Chapter 6:
LOGICAL DATABASE DESIGN

NORMALIZATION (PART 1)

Well-Structured Relations

- A relation that contains minimal data redundancy and allows users to insert, delete, and update rows without causing data inconsistencies.

General rule of thumb: a table should not pertain to more than one entity type

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.

Example – Figure 6-2

EMPLOYEE2

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/200X
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/200X
140	Alain Beeton	Accounting	52,000	Tax Acc	12/8/200X
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/200X
110	Chris Lucero	Info Systems	43,000	C++	4/22/200X
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/200X
150	Susan Martin	Marketing	42,000	Java	8/12/200X

Question – Is this a relation?

Answer – Yes: unique rows and no multivalued attributes

Question – What's the primary key?

Answer – Composite: Emp_ID, Course_Title

Example of Unnormalized relation

- EMPLOYEE 2 (Unnormalized relation)
- Note Emp_ID and Course_Title make a composite primary key

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/1998
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/1998
140	Alain Beeton	Accounting	52,000	Tax Acc	12/8/1996
110	Chris Lucero	Info Systems	43,000	SPSS	1/12/1997
110	Chris Lucero	Info Systems	43,000	C++	4/22/1999
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/1995
150	Susan Martin	Marketing	42,000	Java	8/12/1995

Anomalies in this table

- **Insertion –** can’t enter a new employee without having the employee take a class
- **Deletion –** if we remove employee 140, we lose information about the existence of a Tax Acc class

Anomalies in this table

- **Modification -** giving a salary increase to employee 100 forces us to update multiple records

Why do these anomalies exist?

Because we've combined two themes (entity types) into one relation. This results in duplication, and an unnecessary dependency between the entities

Why is it unnormalized?

- Considerable redundancy
- E.g. values for Emp_ID, Name, Dept_Name and Salary appear in two separate rows for employees 100, 110 and 150.
- If salary is changed for employee 100 it must be updated in all repeating rows

What is the problem?

- So the problem with this relation is that it contains data about two entities:

  - EMPLOYEE &

  -COURSE

- We use Normalization to split this relation into two relations

Normalization

- Is the formal process for deciding which attributes should be grouped together in a relation.
- Use common sense during conceptual data modeling to group attributes into entity types
- It is primarily a tool to validate and improve a logical design
- So ‘Normalization’ is the process of decomposing relations with anomalies to produce smaller, well structured relations.

