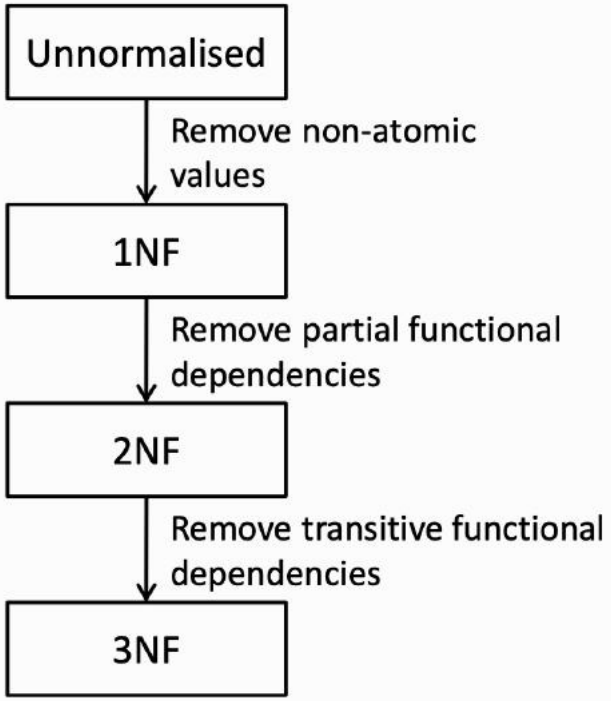
Normalization

Databases and Interface

Matthew Pike

Overview

- Normalization
  - Data Redundancy
  - Functional Dependencies
  - Normal Forms
- Further reading
  - The Manga Guide to Databases, Chapter 3
  - Database Systems, Chapter 14
  - https://docs.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description



Reminder

Term	(Practical) Definition
Candidate Key	A candidate key is a specific type of field in a relational database that can identify each unique record independently of any other data.
Redundancy	Data redundancy is a condition created within a database in which the same piece of data is held in two separate places.

Motivating Normalization

Normalization

- A database design aid
  - Like ER modelling
  - Examines the relationships between attributes
  - Identify functional dependencies
  - Produce relations with desirable properties
- Normalization
  - Aims to reduce data redundancy
  - Redundancy is found/expressed in functional dependencies
  - Normal forms are defined so that they don't contain specific types of functional dependency

Database Design using Normalization: Possible Approach

- Given some data/attributes we want to produce the tables (and the schemas) for our database
- Possible Approach: start with a **Mega Table** that contains all data
- This Mega Table is intentionally not the best data structure because there are a lot of redundancies
- To remove redundancies, we first quantify them in terms of functional dependencies.
- Then we remove functional dependencies by splitting appropriately the Mega Table into smaller tables

Purpose of Normalization

- Identify suitable set of relations for an enterprise data requirements:
  - Minimal number of attributes necessary for the entity
  - Attributes with a close logical relationship (FD)
  - Minimal redundancy
- Specify relations with:
  - Required minimum attributes with ease of:
    - Accessibility
    - Maintenance
    - Storage
  - Updates with minimum operations
    - Avoid data inconsistencies
  - Reduction in storage space

1NF

Normalization to 1NF

- To convert any relation into 1NF, split any non-atomic values

Unnormalised

Module	Dept	Lecturer	Texts
M1	D1	L1	T1, T2
M2	D1	L1	T1, T3
M3	D2	L2	T4
M4	D2	L3	T1, T5
M5	D2	L4	T6

Normalization to 1NF

- To convert any relation into 1NF, split any non-atomic values

Unnormalised

Module	Dept	Lecturer	Texts
M1	D1	L1	T1, T2
M2	D1	L1	T1, T3
M3	D2	L2	T4
M4	D2	L3	T1, T5
M5	D2	L4	T6

1NF

Module	Dept	Lecturer	Text
M1	D1	L1	T1
M1	D1	L1	T2
M2	D1	L1	T1
M2	D1	L1	T3
M3	D1	L2	T4
M4	D2	L3	T1
M4	D2	L3	T5
M5	D2	L4	T6

Problems with 1NF

Example 1

1NF

Module	Dept	Lecturer	Text
M1	D1	L1	T1
M1	D1	L1	T2
M2	D1	L1	T1
M2	D1	L1	T3
M3	D1	L2	T4
M4	D2	L3	T1
M4	D2	L3	T5
M5	D2	L4	T6

- INSERT Anomalies
  - Can't add a lecturer with no texts
- UPDATE Anomalies
  - To change the lecturer for M1, we will need to update two rows
- DELETE Anomalies
  - If we remove M3, we will remove L2 and T4 as well

