Systems Infrastructure for Data Science

Web Science Group
Uni Freiburg
WS 2013/14

Lecture VII: Fragmentation

Fragmentation

- Fragments should be subsets of database relations due to two main reasons:
  - **Access locality:** Application views are subsets of relations. Also, multiple views that access a relation may reside at different sites.
  - **Query concurrency and system throughput:** Sub-queries can operate on fragments in parallel.
- Main issues:
  - Views that cannot be defined on a single fragment will require extra **processing and communication cost**.
  - **Semantic data control** (e.g., integrity checking) of dependent fragments residing at different sites is more complicated and costly.

Fragmentation Alternatives

- Horizontal fragmentation (aka Sharding)
  - Primary horizontal fragmentation
  - Derived horizontal fragmentation
- Vertical fragmentation (=> Column Stores)
- Hybrid fragmentation

Example Database

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

ASG

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E8	P3	Manager	40

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

PAY

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

Horizontal Fragmentation Example

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

Projects with BUDGET < $200,000

Projects with BUDGET ≥ $200,000

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	255000	New York
P4	Maintenance	310000	Paris

Vertical Fragmentation Example

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

Project budgets

Project names and locations

$PROJ_1$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000

$PROJ_2$

PNO	PNAME	LOC
P1	Instrumentation	Montreal
P2	Database Develop.	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris

Hybrid Fragmentation Example

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris

Projects with BUDGET < $200,000

Horizontal

Projects with BUDGET ≥ $200,000

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	255000	New York
P4	Maintenance	310000	Paris

Project budgets

Vertical

Project names and locations

$PROJ_1$

PNO	BUDGET
P1	150000
P2	135000

$PROJ_2$

PNO	PNAME	LOC
P1	Instrumentation	Montreal
P2	Database Develop.	New York

.....

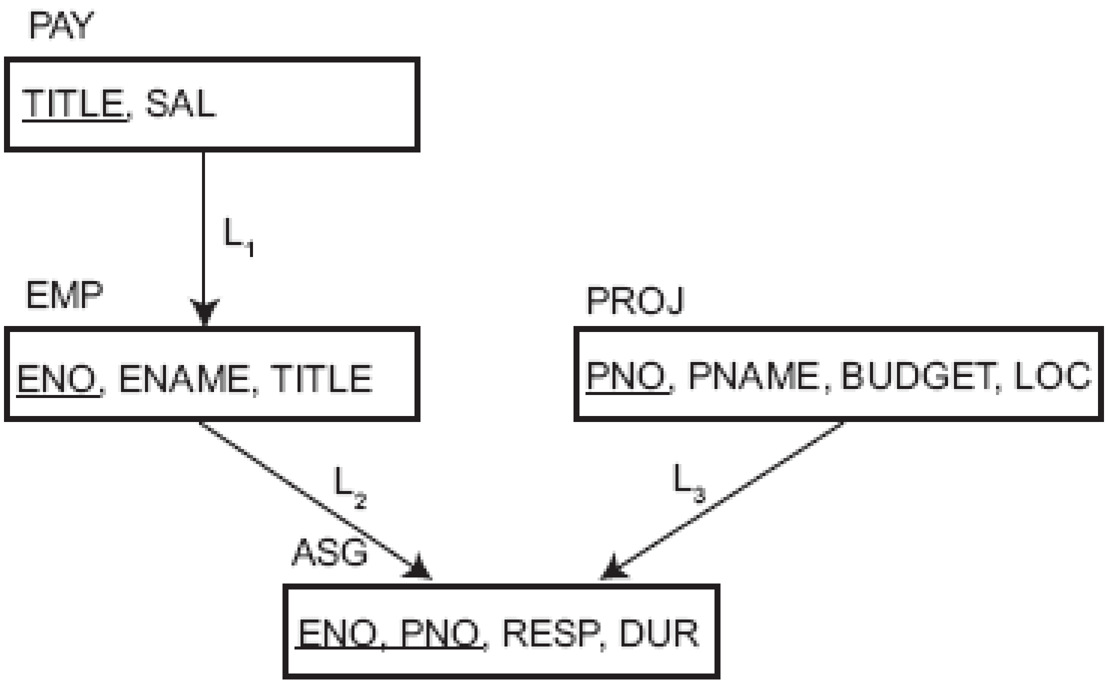
.....

