Dalhousie University
Faculty of Computer Science

CSCI 2141 – Intro to Database Systems

Database Normalization

Disclaimer

- Some slides and images are taken from the following textbook for CSCI 2141:

"Database Systems – Design, Implementation and Management", 12e, by Coronel and Morris

Data Redundancy

- Unnecessarily storing same data at different places
- **Islands of information**: Scattered data locations
  - Increases the probability of having different versions of the same data

Data Redundancy Implications

- Poor data security
- Data inconsistency
- Increased likelihood of data-entry errors when complex entries are made in different files
- **Data anomaly**: Develops when not all of the required changes in the redundant data are made successfully

Types of Data Anomaly

- Update Anomalies
- Insertion Anomalies
- Deletion Anomalies

Normalization

- Normalization is a process for evaluating and correcting table structures to minimize data redundancies
  - Reduces the likelihood of data anomalies
- Involves assigning attributes to tables based on the concept of **determination**
- Normalization works through a series of stages called normal forms
  - Higher normal forms are better from a structural point of view
  - For most purposes in business database design, 3NF is sufficient
  - Highest level of normalization may not be the most desirable

Normalization

- Generally, the higher the normal form:
  - Greater is the number of tables
    - Hence, greater is the number of relational join operations you need to perform for a specified output
    - More resources are required by the database system to respond to end-user queries
  - For a successful design meeting performance requirements, some portions of the database may need to be denormalized
    - Denormalization is often performed after normalization
    - 3NF is normally converted back to a 2NF after normalization

When to Normalize?

- Generally used in two situations
  - When designing a new database structure based on the business requirements of a user, the DB designer
    - Constructs a data model (ERD)
    - Uses normalization to analyze the relationship among the attributes (columns) within each entity (table)
  - When modifying existing data structures that are in the form of flat files, spreadsheets or older database structures
    - Designer uses the normalization process to analyze the relationships and improve the database design

Back to Normalization

- The Normal Forms

TABLE 6.2

NORMAL FORMS

NORMAL FORM	CHARACTERISTIC	SECTION
First normal form (1NF)	Table format, no repeating groups, and PK identified	6.3.1
Second normal form (2NF)	1NF and no partial dependencies	6.3.2
Third normal form (3NF)	2NF and no transitive dependencies	6.3.3
Boyce-Codd normal form (BCNF)	Every determinant is a candidate key (special case of 3NF)	6.6.1
Fourth normal form (4NF)	3NF and no independent multivalued dependencies	6.6.2

Converting to 1NF

- Table format, no repeating groups and PK identified
- Consider the data given below:

Student ID	Name	GPA	Course Code	Course Title	Course Grade	Course Credits
1	Alice	3.5	CS100, CS101, MT112	HTML, Java, Maths	3.5, 3.0, 4.0	4, 4, 3
2	Bob	2.2	CS100, MT112	HTMI, Maths	2.5, 2.0	4, 3
3	Charles	3.2	CS100, MT112	HTML, Maths	3.0, 3.5	4, 4
4	David	3.0	CS101	Java	3.5	4
5	Emily	2.9	HM110	History	3.25	3

Converting to 1NF – Step 1

- Eliminate the repeating groups

Student ID	Name	GPA	Course Code	Course Title	Course Grade	Course Credits
1	Alice	3.5	CS100	HTML	3.5	4
1	Alice	3.5	CS101	Java	3.0	4
1	Alice	3.5	MT112	Maths	4.0	3
2	Bob	2.2	CS100	HTML	2.5	4
2	Bob	2.2	MT112	Maths	2.0	3
3	Charles	3.2	CS100	HTML	3.0	4
3	Charles	3.2	MT112	Maths	3.5	3
4	David	3.0	CS101	Java	3.5	4
5	Emily	2.9	HM110	History	3.25	3

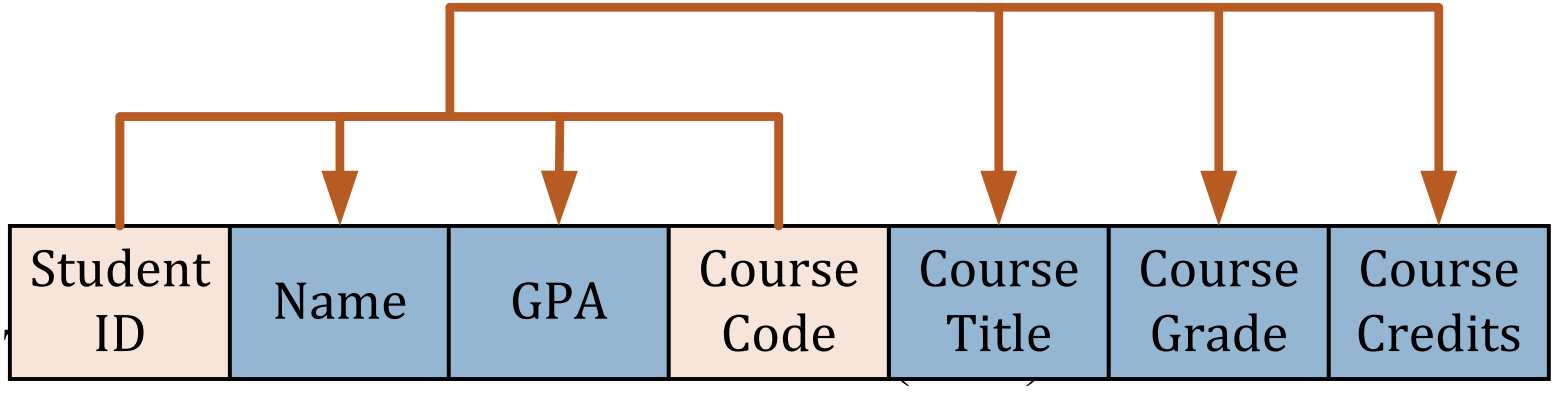
Converting to 1NF – Step 2

- Primary Key?
  - Identify the Super Keys
    - Any combination containing Student ID + Course Code
  - Identify the Candidate Key(s)
    - Student ID + Course Code
  - Identify the Primary Key (PK)

Student ID	Name	GPA	Course Code	Course Title	Course Grade	Course Credits

Converting to 1NF – Step 3

- Identify dependencies
  - All attributes must be functionally dependent on the primary key
  - (Student ID, Course Code) → (Name, GPA, Course Title, Course Grade, Course Credits)
  -

