Distributed Database System (CS-600)

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

(Week 16) Lecture 31-32

Objectives: Learning objectives of this lecture are

- **Fragmentation Rules Examples**
- **Horizontal Fragmentation**
- **Vertical Fragmentation**

Distributed Database System (CS-600)

Mr. Nauman Iqbal **email id: nauman@biit.edu.pk Whatsapp# 0321-5821151**
Mr. Ahsan **email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656**

Overview:

- In current lecture, we will discuss examples of simple fragmentation (on single column).
- We will discuss how to perform horizontal fragmentation on multiple columns.
- We will discuss basics of vertical fragmentation and affinity matrix.

Recap:

In Previous lecture, we had discussed following points.

- Promises of distributed database system, that must be available at implementation phase.
- What are the design issues occurring at the time of designing a distributed database system?
- What is fragmentation?
- Types of fragmentation?
- Correctness rules for fragmentation.

Horizontal Fragmentation Examples:

- Let us consider given database table and create fragments so that none of the correctness rules violates.

Employee

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**	**eDesignation**
E101	Ahmad	23	45000	ISB	Developer
E102	Hina	26	36000	ISB	Developer
E103	Hamza	45	95000	RWP	Manager
E104	Sana	19	18000	LHR	Intern
E105	Suleman	30	15000	LHR	Labour
E106	Shahid	34	25000	LHR	Developer

Example 1:

- Let us consider above table has a constraint applied over eSalary that it must be more than 12000 and less than 99000.
- Let us create three horizontal fragments of Employee table based on eSalary column.

Fragment Name	**Criteria**
F1	eSalary > 12000 And eSalary < 25000
F2	eSalary > 25000 And eSalary < 50000

F3	eSalary >= 50000

- Now considering above three fragments following correctness rules violated.

i. Completeness Violated

Because any employee whose eSalary will be equal to 25000, will not belong to any of the Fragments created. But Completeness rule stated that each record must belong to some of the fragment.

ii. Reconstruction Violated

Because any employee whose salary is 99000 or more than that will belong to Fragment 'F3' but Reconstruction rule stated that no fragment would accept incorrect record. As 99000 or more salary is not allowed in table but fragment is accepting such values.

iii. Disjointness Not Violated

Because no record will belong to more than one fragments.

Correction in Fragments:

Fragment Name	**Criteria**
F1	eSalary > 12000 And eSalary < 25000
F2	eSalary >= 25000 And eSalary < 50000
F3	eSalary >= 50000 And eSalary <99000

- No correctness rule violates on above fragments.

Example 2:

Employee

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**	**eDesignation**
E101	Ahmad	23	45000	ISB	Developer
E102	Hina	26	36000	ISB	Developer
E103	Hamza	45	95000	RWP	Manager
E104	Sana	19	18000	LHR	Intern
E105	Suleman	30	15000	LHR	Labour
E106	Shahid	34	25000	LHR	Developer

- Let us consider that above table has a constraint applied over eDesignation. Only "Developer, Manager, Intern, HR, Labor and QA" are allowed in eDesignation. Any other value will be rejected.

- Let us create four horizontal fragments of above Employee table.

Fragment Name	**Criteria**
F1	eDesignation = ‘Manager’
F2	eDesignation = ‘Developer’
F3	eDesignation = ‘Labor’
F4	eDesignation != ‘Manager’

- Now considering above three fragments following correctness rules violated.

i. Completeness Not Violated

Because all type of eDesignation are acceptable in above fragments.

ii. Reconstruction Violated

Because incorrect values also accepted in Fragment F4. If an employee is inserted whose eDesignation is ‘Prime Minister’ which is incorrect according to the constraints mentioned in table description. But eDesignation ‘Prime Minister’ will be accepted by fragment F4 as it accepts all values other than ‘Manager’.

iii. Disjointness Violated

Because if an employee is inserted whose eDesignation is ‘Developer’, this particular employee will be accepted in fragment F2 and F4. Which according to Disjointness rule is incorrect. A record must only belong to only one fragment.