Correctness of Fragmentation

- Completeness
  - Decomposition of relation R into fragments R_1, R_2, .., R_n is complete iff each data item in R can also be found in one or more of R_i's.
- Reconstruction
  - If a relation R is decomposed into fragments R_1, R_2, .., R_n, then there should exist a relational operator θ such that $R = \theta_{1 \leq i \leq n} R_i$.
- Disjointness
  - If a relation R is horizontally (*vertically*) decomposed into fragments R_1, R_2, .., R_n, and data item d_i (*non-primary key attribute d_i*) is in R_j, then d_i should not be in any other fragment R_k ($k \neq j$).

Horizontal Fragmentation Algorithms
What is given?

- Relationships among database relations



L_i: one-to-many relationship from an "owner" to a "member"

Horizontal Fragmentation Algorithms
What is given?

- Cardinality of each database relation
- Mostly used **predicates** in user queries
- Predicate selectivities
- Access frequencies for data

Horizontal Fragmentation Algorithms
Predicates

- Simple predicate
  - Given $R(A_1, A_2, .., A_n)$, a simple predicate p_j is defined as "p_j: A_i θ value", where θ ϵ {=, <, ≤, >, ≥, ≠} and value ϵ D_i, where D_i is the domain of A_i.
  - Examples:

    PNAME = "Maintenance"

    BUDGET ≤ 200000

- Minterm predicate
  - A conjunction of simple and negated simple predicates
  - Examples:

    PNAME = "Maintenance" AND BUDGET ≤ 200000

    NOT(PNAME = "Maintenance") AND BUDGET ≤ 200000

Primary Horizontal Fragmentation Definition

- Given an owner relation R, its horizontal fragments are given by

  $$R_i = \sigma_{F_i}(R),\ 1 \le i \le w$$

  where F_i is a minterm predicate.
- First step: Determine a set of simple predicates that will form the minterm predicates. This set of simple predicates must have two key properties:
  - completeness
  - minimality

Completeness of Simple Predicates Definition

- A set of simple predicates P is complete iff the accesses to the tuples of the minterm fragments defined on P requires that two tuples of the same minterm fragment have the **same probability of being accessed by any application**.

Completeness of Simple Predicates

Example

Set of simple predicates:

P = {LOC="Montreal",
LOC="New York",
LOC="Paris"}

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York

$PROJ_3$

PNO	PNAME	BUDGET	LOC
P4	Maintenance	310000	Paris

App 1: Find the budgets of projects at each location.
App 2: Find projects with budgets less than $200000.

P = {LOC="Montreal", LOC="New York", LOC="Paris",
BUDGET ≤ 200000, BUDGET > 200000}

complete

Minimality of Simple Predicates
Definition

- A set of simple predicates P is complete iff for each predicate p є P:
  - if p influences how fragmentation is performed (i.e., causes a fragment f to be further fragmented into f_i anf f_j), then there should be at least one application that accesses f_i and f_j differently.

Minimality of Simple Predicates
Example

App 1: Find the budgets of projects at each location.
App 2: Find projects with budgets less than $200000.

P = {LOC="Montreal", LOC="New York", LOC="Paris",
BUDGET ≤ 200000, BUDGET > 200000}

✓ complete & minimal

↓ + PNAME="Instrumentation"

P = {LOC="Montreal", LOC="New York", LOC="Paris",
BUDGET ≤ 200000, BUDGET > 200000,
PNAME="Instrumentation"}

✗ complete & NOT minimal

Primary Horizontal Fragmentation Example

- PAY(title, sal) and PROJ(pno, pname, budget, loc)
- 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)

Primary Horizontal Fragmentation Example

PAY_1

TITLE	SAL
Mech. Eng.	27000
Programmer	24000

PAY_2

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

Primary Horizontal Fragmentation Example

- Fragmentation of relation PROJ
  - App1: Find the name and budget of projects given their location. (issued at 3 sites)
  - App2: Access project information according to budget (one site accesses ≤ 200000, other accesses > 200000)
  - Simple predicates
    - For App1:
      p_1 : LOC = “Montreal”
      p_2 : LOC = “New York”
      p_3 : LOC = “Paris”
    - For App2:
      p_4 : BUDGET ≤ 200000
      p_5 : BUDGET > 200000
    - $P_r = P_r' = \{p_1, p_2, p_3, p_4, p_5\}$

