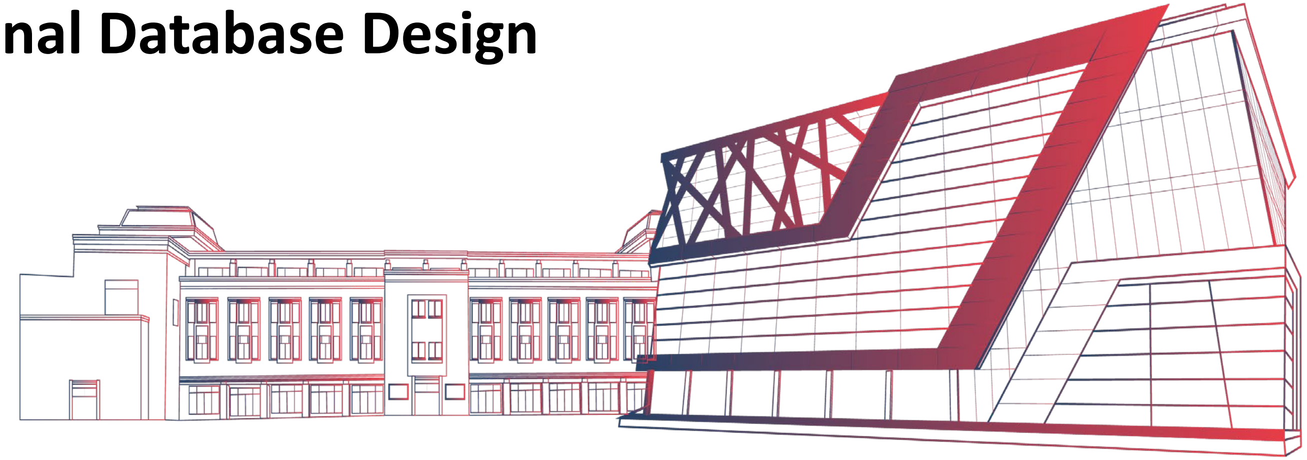
# UNIT III

## Relational Database Design



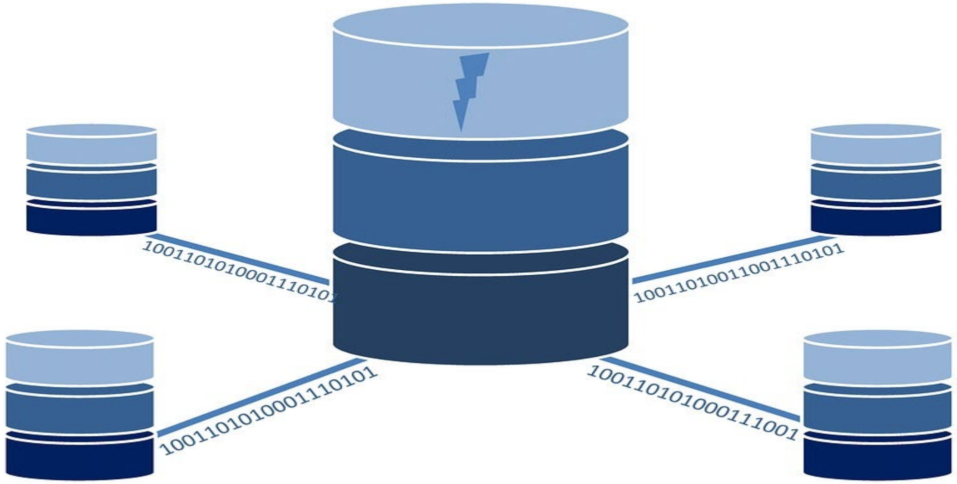
Lecture 18

# Dependency Preservation

# &

# Lossless Join Conditions

# DECOMPOSING RELATIONS TO ACHIEVE HIGHER NORMAL FORMS



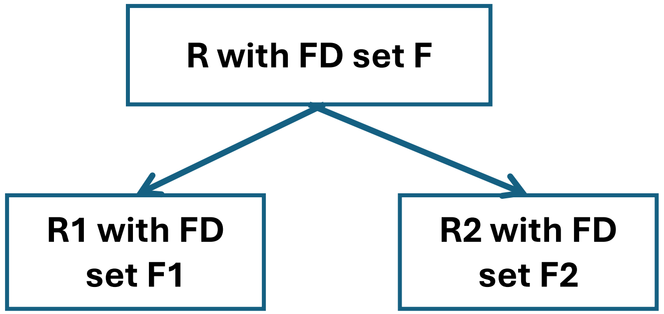
# Properties of Relational Decompositions

- **Attribute preservation condition**: Each attribute in R will appear in at least one relation schema $R_i$ in the decomposition so that no attributes are "lost".

