CHAPTER 4:
LOGICAL DATABASE DESIGN & THE RELATIONAL MODEL
(*PART II - NORMALIZATION*)

Modern Database Management

12th Edition

Jeff Hoffer, Ramesh Venkataraman, Heikki Topi

OBJECTIVES – PART 2

- State two properties of candidate keys
- Define first, second, and third normal form
- Describe problems from merging relations
- Transform E-R and EER diagrams to relations
- Create tables with entity and relational integrity constraints
- Use normalization to convert anomalous tables to well-structured relations

DATA NORMALIZATION

- Normalization is a formal process for deciding which attributes should be grouped together in a relation so that all anomalies are removed
- Normalization is based on *normal forms* and *functional dependencies*
  - Normal form:
    - 1st, 2nd, 3rd
  - Functional dependencies:
    - full, partial, transitive

PURPOSE OF NORMALIZATION (2017 NEW)

- Normalization takes a "big", clumsy table (containing multiple themes/multiple logical dependencies/multiple entity types), and decompose it into several smaller tables that are:

1. One theme/logical dependency in each table;
2. Each "smaller" table is in full dependency;
3. With referential integrity among the related tables

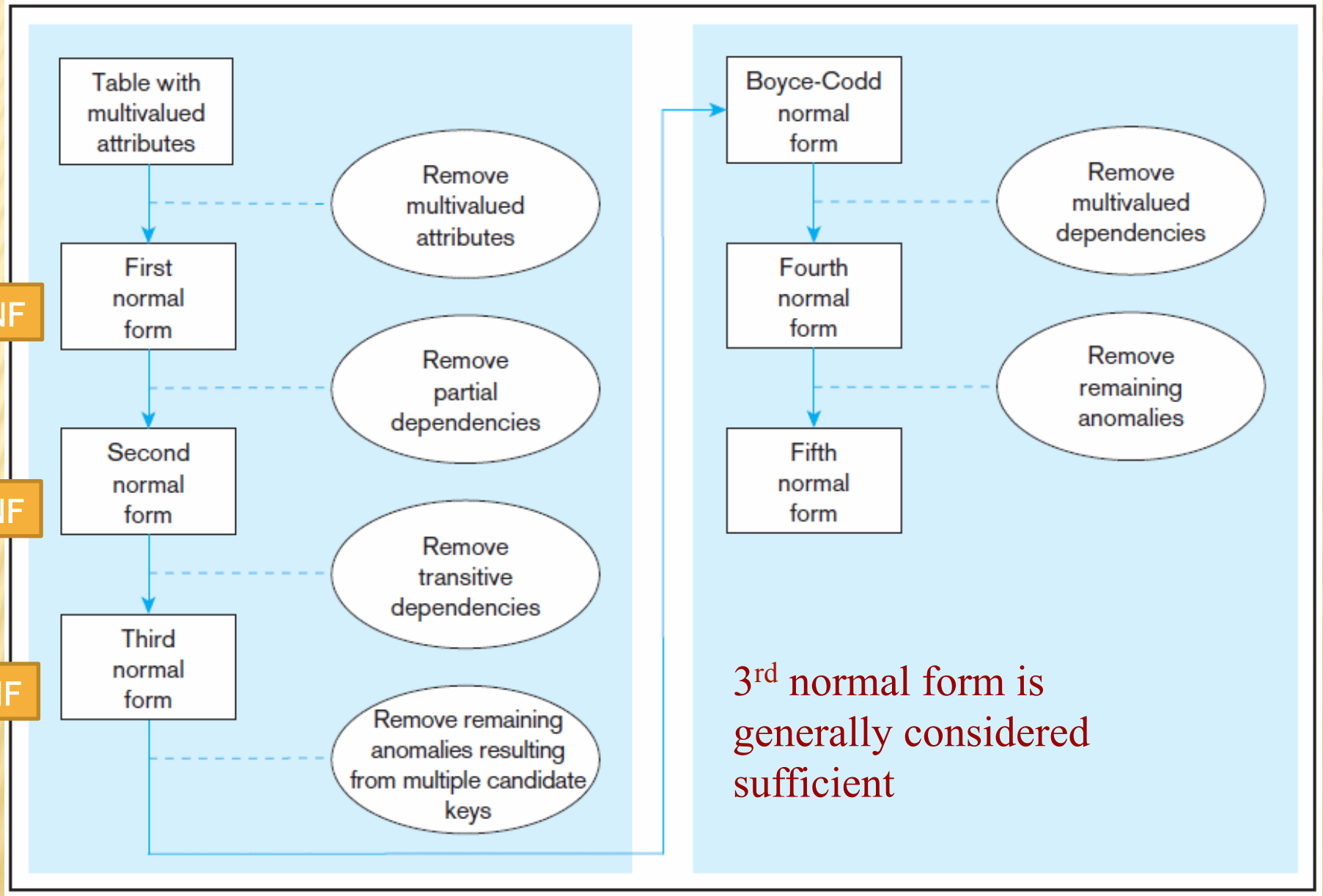
FOUR POINT GOALS OF NORMALIZATION

- Minimize data redundancy, thereby avoiding anomalies and conserving storage
- Simplify the enforcement of referential integrity constraints
- Make it easier to maintain data (Insert, Update and Delete)
- Provide a better design that is improved representation of the real world and a stronger basis for future growth.

WELL-STRUCTURED RELATIONS

- Goal is to avoid anomalies
  - **Insertion Anomaly**–adding new rows forces user to create duplicate data
  - **Deletion Anomaly**–deleting rows may cause a loss of data that would be needed for other future rows
  - **Modification Anomaly**–changing data in a row forces changes to other rows because of duplication

Figure 4.22 Steps in normalization



FUNCTIONAL DEPENDENCIES AND KEYS

- Functional Dependency: The value of one attribute (the **determinant**) determines the value of another attribute (the dependant)
  - A determinant in a database table is any **attribute** that you can use to determine the values assigned to other attribute(s) ***in the same row***.
  - If we call the determinant "D", and other attributes "A", then
    - Knowing D (value), we can know A (values)
      - Denoted as: D → A
    - Or: "Given a D-value, we can have a specific A-value"

Notation

FUNCTIONAL DEPENDENCIES AND KEYS (CONT)

- Candidate Key:
  - A unique identifier. One of the candidate keys will become the primary key
    - E.g. perhaps there is both credit card number and SS# in a table…in this case both can be candidate keys

EXAMPLE–FIGURE 4-2B

Example of Anomalies

EMPLOYEE2

EmpID	Name	DeptName	Salary	CourseTitle	DateCompleted
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/2015
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/2015
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/2015
110	Chris Lucero	Info Systems	43,000	Visual Basic	1/12/2015
110	Chris Lucero	Info Systems	43,000	C++	4/22/2015
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/2015
150	Susan Martin	Marketing	42,000	Java	8/12/2015

Question–Is this a relation?

Question–What's the primary key?

ANOMALIES IN THIS TABLE

1. **Insertion**–can't enter a new employee without having the employee take a class
2. **Deletion**–if we remove employee 140, we lose information about the existence of a Tax Acc class
3. **Modification**–giving a salary increase to employee 100 forces us to update multiple records

Why do these anomalies exist?

