DATABASE - 2

3RD CLASS

COMPUTER SCIENCE DEPARTMENT

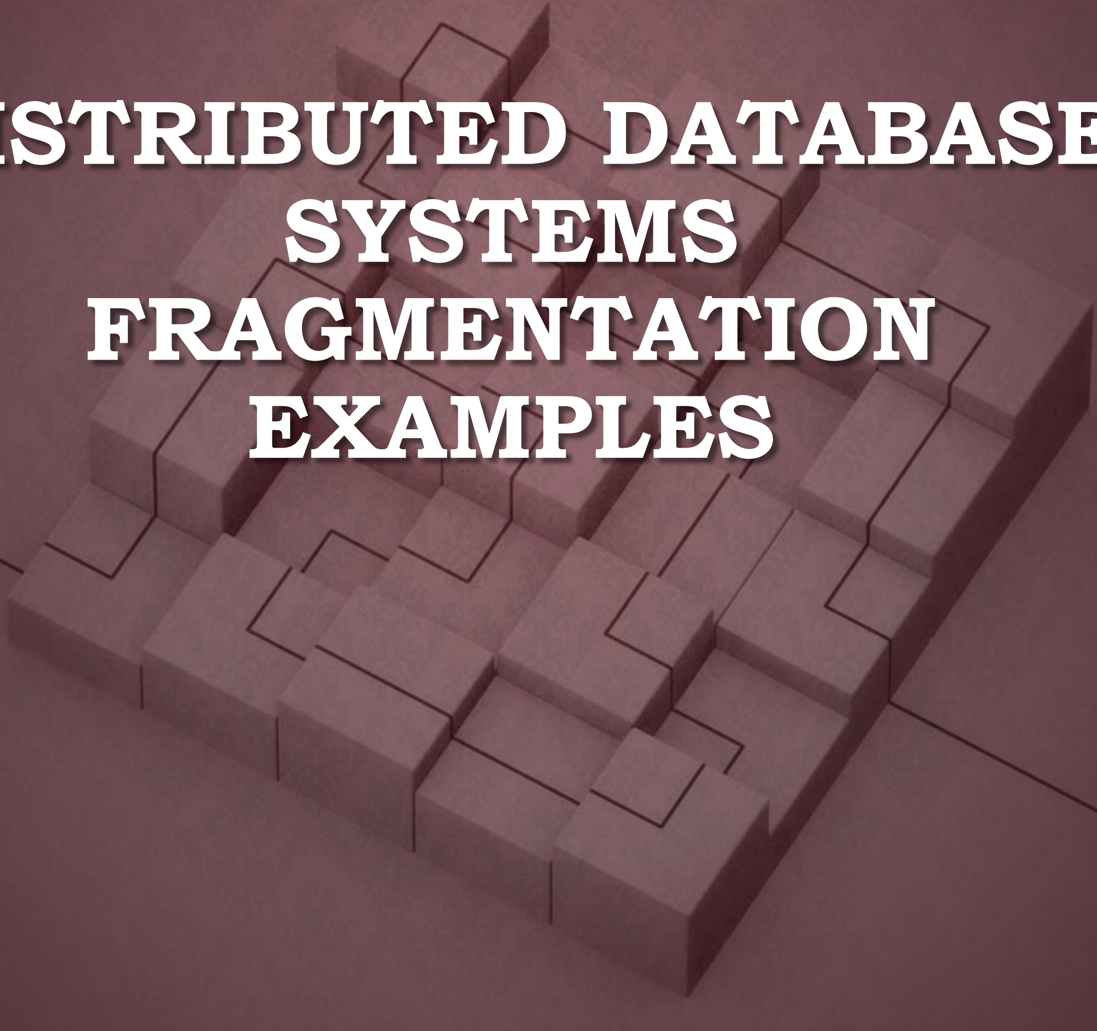
5th Lecture – Distributed Database Systems
Fragmentation Examples

Sunday 16th of Feb. 2025

LECTURER :

DR. RAYAN YOUSIF ALKHAYAT

DISTRIBUTED DATABASE SYSTEMS FRAGMENTATION EXAMPLES



EXAMPLE 1: HORIZONTAL FRAGMENTATION

PROBLEM STATEMENT

A company database contains the **EMPLOYEE** table with the following attributes:

EMP_ID (Primary Key)

NAME

DEPARTMENT

SALARY

LOCATION

The company wants to **horizontally fragment** the table based on employee **salaries**, where:

EMP_LOW: Employees with **SALARY ≤ 50,000**

EMP_HIGH: Employees with **SALARY > 50,000**

Design the horizontal fragmentation and verify its correctness.