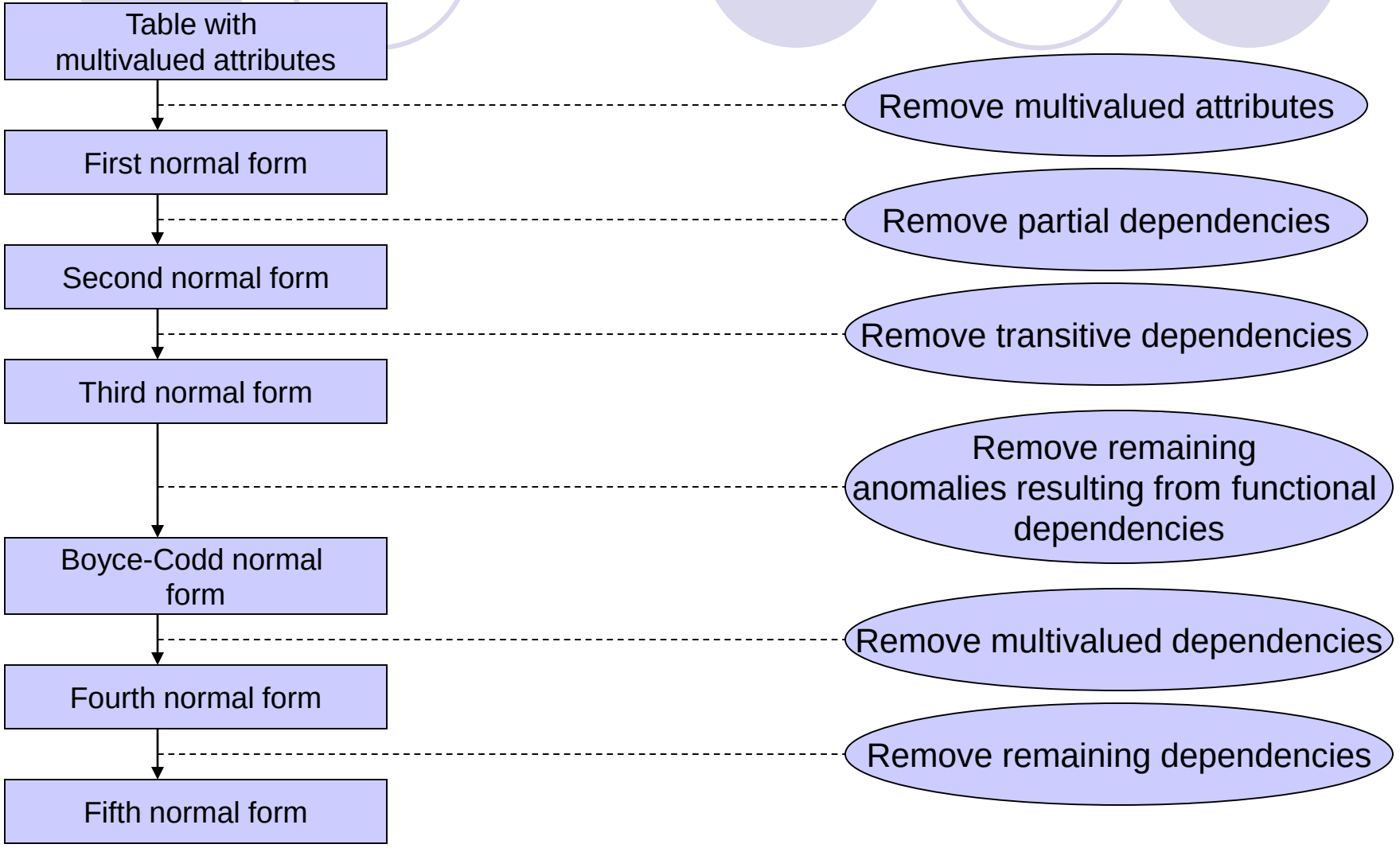
Normalized relations

- Employee 1 (1NF)
- EMP_COURSE (1NF)

Emp_ID	Name	Dept_Name	Salary
100	Margaret Simpson	Marketing	48,000
140	Alain Beeton	Accounting	52,000
110	Chris Lucero	Info Systems	43,000
190	Lorenzo Davis	Finance	55,000
150	Susan Martin	Marketing	42,000

Emp_ID	Course_Title	Date_Completed
100	SPSS	6/19/1998
100	Surveys	10/7/1998
140	Tax Acc	12/8/1996
110	SPSS	1/12/1997
110	C++	4/22/1999
150	SPSS	6/19/1995
150	Java	8/12/1995