Because there are two themes (entity types) in this one relation: EMPLOYEE and COURSE

This results in data duplication and an unnecessary dependency between the entities

EXAMPLE–FIGURE 4-2B

EMPLOYEE2

EmpID	Name	DeptName	Salary	CourseTitle	DateCompleted
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/201X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/201X
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/201X
110	Chris Lucero	Info Systems	43,000	Visual Basic	1/12/201X
110	Chris Lucero	Info Systems	43,000	C++	4/22/201X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/201X
150	Susan Martin	Marketing	42,000	Java	8/12/201X

Theme?

Theme?

FULL DEPENDENCY ("FUNC DEP")

- The dependency that departs from the primary key (could be single-field or composite) and ends at non-key attributes
  - The legitimate dependency (the "Good", the "MUST HAVE")

Basic concept

- Relations ALWAYS have a full dependency
  - There are not always partial or transitive dependencies

Figure 4.22 Steps in normalization

- From 1st normal form (1NF) to 2nd normal form (2NF):
  - Remove partial dependencies ➔ 2nd normal form
  - Partial dependency ("PD"): when there are two attributes working as the determinant, some determinees depend ONLY on one PART of the key

PD

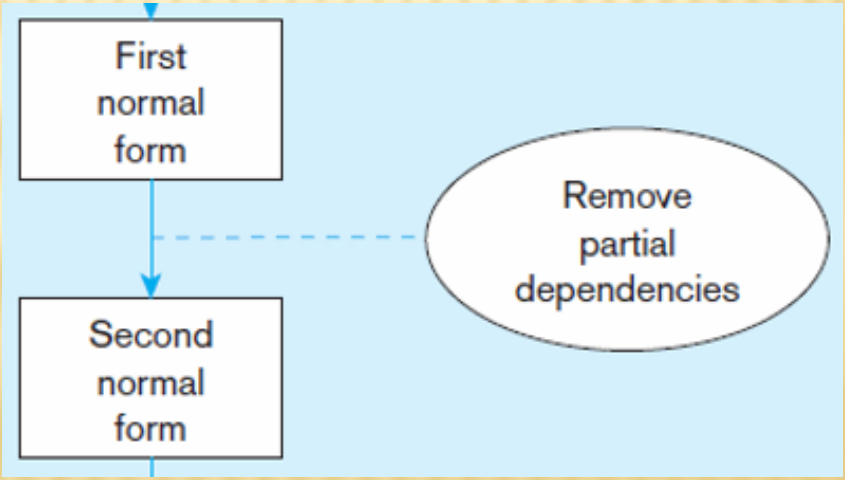
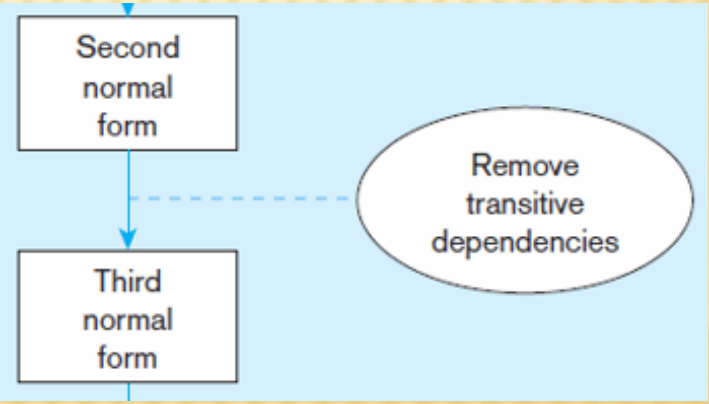


Figure 4.22 Steps in normalization

- From 2nd normal form (2NF) to 3rd normal form (3NF):
  - Remove transitive dependencies ➔ 3rd normal form
  - Transitive dependency ("TD"): when there are non-key attributes acting/behaving as a determinant (of other attributes) - determines depend on a non-key attribute

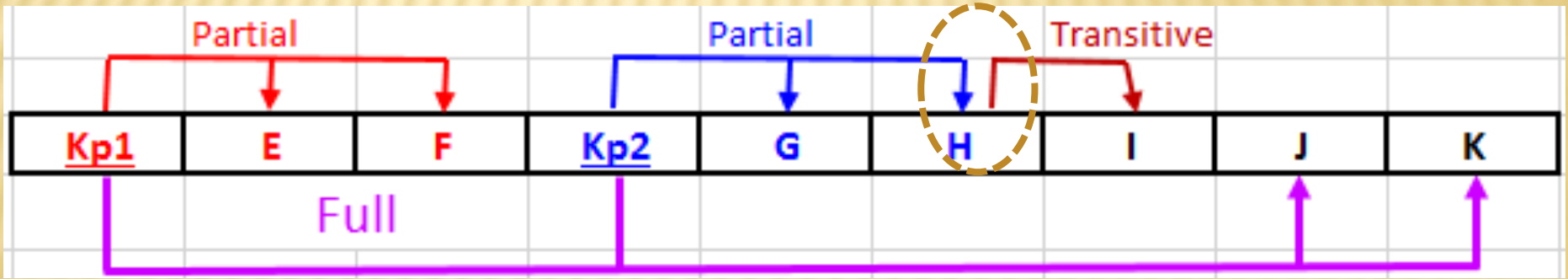
TD



PARTIAL, TRANSITIVE, AND FULL FUNCTIONAL DEPENDENCIES

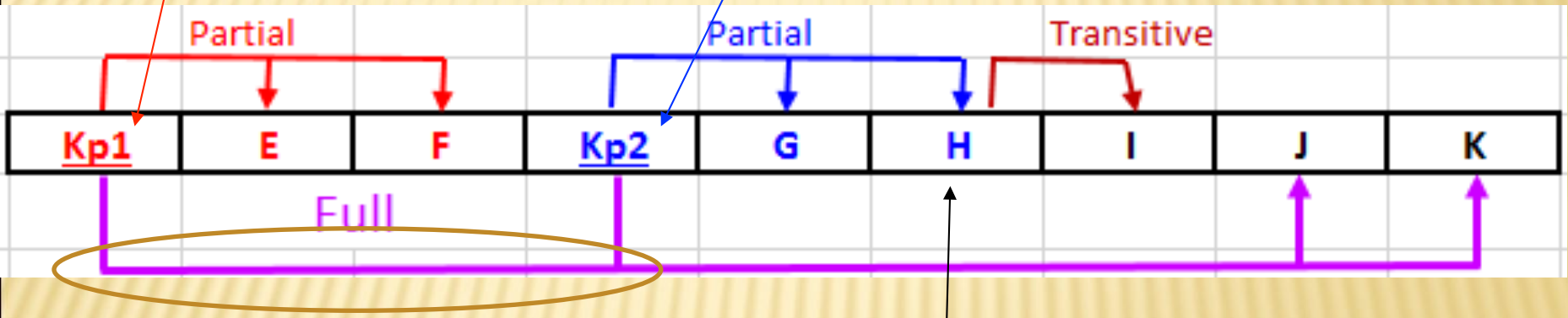
1. Partial: Note the PK here has TWO PARTS
2. Transitive: Like skipping a stone

