there are no (or minimal) applications that access both (or more than one) of the sets?



43

VF -- Algorithm

- Define
  - TQ – set of applications that access only TA
  - BQ – set of applications that access only BA
  - CQ – set of applications that access both
- And
  - CTQ – total number of accesses to attributes by applications that access only TA
  - CBQ – total number of accesses to attributes by applications that access only TB
  - COQ – total number of accesses to attributes by applications that access both TA and TB
- Then find the point along the diagonal that maximizes $CTQ * CBQ – COQ^2$

44

VF – Algorithms

- Two problems:

1. Cluster forming in the middle of CA
   1. Shift a row up, and a column left and apply the algorithm to find the "best" partitioning point
   2. Do this for all possible shifts
   3. Cost $O(m^2)$
2. More than two clusters
   1. *M*-way partitioning
   2. Try 1,2, … m-1 split points along the diagonal and try to find the best point for each of these
   3. *Cost* $O(2^m)$

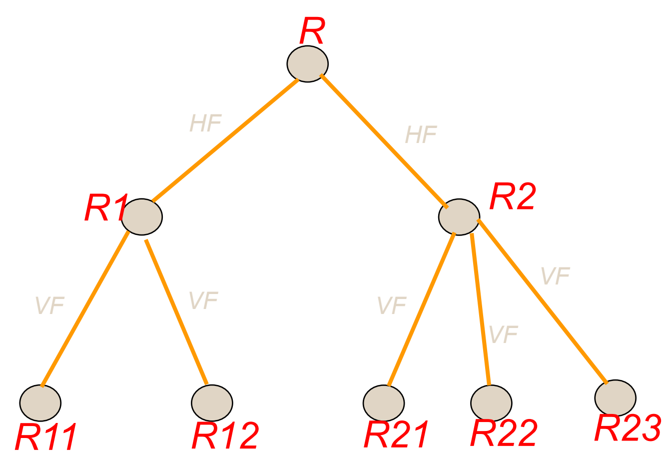
45

VF -- Correctness

- A relation *R*, defined over attribute set *A*, and key *K*, generates the vertical partitioning $F_R = \{R_1, R_2, \ldots, R_r\}$.
- Completeness. $A = \bigcup A_{R_i}$
- Reconstruction. $R = \bowtie_K R_i, \quad \forall R_i \in F_R$
- Disjointness:
  - TIDs are not considered to be overlapping since they are maintained by the system
  - Duplicated keys are not considered to be overlapping

46

Hybrid Fragmentation



47

Fragment Allocation

- Problem:
  - Given
    - *{F1, F2, …, Fn}* Fragments
    - *{S1, S2, …, Sm}* Sites
    - *{Q1, Q2, …, Qq}* Applications
  - Find the "optimal" distribution of F to S.
- Optimality
  - Minimal cost
    - Communication + Storage + processing
    - Cost is usually in terms of time
  - Performance
    - Response time and/or throughput
  - Constraints
    - Per site constraints (storage and processing)

48

Information Requirements

- Database information
  - Selectivity of fragments
  - Size of fragments
- Application information
  - Access types and numbers
  - Access localities
- Communication information
  - Unit cost of storing data at a site
  - Unit cost of processing at a site
- Computer system information
  - Bandwidth
  - Latency
  - Communication overhead

49

Allocation

- File Allocation (*FAP*) vs. Database Allocation (*DAP*)
  - Fragments are not individual files
    - Relationships have to be maintained
  - Access to database is more complicated
    - Remote file access model is not applicable
    - Relationship between allocation and query processing
  - Cost of integrity enforcement should be considered
  - Cost of concurrency control should be considered

50

Allocation – information requirements

- Database information
  - Selectivity of fragments, size of a fragment
- Application information
  - Number of read (update) accesses of a query to a fragment
  - A matrix of which queries update which fragments
  - A similar matrix for retrievals
  - Originating site of each query
- Site information
  - Unit cost of storing (processing) data
- Network information
  - Communication cost/frame between two sites
  - Frame size

51

Allocation Model

- General Form
  - Min(Total Cost) subject to
    - Response time constraint
    - Storage constraint
    - Processing constraint
- Decision Variable

$$x_{ij} = \begin{cases} 1 & \text{if Fragment } F_i \text{ is stored at Site } S_j \\ 0 & \text{otherwise} \end{cases}$$

52

Allocation Model

- Total Cost

$$\sum_{\text{all queries}} \text{processing cost} + \sum_{\text{all sites}} \sum_{\text{all fragments}} \text{cost of storing a fragment at a site}$$

- Storage Cost (of Fragment F_j at site S_k)

  (unit cost of storage at S_k)*size of F_j *x_{jk}

- Query Processing Cost (for one query)

  (processing component)+(transmission component)

53

Allocation Model

- Query Processing Cost
  - Processing component:
    access cost + integrity enforcement cost + concurrency control cost
  - Access cost:

$$\sum_{\text{all sites}} \sum_{\text{all fragments}} (\text{no. of update accesses} + \text{no. of read accesses}) * x_{ij} * \text{local processing cost at a site}$$

  - Other costs can be similarly calculated.

54

Allocation Model

- Query Processing Cost
  - Transmission component:
    cost of processing updates + cost of processing retrievals
  - Cost of updates:

$$\sum_{\text{all sites}} \sum_{\text{all fragments}} (\text{update message cost}) + \sum_{\text{all sites}} \sum_{\text{all fragments}} (\text{acknowledgement cost})$$

  - Retrieval cost:

$$\sum_{\text{all fragments}} \min_{\text{all sites}} (\text{cost of retrieval command} + \text{cost of sending back the result})$$

55

Allocation Model

- Constraints:
  - Response Time:
    execution time of query <= max allowable response time for that query
  - Storage constraint (for a site):

$$\sum_{\text{all fragments}} (\text{storage requirements of a fragment at that site}) \leq \text{storage capacity at site}$$

  - Processing constraint (for a site):

$$\sum_{\text{all queries}} (\text{processing load of a query at that site}) \leq \text{processing capacity at site}$$

56

Allocation Model

- Solution Methods:
  - FAP is NP-Complete
  - DAP also NP-Complete
- Heuristic based upon
  - Single commodity warehouse location (for FAP)
  - Knapsack problem
  - Branch and bound techniques
  - Network flow

57

Allocation Model

- Attempts to reduce the solution space
  - Assume all candidate partitionings known; select the "best" one
  - Ignore replication at first
  - Sliding window on fragments.

58