- Another goal of decomposition is to have each individual relation $R_i$ in the decomposition D be in BCNF or 3NF.

# Dependency Preservation Property of Relational Decompositions

Let R be a relation with FD set F, decomposed into R1, R2,....Rn with FD sets F1, F2,...Fn




**<u>Case 1:</u>** If all FDs of F are implied in

{ F1 U F2 U F3.... U F4 },

**decomposition is dependency preserving**

**<u>Case 2:</u>** If all FDs of F are not implied in

{ F1 U F2 U F3.... U F4 },

**decomposition is not dependency preserving**

# Dependency Preservation Example



Let a relation R(A,B,C,D) and a set FDs

$$F = \{ A \rightarrow B ,\ A \rightarrow C ,\ C \rightarrow D\}$$

A relation R is decomposed into –

$R_1 = (A, B, C)$ with FDs $F_1 = \{A \rightarrow B, A \rightarrow C\}$

$R_2 = (C, D)$ with FDs $F_2 = \{C \rightarrow D\}$.

$F' = F_1 \cup F_2 = \{A \rightarrow B, A \rightarrow C, C \rightarrow D\}$

$F' = F$.

**Thus, the decomposition is dependency preserving decomposition.**

# Practice Drill


1. Relation R ( ABCDE ) with FD set

$$F = \{ A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow E, D \rightarrow B \}$$

$$D = \{ AB, BC, CD, DE \}$$

2. Relation R ( ABCD ) with FD set

$$F = \{ AB \rightarrow CD, D \rightarrow A \}$$

$$D = \{ ABC, BCD, AD \}$$

3. Relation R ( ABCDEF ) with FD set

$$F = \{ A \rightarrow BCDEF, BC \rightarrow ADEF, D \rightarrow E, B \rightarrow F \}$$

$$D = \{ ABCD, DE, BF \}$$

4. Relation R ( ABCDE ) with FD set

$$F = \{ A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A \}$$

$$D = \{ ABCE, BD \}$$

# Solution Practice Drill

1. **F = { A → B, B → C, C → D, D → E, D → B}**

   **D = { AB, BC, CD, DE }**

- Finding non-trivial FDs of decomposed relations in D
- F1 = { A → B}
- F2 = { B → C, C → B}
- F3 = { C → D, D → C }
- F4 = { D → E}
- Check all FDs of F are implied in { F1 U F2 U F3 U F4 } - YES
- **Thus, decomposition is dependency preserved**

# Solution Practice Drill


2. **$F = \{ AB \rightarrow CD, D \rightarrow A \}$**

   **$D = \{ ABC, BCD, AD \}$**

- Finding non-trivial FDs of decomposed relations in D
- $F1 = \{ AB \rightarrow C \}$
- $F2 = \{ BD \rightarrow C \}$
- $F3 = \{ D \rightarrow A \}$
- Check all FDs of F are implied in { F1 U F2 U F3 } – NO

- **Thus, decomposition is not dependency preserved**

# Solution Practice Drill

**3. F = { A → BCDEF, BC → ADEF, D → E, B → F}**

**D = { ABCD, DE, BF }**

- Finding non-trivial FDs of decomposed relations in D
- F1 = { A → BCD, BC → AD}
- F2 = { D → E}
- F3 = { B → F }
- Check all FDs of F are implied in { F1 U F2 U F3 } - YES
- **Thus, decomposition is dependency preserved**

# Solution Practice Drill

**4. F = { A → BC, CD → E, B → D, E → A }**

**D = { ABCE, BD }**

- Finding non-trivial FDs of decomposed relations in D
- F1 = { A → BCE, E → ABC, BC → AE }
- F2 = { B → D}
- Check all FDs of F are implied in { F1 U F2 U F3 } - NO
- **Thus, decomposition is not dependency preserved**

# Cartesian (or Cross) Product Operation

- This operation is used to combine tuples from two relations in a combinatorial fashion.
- In general, the result of $R(A_1, A_2, ..., A_n)$ **X** $S(B_1, B_2, ..., B_m)$ is a relation Q with degree n + m attributes $Q(A_1, A_2, ..., A_n, B_1, B_2, ..., B_m)$, in that order.
- The resulting relation Q has one tuple for each combination of tuples—one from R and one from S.
- Hence, if R has $n_R$ tuples (denoted as $|R| = n_R$ ),