3. Full: non-key attributes FULLY depend on whole set pf key attributes



FULL, PARTIAL, AND TRANSITIVE DEPENDENCY CLOSE-UPS

- "Kp1" – Key part 1; "Kp2" – key part 2



- Kp1 + Kp2 is the whole key
-

H here is non-key

FIRST NORMAL FORM

- No multivalued attributes
  - Every attribute value is atomic
- Fig. 4-25 *is not* in 1st Normal Form (multivalued attributes) ➔ it is not a relation.
- Fig. 4-26 *is* in 1st Normal form.
- ***All relations*** **are in 1st Normal Form.**

Fig 4-25 Table with multivalued attributes, not in 1 NF

FIGURE 4-25 INVOICE data (Pine Valley Furniture Company)

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Descriptio n	Product F inish	Product StandardPrice	Ordered Quantity
1006	10/24/2010	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
					5	Writer's Desk	Cherry	325.00	2
					4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
					4	Entertainment Center	Natural Maple	650.00	3

Fig 4-26 Table w no multivalued attributes & w unique rows, in 1NF

Repeat

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Description	Product Finish	Product StandardPrice	Ordered Quantity
1006	10/24/2010	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

Repeat

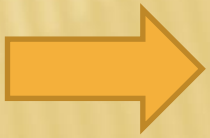
FIGURE 4-26 INVOICE relation (1NF) (Pine Valley Furniture Company)

ANOMALIES IN THIS TABLE (4-26)

- **Insertion**–if new product is ordered for order 1007 of existing customer, customer data must be re-entered, causing duplication
- **Deletion**–if we delete the customer 1006, we lose information concerning dining table's finish and price
- **Update**–changing the price of product ID 4 requires update in multiple records

Why do these anomalies exist?

Because there are multiple themes (entity types) in one relation. This results in duplication and an unnecessary dependency between the entities



Intuitive examination for the "themes" in the following table

Attributes in 1st theme

Attributes in 2nd theme

Attributes in 3rd theme

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Description	Product Finish	Product StandardPrice	Ordered Quantity
1006	10/24/2010	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2010	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
1007	10/25/2010	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

FIGU 4-26 INVOICE relation (1NF) (Pine Valley Furniture mpany)

Attributes in 4th theme

SECOND NORMAL FORM

- 1NF, PLUS *every non-key attribute is fully functionally dependent on the ENTIRE primary key*
  - Every non-key attribute must be defined by the entire key, not by only one part of the key
    - ➔ *No partial functional dependencies*
- Discussion: "Entire primary key" implies that this key is a _____ key?

Figure 4-27 Functional dependency diagram for INVOICE

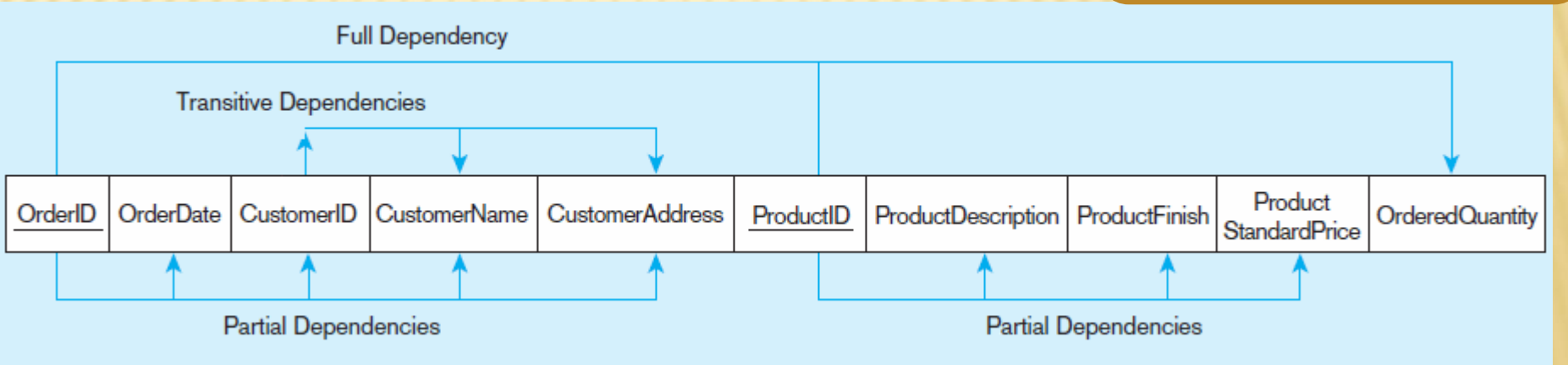
OrderID ➔ OrderDate, CustomerID, CustomerName, CustomerAddress

ProductID ➔ ProductDescription, ProductFinish, ProductStandardPrice

OrderID, ProductID ➔ OrderQuantity

CustomerID ➔ CustomerName, CustomerAddress

Need to know not only func deps, but also the **TYPES** of func deps



Therefore, NOT in 2nd Normal Form

So, what is PD? TD? FD?