Problems with 1NF

Example 2

Unnormalised StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
S100	John	Manager	30000	B005	22 Deer st
S102	Jane	Assistant	12000	B004	16 Heck D
S109	Juli	Supervisor	18000	B004	16 Heck D
S101	Julius	Assistant	9000	B007	Heaven str
S104	Janet	Manger	24000	B004	16 Heck D
S103	Andy	Assistant	9000	B005	22 Deer st

- Redundant data
  - e.g., details of branch
- Insertion Anomalies
  - E.g., insert details of new staff requires branch
  - Insert new branch with no member of staff
- Deletion Anomalies
  - E.g., delete the last member of staff in a branch
- Update Anomalies
  - e.g., change the address of one branch

Design by Decomposition

Unnormalised StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
S100	John	Manager	30000	B005	22 Deer st
S102	Jane	Assistant	12000	B004	16 Heck D
S109	Juli	Supervisor	18000	B004	16 Heck D
S101	Julius	Assistant	9000	B007	Heaven str
S104	Janet	Manger	24000	B004	16 Heck D
S103	Andy	Assistant	9000	B005	22 Deer st

Staff

staffNo	sName	position	salary	branchNo
S100	John	Manager	30000	B005
S102	Jane	Assistant	12000	B004
S109	Juli	Supervisor	18000	B004
S101	Julius	Assistant	9000	B007
S104	Janet	Manger	24000	B004
S103	Andy	Assistant	9000	B005

Branch

branchNo	bAddress
B005	22 Deer st
B004	16 Heck D
B007	Heaven str

- Decompose Table
  - Table decomposition remove some anomalies
- Maintain Lossless-Join
  - All instances remain
- Preserve dependency
  - Constraints still maintained

First Normal Form

- In most definitions of the relational model
  - All data values should be atomic
  - This means that table entries should be single values, not sets or composite objects
  - Simplifies queries and data comparisons
- A relation is said to be first normal form (1NF) if all data values are atomic

1NF

Module	Dept	Lecturer	Text
M1	D1	L1	T1
M1	D1	L1	T2
M2	D1	L1	T1
M2	D1	L1	T3
M3	D1	L2	T4
M4	D2	L3	T1
M4	D2	L3	T5
M5	D2	L4	T6

Functional Dependencies

Functional Dependencies

- Functional dependencies (FD) are the main concept in Normalization
- FD is constraints on related attributes
- Describes relationship between attributes
- When there is a FD between attributes the dependency is specified as constraint between the attributes

Functional Dependencies

- If A and B are attributes of relation R
- A → B
  - Implies B is functionally dependent on A
  - Or A functionally determines B, if each value of A is associated with exactly one value of B
- Identifies relationships between attributes
- A is the determinant of B
- A and B may each have one or more attributes
- When two tuples have the same value of A, they also have the same value of B

Functional Dependencies

Staff

staffNo	sName	position	salary	branchNo
S100	John	Manager	30000	B005
S102	Jane	Assistant	12000	B004
S109	Juli	Supervisor	18000	B004
S101	Julius	Assistant	9000	B007
S104	Janet	Manger	24000	B004
S103	Andy	Assistant	9000	B005

staffNo → **position**

- Find relationship between attributes
- A is the determinant of B
  - **staffNo** functionally determines **position** - 1:1
  - Opposite not true Relationship is 1:many

Functional Dependencies

Staff

staffNo	sName	position	salary	branchNo
S100	John	Manager	30000	B005
S102	Jane	Assistant	12000	B004
S109	Juli	Supervisor	18000	B004
S101	Julius	Assistant	9000	B007
S104	Janet	Manger	24000	B004
S103	Andy	Assistant	9000	B005

staffNo, sName → **branchNo**

- Full functional dependency (FFD)
- If A and B are attributes
  - B FFD on A
  - (IFF B is FD on A, but not on any proper subset of A)
- Otherwise partially dependent
- Is this a Full or Partial dependency?