  and S has $n_S$ tuples, then

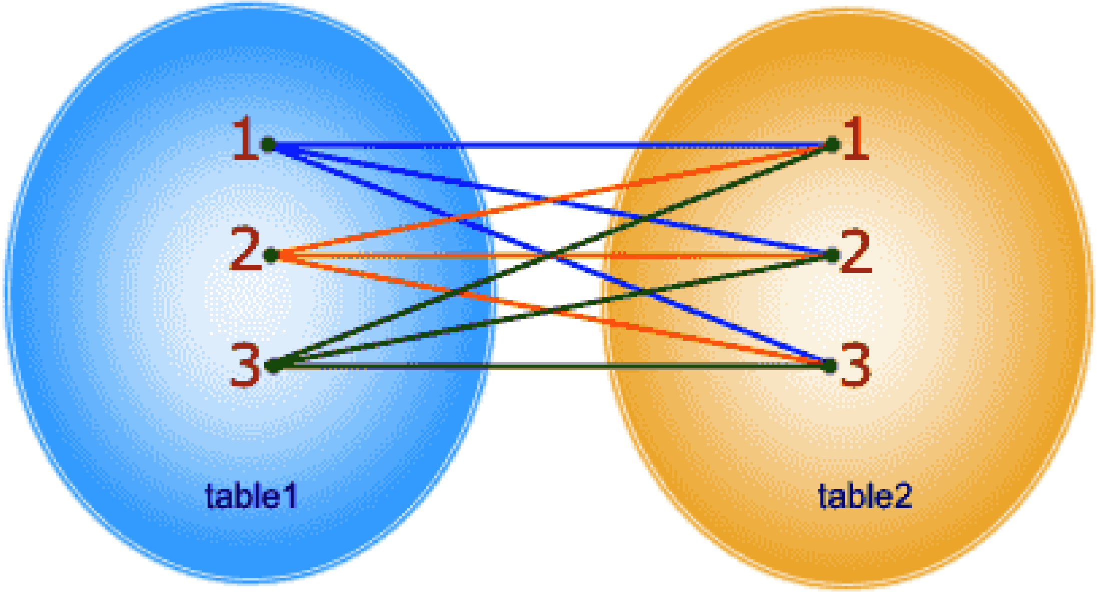
  **| R x S | will have $n_R * n_S$ tuples.**

- **Note: The two operands do NOT have to be "type compatible”**

- **What is Type Compatible?**

The operand relations $R_1(A_1, A_2, \ldots, A_n)$ and $R_2(B_1, B_2, \ldots, B_n)$ must have the same number of attributes, and the domains of corresponding attributes must be compatible; that is, $dom(A_i)=dom(B_i)$ for $i=1, 2, \ldots, n$.

# Cartesian Product Example




**In cartesian product, each row from 1st table joins with all the rows of another table. If first table contains ‘x’ rows and second table contains ‘y’ rows, the result set will contain x * y rows**

# JOIN Operation

- **Join** is a derived operator that uses a sequence of cartesian product followed by selection of related tuples from two relations and then projection of distinct attributes. It is denoted by a . $\bowtie$
- This operation is very important for any relational database with more than a single relation, because it allows us to process relationships among relations.
- The general form of a join operation on two relations $R(A_1, A_2, \ldots, A_n)$ and $S(B_1, B_2, \ldots, B_m)$ is:

$$\mathbf{R} \bowtie_{\textbf{<join condition>}} \mathbf{S}$$

where R and S can be any relations that result from general *relational algebra expressions.*

# Join Operation Example

**Relation R**

| A | B | C |
|---|---|---|
| 2 | 4 | 3 |
| 3 | 4 | 2 |
| 4 | 4 | 3 |
| 5 | 3 | 3 |

**Relation S**

| C | D |
|---|---|
| 3 | 4 |
| 2 | 4 |
| 3 | 3 |

| A | B | C | C | D |
|---|---|---|---|---|
| 2 | 4 | 3 | 3 | 4 |
| 2 | 4 | 3 | 2 | 4 |
| 2 | 4 | 3 | 3 | 3 |
| 3 | 4 | 2 | 3 | 4 |
| 3 | 4 | 2 | 2 | 4 |
| 3 | 4 | 2 | 3 | 3 |
| 4 | 4 | 3 | 3 | 4 |
| 4 | 4 | 3 | 2 | 4 |
| 4 | 4 | 3 | 3 | 3 |
| 5 | 3 | 3 | 3 | 4 |
| 5 | 3 | 3 | 2 | 4 |
| 5 | 3 | 3 | 3 | 3 |

**Step 1: Cross Product R X S**

| A | B | C | C | D |
|---|---|---|---|---|
| 2 | 4 | 3 | 3 | 4 |
| 2 | 4 | 3 | 3 | 3 |
| 3 | 4 | 2 | 2 | 4 |
| 4 | 4 | 3 | 3 | 4 |
| 4 | 4 | 3 | 3 | 3 |
| 5 | 3 | 3 | 3 | 4 |
| 5 | 3 | 3 | 3 | 3 |

**Step 2:** $\sigma_{R.C = S.C}$ ( R X S)

| A | B | C | D |
|---|---|---|---|
| 2 | 4 | 3 | 4 |
| 2 | 4 | 3 | 3 |
| 3 | 4 | 2 | 4 |
| 4 | 4 | 3 | 4 |
| 4 | 4 | 3 | 3 |
| 5 | 3 | 3 | 4 |
| 5 | 3 | 3 | 3 |

**Step 3:** $\pi$ ($\sigma_{R.C = S.C}$ ( R X S))

# Lossless (Non-additive) Join Property of Relational Decomposition

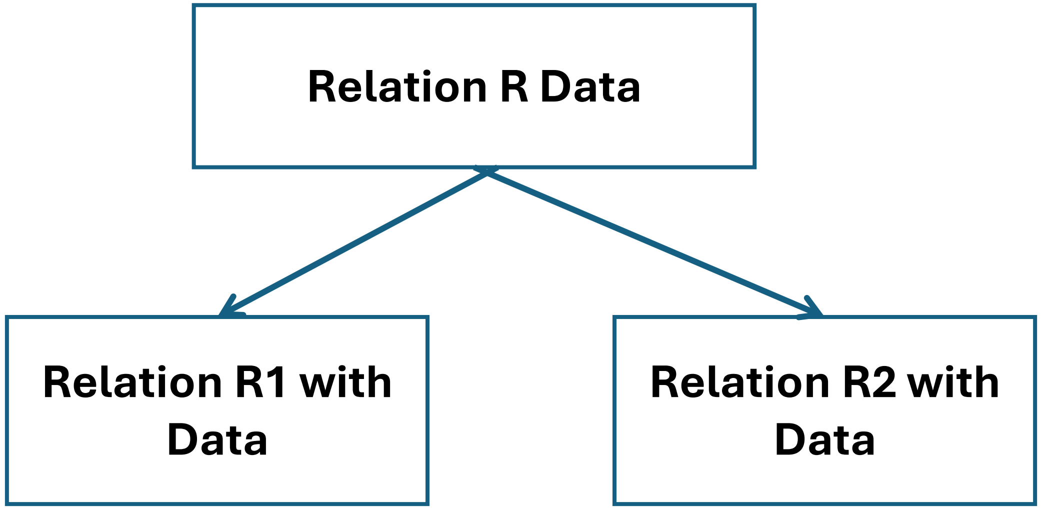
- Let R be the relational schema decomposed into R1, R2,…Rn. In general,

$$\{ R1 \bowtie R2 \bowtie \ldots. \bowtie Rn \} \supseteq R$$

- If $\{ R1 \bowtie R2 \bowtie \ldots. \bowtie Rn \} = R \Rightarrow$ Lossless Join Decomposition
- If $\{ R1 \bowtie R2 \bowtie \ldots. \bowtie Rn \} \supset R \Rightarrow$ Lossy Join Decomposition

# Lossless Join Property (Cont.)

**Note:** The word loss in *lossless* refers to *loss of information*, not to loss of tuples. In fact, for "loss of information" a better term is "<u>addition of spurious information</u>".



# Lossless Join Decompostion Example

**EmpInfo**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc | Dept_ID | Dept_Name |
|---|---|---|---|---|---|
| E001 | Jacob | 29 | Alabama | Dpt1 | Operations |
| E002 | Henry | 32 | Alabama | Dpt2 | HR |
| E003 | Tom | 22 | Texas | Dpt3 | Finance |

**EmpDetails**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc |
|---|---|---|---|
| E001 | Jacob | 29 | Alabama |
| E002 | Henry | 32 | Alabama |
| E003 | Tom | 22 | Texas |

**DeptDetails**

| Dept_ID | Emp_ID | Dept_Name |
|---|---|---|
| Dpt1 | E001 | Operations |
| Dpt2 | E002 | HR |
| Dpt3 | E003 | Finance |

**Join Result**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc | Dept_ID | Dept_Name |
|---|---|---|---|---|---|
| E001 | Jacob | 29 | Alabama | Dpt1 | Operations |
| E002 | Henry | 32 | Alabama | Dpt2 | HR |
| E003 | Tom | 22 | Texas | Dpt3 | Finance |

