

CS 347:
Distributed Databases and
Transaction Processing
Notes02: Distributed DB Design

Hector Garcia-Molina
Zoltan Gyongyi

Distributed DB Design

Chapter 5
Ozsu & Valduriez

Top-down approach: - have DB...
- how to split and
allocate the sites

Multi-DBs (or bottom-up): - no design
issues

Two issues in DDB design:

- Fragmentation
- Allocation

Note: issues not independent,
but will cover separately

Example

Employee relation E (#, name, loc, sal, ...)

40% of queries:

Q_A : select *
from E
where loc= S_A
and ...

40% of queries:

Q_B : select *
from E
where loc= S_B
and ...

Example

Employee relation E (#, name, loc, sal, ...)

40% of queries:

Q_A : select *
from E
where loc= S_A
and ...

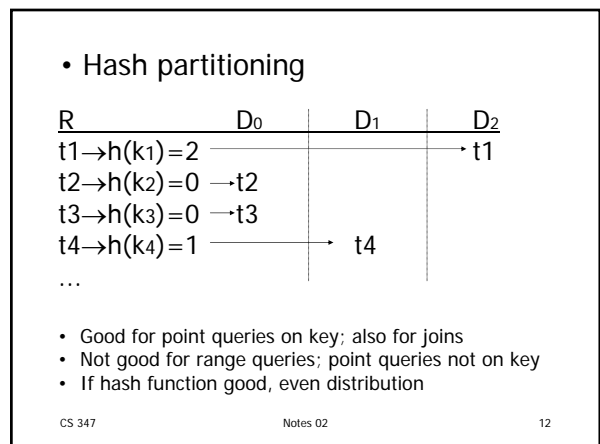
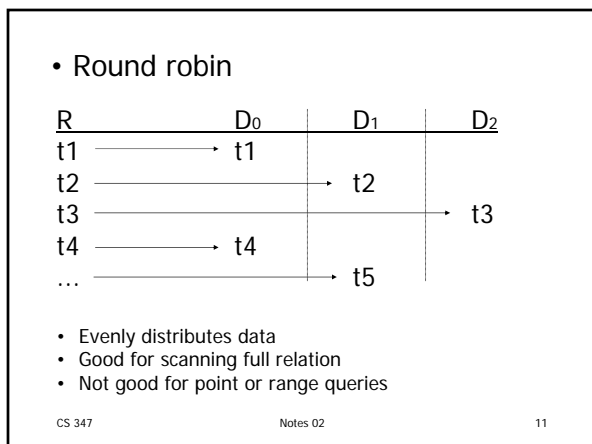
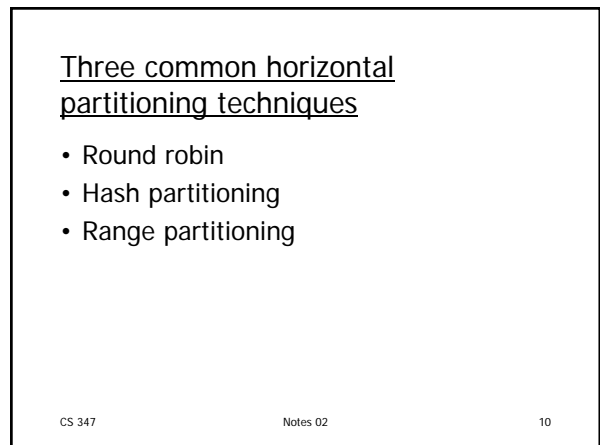
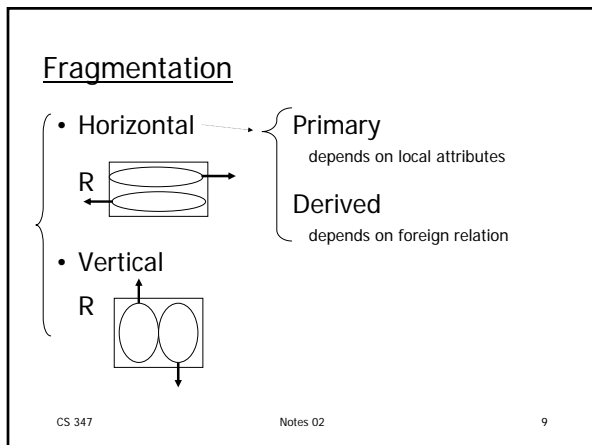
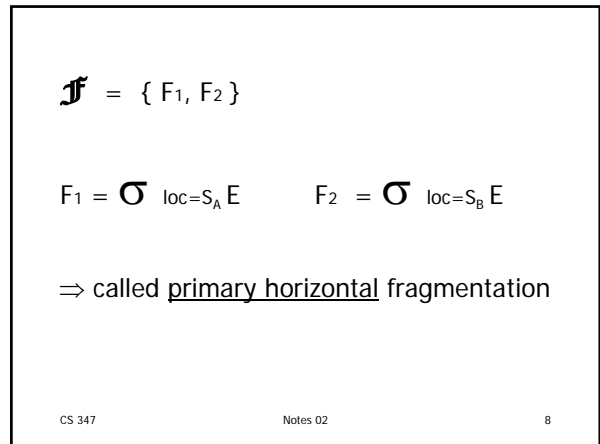
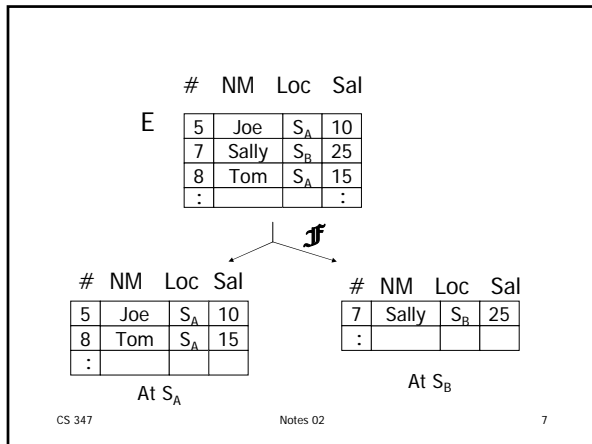
40% of queries:

