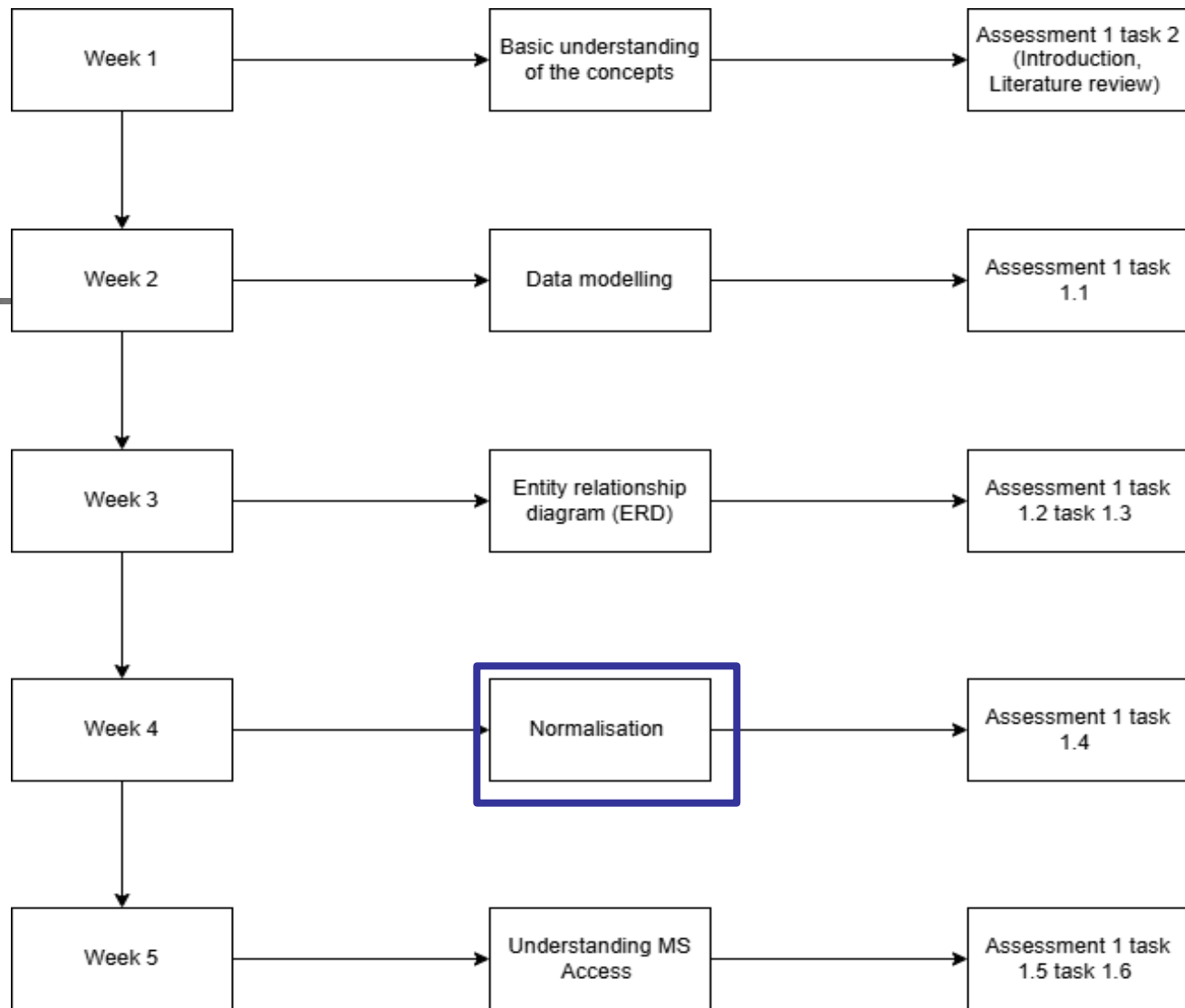
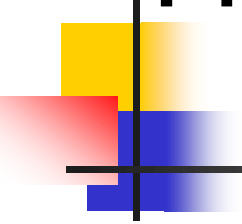