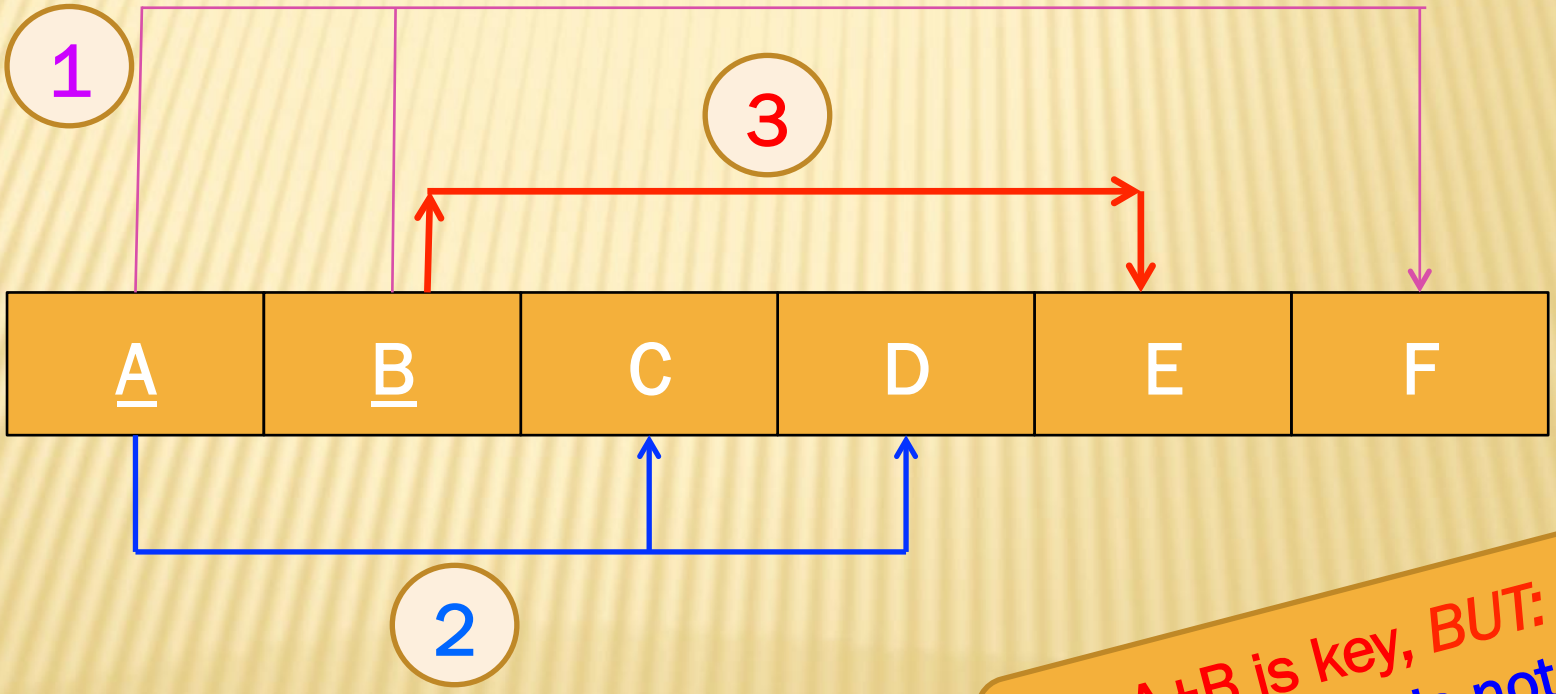
SECOND NORMAL FORM

Exercise 1

1. Create a new relation for each primary key component attribute that is determinant in a partial dependency
   - That attribute is the primary key in the new relation
2. Move the non-key attributes that are dependent on this primary key attribute ("part") from the old relation to the new relation (remove them from the old relation)

SECOND NORMAL FORM

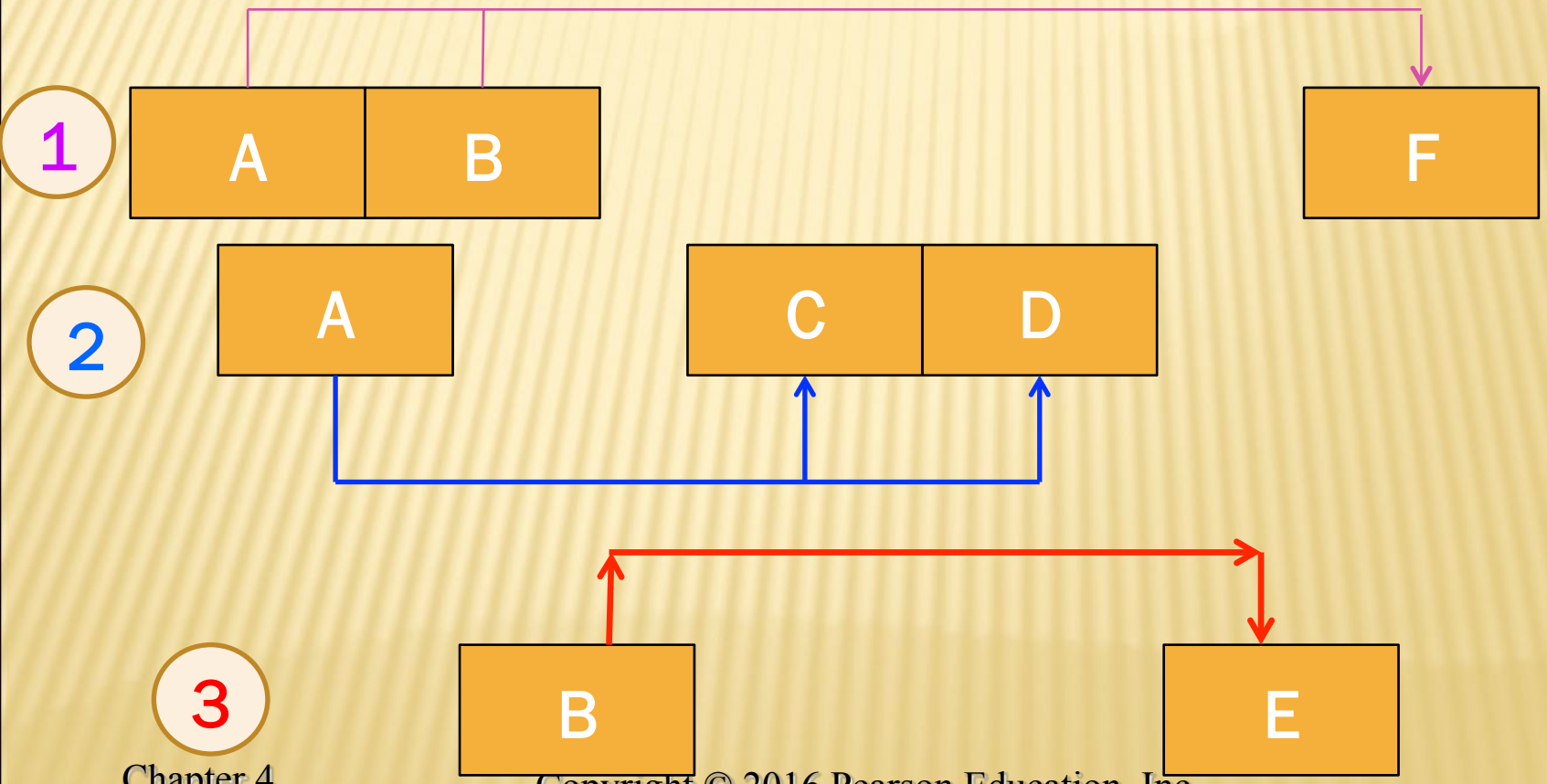
- When A+B is key, then (A+B) *should* ➔ c, d, e, f
- IF (A+B) ➔ f, and A ➔ c,d; B ➔ e - violation



A+B is key, BUT:
A is not key, B is not key;

SECOND NORMAL FORM - SOLUTION

- IF (A+B)➔f, and A➔c,d; B➔e – violation
- Solution: decompose to Relation (1), (2) , and relation (3)



Functional dependency diagram for INVOICE

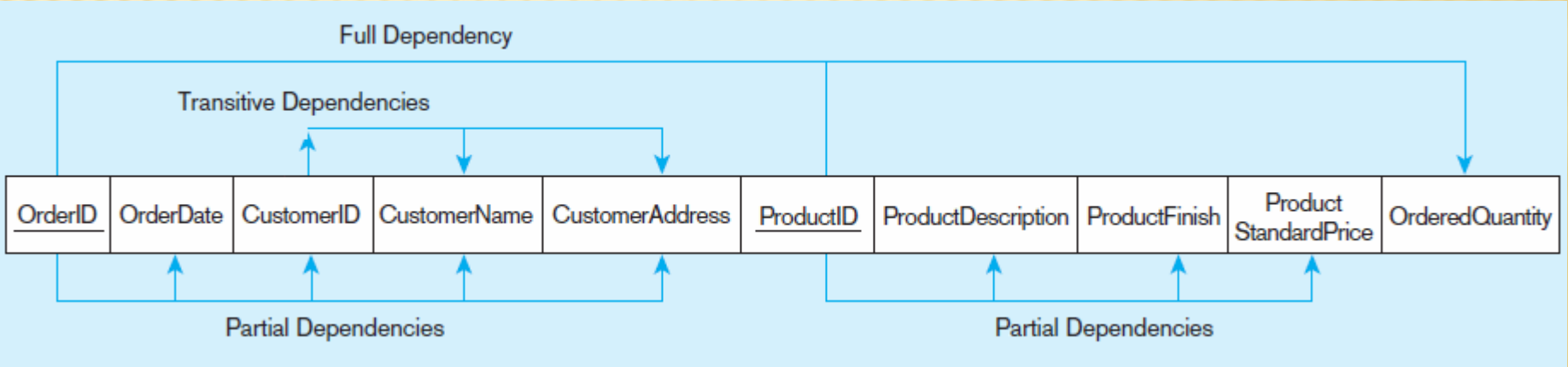
OrderID → OrderDate, CustomerID, CustomerName, CustomerAddress