Q_B : select *
from E
where loc= S_B
and ...

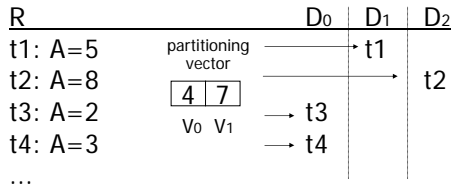
Motivation: Two sites: S_A, S_B

$Q_A \rightarrow S_A$ $S_B \leftarrow Q_B$

- It does not take a rocket scientist to figure out fragmentation...



• Range partitioning



- Good for some range queries on A
- Need to select good vector: else unbalance
 - data skew
 - execution skew

Which are good fragmentations?

Example:

$$\mathcal{F} = \{ F_1, F_2 \}$$

$$F_1 = \sigma_{sal < 10} E \quad F_2 = \sigma_{sal > 20} E$$

Which are good fragmentations?

Example:

$$\mathcal{F} = \{ F_1, F_2 \}$$

$$F_1 = \sigma_{sal < 10} E \quad F_2 = \sigma_{sal > 20} E$$

☒ Problem: Some tuples lost!

Which are good fragmentations?

Second example:

$$\mathcal{F} = \{ F_3, F_4 \}$$

$$F_3 = \sigma_{sal < 10} E \quad F_4 = \sigma_{sal > 5} E$$

☒ Tuples with $5 < sal < 10$ are duplicated...