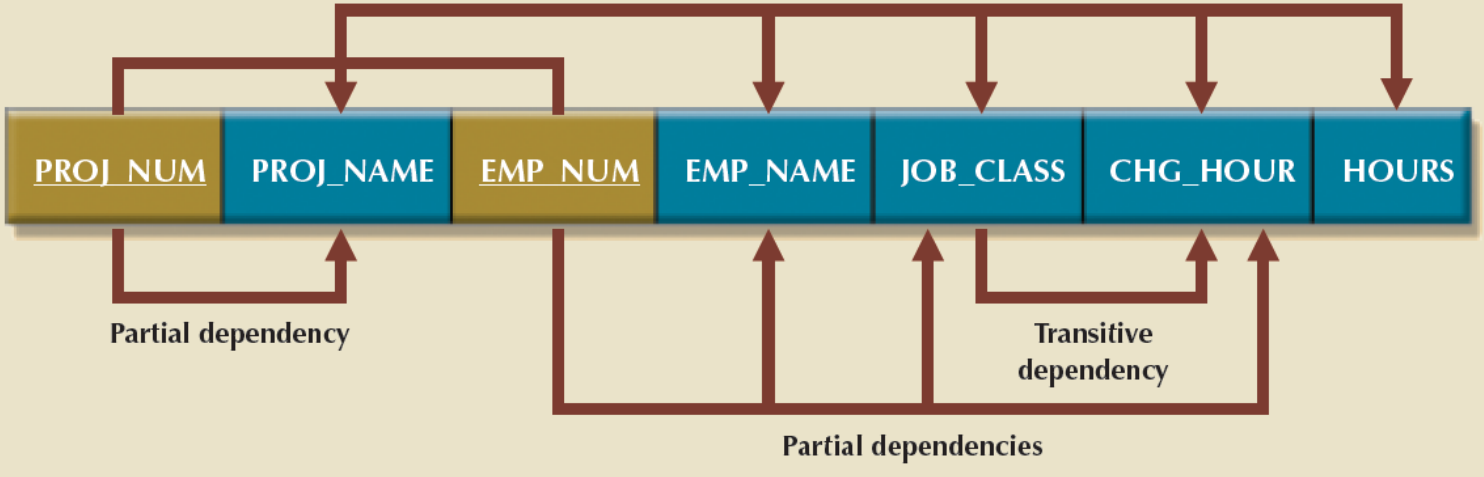


Dependencies

- **Partial dependency**: An attribute depends on only a part of the primary key
  - Only possible when the primary key is composite
  - Straight forward and easy to identify
- **Transitive dependency**: An attribute functionally depends on another non-key attribute
- **Dependency Diagram**: A diagram showing all dependencies
  - Full functional dependencies shown using arrows above the attributes
  - Partial and transitive dependencies shown below the attributes

Dependency Diagram

FIGURE 6.3 FIRST NORMAL FORM (1NF) DEPENDENCY DIAGRAM



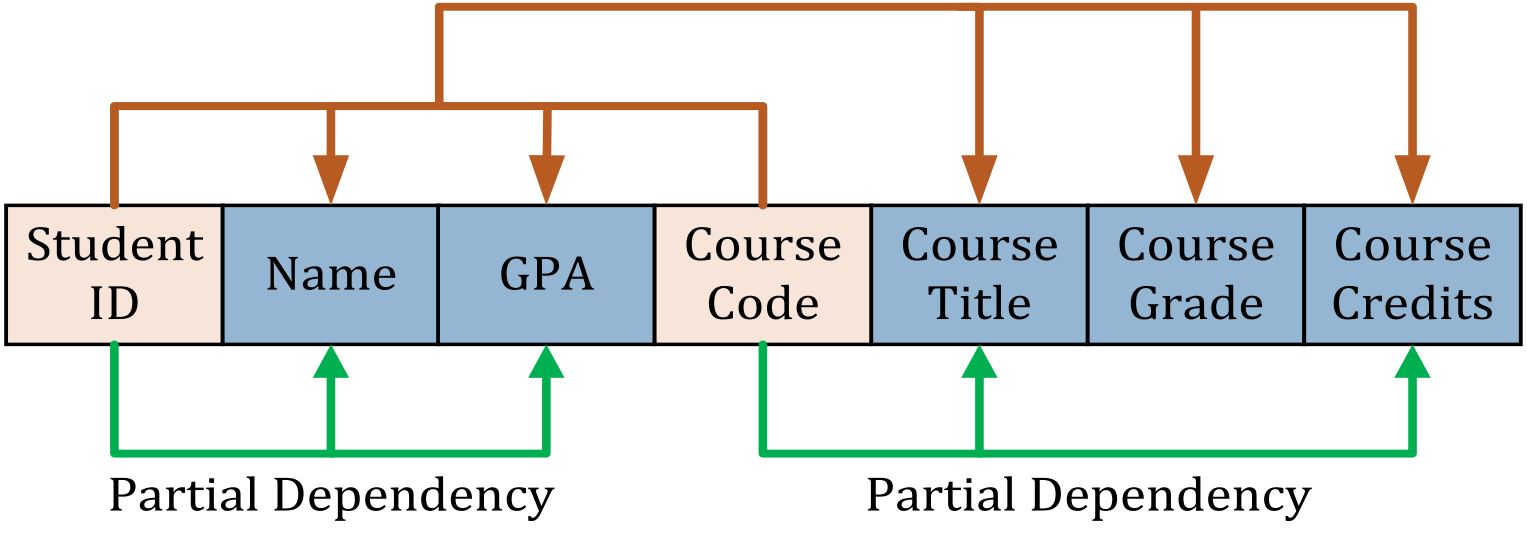
1NF (PROJ_NUM, EMP_NUM, PROJ_NAME, EMP_NAME, JOB_CLASS, CHG_HOURS, HOURS)

PARTIAL DEPENDENCIES:
(PROJ_NUM → PROJ_NAME)
(EMP_NUM → EMP_NAME, JOB_CLASS, CHG_HOUR)

TRANSITIVE DEPENDENCY:
(JOB_CLASS → CHG_HOUR)

Partial Dependency

- Partial dependency exists when there is a composite primary key
  - One or more attributes may depend on part of the primary key

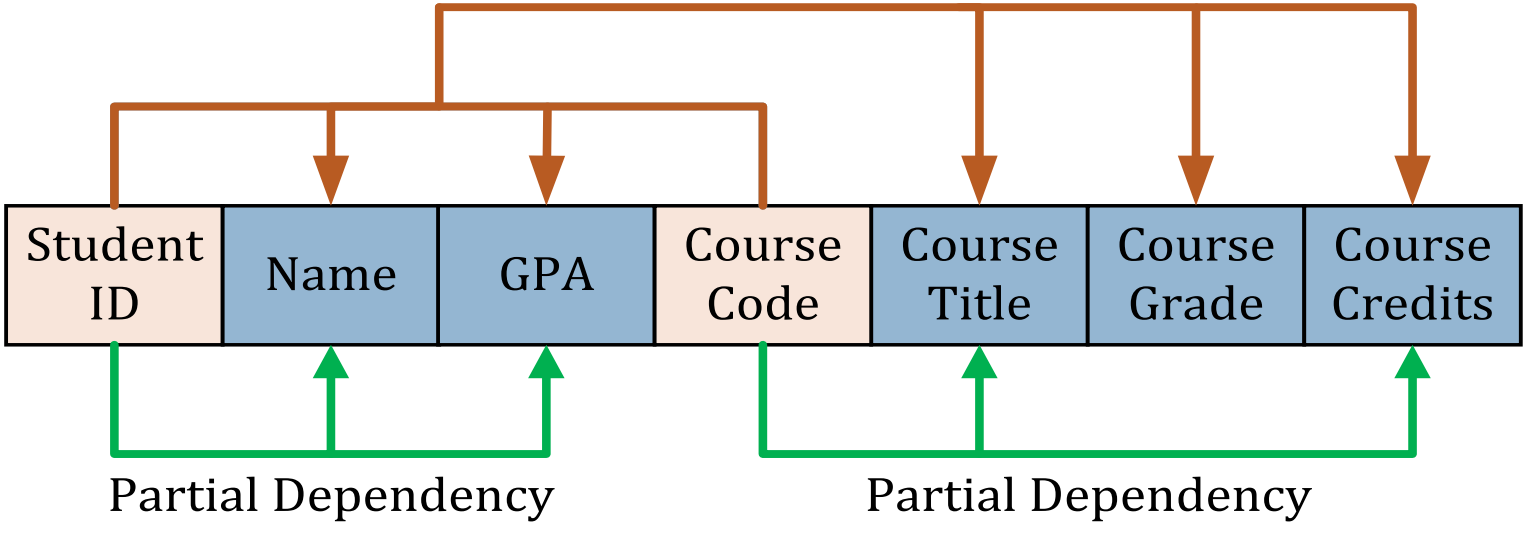


Converting to 2NF

- If 1NF has a single-attribute PK, table is automatically in 2NF
- If table is in 1NF and has a composite PK
  - Identify partial dependencies
  - For each attribute of the PK that acts as a determinant in a partial dependency, create a new table
    - Make the determinant attribute the PK of the new table
    - Reassign corresponding dependent attributes