Primary Horizontal Fragmentation Example

- Fragmentation of relation PROJ
  - 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)

Primary Horizontal Fragmentation Example

PROJ$_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal

PROJ$_3$

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	Maintenance	310000	Paris

Primary Horizontal Fragmentation
Correctness

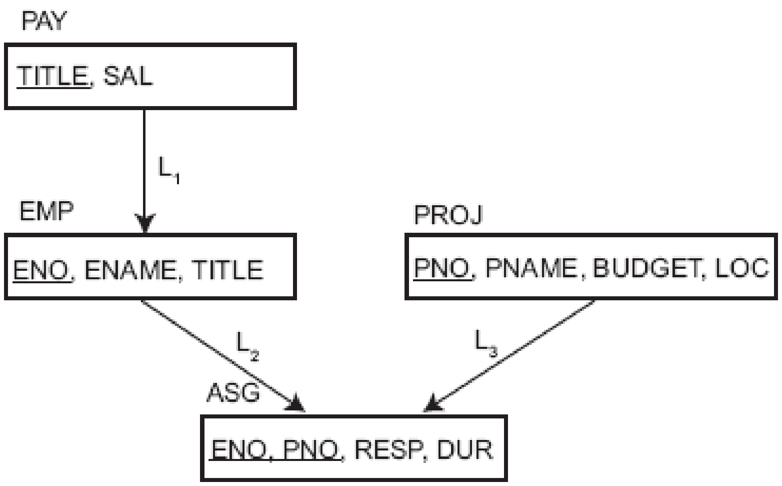
- Completeness
  - Since P_r' is complete and minimal, the selection predicates are complete.
- Reconstruction
  - If relation R is fragmented into $F_R = \{R_1, R_2, .., R_r\}$

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

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

Derived Horizontal Fragmentation

- Defined on a member relation of a link according to a selection operation specified on its owner.
- Two important points:
  – Each link is an equi-join.
  – Equi-join can be implemented using semi-joins.



Semi-join

- Given R(A) and S(B), semi-join of R with S is defined as follows:

$$R \ltimes_F S = \Pi_A(R \bowtie_F S) = \Pi_A(R) \bowtie_F \Pi_{A \cap B}(S)$$
$$= R \bowtie_F \Pi_{A \cap B}(S)$$

- Example:

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

PAY

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

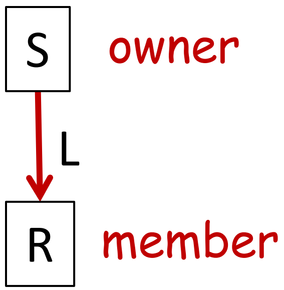
EMP $\ltimes_{\text{EMP.TITLE=PAY.TITLE}}$ PAY

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Analyst
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

Semi-join reduces the amount of data that needs to be transmitted btw sites.

Derived Horizontal Fragmentation

- Given relations R and S:



- The derived horizontal fragments of R are defined as

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

where w is the maximum number of fragments that will be defined on R, and

$$S_i = \sigma_{F_i}(S)$$

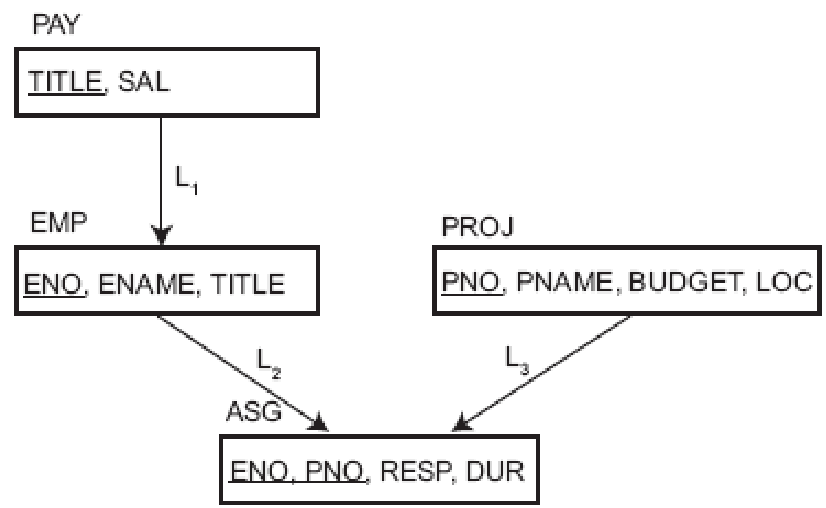
where F_i is the formula according to which the primary horizontal fragment S_i is defined.

