CS4224/CS5424 Lecture 2
Data Partitioning

Partitioning in Centralized Databases

```
CREATE TABLE measurement (
    city_id         int not null,
    logdate         date not null,
    peaktemp        int,
    unitsales       int
) PARTITION BY RANGE (logdate);


CREATE TABLE measurement_y2006m02 PARTITION OF measurement
    FOR VALUES FROM ('2006-02-01') TO ('2006-03-01');

CREATE TABLE measurement_y2006m03 PARTITION OF measurement
    FOR VALUES FROM ('2006-03-01') TO ('2006-04-01');
```

Source: https://www.postgresql.org/docs/current/ddl-partitioning.html

Distributed Database Design Issues

- **Data fragmentation / partitioning** - how to partition data into smaller pieces
- **Data allocation** - how to allocate data to various sites
- **Data replication** - what data to replicate at each site

Data Fragmentation

- Data Fragmentation / Partitioning
  - Partition data into pieces for distributed storage
- Relation R is partitioned into fragments $\{R_1, \cdots, R_n\}$
- Each R_i could be defined by a RA expression on R
- **Example**: `Student`(sid, name, major, year, CAP)
  - $Student_1 = \sigma_{major="CS"}(Student)$
  - $Student_2 = \sigma_{major="Maths"}(Student)$
  - $Student_3 = \sigma_{(major \neq "CS") \wedge (major \neq "Maths")}(Student)$

Why Fragment?

- Support application's locality of access
  - **Example**:
    - Query q_1 at site A: $\sigma_{region='' Asia''}(Customers)$
    - Query q_2 at site B: $\sigma_{region='' Europe''}(Customers)$
- Scale out to manage large data / workload
- Improve performance with parallelized query execution
  - **Example**:
    - $R_1 = \sigma_{a<100}(R), \quad R_2 = \sigma_{a\geq 100}(R)$
    - $S_1 = \sigma_{a<100}(S), \quad S_2 = \sigma_{a\geq 100}(S)$
    - $R \bowtie_a S = (R_1 \bowtie_a S_1) \cup (R_2 \bowtie_a S_2)$

Fragmentation Strategies

- Horizontal fragmentation
- Vertical fragmentation
- Hybrid fragmentation

Desirable Properties of Fragmentation

- Consider a relation R being fragmented
- Completeness: Each item in R can also be found in one of its fragments
- Reconstruction: R can be reconstructed from its fragments
- Disjointness: Data items are not replicated (modulo reconstruction property)

Horizontal Fragmentation

- Partition a relation R into subsets $R_1, \cdots, R_n$

Student

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
2	Lucy	Maths	4	4.2
3	Marcie	Music	2	4.0
4	Sally	CS	3	3.8

$Student_1$

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
3	Marcie	Music	2	4.0

$\sigma_{year \leq 2}(Student)$

$Student_2$

sid	name	major	year	CAP
2	Lucy	Maths	4	4.2
4	Sally	CS	3	3.8

$\sigma_{year > 2}(Student)$

Horizontal Fragmentation (cont.)

- **Completeness**: $\forall t \in R,\ \exists\ R_i$ such that $t \in R_i$
- **Reconstruction**: $R = R_1 \cup \cdots \cup R_n$
- **Disjointness**: $\forall\ R_i, R_j\ (i \neq j \implies R_i \cap R_j = \emptyset)$

Student

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
2	Lucy	Maths	4	4.2
3	Marcie	Music	2	4.0
4	Sally	CS	3	3.8

$Student_1$

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
3	Marcie	Music	2	4.0

$\sigma_{year \leq 2}(Student)$

$Student_2$

sid	name	major	year	CAP
2	Lucy	Maths	4	4.2
4	Sally	CS	3	3.8

$\sigma_{year > 2}(Student)$

Horizontal Fragmentation (cont.)

- Fragmentation techniques:
  - Range partitioning
  - Hash partitioning
  - Derived horizontal fragmentation

Range Partitioning

- Partition R using predicates on some attributes of R
- **Example**:
  - Partition R based on values of attribute A
  - $R_1 = \sigma_{A<100}(R)$
  - $R_2 = \sigma_{A\in[100,500)}(R)$
  - $R_3 = \sigma_{A\geq 500}(R)$

Range Partitioning (cont.)