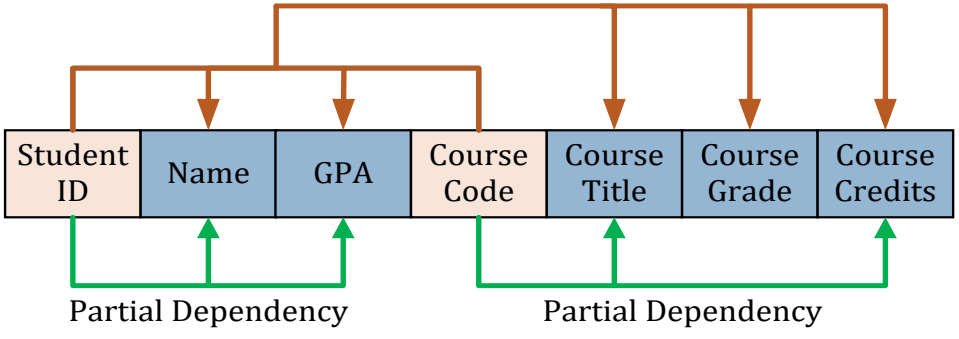
Converting to 2NF – Step 1

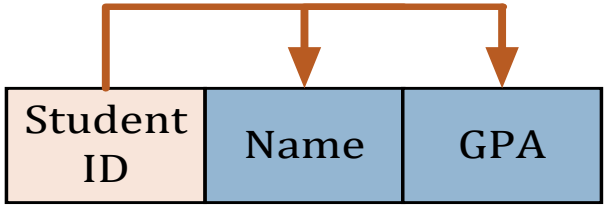
- Identify partial dependencies

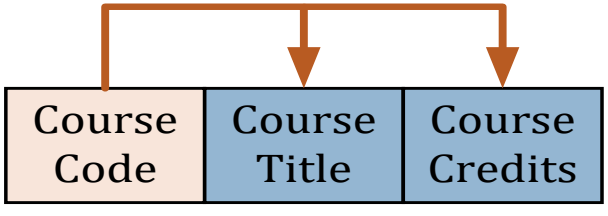


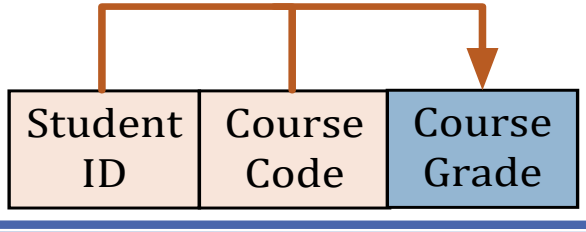
Converting to 2NF – Step 2

- Create new tables for each determinant of a partial dependency

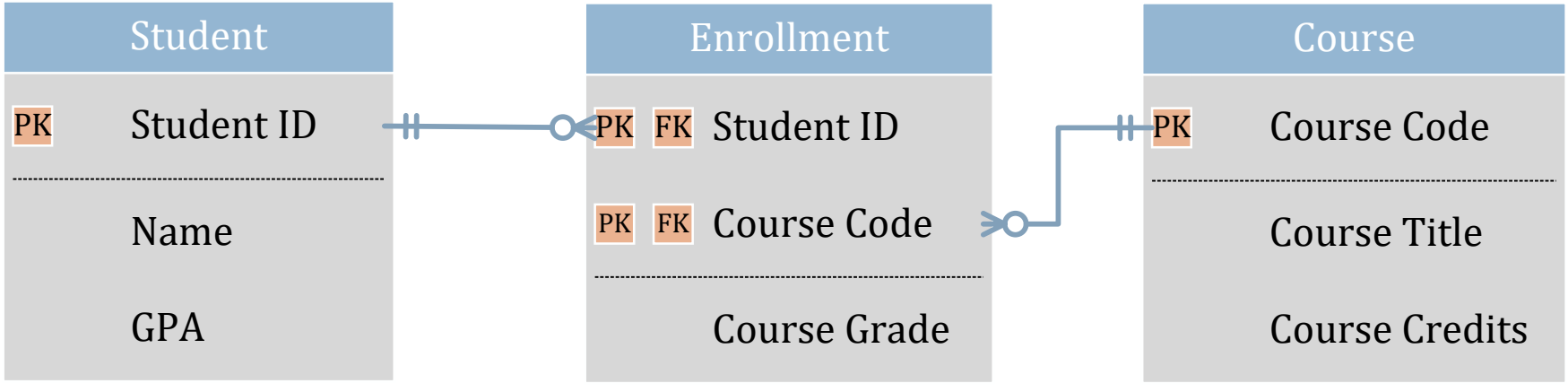






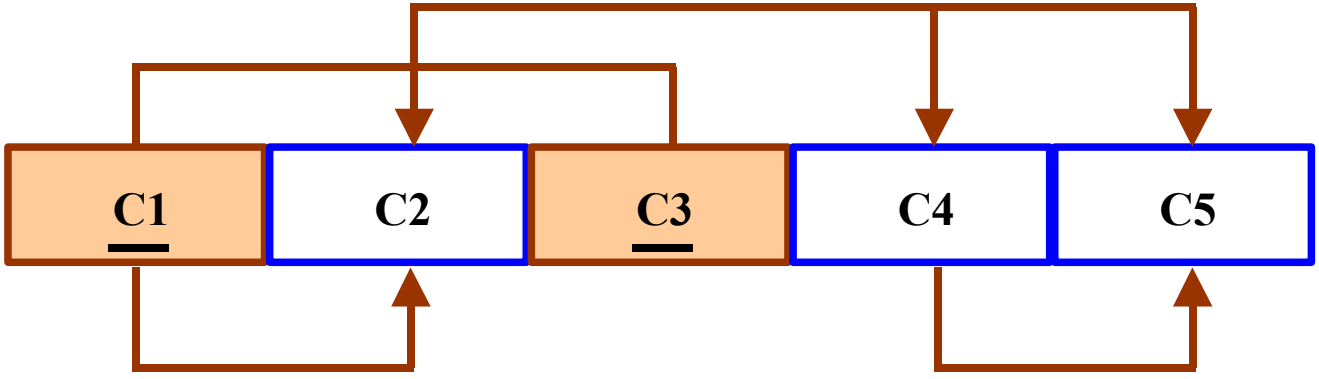


Tables in 2NF



Example

- For the dependency diagram shown below:
  - Identify dependencies and their types
  - Create a database whose tables are in 2NF