Prefer to deal with replication explicitly

Example: $\mathcal{F} = \{ F_5, F_6, F_7 \}$

$$F_5 = \sigma_{sal \leq 5} E \quad F_6 = \sigma_{5 < sal < 10} E$$

$$F_7 = \sigma_{sal \geq 10} E$$

- Then replicate F_6 if convenient (part of allocation problem)

Desired properties for horizontal fragmentation

$$R \Rightarrow \mathcal{F} = \{ F_1, F_2, \dots \}$$

(1) Completeness

$$\forall t \in R, \exists F_i \in \mathcal{F} \text{ such that } t \in F_i$$

(2) Disjointness

$\forall t \in F_i, \neg \exists F_j$ such that
 $t \in F_j, i \neq j, F_i, F_j \in \mathcal{F}$

How do we get completeness and disjointness?

(1) Check it "manually"!

e.g., $F_1 = \sigma_{sal < 10} E$; $F_2 = \sigma_{sal \geq 10} E$

How do we get completeness and disjointness?

(2) "Automatically" generate fragments with these properties

Desired simple predicates \Rightarrow Fragments

Example of generation

- Say queries use predicates:
 $A < 10, A > 5, Loc = S_A, Loc = S_B$
- Next: - generate "minterm" predicates
- eliminate useless ones

Minterm predicates (part I)

- (1) $A < 10 \wedge A > 5 \wedge Loc = S_A \wedge Loc = S_B$
- (2) $A < 10 \wedge A > 5 \wedge Loc = S_A \wedge \neg (Loc = S_B)$
- (3) $A < 10 \wedge A > 5 \wedge \neg (Loc = S_A) \wedge Loc = S_B$
- (4) $A < 10 \wedge A > 5 \wedge \neg (Loc = S_A) \wedge \neg (Loc = S_B)$
- (5) $A < 10 \wedge \neg (A > 5) \wedge Loc = S_A \wedge Loc = S_B$
- (6) $A < 10 \wedge \neg (A > 5) \wedge Loc = S_A \wedge \neg (Loc = S_B)$
- (7) $A < 10 \wedge \neg (A > 5) \wedge \neg (Loc = S_A) \wedge Loc = S_B$
- (8) $A < 10 \wedge \neg (A > 5) \wedge \neg (Loc = S_A) \wedge \neg (Loc = S_B)$

Minterm predicates (part I)

- ~~(1) $A < 10 \wedge A > 5 \wedge Loc = S_A \wedge Loc = S_B$~~
- ~~(2) $A < 10 \wedge A > 5 \wedge Loc = S_A \wedge \neg (Loc = S_B)$~~
- ~~(3) $A < 10 \wedge A > 5 \wedge \neg (Loc = S_A) \wedge Loc = S_B$~~
- ~~(4) $A < 10 \wedge A > 5 \wedge \neg (Loc = S_A) \wedge \neg (Loc = S_B)$~~
- ~~(5) $A < 10 \wedge \neg (A > 5) \wedge Loc = S_A \wedge Loc = S_B$~~
- ~~(6) $A < 10 \wedge \neg (A > 5) \wedge Loc = S_A \wedge \neg (Loc = S_B)$~~
- ~~(7) $A < 10 \wedge \neg (A > 5) \wedge \neg (Loc = S_A) \wedge Loc = S_B$~~
- ~~(8) $A < 10 \wedge \neg (A > 5) \wedge \neg (Loc = S_A) \wedge \neg (Loc = S_B)$~~

$5 < A < 10$

$A \leq 5$

Minterm predicates (part II)

- (9) $\neg(A < 10) \wedge A > 5 \wedge \text{LOC} = S_A \wedge \text{LOC} = S_B$
- (10) $\neg(A < 10) \wedge A > 5 \wedge \text{LOC} = S_A \wedge \neg(\text{LOC} = S_B)$
- (11) $\neg(A < 10) \wedge A > 5 \wedge \neg(\text{LOC} = S_A) \wedge \text{LOC} = S_B$
- (12) $\neg(A < 10) \wedge A > 5 \wedge \neg(\text{LOC} = S_A) \wedge \neg(\text{LOC} = S_B)$
- (13) $\neg(A < 10) \wedge \neg(A > 5) \wedge \text{LOC} = S_A \wedge \text{LOC} = S_B$
- (14) $\neg(A < 10) \wedge \neg(A > 5) \wedge \text{LOC} = S_A \wedge \neg(\text{LOC} = S_B)$
- (15) $\neg(A < 10) \wedge \neg(A > 5) \wedge \neg(\text{LOC} = S_A) \wedge \text{LOC} = S_B$
- (16) $\neg(A < 10) \wedge \neg(A > 5) \wedge \neg(\text{LOC} = S_A) \wedge \neg(\text{LOC} = S_B)$