Derived Horizontal Fragmentation Example

Primary horizontal fragments of PAY:

$PAY_1 = \sigma_{sal \leq 30000}(PAY)$

$PAY_2 = \sigma_{sal > 30000}(PAY)$



Derived horizontal fragments of EMP:

$EMP_1 = EMP \ltimes PAY_1$

$EMP_2 = EMP \ltimes PAY_2$

EMP_1

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

EMP_2

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.

Derived Horizontal Fragmentation Correctness

- Completeness
  - Referential integrity
  - Let R be the member relation of a link whose owner is relation S which is fragmented as $F_S = \{S_1, S_2, .., S_n\}$. 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
  - If relation R is fragmented into $F_R = \{R_1, R_2, .., R_r\}$

    $$R = \bigcup_{R_i \in F_R} R_i$$
- Disjointness
  - Simple join graphs between the owner and the member fragments.

Vertical Fragmentation

- Divide a relation R into fragments R_1, R_2, .., R_r, each of which contains a subset of R's attributes as well as the primary key of R.
- Goal: to minimize the execution time of user applications that run on these fragments.
- Too many alternatives => Use heuristic solutions based on:
  - Grouping: merge attributes to fragments
  - Splitting: divide a relation into fragments
- We need **togetherness** measure

Vertical Fragmentation Algorithms
What is given?

- **Attribute usage matrix** of the application queries
- Example: PROJ(PNO, PNAME, BUDGET, LOC)

Q1: SELECT **BUDGET**
FROM PROJ
WHERE PNO=110

Q2: SELECT **PNAME, BUDGET**
FROM PROJ

Q3: SELECT **PNAME**
FROM PROJ
WHERE LOC="New York"

Q4: SELECT SUM(**BUDGET**)
FROM PROJ
WHERE LOC="New York"

$$\begin{array}{c} \\ Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & A_3 & A_4 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Vertical Fragmentation Algorithms
What is given?

- **Attribute affinity matrix**
- Togetherness measure for attribute pairs
- Given a relation $R(A_1, A_2, .., A_n)$, the affinity between A_i and A_j w.r.t. a set of application queries $Q = \{Q_1, Q_2, .., Q_q\}$ is defined as follows:

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

↑ comes from the attribute usage matrix

$$\text{query access} = \sum_{\text{all sites}} \text{access frequency of a query} * \frac{\text{access}}{\text{execution}}$$

Vertical Fragmentation
Algorithm Sketch

- Cluster step: Permute rows and columns of the attribute affinity matrix to generate a **clustered affinity matrix** where attributes in each cluster are in high affinity to each other.

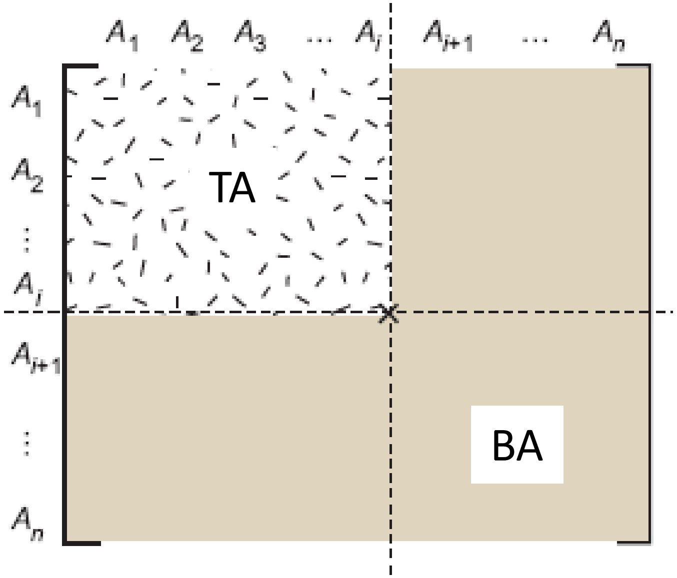
	A_1	A_2	A_3	A_4
A_1	45	0	45	0
A_2	0	80	5	75
A_3	45	5	53	3
A_4	0	75	3	78

	A_1	A_3	A_2	A_4
A_1	45	45	0	0
A_3	45	53	5	3
A_2	0	5	80	75
A_4	0	3	75	78

[Bond Energy Algorithm]

Vertical Fragmentation
Algorithm Sketch

- Partition step: Divide the clustered attributes into non-overlapping partitions such that the number of application queries that access to more than one partition is as small as possible.