Functional Dependencies

1NF

Module	Dept	Lecturer	Text
M1	D1	L1	T1
M1	D1	L1	T2
M2	D1	L1	T1
M2	D1	L1	T3
M3	D1	L2	T4
M4	D2	L3	T1
M4	D2	L3	T5
M5	D2	L4	T6

- Find Primary key
- Identify all FDs up to this point:

Functional Dependencies

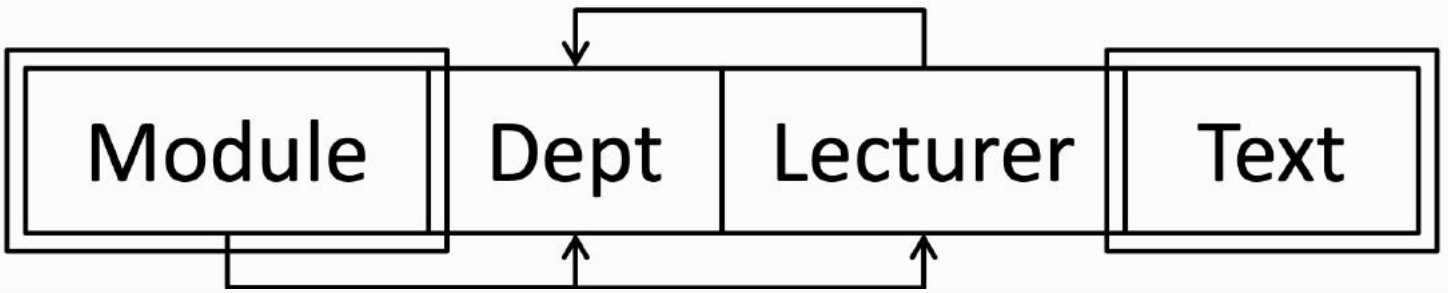
1NF

Module	Dept	Lecturer	Text
M1	D1	L1	T1
M1	D1	L1	T2
M2	D1	L1	T1
M2	D1	L1	T3
M3	D1	L2	T4
M4	D2	L3	T1
M4	D2	L3	T5
M5	D2	L4	T6

- The Primary Key is {Module, Text} so
  - {Module, Text} → {Dept, Lecturer}
- Other FDs are
  - {Module} →{Lecturer}
  - {Module} →{Dept}
  - {Lecturer} → {Dept}

FD Diagrams

- FDs can be represented simply using the headings:



- {Module , Text} is a candidate key, so we put a double box around them
- {Lecturer} →{Dept), so we have an arrow from Lecturer to Dept
- {Module} →{Dept} and {Module} →{Lecturer}, so we have {Module} →{Dept, Lecturer}
- **Note**: Trivial FDs and FDs dependent on an entire candidate key are not included