R

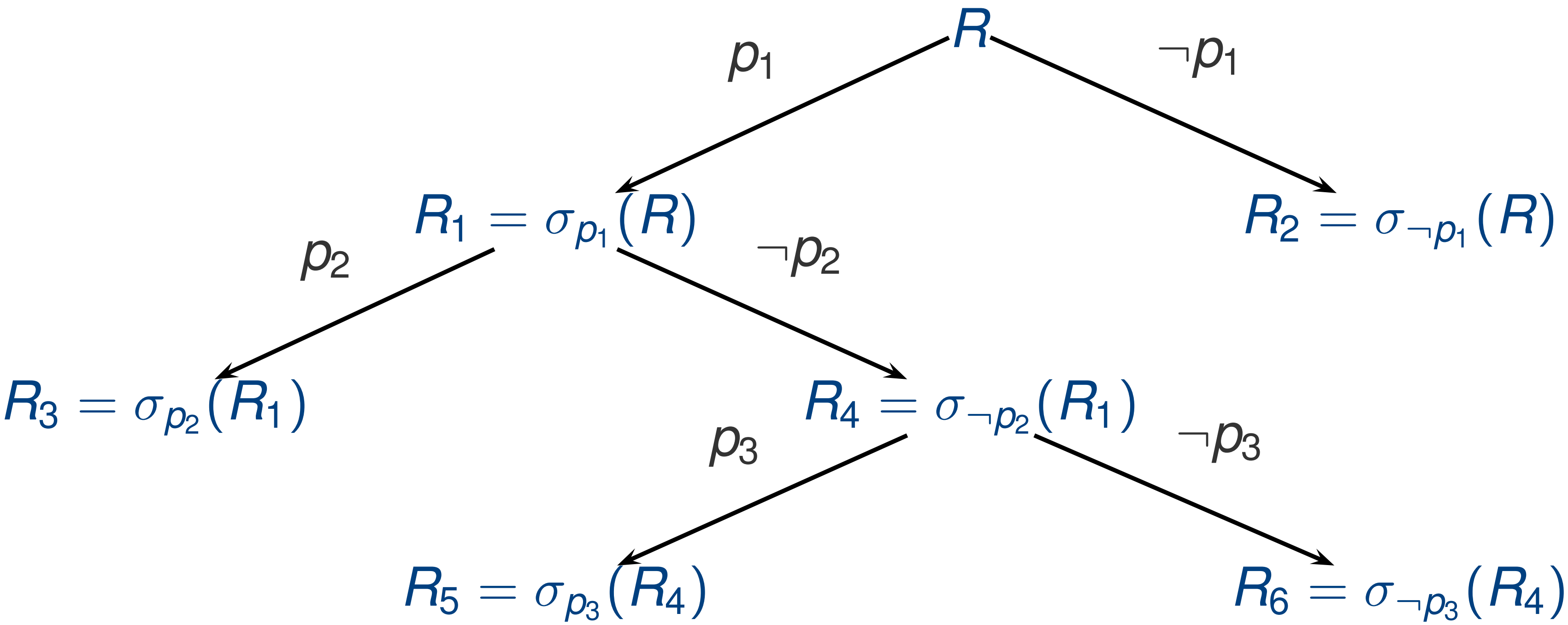
Range Partitioning (cont.)

$$\begin{tikzcd} & R \arrow[dl, "p_1"'] \arrow[dr, "\neg p_1"] & \\ R_1 = \sigma_{p_1}(R) & & R_2 = \sigma_{\neg p_1}(R) \end{tikzcd}$$

Range Partitioning (cont.)

$$
\begin{tikzcd}
 & & R \arrow[dl, "p_1"'] \arrow[dr, "\neg p_1"] & \\
 & R_1 = \sigma_{p_1}(R) \arrow[dl, "p_2"'] \arrow[dr, "\neg p_2"] & & R_2 = \sigma_{\neg p_1}(R) \\
R_3 = \sigma_{p_2}(R_1) & & R_4 = \sigma_{\neg p_2}(R_1) &
\end{tikzcd}
$$

Range Partitioning (cont.)



$$
\boxed{\begin{aligned}
&R = R_2 \ \cup\ R_3 \ \cup\ R_5 \ \cup\ R_6 \\
&R_2 = \sigma_{\neg p_1}(R) \\
&R_3 = \sigma_{p_1 \wedge p_2}(R) \\
&R_5 = \sigma_{p_1 \wedge \neg p_2 \wedge p_3}(R) \\
&R_6 = \sigma_{p_1 \wedge \neg p_2 \wedge \neg p_3}(R)
\end{aligned}}
$$

Hash Partitioning

- Partition R into $\{R_1, \cdots, R_n\}$ based on hash function on some attribute of R (say $R.A$)
- **Method 1**: Modulo Method
  - $R_{i+1} = \{t \in R \mid h(t.A) \mod n = i\}, i \in [0, n)$
- **Method 2**: Consistent Hashing
  - Partition the codomain of h using n values:

$$v_1 < v_2 < \cdots < v_n$$

  - $R_i = \{t \in R \mid h(t.A) \in (v_{i-1}, v_i]\}, i \in [2, n]$
  - $R_1 = R - (R_2 \cup \cdots \cup R_n)$