Correction in Fragments:

Fragment Name	**Criteria**
F1	eDesignation = ‘Manager’
F2	eDesignation = ‘Developer’
F3	eDesignation = ‘Labor’
F4	eDesignation in (‘Intern’, ‘HR’, ‘QA’)

Example 3:

Employee

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**	**eDesignation**
E101	Ahmad	23	45000	ISB	Developer
E102	Hina	26	36000	ISB	Developer
E103	Hamza	45	95000	RWP	Manager
E104	Sana	19	18000	LHR	Intern
E105	Suleman	30	15000	LHR	Labour
E106	Shahid	34	25000	LHR	Developer

- Let us now create fragments of above table based on eAge and eCity.

Fragment Name	**Criteria**
F1	eCity = ‘ISB’
F2	eCity != ‘ISB’
F3	eAge >= 30
F4	eAge < 30

- Now checking which record belongs to which fragment.

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**	**eDesignation**	**Fragment**
E101	Ahmad	23	45000	ISB	Developer	F1, F4
E102	Hina	26	36000	ISB	Developer	F1, F4
E103	Hamza	45	95000	RWP	Manager	F2, F3
E104	Sana	19	18000	LHR	Intern	F2, F4
E105	Suleman	30	15000	LHR	Labor	F2, F4
E106	Shahid	34	25000	LHR	Developer	F2,F3

- As we have seen above, each record belong to more than one fragment, which means Disjointness property violated.

- This is because fragments contains more than one column.

- Whenever fragments are created over more than one column or over multiple criteria then following method is used which is briefly described below.

Types of Horizontal Fragmentation:

1- **Primary Horizontal Fragmentation**

2- **Derived Horizontal Fragmentation**

- Primary Horizontal Fragmentation

- Primary horizontal fragmentation of a relation is performed using predicates that are defined on that relation.

Example:

Employee

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E101	Ahmad	23	45000	ISB
E102	Hina	26	36000	ISB
E103	Hamza	45	95000	RWP

E104	Sana	19	18000	LHR
E105	Suleman	29	15000	LHR
E106	Shahid	34	25000	ISB

- Let us consider that we have four locations where our distributed database is located.
- Each site has an application, which is retrieving data.

- First Application is retrieving all those employees whose city is ‘ISB’.
- Second Application is retrieving all those employees whose city is other than ‘ISB’.
- Third Application is retrieving all those employees whose age is above 30.
- Fourth Application is retrieving all those employees whose age is equal or less than 30.

Step 1:

- Based on application or criteria given, we will list down all predicates (where clause).

P1 → eCity = ‘ISB’

P2 → eCity != ‘ISB’

P3 → eAge > 30

P4 → eAge <=30

- As we have already seen above, if we create P1, P2, P3, P4 as fragments, Disjointness rule violates.
- This is why, we have to create combination of two predicates (because two columns are involved) called “Minterms”.
- Each “Minterm” (combination of predicate) will be examined if it is logically true or not.
- Finally, only correct “Minterms” will be selected as fragments.

Minterms:

Minterm	Predicate	Records	Remarks
m1	P1 ^ P2	0	No record satisfies, so not true
m2	P1 ^ P3	1 [E106]	Logically True
m3	P1 ^ P4	2 [E101, E102]	Logically True
m4	P2 ^ P1	0	Same as P1^P2 which was denied earlier
m5	P2 ^ P3	1[E103]	Logically True
m6	P2 ^ P4	2 [E104, E105]	Logically True

m7	P3 ^ P1	------	Same as P1^P3, no need to repeat
m8	P3 ^ P2	------	Same as P2^P3, no need to repeat
Now, As all our records of table accommodated in any of Minterm, so no need to create more minterms. We will keep on creating different combinations of predicates unless all of our records adjusts.			

Fragments will be,

F1 → m2 = (P1 ^ P3) → eCity = ‘ISB’ and eAge > 30

EmpNo	eName	eAge	eSalary	eCity
E106	Shahid	34	25000	ISB

F2 → m3 = (P1 ^ P4) → eCity = ‘ISB’ and eAge <= 30

EmpNo	eName	eAge	eSalary	eCity
E101	Ahmad	23	45000	ISB
E102	Hina	26	36000	ISB

F3 → m5 = (P2 ^ P3) → eCity != ‘ISB’ and eAge > 30

EmpNo	eName	eAge	eSalary	eCity
E103	Hamza	45	95000	RWP

F4 → m6 = (P2 ^ P3) → eCity != ‘ISB’ and eAge <= 30

EmpNo	eName	eAge	eSalary	eCity
E104	Sana	19	18000	LHR
E105	Suleman	29	15000	LHR

- It is clear that no correctness rule violates in above four fragments.