2NF

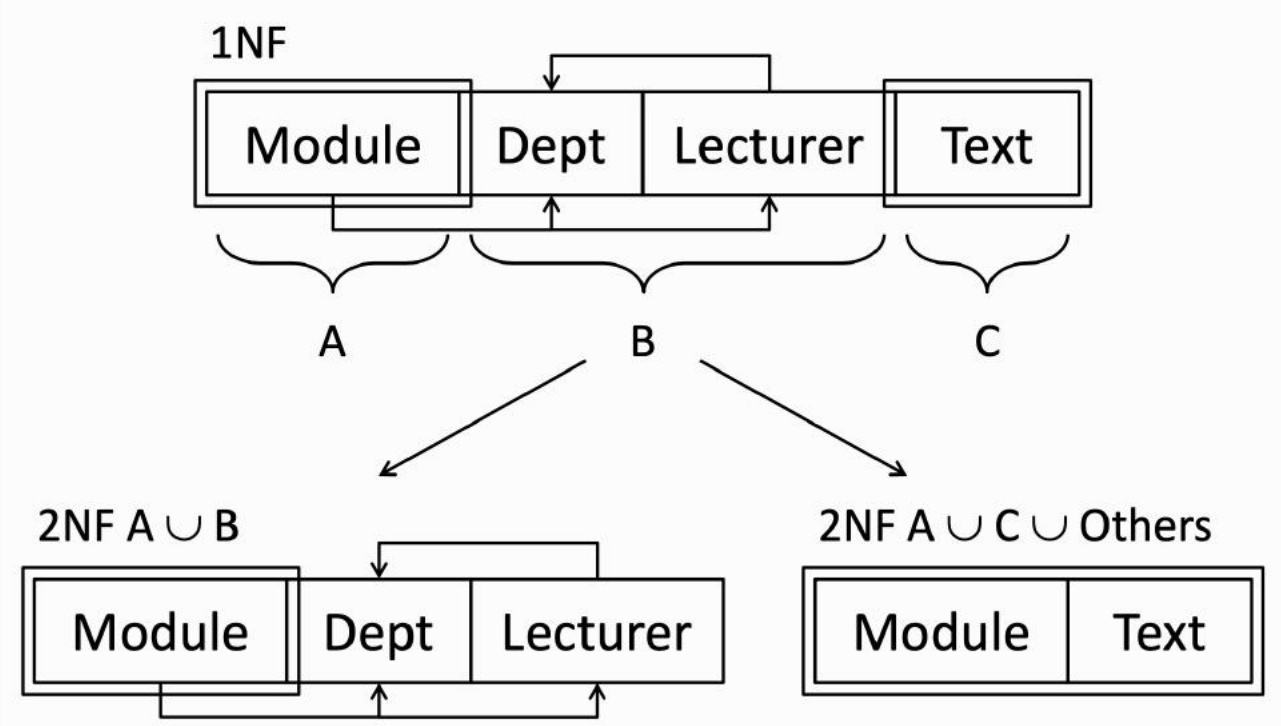
Second Normal Form

- Partial FDs:
  - A FD, A → B is a partial FD, if some attribute of A can be removed and the FD still holds
- Formally, there is some proper subset of A, C ⊂ A, such that C → B

- Second normal form:
  - A relation is in second normal form (2NF) if it is in 1NF and no non-key attribute is partially dependent on a candidate key
  - In other words, no C→B where C is strict subset of a candidate key and B is a non-key attribute.
- **Candidate Keys are minimal!**

Normalising to 2NF

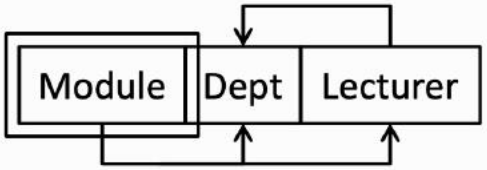
{A,C} → {B}



Normalising to 2NF

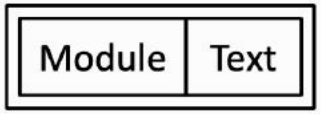
2NFa

Module	Dept	Lecturer
M1	D1	L1
M2	D1	L1
M3	D1	L2
M4	D2	L3
M5	D2	L4



2NFb

Module	Text
M1	T1
M1	T2
M2	T1
M2	T3
M3	T4
M4	T1
M4	T5
M5	T6



Problems Resolved in 2NF

- INSERT Anomalies
  - We can now add a lecturer without texts
  - However, we cannot add a lecturer without teaching a module
- UPDATE Anomalies
  - We only need to change a single row when changing the lecturer of a module
  - However, we need to change 2 rows to update the department for L1
- DELETE Anomalies
  - If we delete M3 then all information about L2 is lost

2NFb

Module	Text
M1	T1
M1	T2
M2	T1
M2	T3
M3	T4
M4	T1
M4	T5
M5	T6

2NFa

Module	Dept	Lecturer
M1	D1	L1
M2	D1	L1
M3	D1	L2
M4	D2	L3
M5	D2	L4

3NF

Functional Dependencies?

2NFa

Module	Dept	Lecturer
M1	D1	L1
M2	D1	L1
M3	D1	L2
M4	D2	L3
M5	D2	L4

- FDs so far?
  - M->D, L
  - L->D
- Which implies
  - M->L->D
- Notice that M is a key for the sub-table