Database Management Systems

Normalisation

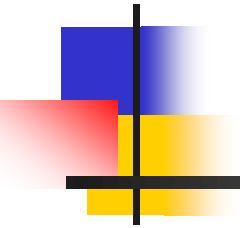
Module Map



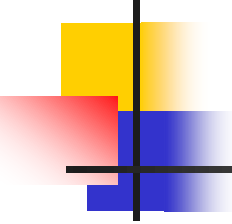


Five Steps in Creating an ERD

1. **Identify the entities in the system that are being modelled.**
2. **Identify the keys for each entity.**
3. **Identify the attributes for each entity.**
4. **Identify the relationships between the entities under cardinality and participation constraint.**
5. **Break down large data groups into smaller groups and ensure there are no data redundancies.**

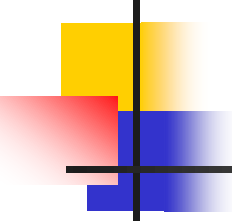


Normalization



Normalization

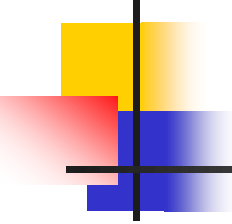
- What is Normalization?



Normalization

- What is Normalization?
- **Normalization** is the process of organizing data in a database to:
 - Reduce duplication
 - Avoid data inconsistencies
 - Make data easier to maintain

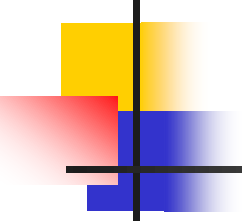
Normalization means breaking big messy tables into smaller, well-structured tables.



Scenario

- You run a small college.
Students enrol on courses, and each course has one tutor.
- You decide to store everything in one big table.

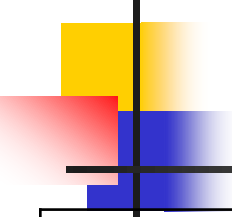
Step 1: Unnormalized Data (UNF)



StudentID	StudentName	CourseIDs	CourseNames	TutorNames
S001	Alice Brown	C101, C102	Maths, English	Dr Smith, Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103, C101	Science, Maths	Dr Lee, Dr Smith

Problems?

Step 1: Unnormalized Data (UNF)

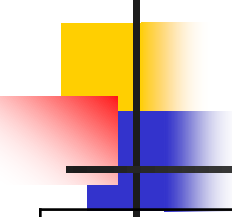


StudentID	StudentName	CourseIDs	CourseNames	TutorNames
S001	Alice Brown	C101, C102	Maths, English	Dr Smith, Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103, C101	Science, Maths	Dr Lee, Dr Smith

Problems:

- Multiple courses in one cell (C101, C102)
- Hard to search
- Hard to update
- Lots of duplication (Dr Smith appears multiple times)

Step 1: Unnormalized Data (UNF)



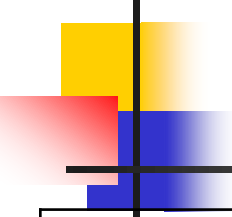
StudentID	StudentName	CourseIDs	CourseNames	TutorNames
S001	Alice Brown	C101, C102	Maths, English	Dr Smith, Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103, C101	Science, Maths	Dr Lee, Dr Smith

Problems:

- Multiple courses in one cell (C101, C102)
- Hard to search
- Hard to update
- Lots of duplication (Dr Smith appears multiple times)

This is Unnormalized Form (UNF).

First Normal Form (1NF)

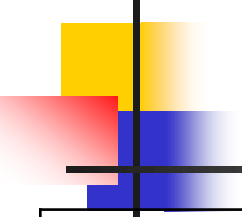


StudentID	StudentName	CourseIDs	CourseNames	TutorNames
S001	Alice Brown	C101, C102	Maths, English	Dr Smith, Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103, C101	Science, Maths	Dr Lee, Dr Smith

Rule for 1NF:

- No multiple values in one cell
- Each field must contain only ONE value

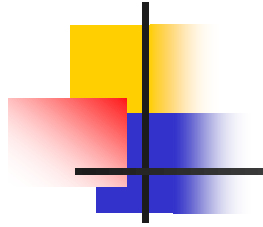
Convert to 1NF



StudentID	StudentName	CourseIDs	CourseNames	TutorNames
S001	Alice Brown	C101, C102	Maths, English	Dr Smith, Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103, C101	Science, Maths	Dr Lee, Dr Smith

Separate each course into its own row

1NF



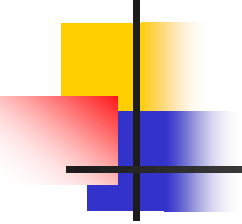
StudentID	StudentName	CourseID	CourseName	TutorName
S001	Alice Brown	C101	Maths	Dr Smith
S001	Alice Brown	C102	English	Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103	Science	Dr Lee
S003	Emma Green	C101	Maths	Dr Smith

What We Fixed

- No more lists inside cells
- One value per field

Problems?

1NF

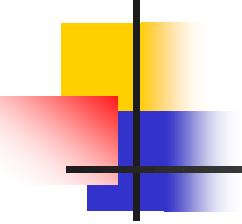


StudentID	StudentName	CourseID	CourseName	TutorName
S001	Alice Brown	C101	Maths	Dr Smith
S001	Alice Brown	C102	English	Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103	Science	Dr Lee
S003	Emma Green	C101	Maths	Dr Smith

Problems we still have:

- Student names repeated
- Course names repeated
- Tutor names repeated

Convert to Second Normal Form (2NF)



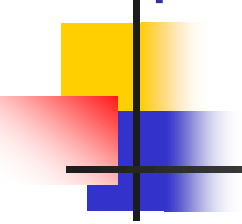
StudentID	StudentName	CourseID	CourseName	TutorName
S001	Alice Brown	C101	Maths	Dr Smith
S001	Alice Brown	C102	English	Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103	Science	Dr Lee
S003	Emma Green	C101	Maths	Dr Smith

Rule for 2NF:

- Must already be in 1NF
- Remove partial dependency
- Data must depend on the whole primary key

Convert to Second Normal Form (2NF)

Step 1: Identify Primary Key

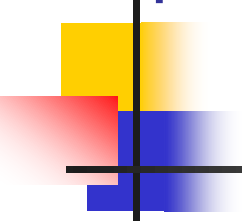


StudentID	StudentName	CourseID	CourseName	TutorName
S001	Alice Brown	C101	Maths	Dr Smith
S001	Alice Brown	C102	English	Prof Jones
S002	John White	C101	Maths	Dr Smith
S003	Emma Green	C103	Science	Dr Lee
S003	Emma Green	C101	Maths	Dr Smith

Identify the primary key

Convert to Second Normal Form (2NF)

Step 1: Identify Primary Key

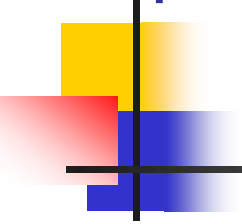


StudentID	CourseID	StudentName	CourseName	TutorName
S001	C101	Alice Brown	Maths	Dr Smith
S001	C102	Alice Brown	English	Prof Jones
S002	C101	John White	Maths	Dr Smith
S003	C103	Emma Green	Science	Dr Lee
S003	C101	Emma Green	Maths	Dr Smith

Identify the primary key

Convert to Second Normal Form (2NF)

Step 1: Identify Primary Key



StudentID	CourseID	StudentName	CourseName	TutorName
S001	C101	Alice Brown	Maths	Dr Smith
S001	C102	Alice Brown	English	Prof Jones
S002	C101	John White	Maths	Dr Smith
S003	C103	Emma Green	Science	Dr Lee
S003	C101	Emma Green	Maths	Dr Smith

The primary key is:

StudentID + CourseID

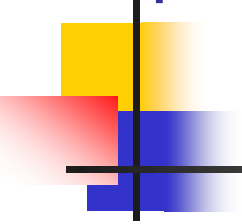
Because:

A student can take many courses

A course has many students

Convert to Second Normal Form (2NF)

Step 1: Identify Primary Key



StudentID	CourseID	StudentName	CourseName	TutorName
S001	C101	Alice Brown	Maths	Dr Smith
S001	C102	Alice Brown	English	Prof Jones
S002	C101	John White	Maths	Dr Smith
S003	C103	Emma Green	Science	Dr Lee
S003	C101	Emma Green	Maths	Dr Smith

Problem

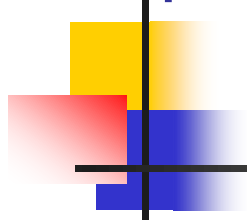
StudentName depends only on StudentID

CourseName and TutorName depend only on CourseID

They do NOT depend on the full key.