SOLUTION

- Using **Primary Horizontal Fragmentation (PHF)**, we define selection predicates:
- **P1:** SALARY $\leq$ 50,000
- **P2:** SALARY $>$ 50,000
- The fragmentation is defined as:
- **EMP_LOW = σ (SALARY $\leq$ 50,000) (EMPLOYEE)**
- **EMP_HIGH = σ (SALARY $>$ 50,000) (EMPLOYEE)**

SOLUTION

Correctness Checks:

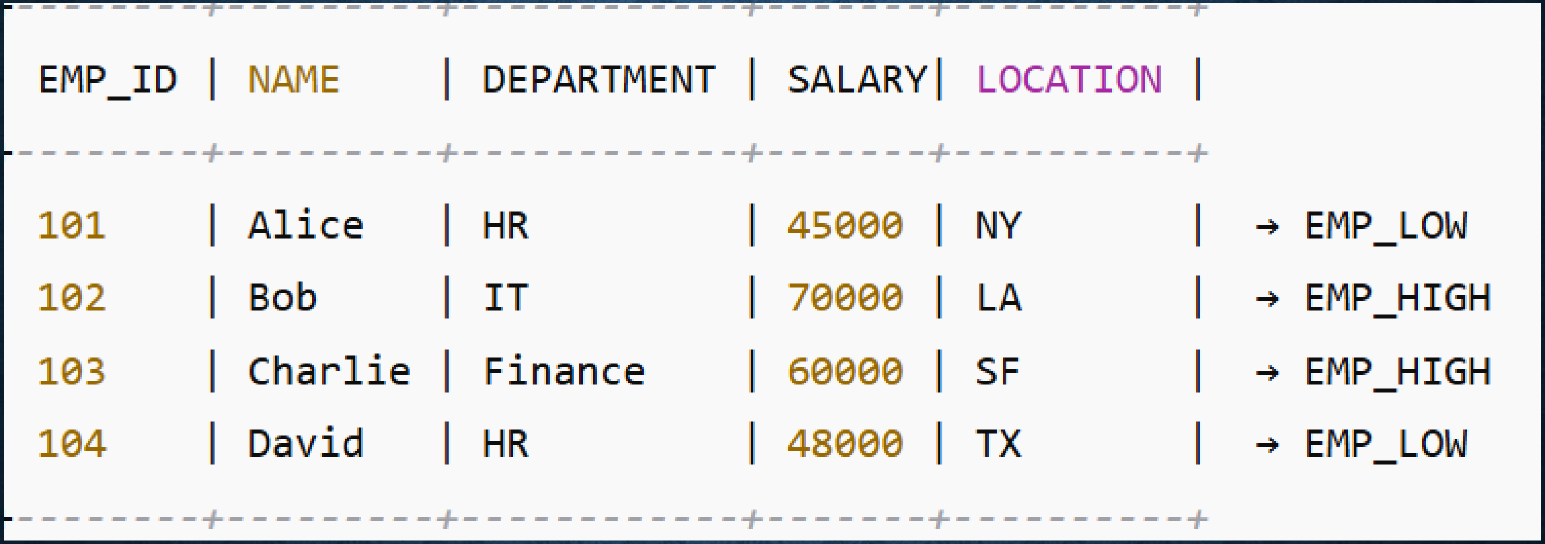
Completeness: Every tuple in **EMPLOYEE** is included in either **EMP_LOW** or **EMP_HIGH**.

Reconstruction: The original table can be reconstructed using **UNION**:

EMPLOYEE=EMP_LOW∪EMP_HIGHEMPLOYEE =

Disjointness: No tuple appears in both fragments since **P1** and **P2** are mutually exclusive.

SOLUTION



```
---------+----------+-------------+-------+-----------+
EMP_ID | NAME     | DEPARTMENT  | SALARY| LOCATION  |
---------+----------+-------------+-------+-----------+
101    | Alice    | HR          | 45000 | NY        |  → EMP_LOW
102    | Bob      | IT          | 70000 | LA        |  → EMP_HIGH
103    | Charlie  | Finance     | 60000 | SF        |  → EMP_HIGH
104    | David    | HR          | 48000 | TX        |  → EMP_LOW
---------+----------+-------------+-------+-----------+
```

EXAMPLE 2: VERTICAL FRAGMENTATION

PROBLEM STATEMENT

- A hospital database contains the **PATIENT** table with the following attributes:

PATIENT_ID (Primary Key)

NAME

AGE

DIAGNOSIS

TREATMENT

The database is accessed by two different applications:

Administrative System: Requires **PATIENT_ID, NAME, and AGE.**

Medical System: Requires **PATIENT_ID, DIAGNOSIS, and TREATMENT**.

- Design a **Vertical Fragmentation** that optimizes access based on application needs.

SOLUTION

Using **Vertical Fragmentation**, we define the following fragments:

PATIENT_ADMIN = π (PATIENT_ID, NAME, AGE) (PATIENT)

PATIENT_MEDICAL = π (PATIENT_ID, DIAGNOSIS, TREATMENT) (PATIENT)

SOLUTION

Correctness Checks:

Completeness:

The set of attributes in **PATIENT_ADMIN** and **PATIENT_MEDICAL** covers all attributes of **PATIENT**.