# Lossy Join Decompostion Example

**EmpInfo**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc | Dept_ID | Dept_Name |
|---|---|---|---|---|---|
| E001 | Jacob | 29 | Alabama | Dpt1 | Operations |
| E001 | Jacob | 29 | Alabama | Dpt2 | HR |
| E002 | Tom | 32 | Alabama | Dpt1 | Operations |
| E003 | Tom | 22 | Texas | Dpt2 | HR |

**EmpDetails**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc |
|---|---|---|---|
| E001 | Jacob | 29 | Alabama |
| E002 | Tom | 32 | Alabama |
| E003 | Tom | 22 | Texas |

**DeptDetails**

| Dept_ID | Dept_Name | Emp_Name |
|---|---|---|
| Dpt1 | Operations | Jacob |
| Dpt2 | HR | Jacob |
| Dpt1 | Operations | Tom |
| Dpt2 | HR | Tom |

**Join Result**

| Emp_ID | Emp_Name | Emp_Age | Emp_Loc | Dept_ID | Dept_Name |
|---|---|---|---|---|---|
| E001 | Jacob | 29 | Alabama | Dpt1 | Operations |
| E001 | Jacob | 29 | Alabama | Dpt2 | HR |
| E002 | Tom | 32 | Alabama | Dpt1 | Operations |
| E003 | Tom | 22 | Texas | Dpt2 | HR |
| **E002** | **Tom** | **32** | **Alabama** | **Dpt2** | **HR** |
| **E003** | **Tom** | **22** | **Texas** | **Dpt1** | **Operations** |

# How to Check a Decomposition is Lossless or Lossy?

- Let R be the relational schema decomposed into R1, R2,...Rn and **$R1 \cap R2 = X$ attribute**
- **Case 1: If X is superkey for atleast one relation R1 or R2, the decomposition is lossless.**
- **Case 2: If If X is not superkey for**

  **atleast one relation R1 or R2,**

  **the decomposition is lossy.**
- **Case 3: If** $R1 \cap R2 = \emptyset$,

  **the decomposition is lossy.**

# Lossless/Lossy Decomposition Example

- Consider a relation schema R ( A , B , C , D ) with the functional dependencies A → B and C → D. Determine whether the decomposition of R into $R_1$ ( A, B ) and $R_2$ (C, D) is lossless or lossy.

- **Solution:**

  $R_1 ( A , B ) \cap R_2 ( C , D ) = \Phi$

  Clearly, intersection of the sub relations is null.

  Thus, the decomposition is lossy.

# Practice Drill

Consider a relation schema and decompositions and determine if it is a lossless or lossy decomposition:

1. R ( A , B , C , D )

   F = {A → B, B → C, C → D, D → B } D = ( AB, BC, BD )

2. R (A, B, C, D, E)

   F = { AB → C, C → D, B → E } D = ( ABC, CD)

3. R (A, B, C, D, E)

   F = { AB → C, C → D, B → E } D = ( ABC, CDE )

4. R (A, B, C, D, E)

   F = { AB → C, C → D, B → E } D = ( ABC, ABDE)

# Solution Practice Drill

1. F = { A → B, B → C, C → D, D → B }

   $R_1$ ( A , B ) , $R_2$ ( B , C ) and $R_3$ ( B , D )

| Case 1 | Case 2 |
|---|---|
| R1 ∩ R2 = B | |
| B+ = B, C for R2 (B, C) ∴ B is superkey in R2 | R2 ∩ R3 = B |
| R12 (A, B, C) ∩ R3 = B | B+ = B, C for R2 (B, C) ∴ B is superkey in R2 |
| B+ = B, C for R12 (A, B, C) | R1 ∩ R23 (B, C, D) = B |
| B+ = B for R3 (B, D) ∴ B is not superkey for any sub-relation | B+ = B, C, D ∴ B is superkey in R23 |

∴ It is a lossless join decomposition

# Solution Practice Drill

2. F = { AB → C, C → D, B → E }    D = ( ABC, CD)

C+ = C, D ∴ C is superkey in relation CD

This is a **lossy join decomposition** though common attribute is a superkey for CD sub-relation, because **E attribute is lost in decomposition.**

3. F = { AB → C, C → D, B → E }    D = ( ABC, CDE )

C+= C, D

Thus, this is **lossy join decomposition** as C is not superkey for any sub-relation.

## Solution Practice Drill

4. $F = \{ AB \rightarrow C, C \rightarrow D, B \rightarrow E \}$

$D = ( ABC, ABDE)$

AB+ = A, B, C for sub-relation ABC

∴ AB is superkey in relation ABC

Thus, this is **lossless join decomposition.**

# Thanks!!