2. Derived Fragmentation:

Derived horizontal fragmentation, on the other hand, is the partitioning of a relation that results from predicates being defined on another relation.

Example:

- Let us consider two relations Employee and Project.

Employee (EmpNo, eName, eAge, eSalary, eCity)

Project (pNo, pName, Bonus, EmpNo)

- Here EmpNo in Project table is foreign key from Employee table.

- Data of Both table is given below.

Employee

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E101	Ahmad	23	45000	ISB
E102	Hina	26	36000	ISB
E103	Hamza	45	95000	RWP
E104	Sana	19	18000	LHR
E105	Suleman	29	15000	LHR
E106	Shahid	34	25000	ISB

Project

pNo	**pName**	**Bonus**	**EmpNo**
P101	Database	15000	E101
P102	CCN	12000	E101
P103	OOP	11000	E103
P104	PF	16000	E102
P105	ENG	5000	E105
P106	DLD	8000	E106
P107	DBS	19000	E104
P108	DDBS	10000	E104
P109	ELE	7000	E103
P110	CAL	9500	E106

  - Let us consider that we have four locations where our distributed database is located.

  - Each site has an application, which is retrieving data.

- First Application is retrieving all those employees whose city is ‘ISB’.

- Second Application is retrieving all those employees whose city is other than ‘ISB’.

- Third Application is retrieving all those employees whose age is above 30.

- Fourth Application is retrieving all those employees whose age is equal or less than 30.

Here we want to create fragments of “Project” table based on above applications.

As Application criteria exists in “Employee” table but we want to create fragments of “Project” table. This is known as “Derived Fragmentation”.

Procedure:

1- Perform Primary Fragmentation.

2- Take Semi-Join of Minterms selected from Primary fragmentation with Project Table.

Distributed Database System (CS-600)

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

Step 1: [Performing Primary Fragmentation]

- Based on application or criteria given, we will list down all predicates (where clause).

P1 → eCity = ‘ISB’

P2 → eCity != ‘ISB’

P3 → eAge > 30

P4 → eAge <=30

- As we have already seen above, if we create P1, P2, P3, P4 as fragments, Disjointness rule violates.
- This is why, we have to create combination of two predicates (because two columns are involved) called “Minterms”.
- Each “Minterm” (combination of predicate) will be examined if it is logically true or not.
- Finally, only correct “Minterms” will be selected as fragments.

Minterms:

Minterm	Predicate	Records	Remarks
m1	P1 ^ P2	0	No record satisfies, so not true
m2	P1 ^ P3	1 [E106]	Logically True
m3	P1 ^ P4	2 [E101, E102]	Logically True
m4	P2 ^ P1	0	Same as P1^P2 which was denied earlier
m5	P2 ^ P3	1[E103]	Logically True
m6	P2 ^ P4	2 [E104, E105]	Logically True
m7	P3 ^ P1	------	Same as P1^P3, no need to repeat
m8	P3 ^ P2	------	Same as P2^P3, no need to repeat
Now, As all our records of table accommodated in any of Minterm, so no need to create more minterms. We will keep on creating different combinations of predicates unless all of our records adjusts.			

Fragments will be,

m2 = (P1 ^ P3) → eCity = ‘ISB’ and eAge > 30

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E106	Shahid	34	25000	ISB

m3 = (P1 ^ P4) → eCity = ‘ISB’ and eAge <= 30

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E101	Ahmad	23	45000	ISB

E102	Hina	26	36000	ISB

m5 = (P2 ^ P3) → eCity != ‘ISB’ and eAge > 30

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E103	Hamza	45	95000	RWP

m6 = (P2 ^ P3) → eCity != ‘ISB’ and eAge <= 30

EmpNo	**eName**	**eAge**	**eSalary**	**eCity**
E104	Sana	19	18000	LHR
E105	Suleman	29	15000	LHR

Step 2: [Taking Semi-Join of Project table with all of above selected Minterms]

F1 → Project ▶m2

pNo	**pName**	**Bonus**	**EmpNo**
P106	DLD	8000	E106
P110	CAL	9500	E106

F2 → Project ▶m3

pNo	**pName**	**Bonus**	**EmpNo**
P101	Database	15000	E101
P102	CCN	12000	E101
P104	PF	16000	E102

F3 → Project ▶m5

pNo	**pName**	**Bonus**	**EmpNo**
P103	OOP	11000	E103
P109	ELE	7000	E103

F4 → Project ▶m6

pNo	**pName**	**Bonus**	**EmpNo**
P105	ENG	5000	E105
P107	DBS	19000	E104
P108	DDBS	10000	E104

- Above four fragments are valid Project table fragments in which no correctness rules violates.

Distributed Database System (CS-600)

Mr. Nauman Iqbal email id: nauman@biit.edu.pk Whatsapp# 0321-5821151
Mr. Ahsan email id: ahsan@biit.edu.pk, Whatsapp# 0333-5189656

Vertical Fragmentation

Vertical fragmentation of a relation R produces fragments R1;R2,…Rr, each of which contains a subset of R's attributes as well as the primary key of R.

The major information required for vertical fragmentation is related to applications.

There are two main steps for vertical fragmentation.

1. Calculate frequency
2. Calculate affinity matrix

Database Information

abcd

A1	**A2**	**A3**	**A4**	**A5**	**A6**

Usage Information

1. q1=Select * from abcd
2. q2=Select A1, A2 from abcd where A5>value
3. q3=Select A1, A3, A4 from abcd where A6=value
4. q4=Select A2, A3, A5 from abcd where A6=value
5. q5=Select A1, A3, A4,A5 from abcd
6. q6=Select A5, A6 from abcd

Access Information

1. Site 1: q1(10),q2(30),q3(20), q4(50),q5(100),q6(200)
2. Site 2: q1(100),q2(300),q3(20), q4(50),q5(0),q6(0)
3. Site 3: q1(10),q2(30),q3(0), q4(0),q5(10),q6(20)
4. Site 4: q1(100),q2(150),q3(200), q4(500),q5(10),q6(0)

Solution:

Create fragments of the given relation (table) "abcd" on the basis of the usage information and access information.

Usage Information:

q1, q2, q3, q4, q5, q6 shows the queries executing at some system.

Access Information:

There are 4 sites (site1, site2, site3 & site4) on which our system is deployed. Different queries execute on these sites for some specific number of times.

Example.

Site 1: q1(10) means at site 1, q1 executes 10 times.

Site 1: q2(30) means at site 2, q2 executes 30 times.

Step:

Now, calculating total number of query executed in a system using access information.

q1(10),	q2(30),	q3(20),	q4(50),	q5(100),	q6(200)
q1(100),	q2(300),	q3(20),	q4(50),	q5(0),	q6(0)
q1(10),	q2(30),	q3(0),	q4(0),	q5(10),	q6(20)
q1(100),	q2(150),	q3(200),	q4(500),	q5(10),	q6(0)
220	510	240	600	140	220

Step:

Now, affinity matrix will be created using the above access information.

	A1	A2	A3	A4	A5	A6
A1	---	730	600	600	870	460
A2	---	---	820	220	1330	820
A3	---	---	---	600	960	1060
A4	---	---	---	---	360	460
A5	---	---	---	---	---	1040
A6	---	---	---	---	---	---

- All columns of Table are listed horizontally and vertically.
- At intersection of A1 and A2 meant that how many times A1 and A2 has appeared in system together.
  - As A1 and A2 has appeared together in q1 and q2.
  - So sum of frequency of q1 and q2 = 220 + 510 = 730
- At intersection of A1 and A3 meant that how many times A1 and A2 has appeared in a system together.
  - As A1 and A3 has appeared together in q1, q3 and q5.
  - So sum of q1, q3 and q5 = 220 + 240 + 140 = 600
- By creating Affinity matrix, we will come to know which column is appearing the most with other columns.
- All those columns, which appears together more often, will be part of vertical fragments.

x-------------------------x