ProductID → ProductDescription, ProductFinish, ProductStandardPrice

Partial dependency (PD)

OrderID, ProductID → OrderQuantity

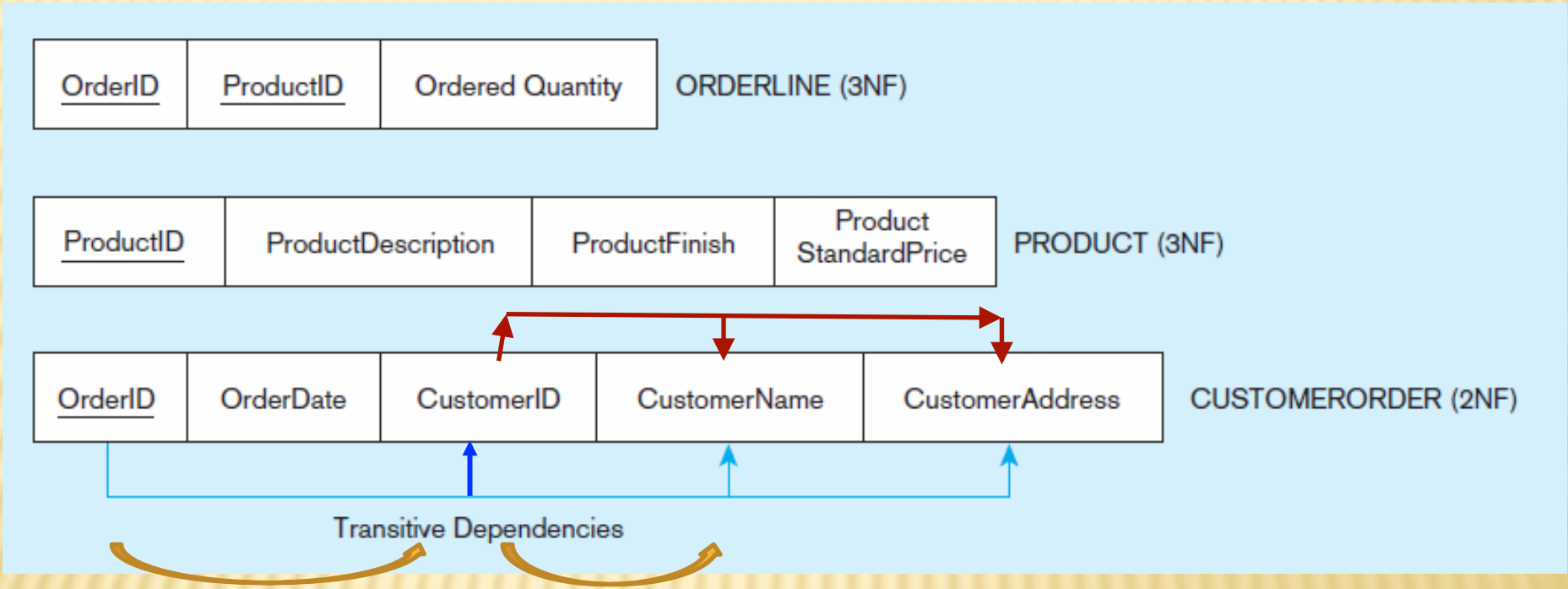
CustomerID → CustomerName, CustomerAddress



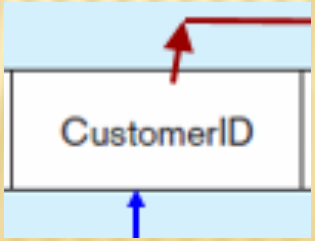
NOT in 2nd Normal Form

Remove PD

Figure 4-28 Removing partial dependencies (PD)



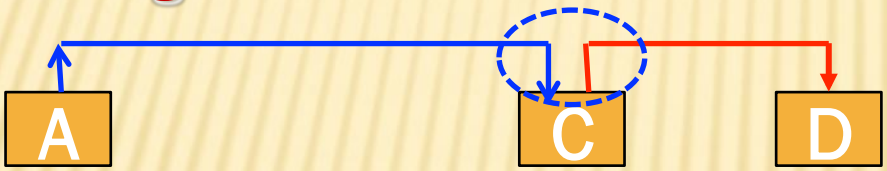
Partial dependencies are removed, but there are still transitive dependencies



"In, and then out" - **Transitive**

THIRD NORMAL FORM

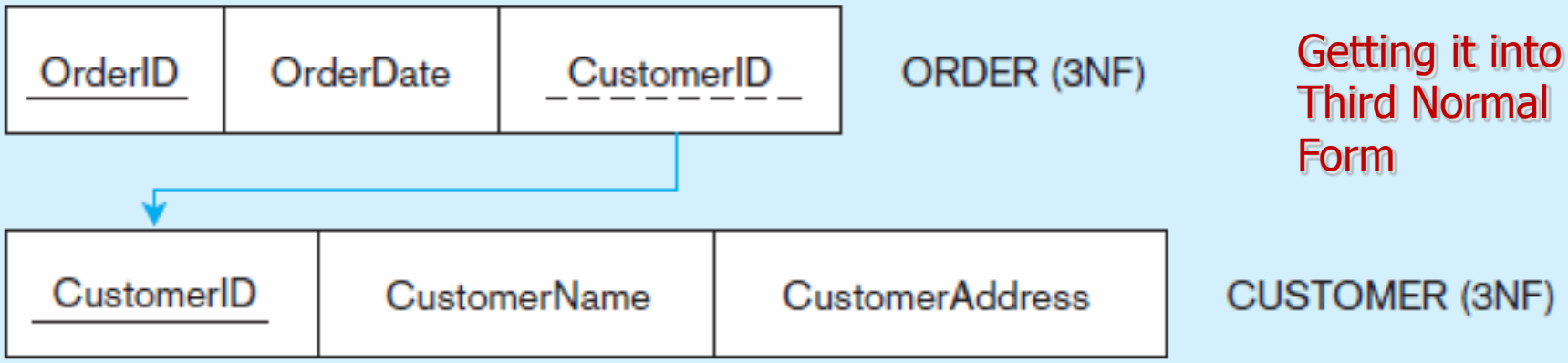
- 2NF PLUS ***no transitive dependencies*** (NO func deps on non-primary-key attributes)
- Note: called transitive because the primary key is a determinant for another attribute, which in turn is a determinant for a third
  - Wrong: A → c and c → d in a relation ("in, and then out")



- Non-key determinant ("c"here) with transitive dependencies go into a new relation; non-key determinant becomes primary key in the new table and stays as foreign key in the old table
  - Dependents are removed from the old relation