Modulo Method

Customers

cust#	cname	city
1	Alice	Singapore
2	Bob	Jarkata
3	Carol	Bangkok
4	Dave	Jarkata
5	Eve	Singapore
6	Fred	Penang
7	George	Hanoi
8	Hal	Bangkok
9	Ivy	Singapore
10	Joe	Penang
11	Kathy	Singapore
12	Larry	Jarkata

Customers_1

cust#	cname	city
3	Carol	Bangkok
6	Fred	Penang
9	Ivy	Singapore
12	Larry	Jarkata

Customers_2

cust#	cname	city
1	Alice	Singapore
4	Dave	Jarkata
7	George	Hanoi
10	Joe	Penang

Customers_3

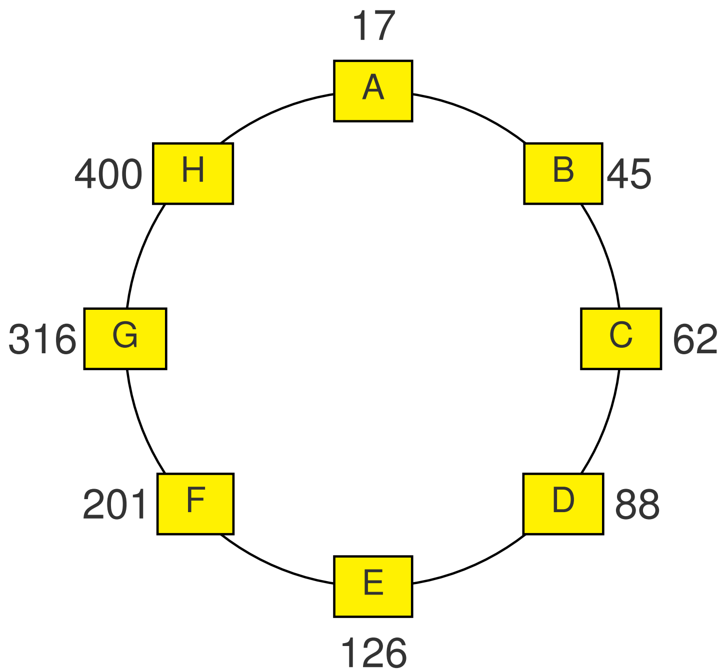
cust#	cname	city
2	Bob	Jarkata
5	Eve	Singapore
8	Hal	Bangkok
11	Kathy	Singapore

$$\text{Customer}_{i+1} = \{t \in \text{Customer} \mid h(t.cust\#) \mod 3 = i\},\ i \in [0, 3)$$

where h(v) = v

Consistent Hashing

- Consider a cluster of 8 nodes $\{A, B, C, D, E, F, G, H\}$
- Each node N is assigned a hashed value $h(N)$
- Nodes are logically organized on a ring based on the order of their hashed values
- A record t is stored in node N if $h(t.A)$ falls in N's region