Reconstruction:

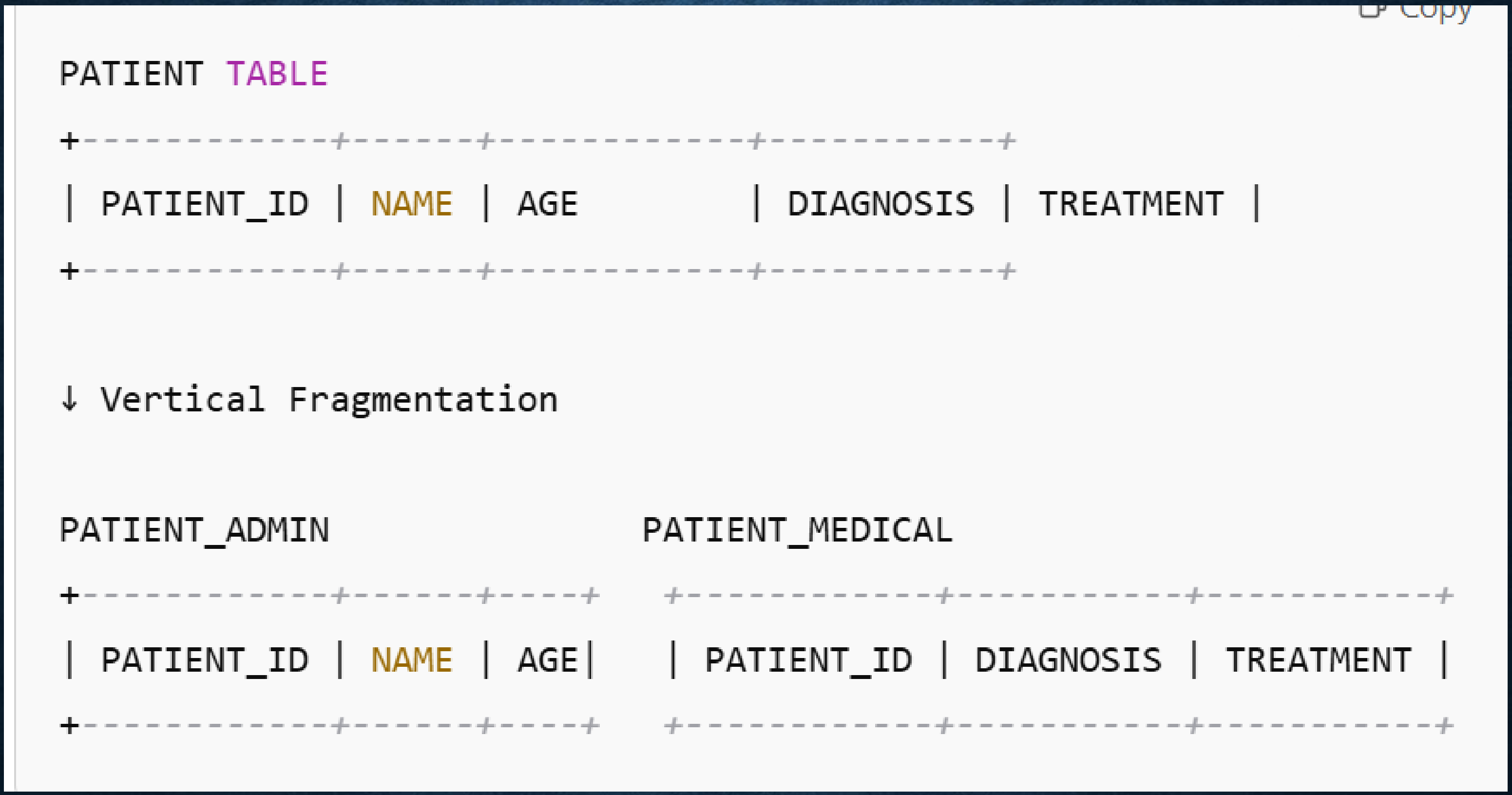
The original table can be reconstructed using **JOIN** on **PATIENT_ID**:

$\text{PATIENT}=\text{PATIENT\_ADMIN}\bowtie\text{PATIENT}$

Disjointness:

The **PATIENT_ID** is duplicated in both fragments to maintain integrity, but all other attributes are uniquely assigned.

SOLUTION



```
PATIENT TABLE
+------------+------+------------+-----------+
| PATIENT_ID | NAME | AGE        | DIAGNOSIS | TREATMENT |
+------------+------+------------+-----------+

↓ Vertical Fragmentation

PATIENT_ADMIN               PATIENT_MEDICAL
+------------+------+----+   +------------+-----------+-----------+
| PATIENT_ID | NAME | AGE|   | PATIENT_ID | DIAGNOSIS | TREATMENT |
+------------+------+----+   +------------+-----------+-----------+
```

END OF LECTURE 5