Figure 4-29 Removing partial dependencies

Old relation



Getting it into Third Normal Form

New relation

Transitive dependencies are removed

Transitive dependency: A→c→d.
Solution: A→c, and also
c→d , a separate, different relation

To avoid 3 anomalies

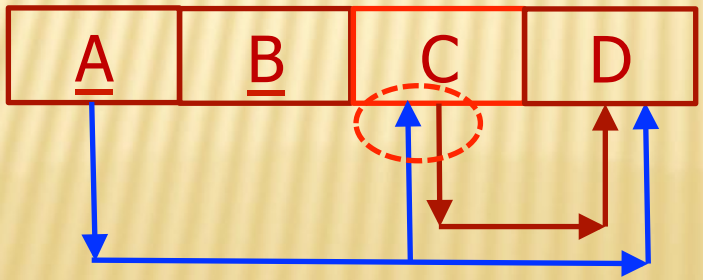
PARTIAL VS TRANSITIVE DEPENDENCY (1)

- Partial: "part" of the key determines others;
  - So only happens when there is a _____ key
  - (A+B) → f; yet also exists B → e
  - Func dep arrows: two arrows out from the composite key should JOINTLY enter all non-key fields; but in violation there are arrows coming out from ONLY ONE PART of the composite key

"Part B of key" determines E: partial

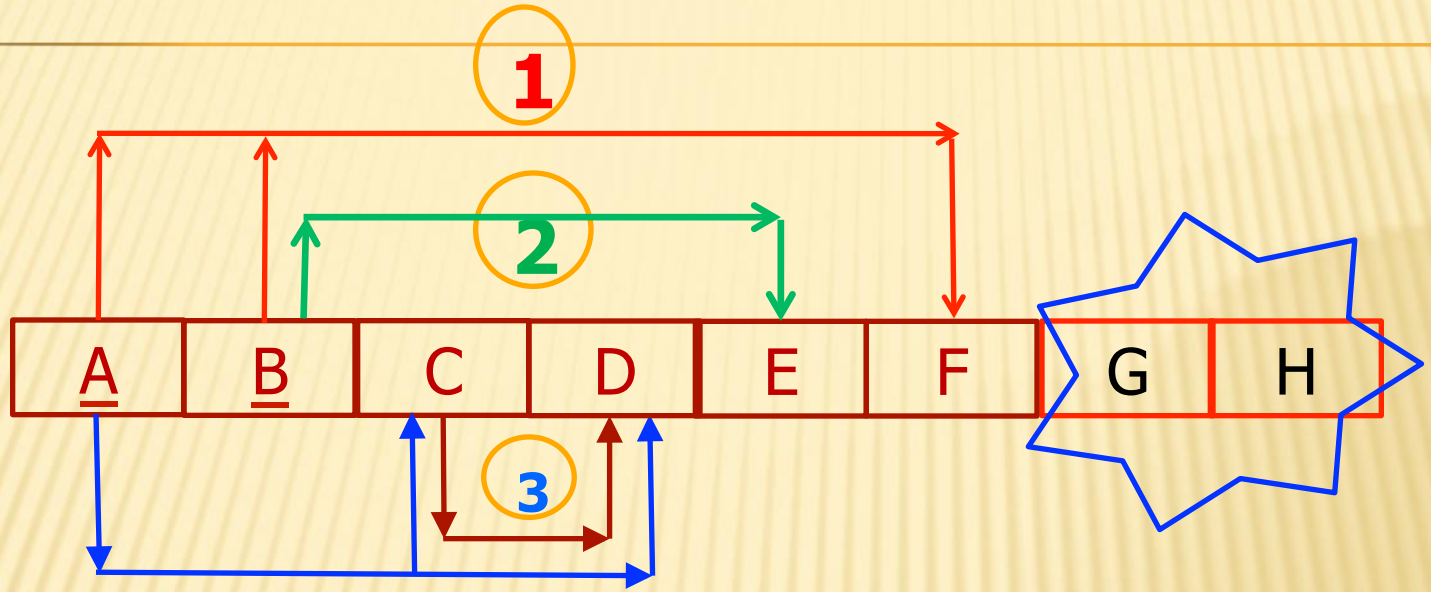
PARTIAL VS TRANSITIVE DEPENDENCY

- Transitive: Key determines non-key-1, and non-key-1 determines non-key-2
  - "You are NOT a key! So you can't determine others!"
  - A (key) → c (non-key-1) → d (non-key-2)
  - Func dep arrows: arrows come out ONLY from the key; but in violation there are arrows also coming out from a non-key attribute



Non-key C determines D: Transitive

PARTIAL VS TRANSITIVE DEPENDENCY

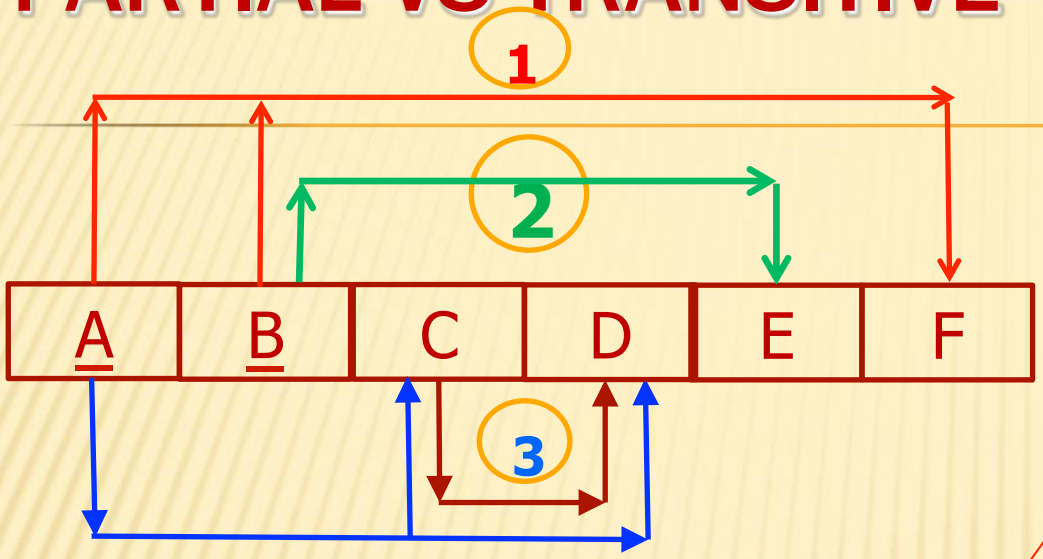


1: _____ dependency?

2: _____ dependency?