Solution – Dependencies

- Functional dependency on PK:

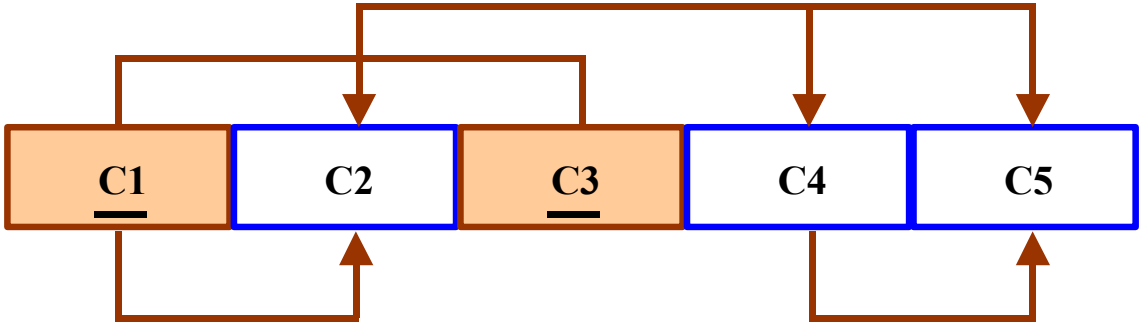
  (C1, C3) → (C2, C4, C5)

- Partial dependency:

  C1 → C2

- Transitive dependency:

  C4 → C5



Solution – Database in 2NF

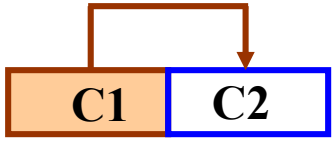


Table 1

Primary key: C1
Foreign key: None
Normal form: At least 2NF

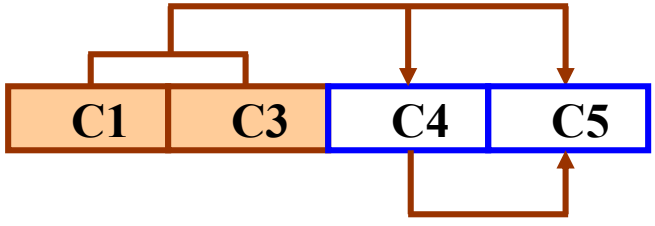


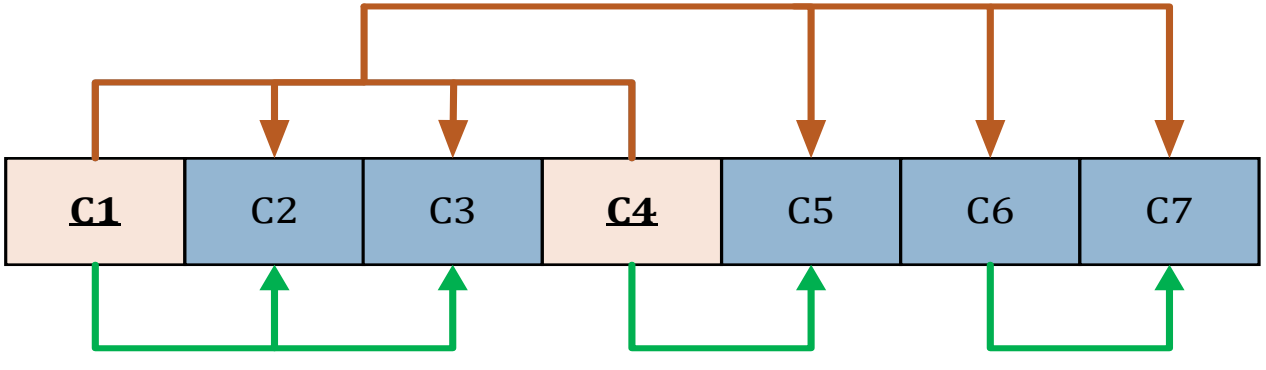
Table 2

Primary key: C1 + C3
Foreign key: C1 (to Table 1)
Normal form: 2NF
Transitive dependency C4 → C5

Exercise 1

a) For the dependency diagram shown below, identify all dependencies and indicate their types

b) Create a database whose tables are in 2NF



Exercise 1 – Solution

- Functional dependency on PK:

  (C1, C4) → (C2, C3, C5, C6, C7)

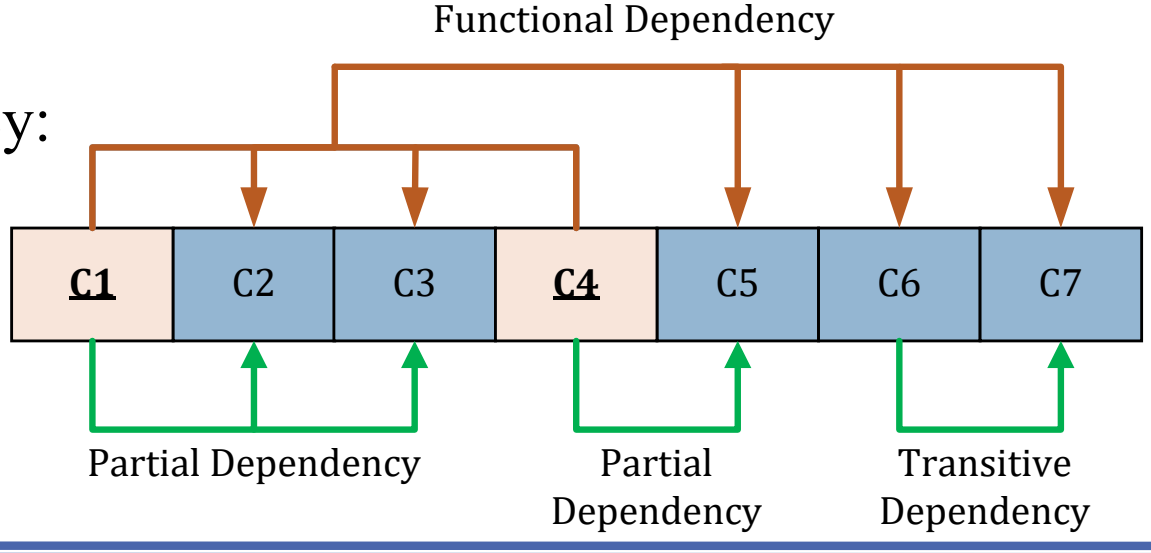
- Partial dependencies:

  C1 → (C2, C3)

  C4 → C5

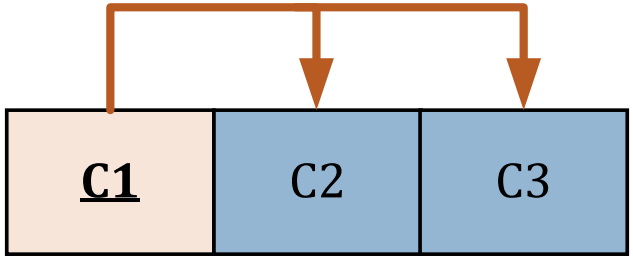
- Transitive dependency:

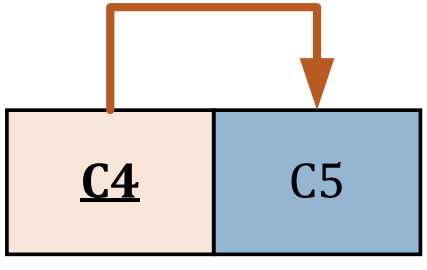
  C6 → C7

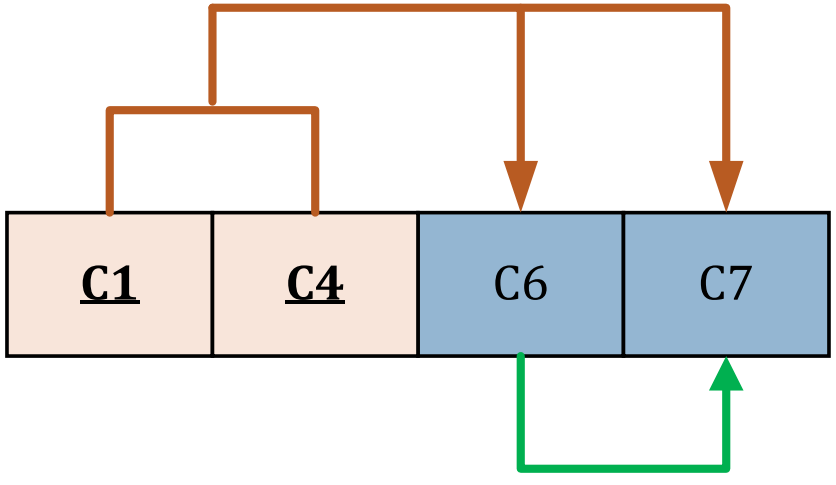


Exercise 1 – Solution

- Tables in 2NF







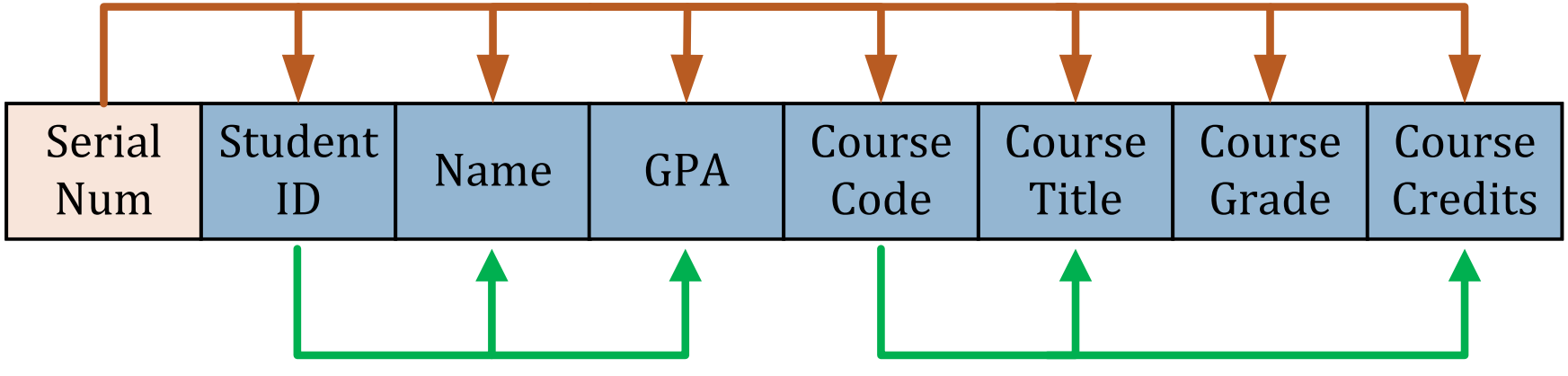
Re-visiting 2NF

- Previous examples had no additional candidate keys (other than the primary key)
- Given the table below:
  - Identify its candidate keys (if any)
  - Identify dependencies
  - What normal form is the table in?

Serial Num	Student ID	Name	GPA	Course Code	Course Title	Course Grade	Course Credits

Re-visiting 2NF

- (Student ID + Course Code) is a candidate key
- Dependencies are shown below:
- Are these dependencies partial or transitive?
  - Answer determines the NF



- Dependencies are partial as they depend on part of a candidate key, hence table is in 1NF

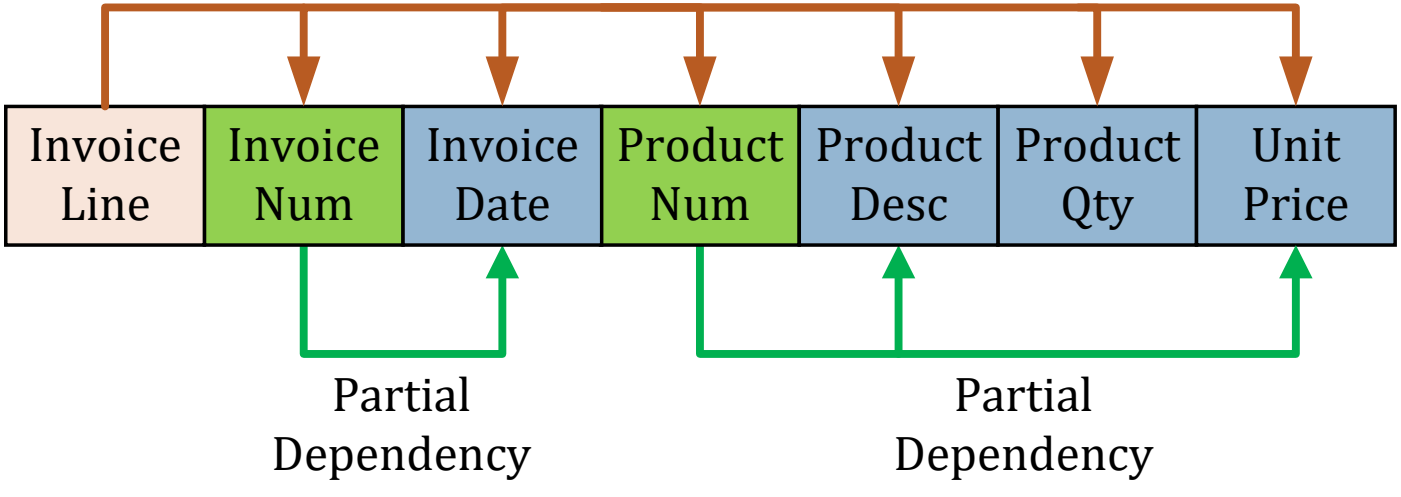
Practice Exercise 1

- Identify the normal form of the tables shown below
- If not already in 2NF, convert to 2NF

Invoice Line	Invoice Num	Product Num	Invoice Date	Product Desc	Product Qty	Unit Price