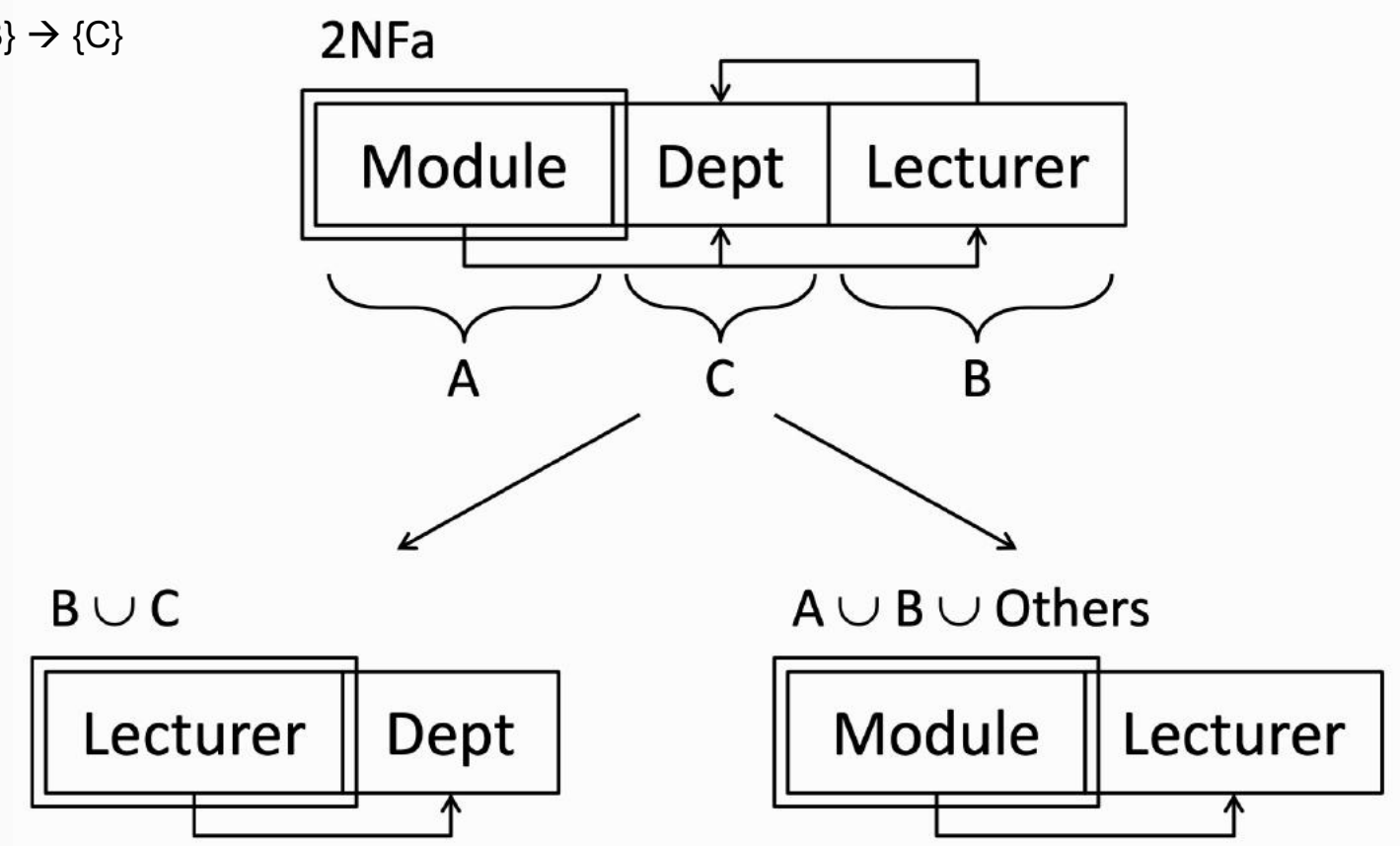
Transitive FDs and 3NF

- Transitive FDs:
  - A FD, A → C is a transitive FD, if there is some set B such that
- A → B and B → C and these are non-trivial FDs
- A→B non-trivial means: B is not a subset of A
- Essentially
  - A →B → C

- Third normal form
  - A relation is in third normal form (3NF) if it is in 2NF and no non-key attribute is transitively dependent on a candidate key

Normalising to 3NF

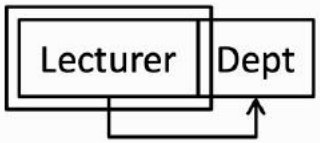
{A} → {B} → {C}



Normalising to 3NF

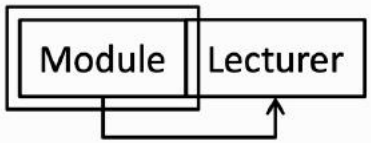
3NFa

Lecturer	Dept
L1	D1
L2	D1
L3	D2
L4	D2



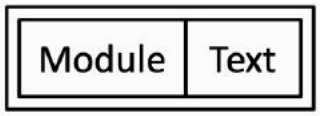
3NFb

Module	Lecturer
M1	L1
M2	L1
M3	L2
M4	L3
M5	L4



2NFb

Module	Text
M1	T1
M1	T2
M2	T1
M2	T3
M3	T4
M4	T1
M4	T5
M5	T6



Problems Resolved in 3NF

- Problems resolved in 3NF
  - **INSERT** - We can now add Lecturers who don't teach any modules
  - **UPDATE** - We need only change a single row to update the department for L1
  - **DELETE** - Ne can delete M3 while preserving L2

3NFa

Lecturer	Dept
L1	D1
L2	D1
L3	D2
L4	D2

3NFb

Module	Lecturer
M1	L1
M2	L1
M3	L2
M4	L3
M5	L4

Summary

Normalization and Design

- Normalization is related to Database design
  - A database should normally be in 3NF at least
  - If your design leads to a non-3NF database, then you might want to revise it
- When you find you have a non-3NF database
  - Identify the FDs that are causing a problem
  - Think if they will lead to any insert, update, or delete anomalies
  - Try to remove them

Summary

- Normalization
  - Data Redundancy
  - Functional Dependencies
  - Normal Forms
  - First, Second and Third Normal Forms