Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

200
1822 - 2022

Normalisation

Man Geni Addysg Uwch yng Nghymru
Where Higher Education Began in Wales

www.uwtsd.ac.uk

Week:
30-Sep-2019

Today's Lecture

- Normalisation
- Functional dependency

What is Normalisation?

- **Normalisation** is a technique for producing a set of *suitable relations that support the data requirements* of an enterprise.

Employee

Man#	Name	Birthdate	JobHistory			Children	
123	Michael Swart	Nov. 22	**JobDate**	**Title**	**SalaryHistory**	**Childname**	**Birthyear**
			2000	Lackey	**SalaryDate** / **Salary**: 2000 / $1,000,000; 2001 / $2,000,000	Mini-me 1	2000
			2002	Senior Lackey	**SalaryDate** / **Salary**: 2002 / $3,000,000; 2003 / $4,000,000	Mini-me 2	2002

What is Normalisation?

- **Normalisation** is a technique for producing a set of *suitable relations that support the data requirements* of an enterprise.

Employee'

Man#	Name	Birthdate
123	Michael Swart	Nov. 22

Children'

Man#	ChildName	BirthYear
123	Mini-me 1	2000
123	Mini-me 2	2002

JobHistory'

Man#	JobDate	Title
123	2000	Lackey
123	2002	Senior Lackey

SalaryHistory'

Man#	JobDate	SalaryDate	Salary
123	2000	2000	$1,000,000
123	2000	2001	$2,000,000
123	2002	2002	$3,000,000
123	2003	2003	$4,000,000

Unnormalized LibraryBooks Table

UNF

BookID	Title	Authors	Genre	PublishedYear	CopiesAvailable
1	"The Great Gatsby"	F. Scott Fitzgerald	Fiction	1925	3
2	"1984"	George Orwell	Fiction	1949	2
3	"A Brief History..."	Stephen Hawking	Non-Fiction	1998	5
4	"The Catcher in..."	J.D. Salinger, John Doe	Fiction	1951	1
5	"Sapiens"	Yuval Noah Harari	Non-Fiction	2011	4

The Purpose of Normalisation

- The main aim of a relational database design is to group attributes into relations to ***minimise data redundancy***.
  - Normalisation, helps to achieve this.
- Benefits of using a database that has a *suitable* set of relations is that the database will be:
  - easier for the user to access and maintain the data.
  - take up minimal storage space on the computer.

Problems of not properly normalised tables

- There are two main problems associated with a relation that is not appropriately normalised:
  - Data Redundancy
  - Update Anomalies for:
    - Insertions (adding new),
    - Deletions, and
    - Updates (modifications).

Data Redundancy and Update Anomalies

- What will be the primary key for this table?
- Can you spot potential problems with this table design?

CampusStaff Table

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

Redundancy

- This CampusStaff table has redundant data.
- The details (e.g. address) of each campus are repeated for each member of staff.
- Relations that have redundant data may have update anomaly problems.
  - An anomaly is an inconsistent, incomplete or contradictory state of the database.
  - Let's have a look at the potential anomaly problems for the produced Staff table.

Redundancy

- Swansea IQ address is repeated for 3 staff members
- Carmarthen is repeated for 2 staff members

CampusStaff Table

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

Insertion (Add New) Anomalies

- There are two types of insertion anomalies.
  - **Problem 1**: To add the details of any new members of staff, we must include the details of the campus the staff is located at.
    - Great care must be taken to enter the correct details (address, phone number) so that the details are *consistent* with the values for other rows in the table.

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

Insertion (Add New) Anomalies

- There are two types of insertion anomalies.
  - **Problem 2**: To add a new campus that currently has no staff assigned (there is no separate campus table), it is necessary to fill the staff related columns with NULL values..
    - This, however, violates entity integrity constraints as it means the staffNo, the primary key, is NULL which is <u>not</u> allowed.

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

Update (Modification) Anomalies

- Suppose a campus moves to another address, we need to update the campus address details.
- In this table design, we must also remember to update the rows of all the staff located at the "moving" campus.
  - If this update is not carried out for any one of the staff at the campus , the database will become inconsistent.

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

Deletion Anomalies

- If we delete a row from the CampusStaff table representing the last member of staff at the campus, the details about that campus are lost.
- For example, deleting the record for Mary Howe means the details for the campus C003 are also lost.

CampusStaff Table

staffNo	staffName	position	salary	campusID	campusAddress
SW101	John Smith	Lecturer	30000	C002	Swansea IQ, SA1 8EW
SW102	Diane Lee	PVC	70000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW103	Ann Beech	Lecturer	18000	C002	Swansea IQ, SA1 8EW
SW105	David Ford	Snr Lecturer	41000	C001	Carmarthen Campus, Carmarthen, SA31 3EP
SW106	Mary Howe	Dean, Humanities	55000	C003	Lampeter Campus Ceredigion SA48 7ED
SW107	Susan Brand	Receptionist	18000	C001	Swansea IQ, SA1 8EW

The Better Design

- The anomaly problems of the Staff table can be eliminated by ***decomposing this relation into smaller relations***.
  - **Campus** (campusID, campusAddress)
  - **Staff** (staffNo, staffName, position, salary, campusID)
- The primary keys (PKs) for each table are underlined.
- CampusID is a foreign key (FK) in the Staff table linking a member of staff to the correct campus.

Anomalies(another simple example to understand anomalies)

student

Student number	Game	Feestructure
1	Basketball	500
2	Basketball	500
3	Basketball	500
4	Cricket	600
5	Cricket	600
6	Cricket	600
7	Tennis	400

student table has from all three anomalies

- Possibility of :
  - Inserting new game without student enrolment?

- Losing information

Eg: deletion of Roll no 7- lost tennis.

- Changes made in more than one place.

Introduction to Functional Dependencies

- What are Functional Dependencies (FD)?

Example Dataset 1: Student Information

Student_ID	Name	Department	HOD
101	Alice	Computer Science	Dr. John
102	Bob	Mechanical Eng.	Dr. Smith
103	Charlie	Computer Science	Dr. John

- Identify the attribute(s) that uniquely determine student's name and department?
- Identify the attribute(s) that uniquely determine the head of department?

Introduction to Functional Dependencies

- What are Functional Dependencies (FD)?

Example Dataset 1: Student Information

Student_ID	Name	Department	HOD
101	Alice	Computer Science	Dr. John
102	Bob	Mechanical Eng.	Dr. Smith
103	Charlie	Computer Science	Dr. John

- Student ID uniquely determines the student's name and department
- Department determines the Head of Department

Introduction to Functional Dependencies

- What are Functional Dependencies (FD)?

Example Dataset 1: Student Information

Student_ID	Name	Department	HOD
101	Alice	Computer Science	Dr. John
102	Bob	Mechanical Eng.	Dr. Smith
103	Charlie	Computer Science	Dr. John

- Student ID uniquely determines the student's name and department

  Student_ID → Name, Department,HOD

  Determinant Dependents

- Department determines the Head of Department

  Department → HOD

  Determinant Dependent

The arrowline is read as 'determines'

Introduction to Functional Dependencies

- What are Functional Dependencies (FD)?

A functional dependency (FD) means that one attribute uniquely determines another. Denoted as **X → Y**, where **X** is a determinant, and **Y** is the dependent attribute.

Activity: Identifying Functional Dependencies

With the following dataset identify the functional dependencies.

Employee_ID	Employee_Name	Department	Manager
201	John	HR	Mr. Parker
202	Sarah	IT	Mr. Stevens
203	Alice	HR	Mr. Parker

Activity: Identifying Functional Dependencies

With the following dataset identify the functional dependencies.

Employee_ID	Employee_Name	Department	Manager
201	John	HR	Mr. Parker
202	Sarah	IT	Mr. Stevens
203	Alice	HR	Mr. Parker

Employee_ID → Employee_Name, Department
Department → Manager

Normalization

- **Normalization** is the process of organizing data to minimize redundancy and improve data integrity.

Normalization - First Normal Form (1NF)

Example Dataset **Non-1NF**

Student_ID	Name	Courses
101	Alice	CSE101, MTH102
102	Bob	PHY101
103	Charlie	CSE101, ENG101, PHY101

Normalization - First Normal Form (1NF)

Example Dataset **Non-1NF**

Student_ID	Name	Courses
101	Alice	CSE101, MTH102
102	Bob	PHY101
103	Charlie	CSE101, ENG101, PHY101

1NF requires that all entries in a column are atomic (indivisible), and each row must be unique.

Normalization - First Normal Form (1NF)

Example Dataset: Non-1NF

Student_ID	Name	Courses
101	Alice	CSE101, MTH102
102	Bob	PHY101
103	Charlie	CSE101, ENG101, PHY101

contains atomic values

1NF requires that all entries in a column are atomic (indivisible), and each row must be unique.

Convert to 1NF:

Student_ID	Name	Course
101	Alice	CSE101
101	Alice	MTH102
102	Bob	PHY101
103	Charlie	CSE101
103	Charlie	ENG101
103	Charlie	PHY101

Activity - First Normal Form (1NF)

Normalize the table below to 1NF

Order_ID	Customer	Products
001	Jane	TV, Laptop
002	John	Smartphone
003	Alice	TV, Camera, Headphones

Activity - First Normal Form (1NF)

1NF requires that all entries in a column are atomic (indivisible), and each row must be unique.

Order_ID	Customer	Products
001	Jane	TV, Laptop
002	John	Smartphone
003	Alice	TV, Camera, Headphones

Expected Result (1NF):

Order_ID	Customer	Product
001	Jane	TV
001	Jane	Laptop
002	John	Smartphone
003	Alice	TV
003	Alice	Camera
003	Alice	Headphones

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Identify the composite key from the above table?

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Identify the composite key from the above table?

composite key is **{StudentID, CourseID}**

Which FD is enough to uniquely determine studentname?

a) {StudentID, CourseID} -> {StudentName}

b) {StudentID} -> {StudentName}

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Identify the composite key from the above table?

composite key is **{StudentID, CourseID}**

Which FD is enough to uniquely determine studentname?

a) {StudentID, CourseID} -> {StudentName}

b) {StudentID} -> {StudentName}

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Identify the composite key from the above table?

composite key is **{StudentID, CourseID}**

Which FD is enough to uniquely determine CourseName?

a) {StudentID, CourseID} -> {CourseName}

b) {CourseID} -> {CourseName}

c) {CourseID} -> {InstructorName}

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Can you create two tables to eliminate partial dependencies?

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	Grade
101	Alice Johnson	C101	Database Systems	Dr. Smith	A
101	Alice Johnson	C102	Data Structures	Dr. Johnson	B
102	Bob Brown	C101	Database Systems	Dr. Smith	B

Students Table (after removing partial dependencies):

StudentID	StudentName
101	Alice Johnson
102	Bob Brown

Courses Table (after removing partial dependencies):

CourseID	CourseName	InstructorName
C101	Database Systems	Dr. Smith
C102	Data Structures	Dr. Johnson

Where to place 'Grade'?

Normalization - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

Students Table (after removing partial dependencies):

StudentID	StudentName
101	Alice Johnson
102	Bob Brown

Courses Table (after removing partial dependencies):

CourseID	CourseName	InstructorName
C101	Database Systems	Dr. Smith
C102	Data Structures	Dr. Johnson

Enrollment Table (no partial dependencies, stores relationships and grades):

StudentID	CourseID	Grade
101	C101	A
101	C102	B
102	C101	B

Activity - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

EmployeeID	EmployeeName	DeptID	DeptName	ProjectID	ProjectName	HoursWorked
E001	Alice Johnson	D01	HR	P101	Payroll System	30
E001	Alice Johnson	D01	HR	P102	Recruitment App	25
E002	Bob Brown	D02	IT	P101	Payroll System	40
E003	Charlie Davis	D01	HR	P103	Employee Portal	20
E004	Daisy White	D03	Finance	P104	Budgeting App	35

Step 1: Identify the Composite Key

Step 2: Identify Partial Dependencies

Step 3: Break the Table into 2NF Tables

Activity - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

Employee Table (2NF)

EmployeeID	EmployeeName	DeptID
E001	Alice Johnson	D01
E002	Bob Brown	D02
E003	Charlie Davis	D01
E004	Daisy White	D03

Department Table (2NF)

DeptID	DeptName
D01	HR
D02	IT
D03	Finance

Activity - Second Normal Form (2NF)

- The table must be in 1NF.
- There should be no **partial dependency**

Project Table (2NF)

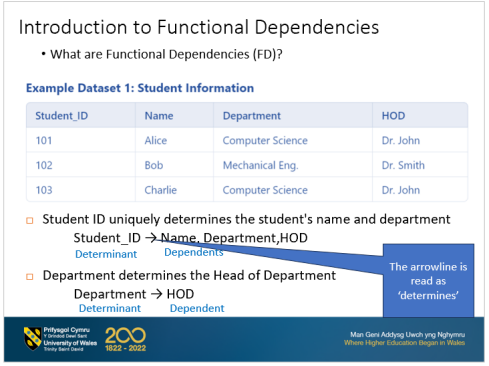
ProjectID	ProjectName
P101	Payroll System
P102	Recruitment App
P103	Employee Portal
P104	Budgeting App

Assignment Table (2NF)

This table contains the relationships between employees and the projects they are working on, including the **HoursWorked**.

EmployeeID	ProjectID	HoursWorked
E001	P101	30
E001	P102	25
E002	P101	40
E003	P103	20
E004	P104	35

Normalization - Third Normal Form (3NF)



- The table must be in 2NF.
- There should be no **transitive dependency**

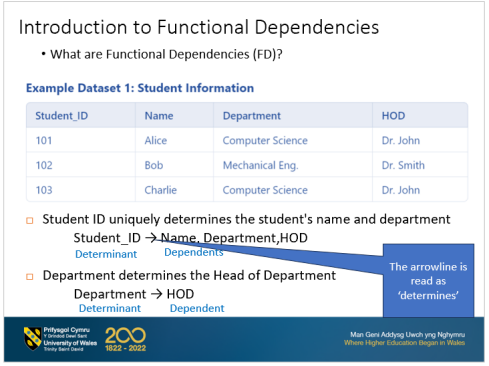
StudentID	StudentName	CourseID	CourseName	InstructorName	InstructorPhone
101	Alice Johnson	C101	Math	Dr. Smith	123-456-7890
102	Bob Brown	C102	History	Dr. Adams	987-654-3210
101	Alice Johnson	C102	History	Dr. Adams	987-654-3210

StudentID → StudentName: The student's name depends on the student's ID.

CourseID → CourseName: The course name depends on the course's ID.

CourseID → InstructorName: The instructor's name depends on the course.
InstructorName → InstructorPhone: The instructor's phone number depends on the instructor.

Normalization - Third Normal Form (3NF)



- The table must be in 2NF.
- There should be no **transitive dependency**

StudentID	StudentName	CourseID	CourseName	InstructorName	InstructorPhone
101	Alice Johnson	C101	Math	Dr. Smith	123-456-7890
102	Bob Brown	C102	History	Dr. Adams	987-654-3210
101	Alice Johnson	C102	History	Dr. Adams	987-654-3210

StudentID → StudentName: The student's name depends on the student's ID.

CourseID → CourseName: The course name depends on the course's ID.

CourseID → InstructorName: The instructor's name depends on the course.
InstructorName → InstructorPhone: The instructor's phone number depends on the instructor.

Normalization - Third Normal Form (3NF)

- The table must be in 2NF.
- There should be no **transitive dependency**

StudentID → StudentName
CourseID → CourseName
CourseID → InstructorName
InstructorName → InstructorPhone

New Tables (in 3NF):

1. **Student Table** (No changes):

StudentID	StudentName
101	Alice Johnson
102	Bob Brown

2. **Course Table** (No transitive dependency):

CourseID	CourseName	InstructorName
C101	Math	Dr. Smith
C102	History	Dr. Adams

3. **Instructor Table** (Instructor phone moved to a separate table):

InstructorName	InstructorPhone
Dr. Smith	123-456-7890
Dr. Adams	987-654-3210

Activity: Student Module Report

StudentID	Student Name	Module Code	ModuleName	Credits	Mark1	Mark2	lecturerID	lecturerName
P100001	John Kay	M5X0110	Database Systems	30	75.00	65.00	LO40	Tina Murphy
		M5X0105	Web Development	20	79.50	69.00	LO93	Tony Shaw
P100004	Aline Stewart	M5X0110	Database Systems	30	55.00	52.00	LO40	Tina Murphy
		M5X0105	Web Development	20	41.00	39.00	LO93	Tony Shaw
		M5X0104	Internet Security	20	62.50	67.00	LO93	Tony Shaw

- Assume each module has two assessments and the marks are stored in mark1 and mark2.

UNF to 1NF Method 1:
Step 2 -Flattening the Table

- Fill the group values down in the empty cells.

studentID	student Name	moduleCode	moduleName	credits	mark1	mark2	lecturerID	lecturerName
P100001	John Kay	M5X0110	Database Systems	30	75.00	65.00	LO40	Tina Murphy
P100001	John Kay	M5X0105	Web Development	20	79.50	69.00	LO93	Tony Shaw
P100004	Aline Stewart	M5X0110	Database Systems	30	55.00	52.00	LO40	Tina Murphy
P100004	Aline Stewart	M5X0105	Web Development	20	41.00	39.00	LO93	Tony Shaw
P100004	Aline Stewart	M5X0104	Internet Security	20	62.50	67.00	LO93	Tony Shaw

Second Normal Form (2NF)

- A relation that is in **1NF** and every non-primary key attribute is ***fully functionally dependent*** on the primary key.
- This step involves *removing partial dependencies*.

1NF to 2NF: StudentModule Table

studentID	student Name	moduleCode	moduleName	credits	mark1	mark2	lecturerID	lecturerName
P100001	John Kay	M5X0110	Database Systems	30	75.00	65.00	LO40	Tina Murphy
P100001	John Kay	M5X0105	Web Development	20	79.50	69.00	LO93	Tony Shaw
P100004	Aline Stewart	M5X0110	Database Systems	30	55.00	52.00	LO40	Tina Murphy
P100004	Aline Stewart	M5X0105	Web Development	20	41.00	39.00	LO93	Tony Shaw
P100004	Aline Stewart	M5X0104	Internet Security	20	62.50	67.00	LO93	Tony Shaw

- What is the primary key?

StudentID and Module code

- What are the functional dependencies? A few - next slide.

1NF to 2NF: StudentModule Table

ModuleLecturer

moduleCode	moduleName	credits	lecturerID	lecturerName
M5X0110	Database Systems	30	LO40	Tina Murphy
M5X0105	Web Development	20	LO93	Tony Shaw
M5X0104	Internet Security	20	LO93	Tony Shaw

Student

studentID	student Name
P100001	John Kay
P100004	Aline Stewart

2NF Tables

Student

studentID	student Name
P100001	John Kay
P100004	Aline Stewart

ModuleLecturer

moduleCode	moduleName	credits	lecturerID	lecturerName
M5X0110	Database Systems	30	LO40	Tina Murphy
M5X0105	Web Development	20	LO93	Tony Shaw
M5X0104	Internet Security	20	LO93	Tony Shaw

StudentModule

studentID	moduleCode	mark1	mark2
P100001	M5X0110	75.00	65.00
P100001	M5X0105	79.50	69.00
P100004	M5X0110	55.00	52.00
P100004	M5X0105	41.00	39.00
P100004	M5X0104	62.50	67.00

2NF to 3NF
Student and StudentModule Tables

- All the non-primary keys in the Student and StudentModule tables are functionally dependent on only their primary keys.
- They do not have any transitive dependencies.

Student

studentID	student Name
P100001	John Kay
P100004	Aline Stewart

StudentModule

studentID	moduleCode	mark1	mark2
P100001	M5X0110	75.00	65.00
P100001	M5X0105	79.50	69.00
P100004	M5X0110	55.00	52.00
P100004	M5X0105	41.00	39.00
P100004	M5X0104	62.50	67.00

2NF to 3NF
ModuleLecturer Table

moduleCode	moduleName	credits	lecturerID	lecturerName
M5X0110	Database Systems	30	LO40	Tina Murphy
M5X0105	Web Development	20	LO93	Tony Shaw
M5X0104	Internet Security	20	LO93	Tony Shaw

- What is the primary key? **moduleCode**
- What are the functional dependencies?
  - moduleCode → moduleName, credits, lecturerID, lecturerName
  - lecturerID → lecturerName
- Any transitive dependencies?
  - Yes: lecturerID → lecturerName

    moduleCode functionally determines lecturerName via the lecturerID.
  - Remove this transitive dependency into a separate relation.

2NF to 3NF
ModuleLecturer Table

- Create two tables from the ModuleLecturer table
  - Lecturer

lecturerID	lecturerName
LO40	Tina Murphy
LO93	Tony Shaw

  - Module

moduleCode	moduleName	credits	lecturerID
M5X0110	Database Systems	30	LO40
M5X0105	Web Development	20	LO93
M5X0104	Internet Security	20	LO93

The Normalised 3NF Relations

Student

studentID	student Name
P100001	John Kay
P100004	Aline Stewart

Module

moduleCode	moduleName	credits	lecturerID
M5X0110	Database Systems	30	LO40
M5X0105	Web Development	20	LO93
M5X0104	Internet Security	20	LO93

Lecturer

lecturerID	lecturerName
LO40	Tina Murphy
LO93	Tony Shaw

StudentModule

studentID	moduleCode	mark1	mark2
P100001	M5X0110	75.00	65.00
P100001	M5X0105	79.50	69.00
P100004	M5X0110	55.00	52.00
P100004	M5X0105	41.00	39.00
P100004	M5X0104	62.50	67.00

Lab Exercise – Normalize to 3NF

BorrowID	StudentID	StudentName	StudentPhone	BookID	BookTitle	Author	BorrowDate	ReturnDate	LibraryBranch
B001	S101	Alice Johnson	123-456-7890	BK101	Database Systems	Dr. Tan	2024-09-01	2024-09-10	Main Branch
B002	S102	Bob Brown	987-654-3210	BK102	Operating Systems	Dr. Wong	2024-09-02	2024-09-12	Main Branch
B003	S101	Alice Johnson	123-456-7890	BK103	Data Structures	Dr. Gupta	2024-09-03	2024-09-13	West Branch
B004	S103	Charlie Davis	555-987-1234	BK101	Database Systems	Dr. Tan	2024-09-05	2024-09-15	Main Branch
B005	S104	Daisy White	666-789-3456	BK104	Network Security	Dr. Smith	2024-09-06	2024-09-16	East Branch

Lab Exercise – Normalize to 3NF

Students Table (2NF)

StudentID	StudentName	StudentPhone
S101	Alice Johnson	123-456-7890
S102	Bob Brown	987-654-3210
S103	Charlie Davis	555-987-1234
S104	Daisy White	666-789-3456

Books Table (2NF)

BookID	BookTitle	Author
BK101	Database Systems	Dr. Tan
BK102	Operating Systems	Dr. Wong
BK103	Data Structures	Dr. Gupta
BK104	Network Security	Dr. Smith

Lab Exercise – Normalize to 3NF

Borrowing Table (3NF)

BorrowID	StudentID	BookID	BorrowDate	ReturnDate
B001	S101	BK101	2024-09-01	2024-09-10
B002	S102	BK102	2024-09-02	2024-09-12
B003	S101	BK103	2024-09-03	2024-09-13
B004	S103	BK101	2024-09-05	2024-09-15
B005	S104	BK104	2024-09-06	2024-09-16

LibraryBranches Table (3NF)

LibraryBranch	Location
Main Branch	Main Street
West Branch	West Avenue
East Branch	East Road

Database Normalization!

1NF made you clean up the mess,

2NF asked you to stop being so *partially dependent*, and

3NF taught you that transitive dependencies are more confusing than they sound!

Further Reading :

Chapter 11 Database Analysis and the DreamHome Case Study

Chapter 14: Normalisation

Database Systems: A Practical Approach to Design, Implementation, and Management Sixth edition by • Thomas Connolly • Carolyn Begg