Steps in Normalization

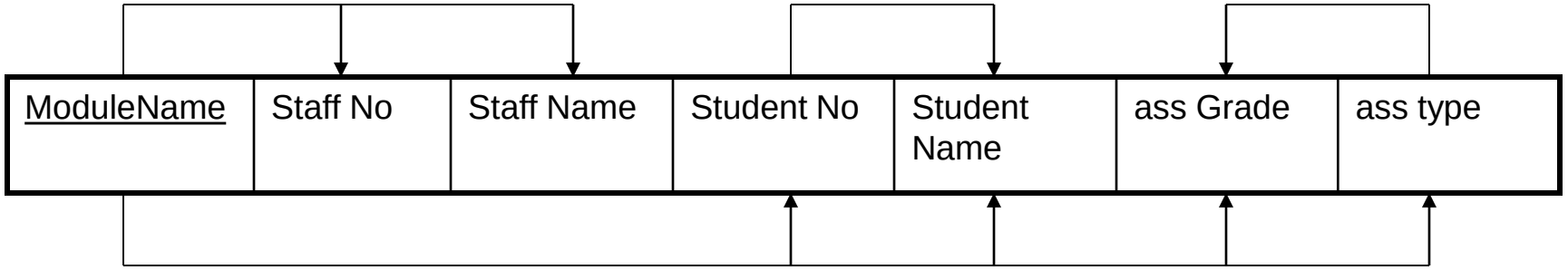


Unnormalized Relation example 2

Module Name	Staff No	Staff Name	StudentNo	Student	Grade	Type
Relational Database Systems	234	Davies T	34698	Smith S	B3	cwk1
Relational Database Systems	234	Davies T	34698	Smith S	B1	cwk2
Relational Database Systems	234	Davies T	37798	Jones S	B2	cwk1
Relational Database Systems	234	Davies T	34888	Patel P	B1	cwk1
Relational Database Systems	234	Davies T	34888	Patel P	B3	cwk2
Relational Database Design	234	Davies T	34698	Smith S	B2	cwk1
Relational Database Design	234	Davies T	34698	Smith S	B3	cwk2
Deductive Database	345	Evans R	34668	Smith J	A1	exam

1 NF relation

- A relation is in first normal form if and only if every non-key attribute is functionally dependent upon the primary key



The attributes StudentNo, StudentName, assGrade and assType are clearly not functionally dependent on our chosen primary key **ModuleName**.

The StaffNo and StaffName are functionally dependent on our chosen primary key

1 NF relation

- Separate the two table:
  - One for the functionally dependent attributes
  - One for the non-dependent attributes
- Declare a compound of ModuleName, StudentNo, assType to be the primary key of this second table.

Normalized relations

Modules		
Module Name	Staff No	Staff Name
Relational Database systems	234	Davies T
Relational Database Design	234	Davies T
Deductive Database	345	Evans R

Assessments				
Module Name	Student No	ass type	Student Name	ass Grade
Relational Database Systems	34698	cwk1	Smith S	B3
Relational Database Systems	34698	cwk2	Smith S	B1
Relational Database Systems	37798	cwk1	Jones S	B2
Relational Database Systems	34888	cwk1	Patel P	B1
Relational Database Systems	34888	cwk2	Patel P	B3
Relational Database Design	34698	cwk1	Smith S	B2
Relational Database Design	34698	cwk2	Smith S	B3
Deductive Database	34668	exam	Smith J	A1

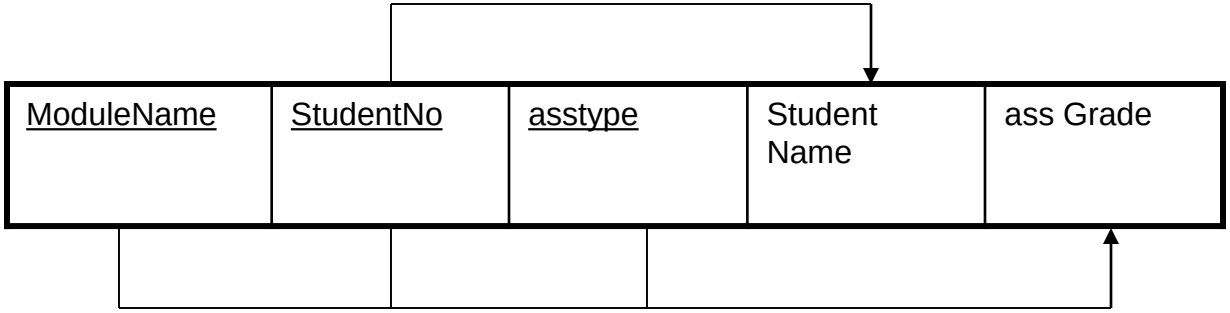
1 NF ⟶ 2 NF

- Remove part-key dependencies.
  - Examine those tables that have compound keys.
  - For each non-key data item in the table ask the following question:

    -Can the data item be uniquely identified by part of the compound key?

2 NF relation

- A relation is in second normal form if and only if it is in 1NF and every non-key attribute is fully functionally dependent on the primary key.



Identify the compound keys:

ModuleName, StudentNo, assType

Questions?

Can the non-key data item (StudentName, assGrade) be uniquely identified by part of the compound key?