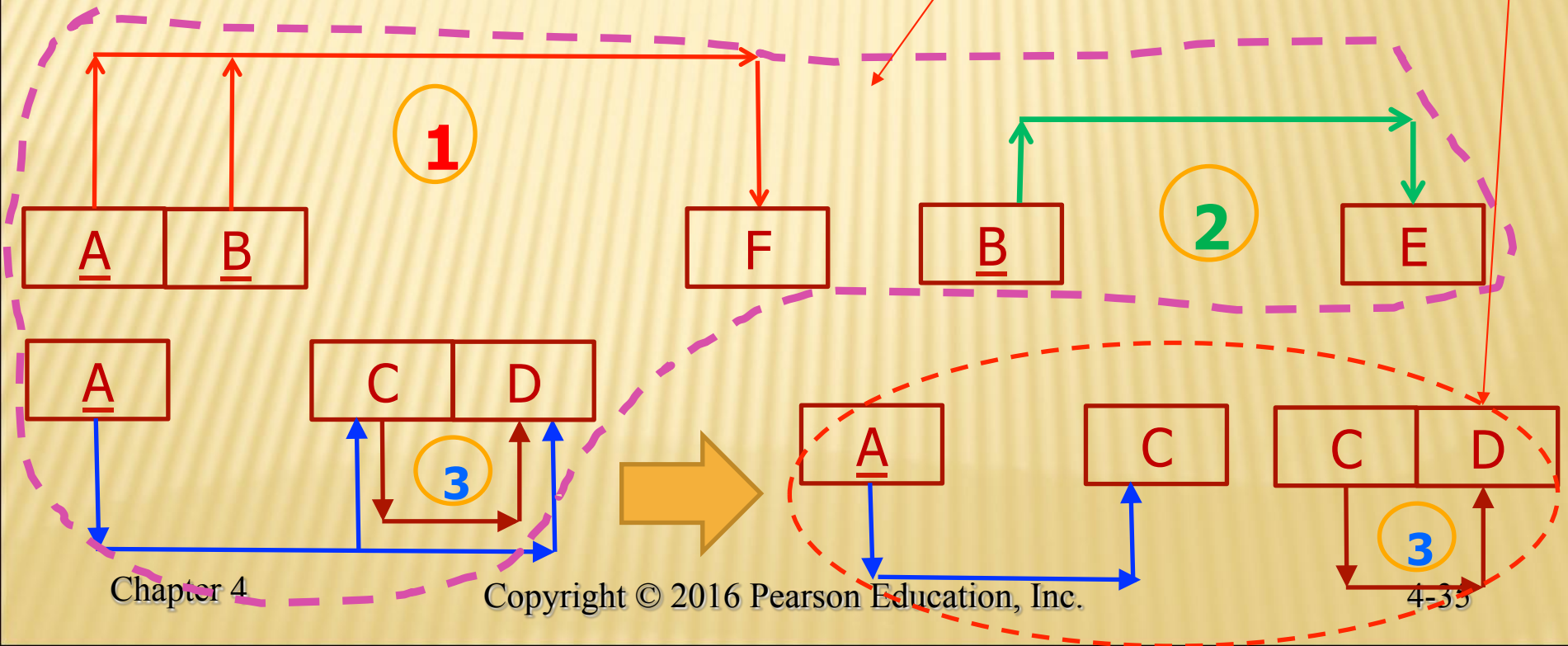
3: _____ dependency?

PARTIAL VS TRANSITIVE - NORMALIZED



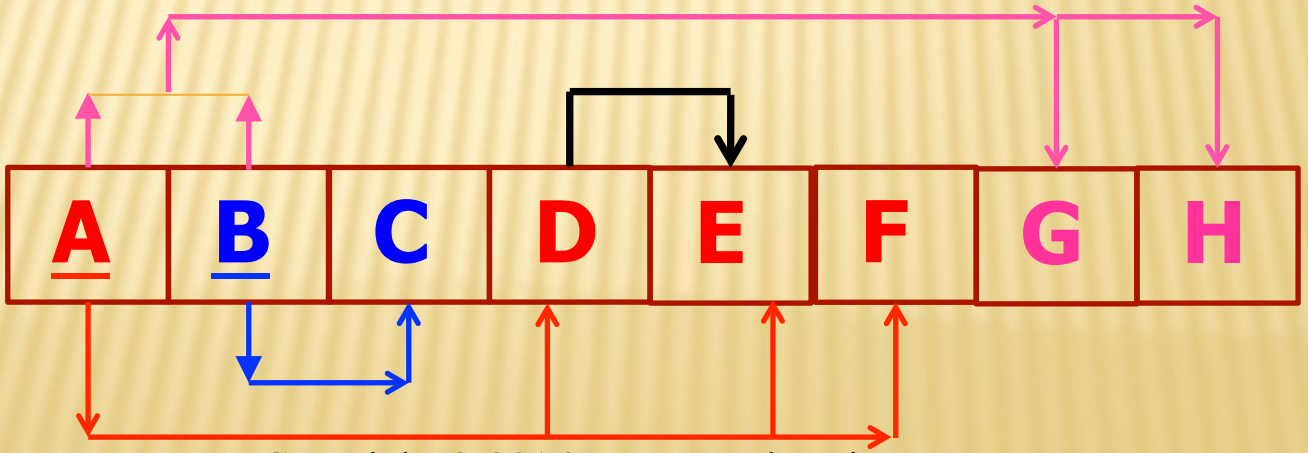
2NF

The central part of 3NF

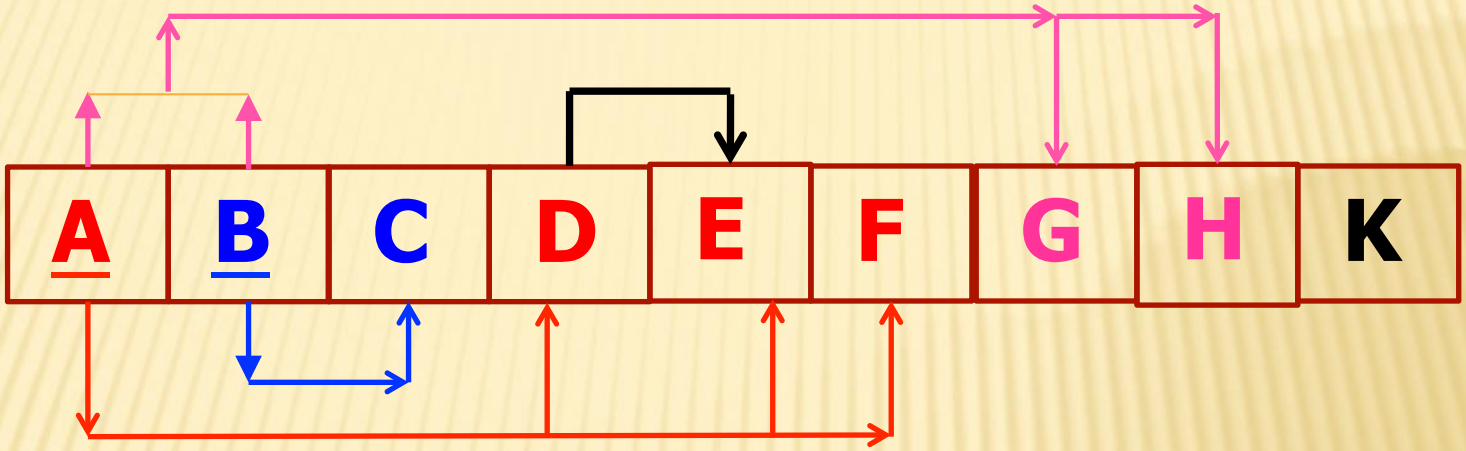


RECAP: IDENTIFY PD AND TD

- 2NF: No partial dependency
  - Implies: composite key
  - Dependencies with the determinant being PART of the composite key ⊘
- 3NF: No transitive dependency
  - Dependencies with the determinant being non-key ⊘