CS 347

Notes 02

25

Minterm predicates (part II)

- ~~(9) $\neg(A < 10) \wedge A > 5 \wedge \text{LOC} = S_A \wedge \text{LOC} = S_B$~~
- ~~(10) $\neg(A < 10) \wedge A > 5 \wedge \text{LOC} = S_A \wedge \neg(\text{LOC} = S_B)$~~
- ~~(11) $\neg(A < 10) \wedge A > 5 \wedge \neg(\text{LOC} = S_A) \wedge \text{LOC} = S_B$~~
- ~~(12) $\neg(A < 10) \wedge A > 5 \wedge \neg(\text{LOC} = S_A) \wedge \neg(\text{LOC} = S_B)$~~
- ~~(13) $\neg(A < 10) \wedge \neg(A > 5) \wedge \text{LOC} = S_A \wedge \text{LOC} = S_B$~~
- ~~(14) $\neg(A < 10) \wedge \neg(A > 5) \wedge \text{LOC} = S_A \wedge \neg(\text{LOC} = S_B)$~~
- ~~(15) $\neg(A < 10) \wedge \neg(A > 5) \wedge \neg(\text{LOC} = S_A) \wedge \text{LOC} = S_B$~~
- ~~(16) $\neg(A < 10) \wedge \neg(A > 5) \wedge \neg(\text{LOC} = S_A) \wedge \neg(\text{LOC} = S_B)$~~

$A \geq 10$

CS 347

Notes 02

26

Final fragments:

- F2: $5 < A < 10 \wedge \text{LOC} = S_A$
- F3: $5 < A < 10 \wedge \text{LOC} = S_B$
- F6: $A \leq 5 \wedge \text{LOC} = S_A$
- F7: $A \leq 5 \wedge \text{LOC} = S_B$
- F10: $A \geq 10 \wedge \text{LOC} = S_A$
- F11: $A \geq 10 \wedge \text{LOC} = S_B$

CS 347

Notes 02

27

Note: elimination of useless fragments depends on application semantics:

e.g.: if LOC could be $\neq S_A, \neq S_B$, we need to add fragments

- F4: $5 < A < 10 \wedge \text{LOC} \neq S_A \wedge \text{LOC} \neq S_B$
- F8: $A \leq 5 \wedge \text{LOC} \neq S_A \wedge \text{LOC} \neq S_B$
- F12: $A \geq 10 \wedge \text{LOC} \neq S_A \wedge \text{LOC} \neq S_B$

CS 347

Notes 02

28

Why does this work?

Predicates: $p_1 \wedge p_2 \wedge p_3 \wedge p_4$
 $p_1 \wedge p_2 \wedge p_3 \wedge \neg p_4$
 \vdots
 $\neg p_1 \wedge \neg p_2 \wedge \neg p_3 \wedge \neg p_4$

CS 347

Notes 02

29

(1) Completeness:

Take $t \in R$, $p_i(t)$ must be T or F!

Say $p_1(t) = T$, $p_2(t) = T$, $p_3(t) = F$, $p_4(t) = F$

Then t is in fragment with predicate

$p_1 \wedge p_2 \wedge \neg p_3 \wedge \neg p_4$

CS 347

Notes 02

30

(2) Disjointness

Say $t \in$ fragment $p_1 \wedge p_2 \wedge \neg p_3 \wedge \neg p_4$

Then

$$p_1(t) = T, p_2(t) = T, p_3(t) = F, p_4(t) = F$$

$\Rightarrow t$ cannot be in any other fragment!