Answers

- We need all the items of the key to tell us what the assessment grade is.
- ModuleName has no influence on the StudentName
- StudentNo alone determines StudentName.
- Break out the determinant and dependent data items into their own table.

Functional dependencies and keys

- Functional Dependency: The value of one attribute (the **determinant**) determines the value of another attribute
- Two data items, A and B, are said to be in a determinant or dependent relationship if certain values of data item B always appear with certain values of data item A.

2 NF relation

Modules		
Module Name	Staff No	Staff Name
Relational Database systems	234	Davies T
Relational Database Design	234	Davies T
Deductive Database	345	Evans R

Students	
StudentNo	Student
34698	Smith S
37798	Jones S
34888	Patel P
34668	Smith J

Assessments			
Module Name	Student No	ass type	ass Grade
Relational Database Systems	34698	cwk1	B3
Relational Database Systems	34698	cwk2	B1
Relational Database Systems	37798	cwk1	B2
Relational Database Systems	34888	cwk1	B1
Relational Database Systems	34888	cwk2	B3
Relational Database Design	34698	cwk1	B2
Relational Database Design	34698	cwk2	B3
Deductive Database	34668	exam	A1

2 NF ⟶ 3 NF

- Examine every table and ask each pair of non-key data items:
  - Is the value of field A dependent on the value of field B, or vice versa?
  - If answer is YES, then split off the relevant data items into a separate table.
- A relation is in third normal form if and only if it is in 2NF and every non-key attribute is non-transitively dependent on the primary key

Transitive dependency

- Consider the table named Module.

A = ModuleName

B = StaffNo

C = StaffName

Modules		
Module Name	Staff No	Staff Name
Relational Database systems	234	Davies T
Relational Database Design	234	Davies T
Deductive Database	345	Evans R

StaffName depends on StaffNo and StaffNo depends on ModuleName

i.e. A ⟶ B ⟶ C

StaffNo determines StaffName

StaffName is hence transitively dependent on Modulename.

3 NF Relations

- StaffNo is therefore asking to be a primary key.
- Create a separate table with StaffNo as the primary key.

Modules (Now in 3NF)	
Module Name	Staff No
Relational Database systems	234
Relational Database Design	234
Deductive Database	345

Lecturers (3NF)	
Staff No	Staff Name
234	Davies T
345	Evans R

3 NF Relations (Cont)

Assessments (3NF)			
Module Name	Student No	ass type	ass Grade
Relational Database Systems	34698	cwk1	B3
Relational Database Systems	34698	cwk2	B1
Relational Database Systems	37798	cwk1	B2
Relational Database Systems	34888	cwk1	B1
Relational Database Systems	34888	cwk2	B3
Relational Database Design	34698	cwk1	B2
Relational Database Design	34698	cwk2	B3
Deductive Database	34668	exam	A1

Students	
StudentNo	Student
34698	Smith S
37798	Jones S
34888	Patel P
34668	Smith J

The Bracketing Notation

- To represent the relational schema in an implementation-independent form.
- List a suitable mnemonic name for the table first.
- Followed by a list of data items or column names delimited by commas.
- List the primary key first and underline this item.
- If primary key is made up of two or more attributes, then underline all the component data items.

The Bracketing Notation (Cont)

Example:

- Module(ModuleName, StaffNo)
- Lecturer(StaffNo, StaffName)
- Assessment (ModuleName, StudentNo, assType, assGrade)
- Student(StudentNo, StudentName)

The Normalization Oath

- No Repeating,
- The Fields Depends Upon The Keys,
- The Whole Key,
- And Nothing But The Key,
- So Help Me Codd.

The Normalization Oath (Cont)

- Line 5 simply reminds us that the techniques were originally developed by E.F.Codd in the 1970s.
- Line 2 state that all data items in a table must depend solely upon the key.
- Line 1 indicates that there should be no repeating groups of data in a table.

The Normalization Oath (Cont)

- Line 3 indicates that there should be no part-key dependencies in a table.
- Line 4 reminds us that there should be no inner-data dependencies in a table.. (transitive dependency)
- The only dependence should be between the key and other data items in a table.