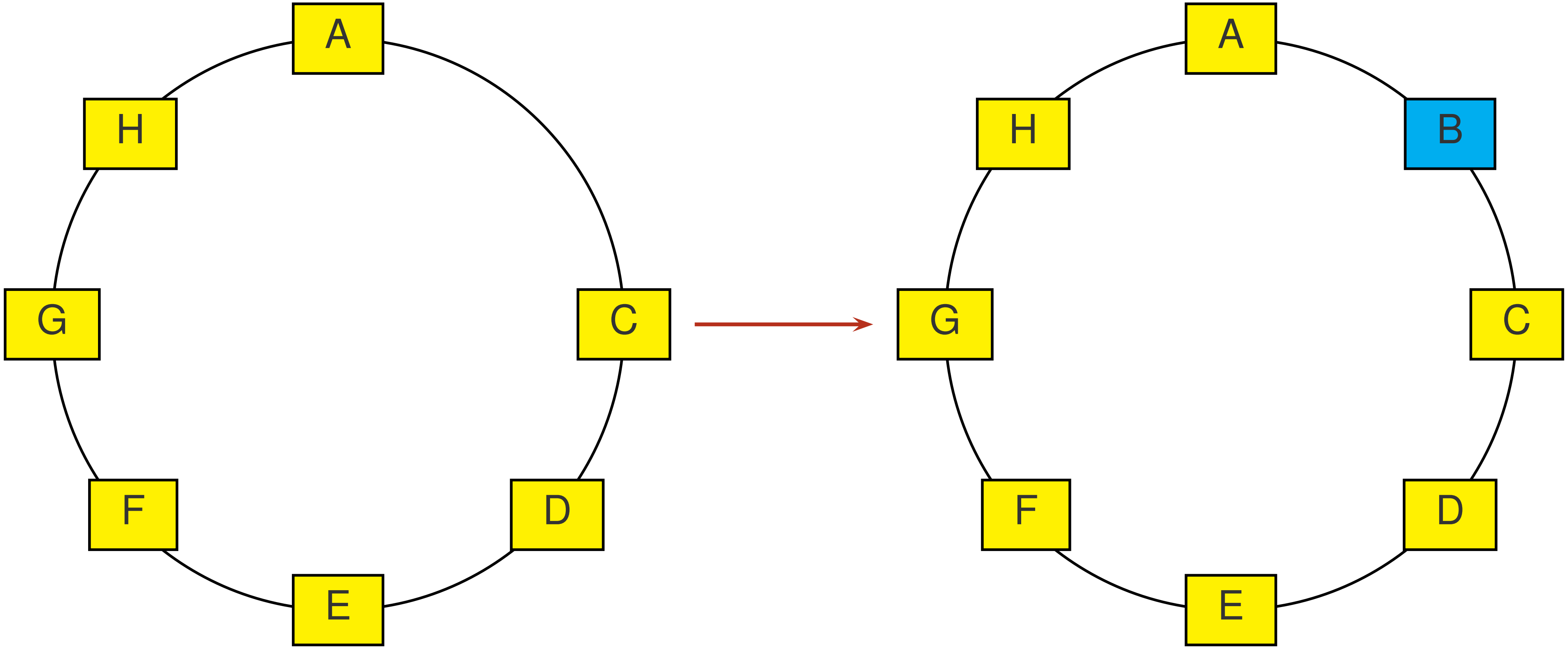


Node	Region
A	$> h(H)$ or $\leq h(A)$ (> 400 or ≤ 17)
B	$(h(A), h(B)] = (17, 45]$
C	$(h(B), h(C)] = (45, 62]$
⋮	⋮
G	$(h(F), h(G)] = (201, 316]$
H	$(h(G), h(H)] = (316, 400]$

$$h(A) < h(B) < h(C) < h(D) < h(E) < h(F) < h(G) < h(H)$$

Incremental Scaling with C. Hashing

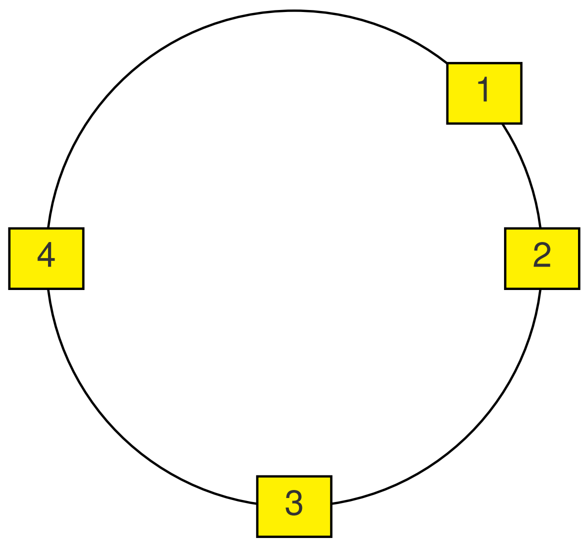
Adding a new node *B* ...



Challenges with Consistent Hashing

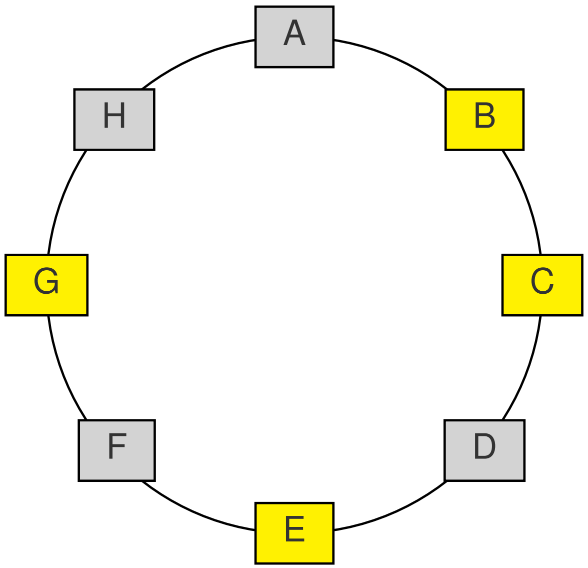
1. Non-uniform data & load distribution
2. Oblivious to heterogeneity in servers' performance