Convert to Second Normal Form (2NF)

Step 2: Split Into Tables



StudentID	StudentName
S001	Alice Brown
S002	John White
S003	Emma Green

CourseID	CourseName	TutorName
C101	Maths	Dr Smith
C102	English	Prof Jones
C103	Science	Dr Lee

StudentID	CourseID
S001	C101
S001	C102
S002	C101
S003	C103
S003	C101

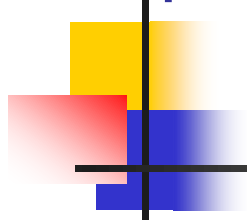
What We Fixed

Student details stored once
Course details stored once
Less duplication

Problems?

Convert to Second Normal Form (2NF)

Step 2: Split Into Tables



StudentID	StudentName
S001	Alice Brown
S002	John White
S003	Emma Green

CourseID	CourseName	TutorName
C101	Maths	Dr Smith
C102	English	Prof Jones
C103	Science	Dr Lee

StudentID	CourseID
S001	C101
S001	C102
S002	C101
S003	C103
S003	C101

Problems:

In the Courses table:

| CourseID | CourseName | TutorName |

What if:

A tutor teaches multiple courses?

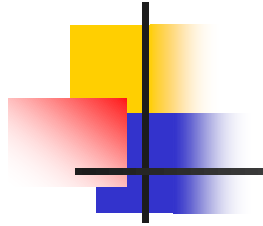
We want to store Tutor Email?

Tutor details depend on Tutor, not Course.

This creates duplication if:

Dr Smith teaches 3 courses.

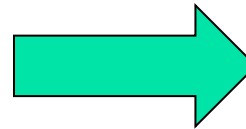
Convert to Third Normal Form (3NF)



StudentID	StudentName
S001	Alice Brown
S002	John White
S003	Emma Green

StudentID	CourseID
S001	C101
S001	C102
S002	C101
S003	C103
S003	C101

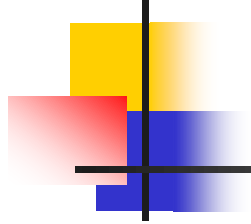
CourseID	CourseName	TutorName
C101	Maths	Dr Smith
C102	English	Prof Jones
C103	Science	Dr Lee



TutorID	TutorName
T01	Dr Smith
T02	Prof Jones
T03	Dr Lee

CourseID	CourseName	TutorID
C101	Maths	T01
C102	English	T02
C103	Science	T03

Final Structure (3NF Achieved)



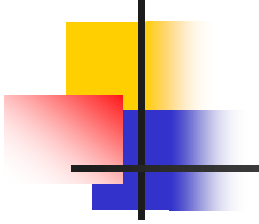
StudentID	StudentName
S001	Alice Brown
S002	John White
S003	Emma Green

StudentID	CourseID
S001	C101
S001	C102
S002	C101
S003	C103
S003	C101

CourseID	CourseName	TutorID
C101	Maths	T01
C102	English	T02
C103	Science	T03

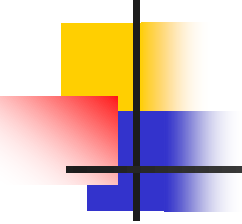
TutorID	TutorName
T01	Dr Smith
T02	Prof Jones
T03	Dr Lee

Normalization



Normal Form	What It Fixes
1NF	Removes multiple values in one cell
2NF	Removes data that depends on only part of a key
3NF	Removes data that depends on another non-key field

In class activity

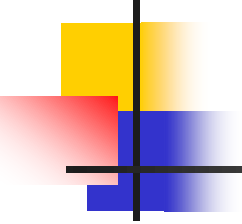


Exercise: Company Training Records

- A company keeps a record of which **employees** attend which **training courses**.
Each course is run by **one instructor**.
- Currently, all the data is stored in **one table**.

EmployeeID	EmployeeName	CourseCodes	CourseTitles	InstructorNames
E001	Sarah Khan	T101, T102	Excel, Python	Mr Adams, Ms Green
E002	Tom Blake	T101	Excel	Mr Adams
E003	Lucy Moore	T103, T101	Cyber Security, Excel	Dr Patel, Mr Adams

In class activity



Exercise: Company Training Records

- A company keeps a record of which **employees** attend which **training courses**.
Each course is run by **one instructor**.
- Currently, all the data is stored in **one table**.

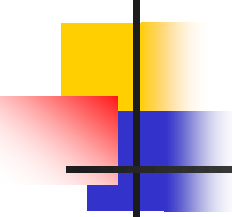
EmployeeID	EmployeeName	CourseCodes	CourseTitles	InstructorNames
E001	Sarah Khan	T101, T102	Excel, Python	Mr Adams, Ms Green
E002	Tom Blake	T101	Excel	Mr Adams
E003	Lucy Moore	T103, T101	Cyber Security, Excel	Dr Patel, Mr Adams

In class activity

■ Scenario (Exercise)

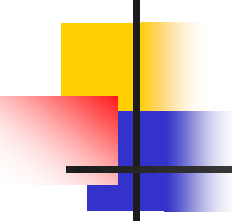
- A **library** keeps track of:
- Members
- The books they borrow
- The author of each book
- Currently, everything is stored in **one table**.

MemberID	MemberName	BookIDs	BookTitles	Authors
M001	Sarah Khan	B101, B102	Database Basics, Python Intro	A. Green, B. J. Black
M002	Tom Lewis	B103	Web Design	M. White
M003	Emma Patel	B101, B104	Database Basics, Networks	A. Green, B. R. Brown



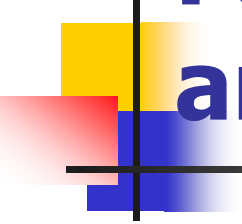
<https://forms.office.com/e/2cQGLEzwwfK>





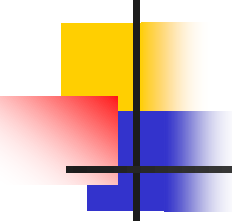
Normalisation

- The main objective of having an efficient database is to reduce the data redundancy for better storage and performance.
- Normalization: a formal method to remove the data redundancy or repetition in the table, by clearly identifying table based on the keys (PK or FK) and the Functional Dependencies (FD) among the attributes.
- The process of transforming data into well formed or natural groupings such that one fact is in one place and the correct relationships between facts exist.



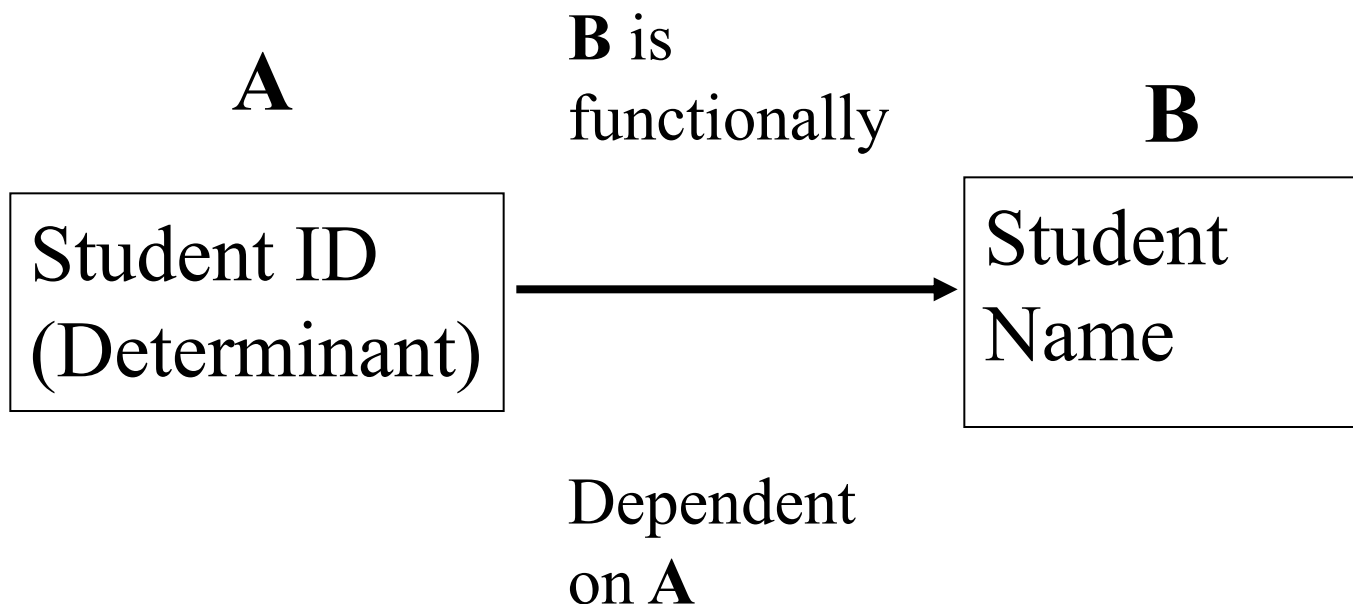
Functional Dependency and Determinant

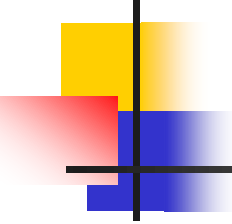
- Functional dependency: describe the relationship between attributes in a relation. Who determines whom and who depends on whom.
- Determinant is any attribute on which other attributes are dependent



Functional Dependency

- Attribute B is functionally dependent on attribute A if the value of B always appears with the value of A, and the value of B is associated with exactly one value of A





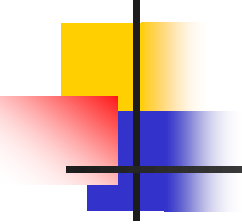
Types of Functional Dependency

- Full FD: one key attribute fully determines another attribute.
- Partial FD: one or more key attributes determine another attributes, however, the non key attribute only depends on one of the key attributes (not the whole keys)
 - A and B $\rightarrow$ C.
- Transitive FD: cross functional dependency; A $\rightarrow$ B, B $\rightarrow$ C, thus, A $\rightarrow$ C via B!!

Unnormalised Form (UNF)

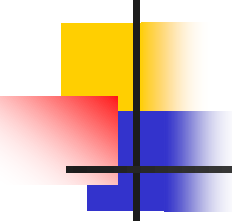
- Each cell should only contain one value, it is illegal to store more than one value with the same data type in a single cell, i.e. tables do not contain “repeating group”. A table that satisfies this condition is said to be normalized or in first normal form.
- Those tables having RG are said to be un-normalized.

Customer_ID	Cutomer_Details	Order_ID	Order_Date	Order_Quantity	Product_ID	Product_Details
C01		O-001 O-746		2 4 5	P001 P002 P003	
C02		O-867 O-746		2 4	P001 P003	



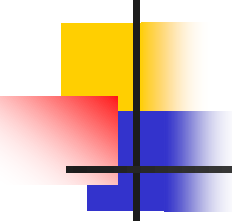
Repeating Groups

A repeating group is an attribute or group of attributes within a table that occurs with multiple values for a single occurrence of the nominated primary key for that table.



Normalization

- There are six types of Normal Form (NF) used to achieve the goal of normalization based on the different FD, step by step.
 - 1NF to remove repeating group with basic FD
 - 2NF to remove Partial FD, non key attributes must depend on the key attribute.
 - 3NF remove transitive FD.
 - Boyce-Codd NF: every determinant is a key attribute.
 - 4NF: further to BC NF, no multiple-valued FD.
 - 5NF: a relation has no join FD.
- Can be used to check table design against the six NF criteria.



First Normal Form (1NF)

- A relation is in the 1NF if the intersection of each row and column contains one and only one value.
- Repeating groups are removed
- Finding the right parent and child.

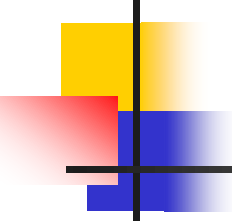
Example of 1NF

Customer_ID → Customer_Details
 Customer_ID → Order_ID.....

Customer_ID (PK)	Customer_Details	Order_ID	Order_Date	Order_Quantity	Product_ID	Product_Details
C01		O-001		2	P001	
		O-746		4	P002	
C02		O-867		2	P001	
		O-746		4	P003	

<u>Customer_ID (PK)</u>	Customer_Details
C01	
C02	

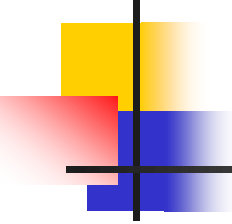
<u>Order_ID (PK)</u>	<u>Customer_ID (FK)</u>	Order_Date	Order_Quantity	Product_ID	Product_Details
O-001	C01	...	2	P001	
O-746	C01		4	P002	
O-867	C02		2	P001	
O-746	C02	...	4	P003	



After 1NF

<u>Customer</u> _ID (K)	Customer _Details	Town ID	Town Name
------------------------------------	----------------------	---------	-----------

<u>Customer</u> _ID (K)	<u>Order</u> _ID (K)	Order_ Date	Order_ Quantity	Pro duct _ID	Product _Details	Pro_ warehouse _no	Ware house Details
--	-------------------------------------	----------------	--------------------	--------------------	---------------------	--------------------------	--------------------------



Second Normal Form (2NF)

- A relation is in the 2NF if it is in the 1NF and every non-key attribute is **fully** functionally dependent on the primary key, i.e.
 - Partial dependency is not allowed
 - Those attributes that depend on only part of the key are removed.

Example of 2NF - Analysis

<u>Customer_ID</u> (K)	Customer_Details	Town ID	Town Name
-------------------------------	------------------	---------	-----------

<u>Customer_ID</u> (K)	<u>Order_ID</u> (K)	Order_Date	Order_Quantity	Product_ID	Product_Details	Pro_warehouse_no	Ware house Details
-------------------------------	----------------------------	------------	----------------	------------	-----------------	------------------	--------------------

Customer_ID, Order ID → Order_date

(Partial FD)

<u>Customer_ID</u> (K)	<u>Order_ID</u> (K)	Order_Date
-------------------------------	----------------------------	------------

<u>Order_ID</u> (K)	Order_Quantity	Product_ID	Product_Details	Pro_warehouse_no	Ware house Details
----------------------------	----------------	------------	-----------------	------------------	--------------------

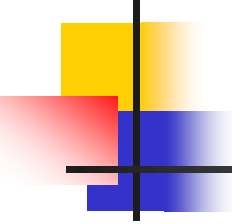
Example of 2NF – Clues and Results

OrderID ProductID $\longrightarrow$ Order_quantity (Partial FD)

Order_ID (K)	Order_Quantity	Product_ID	Product_Details	Pro_warehouse_no	Ware house Details
---------------------	----------------	------------	-----------------	------------------	--------------------

<u>Order_ID(PK)</u>	Order_Quantity	<u>Product_ID (FK)</u>
---------------------	----------------	------------------------

Product_ID (PK)	Product_Details	Pro_warehouse_no	Warehouse Details
-----------------	-----------------	------------------	-------------------



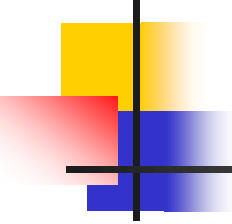
After 2NF

<u>Customer_ID</u> (K)	Customer_Details	Town ID	Town Name
-------------------------------	------------------	---------	-----------

<u>Customer_ID</u> (K)	<u>Order_ID</u> (K)	Order_Date
-------------------------------	----------------------------	------------

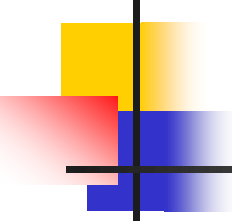
<u>Order_ID</u> (PK)	Order_Quantity	<u>Product_ID</u> (FK)
----------------------	----------------	------------------------

<u>Product_ID</u> (PK)	Product_Details	Pro_warehouse_no	Warehouse Details
------------------------	-----------------	------------------	-------------------



Third Normal Form (3NF)

- A relation is in the 3NF if it is in 2NF and every determinant in the relation is a candidate key for that relation; and no *transitive dependencies* exist.
 - **Transitive dependency:** If attribute A determines attribute B, attribute B determines attribute C, then attribute C is transitively dependent on attribute A.



3NF Analysis

Any Transitive Dependency??

<u>Customer_ID</u> (K)	Customer_Details	Town ID	Town Name
-------------------------------	------------------	---------	-----------

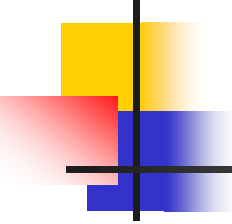
C -> T -> TN

<u>Customer_ID</u> (K)	<u>Order_ID</u> (K)	Order_Date
-------------------------------	----------------------------	------------

<u>Order_ID</u> (K)	Order_Quantity	<u>Product_ID</u> (K)
----------------------------	----------------	------------------------------

<u>Product_ID</u> (PK)	Product_Details	Pro_warehouse_no	Warehouse Details
-------------------------------	-----------------	------------------	-------------------

PID->PWN->WD



3NF - Solution

<u>Customer_ID (K)</u>	Customer_Details	Town ID
-------------------------------	------------------	---------

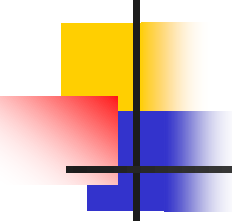
Town ID	Town Name
---------	-----------

<u>Customer_ID (K)</u>	<u>Order_ID (K)</u>	Order_Date
-------------------------------	----------------------------	------------

<u>Order_ID(K)</u>	Order_Quantity	<u>Product_ID (K)</u>
--------------------	----------------	-----------------------

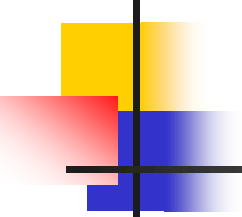
Product_ID (PK)	Product_Details	Pro_warehouse_no
-----------------	-----------------	------------------

Warehouse_No	WarehouseName
--------------	---------------



From 2NF to 3NF

- Removing transitive dependencies by placing the attribute(s) in a new table and set the determinant attribute as the primary key of the new table.



Three Stages of basic Normalisation

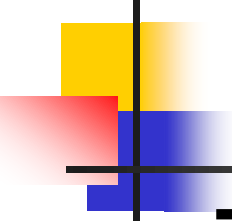
First Normal Form (1NF):

Ensure that all the attributes are atomic (i.e. in the smallest possible components). There is only one possible value for each domain and not a set of values. Repeating groups are removed.

Second Normal Form (2NF)

Ensure that all non-key attributes are functionally dependent (give facts about) all of the key. If this is not the case, split off into a separate relation those attributes that are dependent on only part of the key.

Three Stages of Normalisation

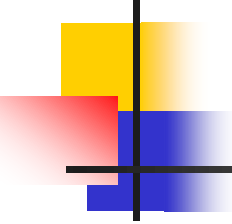


(ii)

Third Normal Form (3NF) and BCNF

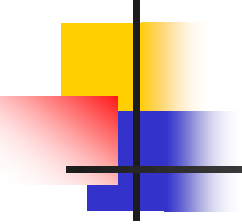
- Ensure that all the determinants are key attributes. If this is not the case, create new relations to realize it.

It is this level of normalisation that is usually used as the basis for the design of the data model, as an end result of data analysis, and for mapping onto a database.



A Memorable Definition of *Normalisation*

- The attributes in a relation must depend on the key, the whole key, and nothing but the key.
 - Keep this in mind as normalisation is developed.



Reading

- Colin Ritchie (2002), Relational Database Principle, 2nd Edition, Thomas Learning.
- Britton C and Boake J (2002), Software System Development – A gentle introduction, 3rd ed.