Given:
TQ = set of applications that access only TA
BQ = set of applications that access only BA
OQ = set of applications that access both TA and BA
CTQ = total number of accesses to attributes by TQ
CBQ = total number of accesses to attributes by BQ
COQ = total number of accesses to attributes by OQ

Find:
The point along the diagonal that maximizes

$$CTQ*CBQ-COQ^2$$

Vertical Fragmentation
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
  - The following should be true for A:

$$A = \bigcup A_{R_i}$$

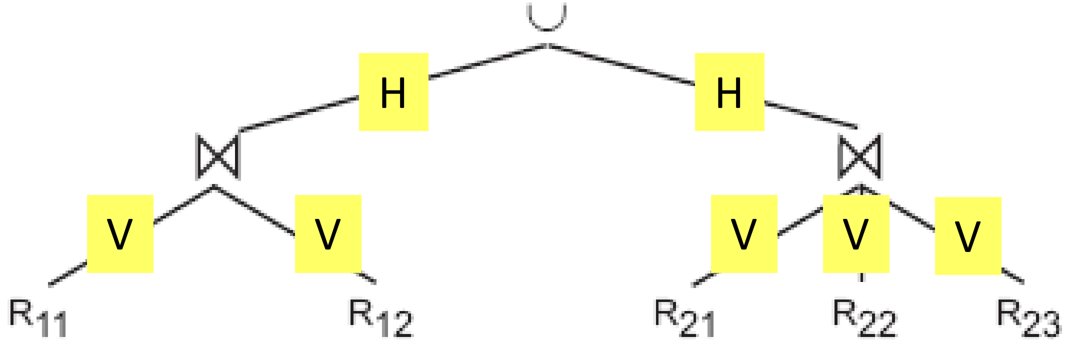
- Reconstruction
  - Reconstruction can be achieved by

$$R = \bowtie_K R_i, \forall R_i \in F_R$$

- Disjointness
  - Duplicated keys are not considered to be overlapping

Hybrid Fragmentation

- Obtained by applying horizontal and vertical fragmentation one after the other.
- In practice, nesting level does not exceed 2.
- Correctness properties are guaranteed if constituent fragmentations are correct.
- Bottom-up reconstruction:



Fragment Allocation

- Problem definition:
  - Given a set of fragments F, a set of network sites S, and a set of application queries Q, find the optimal distribution of F to S.
- Optimality measures:
  - Minimal cost = communication + storage + processing
  - Optimal performance = response time and/or throughput
- Complex problem, heuristic solutions

Fragment Allocation High-Level Model

- Minimize(total cost)
- Subject to
  - Response time constraint
  - Storage constraint
  - Processing constraint
- Decide on variable x_{ij}

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

Fragment Allocation Algorithms
What is given?

- Size of a fragment in bytes
- Selectivity of a fragment w.r.t. a query
- Number of read and update accesses of a query on a fragment
- Access localities
- Max. response time for each application
- Costs and capacities of sites

Fragment Allocation Alternatives

- Non-replicated
  - Partitioned: each fragment at only one site
- Replicated
  - Fully replicated: each fragment at each site
  - Partially replicated: each fragment at some of the sites
- Rule of thumb:
  - If $\frac{\text{read-only queries}}{\text{update queries}} \geq 1$, then replication pays off.

Fragment Allocation Alternatives

	Full replication	Partial replication	Partitioning
QUERY PROCESSING	Easy	← Same difficulty →	
DIRECTORY MANAGEMENT	Easy or nonexistent	← Same difficulty →	
CONCURRENCY CONTROL	Moderate	Difficult	Easy
RELIABILITY	Very high	High	Low
REALITY	Possible application	Realistic	Possible application