Practice Exercise 1

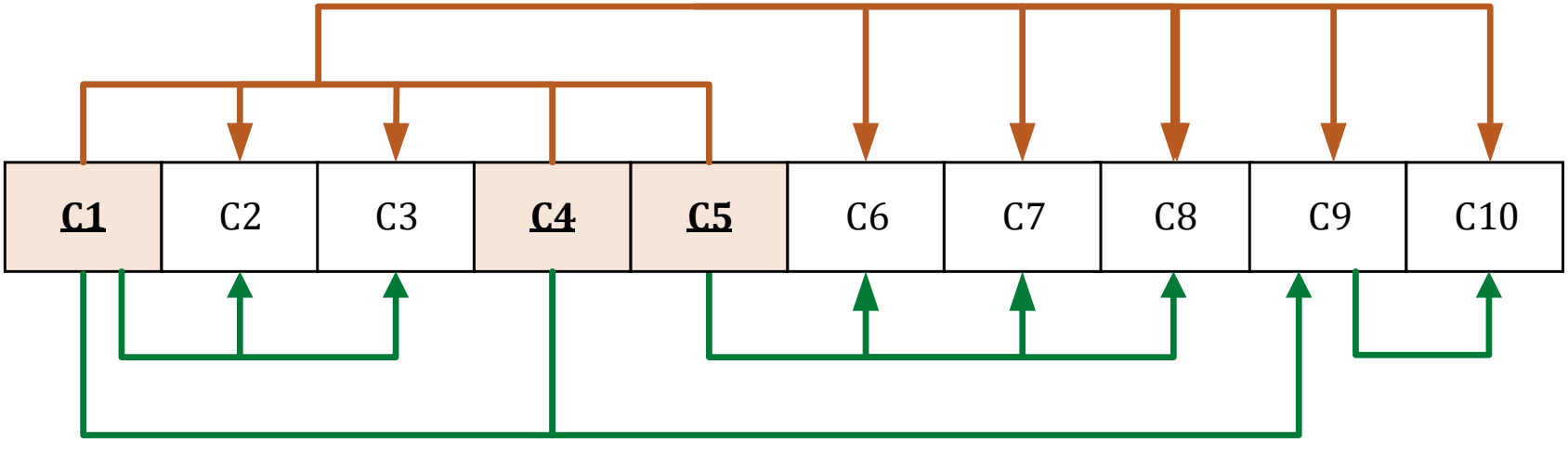
- Dependency diagram is shown below:



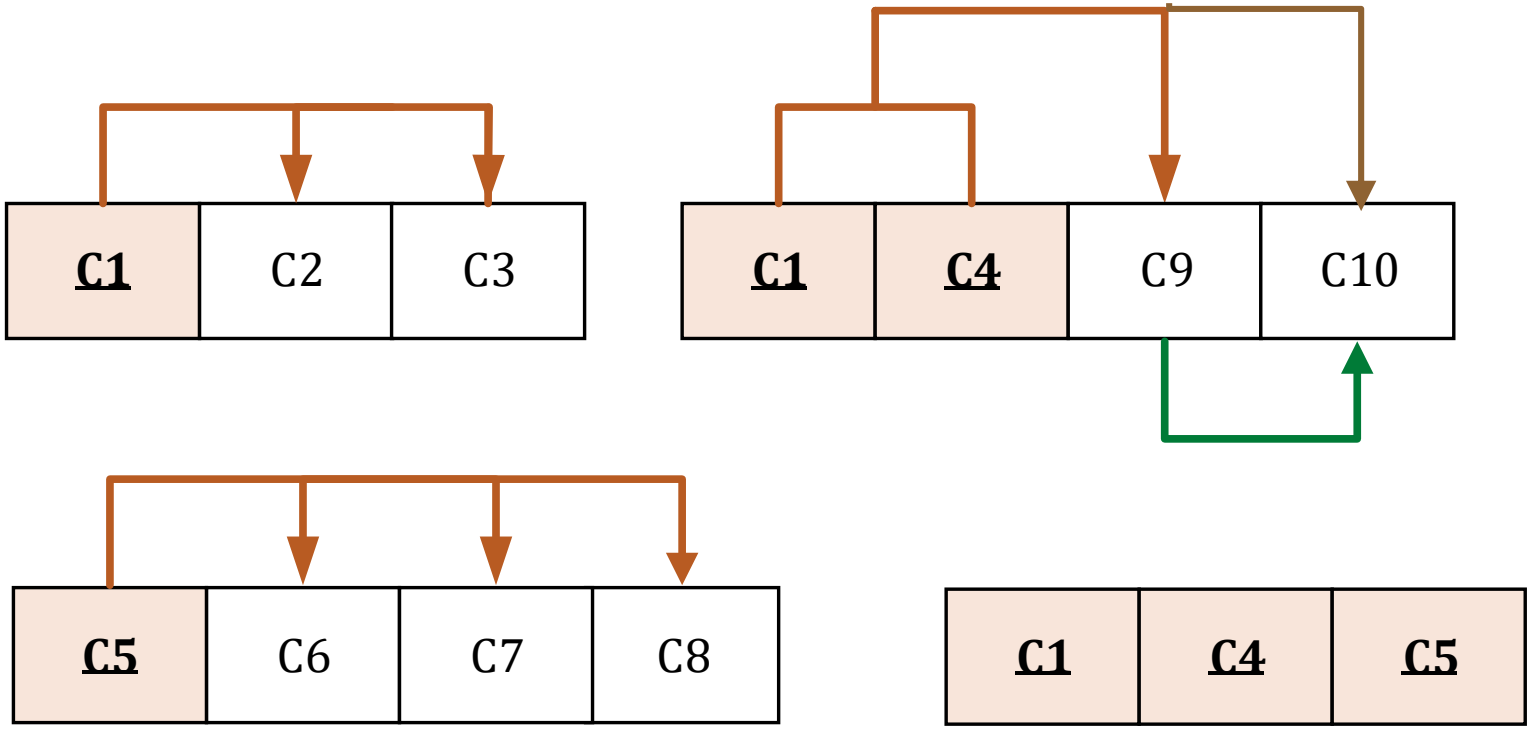
Practice Exercise 2

a) For the dependency diagram shown below, identify all dependencies and indicate their types

b) Create a database whose tables are at least in 2NF



Practice Exercise 2 – Solution



Converting to 3NF

- Tables are in 3NF (Third Normal Form) when
  - They are in 2NF, and
  - They have no transitive dependencies
    - **Transitive dependency**: An attribute functionally depends on another non-key attribute
- Converting to 3NF
  - Ensure that the tables are in 2NF
  - Make new tables to eliminate transitive dependencies
  - Reassign corresponding dependent attributes

Example

a) Identify all dependencies.

b) Identify which normal form the table is in.

c) Convert to the third normal form (3NF).

INV_NUM	INV_DATE	INV_AMOUNT	CUS_NUM	CUS_ADDRESS	CUS_PHONE

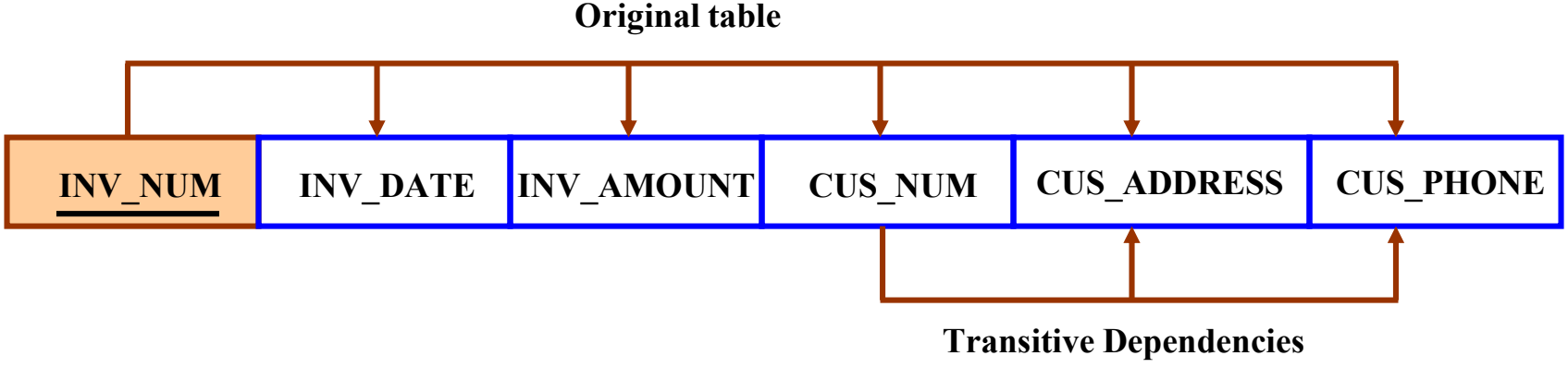
Example – Solution

a) Dependencies are shown in figure below.