CS 347

Notes 02

31

Summary

- Given simple predicates $Pr = \{ p_1, p_2, \dots, p_m \}$ minterm predicates are

$$M = \{ m \mid m = \bigwedge_{p_k \in Pr} p_k^*, 1 \leq k \leq m \}$$

where p_k^* is p_k or is $\neg p_k$

Fragments $\sigma_m R$ for all $m \in M$ are complete and disjoint

CS 347

Notes 02

32

Which simple predicates should we use in Pr?

\Leftrightarrow Desired property of Pr:

- minimality
- completeness

different from
COMPLETENESS of
fragmentation!

CS 347

Notes 02

33

Informal definition

Set of predicates Pr is complete

if for every $F_i \in \mathcal{F}[Pr]$,

every $t \in F_i$ has equal probability of access by every major application.

Note: $\mathcal{F}[Pr]$ is fragmentation defined by minterm predicates generated by Pr

CS 347

Notes 02

34

Informal definition

Set of predicates Pr is minimal if no $Pr' \subset Pr$ is complete

CS 347

Notes 02

35

Summary: horizontal fragmentation

- Type: primary, derived
- Properties: completeness, disjointness
- Predicates: minimal, complete

CS 347

Notes 02

36

Vertical fragmentation

Example:

#	NM	Loc	Sal
5	Joe	Sa	10
7	Sally	Sb	25
8	Fred	Sa	15
...			

#	NM	Loc
5	Joe	Sa
7	Sally	Sb
8	Fred	Sa
...		

#	Sal
5	10
7	25
8	15
...	

CS 347

Notes 02

37

$$R[T] \Rightarrow R_1[T_1] \quad T_i \subseteq T$$

$$\vdots$$

$$R_n[T_n]$$

☒ Just like normalization of relations

CS 347

Notes 02

38

Properties: $R[T] \Rightarrow R_i[T_i]$

(1) Completeness

$$\bigcup_{\text{all } i} T_i = T$$

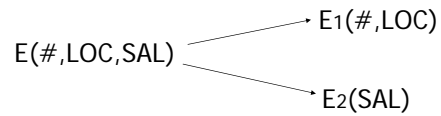
CS 347

Notes 02

39

(2) Disjointness

$$T_i \cap T_j = \emptyset \text{ for all } i, j \text{ } i \neq j$$



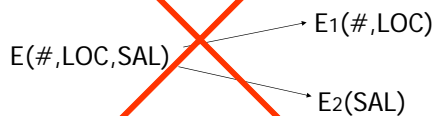
CS 347

Notes 02

40

(2) Disjointness

~~$$T_i \cap T_j = \emptyset \text{ for all } i, j \text{ } i \neq j$$~~



Not a desirable property!
(could not reconstruct R)

CS 347

Notes 02

41

(3) Lossless join

$$\bowtie_{\text{all } i} R_i = R$$

☛ One way to achieve lossless join:
repeat key in all fragments, i.e.,
 $\text{key} \subseteq T_i$ for all i

CS 347

Notes 02

42

⇔ How do we decide what attributes are grouped with which?

Example:

$E(\#,NM,LOC,SAL)$

$\left\{ \begin{array}{l} E1(\#,NM,LOC) \\ E2(\#,SAL) \end{array} \right.$

$\left\{ \begin{array}{l} E1(\#,NM) \\ E2(\#,LOC) \\ E3(\#,SAL) \end{array} \right.$

?