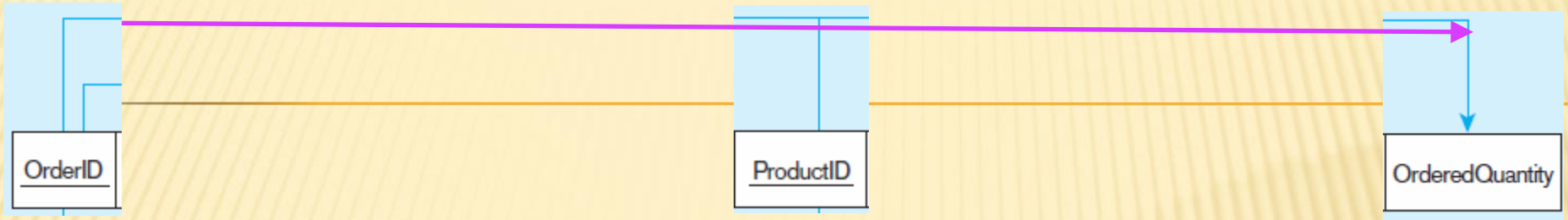


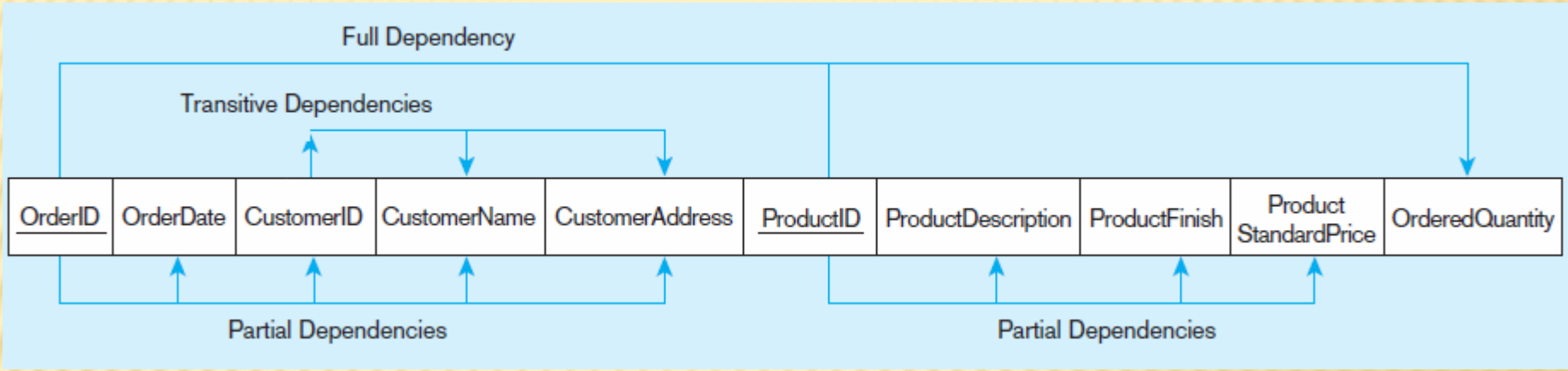
ANATOMY OF FUNCTIONAL DEPENDENCIES (1)

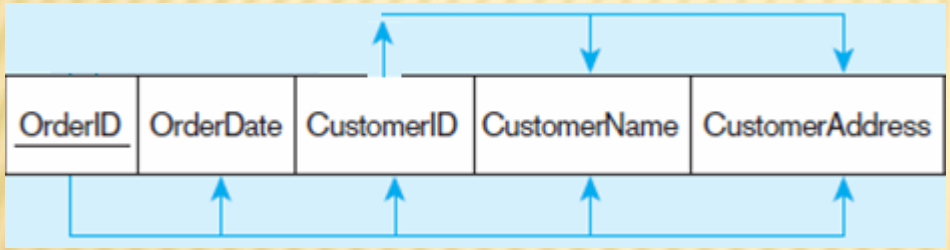


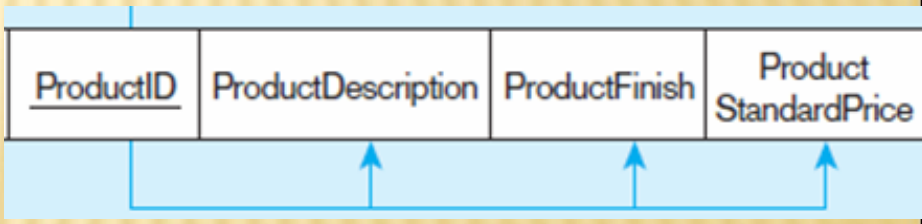
- [A+B] ➔ G, H - ?
- B ➔ C - ?
- A ➔ D, E, F - ?
- D ➔ E - ?

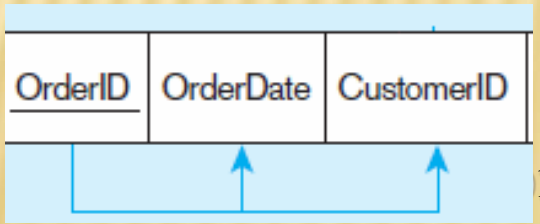
Types?

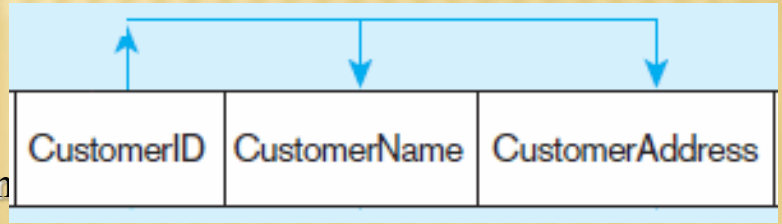

OrderID
ProductID
OrderedQuantity


Full Dependency
Transitive Dependencies
OrderID
OrderDate
CustomerID
CustomerName
CustomerAddress
ProductID
ProductDescription
ProductFinish
Product StandardPrice
OrderedQuantity
Partial Dependencies
Partial Dependencies


OrderID
OrderDate
CustomerID
CustomerName
CustomerAddress


ProductID
ProductDescription
ProductFinish
Product StandardPrice


OrderID
OrderDate
CustomerID


CustomerID
CustomerName
CustomerAddress

MERGING RELATIONS

- View Integration–Combining entities from multiple ER models into common relations
- Issues to watch out for when merging entities from different ER models:
  - Synonyms–two or more attributes with different names but same meaning
  - Homonyms–attributes with same name but different meanings
  - Transitive dependencies–even if relations are in 3NF prior to merging, they may not be after merging
  - Supertype/subtype relationships–may be hidden prior to merging

TO TELL A TD FROM A PD; FD ALWAYS EXISTS

Watch the field where the arrow originates:

- If the arrow goes out from a NON-key field, this func dep is TD;
  - "From non-key to non-key"
- If the arrow goes out from PART of a composite key, this func dep is PD (Partial);
  - "From PART of a key to non-key"
  - PD ONLY exists when/where there is a composite key!
- There ALWAYS exists a Full Dependency
  - "From key to non-key"