b) Table is in 1NF as all attributes show functional dependency on the PK

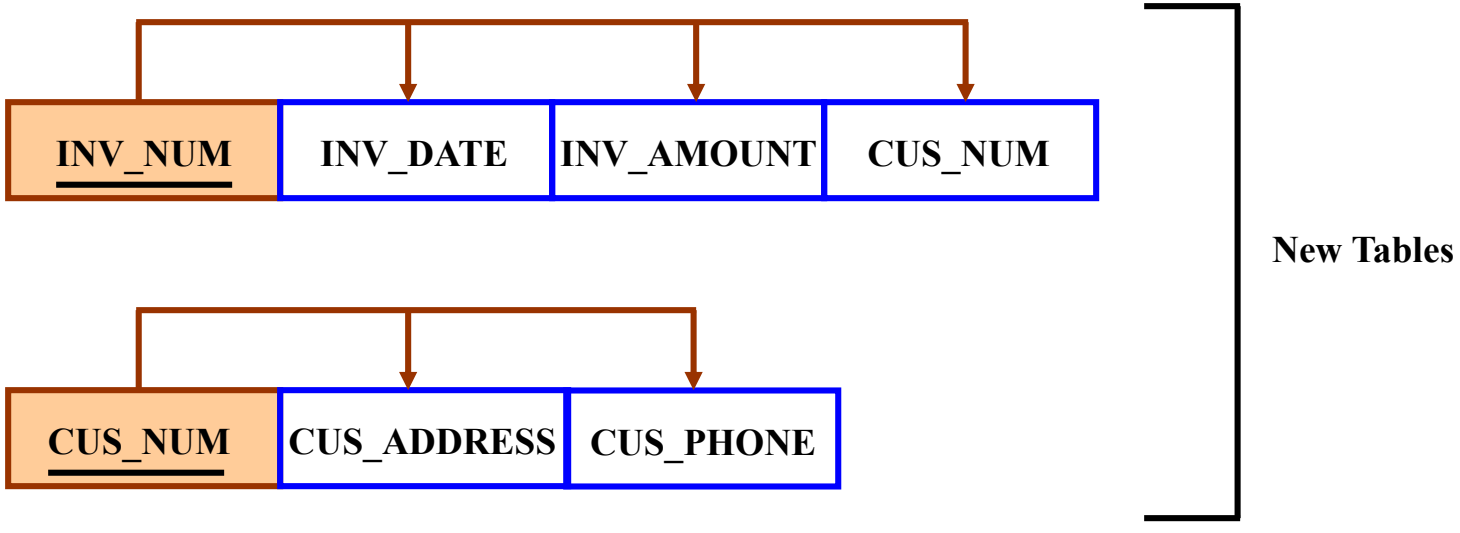
Table is also in 2NF as the PK is not composite, and there is no candidate key.

- There are no partial dependencies



Example – Solution

- Eliminating transitive dependencies result in the following tables



- Each of the table above is in 3NF

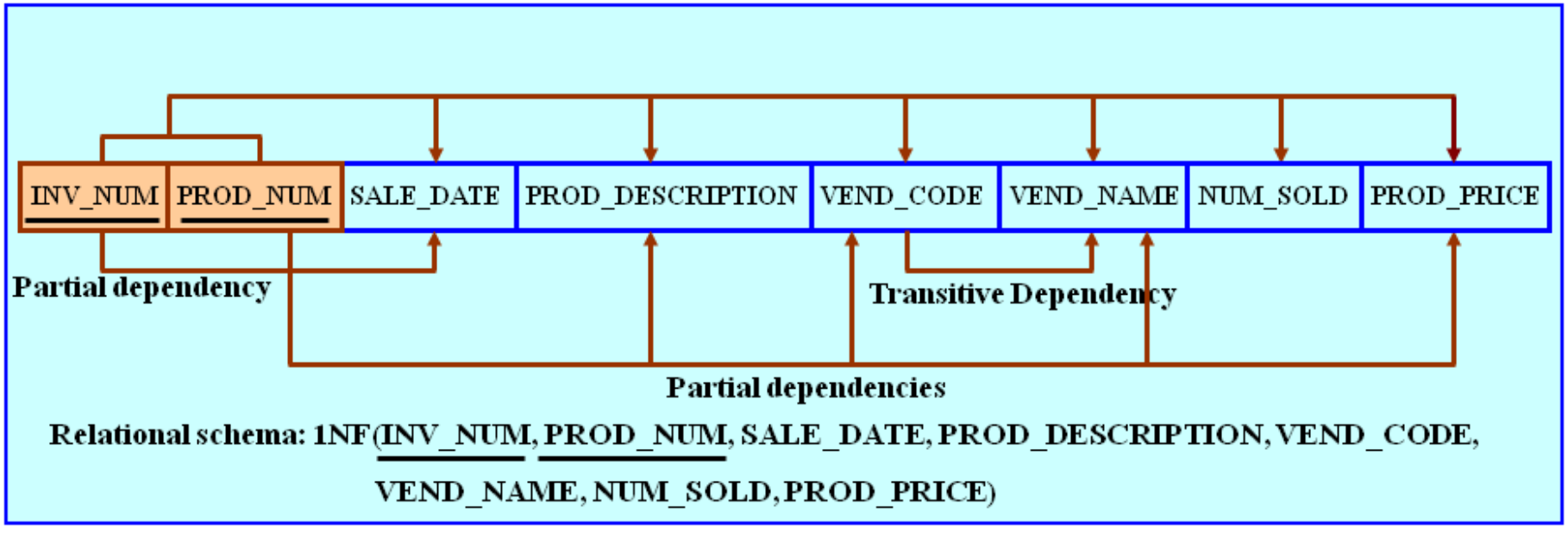
Practice Exercise 5

- Using the INVOICE table structure shown below:
  - Write the relational schema, and draw its dependency diagram identifying all dependencies (Assume there are no repeating groups and invoice number references more than one product)
  - Remove all partial dependencies, write the relational schema, and draw the new dependency diagrams. Identify the normal forms for each table structure you create
  - Remove all transitive dependencies, write the relational schema, and draw the new dependency diagrams. Also, draw the ERD

Attribute Name	Sample Value	Sample Value	Sample Value	Sample Value	Sample Value
INV_NUM	211347	211347	211347	211348	211349
PROD_NUM	AA-E3422QW	QD-300932X	RU-995748G	AA-E3422QW	GH-778345P
SALE_DATE	15-Jan-2016	15-Jan-2016	15-Jan-2016	15-Jan-2016	16-Jan-2016
PROD_LABEL	Rotary sander	0.25-in. drill bit	Band saw	Rotary sander	Power drill
VEND_CODE	211	211	309	211	157
VEND_NAME	NeverFail, Inc.	NeverFail, Inc.	BeGood, Inc.	NeverFail, Inc.	ToughGo, Inc.
QUANT_SOLD	1	8	1	2	1
PROD_PRICE	$49.95	$3.45	$39.99	$49.95	$87.75