- **Example**: 4 servers



Consistent Hashing with Virtual Nodes

- **Example**: 4 physical nodes & 8 virtual nodes



- Virtual nodes: A, B, $\cdots$, G, H
- Physical nodes: 4 server nodes
  - Server 1 responsible for A & B
  - Server 2 responsible for C & D
  - Server 3 responsible for E & F
  - Server 4 responsible for G & H

Derived Horizontal Fragmentation

- Partitions a relation R based on the partitioning defined for a related relation S
- Let $\{S_1, \cdots, S_n\}$ be the partitioning of S
- If R & S are related by some attribute(s), R can be partitioned based on S's partitioning
- **Example**:
  - S = Customers(<u>cust#</u>,name,region) is partitioned into:
    - ★ $S_1 = \sigma_{region='' Asia''}(S)$
    - ★ $S_2 = \sigma_{region\neq'' Asia''}(S)$
  - R = Orders(<u>order#</u>,cust#,amount) is related to S via `cust#`
  - Partition R as follows: $R_i = R \ltimes_{cust\#} S_i$

Derived Horizontal Fragmentation

- Consider $R_i = R \ltimes_A S_i$, where each S_i is a partition of S
- For partitioning of R to be complete, **R**.**A** $\subseteq$ **S**.**A**

S_1

$\cdots$	A
$\cdots$	1
$\cdots$	2

S_2

$\cdots$	A
$\cdots$	3
$\cdots$	4

R

$\cdots$	A
$\cdots$	1
$\cdots$	2
$\cdots$	5
$\cdots$	4
$\cdots$	4
$\cdots$	1

R_1

$\cdots$	A
$\cdots$	1
$\cdots$	2
$\cdots$	1

R_2

$\cdots$	A
$\cdots$	4
$\cdots$	4

Derived Horizontal Fragmentation

- For partitioning of R to be disjoint, **S.A must be a key**

S_1

...	A
...	1
...	2

S_2

...	A
...	3
...	1
...	4

R

...	A
...	1
...	2
...	4
...	4
...	1

R_1

...	A
...	1
...	2
...	1

R_2

...	A
...	1
...	4
...	4
...	1

Derived Horizontal Fragmentation

- For partitioning of R to be complete, $\mathbf{R.A} \subseteq \mathbf{S.A}$
- For partitioning of R to be disjoint, **$\mathbf{S.A}$ must be a key**
- Therefore, for partitioning of R to be complete & disjoint, $R.A$ must be a foreign key of S with non-null values for $R.A$

Vertical Fragmentation

- Partition a relation R into $\{R_1, \cdots, R_n\}$ where
  - attributes(R_i) $\subset$ attributes(R), and
  - key(R) $\in$ attributes(R_i)

Student

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
2	Lucy	Maths	4	4.2
3	Marcie	Music	2	4.0
4	Sally	CS	3	3.8

$Student_1$

sid	name	major	year
1	Charlie	CS	2
2	Lucy	Maths	4
3	Marcie	Music	2
4	Sally	CS	3

$\pi_{sid,name,major,year}(Student)$

$Student_2$

sid	CAP
1	4.5
2	4.2
3	4.0
4	3.8

$\pi_{sid,CAP}(Student)$

Vertical Fragmentation (cont.)

- **Completeness**: $\forall A_i \in attributes(R)$, $\exists$ R_j such that $A_i \in attributes(R_j)$
- **Reconstruction**: $R = R_1 \bowtie \cdots \bowtie R_n$
- **Disjointness**: $\forall$ R_i, R_j $(i \neq j \implies attributes(R_i) \cap attributes(R_j) = \{key(R)\})$

Student

sid	name	major	year	CAP
1	Charlie	CS	2	4.5
2	Lucy	Maths	4	4.2
3	Marcie	Music	2	4.0
4	Sally	CS	3	3.8

$Student_1$

sid	name	major	year
1	Charlie	CS	2
2	Lucy	Maths	4
3	Marcie	Music	2
4	Sally	CS	3

$\pi_{sid, name, major, year}(Student)$

$Student_2$

sid	CAP
1	4.5
2	4.2
3	4.0
4	3.8

$\pi_{sid, CAP}(Student)$

Vertical Fragmentation: Heuristics

- Attribute affinity measure
  - **aff(A_i, A_j)**: measures how often A_i & A_j are referenced in queries

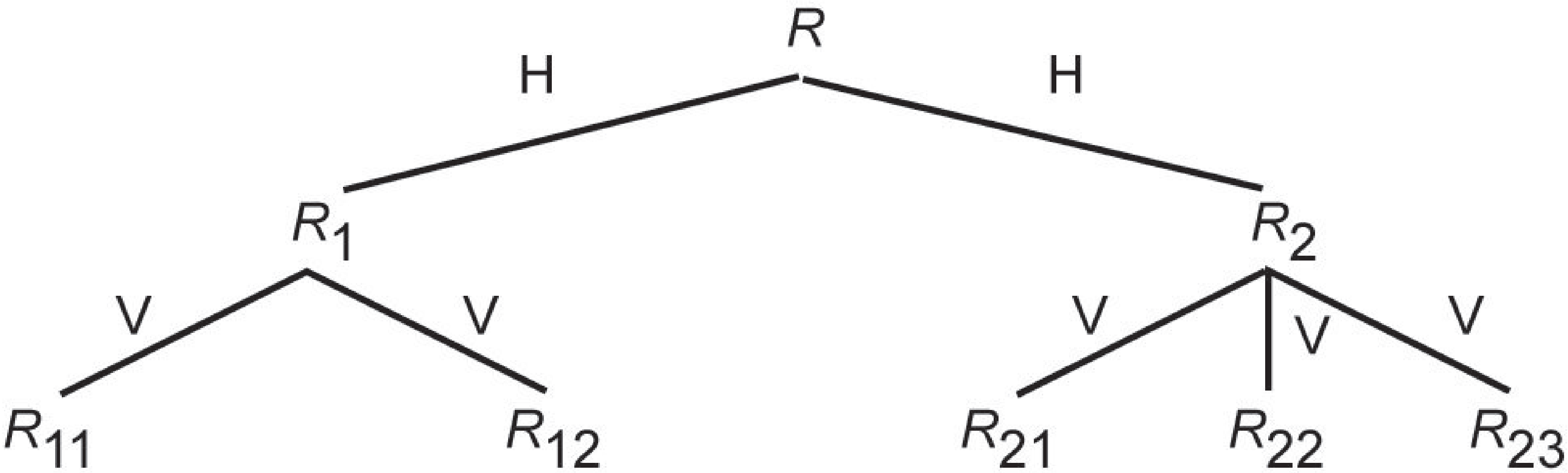
$$\begin{array}{c|cccc} & A_1 & A_2 & A_3 & A_4 \\ \hline A_1 & - & 0 & 45 & 0 \\ A_2 & 0 & - & 5 & 75 \\ A_3 & 45 & 5 & - & 3 \\ A_4 & 0 & 75 & 3 & - \end{array}$$

(Özsu & Valdureiz, 2011)

- Apply clustering algorithm on `aff(.,.)`

Hybrid Fragmentation

- Combinations of horizontal/vertical fragmentations
- **Example**:



(Özsu & Valdureiz, 2011)

Complete Partitioning wrt Query

- Let $F = \{R_1, \cdots, R_m\}$ be a partitioning of relation R
  - $R = R_1 \cup R_2 \cup \cdots \cup R_m$
- Let Q be a query on R
- F is a complete partitioning of R wrt Q if for every fragment $R_i \in F$, either every tuple in R_i matches Q or every tuple in R_i does not match Q

Complete Partitioning: Example

- `Student`(<u>sid</u>, name, major, year, CAP)
  - Domain(major) = $\{CS, Maths\}$
  - Domain(year) = $\{1, 2, 3, 4, 5\}$
- Let $F = \{S_1, S_2, S_3, S_4, S_5\}$ be a partitioning of Student
  - $S_1 = \sigma_{major=''Maths'' \wedge year=1}(Student)$
  - $S_2 = \sigma_{major=''Maths'' \wedge year=5}(Student)$
  - $S_3 = \sigma_{major=''Maths'' \wedge year>1 \wedge year<5}(Student)$
  - $S_4 = \sigma_{major=''CS'' \wedge year=1}(Student)$
  - $S_5 = \sigma_{major=''CS'' \wedge year>1}(Student)$
- Is F a complete partitioning wrt
  - $Q_1 = \sigma_{major=''CS''}(Student)$?
  - $Q_2 = \sigma_{year>1}(Student)$?
  - $Q_3 = \sigma_{(Major=''Maths'') \wedge (year<5)}(Student)$?
  - $Q_4 = \sigma_{year<3}(Student)$?

Minterm Predicates

- Let $P = \{p_1, \cdots, p_n\}$ be a set of selection predicates on a relation R
- A minterm predicate m for P is the conjunction of all the predicates in P of the form

$$m = p_1^* \wedge p_2^* \wedge \cdots \wedge p_n^*$$

where

$$p_i^* \text{ is either } p_i \text{ or } \neg p_i$$

- Let MTPred(P) denote the set of all minterm predicates for a set of predicates P

Minterm Predicates: Example

- `Student`(<u>sid</u>, name, major, year, CAP)
  - Domain(major) = $\{CS, Maths\}$
  - Domain(year) = $\{1, 2, 3, 4, 5\}$
- Query workload = $\{Q_1, Q_2, Q_3\}$, where $Q_i = \sigma_{p_i}(\text{Student})$
  - p_1 = (major = "CS")
  - p_2 = (year $>$ 1)
  - p_3 = (major = "Maths") $\wedge$ (year $<$ 5)

Minterm Predicates: Example (cont.)

- $P = \{p_1, p_2, p_3\}$
- MTPred(P) = $\{m_0, m_1, \cdots, m_6, m_7\}$
  - $m_0 = \neg p_1 \wedge \neg p_2 \wedge \neg p_3$
  - $m_1 = \neg p_1 \wedge \neg p_2 \wedge p_3$
  - $m_2 = \neg p_1 \wedge p_2 \wedge \neg p_3$
  - $m_3 = \neg p_1 \wedge p_2 \wedge p_3$
  - $m_4 = p_1 \wedge \neg p_2 \wedge \neg p_3$
  - $m_5 = p_1 \wedge \neg p_2 \wedge p_3$
  - $m_6 = p_1 \wedge p_2 \wedge \neg p_3$
  - $m_7 = p_1 \wedge p_2 \wedge p_3$

Review of Boolean Algebra

1. $p_1 \wedge p_2 \equiv p_2 \wedge p_1$
2. $p_1 \vee p_2 \equiv p_2 \vee p_1$
3. $p_1 \wedge (p_2 \wedge p_3) \equiv (p_1 \wedge p_2) \wedge p_3$
4. $p_1 \vee (p_2 \vee p_3) \equiv (p_1 \vee p_2) \vee p_3$
5. $p_1 \wedge (p_2 \vee p_3) \equiv (p_1 \wedge p_2) \vee (p_1 \wedge p_3)$
6. $p_1 \vee (p_2 \wedge p_3) \equiv (p_1 \vee p_2) \wedge (p_1 \vee p_3)$
7. $p \wedge \mathit{true} \equiv p$
8. $p \vee \mathit{false} \equiv p$
9. $p \wedge \mathit{false} \equiv \mathit{false}$
10. $p \vee \mathit{true} \equiv \mathit{true}$

10. $p \wedge p \equiv p$
11. $p \vee p \equiv p$
12. $p_1 \wedge (p_1 \vee p_2) \equiv p_1$
13. $p_1 \vee (p_1 \wedge p_2) \equiv p_1$
14. $p \wedge \neg p \equiv \mathit{false}$
15. $p \vee \neg p \equiv \mathit{true}$
16. $\neg(\neg p_1) \equiv p_1$
17. $\neg(p_1 \wedge p_2) \equiv \neg p_1 \vee \neg p_2$
18. $\neg(p_1 \vee p_2) \equiv \neg p_1 \wedge \neg p_2$

Minterm Predicates: Example (cont.)

p_1 = (major = "CS")
p_2 = (year $>$ 1)
p_3 = (major = "Maths") $\wedge$ (year $<$ 5)

m_0: major$\neq$"CS" $\wedge$ year$\not>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)
m_1: major$\neq$"CS" $\wedge$ year$\not>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)
m_2: major$\neq$"CS" $\wedge$ year$>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)
m_3: major$\neq$"CS" $\wedge$ year$>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)
m_4: major="CS" $\wedge$ year$\not>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)
m_5: major="CS" $\wedge$ year$\not>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)
m_6: major="CS" $\wedge$ year$>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)
m_7: major="CS" $\wedge$ year$>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)

Minterm Predicates: Example (cont.)

p_1 = (major = "CS")
p_2 = (year $>$ 1)
p_3 = (major = "Maths") $\wedge$ (year $<$ 5)

$\cancel{m_0:}$ $\cancel{\text{major}\neq\text{"CS"} \wedge \text{year}\not>1 \wedge (\text{major}\neq\text{"Maths"} \vee \text{year}\not<5)}$

m_1: $\cancel{\text{major}\neq\text{"CS"} \wedge \text{year}\not>1 \wedge (\text{major}=\text{"Maths"} \wedge \text{year}<5)}$

major="Maths" $\wedge$ **year**$\not>$**1** $\wedge$ **year**$<$**5**

m_2: $\cancel{\text{major}\neq\text{"CS"} \wedge \text{year}>1 \wedge (\text{major}\neq\text{"Maths"} \vee \text{year}\not<5)}$

major="Maths" $\wedge$ **year**$>$**1** $\wedge$ **year**$\not<$**5**

m_3: $\cancel{\text{major}\neq\text{"CS"} \wedge \text{year}>1 \wedge (\text{major}=\text{"Maths"} \wedge \text{year}<5)}$

major="Maths" $\wedge$ **year**$>$**1** $\wedge$ **year**$<$**5**

m_4: $\cancel{\text{major}=\text{"CS"} \wedge \text{year}\not>1 \wedge (\text{major}\neq\text{"Maths"} \vee \text{year}\not<5)}$

major="CS" $\wedge$ **year**$\not>$**1**

$\cancel{m_5:}$ $\cancel{\text{major}=\text{"CS"} \wedge \text{year}\not>1 \wedge (\text{major}=\text{"Maths"} \wedge \text{year}<5)}$

m_6: $\cancel{\text{major}=\text{"CS"} \wedge \text{year}>1 \wedge (\text{major}\neq\text{"Maths"} \vee \text{year}\not<5)}$

major="CS" $\wedge$ **year**$>$**1**

$\cancel{m_7:}$ $\cancel{\text{major}=\text{"CS"} \wedge \text{year}>1 \wedge (\text{major}=\text{"Maths"} \wedge \text{year}<5)}$

Minterm Predicates: Example (cont.)

p_1 = (major = "CS")
p_2 = (year $>$ 1)
p_3 = (major = "Maths") $\wedge$ (year $<$ 5)

~~m_0:~~ ~~major$\neq$"CS" $\wedge$ year$\not>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)~~

m_1: ~~major$\neq$"CS" $\wedge$ year$\not>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)~~
major$=$"Maths" $\wedge$ ~~year$\not>$1 $\wedge$ year$<$5~~ **year=1**

m_2: ~~major$\neq$"CS" $\wedge$ year$>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)~~
major$=$"Maths" $\wedge$ ~~year$>$1 $\wedge$ year$\not<$5~~ **year$=$5**

m_3: ~~major$\neq$"CS" $\wedge$ year$>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)~~
major$=$"Maths" $\wedge$ **year$>$1** $\wedge$ **year$<$5**

m_4: ~~major="CS" $\wedge$ year$\not>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)~~
major="CS" $\wedge$ ~~year$\not>$1~~ **year$=$1**

~~m_5:~~ ~~major="CS" $\wedge$ year$\not>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)~~

m_6: ~~major="CS" $\wedge$ year$>$1 $\wedge$ (major$\neq$"Maths" $\vee$ year$\not<$5)~~
major="CS" $\wedge$ **year$>$1**

~~m_7:~~ ~~major="CS" $\wedge$ year$>$1 $\wedge$ (major$=$"Maths" $\wedge$ year$<$5)~~

Minterm Predicates: Example (cont.)

p_1 = (major = "CS")
p_2 = (year $>$ 1)
p_3 = (major = "Maths") $\wedge$ (year $<$ 5)

After simplification, MTPred(P) = $\{m_1, m_2, m_3, m_4, m_6\}$

m_1: major$=$"Maths" $\wedge$ year=1
m_2: major$=$"Maths" $\wedge$ year$=$5
m_3: major$=$"Maths" $\wedge$ year$>$1 $\wedge$ year$<$5
m_4: major="CS" $\wedge$ year$=$1
m_6: major="CS" $\wedge$ year$>$1

Minterm Predicate Partitioning

- `Student`(<u>sid</u>, name, major, year, CAP)
  - Domain(major) = $\{CS, Maths\}$
  - Domain(year) = $\{1, 2, 3, 4, 5\}$
- Query workload $Q = \{Q_1, Q_2, Q_3\}$, where $Q_i = \sigma_{p_i}$(Student)
  - p_1 = (major = "CS")
  - p_2 = (year > 1)
  - p_3 = (major = "Maths") $\wedge$ (year < 5)
- $P = \{p_1, p_2, p_3\}$, MTPred(P) = $\{m_1, m_2, m_3, m_4, m_6\}$
- Student = $S_1 \cup S_2 \cup S_3 \cup S_4 \cup S_6$, where $S_i = \sigma_{m_i}$(Student)
- $\{S_1, S_2, S_3, S_4, S_6\}$ is the minterm predicate partitioning of Student wrt Q

Property of Minterm Predicate Partitioning

- Let $Q = \{Q_1, \cdots, Q_k\}$ be a set of queries on relation R, where each $Q_i = \sigma_{p_i}(R)$
- Let $P = \{p_1, \cdots, p_k\}$
- Let $F = \{R_1, \cdots, R_m\}$ be the minterm partitioning of R based on MTPred(P)
- **Theorem**: F is a complete partitioning wrt every query in Q

References

- T. Özsu & P. Valdureiz, *Distributed Database Design*, Chapter 2, Principles of Distributed Database Systems, 4^{th} Edition, 2020
- G. DeCandia, et al., *Dynamo: Amazon's highly available key-value store*, SOSP 2007.

  http://www.allthingsdistributed.com/2007/10/amazons_dynamo.html