CS 347

Notes 02

43

Attribute affinity matrix

	A1	A2	A3	A4	A5
A1	-	-	-	-	-
A2	50	-	-	-	-
A3	45	48	-	-	-
A4	1	2	0	-	-
A5	0	0	4	75	-

CS 347

Notes 02

44

Attribute affinity matrix

	A1	A2	A3	A4	A5
A1	-	-	-	-	-
A2	50	-	-	-	-
A3	45	48	-	-	-
A4	1	2	0	-	-
A5	0	0	4	75	-

$R1[K,A1,A2,A3]$

$R2[K,A4,A5]$

CS 347

Notes 02

45

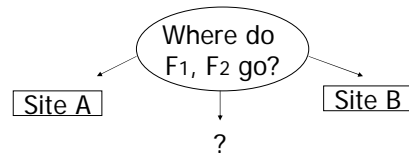
Allocation

Example: $E(\#,NM,LOC,SAL) \Rightarrow$

$$F_1 = \sigma_{loc=S_A} E ; F_2 = \sigma_{loc=S_B} E$$

Q_A : select ... where $loc=S_A$...

Q_B : select ... where $loc=S_B$...



CS 347

Notes 02

46

Issues

- Where do queries originate
- What is communication cost? and size of answers, relations, ...
- What is storage capacity, cost at sites? and size of fragments?
- What is processing power at sites?

CS 347

Notes 02

47

More issues

- What is query processing strategy?
 - How are joins done?
 - Where are answers collected?

CS 347

Notes 02

48

Do we replicate fragments?

- Cost of updating copies?
- Writes and concurrency control?
- ...

CS 347

Notes 02

49

Optimization problem:

- What is best placement of fragments and/or best number of copies to:
 - minimize query response time
 - maximize throughput
 - minimize “some cost”
 - ...
- Subject to constraints?
 - Available storage
 - Available bandwidth, power, ...
 - Keep 90% of response time below X
 - ...

Extremely hard problem

CS 347

Notes 02

50

Example: Single fragment F

$$\text{Read cost: } \sum_{i=1}^m [t_i \times \text{MIN}_j C_{ij}]$$

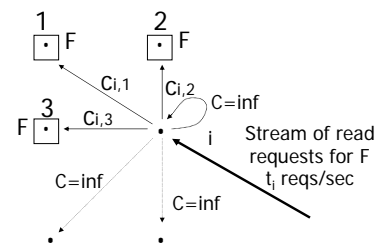
- i: Originating site of request (1...m)
- t_i: Read traffic at S_i
- C_{ij}: Retrieval cost
Accessing fragment F at S_j from S_i

CS 347

Notes 02

51

Scenario - Read cost



CS 347

Notes 02

52

Write cost

$$\sum_{i=1}^m \sum_{j=1}^m X_{ij} u_i C'_{ij}$$

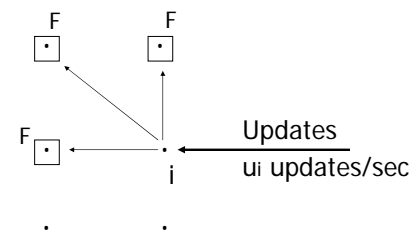
- i: Originating site of request
- j: Site being updated
- X_{ij}: $\begin{cases} 0 & \text{if F not stored at S}_j \\ 1 & \text{if F stored at S}_j \end{cases}$
- u_i: Write traffic at S_i
- C'_{ij}: Write cost
Updating F at S_j from S_i

CS 347

Notes 02

53

Scenario - Write cost



CS 347

Notes 02

54

Storage cost:

$$\sum_{i=1}^m X_i d_i$$

X_i : $\begin{cases} 0 & \text{if F not stored at } S_i \\ 1 & \text{if F stored at } S_i \end{cases}$
 d_i : storage cost at S_i

CS 347

Notes 02

55

Target function:

$$\min \left\{ \sum_{i=1}^m [t_i \times \text{MIN}_j C_{ij}] + \sum_{j=1}^m X_j \times u_j \times C'_{ij} \right. \\ \left. + \sum_{i=1}^m X_i \times d_i \right\}$$

CS 347

Notes 02

56

Can add more complications:

Examples:

- Multiple fragments
- Fragment sizes
- Concurrency control cost

CS 347

Notes 02

57

Summary

- Description of fragmentation
- Good fragmentations
- Design of fragmentation
- Allocation

CS 347

Notes 02

58