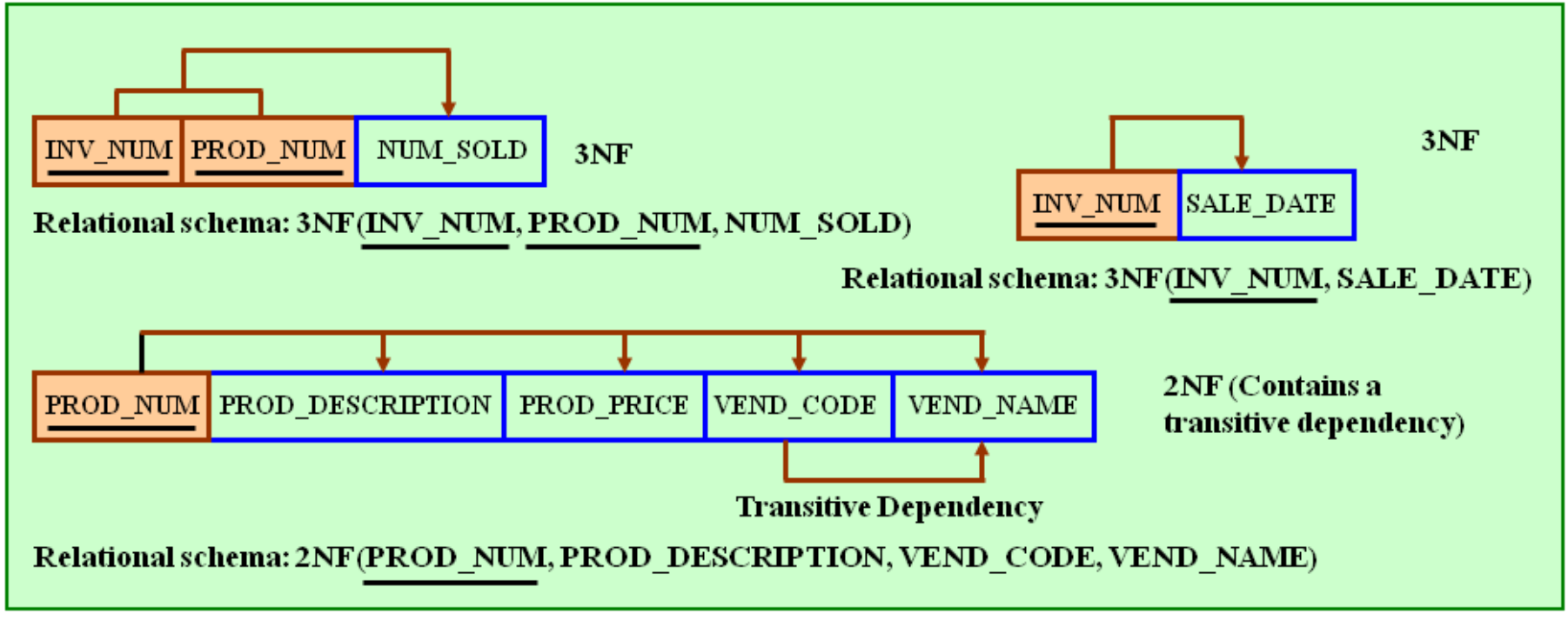
Practice Exercise 5 – Solution

- Relational schema and dependency diagram are as shown



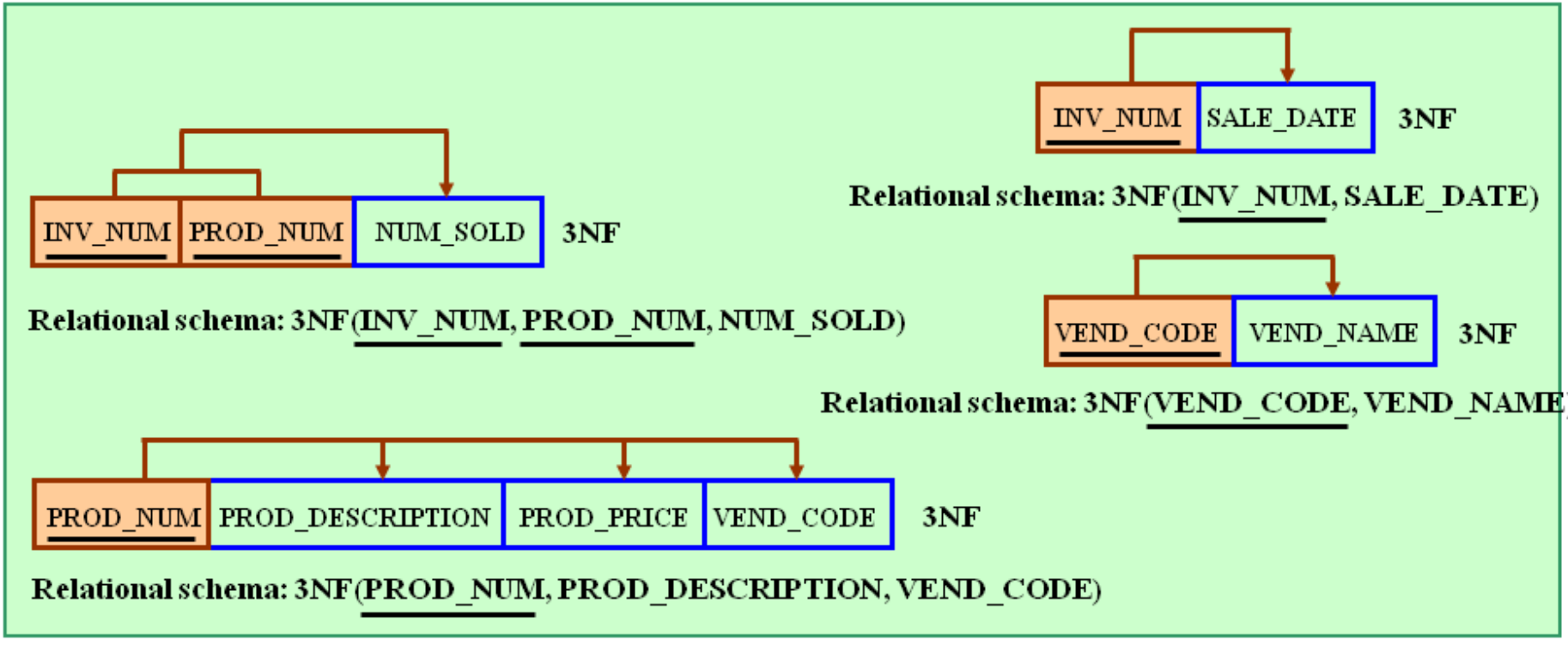
Practice Exercise 5 – Solution

- Relational schema, and tables with partial dependencies removed (at least 2NF) are as shown below:



Practice Exercise 5 – Solution

- Relational schema, and tables with transitive dependencies removed (3NF) are as shown below:



Practice Exercise 5 – Solution

- ERD is shown below: